A HYBRID OPTIMIZATION ALGORITHM BASED ON GENETIC ALGORITHM AND ANT COLONY OPTIMIZATION

Zainudin Zukhri[1] and Irving Vitra Paputungan[1]

[1]Faculty of Industrial Technology, Islamic University of Indonesia,
Jl. Kaliurang KM 14.4 Yogyakarta Indonesia

ABSTRACT

In optimization problem, Genetic Algorithm (GA) and Ant Colony Optimization Algorithm (ACO) have been known as good alternative techniques. GA is designed by adopting the natural evolution process, while ACO is inspired by the foraging behaviour of ant species. This paper presents a hybrid GA-ACO for Travelling Salesman Problem (TSP), called Genetic Ant Colony Optimization (GACO). In this method, GA will observe and preserve the fittest ant in each cycle in every generation and only unvisited cities will be assessed by ACO. From experimental result, GACO performance is significantly improved and its time complexity is fairly equal compared to the GA and ACO.

KEYWORDS

Ant, Ant Colony Optimization, Genetic Algorithm, Hybridization, TSP.

1. INTRODUCTION

Genetic Algorithm (GA) is one of the powerful optimization methods based on the process of natural evolution[1]. The survival of the fittest idea is adopted to provide a different searching technique which explores selected possible solution to obtain good result. Due to the performance problem in GA during the searching, several works have been done aiming at it by, such as, the development of the selection mechanism strategy[2], adaptive mutation probability and GA operator improvement[3], and elitism selection mechanism enhancement through successive generations using threshold values[4]. In this paper, the same goal is proposed by incorporating GA with Ant Colony Optimization (ACO) as one of bio inspired approaches to solve Travelling Salesman Problem (TSP). The algorithm will seek the shortest path from the nest to the food source by maintaining the fittest ant. ACO is selected because it has good performance on solving TSP. However, ACO lacks of capability of melioration when searching is progressing[5]. Hence, the performance of both algorithms, GA and ACO, in solving TSP is aimed.

GA and ACO are enlightened by biologic evolutionism[6]. In GA, the solution in each generation is evolved and improved as the chromosome performing fitness enhancement. Meanwhile in ACO, the TSP solution is improving from the process of seeking the optimal path between the nest and the source of food. Thus, both techniques are combined in order to achieve the best solution using not only the ant colony instinct to find a path, but also equipped by individual capability having good fitness.

DOI : 10.5121/ijaia.2013.4505

The structure of this paper is as follows: a review on how important to solve TSP and techniques that are used are presented in the following section. Section 3 describes the proposed hybrid model. In section 4 the result and the discussion are shown. Finally, the conclusion is provided at the last section.

2. SOLVING TRAVELING SALESMAN PROBLEM

Travelling Salesman Problem (TSP) is a Non-deterministic Polynomial-time (NP-Hard) problem in combinatorial optimization area [7] and common among the researchers. TSP is able to represent many practical cases to validate a new developed algorithm and literally understandable[7][8][9]. However in the other side, NP-Hard is also characterized by its unrealistic computational time in problem solving. In Given a list of cities and their pairwise distances, TSP will then find a shortest possible tour that visits each city exactly once and return to the first visit. There are two types of TSP applications, either implemented directly or adopted as framework, such as transportation problem, logistic distribution problem, delivery order problem, minimum spanning tree (MST) for communication network, electricity network or water pipelining, and machine flow shop scheduling[10]. Some different types of TSP are Symmetric, Asymmetric, and Hamiltonian[11]

Many known approaches have been applied to resolve TSP, such as heuristic method, operational research, and other natural inspired algorithms. Among them, GA and ACO are the most interesting methods for researchers. Both have pros and cons that look complementary to each other. That is the main motivation to combine GA and ACO. The next section is the review on both methods used and some related works on those combinations.

2.1. Genetic Algorithm

Genetic Algorithm (GA) is a computational method designed to simulate the evolution processes and natural selection in organism[12], which follows the sequence as generating the initial population, evaluation, selection, crossover, mutation, and regeneration, see Figure 1[13]. The initial population becomes important as this represents the solution and it is normally randomly generated. Using a problem specific function, the population is then evaluated. GA will select some of them, based on certain probability, that will mate in the next process. Crossover and Mutation will be performed to them to get a new and better ones. The idea of GA is that the new generation of solution should be better than the previous one. This process is repeated until some stopping criteria are reached.

GA is vastly applicable in many optimization problems as this method does not need a good initial knowledge of the problem solved in the nature; it is considered the pros of GA. In addition, this algorithm is possible to find the global optimum and well adapted to the problem[14]. On the other hand, GA is easy to fall into premature convergence that makes the best solution difficult to be achieved; and GA also needs longer processing time for problem with large data are considered as its disadvantages[14][15]

2.2. Ant Colony Optimization

Ant Colony Optimization (ACO) is a computational method that is inspired from the way of ant colony seeking the shortest path from the food resource to the nest without visual aid[17]. In their searching, ants deposit a certain amount of pheromone while walking to form a line and communicate with other ants. Those that could not smell the pheromone, they keep travelling at random route. The pheromones of certain path is enhancing when more ants are attractively tracking on it to obtain the shortest one. The ACO flow is depicted in Figure 2[18]. The ant

algorithm is started by spreading out the ants randomly to every city to be set as the initial city for the respective ant. Such ant will select the next city based on certain probability as in equation (2). This probability is a function of pheromone matrix, distance matrix, and parameters. This selection is repeated until each ant visited every city one time. That is the first cycle in the algorithm, and the cycle is carried on until reaching the stopping criteria. In each cycle the overall route is changing methodically as the pheromone matrix is updated.

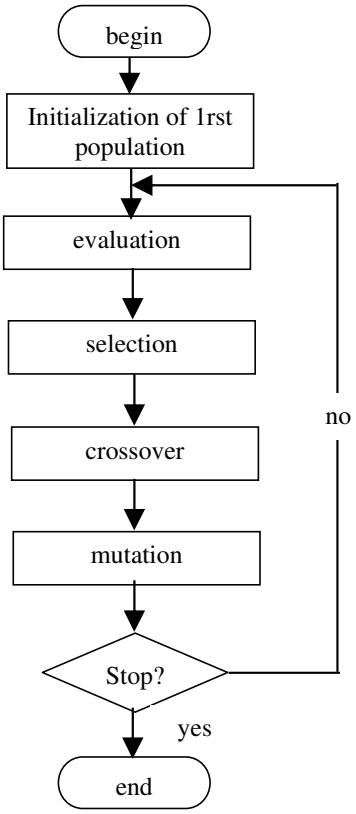


Figure 1. Flowchart of basic GA [13]

ACO is dedicated to solve TSP[8][15][16]. Nonetheless, this algorithm has several weaknesses, such as its performance extremely depends on previous cycle, easy to convergent and stagnant, and need a long time processing time. This fact causes challenge, the searching space and computation time of ACO[16].

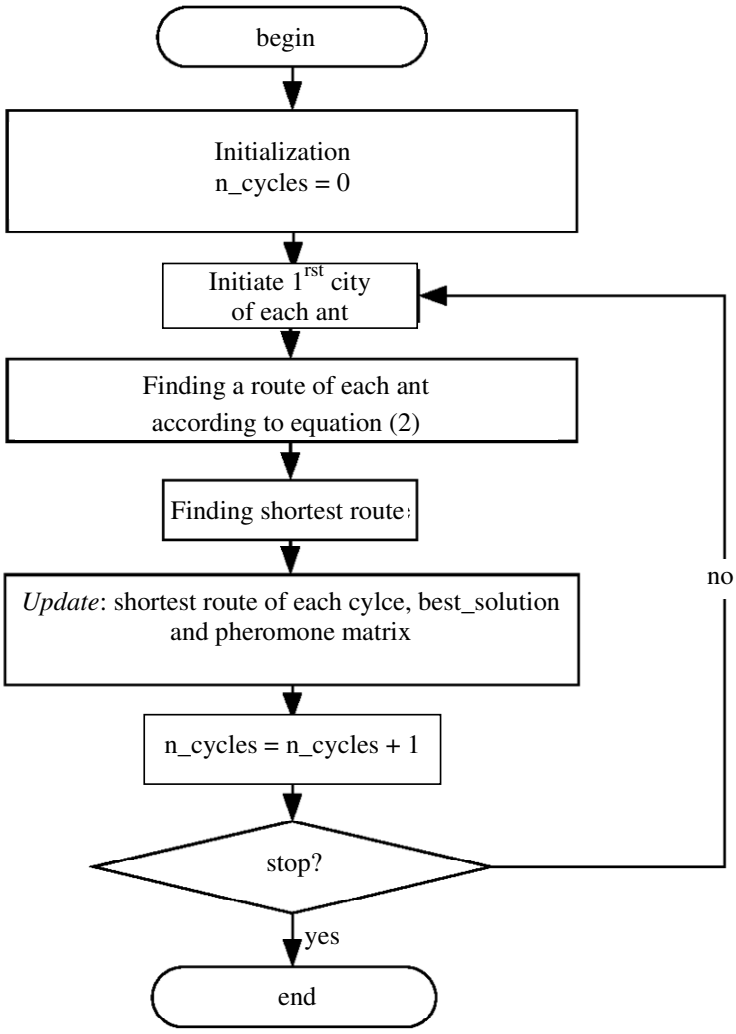


Figure 2. Flowchart of basic ACO [18]

2.3. Hybridization of Genetic Algorithm Ant Colony Optimization

There are several works based on the hybridization of GA and ACO. Shang et. al. proposed to hybrid GA and ACO as a new algorithm to solve TSP[19]. GA, in such work, is benefited to initiate the matrix of pheromone in ACO and to recombine the route from ACO. The author claimed the hybrid algorithm is more effective compare to the GA and ACO. Duan and Yu suggested the use of memetic algorithm to find the parameters combination in ACO[20]. The author claimed this hybridization will simplify in selecting the adjustable parameter in ACO where human experience is needed and in most cases depending on coincidence. Jin-Rong et al. developed two sub-algorithms based on GA and ACO hybridization that is covering up the weaknesses both algorithms[21]. ACO is employed to help GA to eliminate the appearance of invalid tour, while GA is used to overcome the dependency on the matrix of pheromone in ACO. Al-Salami developed a combination of GA and ACO by incorporating each basic method as a sub-solution generator and followed by selecting the better sub-solution as a new population in the next iteration[23]. In this work ACO is utilized as a temporary solution generator that will be improved by implementing GA operators iteratively until stopping criteria is reached. Takahashi

proposed GA and ACO hybridization that is slightly more efficient than Al-Salami work by extending crossover operator in GA (EXO) to produce better solution generated by ACO[23]. There are two steps to get the solution. First is to use ACO to generate the solution that reached local optima. The solution creation in this step is repeated iteratively and independent to each other. Such solution is then treated as a population in GA or in other word it becomes the chromosome that will be recombined in GA to acquire better global optima. With almost similar way, Dong et. al. improved the previous hybridization work by inserting GA as a sub-process in each iteration in ACO[24]. By this proposed work, the probability to tweak the route resulted in ACO becomes high. However, in the aforementioned works, they did not consider the dependency of cycle to the previous one to prevent premature convergent in ACO. In this paper, the chromosomes of GA are utilised to optimize the number of next city visited from each ant, in order to avoid dependency on previous cycle. Of these chromosomes, the diversity of the tour of ants can be preserved. The next section presents the detail of the proposed work.

3. PROPOSED HYBRIDIZATION METHOD

3.1. Hybridization technique

The main idea of this paper is to define an appropriate approach to hybridize GA and ACO to find TSP solution by which constructs the concept combining certain steps in GA and ACO to perform GACO. Hybridization is also applied to some parameters and variables of GA or ACO that share same characteristics in the computation, i.e. population size in GA and number of ants in ACO, number of generations in GA and number of cycles in ACO, and chromosome in GA and Tabu list in ACO. The proposed technique in this paper is introducing the evolution steps of GA into the computation step of ACO as shown in the Figure 3 and Table 1. The modification is indicated in as shaded drawing.

3.2. Chromosomal representation on GACO

In the experiment, the performance of the proposed hybrid method is compared to the basic methods, both GA and ACO. In that case, a compatible chromosomal representation must be well designed to be used by GA and GACO. Binary representation is therefore chosen reasonably. These representations guarantee such solutions, which are obtained by classical operators, are valid. There is no need to define special operators for these representations. To solve TSP of n cities, the chromosomal representation is designed as followed:

1. Each chromosome consists of (n-1) groups of gene.
2. The number of genes in i^{th} group is equal to (n-i) bit, so that the total number of genes (N_{gen}) in the chromosome follows equation (1).

$$N_{gen} = \sum_{i=1}^{n-1} i \tag{1}$$

The chromosomal representation is shown in Figure 4.

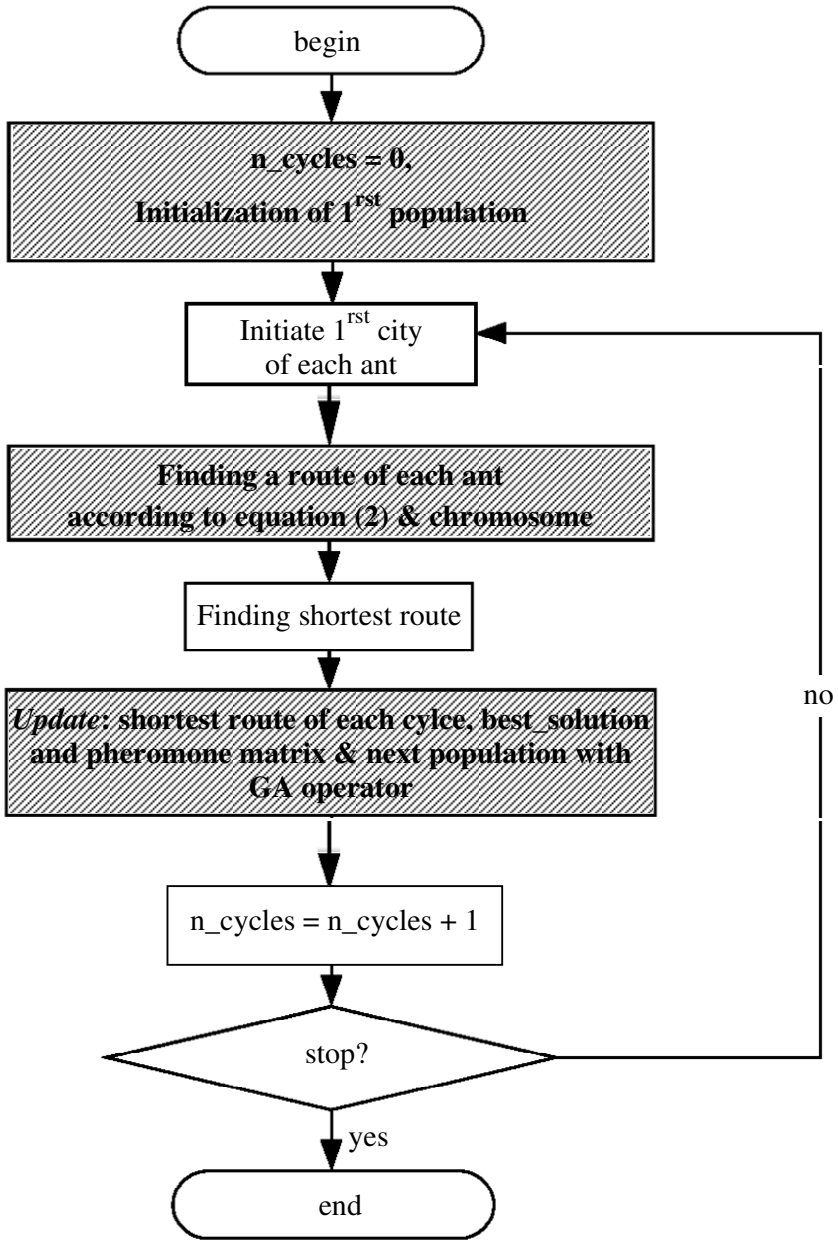


Figure 3. Flowchart of GACO

$b_{1,1}$	$b_{1,2}$	..	$b_{1,n-1}$	$b_{2,1}$	$b_{2,2}$	..	$b_{2,n-2}$	..	..	$b_{n-3,1}$	$b_{n-3,2}$	$b_{n-3,3}$	$b_{n-2,1}$	$b_{n-2,2}$	$b_{n-1,1}$
1				2						n-3			n-2		n-1

Figure 4. Chromosomal representation of *n* cities

Table 1. Comparison of details in GA, ACO and GACO.

ACO	GA	GACO
Hybridization on Parameters and variables of the algorithms		
Number of cycles	Number of generations	Number of cycles
Number of ants	Population size	Number of ants; it is equal to the population size
Tabu list	Chromosome	Tabu list and Chromosome. Tabu list is the representation of solution. Chromosome is indirect representation of solution. It represents the remaining city to be visited by the ant based on equation (2)
Tabu list and chromosome are the representations of solution		
Hybridization on steps of the algorithms		
Initialization: Pheromone	Initialization: first population	Initialization: Pheromone and first population
Initiate first city of each ant		Initiate first city of each ant
Finding a route of each ant according to equation (2)		Finding a route of each ant according to equation (2) & chromosome
Finding shortest route		Finding shortest route
	Evaluation	Evaluation process is hybridized on finding a route
Updating pheromone matrix and shortes route of each cycle	Updating population	Updating population, pheromone matrix and shortes route of each cycle

The number of bit 1 is used in i^{th} group of gene to determine the cities that is possible to be visited by the ants. The determining process follows:

1. Indexes of unvisited cities are defined as a set called *allowed*.

2. The first visited city is determined by the number of gene 1 in the first groups of gene directly. If the number of gene 1 is *c*, then the first visited city is c^{th} index in *allowed*. If there is no gene 1 in the first group, then the first visited city is the last index in *allowed*.

3. For the other cities, the number of gene(s) 1 is used in other group of gene and equation (2) to determine the cities that will be visited one by one. If there is no gene 1 in the i^{th} group of gene, then the equation (2) is utilized as the basic ACO, all indexes in *allowed* have a chance to be visited according to its probabilities. If there is a single gene 1, the visited city is the city that the biggest probabilities as consequence. And if the number of gene 1 is c, then only the first-*c* indexes in *allowed* have a chance to be visited. Equation (2) is a formula by Dorigo et al [17] to find the next city to be visited.

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{k \in allowed_k} [\tau_{ik}(t)]^{\alpha} \cdot [\eta_{ik}]^{\beta}} & \text{if } j \in allowed_k \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where α and β are parameters that control the relative importance of trail versus visibility, k is index of ant, $allowed_k$ is the set of unvisited city that probably will be visited by k^{th} ant, i is index of current last visited city, j is index of next city that visited from i^{th} city, t is index of time, τ is intensity of trail, and η is visibility.

4. Once a city has been set to be visited, index of this city is removed from *allowed*.

5. The last index remaining in *allowed* is the last city visited.

In basic GA, the chromosome represents the solution of TSP indirectly. Below is the process to decode the chromosome to the solution of TSP:

1. Indexes of unvisited cities are defined as a set called *allowed*.
2. The i^{th} visited city is determined by the number of gene 1 in the i^{th} groups of gene. If the number of gene 1 is k, then the i^{th} visited city is k^{th} index in *allowed*. If there is no gene 1 in the i^{th} group, then the i^{th} visited city is the last index in *allowed*.
3. Once a city has been set visited, index of this city is eliminated from *allowed*.
4. The last city is the last index remaining in *allowed*.

4. RESULTS AND DISCUSSION

4.1. Experimental parameters

Simulation software is developed to evaluate the proposed method. The simulation runs on Intel-Cores i7-2620M 2.70 GHz. Following how hybrid technique is evaluated by Takahashi[22] and Lee[25], result given by the simulator is then compared to GA, ACO, and combination of both. GA by De Jong (In [26] and ACO by Dorigo et al [17] are chosen. Some arbitrary data are taken from Library of TSP (TSPLIB[1]) to be used in the experiment. The data consist of 13, 16, 22, 25 and 35 cities respectively. Each computational time resulted from all methods is assessed and compare the TSP solutions. Measurements used in this testing are time unit and length unit.
The simulation is performed 10 times with various recommended parameters. Depending on the number of cities (n), the number of ants (m) or the populations size (*popsize*), and number of iterations are adjusted. The best experimental result for each method is obtained by using the parameters as follows:

1. For basic GA, the crossover probability, P_c is 0.50, and mutation probability, P_m is 0.05. In addition, the crossover method used is uniform random crossover.
2. For basic ACO, α is 1, β is 2, ρ is 0.5, τ is 0.1, and a constant of the trail quantity, Q is 1.
3. For GACO, P_c is 0.70, P_m is 0.05, α is 1, β is 2, ρ is 0.5, τ is 0.1, and Q is 100, except for the data consist of 25 cities, P_c is 0.90, P_m is 0.1, and ρ is 0.9. In addition, the crossover method used is also uniform random crossover.

[1] http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/tsp/

For the 5 variations of data, we set the number of ants or the population size 10, 15, 20, 25, 50 respectively, and the number of iterations 100, 150, 200, 250, 500 respectively.

Our simulation shows that this hybrid method is relatively outperformed than its original methods, especially where the cities more than 20. It is based on the best solution or the average solution of each method. Only in the experiment with 16 cities, GACO was slightly worse than ACO. Running time of GACO is slightly longer than the basic methods. GA is the fastest method, but the GA's solution is the worst. The summary of the simulation is presented in Table 2, and the comparison of the best simulation results is presented in Figure 5. In every chart, we differentiate the lines by putting different label in each line.

Table 2. The summary of the simulation.

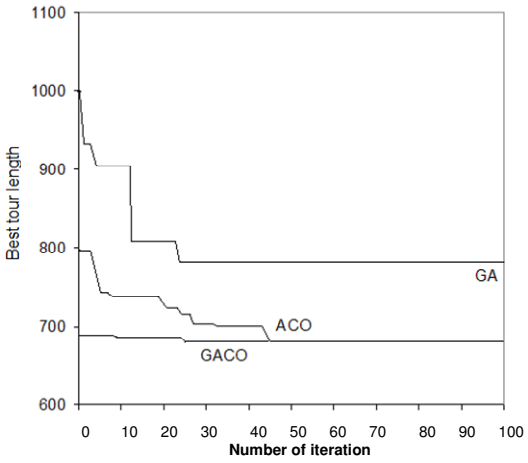
No	Data	Number of Cities	GA			ACO			GACO		
			Time (s)	Solution		Time (s)	Solution		Time (s)	Solution	
				Best	Average		Best	Average		Best	Average
1	Random13	13	1.0	778,92	867,15	1.0	679,50	683,00	1.5	679,50	686,43
2	ulysses16	16	2.0	8765,38	9603,45	2.0	7399,98	7447,15	2.0	7445,15	7499,86
3	ulysses22	22	2.5	10351,66	10743,26	3.0	7604,30	7639,82	3.0	7596,92	7634,43
4	Random25	25	3.0	1471,78	1563,31	3.0	803,21	818,66	4.0	802,97	815,10
5	Random35	35	8.0	1278,64	1327,85	16.0	499,94	499,98	24.0	498,84	499,61

4.2. Result analysis

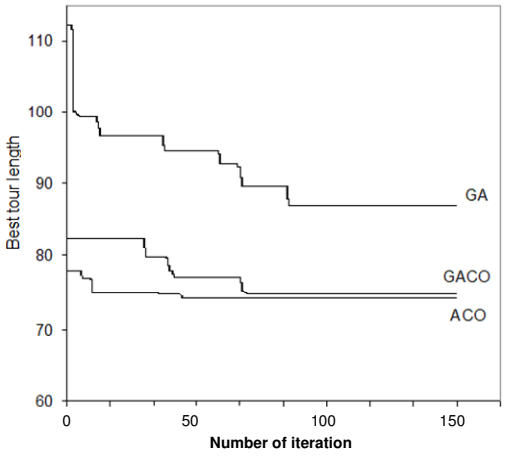
The experiments show that GA is the worst approach after all. Since GA can fail in any cases [27], it is definitely unexpected finding, and the propose method absolutely improves the performance of GA. In the other side, in small amount of data, experiments show that ACO did not just give the same solution as GACO, but it also gave better solution. It does not mean that ACO will constantly perform better than GACO because small data merely reflects simple problem. That advance method is not effective to be applied for simple problem, it is a common sense.

In the experiment with the large data, the significance of the propose method is very clear. GACO is not only better on its best solution but it is also better on its average solution. For example we can compare the best solution for random35. In this case the best solution of GACO is 498.84 and the solution of ACO is 499.94. It is very small difference solution, but it is significant as shown in Figure 6. In this case, the hybridization can improve the quality of the solution found by ACO, i.e. change edge 5-22 into edge 5-8, and edge 8-34 into 22-34.

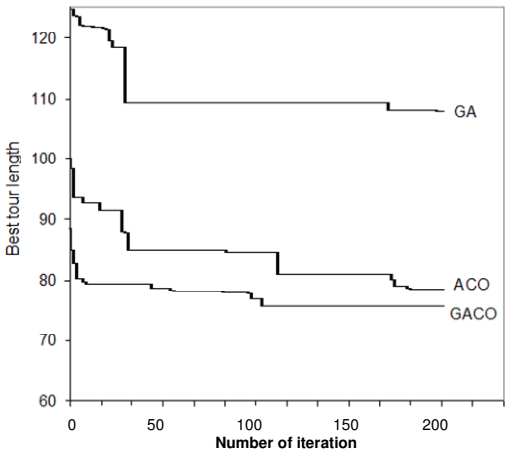
In the larger amount of data, experiments show that GACO can produce better solutions than the basics method consistently. It is because in the proposed method, the chromosome represents the TSP solution indirectly. This chromosome is the initial step to guide the ants in the better solution construction. The better solution resulted from the diversity of the tour is more than it generated by the basic method. Since GA element of GACO can preserve allowed cities that different from allowed cities of ACO, so this hybrid method can avoid the dependency on prior cycle that commonly occurs in ACO. The insertion of evolutionary element into ACO has been succeeded improving the performance of searching process. In other word, the better ant colony can feasibly choose the most considerable cities.



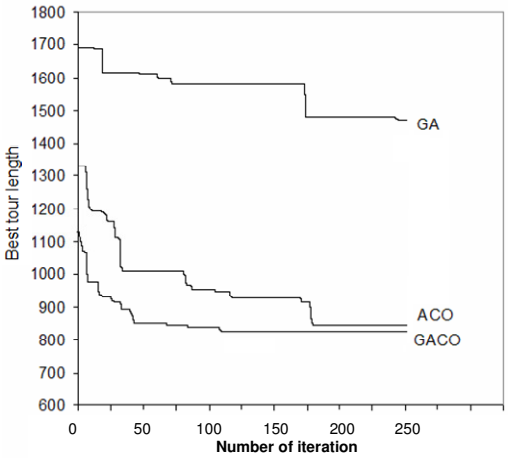
(a) Random13



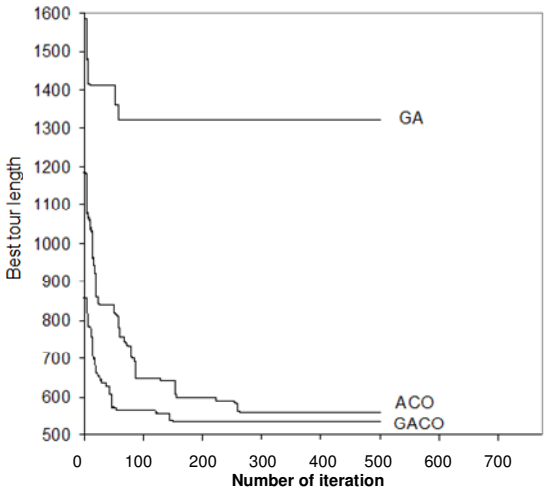
(b) Ulysses16



(c) Ulysses22



(d) Random25

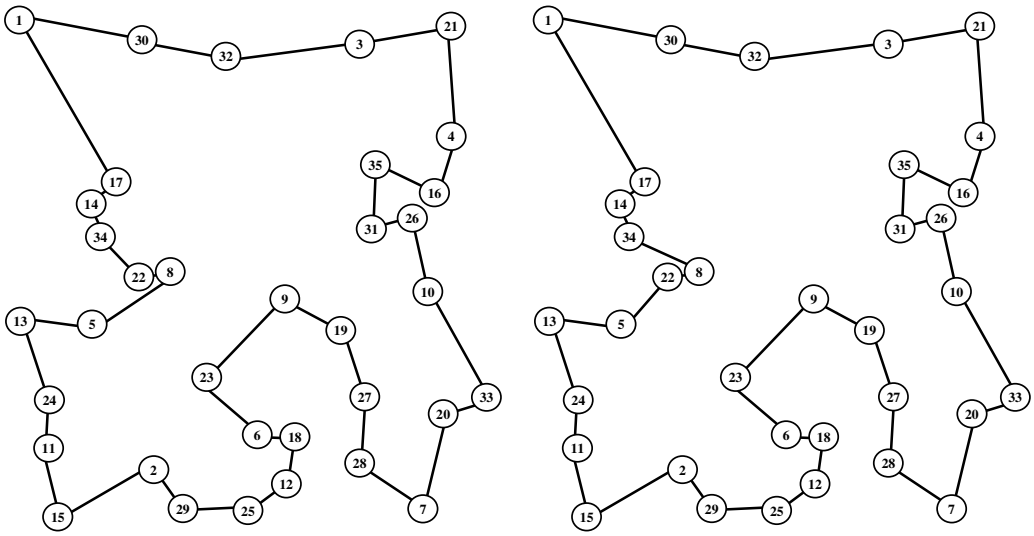


(e) Random35

Figure 5. Comparison of the best solution in the simulation

All in all, the evolution process to make ant colony becomes stronger than the starting, as stated in the initial assumption, has been experimentally proven. The fitter ant colony would have a higher performance than the ordinary colony. In the computational context, we conclude that updating the pheromone matrix and applying the genetic operators will improve the searching process.

The hybrid algorithm has more parameters as every basic algorithm brings their parameters into it. Indeed, we have designed some specific pair of parameters can be fused into some parameters, in fact the fusion of these parameters still produce a couple more parameters. Hence, we think that the experiment to evaluate the proposed method with other different parameter combination is not needed and we assume that the simplification of hybrid parameters is another challenge for future research.



(a) GACO's best solution for random35 (b) ACO's best solution for random35

Figure 6. The small difference solution procuded by ACO and GACO on large data

5. CONCLUSION

In this paper, a hybrid optimization algorithm based on GA and ACO to solve TSP is proposed and is then evaluated with some data, both random data and sample data from the library of TSP. Evolutional process of the GA together with instinct of ant colony in finding the shortest route to seek food are fully combined and formulated as new optimization method called GACO. Experimental studies demonstrate that with small amount of data, it shows insignificancy. But on the big data, it can improve the performance over both GA and ACO. In this case, the solution of the proposed hybridization method has been significantly improved. However, since this work only focused on the how to combine GA and ACO procedurally, some parameters of both sides should be set optimally to get better result and performance in the future.

ACKNOWLEDGEMENTS

This work is fully funded by fundamental research grant Directorate General of Higher Education, Ministry of National Education, Republic of Indonesia No. 229/SP2H/PL/Dit.Litabmas/IV/2011.

REFERENCES

[1] Bagheri, A., Akbarzadeh, T. M., Saraee, M., (2008), "Finding Shortest Path with Learning Algorithms", Int. J. of Artificial Intelligence, Vol. 1, No. A8, pp86-95.

[2] Lu, J., Fang, N., Shao, D., Liu, C., (2007), An Improved Immune-Genetic Algorithm for the Traveling Salesman Problem, in proc. IEEE Int. Conf. Natural Computation.

[3] Serpell, M., Smith, J. E., (2010), "Self-Adaptation of Mutation Operator and Probability for Permutation Representations in Genetic Algorithms", J. Evolutionary Computation, Vol 18, No. 3, pp491–514.

[4] Khanbary, L. M. O., Vidyarthi, D. P., (2009), "Modified Genetic Algorithm with Threshold Selection", Int. J. of Artificial Intelligence, Vol. 2, No. S09, pp14-26.

[5] Hung, K. S., Su, S. S., Lee, Z. J., (2007), "Improving Ant Colony Optimization Algorithms for Solving Traveling Salesman Problems", J. Advanced Computational Intelligence and Intelligent Informatics, Vol. 11, No. 4, pp433-442.

[6] Li, K., Kang, L., Zhang, W., Li, B., (2008), Comparative Analysis of Genetic Algorithm and Ant Colony Algorithm on Solving Traveling Salesman Problem, in IEEE Int. Workshop. Semantic Computing and Systems.

[7] Jingui, L., Ning, F., Dinghong, S., Congyan, L., (2007), An Improved Immune-Genetic Algorithm for the TSP, in Proc. IEEE Int. Conf. Natural Computation.

[8] Wang, S., Zhao, A., (2009), An Improved Hybrid Genetic Algorithm for Traveling Salesman Problem, in IEEE Proc. Int. Conf. Computational Intelligence and Software Engineering.

[9] Rodriguez, M. A. V., Gutierrez-Gil, R., Avila-Roman, J. M., Sanchez-Perez, J. M., Gomez-Pulido, J.A., (2005), Genetic Algorithms Using Paralelism and FPGAs: The TSP as Case Study, in IEEE Proc. Int. Conf. Parallel Processing Workshops.

[10] Corwin, B. D., Esogbue, A. O., (2006), "Two Machine flow shop scheduling problems with sequence dependent setup times: a dynamic programming approach", J. Naval Research logistics.

[11] Li-Ying, W., Jie, Z., Hua, L., (2007), An Improved Genetic Algorithm for TSP, in IEEE Proc. Int. Conf. Machine Learning and Cybernetics.

[12] Raza, Z., Vidyarthi, D. P., (2009), "A computional grid scheduling model to minimize turnaround using modified GA", Int. J. of Artificial Intelligence, Vol. 3, No. A9, pp86-106.

[13] Banga, V. K., Singh, Y., Kumar, R., (2007), "Simulation of Robotic Arm using Genetic Algorithm & AHP", J. Engineering and Technology, Vol. 25, pp95-101.

[14] Qing-Bao, Z., Shuyan, C., (2007), A new ant evolution algorithm to resolve TSP problem, in IEEE Proc. Int. Conf. Machine Learning and Applications.

[15] Yingzi, W., Yulan, H., Kanfeng, G., (2007), Parallel Search Strategies for TSP's using a Greedy Genetic Algorithm, in IEEE Proc. Int. Conf. Natural Computation.

[16] Zhenchao, W., Haibin, D., Xiangyin, Z., (2009), An Improved Greedy Genetic Algorithm for Solving TSP, in IEEE Proc. Int. Conf. Natural Computation.

[17] Dorigo, M., Maniezzo, V., Colorni, A., (1996), "The Ant System: Optimization by a colony of cooperating agents", IEEE J. Systems, Man, and Cybernetics, Vol. 26, No. 1, pp1-13.

[18] Zhou, R., Lee, H. P., Nee, A. Y. D., (2008), "Applying Ant Colony Optimization Algorithm to Dynamic Job Shop Scheduling Problems", Int. J. Manufacturing Research, Vol. 3, No. 3, pp301-320.

[19] Shang, G., Xinzi, J., Kezong, T., (2007), Hybrid Algorithm Combining Ant Colony Optimization Algorithm with Genetic Algorithm, in Proc. of the 26th Chinese Control Conf.

[20] Duan, H., Yu, X., (2007), Hybrid Ant Colony Optimization Using Memetic Algorithm for Traveling Salesman Problem, in Proc. IEEE Int. Symposium. Dynamic Programming and Reinforcement Learning.

[21] Jin-rong, X., Yun, L., Hai-tao, L., Pan, L., (2008), "Hybrid Genetic Ant Colony Algorithm for Traveling Salesman Problem", J. Computer Applications, Vol. 28, No. 8, pp2084-2112.

[22] Takahashi, R., (2009), A Hybrid Method of Genetic Algorithms and Ant Colony Optimization to Solve the Traveling Salesman Problem, in Proc. Int. Conf. Machine Learning and Applications.

[23] Al-Salami, N. M. A., (2009), "Ant Colony Optimization Algorithm", J. UbiCC, Vol. 4, No. 3, pp823-826.

[24] Dong, G., Guo, W. W., Tickle, K., (2012), "Solving The Traveling Salesman Problem Using Cooperative Genetic Ant Systems", J. Expert Systems with Applications, Vol. 39, pp5006-5011.

[25] Lee, Z. J., Su, S. F., Chuang, C. C., Liu, K. H., (2008), "Genetic Algorithm with Ant Colony Optimization (GA-ACO) for Multiple Sequence Alignment", J Applied Soft Computing, Vol. 8, pp55-78.
[26] Hopgood, A. A., (2001). Intelligent Systems for Engineers and Scientists. Boca Raton: CRC Press LLC.
[27] Whitley, D., Lunacek, M., Knight, J., (2004). Ruffled by Ridges: How Evolutionary Algorithms can fail. Lecture Notes in Computer Science, The Genetic and Evolutionary Computation Conf.

Authors

Zainudin Zukhri received his B. Elect. Eng. in Electro Engineering from Gadjah Mada University, Indonesia in 1994. Then he received his Master in Computer Science from Universiti Kebangsaan Malaysia in 2008. His employment experience includes the JICT Company, Jakarta. Since 1996, he has been with Department of Informatics Engineering, Faculty of Industrial Technology, Islamic University of Indonesia. His research interests includes artificial intelligence and data mining.

Irving Vitra Paputungan received his B. at Informatics Engineering Department, Islamic University of Indonesia after in 1999. Then he received his at Computer and Information Science, University Technology Petronas Malaysia in 2008. His research interests includes modelling, cloud computing, artificial intelligence and data mining.