A Parallelized Solution for the Traveling Salesman Problem using Genetic Algorithms

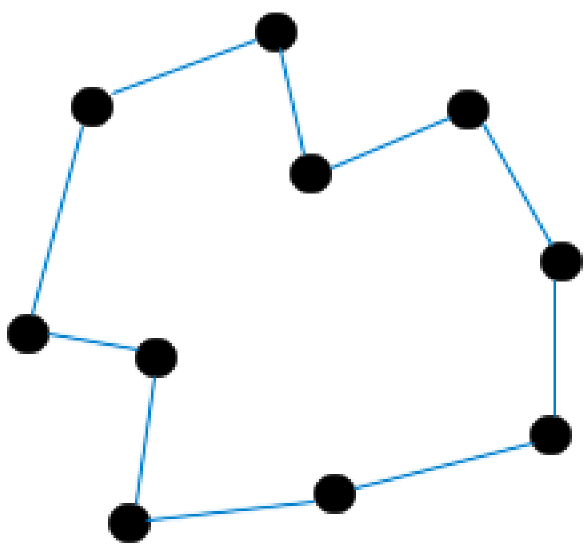
Sagar Keer
CSE 633 Fall 2010
Advisor: Dr. Russ Miller

University at Buffalo The State University of New York

Introduction

- **Traveling Salesman Problem**

  Given a set of ‘n’ cities, we are to find the shortest closed non-looping path that covers all the cities. Thus, no city may be visited more than once



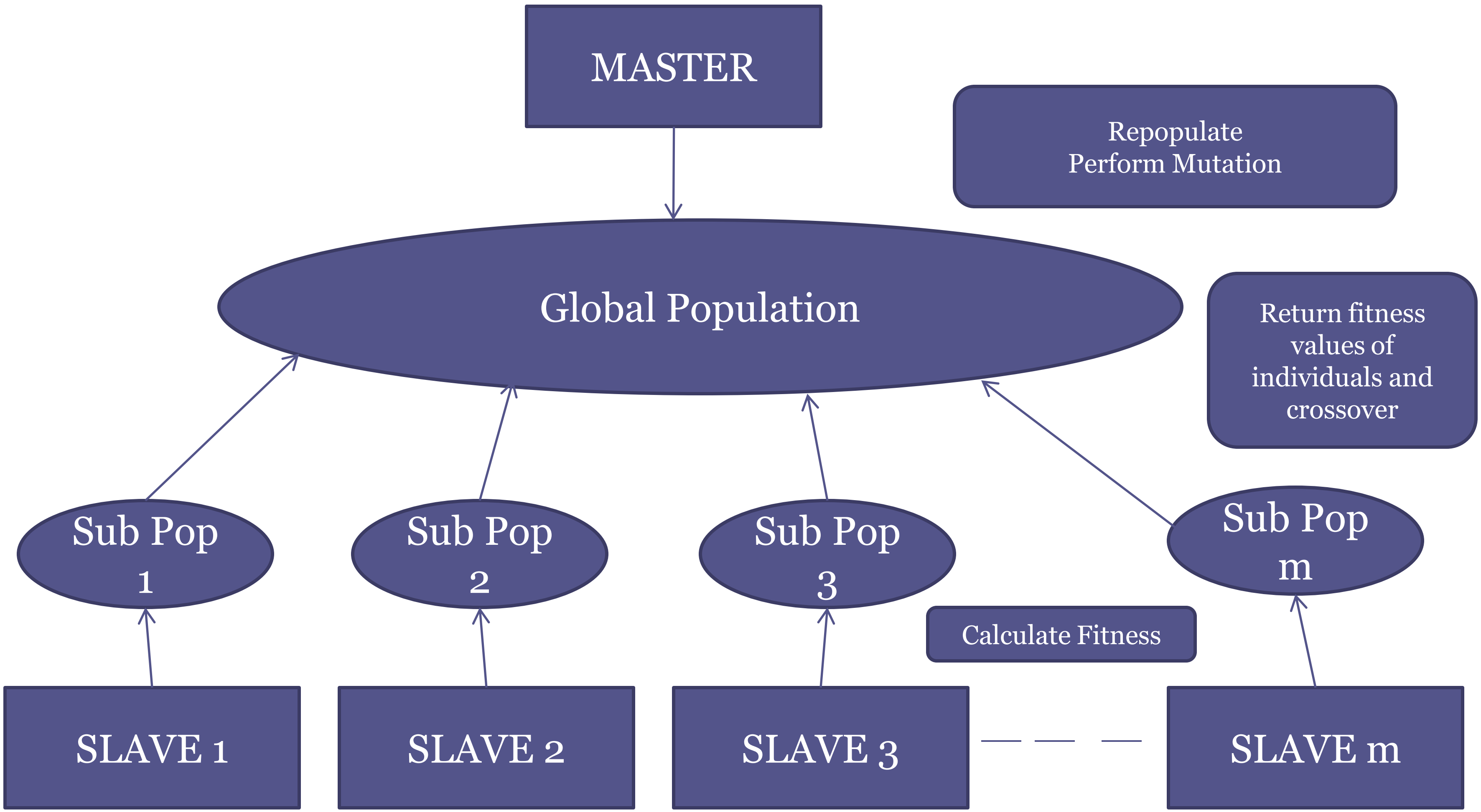
Introduction

- **Genetic Algorithms**
- Belong to class of Evolutionary Algorithms
- Apply concepts inspired by natural evolution such as selection, mutation and crossover for problem solving
- After successive generations, we converge to an optimal solution.
- Highly suited for problems involving optimization and search-based solutions

Solution Basics

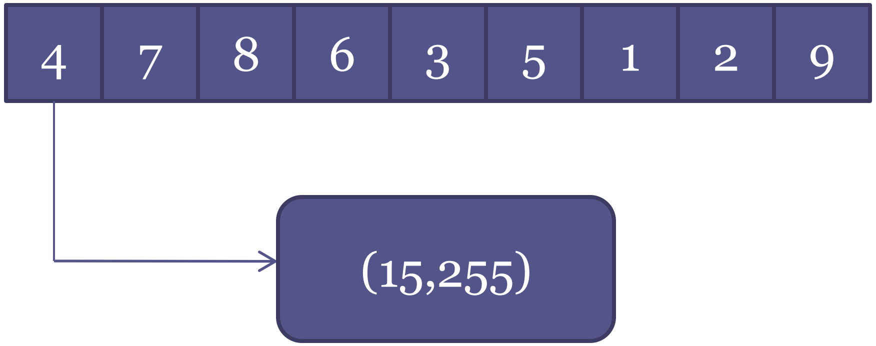
- **Applying Genetic Algorithm to TSP**
- Individuals → Closed non-looping paths across all cities
- Initial Population → Set of randomly selected individuals, ie. Set of randomly generated paths
- Fitness Function → Derived from the total distance of a given path
- Selection → Select the fittest individuals
- Breeding → Perform cross-over between the fittest individuals to create new individuals to replace the weakest ones. Also perform mutation.

Parallelization



Details

- **Individuals**



- Each individual is a closed, non-looping path with the cities represented by numeric identifiers
- Each city is a point on the first-quadrant co-ordinate system

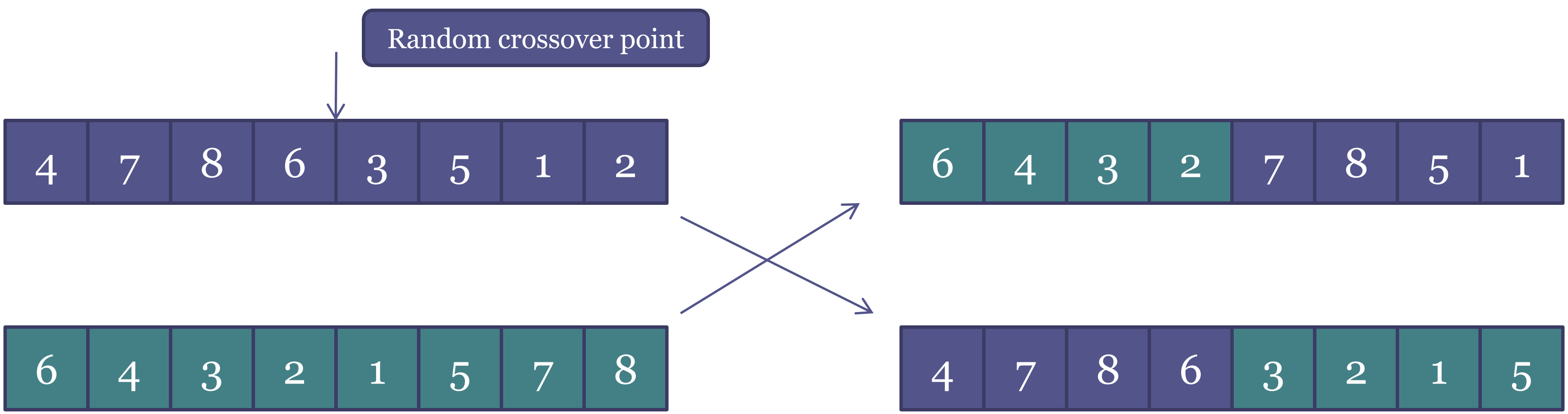
Details

- **Fitness Function f**
- Calculate the total Euclidean distance for each path
- For an individual i, fitness function is given as

  f(i) = 1/Di   (Changed from Dmax – Di)
- **Selection**
- Using Tournament Selection: Run a 'tournament' among 'k' randomly selected individuals to find the fittest individuals
- Use successive tournaments to pair the fittest individuals

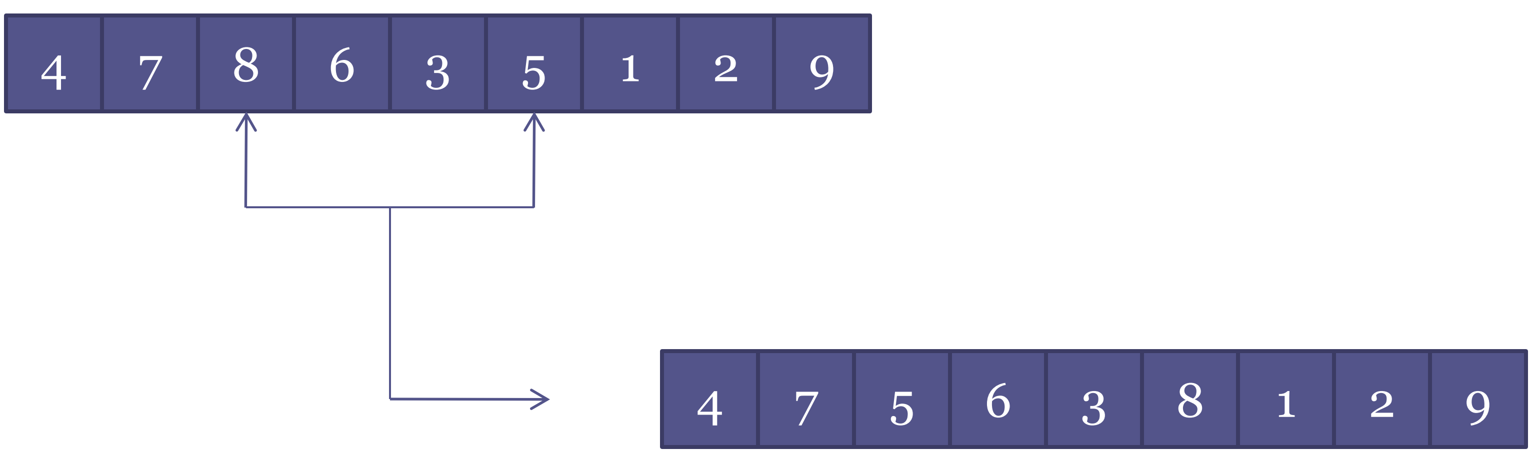
Details

- **Breeding**
- Through a single-point crossover, the two selected individuals will breed and create two children



Details

- **Breeding**
- Through swap-based Mutation, by randomly swapping two cities with each other



Algorithm

Initialize the global population with random individuals
While iteration_count != max_iterations
    Distribute sub-populations
    Calculate fitness for each sub-population
    Perform selection
    Perform crossover
    Repopulate global population
    Perform random mutation
End of Loop
    Find fittest individual & report as solution

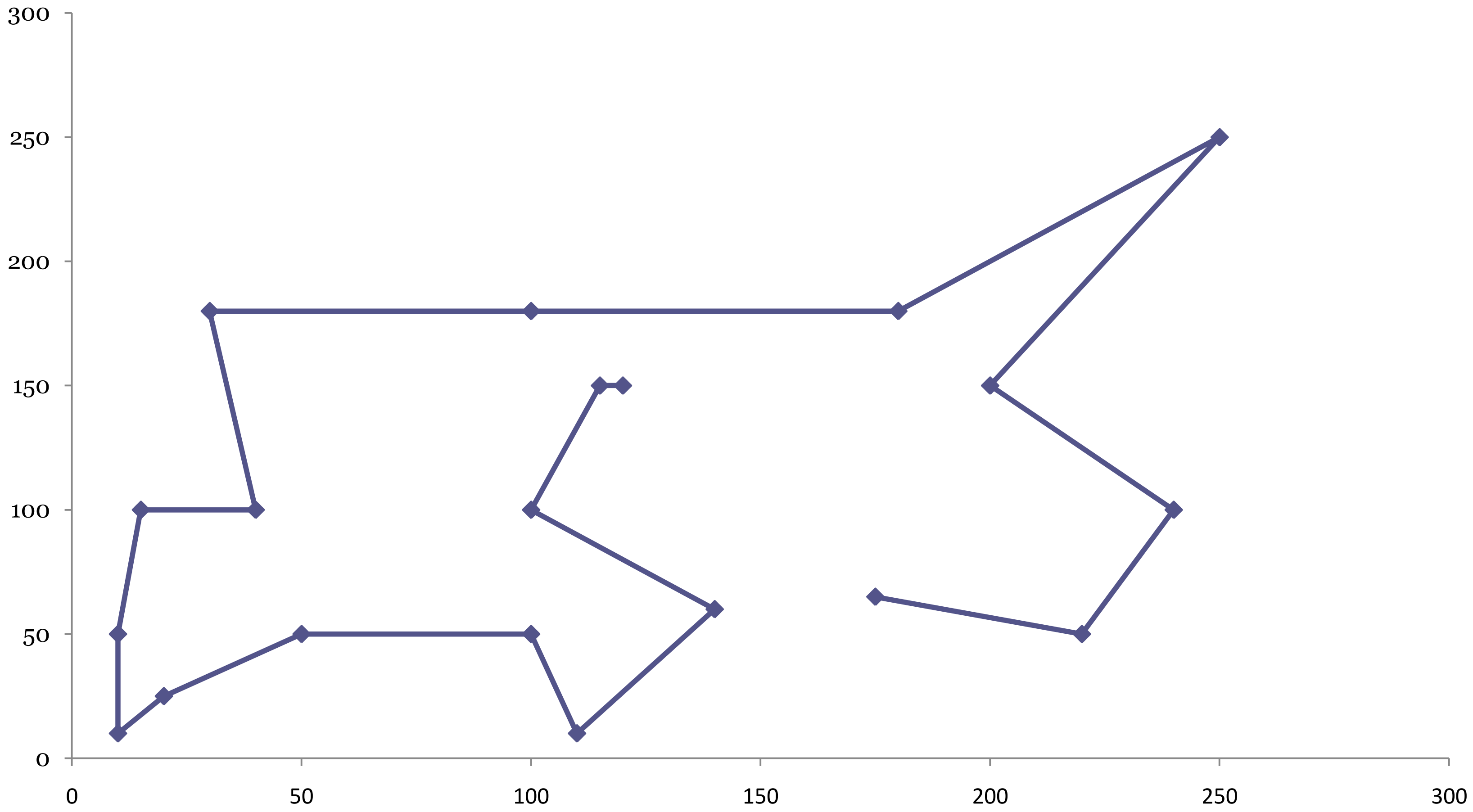
Implementation Specifics

- Implemented in C with MPI
- Used the Edge Cluster
- Working with a set of 10-30 cities and initial population ranging from 700 to 46000
- Used 10^5 iterations
- Crossover breeding occurrence – 80-90%
- Mutation Occurrence – 12-13%
- Worked using 8,16 and 32 processors

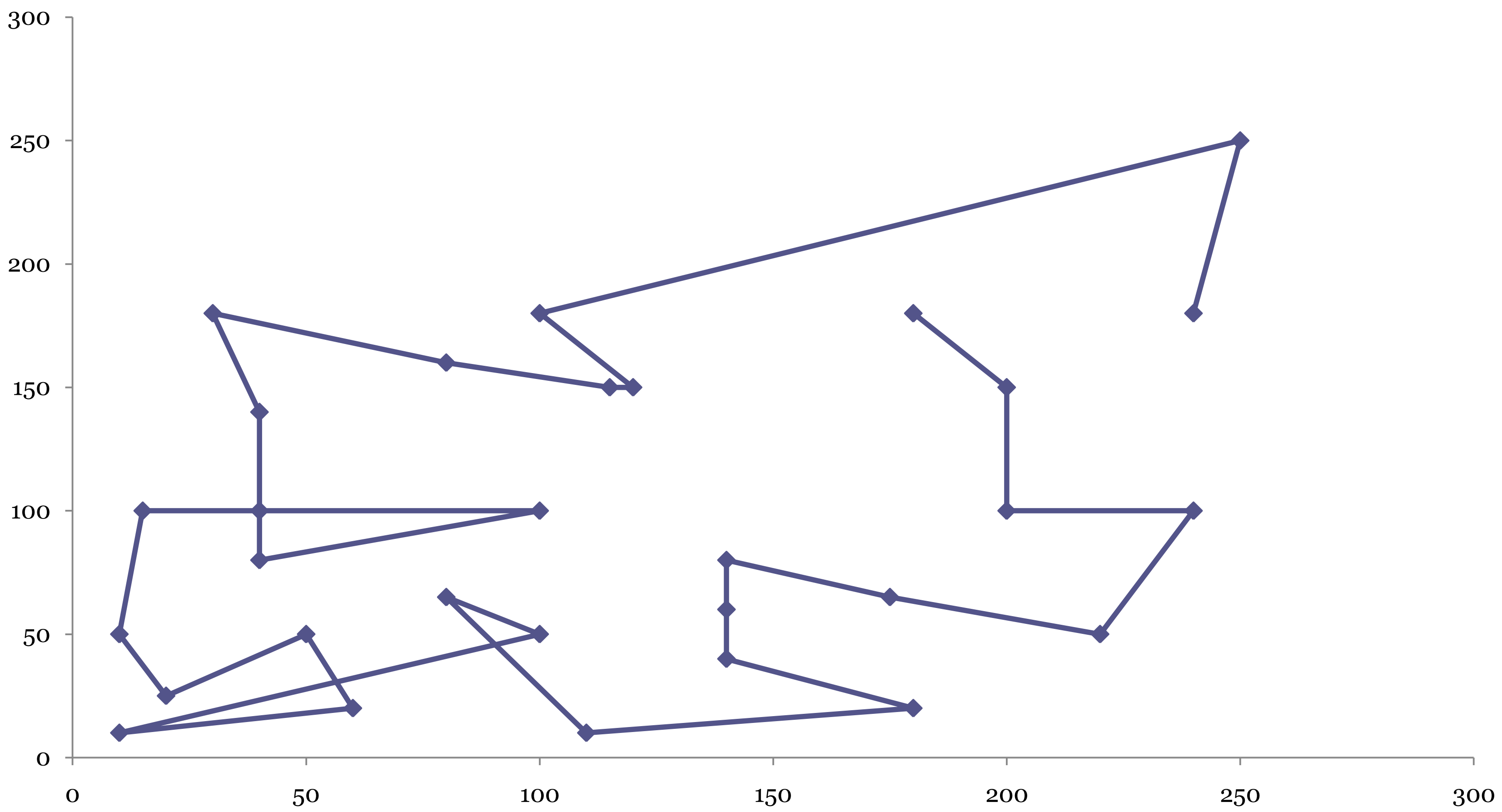
Implementation Specifics

- Used MPI Send/Recv calls to pass sub-populations between the master and all slave processors
- Used MPI_Create_struct to create MPI Struct datatype for passing sub-populations
- Completely randomized initial population
- Probability of crossover based on tournament selection of individuals
- Probability of mutation fixed deterministically

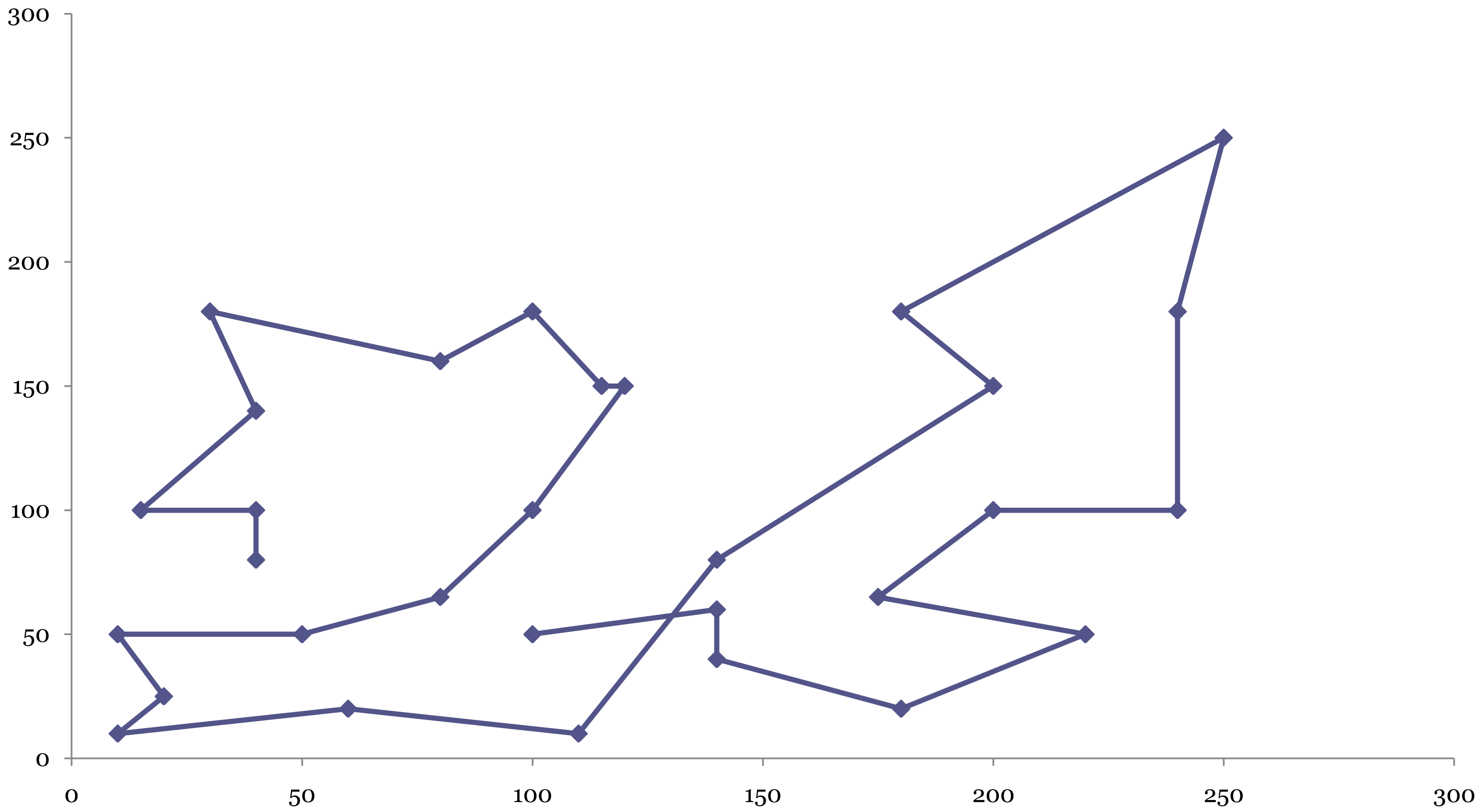
Solution for a 20 City Problem



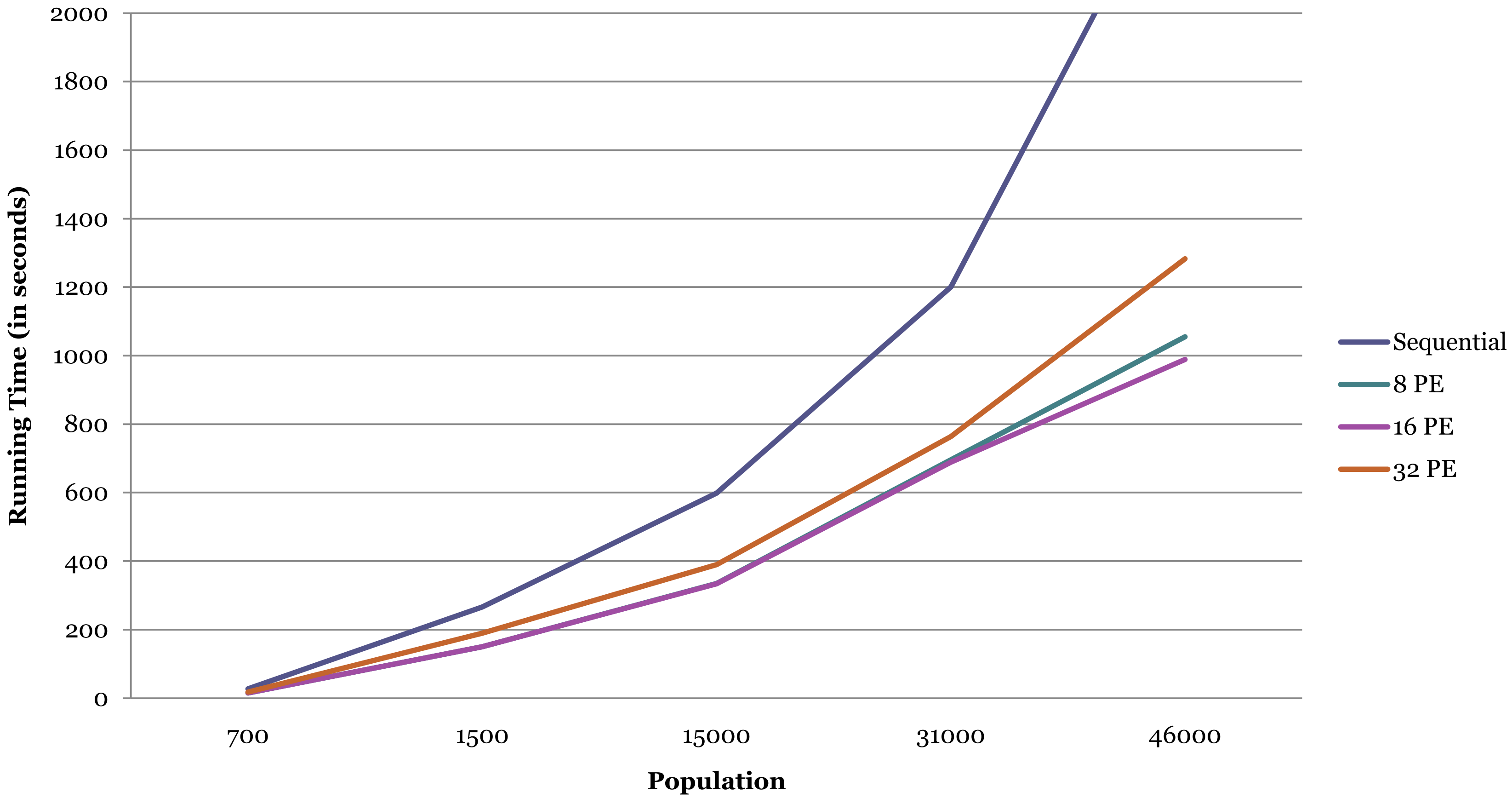
16 PE Solution for a 30 City Problem



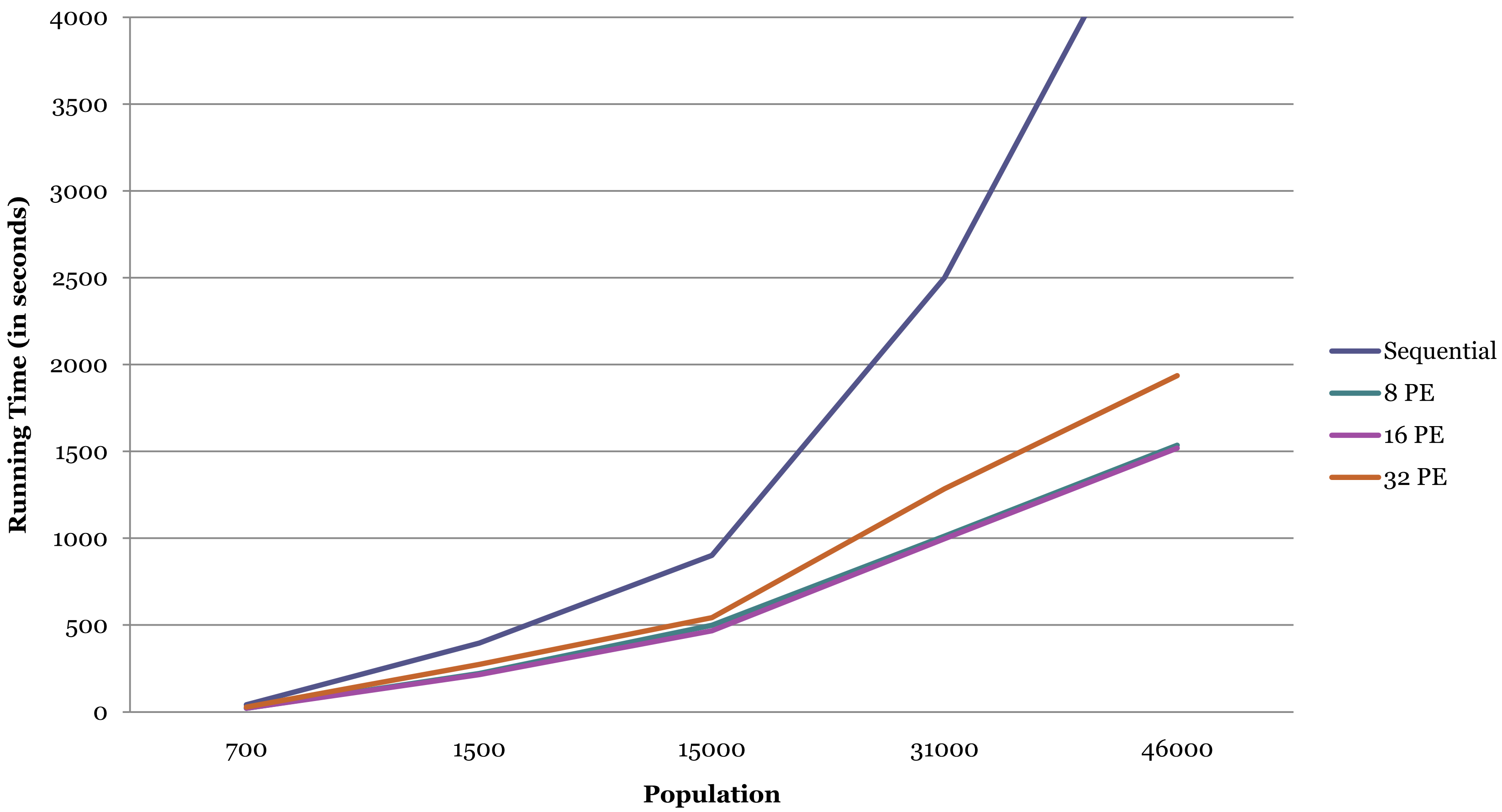
32 PE Solution for a 30 City Problem



Performance for 20 Cities



Performance for 30 Cities



Results

- The number of iterations were sufficient for proper convergence
- However, increasing number of processors did not result in more speedup
- The sequential algorithm tends to converge early
- In parallel, distributed sub-populations allowed selection of less fitter individuals, thus allowing better range before converging to a solution

Future Work

- Refine the parallel algorithm further to improve speedup
- Increase the data set and population size
- More manipulations of existing parameters and also add few more deterministic parameters to improve solutions
- Try out a different approach like using CUDA on the MAGIC Cluster

References

- Borovska, Plamenka. ‘Solving the Travelling Salesman Problem in Parallel by Genetic Algorithm on Multicomputer Cluster’. CompSysTech’06
- Al-Dulaimi, Buthainah Fahran and Ali, Hamza. 'Enhanced Traveling Salesman Problem Solving by Genetic Algorithm Technique(TSPGA)'. World Academy of Science, Engineering and Technology 38 2008
- Heavner, Matt. ‘Massively Parallel Travelling Salesman Genetic Algorithm’. http://www.cse.buffalo.edu/faculty/miller/Courses/CSE710/710mheavnerTSP.pdf (Diagrams)

Thank You

Questions?