# SELF ORGANIZED GRAPH PARTITIONING APPROACH FOR TRAVELING ROUTE RECOMMENTATION

M.MUNIRATHINAM M.Phil., RESEARCH SCHOLAR DEPARTMENT OF COMPUTER SCIENCE, PADMAVANI ARTS AND SCIENCE COLLEGE FOR WOMEN, SALEM-11

MRS. D.M.CHITRA, M.Phil, ASSISTANT PROFESSOR,DEPARTMENT OF COMPUTER SCIENCE, PADMAVANI ARTS AND SCIENCE COLLEGE FOR WOMEN, SALEM-11.

**Abstract:**

Traveling salesman problem becomes more dominant solution in many areas of real world like scheduling. There many solutions exists but lacks with time and repeated visit. We propose a new technique to find the next location to visit, using earlier records with the help of mobile agents. Based on the earlier records of the customer availability the mobile agent choose the next location to visit in order to avoid the person unavailability conditions. We use multiple agents to reduce overall traverse time. Each agent choose the location to visit and once visit it set the flag as visited to avoid visited by another agent. At the time of scheduling the mobile agent accounts the visit flag and person earlier records. For example in an service oriented customer handling process, the service agent has to visit more locations to provide service to the customers. In that case the service agent can schedule and choose its next location to visit using our algorithm. While using our methodology and using multiple agents in that customer handling scenario , it reduces the overall time to visit all locations. Our methodology reduces the traverse time by using multiple agents.

**Index Terms:** Scheduling , ant colony system, traveling salesman problem.

## Introduction:

Lets think you are a sales executive come collection executive, you will be roaming all over the country for collection of funds from various distributor of your product. Your visit has to be scheduled according to the place visited and unvisited, but has to be visit only once in a month. How would you determine the order of visit, because various consumers will be available at various location of the country and at various times? If we really consider about the distance of the locations we can construct a weighted graph and find the best shortest path from the graph. If we interested in time factor we construct a weighted graph as time between nodes and choose a best path accordingly. Our interest is to reduce the overall traverse time which takes overall to visit all the locations. The real situation here is to reduce the polynomial traverse time in an efficient way, so that there must be a specific solution to the TSP in all the ways. In the traveling sales man problem once the route has selected then the start node is the end node of the route and all the nodes has to be visited only once. Further, unless the graph can be redrawn so that the only vertices that are part of the graph are the
ones that must be visited, the TSP algorithm would not provide an appropriate solution for a delivery vehicle driver. The TSP at a specific snapshot in time may not be the solution determined by the ITS-TSP, which will change based on the current edge weights and the current location of the vehicle. As the entire set of intermediate nodes must be traversed (which will be a subset of the total set of nodes in the graph), local routing decisions must take into account the remaining nodes that must be traversed. To determine the efficiency of the final path, the optimal path with respect to the changing edge weights would need to be determined.

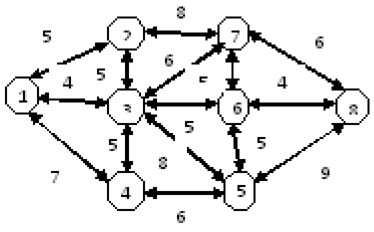
As we cannot determine future roadway conditions to prove the optimality of the ITS-TSP, we will compare the route determined from the ITS-TSP algorithm with the optimal route as determined using brute force on the historical edge weights during the time the route was being traversed.

In this paper we analyze the live roadway and modeled as a graph with edge weights representing both distance and time. Unlike earlier solutions we used self scheduling mobile agents to schedule and choose their next location to visit. We used multiple agents from different locations in order to reduce the polynomial traverse time.

## Background:

Shortest path graph algorithms have been studied for many years. Single source shortest path algorithms are used to determine the shortest path between one node and every other node in a graph. Dijkstra’s [7] and Bellman-Ford’s [8, 9] algorithms are popular single source shortest path
algorithms. Using these algorithms, the all pairs shortest path algorithms compute the shortest path between every pair of

nodes in a graph. Johnson's [10] and Floyd- Warshall's [11] algorithms are popular all pairs shortest path algorithms. Although these algorithms could all be applied to a graph with constantly-changing edge weights, they were originally designed assuming the weights of the edges were rather static. If the weight of an edge changes, the algorithm will have to be re-executed to determine if any of the shortest paths change.



The graph shows that the distance between locations, based on the locations the time taken to visit from location to location can be calculated.

Because of this inefficiency, dynamic fastest path (DynFast) algorithms were created. In [13], dynamic fastest path algorithms in general are explained, and in [14] an algorithm for solving the dynamic all pairs shortest path problem is proposed. As was explained earlier, the TSP [12] is similar to the problem posed in this paper, with the difference that the TSP does not deal with changing edge weights or only visiting a subset of the nodes in the graph. The Dynamic Traveling Salesman Problem (DTSP) [4, 5] is a specialized case of the TSP that allows nodes to be inserted into the graph after the traversal has already begun. The Dynamic TSP with Time Windows is another variation on the TSP where the times at which the traveler can visit each node is defined by a specific time window [6]. Even though there are a number of special cases of the TSP, the TSP with a subset of intermediate cities and dynamic edge weights applied to a transportation network has not yet been studied with respect to ITS
architectures. with respect to ITS architectures. One additional variation of the TSP which seemed relevant to this study is the Probabilistic TSP [1]. Jaillet presented the Probabilistic TSP, in which a random subsetof cities is visited, but his solution used the solution to the traditional TSP. He then removed the nodes from the TSP solution that were not in the random subset to find the solution
to the Probabilistic TSP. We discovered that the optimal solution to the subset of cities did not have to coincide with the ordering of the cities in the solution to the traditional TSP, which ruled out Jaillet's work in our specific problem.

Focusing on ITS architectures, vehicle-to-vehicle (V2V) [15] and vehicle-to-infrastructure (V2I) [16] ITS architectures have been widely studied in literature. The two architectures were combined into the vehicle-to-vehicle to- infrastructure (V2V2I) moves from one location to other location then we can say that the particular location has been visited. After visited the multi architecture in [19], and vehicular ad-hoc networks (VANETs) are becoming increasingly more popular [17]. The means by which speed is determined from gathering data discretely at loop detectors is widely used by departments of transportation and was
originally proposed in [18]. For the majority of work done using speed or location data, the data was either gathered at discrete locations (from
devices such as loop detectors or video cameras) or the data was simulated to be continuous. More recent projects, such as MIT's CarTel [20] and UC Berkeley's Mobile Millennium [21] projects, are gathering data in a distributed manner from cellular phones, though very little data has
been published from these projects. Further, there are additional challenges of trying to maintain anonymity, as well as determining if the cellular device is actually located within a vehicle when it is transmitting the speed and
location. A company called Airsage [22] has attempted a similar project in the Washington DC area. At the University of Alaska, Anchorage, 50 vehicles have dedicated vehicle-tracking devices installed, so there is no
issue in determining which devices are communicating from vehicles. The devices maintain the privacy of the users because there is no identifying information relating the device to a vehicle or a driver.

There is no such system which can schedule themselves to choose the next location and independent. So that there is a need to develop such a system which can schedule its own to visit next location.

Intelligent transport system is used first to the problem of customer oriented Service problem. But in earlier systems there is no solution specified to the problem of customer oriented service problem.
In our system we used multiple agents to solve the problem. Unlike earlier systems we can have multiple agents to provide service in the customer places, which can schedule themselves to the next location.

**Multi-Agent Self Scheduling System:**
We develop an multi-agent self scheduling system to choose the traverse path from any point of the road map. In our system the mobile agents are able to schedule the next location to traverse at any point of the road map and traverse accordingly.

Our system contains the following components namely multi agent system, self scheduling system and visit log producer.

**Multi Agent System:**
This contains collection of mobile agents , which move from one location to other location in the road map. Once this agent
agent system will set a visit flag which specifies that the particular location is visited. This flag is a global variable

which can be accessed by any mobile agent and self scheduling system. Then the mobile agent will ask the self scheduling

**Self Scheduling System:**

The self scheduling system schedules the next location to be visit by the mobile agent. Each mobile agent contains its own self scheduling system and global road map and global visit flag which are commonly accessed by all the self scheduling system. Instead of using a single agent to visit to various location of the country through the road map we used multi agent system and self scheduling system to walk through and reduce the time complexity. The self scheduling system chooses the unvisited locations from the road map using the visit flags and it calculates the direction of the location and distance and time. Based on these details it schedules and select the next appropriate location to visit and returns to the mobile agent.

**Visit Log Producer:**

The log producer produces the previous visits made by the agents and from the logs it analyses the number of times the location has been visited and the number of times the customer was present and absent. Based on these details an visit weight will be calculated and stored.

The same will be repeated for all the unvisited locations by the agent and will be sorted. Finally an location will be selected as the next traverse point based on the visit weight.

VW = (NTp/NTa)*Tv

VW-Visit weight

NTp-No of time present

NTa-No of time absent

Tv – Total no of visits.

$$E_a\,[c] = \sum_{i=1}^{3} w_i |A_i| \qquad (1)$$

$$P_a\,[c] = \sum_{i=1}^{3} w_i |A_i| \qquad (2)$$

$$TQM\,[O] = \frac{1}{n+m}\left\{\sum_1^n E_a * \mu + \sum_1^m P_a * \beta\right\}$$

**Result And Discussion:**

Computer experiments were performed using the road map of india.

[Siva] Give screen shots of experiment

The roadmap contains of 250 cities with the lines connecting them. We use a text file named database.txt which contains the city names and distance. a single line contains two city names and the distance between them. The darken cities represents the starting places of the mobile agents. The visit weight of the starting location is calculated using the previous history.

The following table represents the overall time taken with various number of mobile agents.

| No of agents | overall time |
|---|---|
| 5 | 90 |
| 10 | 80 |
| 15 | 65 |
| 20 | 50 |
| 25 | 40 |

system to choose the next location to visit.

The following graph displays the overall time taken for different no. of cities to be visited using different algorithm.

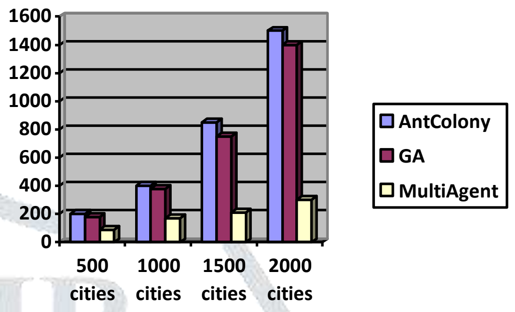


Fig2: show the chart between no of cities and time taken.

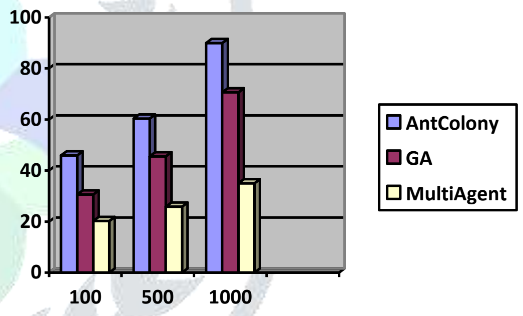


Fig 3: shows the space complexity

The figure shows the space complexity of all three algorithms according to the total number of cities and average distance to be traveled by each agent. The proposed algorithm reduces the overall distance covered by each agent.

**Conclusion:**

Our algorithm produces efficient results. Compared to ant colony system , ours reduces the time complexity and because of using multiple agents ours reduces the polynomial time. when we use more number of mobile agents. Genetic algorithm only Uses the static data’s and the history data ,but ours computes dynamic data and acts based on that. We further investigate the concept for the mobile network for better experiments.

While using ant colony algorithm it chooses the shortest path for the ant to reach the destination and follows, but in our algorithm the destination is not a fixed one and also it is not assigned to any mobile agent and only necessity is to visit all locations in a reduced time. Compare to ant colony algorithm it works for a single ant and schedules and choose the path to be followed but in ours the destination is temporary one and will change dynamically.

**References:**

[4] Ghiani, Gianpaolo, Antonella Quaranta, Chefi Triki. "New Policies for the Dynamic Traveling Salesman Problem." Optimization Methods and Software, Volume 22, Issue 6, December 2007.
[5] Lu, Xiangwen, Amelia C. Regan, Sandra Irani. "Theory Dynamic Traveling Salesman Problem: An Examination of Alternative Heuristics." Transportation Science, 2001.
[6] Larsen, Allan, Oli B.G. Madsen, Marius M. Solomon. "The A-Priori Dynamic Traveling Salesman Problem with Time Windows." Transportation Science, Volume 38, Issue 4, November 2004.

[7] Dijkstra, E.W. "A note on two problems in connexion with graphs." Numerische Mathematik, 1959.
[8] Bellman, Richard. "On a Routing Problem", Quarterly of Applied Mathematics. Volume 16, Issue 1, 1958.
[9] Ford Jr., Lestor R., D.R. Fulkerson. Flows in Networks. Princeton University Press, 1962.
[10] Johnson, Donald. "Efficient Algorithms for Shortest Paths in Sparse Networks." Journal of the ACM, 1977.
[11] Floyd, Robert. "Algorithm 97 (SHORTEST PATH)." Communications of the ACM, 1977.
[12] Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. Introduction to Algorithms – 2nd Edition. The MIT Press, 2001.
[13] Demetrescu, Camil and Giuseppe Italiano. "A New Approach to Dynamic All Pairs Shortest Paths." ACM Symposium on Theory of Computing, June 2003.
[14] Misra, S, B.J. Ooommen. "New Algorithms for Maintaining All-Pairs Shortest Paths." *IEEE 10th Symposium on Computers and Communications*, June 2005.

[15] Blum, Jeremy, Azim Eskandarian. "A Reliable Link-Layer Protocol for Robust and Scalable Intervehicle Communications." IEEE Transactions on Intelligent Transportation Systems. Volume 8, Number 1, March 2007.
[16] Tarng, Jenn-Hwan, Bing-Wen Chuang. "Investigation of Vehicle-to-Infrastructure Communications based on IPv6-based Automotive Telematics." *IEEE 7th Conference on Intelligent Transportation Systems Telecommunications*, June 2007.
[17] Sklavos, Nicolas, Maire McLoone, Xinmiao Zhang. "MONET Special Issue on Next Generation Hardware Architectures for Secure Mobile Computing." Mobile Networks & Applications, Volume 12, Number 4, August 2007.
[18] Zhanfeng, Jia, Chao Chen, Ben Coifman, Pravin Varaiya. "The PeMS algorithms for accurate, real-time estimates of g-factors and speeds from single-loop detectors." *IEEE 4th Intelligent Transportation Systems Conference*, February 12, 2001.
[19] Miller, Jeffrey. "Vehicle-to-Vehicle-to-Infrastructure (V2V2I) Intelligent Transportation System Architecture." *IEEE 4th Intelligent Vehicles Symposium*, June 2008.
[20] MIT's CarTel Project. http://cartel.csail.mit.edu.
[21] UC Berkeley's Mobile Millennium Project. http://traffic.berkeley.edu.
[22] Airsage Web Site. http://www.airsage.com.