A COMPARATIVE STUDY OF FIVE PARALLEL GENETIC ALGORITHMS USING THE TRAVELING SALESMAN PROBLEM

Lee Wang[†], Anthony A. Maciejewski[‡], Howard Jay Siegel[‡], and Vwani P. Roychowdhury[*]

[†]Microsoft Corporation
Redmond, WA 98052-6399 USA
e-mail: leew@microsoft.com

[‡]Parallel Processing Laboratory
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907-1285 USA
e-mail: {maciejew, hj}@ecn.purdue.edu

[*]Electrical Engineering Department
UCLA
Los Angeles, CA 90095-1594 USA
e-mail: vwani@ee.ucla.edu

Abstract

Parallel genetic algorithms (PGAs) have been developed to reduce the large execution times that are associated with serial genetic algorithms (SGAs). They have also been used to solve larger problems and to find better solutions. In this paper, a comparative analysis of five different coarse-grained PGAs is conducted using the traveling salesman problem as the basis of this case study. To make fair comparisons, all of these PGAs are based on the same baseline SGA, implemented on the same parallel machine (IBM SP2), tested on the same set of traveling salesman problem instances, and started from the same set of initial populations. As a result of the experiments conducted in this study, a particular PGA that combines a new subtour technique with a known migration approach is identified to be the best for the traveling salesman problem among the five PGAs being compared.

1. Introduction

Serial genetic algorithms (SGAs) are a promising search heuristic for finding near-optimal solutions in large search spaces. To reduce the large amount of computation time associated with SGAs, parallel GAs (PGAs) have been proposed. PGAs have also been used to solve larger problems and to find better solutions.

There exists a large body of literature that discusses the parallelization of genetic algorithms for many different applications [2]. Previous studies have compared the performance of PGAs with and without migration, demonstrating that migration, in general, results in superior performance, e.g., [7, 8]. One of the features that distinguishes this paper from previous work is that it compares four conceptually different coarse-grain parallelization techniques, including two new approaches along with the standard independent and migration PGAs. In addition, it compares a PGA that is a hybrid of a new and a standard approach. All of these comparisons are done using a common framework. (This is a summary of the research presented in [9].)

The traveling salesman problem (TSP) is used as the basis of this case study. The TSP is a well-known NP-hard combinatorial optimization problem [5]. There are C cities and the distance from city i to city j is d_{ij}. A tour is a path that starts from a city, visits each city exactly once, and goes back to the starting city. A tour is represented by a vector t of C cities. The tour starts from t(0), visits cities in the order they appear in t, and then goes back to t(0) after visiting t(C–1). The goal of the TSP is to find a tour t with the minimum tour length. Five 100-city symmetric ($d_{ij} = d_{ji}$) Euclidean TSP instances with shortest known tour lengths from [6] are used as test cases. They are the Krolak A (KA), Krolak C (KC), Krolak D (KD), Reinelt (RE), and lattice (LT) TSPs.

The hardware platform used in this study was an IBM SP2 distributed memory MIMD machine [1]. All SGA experiments were conducted using one thin node; all PGA experiments were conducted using 16 thin nodes. To describe the PGAs in a general way, PE is used to refer to a node or processing element consisting of a processor-memory pair, and N is used as the number of PEs.

All PGAs were based on the same baseline SGA (presented in Section 2) and started from the same set of initial populations. These PGA approaches are described in Section 3. The experiments quantify each PGA's ability to find quality solutions and to test how quickly the PGAs can find solutions of similar quality. These experiments are presented and analyzed in Section 4.

This research was supported in part by Architecture Technology Corporation under contract number 6005.

2. A Baseline Serial Genetic Algorithm for the TSP

From a search of the literature, it was determined that the SGA in [3] was among the best SGAs for the TSP (referred to as Jog's SGA in this work). A slightly modified version of Jog's SGA is used as the baseline SGA. The chromosome used to represent a tour is the vector t defined in Section 1. An initial population of 128 chromosomes is randomly generated. At each iteration of the baseline SGA, three steps are executed in the following order: the modified Heuristic crossover step, the 2-opt step, and the modified Or-opt step. The baseline SGA stopped when the best solution was not improved for 150 iterations.

The Heuristic crossover step [3] randomly picks 50% of the chromosomes from the population. The picked chromosomes are then randomly paired and two offspring chromosomes are generated from each pair. The Heuristic crossover randomly picks a city as the starting point for each offspring tour. Then the Heuristic crossover operation extends the current partially constructed offspring tour by trying to add the shorter parental next city from the current last city (c) on this partial offspring tour. When the next city of c on both parent tours is already on the current partial offspring tour, a random city is selected from those that are not yet on the partial offspring tour. After this crossover operation, the two offspring tours replace the two parent tours. In the baseline SGA, a modified Heuristic crossover is used, where each pair of parent chromosomes generate only one offspring chromosome, and this offspring replaces the parent tour with the longer length to guarantee that the currently best chromosome will be kept.

The 2–opt step [3] randomly selects one half of the remaining chromosomes of the population (25% of the total chromosomes). On each selected tour, ten 2-opt attempts are made to try to shorten the tour length, each time randomly picking two pairs of adjacent cities on the tour. The 2-opt operator takes these two pairs of adjacent cities (a, b) and (c, d) from a tour. If $d_{ab} + d_{cd} > d_{ac} + d_{bd}$, then the paths between cities a, b and cities c, d on the tour are removed and replaced by the new paths between cities a, c and cities b, d.

The Or–opt step [3] applies the Or-opt operator to the rest of the chromosomes of the population (25% of the total chromosomes). In each chromosome selected, all s-city subtours are chosen one by one (where s is first three, then two, and then one). If the subtour considered can be relocated between two other adjacent cities to form a shorter length tour, the appropriate tour changes are made. In Jog's SGA, for each s-city subtour, one pair of adjacent cities (not on the subtour) is randomly chosen for consideration of the s-city subtour relocation. In the baseline SGA, for each s-city subtour of a chosen tour, rather than only attempt to relocate this subtour between one pair of cities that is randomly chosen, the modified Or–opt step considers all possible pairs of cities for the relocation exhaustively, and the subtour is put between a pair of cities that results in the shortest tour length.

To test the baseline SGA, a total of 50 runs were conducted on each TSP instance. Define the percentage quality difference, D, of one run of a given GA (SGA or PGA) to be the length of the shortest tour found by that GA minus the length of the shortest known tour, divided by the length of the shortest known tour. Table 1 shows the performance of the baseline SGA in terms of solution quality and algorithm execution time.

In [3], Jog, *et al.*, also used the KA and LT TSP instances. The baseline SGA achieved better performance on these TSP instances; however, it typically took longer to execute than Jog's SGA. For the PGA study, it was decided to use the baseline SGA as a basis because it could achieve results within 1% of the best known solutions for the TSPs considered.

3. PGA Definitions

The simplest way of parallelizing a GA is to execute multiple copies of the same SGA, one on each PE. Each of the 16 PEs starts with a different initial subpopulation (128/16 chromosomes), and evolves and stops independently. The complete PGA halts when all PEs stop. There are no inter-PE communications. The advantage of this independent PGA approach is that each PE starts with an independent subpopulation. Such subpopulation diversity reduces the chance that all PEs prematurely converge to the same poor quality solution. This approach is equivalent to simply taking the best solution after multiple executions of the SGA on different initial populations.

The second PGA tested, the migration PGA approach, augments the independent approach with periodic chromosome migrations among the PEs to prevent premature convergence and share high quality solutions (see Figure 1). The migration approach in this research used the scheme proposed in [7]. Chromosome migrations occur every five iterations, with each PE sending a copy of its locally best chromosome to PE P + 1 modulo N at the first migration step, then to PE P + 2 modulo N at the second migration step, and so on [7]. The chromosome received replaces the locally worst chromosome unless an identical chromosome already exists in the local population.

A third way of parallelizing GAs is to partition the search space into disjoint subspaces and to force PEs to search in different subspaces. The only difference

between this partitioned PGA approach and the independent PGA approach is that in the partitioned approach each PE searches in a different subspace while in the independent approach all PEs search in the entire search space. This approach was the worst in terms of solution quality and execution time for each TSP instance because most PEs were limited to searching subspaces that did not contain good solutions [9].

A novel PGA proposed in this research specifically for the TSP involves tour segmentation and recombination. This segmentation PGA approach starts by segmenting tours into subtours. Then after subtour improvements, the subtours are recombined into longer subtours. This subtour improvement and recombination is performed repeatedly until full tours are formed. The execution time of an iteration on subtours is shorter than that on full tours. Each PE performs versions of the modified Heuristic crossover, 2-opt, and modified Or-opt that are restricted for operation on local subtours [9]. Iterations on full tours stop when the same stopping criterion used in the other PGAs is met.

At the initial population generation step, each of the T tours is segmented into N disjoint subtours. PE P is assigned the P-th subtour of each chromosome. The length of each subtour is the distance from the first city of the subtour being considered to the first city of the next subtour of the same full tour. Each PE also knows the first city of the next segment of each full tour.

For each chromosome, at the evaluation step, each PE first locally computes the subtour length from this subtour's first city to the first city of the next subtour. Then all the PEs that share a chromosome perform a recursive doubling operation to get full lengths of all tours. These full tour lengths are used in the crossover step for deciding which parent tour to be replaced by the offspring tour and in the recombination step below.

There are $n+1$ phases in the segmentation approach. Assume that N and T are powers of two, and $n = \log_2 N$. At a given phase e, $0 \le e \le n$, 2^{n-e} PEs form a group of PEs that share a set of tours. For phase e, there are 2^e groups, numbered from 0 to $2^e - 1$. Each PE $P = p_{n-1}p_{n-2}\cdots p_1 p_0$ belongs to group $p_{e-1}\cdots p_1 p_0$ when $e > 0$ and belongs to group 0 when $e = 0$. At the first phase, when $e = 0$, all PEs start as a single group sharing all tours, and in the last phase, where $e = n$, each PE is in its own group and shares no tours with other PEs.

Each PE contains $T/2^e$ subtours, where each subtour involves $C/2^{n-e}$ cities. Recombination occurs at the end of each phase e, except for the last phase when $e = n$. At the beginning of a recombination, the subtours contained by a group of PEs are numbered in ascending order of the corresponding full-tour lengths, which are known to all PEs in the same group. Therefore, all subtours that belong to the same tour have the same order number across the PEs in a group.

Recombination is performed within each group. Each PE $p_{n-1}p_{n-2}\cdots p_{e+1}0\,p_{e-1}\cdots p_0$ sends each corresponding PE $p_{n-1}p_{n-2}\cdots p_{e+1}1\,p_{e-1}\cdots p_0$ its odd-numbered subtours, and each PE $p_{n-1}p_{n-2}\cdots p_{e+1}1\,p_{e-1}\cdots p_0$ sends each corresponding PE $p_{n-1}p_{n-2}\cdots p_{e+1}0\,p_{e-1}\cdots p_0$ its even-numbered subtours. Each PE then concatenates each received subtour to the appropriate end of the locally held subtour with the same tour number. Each PE now has half the number of subtours it had before, but each subtour is twice as long.

Only one evolution iteration is performed for each of the first n phases. At the last phase, when $e = n$, each PE has T/N chromosomes, each of which is a full tour of C cities. At this phase, the segmentation approach becomes the independent approach (see Figure 1).

When N is large, the sum of the time for all of the first n phases is approximately equal to that of one iteration of the last phase. However, it is possible that performing the genetic operations in one iteration on full tours may result in generating better new populations. Thus, the experiments in Section 4 are needed to evaluate the trade-offs.

The last PGA considered in this research is the segmentation-migration approach, which augments the segmentation approach with periodic chromosome migrations. The motivation for designing this algorithm is to combine the characteristics of both the migration and the segmentation approaches, both of which performed well in the initial experimentation.

4. PGA Experiments

To make fair comparisons, the same set of 50 different initial populations of 128 full tours used for testing the baseline SGA were also used for each PGA on each TSP instance. In the independent, the migration, and the partitioned approaches, each initial population was divided into 16 subpopulations, each of which resided on a different PE. In each run of the segmentation and the segmentation-migration approaches, each full tour in the initial population was segmented into 16 disjoint subtours.

Two sets of experiments were designed to quantify the performance of these five PGAs. The purpose of the first set of experiments was to measure the quality of the best solution each PGA could find. The purpose of the second set was to determine how quickly each PGA could find comparable solutions of high quality.

The first set of experiments used a stopping criterion of no improvement in the best solution found after 150 iterations. Each run stopped when all PEs achieved this stopping criterion. On each TSP instance, each PGA was run 50 times, each time starting from a different initial

population. Table 2 shows the statistics on the execution times for each PGA on each TSP instance. In this table, confidence intervals [4] of all mean values were calculated. In this research, a 90% confidence was used, which means that if another set of runs were conducted (the number of which must be statistically significant (e.g., 50)), there is a 0.9 probability that the sample mean of this second set of runs will be in the interval between $x \pm \Delta x$.

From Table 2, it can been seen that on all TSP instances the best solutions found by all PGAs, other than the partitioned approach, were within 1% of the best known solutions, the same range of the solution quality found by the baseline SGA. (The partitioned approach had the worst performance in both the solution quality and the mean execution time for this set of experiments.) On each TSP instance, the migration approaches took significantly shorter mean time to finish than the non-migration approaches. The segmentation-migration approach had the shortest mean execution time. Chromosome migration is effective at reducing the total execution time because the execution time was determined by when the last PE finished. In migration approaches, once the globally best solution is found by one PE, after at most five migration steps all PEs have a copy of this chromosome. However, each PE in the PGAs without chromosome migrations stopped independently.

Comparing the SGA and PGA execution times is meaningful because all experiments used the same stopping criterion and (other than the partitioned approach) all algorithms found comparable quality solutions. Define speedup to be the mean execution time of the baseline SGA divided by that of the segmentation-migration PGA approach on a given TSP instance. The average speedup over the five TSP instances was 13.83, which indicates that the parallelization techniques used by the segmentation-migration PGA approach (i.e., chromosome migration, and tour segmentation and recombination) is quite effective in decreasing the execution time while maintaining the solution quality.

For the first set of experiments, the segmentation-migration approach found solutions all within 1% of the best known solutions with the smallest mean execution time on each TSP instance. To compare the mean execution times of different PGA approaches when they achieve virtually identical quality solutions, a second set of experiments was conducted.

To make fair comparisons, the longest final solution found by a PGA (other than the partitioned approach) for each TSP instance in the first set of experiments was chosen as the stopping criterion, i.e., the PGA stopped when any PE found a tour that is at least as good. For this second set of experiments, the migration approaches achieved the shortest mean execution time for each of the TSP instances. Specifically, the migration approach was the best for two of the five TSP instances (KA and LT TSPs) and the the migration-segmentation approach was the best for the other three TSP instances. For each TSP instance, the independent approach always had a longer mean execution time than the migration approach, and the segmentation approach always had a longer mean execution time than the segmentation-migration approach.

5. Conclusions

Four conceptually different PGA approaches and one hybrid of two of these approaches were compared using the TSP as an example application problem. For fair comparisons, all PGAs were developed based on the same baseline SGA, implemented on the same 16 thin PEs of an IBM SP2 parallel machine, and tested using the same set of initial populations on the same set of TSP instances. Two techniques were identified as the most effective ones for improving the PGA performance: (1) inter-PE information exchange during the evolution progress (e.g., migration of the locally best chromosomes) and (2) tour segmentation and recombination. The segmentation-migration PGA approach, which combined the above two techniques, took the shortest execution time in the first set of experiments (where a practical stopping criterion was used). It was shown in the second set of experiments that the migration and the segmentation-migration approaches were able to find solutions of similar high quality faster than the other approaches.

Acknowledgments: The authors thank Muthucumaru Maheswaran for his comments and assistance.

References

[1] T. Agerwala, J. L. Martin, J. H. Mirza, D. C. Sadler, D. M. Dias, and D. M. Snir, "SP2 system architecture," *IBM Systems Journal*, Vol. 34, No. 2, Feb. 1995, pp. 152-184.

[2] J. T. Alander, *An Indexed Bibliography of Distributed Genetic Algorithms,* Technical Report No. 94-1-PARA, Department of Information Technology and Production Economics, University of Vaasa, Finland, Aug. 1997.

[3] P. Jog, J. Y. Suh, and D. Van Gucht, "The effects of population size, heuristic crossover and local improvement on a genetic algorithm for the traveling salesman problem," *3rd Int'l Conf. Genetic Algorithms*, July 1989, pp. 110-115.

[4] A. M. Law and W. D. Kelton, *Simulation Modeling and Analysis.* New York, NY: McGraw-Hill, 1982.

[5] E. L. Lawler, J. K. Lenstra, and D. B. Shmoys, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization.* New York, NY: Wiley-Interscience, 1985.

[6] G. Reinelt, "TSPLIB - A traveling salesman problem library," *ORSA Journal on Computing,* Vol. 3, No. 4, Apr. 1991, pp. 376-384.

[7] T. Starkweather, D. Whitley, and K. Mathias, "Optimization using distributed genetic algorithms," in *Parallel Problem Solving from Nature,* H. P. Schwefel and R. Manner, Eds. New York, NY: Springer-Verlag, 1992, pp. 176-183.

[8] R. Tanese, "Distributed genetic algorithms," *3rd Int'l Conf. Genetic Algorithms,* June 1989, pp. 434-439.

[9] L. Wang, A. A. Maciejewski, H. J. Siegel, and V. P. Roychowdhury, "A study of five parallel approaches to a genetic algorithm for the traveling salesman problem," http://dynamo.ecn.purdue.edu/~maciejew.

Table 1. Baseline SGA performance.

TSP	D (%)		execution time(sec)	
	μ	max	μ	median
KA	0.00	0.00	399	394
KC	0.00	0.00	419	419
KD	0.04	0.05	459	452
RE	0.09	0.18	490	466
LT	0.31	0.83	393	359

```
each PE executes the following:
   initial population generation;
   if segmentation approach
       then generate subtours;
   evaluation;
   if segmentation approach then
       do for each of the first log2N phases
       { /* all operators are restricted
            to local subtours */
         modified Heuristic crossover;
         2-opt;
         modified Or-opt;
         evaluation;
         recombination;
       }
   while(stopping criteria not met){
      modified Heuristic crossover;
      2-opt;
      modified Or-opt;
      evaluation;
      if migration approach
        then perform chromosome migration
        every fifth iteration;
   }
   output best chromosome;
```

Fig. 1. Outline of the PGA approaches.

Table 2. Performance comparisons of the five PGAs.

TSP	D(%) for the five PGAs									
	independent		migration		partitioned		segmentation		segment.-migrat.	
	μ	max	μ	max	μ	max	μ	max	μ	max
KA	0.00±0.00	0.00	0.00±0.00	0.00	0.12±0.04	0.46	0.00±0.00	0.00	0.00±0.00	0.00
KC	0.00±0.00	0.00	0.00±0.00	0.00	0.77±0.12	1.67	0.01±0.00	0.09	0.00±0.00	0.00
KD	0.03±0.01	0.07	0.01±0.01	0.07	0.64±0.08	1.21	0.02±0.01	0.07	0.03±0.02	0.45
RE	0.01±0.01	0.43	0.00±0.00	0.08	0.69±0.09	1.67	0.01±0.00	0.09	0.01±0.01	0.43
LT	0.38±0.04	0.83	0.04±0.01	0.83	1.13±0.11	1.66	0.16±0.02	0.83	0.16±0.01	0.83
	execution time (sec) for the five PGAs									
	independent		migration		partitioned		segmentation		segment.-migrat.	
TSP	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
KA	42.04±1.45	6.12	28.24±0.41	1.76	53.93±1.73	7.28	43.35±1.43	6.03	26.89±0.42	1.79
KC	45.92±1.37	5.78	32.42±0.98	4.13	54.81±2.07	8.75	42.50±1.09	4.60	29.37±0.77	3.23
KD	48.70±2.03	8.56	35.80±1.00	4.23	60.38±2.15	9.07	48.76±1.50	6.33	30.97±0.90	3.82
RE	54.08±1.68	7.07	38.33±1.52	6.42	59.28±2.18	9.19	53.11±1.67	7.02	37.92±2.33	9.82
LT	39.59±1.16	4.90	31.65±1.45	6.14	44.76±1.17	4.95	39.93±1.56	6.60	30.04±1.14	4.82