Comparing Genetic Algorithm and Guided Local Search Methods by Symmetric TSP Instances

Mehrdad Nojoumian
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Canada
mnojoumi@cs.uwaterloo.ca

Divya K. Nair
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Canada
dknair@cs.uwaterloo.ca

ABSTRACT
This paper aims at comparing Genetic Algorithm (GA) and Guided Local Search (GLS) methods so as to scrutinize their behaviors. Authors apply the GLS program with the Fast Local Search (FLS), developed at University of Essex, and implement a genetic algorithm with partially-mapped and order crossovers, reciprocal and inversion mutations, and rank and tournament selections in order to experiment with various Travelling Salesman Problems. The paper then ends up with two prominent conclusions regarding the performance of these meta-heuristic techniques over wide range of symmetric-TSP instances. First, the GLS-FLS strategy on the s-TSP instances yields the most promising performance in terms of the near-optimality and the mean CPU time. Second, the GA results are comparable to GLS-FLS outcomes on the same s-TSP instances. In the other word, the GA is able to generate near optimal solutions with some compromise in the CPU time.

Categories and Subject Descriptors
I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search – *Heuristic methods.*

General Terms
Experimentation

1. INTRODUCTION
The Travelling Salesman Problem (TSP) has been a widely accepted combinatorial optimization problem, studied for exploring the effectiveness of optimization techniques focused in seeking near optimal solutions to NP-hard problems. There is an abundance of GA approaches and Local Search (LS) heuristic methods proposed for finding better-quality solutions for the TSP problem within an acceptable time limit. Although, GA and LS methods aim at achieving the same goal of yielding near optimal solutions for the hard optimization problems, we are interested to analyze their problem solving behaviors, by executing two classes of methods on the standard s-TSP instances.

The major objective of this paper is to evaluate two well-known meta-heuristic methods by challenging the travelling salesman problem with various numbers of cities. This issue has been in the center of attention by scientists and engineers because numerous of other problems can be mapped into the TSP. Our motivation is to specifically focus on GA and GLS approaches in order to compare them by the assessment of their performance on symmetric TSP instances.

GECCO'08, July 12-16, 2008, Atlanta, Georgia, USA.
ACM 978-1-60558-130-9/08/07.

2. GA and GLS
The genetic algorithm refers to a model introduced and investigated by John Holland and his student [2]. This algorithm encodes potential solutions to a specific problem on a chromosome-like data structure and applies recombination operators to these structures, such as cross over and mutation, in order to explore the solution space. It generates a finite set of random solutions at first and iteratively works towards generating better solutions in each successive step depending on a defined fitness function, in accordance with its selection schemes.

The Guided Local Search proposed by Voudouris [1] is also one of the most effective heuristic search strategies. The major property of the GLS is to escape from the local minimum resulting from the local search process and advance the search process further to promising regions of the search space. In the GLS, search information is converted into constraints on features which then are incorporated in the cost function using modifiable penalty terms. Constraints confine local search to the promising solutions with respect to the previous search information. The GLS provides a simple mechanism for introducing or strengthening constraints on solutions features. Each time local search is trapped in a local minimum, the GLS can increment the penalty parameter of one or more of the features defined over solutions. If the penalty parameter of a feature is incremented, then solutions which have this feature are avoided by local search. A first step in the process of applying the GLS to a problem is to find a set of solution features that are responsible for part of the overall solution cost. Once features and their costs have been defined, GLS can be applied to the TSP.

3. EXPERIMENTAL RESULT
The results are reported for the symmetric TSP problem by extracting benchmark instances from the TSP Library. The s-TSP instances range from 48-1002 cities and covers wide range of distance permutations. The experiments are conducted using the IBM ThinkPad LENOVO, Intel Core 2 Duo CPU; 2.00 GHz with 2.00 GB of RAM and with the Windows XP as the operating system. We compared our results against the known optimal solutions for corresponding instances by a parameter, *Mean Excess %*. This parameter shows the deviation of our results from the known optimal solutions for the instances respectively and is calculated by: *Mean Excess% = [(Obtained Cost – Known*

*Optimal Cost) / Known Optimal Cost] *100*. We executed the s-TSP instances for various runs with a specific time budget.

For GLS we used the GLS solver developed in C++ by [1]. This GLS Solver algorithm uses Fast Local Search and relies on the simple 2-Opt heuristic as the only move operator. The GA algorithm is implemented in Java, for each s-TSP instance, we first generated (4*N, N-number of cities) number of random solutions and then applied the partially mapped crossover, the inverse mutation and the rank selection; since this combination yielded the best outcomes.

Table 1 presents the experimental results of GLS-FLS-2-opt and GA on the selected s-TSP benchmark instances, Figure 1 demonstrates the graphical findings regarding the quality of solutions depicted by *Mean Excess %* and Figure 2 shows the graphical comparison of the two approaches in terms of *Mean CPU Time*. These experimental results indicate that both approaches produce almost the same optimal solutions. This compelled us to deduce that the genetic algorithm and the GLS-FLS-2opt are comparable in terms of near optimal solutions.

Although GA works very similar to GLS-FLS in terms of near optimal solutions, we observed a considerable time difference between these methods for the selected s-TSP instances. This difference is illustrated in Figure 2. First of all, the GLS-FLS searches only a subset of sub-neighborhoods; moreover, the guided local search algorithm, by its inherent nature, is guaranteed to proceed only to the promising regions of the search space. In contrast, the genetic algorithm uses a random search behavior as it proceeds to distinct regions of the search space.

Table 1. Comparison of GLS-FLS-2opt and GA (Time: Sec).

TSP Instances	GLS-FLS-2opt				GA			
	Mean CPU Time	Tour Length	# of Iteration	Excess%	Mean CPU Time	Tour Length	# of Generation	Excess%
eil51	0.57	426	1307	0	3	438	2799	0.46
berlin52	0.26	7542	563	0	2.78	7542	1731	0
eil76	0.72	538	3141	0	13	557	4754	3.53
kroa100	1.7	21282	7485	0	25.2	21466	8942	0.86
kroc100	0.75	20749	9293	0	33.2	21096	6869	1.6
eil101	0.5	629	2315	0	27.9	651	4999	3.49
bier127	6.6	118282	32324	0	90.35	121089	9446	2.37
pr136	12.37	96772	42379	0	103.8	100986	14123	4.3
kroa150	7.55	26524	25902	0	225.3	28073	9100	5.8
u159	4.6	42080	20400	0	445.36	49811	12134	18.37
rat195	11.6	2323	32718	0	236.6	2467	14275	6.19
d198	270	15780	787244	0	264.1	16102	13369	2.04
kroa200	10.25	29368	39244	0	300.275	29368	14998	0
krob200	11.7	29437	24714	0	426.45	30122	14113	2.32
gil262	35.79	2378	95302	0	431.78	2452	6456	3.11
lin318	24.44	42029	151129	0	545.88	42029	14312	0
pcb442	298.7	50778	1954669	0	658	62556	15000	23.14
rat575	66.3	6776	272299	0.04	344.78	6776	5098	0.04
rat783	231.5	8809	959621	0.03	892.56	8915	15000	1.23
pr1002	279.9	259162	1304819	0.045	1008.67	260987	15000	0.74

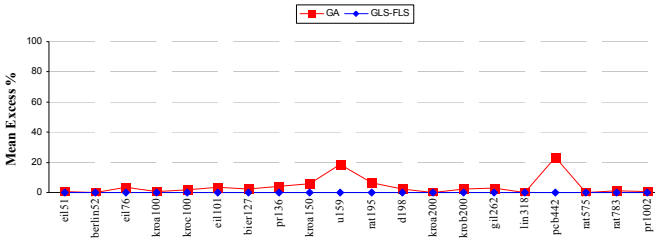

Figure 1. GLS-FLS-2opt vs. GA on Mean Excess %.

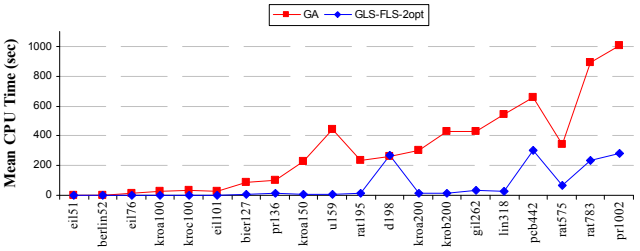

Figure 2. GLS-FLS-2opt vs. GA on Mean CPU Time.

4. CONCLUSION AND FUTURE WORK

The paper first illustrates genetic algorithm and guided local search methods and demonstrates the experimental results regarding these algorithms. Authors then end up with two major conclusions with respect to the performance of these meta-heuristic techniques over wide range of symmetric-TSP instances. First, the GLS-FLS strategy on the s-TSP instances yields the best performance in terms of the near-optimality and the mean CPU time. Second, the GA results are comparable to GLS-FLS outcomes on the same s-TSP instances. In the other word, the GA is able to generate near optimal solutions with some compromise in the CPU time.

Authors also observed that both the GLS-FLS and the GA take almost the same CPU time over multiple runs in order to solve the *d198*, which is different from other instances. Although, in most s-TSP instances cities are distributed all over the XY-coordinate or the density of cities on the XY-coordinate is almost around one region, the distribution of cities in the *d198* is dissimilar; multiple densities and multiple sparsities. As a future work, we intend to investigate this issue in details in order to see if we can come up with some new patterns in TSP instances. We also would like to execute the same s-TSP instances on the Genetic Local Search algorithm. Since this approach is based on the combination of the GLS and the GA; the GLS part of the approach would efficiently handle the local optimum and the GA part of the approach uses the search space of the local optimum to ultimately find the global optimum, thus balancing intensification and diversification.

5. REFERENCES

[1] Voudouris, C. and Tsang, E. 1998. Guided Local Search. European Journal of Operations Research 113, 80-119.

[2] DeJong, K. 1975. An Analysis of the Behavior of a Class of Genetic Adaptive Systems, PhD Dissertation, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor.