# Multi-Strategy Ant Colony Optimization for Quota Travelling Salesman Problem with Passengers, Incomplete Rides and Collection Time

1st Satvik Watts
*B.Tech, IIT2020029*
*IIIT Allahabad*
Prayagraj, India

2nd Tarun Pal
*B.Tech, IIT2020098*
*IIIT Allahabad*
Prayagraj, India

3rd Kunal Prajapati
*B.Tech, IIT2020099*
*IIIT Allahabad*
Prayagraj, India

4th Ashok Malhotra
*B.Tech, IIB2020501*
*IIIT Allahabad*
Prayagraj, India

***Abstract*—The Quota Travelling Salesman Problem incorporating Passengers, Incomplete Rides, and Collection Time (QTSP-PIC) introduces a fresh perspective to the conventional Quota Travelling Salesman Problem (QTSP-PIC). In this modified version, the salesman effectively utilizes a versatile ride-sharing system to reduce travel expenses while accomplishing specific objectives, such as visiting a predetermined set of vertices to meet a predefined quota. This problem takes into account various operational constraints, including considerations for vehicle capacity, travel time limitations, passenger constraints, and penalties for rides that fail to adhere to passenger prerequisites. To address this complex challenge, we introduce both a mathematical formulation and heuristic approaches centred on Ant Colony Optimization.**

## I. Introduction

Currently, one of the major paradigms that is influencing the modern commercial world is collaborative consumption. Collaborative consumption means when a good or service is shared by a group. Whereas in normal consumption, an individual pays the full price of a good and maintains exclusive access to it, in collaborative consumption multiple people have access to the good or service, and all of them bear the cost of it either equally or unequally. Some examples of collaborative consumption are Airbnb, ride-sharing, etc. In the modern transportation world, this paradigm has led to the making of Mobility on Demand (MoD) systems like Uber and Old which have become an important part of our day-to-day lives. These systems encourage the use of shared mobility instead of traditional exclusive mobility, which would be underutilized.

These systems will have more importance in the society of the future than what is being advertised on the surface. If the systems become widely accepted, these will have benefits ranging from less waiting time, less traffic on the roads, and less noise pollution and it will even help in reducing climate change, the most urgent problem of our generation.

## II. Related Work

A lot of the recent work done in this field focuses on the following aspects: Planning optimal routes for a fixed-sized vehicle that satisfies the most amount of travel requests ([2]), managing the mobility on demand systems on the basis of self-driving vehicles ([3]), finding a single trip that serves given set of travel requests ([4]), analyzing the effect of dynamic ridesharing in a network containing multiple nodes([5]), effects of dynamic ride sharing in specific cities (NYC) using traditional taxis([6]), and last is the flexible ridesharing problem formulations ([7]).

Most if not all of these works focus on matching drivers and customers demanding the rides, or they focus on fulfilling the maximum number of requests. Also, these works mainly focus on professional drivers, that is a person whose job is to transport goods or provide driving-related services to people. However, there is another subset of the ride-sharing paradigm. There are cases when a person is travelling to different cities and may want to offer their vehicle of ride-sharing and hence save some cost. They may change their route in order to meet the requirements of some requests.

The problem implemented in this study is called the Quota Travelling Salesman problem with Passengers, Incomplete Rides and Collection Time. It models applications in which a person travelling to different cities can offer their ride to the above-mentioned system. The person can share his or her ride with other passengers and hence reduce the cost, and it also helps in reducing the cost for the passengers in comparison to other modes of transportation. This model also deals with the flexibility of a different drop-off point, which means dropping off a person to a different point than is previously asked. This will help in reducing the overall traffic. This idea will work as the passengers may have some money-related or environmental concerns which will lead them to drop off at an alternate point. Also, there may be some paths that are blocked or temporarily unavailable or are spatially near to other drop-off points, where a passenger may choose to drop based on the above concerns.

This model has a requirement that the driver must know the travel requests in advance so he can plan accordingly. It can be considered a realistic assumption, as most of the requirements of a passenger are known in advance and can be shared to allow the perfect match with the driver. It also helps in reducing the amount of time a vehicle travels without any passengers or is yet to be assigned a passenger. ([8]) indicates that the unallocated traffic (the vehicles with no passengers) would add more than 30 percent to the total traffic.

## III. Problem Definition

### A. Travelling Salesman Problem (TSP):

This problem can be modelled as a complete weighted directed graph $[G] = (N, E)$, where $N$ is the set of nodes and $[E] = (i, j)|i, j \in N$ is the set of edges. Each edge has a weight associated with it, stored in the matrix [EC], such that [EC]ij is the cost associated with the edge (i,j). The goal is to find the shortest or the least costly Hamiltonian cycle in G. A Hamiltonian cycle is a closed loop, where each node is visited exactly once.

### B. Quota Travelling Salesman Problem (QTSP):

In QTSP, with the conditions given in the TSP there is a bonus or quota associated with each vertex in [N]. The quotas are stored in the list [Q], where [Q]i is the quota associated with the vertex i. The salesman has a minimum quota that he has to collect. The goal of this problem is to find a path such that the sum of the quota of the visited vertices is at least the minimum quota and the cost of the path is minimum.

### C. Quota Travelling Salesman Problem with Passengers, Incomplete Rides and Collection Time (QTSP-PIC):

This is again a variant of the QTSP. In this variant, the salesman has a vehicle with a fixed number of seats. The salesman can reduce the expense of travelling to different cities by sharing the vehicle with other passengers. In this variant, along with the quota, there is also a time associated with each vertex which is the time it would require for the salesman to obtain that quota. The time is stored in the list [T] where [T]i is the time it would require the salesman to get the quota on the i'th vertex. Along with the time for accumulating each quota, there is also the time required to travel from one city to another, this is represented by the matrix [ET]. [ET]ij represents the time it would require the salesman to go from city i to city j.

For each passenger, there is a travel request associated with it. The request consists of the pickup point [PI], the drop-off point [DR], the budget of the passenger or the maximum amount of money the passenger is willing to pay [BU], the maximum duration the passenger is willing to travel [MD] and the penalties for dropping the passenger on any node other than the drop-off point [PE], where [PE]i represents the penalty of dropping the passenger off at the i'th node.

The salesman starts at a certain city, say city 1. The salesman needs to travel to some of the cities, and he needs to achieve a minimum quota. The salesman can visit any city at most once and he needs to return to city 1 at the end of the journey. In each city, the salesman can decide to either get the quota or leave it. The salesman will have a number of travel requests, each with their pickup, drop off points, budget, maximum duration and penalties for dropping at an alternate destination. For each request, the salesman can choose to either satisfy that request or not. The salesman has a vehicle with a fixed number of seats, and at any moment the number of passengers should not exceed the number of seats in the vehicle. If the salesman drops off a passenger anywhere other than the requested drop-off point, there is a penalty to be paid. The goal is to find a cycle in the graph [G] such that the cost of the route is maximized, and the minimum quota is achieved. The parameters are listed in Table I.

| **Class Node** | |
|---|---|
| N | Number of nodes |
| Ci | Coordinates of i'th node |
| Qi | Quota on i'th node |
| Ti | Time required to get quota on i'th node |
| **Class Graph** | |
| ADij | 1 if (i,j) are adjacent, else 0 |
| ECij | Cost required to traverse edge |
| ETij | Time required to traverse edge |
| **Class Passenger** | |
| Pli | Pickup point for i'th passenger |
| DRi | Drop off point for i'th passenger |
| BUi | Budget of the i'th passenger |
| MDi | Maximum duration of time i'th passenger is willing to travel |
| PEij | Penlty associated with dropping i'th passenger at the j'th node |

TABLE I
CLASSES USED IN THE CODE TO SOLVE THE PROBLEM

| **Algorithm-01** : $Solution(G, minimum_quota, P)$ |
|---|
| $S < -MSACO(G, minimum_quota)$ |
| $S' < -RMH(G, S, passengers)$ |
| $ReturnS'$ |

## IV. Methodology

In this section, we represent the solution representation and introduce the methods and algorithms we have used to implement the solution to the QTSP-PIC problem.

### A. Solution Representation

A solution for the QTSP-PIC is defined as [S] = N', Q', T', D'. Here, N' is the list of vertices that contains a cycle such that it satisfies the minimum quota, K. Q' is the list of vertices such that each vertex is in N', and it represents the vertices where the salesman stops to get the quota. T' contains the travel requests that the salesman chooses to accept and D' contains the nodes where the salesman drops off the passengers where [D']i contains the node where the passenger [T']i gets dropped off.

Our solution to the QTSP-PIC problem will consist of two parts: Finding the solution cycle [S] in [G] using the multi-strategy ant colony optimization, then giving this solution to the ride-matching heuristic function along with the list of passengers to get the reduced cost. The pseudo-code is given in Algorithm-01.

### B. Ride Matching Heuristic

Let S be a solution and X be the cycle in S, which consists of a list of vertices N'. RMH is a greedy method that assigns rides to S. The pseudo-code is given in Algorithm-02.

In this algorithm, we first sort the ride requests in decreasing order of the budget of the passengers, as passengers with more

**Algorithm-02** : $RMH(S,G,P)$

$S' <- copy(S)$
$L <- P sorted in decreasing order of P.BU$
$For each i \in L$
$If [PI]i \in S.N'$ :
$A <- get_drop_off(S,G,i)$
$Assign passenger i to S with drop-off vertex A$
$S.cost <- evaluate_cost(S)$
$If S.cost < S'.cost$ :
$S' <- S$
$Else$ :
$Remove passenger i from S$
$Return S'$

**Algorithm-03** : $AS(m,\alpha,\beta,\gamma,PH^0)$.

$Initialize pheromone for each edge$
$For j <- 1 to m$ :
$Wj[1] <- s$
$For j <- 1 to m$ :
$Wj <- build_route(\alpha,\beta)$
$Sk <- assign_passengers(Wk)$
$Pheromone_update(Wk,\rho,\tau i0j)$
$Update(S^*)$
$Return S^*$

budget are potentially more profitable for the salesman. Now, we traverse the list of passengers. For each passenger, if the pickup point is not on one of the vertices in the solution we skip the passenger. Otherwise, we compute the best possible drop-off point for the passenger among the remaining vertices in N', after the pickup point. This step is done in the function $[get_drop_off]$. Now, we calculate the cost of the ride if we accept the request of the current passenger. If the cost is less than the previous cost, we take the passenger. Otherwise, we cancel the request of the passenger.

### C. Ant based approaches

The section presents the Multi Strategy Ant Colony System, The approach proposed here is an adaptation of the Ant System (AS) ([9], and the Ant Colony System (ACS) ([10]).

*1) Ant System:* The AS was the first ant-based algorithm. It has the following parameters: number of ants $[m]$, initial pheromone in edges $[PH^0]$, where $[PH0]ij$ indicates the pheromone in edge $(i,j)$, pheromone coefficient $[\alpha] \in R > 0$, heuristic coefficient, $\beta \in R > 0$, evaporation factor, $\rho \in [0,1]$. The pseudo-code is in Algorithm-03.

Each ant starts at the vertex s. It builds its route till it achieves the minimum quota. After the route has been built, the path gets passed on to the RMH algorithm to assign passengers to the path and find out the reduced cost.

The ant moves from vertex i to an adjacent vertex j based on a probability function $\gamma$, the formula for this function is given as follows:

$$(\gamma^k)_{ij}(t) = [[PH_{ij}(t)]^\alpha[\eta_{ij}]^\beta]/[\sum_W \in AD[i][PH_{iw}(t)]^\alpha[[\eta_{ij}]^\beta]$$

Here, $\eta_{ij}$ is the inverse of the cost of the edge (i, j). The pheromone update is done as follows:

**Algorithm-04** : $ACS(maxIter,m,\alpha,\beta,\rho,PH^0,q0)$

$Initialize pheromone for each edge$
$For j <- 1 to m$ :
$Wj[1] <- s$
$For i <- 1 to maxIter$
$For j <- 1 to m$ :
$W^j <- build_route(\alpha,\beta,q0)$
$S^k <- assign_passengers(W^k)$
$Pheromone_update(Wk,\rho,\tau i0j)$
$Update(S*)$
$Global_pheromone_update(S^*,\rho)$
$Return S^*$

$$\Delta PH_{ij} = 1/len(W^k),\ if\ edge(i,j) \in W^k$$

$$= 0, otherwise$$

*2) Ant Colony System:* The ACS is similar to AS with the added greedy bias by the coefficient q0. Let $\omega$ be a random number from the range $[0,1]$. If $\omega < q0$, the ant chooses to move to j such that $\eta_{ij} = c1$ is maximum. The pseudo-code for ACS is given in Algorithm-04.

ACS also uses maxIter, the maximum number of iterations, and it also applies the concept of global pheromone update after each iteration. After each iteration, pheromone levels on each edge are multiplied by the evaporation factor. Other than that, it also calculates the probability of moving to the next edge same as in AS and the local pheromone updation is also the same as in the case of AS.

*3) Multi Strategy Ant Colony System:* There are multiple heuristics of many types in the problem QTSP-PIC: the cost of each edge, the quota on each vertex, the time taken to travel each edge, the number of passengers on each vertex and so on. So, in a multi-strategy ant colony system, the idea is to use different heuristics for different ants at random. In simple ant colony optimization, all ants use the same heuristic information. However, in MS-ACS, we have implemented four different types of strategies for ants to follow:

- Cost Oriented: Ants will prioritize the edge with the lowest cost, i.e., $\eta_{ij} = 1/EC_{ij}$.
- Time-Oriented: Ants will prioritize the edge which can be traversed in the lowest time, $\eta_{ij} = 1/ET_{ij}$.
- Quota Oriented: The heuristic is : $\eta_{ij} = Q_j/EC_{ij}$ .
- Passenger Oriented: The ants will prioritize the number of passengers in a node, $\eta_{ij} = L_j/EC_{ij}$.

## V. Experiments and Result

We have created 10,000 instances. The size of instances vary between 10 to 30 vertices, and the number of passengers in each instance vary between 2 times and 4 times the number of vertices in that instance.

In each instance, the nodes are created with random x and y coordinates. The quota and time required on each node is a random value between 0 and 1. The graph is a complete graph, that is there is an edge between any two pair of nodes. The time required to traverse an edge is a random value between

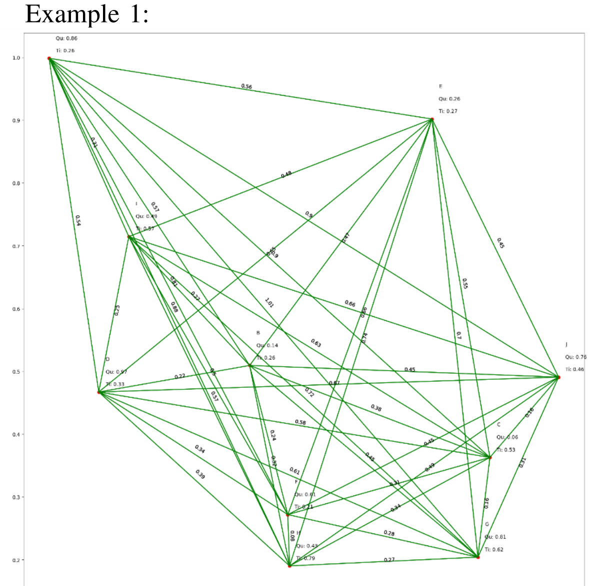
0 and 1 and the weight of each edge is the euclidean distance between the nodes.

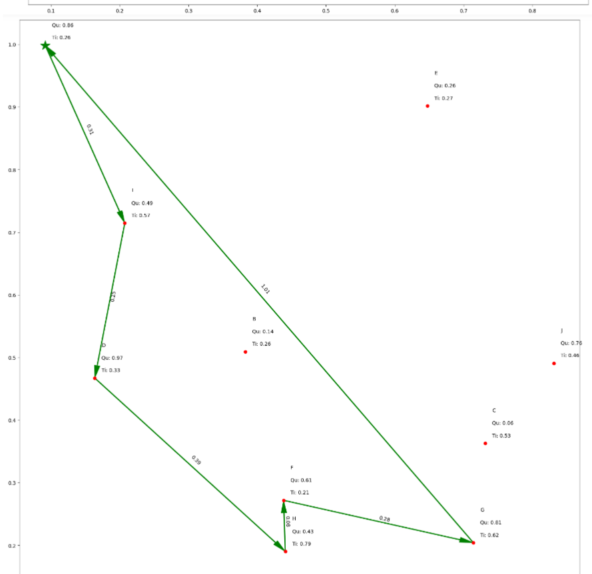
The passengers have a random pickup and destination point from the available number of nodes. The budget and the maximum duration for each passenger is a random integer between 0 and 10. The penalty for dropping any passenger to any node other than the drop-off point is the distance between that node and the drop-off node.

After we ran all of the 10,000 instances, we calculated the average percentage decrease in the total cost after the passengers were added which was a decrease of 35.519 per cent. This is not an insubstantial amount and it can have a great impact on the lives of people.

Some examples of the algorithm are shown below:

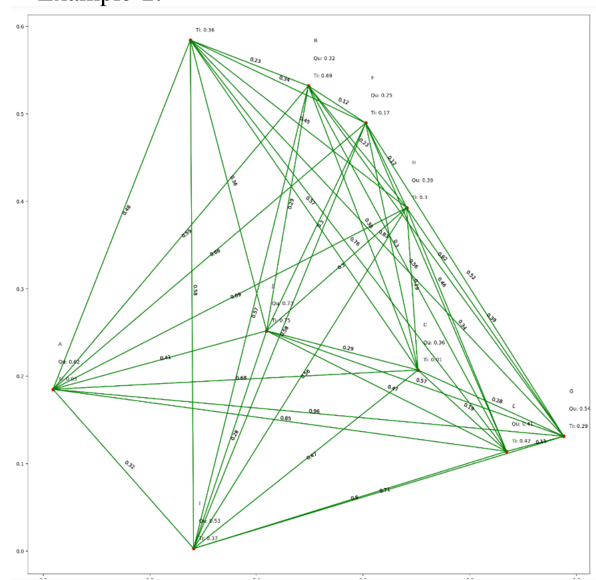
Example 1:

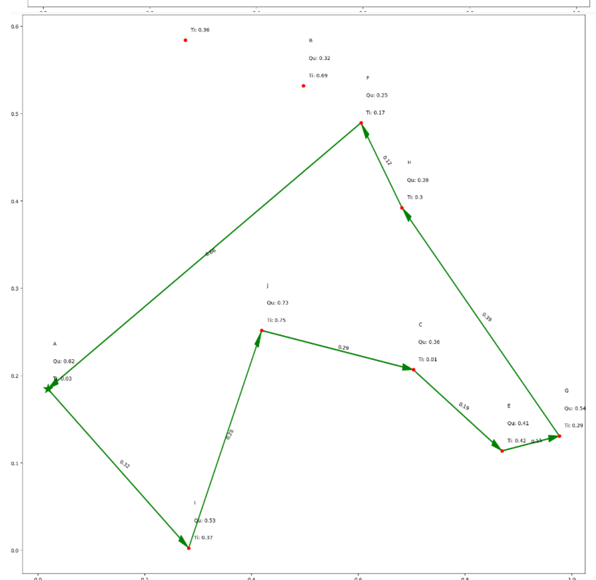




These are the details of the solution:
Path:['A', 'I', 'D', 'H', 'F', 'G']
Min Quota:4.045291650242762
Achieved Quota:4.177188869760784
Cost:1.3159092975889286
Reward:0.23678522217950765
Total Cost:1.079124075409421

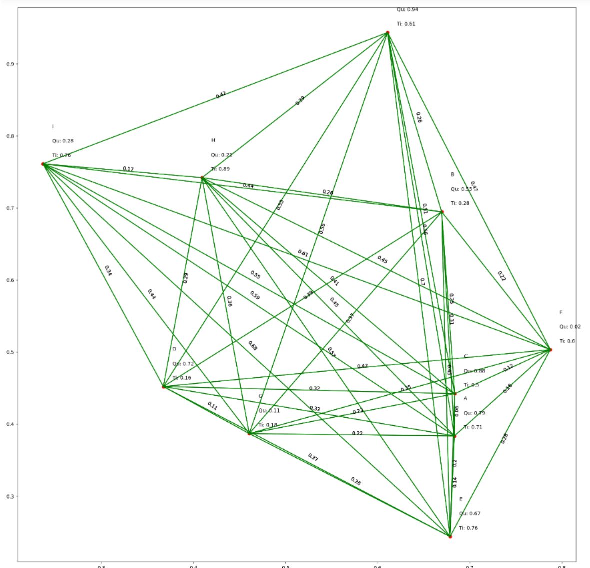
Example 2:

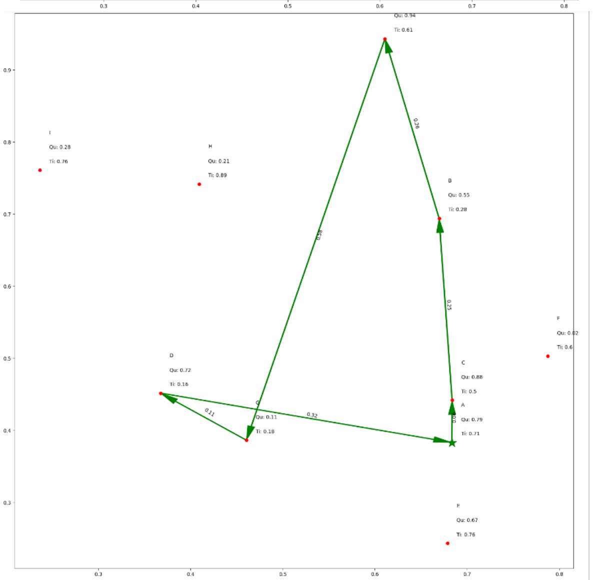




These are the details of the solution:
Path:['A', 'I', 'J', 'C', 'E', 'G', 'H', 'F']
Min Quota:3.7682914665391767
Achieved Quota:3.839176211304449
Cost:1.708431527280736
Reward:0.3877235721704543
Total Cost:1.3207079551102816

Example:





These are the details of the solution:
Path:['A', 'C', 'B', 'J', 'G', 'D']
Min Quota:3.883180414488326
Achieved Quota:3.9944410203425527
Cost:1.257767550251595
Reward:0.47317931931210716
Total Cost:0.7845882309394878

## VI. References


1) Bruno C.H. Silva, Islame F.C. Fernandes, Marco C. Goldbarg, Elizabeth F.G. Goldbarg, Quota travelling salesman problem with passengers, incomplete ride and collection time optimization by ant-based algorithms, Computers and Operations Research, Volume 120, 2020, 104950, ISSN 0305-0548, https://doi.org/10.1016/j.cor.2020.104950.
2) Ho, Sin and Szeto, W. and Kuo, Yong-Hong and Leung, Janny and Petering, Matthew and Tou, Terence. (2018). A survey of dial-a-ride problems: Literature review and recent developments. Transportation Research Part B: Methodological. 111. 10.1016/j.trb.2018.02.001.
3) Reza Vosooghi, Jakob Puchinger, Marija Jankovic, Anthony Vouillon, Shared autonomous vehicle simulation and service design, Transportation Research Part C: Emerging Technologies, Volume 107, 2019, Pages 15-33, ISSN 0968-090X, https://doi.org/10.1016/j.trc.2019.08.006.
4) Alonso-Mora, Javier and Samaranayake, Samitha and Wallar, Alex and Frazzoli, Emilio and Rus, Daniela. (2017). On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. Proceedings of the National Academy of Sciences. 114. 201611675. 10.1073/pnas.1611675114.
5) Chen, Z., Liu, X.C., Wei, R., 2018. Agent-based approach to analyzing the effects of dynamic ridesharing in a multimodal network. Comput. Environ. Urban Syst. doi:10.1016/j.compenvurbsys.2018.10.004.
6) Lokhandwala, Mustafa and Cai, Hua. (2018). Dynamic ride sharing using traditional taxis and shared autonomous taxis: A case study of NYC. Transportation Research Part C Emerging Technologies. 97. 45. 10.1016/j.trc.2018.10.007.
7) Vincent Armant, Kenneth N. Brown, Fast optimised ridesharing: Objectives, reformulations and driver flexibility, Expert Systems with Applications, Volume 141, 2020, 112914, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2019.112914.
8) Fiedler, David and Čáp, Michal and Certicky, Michal. (2017). Impact of Mobility-on-Demand on Traffic Congestion: Simulation-based Study. 10.1109/ITSC.2017.8317830.
9) Dorigo, Marco and Maniezzo, Vittorio and Colorni, Alberto. (1996). Ant System: Optimization by a colony of cooperating agents. IEEE Trans Syst Man Cybernetics - Part B. IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics: a publication of the IEEE Systems, Man, and Cybernetics Society. 26. 29-41. 10.1109/3477.484436.
10) Dorigo, Marco and Gambardella, Luca Maria. (1997). Gambardella, L.M.: Ant Colony System: A cooperative learning approach to the Traveling Salesman Problem. IEEE Tr. Evol. Comp. 1, 53-66. Evolutionary Computation, IEEE Transactions on. 1. 53 - 66. 10.1109/4235.585892.