Genetic Algorithm Performance with Different Selection Strategies in Solving TSP

Noraini Mohd Razali, John Geraghty

***Abstract*—A genetic algorithm (GA) has several genetic operators that can be modified to improve the performance of particular implementations. These operators include parent selection, crossover and mutation. Selection is one of the important operations in the GA process. There are several ways for selection. This paper presents the comparison of GA performance in solving travelling salesman problem (TSP) using different parent selection strategy. Several TSP instances were tested and the results show that tournament selection strategy outperformed proportional roulette wheel and rank-based roulette wheel selections, achieving best solution quality with low computing times. Results also reveal that tournament and proportional roulette wheel can be superior to the rank-based roulette wheel selection for smaller problems only and become susceptible to premature convergence as problem size increases.**

***Index Terms*— Genetic algorithm, Selection, Travelling salesman problem, Optimization**

I. INTRODUCTION

Basic genetic algorithm (GA) is generally composed of two processes. The first process is selection of individuals for the production of the next generation and the second process is manipulation of the selected individuals to form the next generation by crossover and mutation techniques. The selection mechanism determines which individuals are chosen for mating (reproduction) and how many offspring each selected individual produces. The main principle of selection strategy is "the better is an individual; the higher is its chance of being parent." Generally, crossover and mutation explore the search space, whereas selection reduces the search area within the population by discarding poor solutions. However, worst individuals should not be discarded and they have some chances to be selected because it may lead to useful genetic material. A good search technique must find a good trade-off between exploration and exploitation in order to find a global optimum [1]. Hence, it is important to find a balance between exploration (i.e. poor solutions must have chance to go to the next generation) and exploitation (i.e. good solutions go to the next generation more frequently than poor solutions) within the mechanism of the selection.

Manuscript received March 6, 2011; revised March 21, 2011. This work was supported by the University Malaysia Pahang (UMP) in collaboration with Dublin City University, Ireland. Noraini Mohd Razali is with School of Mechanical & Manufacturing Engineering, Dublin City University, Ireland (e-mail: norainimbr@ump.edu.my). John Geraghty is with Enterprise Research Process Centre, Dublin City University, Ireland (e-mail: john.geraghty@dcu.ie).

The different selection strategy used in the GA process will significantly affect the performance of the algorithm differently. This study is intended to examine the performance of GA when using different selection strategy specifically in solving the travelling salesman problem (TSP). TSP is a classical example of a NP-hard combinatorial optimization problem. Many production and scheduling problems can be reduced to a simple concept that there is a salesman who must travel from city to city, visiting each city exactly once and returning to the home city [2]. It is possible for the salesman to select the orders of the cities visited so that the total distances travelled in his tour is as small as possible which will apparently save him time and money [2]. Although TSP is conceptually simple, it is difficult to obtain an optimal solution. The main difficulty of this problem is the enormous number of possible tours; $(n-1)!/2$ for symmetric n cities tour. As the number of cities in the problem increases, the numbers of permutations of valid tours are also increase. It is this factorial growth that makes the task of solving the TSP immense even for modest n sized problems.

The remainder of this paper is organized as follows: Section II presents a brief summary of the previous works on selection strategy. Section III contains an overview of the genetic algorithm for TSP, while Section IV describes into more detail on selection strategy that used in the experiments. Section V tests the performance of GA and discusses the experimental results. The conclusions are summarized in Section VI.

II. PREVIOUS WORK ON SELECTION STRATEGY

Several researchers have studied the performance of GA using different selection strategy; yet almost none of them tested on TSP problem. The performance of GA is usually evaluated in terms of convergence rate and the number of generations to reach the optimal solution. Jadaan et al. [3] for example compared the results of GA between proportional roulette wheel and rank-based roulette wheel selection method using several mathematical fitness functions and found that rank-based outperformed proportional in number of generations to come out with the optimal solution. He observed that rank-based is steadier, faster, certainty and more robust towards the optimum solutions than proportional roulette wheel. On the other hand, Zhong et al. [4] compared proportional roulette wheel with tournament selection, with tournament size equal 6 at seven general test functions and concluded algorithm with the tournament selection is more efficient in convergence

ISBN: 978-988-19251-4-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

than proportional roulette wheel selection. Julstrom [5] investigated the computing time efficiency of two types of rank-based selection probabilities; linear ranking and exponential ranking probabilities and compared with tournament selection. He pointed that tournament selection is preferred over rank-based selection because repeated tournament selection is faster than sorting the population to assign rank-based probabilities. In addition, Mashohor et al. [6] evaluated the performance of PCB inspection system using three GA selection method; deterministic, tournament and roulette wheel and discovered that deterministic has the ability to reach the highest maximum fitness with lowest number of generations for all test images. This is then followed by roulette wheel and tournament selection.

Goh et al. [7] in his work entitled sexual selection for genetic algorithms focused on the selection stage of GA and examined common problems and solution methods for such selection schemes. He proposed a new selection scheme called sexual selection and compared the performance with commonly used selection methods in solving the Royal road problem, the open shop scheduling and the job shop scheduling problem. He claimed that the proposed selection scheme performed either on-par or better than roulette wheel selection on average when no fitness scaling is used. The new scheme also performed better on average when compared to tournament selection in the more difficult test cases when no scaling is used. Apart from that, Goldberg and Deb [8] did comprehensive studies on proportional, ranking, tournament and Genitor (steady state) selection schemes on the basis of solutions to differential equations. Their studies have been performed to understand the expected fitness ratio and convergence time. They found that ranking and tournament selection outperformed proportional selection in terms of maintaining steady pressure toward convergence. They further demonstrated that linear ranking selection and stochastic binary tournament selection have identical expectations, but recommended binary tournament selection because of its more efficient time complexity.

III. GENETIC ALGORITHM FOR TSP

This section provides the general overview of the genetic algorithm component and operation for solving TSP. Genetic algorithm is an optimization method that uses a stochastic approach to randomly search for good solutions to a specified problem. These stochastic approaches use various analogies of natural systems to build promising solutions, ensuring greater efficiency than completely random search. The basic principles of GA were first proposed by Holland in 1975 [9]. The GA operation is based on the Darwinian principle of 'survival of the fittest' and it implies that the fitter individuals are more likely to survive and have a greater chance of passing their good genetic features to the next generation. In genetic algorithm, each individual i.e. chromosome that is a member of the population represents a potential solution to the problem. There are a number of possible chromosome representations, due to a vast variety of problem types. The 'path' representation is more natural to represent the chromosome in TSP [10]. The TSP consists a number of cities, where each pair of cities has a corresponding distance. The aim is to visit all the cities such that the total distance travelled will be minimized. Obviously, a solution, and therefore a chromosome which represents that solution to the TSP, can be given as an order, that is, a path, of the cities.

The procedure for solving TSP can be viewed as a process flow given in Fig. 1. The GA process starts by supplying important information such as location of the city, maximum number of generations, population size, probability of crossover and probability of mutation. An initial random population of chromosomes is generated and the fitness of each chromosome is evaluated. The population is then transformed into a new population (the next "generation") using three genetic operators: selection, crossover and mutation. The selection operator is used to choose two parents from the current generation in order to procreate a new child by crossover and/or mutation. The new generation contains a higher proportion of the characteristics possessed by the 'good' members of the previous generation and in this way good characteristics are spread over the population and mixed with other good characteristics. After each generation, a new set of chromosomes where the size is equal to the initial population size is evolved. This transformation process from one generation to the next continues until the population converges to the optimal solution, which usually occurs when a certain percentage of the population (e.g. 90%) has the same optimal chromosome in which the best individual is taken as the optimal solution.

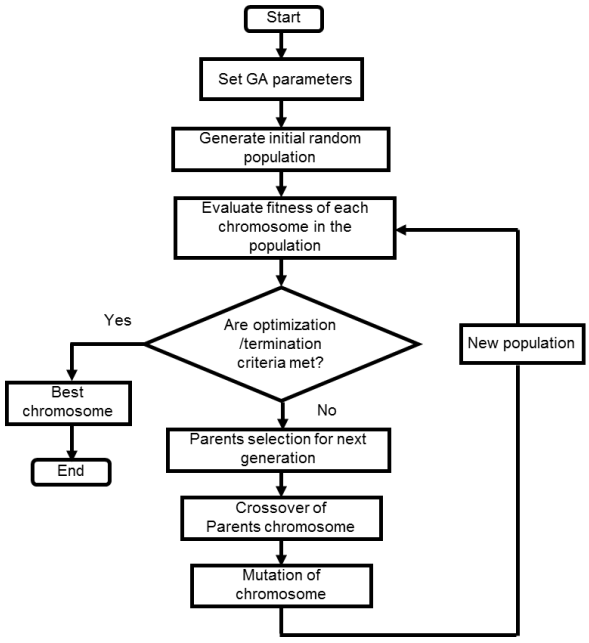


Fig. 1. Genetic algorithm procedure for TSP

IV. SELECTION STRATEGY FOR REPRODUCTION

The selection strategy addresses on which of the chromosomes in the current generation will be used to reproduce offspring in hopes that next generation will have even higher fitness. The selection operator is carefully formulated to ensure that better members of the population

ISBN: 978-988-19251-4-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

(with higher fitness) have a greater probability of being selected for mating or mutate, but that worse members of the population still have a small probability of being selected, and this is important to ensure that the search process is global and does not simply converge to the nearest local optimum. Different selection strategies have different methods of calculating selection probability. The differing selection techniques all develop solutions based on the principle of survival of the fittest. Fitter solutions are more likely to reproduce and pass on their genetic material to the next generation in the form of their offspring. There are three major types of selection schemes will be discussed and experimented in this study; tournament selection, roulette wheel, and rank-based roulette wheel selection. The subsequent section will describe the mechanism of each strategy. A more detailed of selection method can be found in [8, 11, 12, 13].

A. *Tournament Selection*

Tournament selection is probably the most popular selection method in genetic algorithm due to its efficiency and simple implementation [8]. In tournament selection, *n* individuals are selected randomly from the larger population, and the selected individuals compete against each other. The individual with the highest fitness wins and will be included as one of the next generation population. The number of individuals competing in each tournament is referred to as tournament size, commonly set to 2 (also called binary tournament). Tournament selection also gives a chance to all individuals to be selected and thus it preserves diversity, although keeping diversity may degrade the convergence speed. Fig. 2 illustrates the mechanism of tournament selection while Fig. 3 shows the procedure for tournament selection. The tournament selection has several advantages which include efficient time complexity, especially if implemented in parallel, low susceptibility to takeover by dominant individuals, and no requirement for fitness scaling or sorting [8, 12].

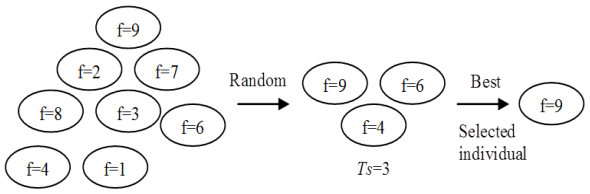


Fig. 2. Selection strategy with tournament mechanism

In the above example, the tournament size, *Ts* is set to three, which mean that three chromosomes competing each other. Only the best chromosome among them is selected to reproduce. In tournament selection, larger values of tournament size lead to higher expected loss of diversity [12, 14]. The larger tournament size means that a smaller portion of the population actually contributes to genetic diversity, making the search increasingly greedy in nature. There might be two factors that lead to the loss of diversity in regular tournament selection; some individuals might not get sampled to participate in a tournament at all while other individuals might not be selected for the intermediate population because they lost a tournament.

```
Procedure: Roulette wheel selection
  While population size < pop_size do
        Generate pop_size random number r
        Calculate cumulative fitness, total fitness ( p_i ) and sum of proportional fitness (Sum)
        Spin the wheel pop_size times
        If Sum < r then
          Select the first chromosome, otherwise, select jth chromosome
        End If
  End While
  Return chromosomes with fitness value proportional to the size of selected wheel section
End Procedure
```

Fig. 3. Procedure for tournament selection

B. *Proportional Roulette Wheel Selection*

In proportional roulette wheel, individuals are selected with a probability that is directly proportional to their fitness values i.e. an individual's selection corresponds to a portion of a roulette wheel. The probabilities of selecting a parent can be seen as spinning a roulette wheel with the size of the segment for each parent being proportional to its fitness. Obviously, those with the largest fitness (i.e. largest segment sizes) have more probability of being chosen. The fittest individual occupies the largest segment, whereas the least fit have correspondingly smaller segment within the roulette wheel. The circumference of the roulette wheel is the sum of all fitness values of the individuals. The proportional roulette wheel mechanism and the algorithm procedure are depicted in Fig. 4 and Fig. 5 respectively. In Fig. 4, when the wheel is spun, the wheel will finally stop and the pointer attached to it will point on one of the segment, most probably on one of the widest ones. However, all segments have a chance, with a probability that is proportional to its width. By repeating this each time an individual needs to be chosen, the better individuals will be chosen more often than the poorer ones, thus fulfilling the requirements of survival of the fittest. Let f_1, f_2,…, f_n be fitness values of individual 1, 2,…, *n*. Then the selection probability, P_i for individual *i* is define as,

$$p_i = \frac{f_i}{\sum_{j=1}^{n} f_j} \tag{1}$$

The basic advantage of proportional roulette wheel selection is that it discards none of the individuals in the population and gives a chance to all of them to be selected. Therefore, diversity in the population is preserved. However, proportional roulette wheel selection has few major deficiencies. Outstanding individuals will introduce a bias in the beginning of the search that may cause a premature convergence and a loss of diversity. For example, if an initial population contains one or two very fit but not the best individuals and the rest of the population are not good, then these fit individuals will quickly dominate the whole population and prevent the population from exploring other potentially better individuals. Such a strong domination causes a very high loss of genetic diversity which is definitely not advantageous for the optimization process. On the other hand, if individuals in a population have very similar fitness values, it will be very difficult for the population to move towards a better one since selection probabilities for fit and unfit individuals are very similar.

ISBN: 978-988-19251-4-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Moreover, it is difficult to use this selection scheme on minimization problems whereby the fitness function for minimization must be converted to maximization function as in the case of TSP. Although to some degree this solves the selection problem, it introduces confusion into the problem. The best chromosome in the TSP problem, for instance, will continually be assigned a fitness value that is the maximum of all other fitness functions, and thus we are seeking the minimum tour but the fitness maximizes the fitness value. As a consequence several other selection techniques with a probability not proportional to the individual's fitness values have been developed to encounter proportional selection problem. In general there are two types of such non-proportional selection operators: tournament based selection techniques which already been described in the previous section, and the rank-based selections that assign the probability value depending on the order of the individuals according to their fitness values, which will be discussed in the following section.

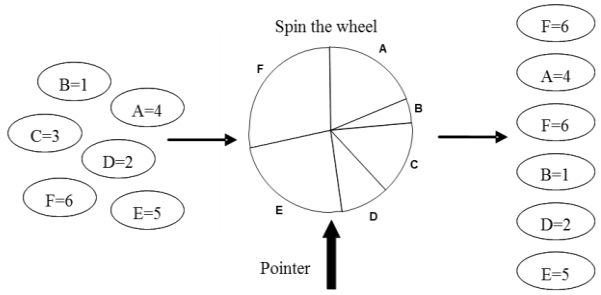


Fig. 4. Selection strategy with roulette wheel mechanism

Procedure: Roulette wheel selection
  While population size < *pop_size* **do**
    Generate *pop_size* random number *r*
    Calculate cumulative fitness, total fitness (p_i) and sum of proportional fitness (*Sum*)
    Spin the wheel *pop_size* times
    If *Sum* < r **then**
      Select the first chromosome, otherwise, select jth chromosome
    End If
  End While
  Return chromosomes with fitness value proportional to the size of selected wheel section
End Procedure

Fig. 5. Procedure for proportional roulette wheel

C. Rank-based Roulette Wheel Selection

Rank-based roulette wheel selection is the selection strategy where the probability of a chromosome being selected is based on its fitness rank relative to the entire population. Rank-based selection schemes first sort individuals in the population according to their fitness and then computes selection probabilities according to their ranks rather than fitness values. Hence rank-based selection can maintain a constant pressure in the evolutionary search where it introduces a uniform scaling across the population and is not influenced by super-individuals or the spreading of fitness values at all as in proportional selection. Rank-based selection uses a function to map the indices of individuals in the sorted list to their selection probabilities. Although this mapping function can be linear (linear ranking) or non-linear (non-linear ranking), the idea of rank-based selection remains unchanged. The performance of the selection scheme depends greatly on this mapping function.

For linear rank-based selection, the biasness could be controlled through the selective pressure *SP*, such that $2.0 \geq SP \geq 1.0$ and the expected sampling rate of the best individual is *SP*, the expected sampling rate of the worst individual is 2-*SP* and the selective pressure of all other population members can be interpreted by linear interpolation of the selective pressure according to rank. Consider *n* the number of individuals in the population, *Pos* the position of an individual in the population (least fit individual has *Pos*=1, the fittest individual *Pos*=*n*) and *SP* the selective pressure. Instead of using the fitness value of an individual, the rank of individuals is used. The rank for an individual may be scaled linearly using the following formula,

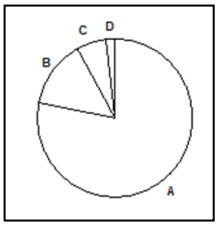
$$Rank(Pos) = 2 - SP + \left(2.(SP-1).\frac{(Pos-1)}{(n-1)} \right) \quad (2)$$

TABLE 1 contains the fitness values of the individuals for various values of the selective pressure assuming a population of 11 individuals and a minimization problem.

TABLE 1. EXAMPLE OF SCALED RANK WITH DIFFERENT *SP* VALUES

Individual fitness value	Rank	Scaled rank with *SP*=2.0	Scaled rank with *SP*=1.1
1	1	2.0	1.1
3	2	1.8	1.08
4	3	1.6	1.06
7	4	1.4	1.04
8	5	1.2	1.02
9	6	1.0	1.00
10	7	0.8	0.98
15	8	0.6	0.96
20	9	0.4	0.94
30	10	0.2	0.92
95	11	0	0.9

Rank-based selection schemes can avoid premature convergence and eliminate the need to scale fitness values, but can be computationally expensive because of the need to sort populations. Once selection probabilities have been assigned, sampling method using roulette wheel is required to populate the mating pool. Rank-based selection scheme helps prevent premature convergence due to "super" individuals, since the best individual is always assigned the same selection probability, regardless of its objective value. However this method can lead to slower convergence, because the best chromosomes do not differ so much from other ones. The different between roulette wheel selection with proportionate fitness and rank-based fitness is depicted in Fig. 6a and Fig. 6b respectively while the GA procedure for rank-based selection implementation is given in Fig. 7.



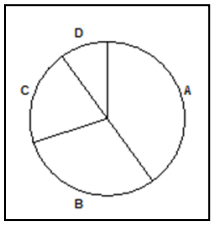


Fig. 6a. Proportionate fitness Fig. 6b. rank-based fitness

ISBN: 978-988-19251-4-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

```
Procedure: Linear rank-based roulette wheel selection
  While population size < pop_size do
        Sort population according to rank
        Assign fitnesses to the individuals according to linear rank function
        Generate pop_size random number (r)
        Calculate cumulative fitness, total fitness and sum of proportional fitness (Sum)
        Spin the wheel pop_size times
        If Sum < r then
          Select the first chromosome, otherwise, select jth chromosome
        End If
  End While
  Return chromosomes with fitness value proportional to the size of selected wheel section
End Procedure
```

Fig. 7. Procedure for rank-based roulette wheel

V. COMPUTATIONAL EXPERIMENTS AND RESULTS

A. Experimental Set-up

This section will focus on computational experiment that use three GA selection schemes discussed in this paper to obtain optimal solution for TSP. The algorithms are coded in MATLAB version 2009b. The performance of GA is tested at eight TSP instances: randomly generated of 10-city, 20-city, 30-city and 40-city, and the known optimal solution TSP instances taking from TSPLIB [15]; burma14, bay29, dantzig42 and eil51. For all experiments, the GA procedure employed a combination of linear order crossover and inversion mutation for producing offspring at every generation. The tournament size used in the tournament selection is set to 2, while the selective pressure used in the rank-based selection is set to 1.1 for all runs. The objective of the experiment is to investigate the performance of GA with different selection strategies in terms of number of generations and iteration time to come out with the optimal solution for TSP.

One of the main difficulties in building a practical GA is in choosing suitable values for parameters such as population size, probability of crossover (*Pc*), and probability of mutation (*Pm*). In this experiment, we follow De Jong's guideline which is to start with a relatively high *Pc* (≥ 0.6), relatively low *Pm* (0.001-0.1), and a moderately sized population [16]. The selections of parameter values are very depend on the problem to be solved. This experiment will use a constant population size which is approximately 10 times larger than number of instance. Noted that the larger the population size, the longer computation time it takes. In this experiment, the GA parameters were obtained from the screening experiment and trial run. For each experiment, the algorithms were run ten times and the lowest travelling distance is taken as a final result. For all experiments in this study, termination is performed when number of generation reached the maximum number of generation. The maximum number of generation is set earlier in the program code.

B. Experimental Results

TABLE 2 shows the best results obtained for eight TSP instances run with different selection strategy. It is clearly shows that GA with rank-based roulette wheel selection always gives the highest solution quality (i.e. minimum travelling distance) for all TSP instances tested. This is then followed by tournament and proportional roulette wheel. Tournament and proportional roulette wheel is able to achieve optimal solution for small size instances; however the quality of solution reduces as the size of instance increase. The percentage of deviation from the known optimal solution concerning problems in the TSPLIB can be seen as a chart in Fig. 8. It shows that GA with rank-based roulette wheel selection is superior than that of tournament and proportional roulette wheel where the results of rank-based roulette wheel does not gives any deviation (0%) from the optimal solution for the three instances: burma14, bay29, and dantzig42, and less than 1% deviation for eil51. Tournament selection apparently gives better results than proportional roulette wheel for all size of problems tested.

TABLE 2. RESULTS OF THE BEST SOLUTION FOR ALL INSTANCES

Instances	Known optimal solution	Tournament	Proportional	Rank-based
10-city	-	2.8567	2.8567	2.8567
20-city	-	4.0772	4.0772	4.0772
30-city	-	4.8352	4.9075	4.6683
40-city	-	6.1992	6.5127	5.7311
burma14	30.8785	30.8785	30.8785	30.8785
bay29	9074	9077	9079	9074
dantzig42	679	725	760	679
eil51	425	470	513	430

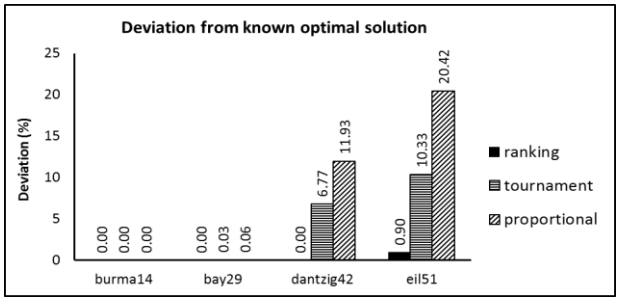


Fig. 8. Deviation from known optimal solution

The performance graphs in Fig. 9 show the minimum distance found by the algorithm in each generation. As we can see from the graph, the distance reduced towards optimal solution as the generation increased and finally converged at a certain generation. For instance in dantzig42, it shows that the algorithm with tournament and proportional roulette wheel selection converged at generation 82 and 135 respectively, where there is no more improvement made after this generation. On the other hand rank-based selection is able to reach optimal solution without premature convergence. Although with slower convergence (i.e. high number of generations), rank-based algorithm performs highly competitive in terms of solution quality, achieving minimum travelling distance.

The graphs in Fig. 10 compare the iteration time between three different strategies. Obviously, rank-based roulette wheel consumes the highest iteration time, hence high computation time due to large number of generations involved to complete the evolution process. The iteration time for tournament is slightly better than proportional roulette wheel in producing comparable results of minimum travelling distance. This indicates that in general tournament is superior to proportional roulette wheel in achieving good quality solution with less computation time.

ISBN: 978-988-19251-4-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

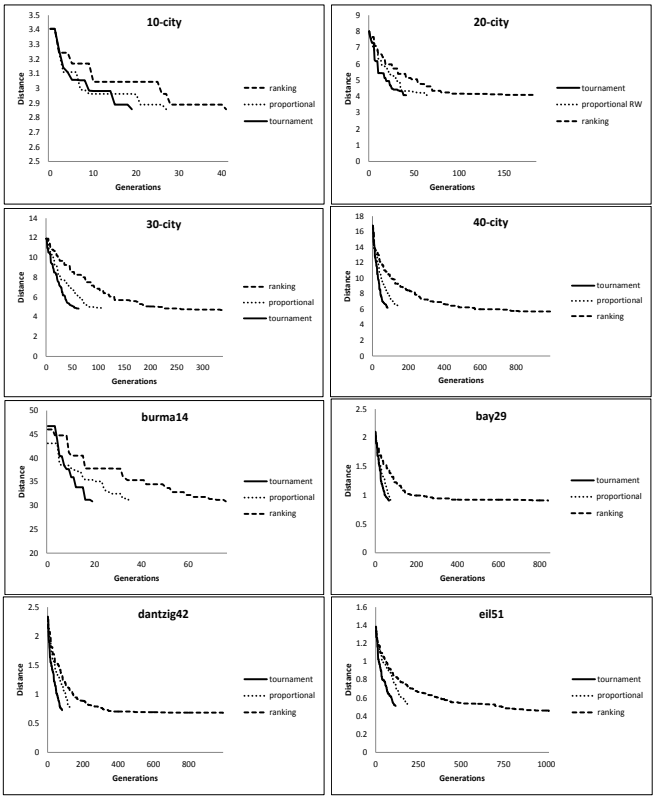


Fig. 9. Performance graph for all instances showing number of generations to converge

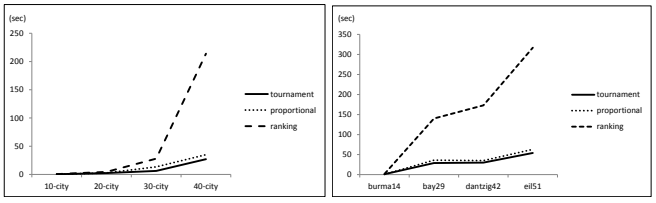


Fig. 10. Iteration time comparisons

VI. CONCLUSIONS

In this paper we have described three types of selection strategy in the GA procedure to solve TSP and compare their performance in terms of solution quality and number of generations to come out with the best solution. From the results of experiment on eight TSP instances, it can be conclude that the quality of solution improved with rank-based roulette wheel selection scheme. We have found optimal solutions for every TSP instance we have tried except for eil51, to within 0.9% deviation of a known optimal solution. GA cannot be expected reliably to find optimum solutions, but it can yield excellent near optimal solutions, which are adequate for most practical problems where input data are only approximate. The results also revealed that the GA based tournament selection is more efficient in obtaining minimum total distance with less number of generation and fastest iteration time compared to the other two strategies. However, this is only applicable for small problem size (i.e. 10-city, 20-city and burma14). As the size of problem increase, tournament selection as well as proportional roulette wheel becomes susceptible to premature convergence. Rank-based selection on the other hand continues to explore the search space and reaching the lowest traveling distance in the tour. Therefore it can be conclude that tournament selection is more appropriate for small size problem while rank-based roulette wheel can be used to solve larger size problem. There is always a trade-off between computation time and the solution quality. If solution quality is the main concern and computation time is still negotiable, then rank-based selection strategy is the best choice.

Future work could be to evaluate the interaction between selection pressure and selection strategies. For example, instead of using binary tournament, we could vary the tournament size to increase the selection pressure. Future work could also extend the model to include precedence constraint TSP. Precedence constraint can increase problem complexity and may results in a different convergence behavior, which could lead to a conclusion on whether a superior selection method exists without regard to problem size and complexity.

REFERENCES

[1] D. Beasley, D. Bull, and R. Martin, An Overview of genetic algorithms: Part 1, Fundamentals, University Computing, vol. 2, pp. 58-69, 1993

[2] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, D. B. Shmoys, The Traveling Salesman Problem, John Wiley & Sons Ltd. , 1985

[3] O. A. Jadaan, L. Rajamani, C. R. Rao, "Improved Selection Operator for GA," *Journal of Theoretical and Applied Information Technology*, 2005

[4] J. Zhong, X. Hu, M. Gu, J. Zhang, "Comparison of Performance between Different Selection Strategies on Simple Genetic Algorithms," *Proceeding of the International Conference on Computational Intelligence for Modelling, Control and automation, and International Conference of Intelligent Agents, Web Technologies and Internet Commerce*, 2005

[5] B. A. Julstrom, It's All the Same to Me: Revisiting Rank-Based Probabilities and Tournaments, Department of Computer Science, St. Cloud State University, 1999

[6] S. Mashohor, J. R. Evans, T. Arslan, Elitist Selection Schemes for Genetic Algorithm based Printed Circuit Board Inspection System, Department of Electronics and Electrical Engineering, University of Edinburgh, 974 – 978, 2005

[7] K. S. Goh, A. Lim, B. Rodrigues, Sexual Selection for Genetic Algorithms, *Artifial Intelligence Review* 19: 123 – 152, Kluwer Academic Publishers, 2003

[8] D.E. Goldberg and K. Deb, A comparative analysis of selection schemes used in genetic algorithms, in: G.J.E. Rawlins (Ed.), Foundations of Genetic Algorithms, Morgan Kaufmann, Los Altos, 1991, pp.69–93.

[9] J. H. Holland, Adaptation in natural and artificial systems, The University of Michigan press, 1975

[10] P. Larranaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, S. Dizdarevic, Genetic algorithms for the Travelling Salesman Problem: A Review of Representations and Operators, *Artificial Intelligence Review* 13: 129 – 170, 1999

[11] Handbook of Evolutionary Computation, IOP Publishing Ltd. and Oxford University Press, 1997

[12] T. Blickle, L. Thiele, A Comparison of Selection Schemes used in Genetic Algorithms. TIK-Report, Zurich, 1995

[13] J. E. Baker, "Adaptive selection methods for genetic algorithm," *Proceeding of an International Conference on Genetic Algorithms and Their Applications*, 100 – 111, 1985

[14] D. Whitley, "The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is the best," *In Proceeding of the 3rd International Conference on Genetic Algorithms*, 1989

[15] G. Reinelt, TSPLIB – A Travelling Salesman Problem Library. *ORSA Journal on Computing*, Vol.3, No.4, 376 – 384, 1991

[16] K. De Jong, W. M. Spears, "Using Genetic Algorithms to Solve NP Complete Problems," *Proceedings of the Third International Conference on Genetic Algorithm, Morgan Kaufman*, Los Altos, CA, 124 – 132, 1989

ISBN: 978-988-19251-4-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)