Main report submitted to

the Department of Computer Engineering of Cankaya University

in partial fulfillment of the requirement for

CENG505 Parallel Computing

TRAVELLING SALESMAN PROBLEM

By

Duygu ÖZEN

200972200

Fall 2010 – 2011

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

The aim of this study is why parallel computation is used for Travelling Salesman Problem (TSP). Firstly, TSP is explained and its application areas also explained.

Mathematical model and solution procedures of TSP are explained. In the light of this explanation, why parallel computing is applying in TSP and what is of aim. This explanation is described with an example by the article of Evolution Algorithms in Combinatorial Optimization which is studied by Mühlenbein, Schleuter and Kriamer. By the way, the evolution methods of Genetic Algorithms and Simulated Annealing are running for 442 nodes. Finally, these solutions are comparison.

2. TRAVELLING SALESMAN PROBLEM (TSP)

The Travelling Salesman Problem (TSP) is an NP-hard problem in combinatorial optimization. TSP is studied in operations research and theoretical computer science. Given a list of cities and their pair wise distances, the task is to find a shortest path to traverse all cities exactly once and return to the starting city which is shown in Figure1. For N cities there are (N- 1)!/2 possible tours.

Figure 1 TSP example

The TSP has several applications even in its purest formulation, such as;

- Planning
- Logistics
- The manufacture of microchips.

2.1. THE FORMULATION OF TSP

TSP is formulated as followed with the aim of the tour (a directed cycle that contains all *n* cities) of minimal length by Pataki.

Desicion Variables

c_{ij}: The costs between the arc

x_{ij}: $\begin{cases} 1 \text{ if arc (i,j) is in the tour} \\ 0 \text{ otherwise} \end{cases}$

Objective Function

$$Min \sum_{ij} c_{ij} x_{ij}$$

Subject to

$$\sum_{i} x_{ij} = 1 \; \forall i$$

$$\sum_{j} x_{ij} = 1 \; \forall j$$

$$\mathbf{0 \leq x_{ij} \leq 1, x_{ij}\ integer}$$

Slightly modified, it appears as a sub-problem in many areas, such as DNA sequencing. In these applications, the concept city represents, for example, customers, soldering points, or

2.2. SOLUTION PROCEDURES

The TSP is an important problem. It has both theoretical and practical reasons. The TSP can be studied to achieve a better understanding of the TSP from a theoretical point of view. On the other hand, by incorporating additional side constraints such as capacity, distance and time windows restrictions, it could easily be extended to a variety of vehicle routing problems (VRPs).

In the following table solution procedures are proposed for TSP, the first column of the table indicates type of approach and the second column refers to the specific solution procedure

Table 1 The solution procedures of TSP

Type of Approach	Solution Procedure
Exact Solution	Integer linear programming formulations
	Cutting plane
	Branch and bound
	Langrangean relaxation + branch and bound
Heuristics	Simple heuristics
	Evolutionary algorithm
	Simulated annealing
	Tabu search
	Genetic algorithm
	Neural networks

In many applications of TSP, additional constraints such as limited resources or time windows make the problem considerably harder.

2.3. APPLYING PARALLEL COMPUTING FOR TSP

TSP belongs to the class of NP-complete problem. Thus, when the number of cities is increased exponentially, it is likely the worst case running time for any algorithm for the TSP. The main purpose of parallel processing is to perform computations faster than can be done with a single processor by using a number of processors concurrently. TSP can be solved to speed up the computations and to attempt a more thorough exploration of the solution space by the parallel computation that is used to increase the size of the problems by Lei-fu and Wei.

2.4. A TSP EXAMPLE

The article of Evolution Algorithms in Combinatorial Optimization is studied for understanding of why parallel computing would be studied in TSP. This article was studied by Mühlenbein, Schleuter and Kriamer. In this study, TSP is solved with different number of nodes. The 100 cities problem solved by Krolak and the 442 cities problem solved by Grötschel. The best solution of the famous TSP with 442 cities is solved by the algorithm is inherently parallel and shows a super linear speedup in multiprocessor systems that is published by Mühlenbein and the others.

In this study, the unique power of evolution algorithms shows up with parallel computers. With this evolution algorithm for the TSP can be adapted easily to other combinatorial optimization problems. They expect super linear speedup in many different applications.

The evolution process can be described by two models which are followed,

- Models using differential equations for the selection dynamics,
- Models using the 'evolution operators' replication, mutation, selection acting on the individuals.

2.4.1. Computational Results for the TSP

The 100 cities problem solved by Krolak and the 442 cities problem solved by Grötschel and all runs were made on an Encore System with 16 processors and 32 MByte memory, because of scheduling problems, it was not possible to run more than one process on every processor. So the population was small with not more than 16 individuals. The genetic algorithm is implemented in Multiprocessor C (MPC) by Vrsalovic and others. In the context of the shared memory model, MPC is the part of PIE, by Segall and others, effort which looks at parallel processing.

The quality of the solution depends on the number of individuals in the population which is shown in Figure 2. Even with only a few individuals it is possible to be lucky and find the global optimum. But the computational results for the 100 city problem shows, that the chance of getting a high quality solution increases with the number of individuals.

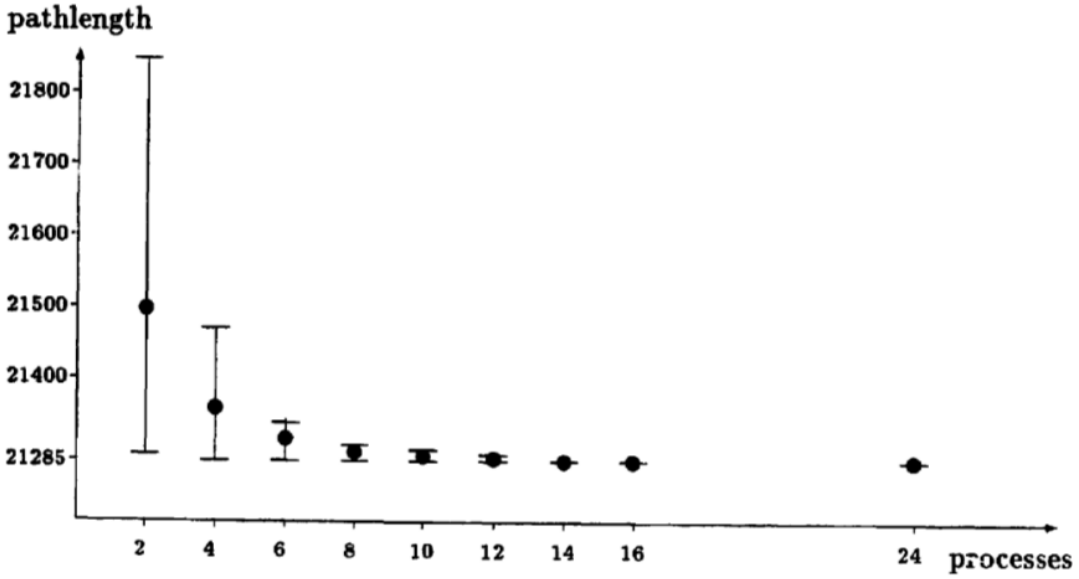

Figure 2 Quality of solution vs. processor

In numbers, with 2 individuals the algorithm found the global optimum in no case, with 4 individuals in 6 cases out of 25 runs, with 12 individuals it was found in 24 cases and with more than 14 in every run. By the way, the maximum, minimum and average of 25 runs made for a different number of processes by Mühlenbeiu, and the others.

The evolution of a good solution is applied with using Genetic Algorithm by Grötschels 442 nodes problem in the light of parallel computing. The first point to Figure 3 in the graphic is the best result found during the competition start phase of the algorithm. Moreover, until with 51.21 the best solution was found with using the cooperation method, the length of the path drops down first quickly and then slowly.

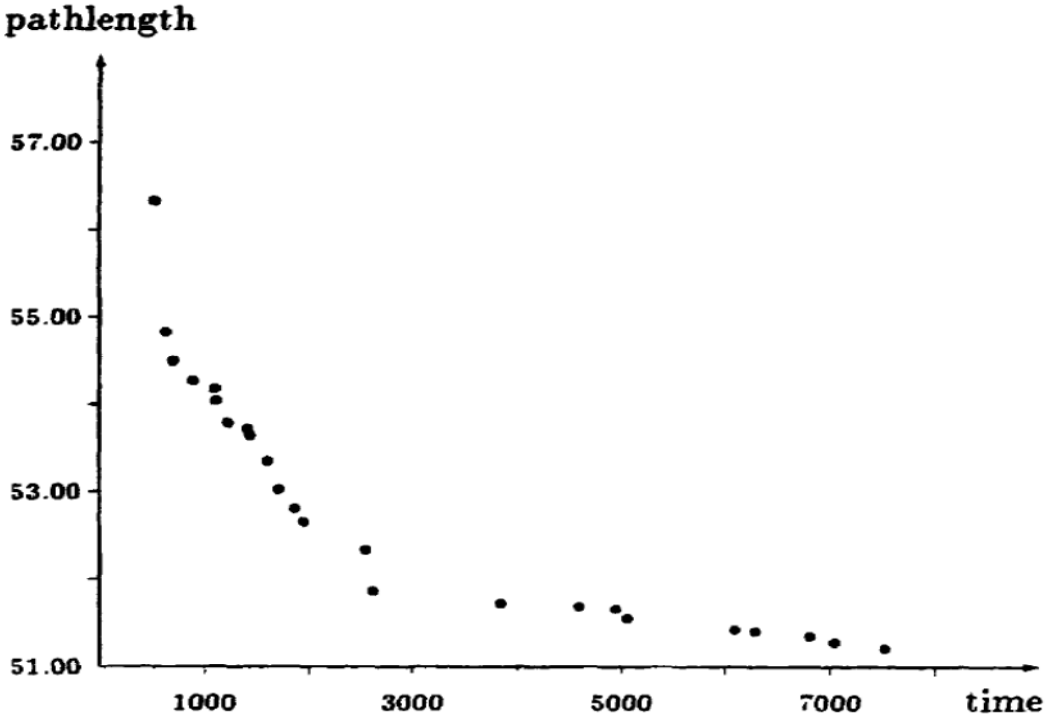

Figure 3 Evolution of the best path for problem size 442

The other evolution method is applied with using the heuristics method of simulated annealing and the next best result of 52.42 can be found by Rossier and the other.

The comparison charts of Genetic Algorithm (Figure 4.a) and Simulated Annealing (Figure 4.b) is shown following figure.

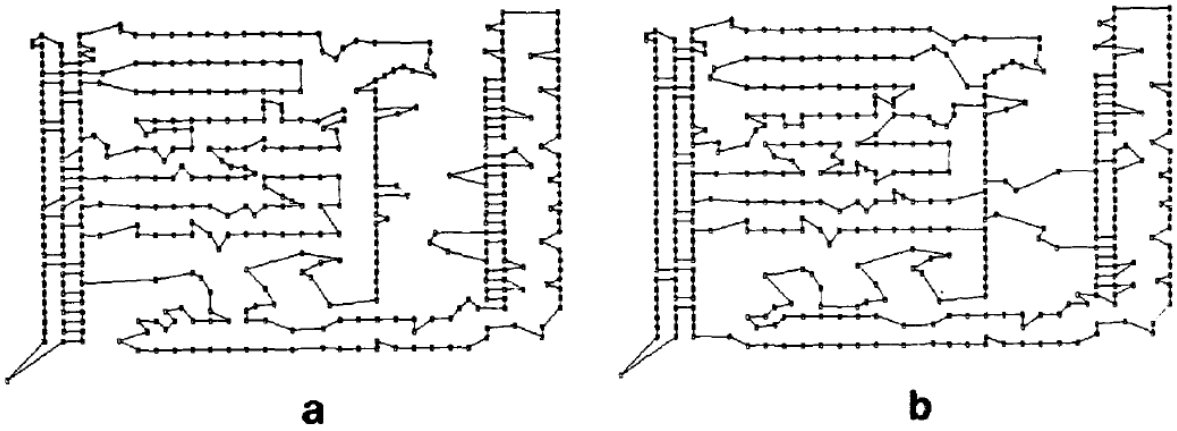

Figure 4 Genetic Algorithm result a=51,21 and Simulated Annealing result b=52,42

Why the genetic algorithm is best suited for running in parallel becomes clearer when observing the behavior in connection with the number of processes.

2.4.2. Speedup for Solution of TSP

When the number of processor is one, TSP is solving but the speedup of the solution is changing with the number of nodes. Speedup is computed as following that is studied by Müblehbein and the others.

Speedup **=** *The computing time of N processes on a single processor / The computing time on a N processor system*

The computation times for few processors are not comparable with the computation time for more processors, because the quality of the solution is different. Moreover, the minimum number of processors needed depends on the problem size to receive a satisfying quality. By the way, parallelization is only effective if the cost of the evaluation of an element is high (i.e., the computation is expensive and/or there are numerous and difficult constraints to evaluate), which is not the case for the TSP by Randall.

3. CONCLUSION

The evolution algorithms are applied for TSP in combinatorial optimization. Genetic Algorithm and Simulated Annealing heuristics are used for applying and comparing evolution algorithm. After using parallel computing application for Genetic Algorithm, the best solution is found with decreasing computation time.

REFERENCES

D. Vrsalovic, Z. Segall, D. Siewurek and M.Rnssinovich, MPC: Multiprocessor C, Internal Report, Carnegie-Mellon University, 1987.

G. Pataki, Teaching Integer Programming Formulations Using the Travelling Salesman Problem, *2003 Society for Industrial and Applied Mathematics*, Vol. 45,No . 1,pp . 116–123.

G. Lei-fu, and D. Wei, A parallel variable neighborhood search for the Travelling salesman Problem, Institute of Mathematics and Systems Science College of Science, China.

H. Mühlenbeia, M. Gorges-Schleuter and O. Kramer, New solutions to the mapping problem of parallel systems --The evolution approach, *Parallel Comput.* 4 (1987) 269-279.

M. Randall and A. Lewis, A parallel implementation of Ant Colony Optimization, Journal of Parallel and Distributed Computing 62, 1421–1432 (2002).

P. Krolak and W. Felts, A man-machine approach toward solving the TSP, *Comm. ACM* 14 (5) (1971) 327-334.

Y. Rossier, M. Troyan and T.M. Liebling, Probabilistic exchange algorithms and Euclidean travelling salesman problems, *OR* Spe/mum 8 (1986) 151-164.

Z. Segall and L. Rudolph, PIE: A programming and instrumentation environment for parallel processing, *IEEE Software* (198~).

www.wikipedia.org.tr