The Interactive Effects of Operators and Parameters to GA Performance Under Different Problem Sizes*

Jaderick P. Pabico†
Institute of Computer Science
University of the Philippines Los Baños
College 4031, Laguna
jppabico@uplb.edu.ph

Eliezer A. Albacea‡
Institute of Computer Science
University of the Philippines Los Baños
College 4031, Laguna
eaalbacea@uplb.edu.ph

ABSTRACT

The complex effect of genetic algorithm's (GA) operators and parameters to its performance has been studied extensively by researchers in the past but none studied their interactive effects while the GA is under different problem sizes. In this paper, We present the use of experimental model (1) to investigate whether the genetic operators and their parameters interact to affect the offline performance of GA, (2) to find what combination of genetic operators and parameter settings will provide the optimum performance for GA, and (3) to investigate whether these operator-parameter combination is dependent on the problem size. We designed a GA to optimize a family of traveling salesman problems (TSP), with their optimal solutions known for convenient benchmarking. Our GA was set to use different algorithms in simulating selection (Ω_s), different algorithms (Ω_c) and parameters (p_c) in simulating crossover, and different parameters (p_m) in simulating mutation. We used several n-city TSPs ($n = \{5, 7, 10, 100, 1000\}$) to represent the different problem sizes (i.e., size of the resulting search space as represented by GA schemata). Using analysis of variance of 3-factor factorial experiments, we found out that GA performance is affected by Ω_s at small problem size (5-city TSP) where the algorithm Partially Matched Crossover significantly outperforms Cycle Crossover at 95% confidence level. Under intermediate problem sizes (7-city and 10-city TSPs), we found out that the mean GA performance is affected by the $\Omega_s \times \Omega_c$ interaction where the average performance of GA across p_c and p_m varies at different Ω_s-Ω_c combinations. At big problem sizes (100-city and 1000-city TSPs), we observed that a 3-way interaction among Ω_s, Ω_c, and p_m exist to affect the GA performance averaged across different p_c. Similarly, we also observed that the 3-way interaction among Ω_s, p_c and p_m affects the GA performance averaged across all Ω_c. To explain these three-way interactions, we used the Duncan's Multiple Range Test at 5% probability level to perform pairwise comparison of means of GA performance.

Keywords

Genetic algorithms, traveling salesman problem, experimental models, combinatorial optimization.

1. INTRODUCTION

Genetic Algorithms (GAs) are probabilistic search techniques suited for solving large, complex, multidimensional, multimodal, discontinuous, and/or noisy search and optimization problems. Applied to such problems, GAs outperformed several tested search and optimization procedures such as the gradient techniques and some various forms of random search [5, 6, 7, 9, 12, 13]. In the past years, the GA algorithms for selection, crossover, and mutation and the GA parameters population size, crossover probability, and mutation rate have received much attention in research [14, 3, 18]. These studies show that depending on the operators used and the parameter setting, the behavior of the GA can range from that of random search to hill climbing [14]. Thus, designing a GA that would meet a specific problem domain's resource constraints would require a significant effort in trying to find out the right GA operator-parameter combination.

Many researchers have attempted to find a set of genetic operators and parameters for GAs to perform optimally for solving a given problem domain [8, 11, 17, 7, 3, 18, 14]. These researchers have used techniques such as hand optimization, a meta-GA, brute force search, and adapting parameters which are costly and time consuming [8, 11, 17, 7]. The techniques' results can only give parameter

*The preliminary result of this study was presented as a plenary session paper and subsequently published in the proceedings (CD-ROM) of the *Workshop and Conference on Modeling, Simulation, and Scientific Computing (Model 99)* held on 19–20 November 1999 at the Ateneo de Manila University, Quezon City.

†Assistant Professor 6; http://www.ics.uplb.edu.ph/jppabico

‡Professor 11 and Institute Director

Contributed article to the Philippine Computing Journal
ISSN 1908-1995

settings that are robust on a particular problem (such as the Traveling Salesman Problem (TSP)), but not on all other problems in a particular domain (such as the combinatorial problem domain where TSP is classified) [7]. Furthermore, the parameters found in any of these techniques become a liability for GA when the GA structure is modified, such as using another crossover algorithm. Thus, the optimal parameters that resulted from any of the techniques described above may not be good for any GA solving another problem, even to those belonging to the same domain. On the other hand, experimental models can be used to answer the following questions which can not be answered by the techniques used by other researchers:

1. Are these genetic operators and their parameters act independently or dependently on GA performance?

2. If they act independently, how these operators and their parameters affect GA performance? What trend (i.e, linear, quadratic, etc.) these parameters give on GA performance?

3. If they act dependently, which of these operators and their parameters interactively affect GA performance and how?

Results of past studies [16, 15] have shown that experimental models can be a standardization technique for GAs. In these studies, an optimal set of genetic operators and parameters for GAs solving problems under the parametric optimization domain was found. The interactive effects of crossover probability, mutation rate, and population size on GA convergence velocity in parameterizing a multiple objective model were determined [15]. The convergence velocity was measured using the offline metric proposed by de Jong [8] while the interaction was measured using a three-factor factorial analysis on the variance of the GA operator-parameter combinations. A GA that uses the combination of 0.60 one-point crossover probability, mutation rate varied over generation and gene representation, and a population density of 30 was found efficient under this problem domain [15]. No explanation, however, was given on how these operators and parameters affect GA performance. In our current effort, we aim to find the same optimal set of genetic operators and parameters for a GA solving problems under the combinatorial optimization domain. In addition, we will attempt to explain how these operators and parameters affect GA performance and investigates whether problem size is also a factor.

In this paper, we report the results of applying experimental models in measuring the interactive effects of operators and parameters on GA performance. Measuring the effects follows that the specific operators and parameters can be determined to give GA its best performance. Specifically, we used the n-factor ANOVA on the interactive effects of operators and parameters to GA convergence. An n-factor ANOVA, depending upon a certain probability level, tells how n factors interactively affect a certain response measure (i.e., GA performance) via the goodness-of-fit of the data to the n-factor linear model. Although only a few researches have been reported to have used experimental models to compute for and compare different algorithms' performance [1, 2, 16, 15], this method offers flexibility and ease of use compared to mathematical analyses or analyses of algorithms.

Our main objective in this study is to show that experimental models can be a standardization method for GA. Specifically, we aim (1) to investigate the relationship between the problem size and the GA operators and their parameters, (2) to investigate whether the selection, crossover and mutation operators act independently on GA performance using n-factor ANOVA, and (3) to suggest genetic operators and their parameters for GA in solving optimization problems under the combinatorial domain. With the promise of GA's general applicability to solve problems, many optimization and search studies can be conducted to try and use this technique. Knowing the relationships between problem size and the genetic operators and parameters that would give GAs an optimal performance, researchers can save time fine tuning their GAs. Further, having known that experimental model can be a standardization technique for GAs, more genetic operators can be devised that can give efficient GAs.

2. REVIEW OF RELATED LITERATURE

2.1 Refinements on Traditional Parameters

The operators of a traditional GA are selection (Ω_s), crossover (Ω_c), and mutation (Ω_m). The GAs parameter settings are population size (λ), crossover probability (p_c), and mutation rate (p_m). A traditional GA uses the roulette wheel selection, one-point crossover with $p_c = 0.6$, and bit-mutation with $p_m = 0.033$. The population size, set according to the user's discretion, is an important factor because the population of individuals serves as a mechanism with distributed knowledge. This knowledge is being represented by all the genes in the entire population [14]. Other parameter settings reported in the literature are $p_c = 0.6$, $p_m = 0.001$, $50 \leq \lambda \leq 100$ [8], $p_c \in [0.75, 0.95]$, $p_m \in [0.005, 0.01]$, $20 \leq \lambda \leq 30$ [17], and $p_c = 0.95$, $p_m = 0.01$, $\lambda = 30$ [11].

GA has been used in parametric optimization and much effort has been put into refining the GA to improve its convergence speed. Researchers [8, 11, 17, 7] have used four techniques to find good parameter seetings for GA. These techniques are (1) hand optimization, (2) using a meta-GA, (3) brute force search, and (4) parameters that adapt. de Jong [8] carried out hand optimization to find parameter values for the traditional GA which were good across a set of numerical function optimization problems. The parameter values for single-point crossover and bit mutation were worked out by hand while holding the population size constant.

Using a meta-GA, the same parameters were optimized by the use of another GA [11]. With the same set of problems, the GA-optimized GA improved slightly over the GA with hand-optimized parameters. However, a robust parameter setting that would perform well across the range of problems considered was not found.

Davis [7] proposed a method that would make the operators evolve or adapt to the problem as the GA iterates. The adapting parameters can be used to study new operators and evaluate its performance. This could be an effective technique for separating the valuable operators from those that are not. Schaffer, et al. [17] sampled the possible parameter settings across a range of values using the same set of problems that Grefenstette [11] and de Jong [8] used. It was concluded that a GA's optimal parameter setting vary from one problem to another.

2.2 Measures of GA Performance

de Jong [8] designed two measures to quantify GA's search technique's performance. These are online performance and offline performance. The online performance measures the ongoing performance of the GA and is the running average of all evaluations performed. Mathematically, the online performance is given as

$$\text{Online} = \frac{1}{\Lambda}\sum_{i=1}^{\Lambda} f_i \qquad (1)$$

where Λ is the current number of evaluations and f_i is the ith value of the objective function. This measure is appropriate in situations where the cost of evaluating an individual is related in a monotonically increasing way to its fitness value. The offline performance measures convergence and is the running average of the best performance value. The offline performance is computed as

$$\text{Offline} = \frac{1}{G}\sum_{i=1}^{G} f_{\max,i} \qquad (2)$$

where G is the current generation and $f_{\max,i} = \max\{f_{i,j} : 1 \leq j \leq \lambda\}$ is the best function value obtained from the ith generation. This measure can be used when there is no additional cost for evaluating less-fitted individuals.

3. METHODOLOGY

3.1 GA Architectures for TSP

To solve for TSP, we considered different GA architecture designs. In designing these architectures, the choice for genetic operators is important. Our reasons for choosing the specific genetic operators considered in this study are discussed in the following subsections and are summarized in Table 1.

1. **Selection algorithms**. We considered two selection algorithms in this study: Remainder Stochastic Independent Sampling (RSIS) and Stochastic Universal Sampling (SUS). We selected these two algorithms over the usual roullete–wheel method because they are known to have reduced selection bias [9], giving us assurance that the highly fit individual found at each generation will not be lost by chance in the succeeding generations [4].

2. **Crossover algorithms and probabilities**. We considered two crossover algorithms specifically designed for solving combinatorial problems: Partially Matched Crossover (PMX) and Cycle Crossover (CX). For each algorithm, five crossover probabilities were used, 0.60, 0.65, 0.70, 0.75, and 0.80, which gave us 10 algorithm–probability combinations.

3. **Mutation algorithms**. We decided to use the inversion algorithm to simulate mutation because this method was designed solely for combinatorial problems. We considered five levels of mutation rates as a parameter for this algorithm: 0.02, 0.04, 0.06, 0.08, and 0.10.

To determine whether these GA architectures are dependent or independent on the problem size, we considered five different n-city TSPs, where $n = \{5, 7, 10, 100, 1000\}$. Varying the size of the problem is important to see whether it will have an effect on the operators and parameters found by ANOVA (i.e., will ANOVA give the same operators and parameters regardless of the size of the problem?). Each n-city TSP corresponds to a search space whose size is $n! = \Pi_{k=1}^{n} k = 1 \times 2 \times \cdots \times n$.

We have utilized a total of 100 GA architecures solving TSP under five different problem sizes. We run all GAs until the optimum value for the TSP was reached. For each GA run, we recorded the corresponding offline performance. We performed all GA runs under a multi-programming operating system that is why we only measured the offline performance instead of the actual wall-clock running time.

3.2 Fitness Function for TSP

We transformed the TSP into a maximization problem (i.e., the closed-route that will give the maximum profit) and built the problem around a profit matrix, $\mathbf{PR}$, of known optimum. $\mathbf{PR}$ is similar to a graph's weighted adjacency matrix, encoding the profit of going from one node to the connecting node. Thus, adjacency and profit between the ith and the jth nodes is defined if $\mathbf{PR}_{ij} > 0$. If all off-diagonal elements in the matrix are positive, then the graph is fully-connected. In TSP, the value of the elements along the diagonal of the matrix does not matter.

We constructed $\mathbf{PR}$ creating an $n \times n$ diagonally symmetric positive sparse matrix, $\mathbf{SMat}$, of random elements and by creating a vector, $\mathbf{Rt}$, of length $n+1$ whose first n elements are the random permutation of the first n integers and $\mathbf{Rt}_{n+1} = \mathbf{Rt}_1$. $\mathbf{Rt}$ is the closed route where the maximum profit can be obtained. For example, if $n = 5$, $\mathbf{SMat}$

and **Rt** might be:

$$\mathbf{SMat} = \begin{bmatrix} 17 & 22 & 27 & 15 & 17 \\ 22 & 16 & 18 & 20 & 15 \\ 27 & 18 & 18 & 16 & 17 \\ 15 & 20 & 16 & 13 & 16 \\ 17 & 15 & 17 & 16 & 10 \end{bmatrix}$$

$$\mathbf{Rt} = \begin{bmatrix} 4 & 3 & 5 & 1 & 2 & 4 \end{bmatrix} \quad (3)$$

By taking notice of the maximum element of **SMat**, max(**SMat**) = 27, and adding it by a constant, say MAd = 1, **PR** can be computed using:

$$\mathbf{PR}_{i,j} = \begin{cases} \mathbf{SMat}_{i,j}, & \text{if } i \neq \mathbf{Rt}_y \text{ and } j \neq \mathbf{Rt}_{\mathbf{y+1}} \ \forall 1 \leq y \leq n \\ \mathbf{PR}_{j,i} = \max(\mathbf{SMat}) + \text{MAd}, & \text{otherwise.} \end{cases} \quad (4)$$

The second case, $\mathbf{PR}_{i,j} = \mathbf{PR}_{j,i}$, in equation 4 is necessary so that the same closed route but of different direction (example, in equation 3, $\mathbf{Rt}^* = [4 \ 2 \ 1 \ 5 \ 3 \ 4]$) will have the same maximum profit. The above equation makes sure that the maximum profit TSP will have a maximum profit of $n \times (\max(\mathbf{SMat}) + \text{MAd})$. With respect to our example, the profit of traversing the optimum route is $5 \times (27 + 1) = 168$.

The fitness, f_i, of the ith randomly generated closed-route can be computed by traversing the route using the profit matrix:

$$f = \sum_{y=1}^{n} \mathbf{PR}_{\mathbf{Rt}_y, \mathbf{Rt}_{y+1}}. \quad (5)$$

3.3 Experimental Model

To provide basis for comparison of GA performance as affected by four factors, we used a four-factor ANOVA model. The factors known to have an effect on GA performance are (1) the algorithm used in simulating selection, (2) the algorithm and (3) parameter used in simulating crossover, and (4) the algorithm and parameter used in simulating mutation. If two selection algorithms produce the same relative GA efficiencies with two crossover and mutation algorithms, then either selection algorithms can be used to evaluate GA efficiencies for any combination of crossover and mutation algorithms. If the results are dependent of selection algorithm, then any one or all combinations of the crossover and mutation algorithms may not be adequate for discriminating among the selection-crossover-mutation algorithm combinations.

The factorial treatment design was used to evaluate whether the four factors act independently on GA performance. The factors that we specifically considered in this study are :

1. the selection algorithms (Ω_s) assumed to be discrete with two levels, RSIS and SUS;
2. the crossover algorithms (Ω_c) assumed to be discrete with two levels, PMX and CX;
3. the crossover probabilities (p_c) assumed to be continuous with five levels from 0.60 to 0.80 on 0.05 intervals; and
4. the mutation rate (p_m) with five continuous levels from 0.02 to 0.10 via 0.02 intervals.

By determining whether Ω_s, Ω_c, p_c, and p_m in combination interact to influence the offline performance of the GA, we can find the combinations of GA operators and parameters that would give the best GA offline performance.

The performance (P) of the GA is a function of selection algorithm used (Ω_s), crossover algorithm used (Ω_c), crossover probability used (p_c), mutation rate (p_m) used, the random error (ϵ[1]) inherrent to the experiments used which can not be accounted for by Ω_s, Ω_c, p_c, and p_m, and the interactive effects of Ω_s, Ω_c, p_c, and p_m. The ANOVA model is therefore

$$\begin{aligned} P = {} & \epsilon + \alpha_1\Omega_s + \alpha_2\Omega_c + \alpha_3 p_c + \alpha_4 p_m + \\ & \alpha_5\Omega_s\Omega_c + \alpha_6\Omega_s p_c + \alpha_7\Omega_s p_m + \alpha_8\Omega_c p_c + \\ & \alpha_9\Omega_c p_m + \alpha_{10} p_c p_m + \alpha_{11}\Omega_s\Omega_c p_c + \\ & \alpha_{12}\Omega_s\Omega_c p_m + \alpha_{13}\Omega_s p_c p_m + \alpha_{14}\Omega_c p_c p_m + \\ & \alpha_{15}\Omega_s\Omega_c p_c p_m. \end{aligned}$$

We replicated each GA run four times, each replicate using different random seeds but starting with the same initial population. The analysis of variance tests the hypothesis that $\alpha_i = 0$, $\forall\ i$, with a probability of 5%.

3.3.1 *Varying the Problem Size*

To represent varying problem size, we used different TSP sizes. These sizes are the family of n-city TSPs where $n = \{5, 7, 10, 100, 1000\}$. Interestingly, we note here that when solutions are encoded into GA chromosomes using the permutation form, the size of the problem space becomes $n!$. Increasing the search space from $(n-1)!$ is not disadvantageous to GA but rather advantageous because each chromosome can provide n more schemes, a desirable characteristics according to GA's schema theorem [9]. Thus, problem sizes were grouped in terms of the size of the search space brought about by the normal encoding of the solutions to chromosomes. Both $n = 7$ and $n = 10$ (with search spaces of 6! and 9!, respectively) belong to the intermediate problem size while both $n = 100$ and $n = 1000$ (with search spaces of 99! and 999!, respectively) belong to the big problem size. $n = 5$ represent the small problem size with 120 search points. Because of the extensive computing resources required for performing the experiment involving the bigger problem sizes (i.e, $n = 100$ and $n = 1000$), only the following levels of genetic parameters were used:

1. the crossover probabilities (p_c) with three levels 0.60, 0.70, and 0.80 ; and

[1]The random error effect for each test run is assumed to be $N(0, \sigma^2)$, where N is the normal distribution function with mean 0 and variance σ^2.

2. the mutation rate (p_m) with three levels 0.001, 0.010, and 0.100.

3.3.2 Comparing the Mean GA Performance

To analyze the factors with continuous levels (i.e., p_c and p_m), we partitioned their of sum of squares using trend contrasts. Based on the result of the trend comparison, we performed a regression analysis to model the effect of the factors on GA performance. However, we did not perform the regression when the number of points for regression is less than four. Instead, we performed pairwise comparison on the means of the factors involved. For other factors such as Ω_s and Ω_c, we conducted a pairwise comparison of means using the Duncan's Multiple range Test (DMRT) at 5% probability level to explain the significant effect of these factors to GA performance.

4. RESULTS AND DISCUSSION

4.1 Optimum GA Operators for 5-City TSP

The ANOVA result for the 5-city TSP shows that there is no z-way interaction present, where $z \geq 2$. Table 2 shows that only Ω_c has a significant effect on the average GA performance. All other factors have no effect. A simple comparison of means shows that PMX is a better crossover scheme than CX.

The difference of mean offline performance between PMX and CX can be explained by how these two crossover algorithms behave for some inputs. Given two strings C_A and C_B, $C_A \neq C_B$, that encode the solutions to the 5-city TSP, PMX will always create two new strings C'_A and C'_B where $C_i \neq C'_i$ and $f(C_i) \neq f(C'_i)$. However, in CX, for some C_A and C_B, the created strings might be the same as the parents strings, $C'_A = C_B$ and $C'_B = C_A$. This defeats the purpose of creating new solutions by crossing-over the parent strings. Take for instance $C_A = \{6, 2, 0, 3, 4, 7, 9, 1, 8, 5\}$ and $C_B = \{7, 0, 5, 2, 8, 1, 3, 4, 9, 6\}$. Applying CX on these two solutions gives $C'_A = \{7, 0, 5, 2, 8, 1, 3, 4, 9, 6\}$ and $C'_B = \{6, 2, 0, 3, 4, 7, 9, 1, 8, 5\}$. Inputs of this type make CX unable to create new solutions. Table 3 shows the relative performance of PMX over CX in terms of new solutions found for all Ω_s–p_c–p_m combinations.

4.2 Optimum GA Operators for 7-City and 10-City TSPs

A z-way interaction is present when simple interaction effects of $z - 1$ control variables are not the same at different levels of the zth control control variable. As shown in the analysis of variance tables (Tables 4 and 5) a four-way interaction is not present among Ω_s, Ω_c, p_c, and p_m. However, a two-way interaction is present between Ω_s, and Ω_c. The offline performance of the GA behave differently at different Ω_s–Ω_c combinations (averaged across p_c and p_m) which means that varying the values of p_c and p_m will not affect the average offline performance of the GA. The DMRT groupings explain these interactions as shown in Table 6. At 7-City TSP, RSIS–CX, RSIS–PMX, and SUS–PMX are not different from each other while SUS–CX and SUS–PMX have the same effect on GA performance. At 10-City TSP, RSIS–PMX, SUS–CX, and SUS–PMX have the same effect on GA performance and are different from RSIS–CX. The effect of replication (i.e, random seed) on mean GA performance is significant at 7-City TSP only. The presence of significant variability among replications at 7-City TSP suggests that the GA offline performance is dependent on the random number used. This confirms the earlier results of experiments conducted by Goldberg, et al. [10] that GA offline performance is dependent also on the initial population used.

4.3 ANOVA Result for 100-City and 1000-City TSPs

Tables 7 and 9 show the ANOVA of GA offline performance for 100-city and 1000-city TSP, respectively. As both results show, two three-way interactions, Ω_s–Ω_c–p_m and Ω_s–p_c–p_m, exhibit significant differences among their factors.

DMRT explains the significant differences of these factors (Tables 8, 10, 8, and 12). Solving a 100-city TSP, the least Ω_s–Ω_c–p_m combination for a GA is SUS, CX, and 0.001, respectively. No specific best combination can be recommended as several combinations can be bests as seen by the DMRT groupings (Table 8). Three different groupings were identified by DMRT for the Ω_s–p_c–p_m combinations (Table 10). The least Ω_s–Ω_c–p_m combination for a GA that solves 1000-city TSP has Ω_s = SUS, Ω_c = CX, and $p_m = 0.001$ (Table 11). Two inferior Ω_s–p_c–p_m combinations were also identified , SUS–0.70–0.001 and SUS–0.80–0.001 (Table 12). All other combinations are better.

5. SUMMARY AND CONCLUSION

This study aimed to find the interactive effects of different genetic operators and their parameters on GA offline performance using 4-way ANOVA. Several n-city TSPs were considered as test beds, where $n = \{5, 7, 10, 100, 1000\}$. Problem size (i.e., search space) was hypothesized to have an effect on the optimum GA operators and parameter settings.

ANOVA shows that at a smaller problem size (i.e., 5-city TSP), only Ω_c has a significant effect on GA offline performance. All other operators and parameters do not affect GA offline performance when the problem size is small. This difference was explained by the way the two Ω_c algorithms behave. It was found out that PMX is better than CX. When the problem size is intermediate (i.e., 7-City and 10-City TSPs), Ω_s and Ω_c interact to affect the mean GA performance. No trend as to what Ω_s–Ω_c combination is best for this problem size can be concluded as DMRT showed different groupings at different problem sizes.

At bigger problem sizes (n-city TSPs where $n = \{100, 1000\}$), the Ω_s–Ω_c–p_m and Ω_s–p_c–p_m combinations affect the GA offline performance. No specific behavior on the continuous parameters (i.e, p_m and p_c) were found by the regression analysis. Instead DMRT explains the significant three-way interaction among the factors (Ω_s, Ω_c, p_c, and p_m). Table 13 summarizes the results of this study.

It is now therefore concluded that at a smaller problem size, only Ω_c will have a significant effect on GA offline performance. Between the two Ω_c considered, PMX has a significantly higher mean GA offline performance than that of CX. When the problem size is intermediate, Ω_s and Ω_c interact to affect GA performance. No recommendation as to what combination is best can be given as different groupings were found by DMRT at different problem size within the intermediate range. At bigger problem sizes, the combination of Ω_s–Ω_c–p_m and Ω_s–p_c–p_m significantly affect the mean GA offline performance. Ω_s = SUS, Ω_c = CX, $p_m = 0.001$ is a worst setting for a GA that solves 100-city TSP. The combination of Ω_s = SUS, Ω_c = CX, $p_m = 0.001$ is worst for a GA that solves a 1000-city TSP. Similarly, both Ω_s = SUS, $p_c = 0.70$, $p_m = 0.001$ and Ω_s = SUS, $p_c = 0.80$, $p_m = 0.001$ combinations are worst for the same problem.

6. ACKNOWLEDGEMENTS

This work was supported by (CHED's National Center of Excellence in Information Technology Education) Institute of Computer Science, College of Arts and Sciences, University of the Philippines Los Baños through UPLBGF #2326103 and UPLBFI #2004987.

7. REFERENCES

[1] S. M. Alviar. Application of experimental models for algorithm performance experiments. In I. S. Francis, B. F. J. Manly, and F. C. Lam, editors, *Pacific Statistical Congress*, pages 179–181. Elsevier Science Publishers, North-Holland, 1986.

[2] S. M. Alviar. Performance estimation and comparison experiments on personal computer systems. Technical report, Institute of Computer Science, University of the Philippines Los Baños, 1997. Professorial Chair Lecture.

[3] T. Bäck. The interaction of mutation rate, selection, and self-adaptation within a genetic algorithm. In R. Männer and B. Manderick, editors, *Parallel Problem Solving from Nature*, pages 85–94, Amsterdam, 1992. Elsevier.

[4] J. E. Baker. Reducing bias and inefficiency in the selection algorithm. In *Proceedings of the Second International Conference on Genetic Algorithms*, pages 14–21, Cambridge, Massachusetts, 1987. Erlbaum.

[5] S.J. Beaty. Genetic algorithms versus tabu search for instruction scheduling. In Rudolf F. Albrecht, Colin R. Reeves, and Nigel C. Steele, editors, *Artificial Neural Nets and Genetic Algorithms: Proceedings of the International Conference in Innsbruck, Austria, 1993*, pages 496–501. Springer-Verlag, 1993.

[6] L. Davis. *Genetic Algorithms and Simulated Annealing*. Morgan Kaufmann Publishers, Inc, Los Altos, California, 1987.

[7] L. Davis, editor. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, 1991.

[8] K. de Jong. An analysis of the behaviour of a class of genetic adaptive systems. Ph.D. Thesis. University of Michigan. University Microfilms Number 76-9381. Print Index Reference: DAI36-10B:5140, 1975.

[9] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Publishing Company, Inc, Reading, Massachusetts, 1989.

[10] D. E. Goldberg, K. Deb, and J. Clark. Genetic algorithms, noise, and the sizing of population. *Complex Systems*, 6(4):333–362, 1992.

[11] J.J. Grefenstette. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 16(1):122–128, 1986.

[12] John J. Grefenstette and James E. Baker. How genetic algorithms work: A critical look at implicit parallelism. In J.D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 20–27, San Mateo, California, 1989. Morgan Kaufmann Publishers Inc.

[13] J. H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology*. MIT Press, Cambridge, Massachusetts, 1992.

[14] M. Lee and H. Takagi. A framework for studying the effects of dynamic crossover, mutation, and population sizing in genetic algorithms. In T. Furuhashi, editor, *Advances in Fuzzy Logic, Neural Networks and Genetic Algorithms*, pages 111–126. Springer, Tokyo, Japan, 1994.

[15] J.P. Pabico. A genetic algorithm approach for the determination of cultivar coefficients of crop models. M.S. Thesis, University of Georgia, 1996.

[16] J.P. Pabico, G. Hoogenboom, and R.W. McClendon. Determination of cultivar coefficients of crop models using a genetic algorithm:a conceptual framework. *Transactions of the ASAE*, 1(42):223–232, 1999. Presented also as ASAE Paper No. 96-3101.

[17] J.D. Schaffer, R. Caruana, L. Eshelman, and R. Das. A study of control parameters affecting online performance of genetic algorithms for function optimization. In J.D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 51–60. Morgan Kaufmann, 1989.

[18] N. Schraudolph and R. Belew. Dynamic parameter encoding for genetic algorithms. Technical Report Technical Report CS90-175, University of California San Diego, La Jolla, CA 92093-0114, 1992.

Table 1: Genetic operators and parameters considered in designing a GA for solving TSP.

Genetic Operator	Algorithm	Parameter Setting
Selection	RSIS	
	SUS	
Crossover	PMX	0.60, 0.65, 0.70, 0.75, 0.80
	CX	0.60, 0.65, 0.70, 0.75, 0.80
Mutation	Inversion	0.02, 0.04, 0.06, 0.08, 0.10

Table 2: ANOVA table of offline performance of a GA solving a 5–City TSP.

Source of Variation	Degree of Freedom	Sum of Squares	Mean Square	F-Value	Pr> F
Replication	3	86682.58	28894.19	1389.71	0.0001
Ω_s	1	23.22	23.22	1.12	0.2914
Ω_c	1	1817.18	1817.18	87.40	0.0001
p_c	4	32.72	8.18	0.39	0.8133
p_m	4	31.88	7.97	0.38	0.8204
$\Omega_s \times \Omega_c$	1	3.97	3.97	0.19	0.6623
$\Omega_s \times p_c$	4	58.95	14.73	0.71	0.5864
$\Omega_s \times p_m$	4	158.94	39.73	1.91	0.1085
$\Omega_c \times p_c$	4	61.31	15.32	0.74	0.5672
$\Omega_c \times p_m$	4	45.87	11.46	0.55	0.6980
$p_c \times p_m$	16	8.19	0.51	0.02	1.0000
$\Omega_c \times \Omega_c \times p_c$	4	42.79	10.69	0.51	0.7251
$\Omega_s \times \Omega_c \times p_m$	4	28.68	7.17	0.34	0.8475
$\Omega_s \times p_c \times p_m$	16	19.81	1.23	0.06	1.0000
$\Omega_c \times p_c \times p_m$	16	37.54	2.34	0.11	1.0000
$\Omega_s \times \Omega_c \times p_c \times p_m$	16	25.81	1.61	0.08	1.0000
Error	297	6175.07	20.79		
Total	399	95254.58			
CV=2.07					

Table 3: Comparison of performance between PMX and CX.

Ω_s	p_c	p_m	PMX Actual Count	PMX Expected Count	PMX %	CX Actual Count	CX Expected Count	CX %
RSIS	0.6	0.001	2988	2988	100	1872	2992	62.57
RSIS	0.6	0.010	2981	2981	100	1871	3004	62.28
RSIS	0.6	0.100	2945	2945	100	1895	3014	62.87
RSIS	0.7	0.001	3520	3520	100	2264	3504	64.61
RSIS	0.7	0.010	3499	3499	100	2158	3504	61.82
RSIS	0.7	0.100	3508	3508	100	2202	3516	62.63
RSIS	0.8	0.001	4000	4000	100	2481	3981	62.32
RSIS	0.8	0.010	3992	3992	100	2495	3986	62.09
RSIS	0.8	0.100	4001	4001	100	2583	4055	63.70
SUS	0.6	0.001	2962	2962	100	1842	2989	61.63
SUS	0.6	0.010	2955	2955	100	1827	2975	61.41
SUS	0.6	0.100	2975	2975	100	1908	2943	64.83
SUS	0.7	0.001	2497	2497	100	2143	3488	61.44
SUS	0.7	0.010	3490	3490	100	2138	3474	61.54
SUS	0.7	0.100	3463	3463	100	2240	3461	64.72
SUS	0.8	0.001	3957	3957	100	2472	4001	61.78
SUS	0.8	0.010	3955	3955	100	2468	3991	61.84
SUS	0.8	0.100	3985	3985	100	2555	3965	64.44

Table 4: ANOVA table of offline performance of a GA solving a 7–City TSP.

Source of Variation	Degree of Freedom	Sum of Squares	Mean Square	F-Value	Pr> F
Replication	3	502.88	167.63	7.15	0.0001
Ω_s	1	443.50	443.50	18.92	0.0001
Ω_c	1	120.51	120.51	5.14	0.0241
p_c	4	57.31	14.33	0.61	0.6550
p_m	4	198.60	49.65	2.12	0.0786
$\Omega_s \times \Omega_c$	1	256.91	256.91	10.96	0.0010
$\Omega_s \times p_c$	4	99.53	24.88	1.06	0.3759
$\Omega_s \times p_m$	4	123.41	30.85	1.32	0.2640
$\Omega_c \times p_c$	4	66.55	16.64	0.71	0.5859
$\Omega_c \times p_m$	4	122.37	30.59	1.30	0.2682
$p_c \times p_m$	16	179.65	11.23	0.48	0.9562
$\Omega_c \times \Omega_c \times p_c$	4	45.69	79.44	1.95	0.1024
$\Omega_s \times \Omega_c \times p_m$	4	32.94	723.85	1.41	0.2322
$\Omega_s \times p_c \times p_m$	16	8.98	314.55	0.38	0.9857
$\Omega_c \times p_c \times p_m$	16	16.77	186.71	0.72	0.7782
$\Omega_s \times \Omega_c \times p_c \times p_m$	16	13.90	106.09	0.59	0.8888
Error	297	6963.44	23.45		
Corrected Total	399	10083.49			

CV=1.54

Table 5: ANOVA table of offline performance of a GA solving a 10–City TSP.

Source of Variation	Degree of Freedom	Sum of Squares	Mean Square	F-Value	Pr> F
Replication	3	243.23	81.08	1.06	0.3683
Ω_s	1	1512.35	1512.35	19.69	0.0001
Ω_c	1	2461.45	2461.45	32.05	0.0001
p_c	4	690.55	172.64	2.25	0.0640
p_m	4	619.73	154.93	2.02	0.0920
$\Omega_s \times \Omega_c$	1	735.17	735.17	9.57	0.0022
$\Omega_s \times p_c$	4	331.62	82.90	1.08	0.3668
$\Omega_s \times p_m$	4	273.36	68.34	0.89	0.4703
$\Omega_c \times p_c$	4	585.87	146.47	1.91	0.1092
$\Omega_c \times p_m$	4	122.16	30.54	0.40	0.8103
$p_c \times p_m$	16	816.52	51.03	0.66	0.8282
$\Omega_c \times \Omega_c \times p_c$	4	45.25	79.44	0.59	0.6708
$\Omega_s \times \Omega_c \times p_m$	4	59.33	723.85	0.77	0.5438
$\Omega_s \times p_c \times p_m$	16	47.43	314.55	0.62	0.8694
$\Omega_c \times p_c \times p_m$	16	51.30	186.71	0.67	0.8249
$\Omega_s \times \Omega_c \times p_c \times p_m$	16	1576.64	1.28	0.59	0.2065
Error	297	22811.24	76.81		
Corrected Total	399	34778.01			

CV=1.91

Table 6: DMRT on mean GA performance for 7–City and 10–City TSPs.

Ω_s–Ω_c Combination	Mean GA Performance	
	7-City TSP	10-City TSP
RSIS–CX	316.26a	453.58b
RSIS–PMX	315.76a	461.25a
SUS–CX	312.55b	460.18a
SUS–PMX	315.25ab	462.43a

Table 7: **ANOVA table of offline performance of a GA solving a 100–City TSP.**

Source of Variation	**Degree of Freedom**	**Sum of Squares**	**Mean Square**	**F-Value**	Pr> F
Replication	3	240337	80112	14.96	0.0001
Ω_s	1	28871	28871	5.39	0.0222
Ω_c	1	175147	175147	32.70	0.0001
p_c	2	5659	2829	0.53	0.5912
p_m	2	25907	12953	2.42	0.0940
$\Omega_s \times \Omega_c$	1	18249	18249	3.41	0.0677
$\Omega_s \times p_c$	2	11559	5779	1.08	0.3437
$\Omega_s \times p_m$	2	88359	44179	8.25	0.0005
$\Omega_c \times p_c$	2	31973	15986	2.98	0.0549
$\Omega_c \times p_m$	2	51259	25629	4.78	0.0103
$p_c \times p_m$	4	15830	3957	0.74	0.5676
$\Omega_c \times \Omega_c \times p_c$	2	2684	1342	0.25	0.7788
$\Omega_s \times \Omega_c \times p_m$	2	97260	48630	9.08	0.0002
$\Omega_s \times p_c \times p_m$	4	67941	16985	3.17	0.0167
$\Omega_c \times p_c \times p_m$	4	12972	3243	0.61	0.6596
$\Omega_s \times \Omega_c \times p_c \times p_m$	4	37991	9497	1.77	0.1397
Error	105	562436	5356		
Total	143	1474443			

CV=2.33

Table 8: **DMRT of average GA performance at different combinations of Ω_s, Ω_c, and p_m for 100–city TSP (means with the same letter are not significantly different at 5% level).**

	p_m		
Ω_s–Ω_c	0.001	0.010	0.100
RSIS, CX	4599.1a-c	4626.4ab	4535.2c
RSIS, PMX	4639.6a	4620.6ab	4643.7a
SUS, CX	4438.2d	4555.8bc	4616.3ab
SUS, PMX	4638.7a	4618.6ab	4634.9a

Table 9: **ANOVA table of offline performance of a GA solving a 1000–City TSP.**

Source of Variation	Degree of Freedom	Sum of Squares	Mean Square	F-Value	Pr> F
Replication	3	24256820	8085606	15.15	0.0001
Ω_s	1	2799316	2799316	5.24	0.0240
Ω_c	1	17317700	17317700	32.44	0.0001
p_c	2	575886	287943	0.54	0.5847
p_m	2	2576564	1288282	2.41	0.0945
$\Omega_s \times \Omega_c$	1	1929517	1929517	3.61	0.0600
$\Omega_s \times p_c$	2	1039831	519915	0.97	0.3810
$\Omega_s \times p_m$	2	8608709	4304354	8.06	0.0006
$\Omega_c \times p_c$	2	3224993	1612496	3.02	0.0530
$\Omega_c \times p_m$	2	5050079	2525039	4.73	0.0108
$p_c \times p_m$	4	1675328	418832	0.78	0.5377
$\Omega_c \times \Omega_c \times p_c$	2	281360	140680	0.26	0.7688
$\Omega_s \times \Omega_c \times p_m$	2	9601609	4800804	8.99	0.0002
$\Omega_s \times p_c \times p_m$	4	6794845	1698711	3.18	0.0164
$\Omega_c \times p_c \times p_m$	4	1407357	351839	0.66	0.6218
$\Omega_s \times \Omega_c \times p_c \times p_m$	4	3891617	972904	1.82	0.1300
Error	105	56052384	533832		
Total	143	147083926			
CV=2.01					

Table 10: **DMRT of average GA performance at different combinations of Ω_s, p_c, and p_m for 100–city TSP (means with the same letter are not significantly different at 5% level).**

		p_m		
Ω_s	p_c	0.001	0.010	0.100
RSIS	0.60	4613.2a-c	4689.1a	4567.3bc
RSIS	0.70	4643.4ab	4564.4bc	4606.7a-c
RSIS	0.80	4601.8a-c	4617.1a-c	4597.8a-c
SUS	0.60	4551.9bc	4561.4bc	4621.7a-c
SUS	0.70	4528.7c	4623.6a-c	4648.4ab
SUS	0.80	4534.8c	4576.6bc	4606.7a-c

Table 11: **DMRT of average GA performance at different combinations of Ω_s, Ω_c, and p_m for 1000–city TSP (means with the same letter are not significantly different at 5% level).**

	p_m		
Ω_s–Ω_c	0.001	0.010	0.100
RSIS, CX	45960.6a-c	46185.2ab	45338.5c
RSIS, PMX	46348.5a	46149.5ab	46375.2a
SUS, CX	44308.1d	45517.6bc	46129.6ab
SUS, PMX	46311.4a	46143.2ab	46276.4a

Table 12: DMRT of average GA performance at different combinations of Ω_s, p_c, and p_m for 1000-city TSP (means with the same letter are not significantly different at 5% level).

		p_m		
Ω_s	p_c	0.001	0.010	0.100
RSIS	0.60	46085.7a-d	46844.0a	45677.5b-d
RSIS	0.70	46393.8a-c	45602.0b-d	46005.5a-d
RSIS	0.80	45984.2a-d	46056.1a-d	45927.5a-d
SUS	0.60	45438.5cd	45574.8b-d	46751.1a-d
SUS	0.70	45226.2d	46184.9a-d	46431.5ab
SUS	0.80	45264.5d	45731.5b-d	46026.4a-d

Table 13: Recommended genetic operator and parameter settings for different problem sizes.

Problem Size	**Significant Factor**	**Best/Worst Setting**
5-city TSP	Ω_c	PMX is better than CX
7-City TSP	Ω_s–Ω_c	RSIS–CX, RSIS–PMX, and SUS–PMX behave the same while SUS–CX and SUS–PMX have the same effect
10-city TSP	Ω_s–Ω_c	RSIS–CX is an inferior combination than the other
100-city TSP	Ω_s–Ω_c–p_m Ω_s–p_c–p_m	both SUS–CX–0.001 is worst No recommendation
1000-city TSP	Ω_s–Ω_c–p_m Ω_s–p_c–p_m	SUS–CX–0.001 is worst both SUS–0.70–0.001 and SUS–0.80–0.001 are inferior