Project 3

Traveling Salesperson Problem

Jose Eduardo Gamboa Barraza
Universidad Autonoma de Guadalajara

Guadalajara, Jalisco

The traveling salesperson problem is a well known amongst the proramming community, and the solution has been an ever improving method to do so, which is why it is so important, in this case we will look through different methods to solve this problem, comparing different situations. With cities ranging from 20 to 200, and alternating between a basic method and the taboo search.

I. Introduction

The Traveling Salesperson Problem(TSP) consists of, for a list of cities, find the best route, given a starting city, to travel through each city and come back to the start with the lowest cost possible, cost being distance, time, price, etc.

Besides the TSP, we will also solve a Shortest Path Algorithm(SPA), given a start city and a destiny, find the cheapest path, and the Minimum Spanning Tree(MST), to find a way to connect all of the cities or nodes, with the cheapest cost.

There will be 4 different situations, the first one is a case where we have 20 cities, where a road might go from place A to B, but not to C. For the other 3, Krolak-A problems 100,150 and 200 hundred will be used, in this case, all cities are connected with each other, and the distance(cost) is set using x,y coordinates.

II. Description

For the first problem(20 cities), we used some of the capital cities from Mexico, some cities connect directly to 1, 2 and up to 6 other cities, as shown in the map, for this reason, the actual roads are the ones that tell whether you can get from one city to another, or if you need to go to another city first.

The problems to solve in this situation will be a SPA, find the cheapest route to go from one city to another one, find the list's MST, cheapest way to connect all of the cities, and the TSP, a route to visit all cities once and return to starting point.

For all of these problems, the cost will be considered as the distance from point A to B, using only terrestrial roads, meaning no air or water distance is considered.

The other 3 problems as previously said, are Krolak-A-100, Krolak-A-150 and Krolak-A-200, and in this situation only TSP will be solved, since a SPA would be pointless, as you can go from any given point to another directly.

III. Solution

PSA has different methods that follow different steps, in this case, we took Dijkstra's algorithm, this method first set cost to all neighbors to a really high number, usually infinity, and updates the cost to the neighbor if the cost is higher to what was previously recorded, moving to the lowest cost's neighbor and repeating the process, now updating all of the unvisited neighbors.

MST's algorithm used, was Prim's algorithm, first we take an arbitrary city, call node, and look for the closest neighbor, and connect it to the first node, and continue searching for the closest directly connected node in the Tree that is being created until all of the nodes are inside the tree.

For the TSP, we used three different methods, for the 20 cities, first we create the MST using Prim's algorithm and then we do a preorder traverse, to create the route, since an MST is a Tree, we do this preorder traverse by starting from up to down, and left to right child, a demonstration of this is shown later.

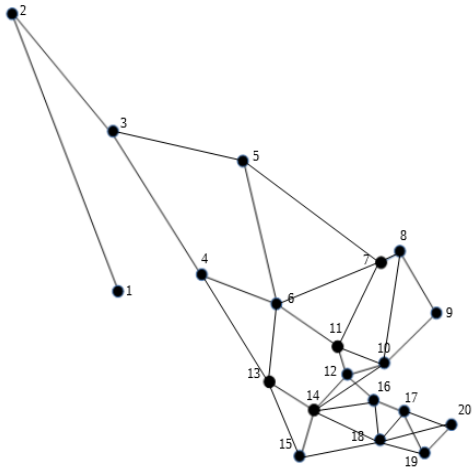
In Krolak's problems, we used a nearest neighbor search, which consists of, as the name points, creating the route by looking at the closest neighbor, and adding the node to the route. The problem with this solution is that you only get a local solution and not a global solution.

Finally, Taboo search is an algorithm that could be seen as an improved local solution, to do this, first a random route is created, in this case, we used the previous algorithm(nearest neighbor search), and then swap two nodes in the route and calculate the new cost, this is done with all the possible swaps

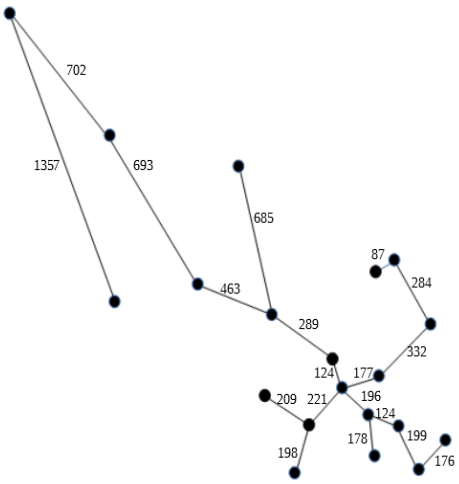
and choose the best one(the lowest cost route), and keep doing this for an arbitrary number of cycles, to keep from the same two nodes swapping position, a short term and a long term memory is used, the short term memory keeps track of the recently nodes swapped, and add a taboo time, which is the time the same nodes can be swapped again, the long term memory saves the frequency in which two nodes are swapped, and penalize the ones that are repeated too many times.

IV. Results

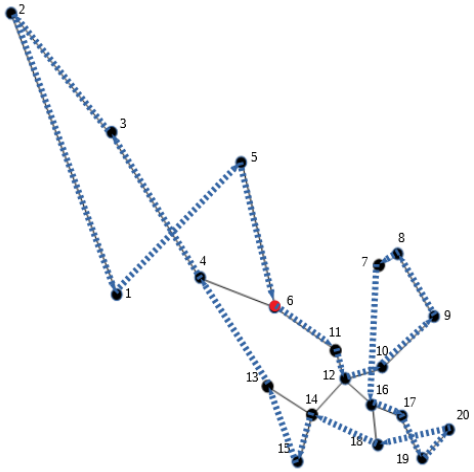
First we have the results for the 20 cities, the next graph shows all of the connections for each node(city).



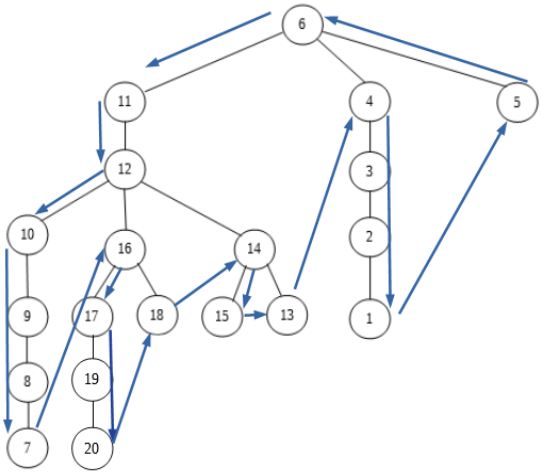
In this case, for the MST, we got the next graph, which show the cost between each node.



For TSP as said before, we used the previous MST and did a preorder traversal, the next image shows the result with the starting city being Durango(node 6).



The next image shows how the preorder traversal functions following the first MST to create the TSP route. Left children are the nodes with the lower cost(distance).



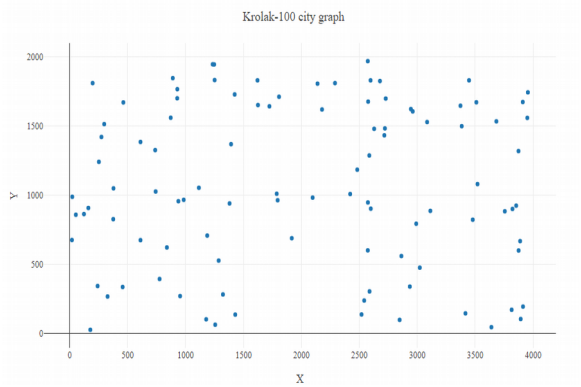
Times for previous results are shown in the next table.

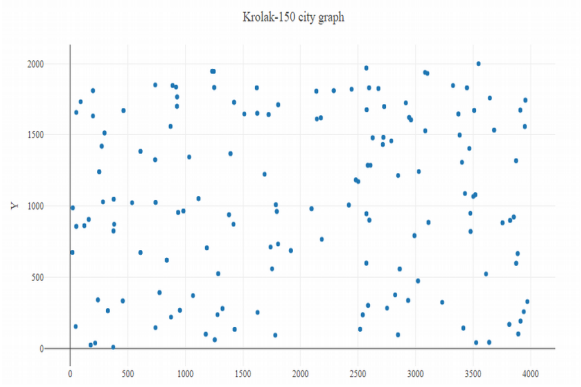
	MST	SPA	TSP
TIME	224.82 us	284.09 us	657.88 us

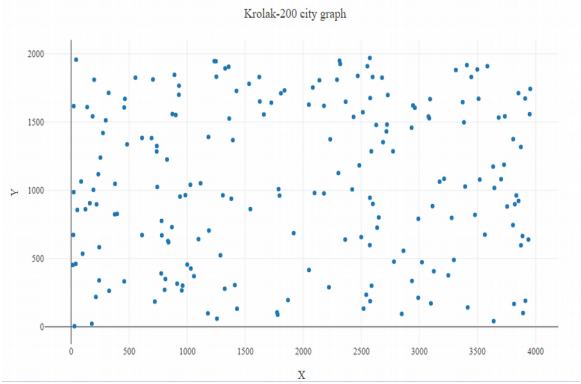
Table 1. Execution time for 20 cities

The cost for this route can not be determined, because there is no actual distance between node 1 and node 5, the TSP says that you can't visit a node twice, therefore there is no actual way to return from node 1 to any other node.

The next results are obtained from the Krolak's problems, first the graphs for each problem, showing the node in a plot.







For this problems, first the previous algorithm was used(algorithm used in 20 cities problem), then compared the results with the second algorithm(nearest neighbor search) and finally to Taboo search.

The table comparing results is shown next.

	Krolak-100	MST	Nearest Neighbor Search	Taboo Search
Krolak-100	**TIME**	46.87 ms	27.96 ms	140 s
	COST	27764 units	27772 units	24109 units
Krolak-150	**TIME**	115.99 ms	74.38 ms	836 s
	COST	34753 units	34063 units	31510 units
Krolak-200	**TIME**	254.65 ms	123.09 ms	1242 s
	COST	41642 units	35715 units	33102 units

Table 2. Comparison between algorithms for cost and time.

The results obtained in taboo search vary depending on different factors, first the total iterations, the penalization, maximum frequency and the taboo time.

The next table shows the different results obtained with different parameters, tested in the Krolak-A 100 problem.

Iterations	Penalization	Maximum frequency	Taboo time	Cost
50	350	4	3	24132
100	350	7	2	24109
200	500	5	5	24216
300	400	5	10	24170
500	300	5	8	24157

Table 3. Parameters comparison for taboo search.