Dynamic Ant Colony Optimization

TJHSST Computer Systems Lab 2006/2007
Ryan Ward

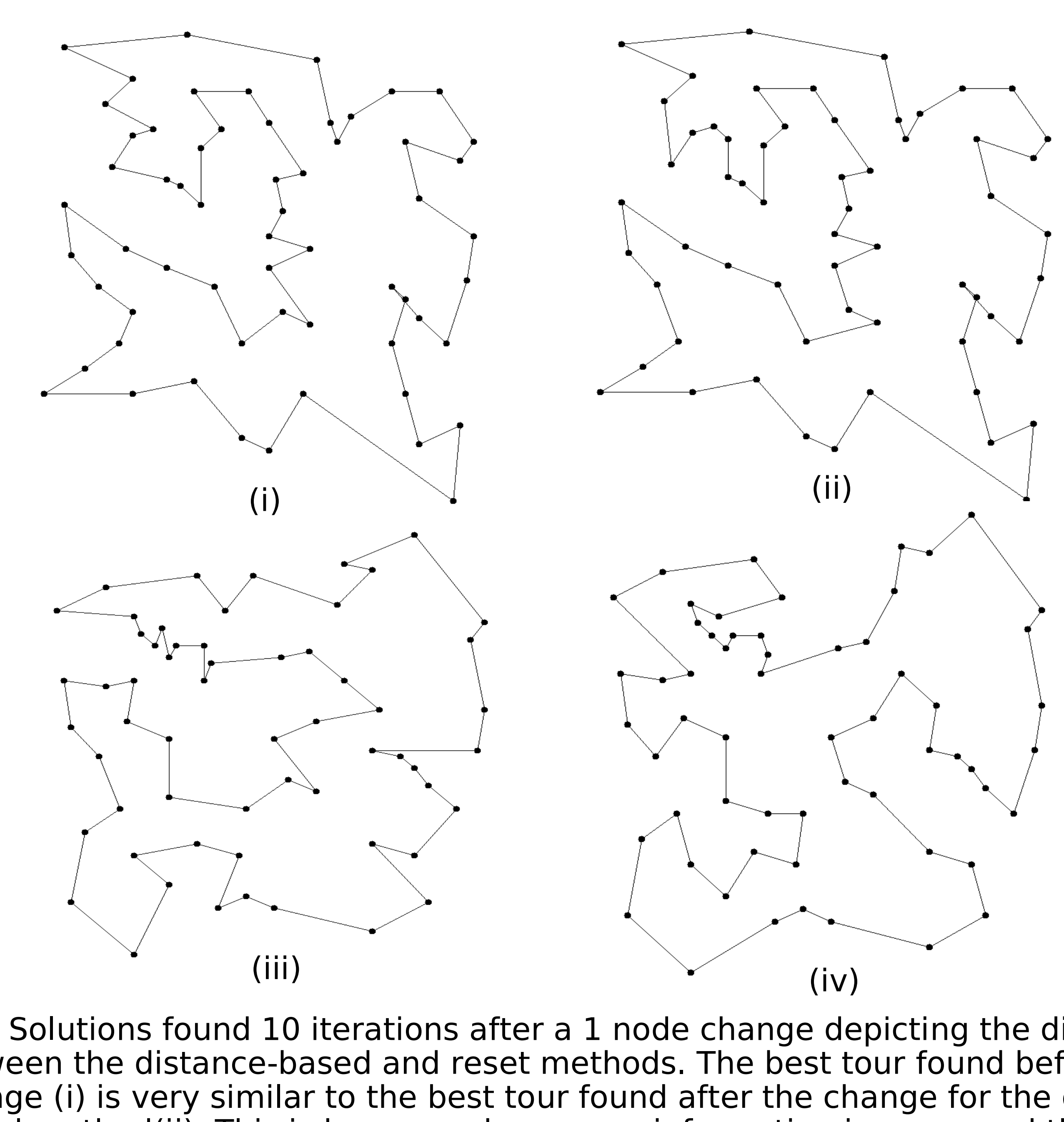
Background:

Ant Colony Optimization (ACO) is a metaheuristic that is used to find near-optimal solutions to combinatorial optimization problems. It was originally inspired by the natural system that ants use to develop a short path between their colony and a food source. When ants find a food source and travel back to their colony, they leave behind pheromone, a hormone that other ants follow. ACO algorithms, also known as Ant Algorithms, mimic this system, using agents that independently solve a problem and then modify pheromone values in the problem to increase or decrease the probability of a solution area being searched again.

The Traveling Salesman Problem (TSP) is, formally: given a weighted graph, find the tour that visits all the nodes with the smallest cost. This research will be using Euclidean 2D dynamic TSP (dTSP), where the cost from moving to node *i* to node *j* is related to the distance between the nodes, and where the set of nodes in the graph changes over time.

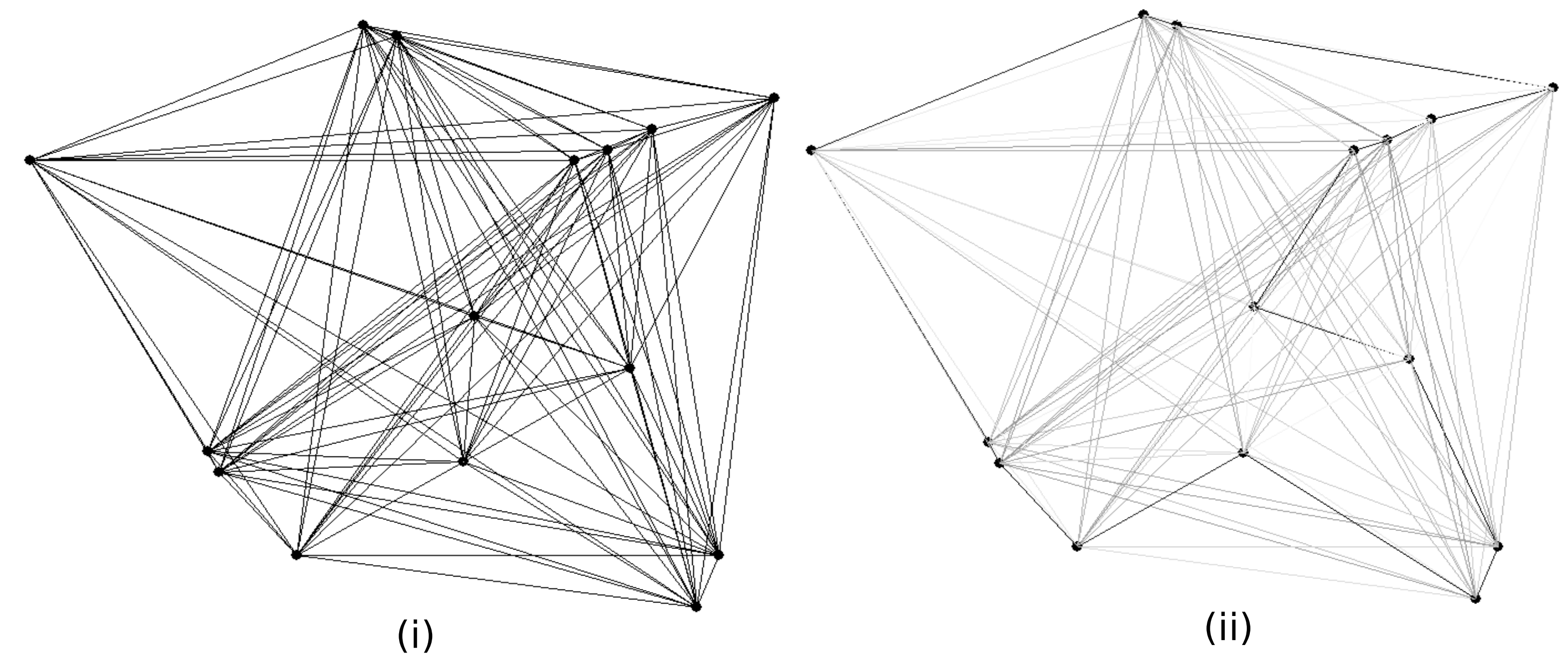
The two main factors that agents use when constructing a tour are the cost of moving across an edge (in this case, the distance between nodes) and the pheromone value associated with an edge (this changes during runtime). The cost of an edge represents known information about the problem, while the pheromone represents learned information about the problem. In a dTSP, the set of nodes in the problem changes, so learned information, pheromone, may become obsolete based on the location, frequency, and severity of the changes. It is important that pheromone is altered during a change in such a way that useful pheromone information is kept while obsolete pheromone is reset.

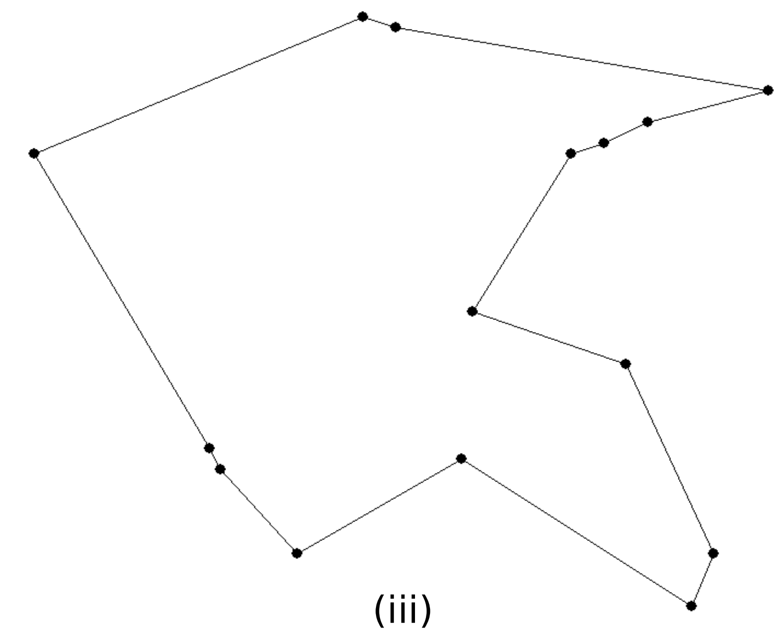
This research looked at two main ways of modifying pheromone information after a change in the problem. The first is the reset method, where all pheromone information is set to its initial value, effectively dropping all learned information about the problem. The second is the distance-based method, which resets pheromone connected to a node based on that node's proximity to a change in the problem.



Solutions found 10 iterations after a 1 node change depicting the difference between the distance-based and reset methods. The best tour found before the change (i) is very similar to the best tour found after the change for the distance-based method(ii). This is because pheromone information is preserved throughout a change based on a particular nodes distance from the changes, increasing the likelihood that the new solution will contain elements of the old one.

On the other hand, the best tour found before the change for the reset method (iii) is radically different from the best tour found after the change (iv). Pheromone information is completely reset, so the algorithm searches the entire solution space equally after the change, often arriving at a different solution area.





Various images depicting pheromone in a 15 node TSP problem. At the beginning of the problem, the algorithm assigns an equal pheromone value to all moves (i). During runtime, the algorithm either increases or decreases pheromone on certain moves based on whether or not agents who use those moves generate good solutions (ii). The resulting solution is a near optimal tour of all the nodes (iii). It is important to note that while the moves leading to the solution are favored by a high pheromone amount, the pheromone amount of other moves does not approach zero. This is to ensure continuous exploration of the solution area.

Conclusion:

After testing, it was determined that the distance-based method performed better than the reset method for small values of *f* (frequency) and *s* (severity), while the reset method outperformed the distance-based method for large values of *f* and *s*. When changes to the problem are small, preserving pheromone information is useful, since the solution to the new set is likely to share moves with the old solution. Large changes in the problem cause the solution to change radically, so preserving pheromone information is harmful, since it can mislead the algorithm initially and cause the algorithm to be stuck in a sub-optimal local minimum.

When changes to the problem are frequent, the reset method does not have enough time to converge on an answer. But, since the distance-based method saves pheromone information, it converges on a solution area faster, therefore being well suited for problems with frequent changes. In general, *s* is a more important variable for deciding which method performs better than *f*, unless *f* is a very small value (f<10 for eil101).

For the completely dynamic situation, the combined method outperformed either method separately. This is because it uses a different ratio of the reset and distance-based methods to utilize their strengths for different *s* and *f* values.

Application:

Dynamic Traveling Salesman Problems can be used to model many industrial problems, such finding optimal delivery routes, where customers are added and removed from the route, and computer network data flow, where servers go on and off line and ping time changes between servers depending on traffic.

TSP is a particular problem in the set of combinatorial optimization problems. Other combinatorial optimization problems include the dynamic Job-Shop Problem and the Quadratic Assignment Problem. ACO excels over Genetic Algorithms, Simulated Annealing, and Tabu Search for dynamic situations since it can run continuously.

This project uses the Ant Colony System, where at each iteration the pheromone values for the edges composing the best-so-far tour are increased and all pheromone values are slightly evaporated

ACO Pseudo Code

```
Initialize parameters, pheromone matrix
while (end conditions not met) do
        ConstructAntSolutions
        ModifyPheromoneValues
```