Parallelizing Genetic Algorithm for Estimating Optimal Hamilton Cycle in Random Graphs using OpenMp

Abstract

This Travelling Salesperson Problem being a classic combinatorial optimization problem is an interesting but a challenging problem to be solved. It falls under the class of NP- hard problems and becomes non-solvable for large data sets by traditional methods like integer linear programming and branch and bound method, being the earlier popular approaches. Genetic Algorithm based solutions emerged as the most popular tool to solve the problem which is a heuristic mechanism to find the closest approximate solution to the problem. In this paper, we approach Travelling Salesman Problem using GA approach, a natural selection technique which repeatedly modifies a population of individual solutions, with the added power of modern computing systems. Here we come up with a parallel version of GA for multicore via using OpenMP.

Introduction

Travelling Salesperson Problem is a classic NP-hard combinatorial optimization problem to find the shortest route for an N-vertices graph where each vertex gets visited exactly once. With one of the most interesting problems in the field of graph theory to be solved, TSP has found its implication in many real world problems like crew scheduling, vehicle routing, connection decision problem for a PCB. As a result, the problem has attracted many researchers and scholars to investigate upon it and find its applicability to solve a real world problem in a most comprehensive way and manner. The problem has a number of problem solving approaches.

Hamiltonian Cycle

A hamiltonian cycle which can also be called a hamiltonian circuit is a closed loop through a graph that visits each and every node in the graph exactly once. The problem of finding a hamiltonian cycle is NP complete. The only way to find a hamiltonian cycle in a graph is to do a complete search.

Travelling Salesperson Problem

Finding the shortest hamiltonian cycle in a graph is called the Travelling Salesperson problem. The brute force solution to this problem is as follows.

1. Find all the hamiltonian cycles in a graph - NP hard problem

2. Find the length of each hamiltonian cycle among the cycles found in 1.
3. Select the cycle with the shortest path.

While the brute force algorithm is optimal and will always yield the correct answer, it is not an efficient algorithm. Consider the case of an air travel graph of some cities in an airport. Checking each and every possible cycle can turn out to be a time consuming task especially if the graph is a complete graph or if the graph has a huge number of nodes or both! Unfortunately, there is no efficient and optimal algorithm to solve the Travelling Salesperson Problem. Since it is not a practical method to use the brute force algorithm to solve the Travelling Salesperson Problem, we try to solve the problem using a type of heuristic algorithm. Heuristic algorithms are algorithms that are efficient algorithms that give good approximate solutions to problems. The heuristic algorithm used in our project is a search heuristic inspired by Charles Darwin's theory of natural evolution called genetic algorithm.

Genetics Algorithm

Genetic Algorithm is a biological method which helps evolution. It is used to solve constrained and unconstrained problems of optimization. They belong to the class of Evolutionary Algorithms. The concepts of natural evolution like selection, mutation and crossover are used for problem solving. Optimal solution is selected after successive generations. Therefore, it is highly suited for problems involving optimization and search-based solutions to find the approximate solution. The applications of GA are computer simulation where subjects undergo evolution by repeated recombination and mutation, thus generating optimal or near-optimal solutions. GA speaks to an iterative strategy. Each era is synonymous to an iterative cycle. A common scope of eras for a basic GA can extend from 50 to more than 500. The whole arrangement of eras is known as a run. Utilising GA as a seeking heuristic, the process will be started by an arrangement of populace or population.

Genetics Jargon :

1. **Gene** : a city (or a graph node)
2. **Chromosome** : A single possible hamilton cycle
3. **Population** : Collection of Chromosomes
4. **Parents** : Two routes that are combined to create a new route
5. **Mating Pool** : a collection of parents that are used to create our next population
6. **Fitness** : a function that tell us how good each route is (in out cas how short it is)
7. **Mutation** : way to introduce variation in out population by randomly swapping two cities in a route

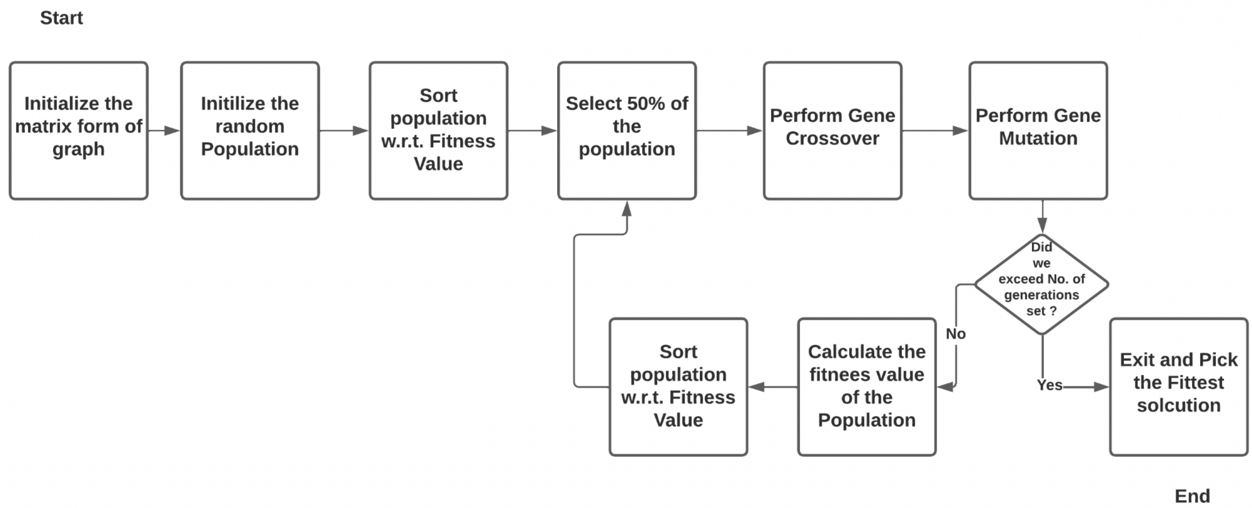


Figure 1 : Showing the Computation Graph of Genetics Algorithm

The whole procedure for working up the arrangement can be abridged as:

1) Randomly initialise populations p

2) Determine fitness of population

3) Until convergence repeat:

a) Select parents from population

b) Crossover and generate new population

c) Perform mutation on new population

d) Calculate fitness for new population

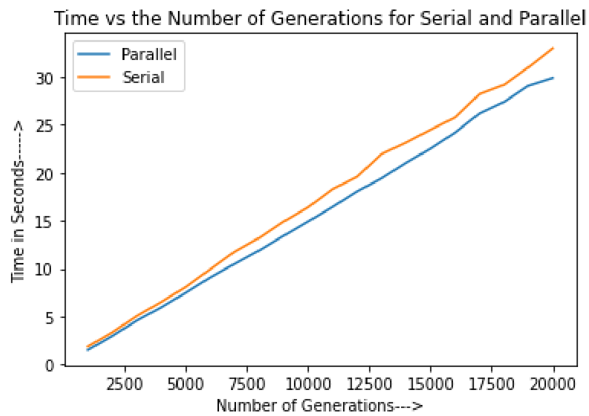
Proposed Methodology

In order to parallelize the Genetics Algorithm, we have parallelized each of its steps. Since in figure 1 you can see that the majority of steps in Genetics algorithms are co-dependant on its previous step, hence parallely running each of the steps will not be possible.. So we have to parallelize the components of each of its steps in order to parallelize the code as a whole.

In figure 2, the steps coloured yellow were the most important steps. When these steps were parallelized the Computation of Genetics Algorithm became faster.

Result Analysis

The code was run for 2500 to 2000 generations. Both Serial and OpenMP code was run for this generation interval. Graph 1 shows the graph for the time taken by both



Graph 1 : Showing the Time Complexity of Parallel and Serial code

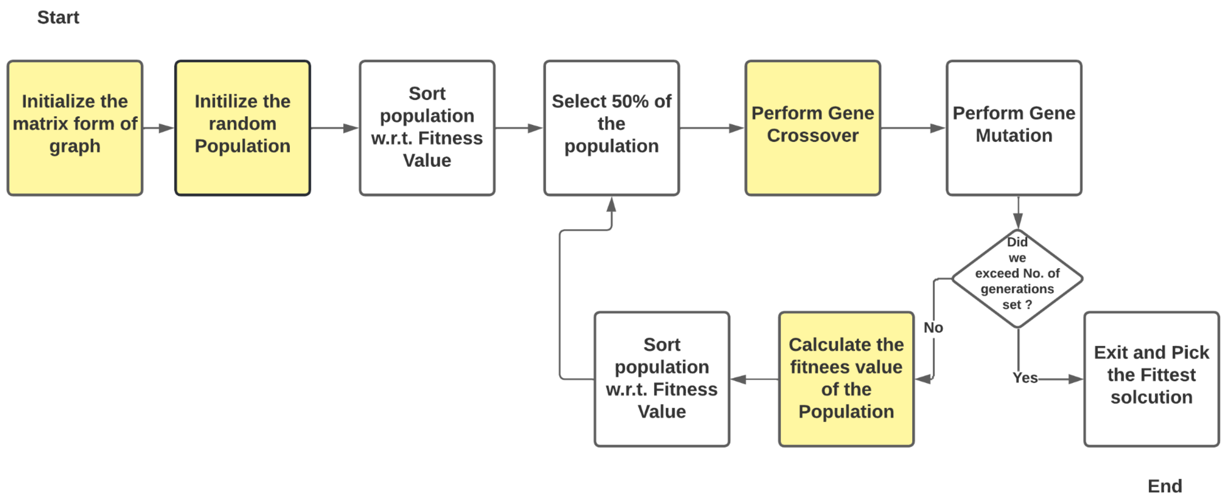


Figure 2: Showing the Parallelize Computation graph of Genetics Algorithm

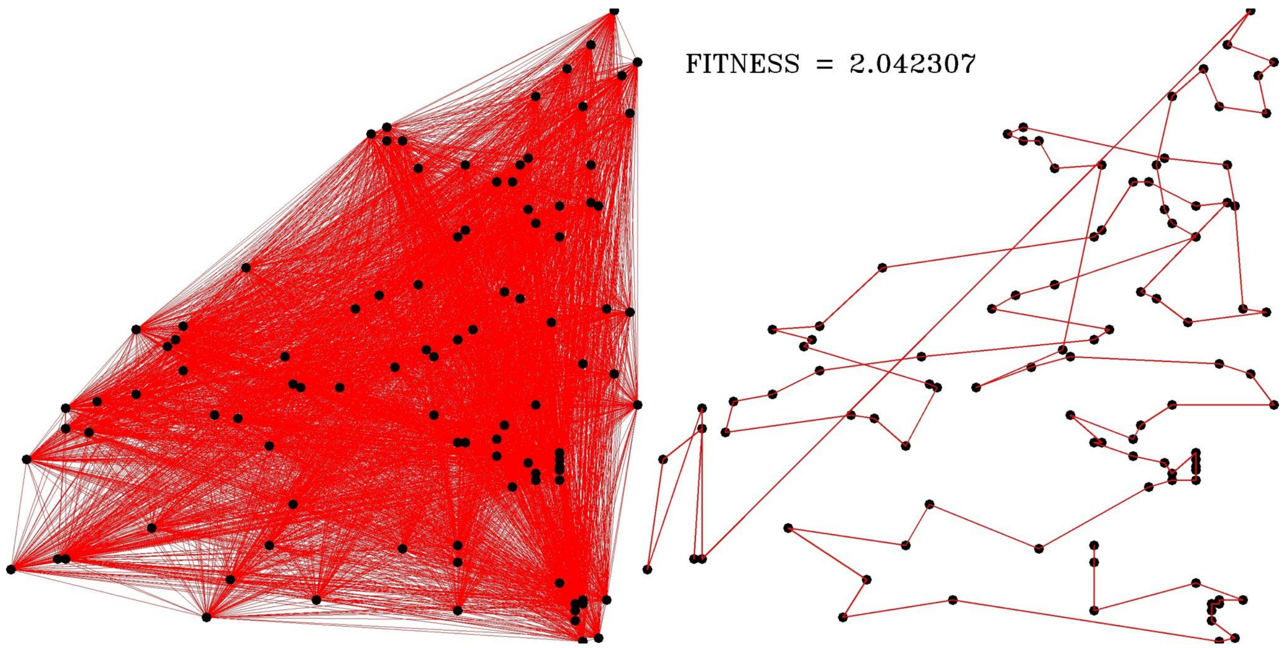
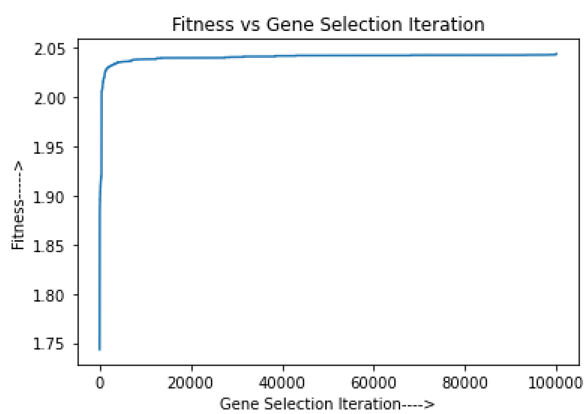


Figure 3: Showing the Random Graph Initialised and the Final Estimated Optimal Hamilton Cycle

Serial and Parallel code of Genetics Algorithm written in C.

Graph 2: Shows the Fitness vs Iteration graph For the Genetics Algorithm. As you can see



Graph 2 : Showing the Fitness Value vs Generation

that almost after 20000th iteration the Fitness function kind of Stabilised and grew slowly.

Conclusion

In this paper, we have discussed genetic algorithms as the heuristic approach to TSP solving, being the most popular among others. Solving large problem instances for applications like vehicle routing problem, DNA sequencing, routing based deterministic decisions in mesh wireless networks etc. in real time has been possible through these heuristics. But these heuristics again have computational limitations and processing time is significantly high when the algorithm is executed over a single-core machine. So the paper comes with the execution of the proposed parallel model over OpenMP for CPU based optimization to achieve high performance efficiency. The experimental results show that there is not a very significant cut-down in the execution time of the parallel algorithm.