A Hueristic Learning Algorithm for Traveling Salesman Problems

Sim Kim Lau
University of Wollongong, simlau@uow.edu.au

L. Y. Shue
University of Wollongong, li_yen_shue@uow.edu.au

A Heuristic Learning Algorithm for Traveling Salesman Problems

Sim Kim Lau, Department of Information Systems, University of Wollongong, Australia, sim_kim_lau@uow.edu.au
Shue, L.Y., Department of Information Systems, University of Wollongong, Australia, li_yen_shue@uow.edu.au

Abstract

This paper describes the development of a heuristic learning algorithm to solve traveling salesman problems. The heuristic evaluation function of this algorithm considers both local and global estimated distance information. The heuristic learning mechanism allows the algorithm to update the heuristic estimates of visited states and hence modify the tour configuration along the search process. The search engine of the algorithm incorporates Delaunay Triangulations as part of the search strategy to identify candidate edges during the search process.

Introduction

In traveling salesman problem (TSP), a salesman is to complete a tour of N cities by visiting each city exactly once, and the tour begins and ends with the same city such that the total distance travelled is minimised. The combinatorial nature of TSP results in the problem being inherently intractable. In view of computational difficulties heuristic methods are frequently used to find near-optimal solution with reasonable computation time. Heuristic methods for TSP can be categorised into either tour construction or tour improvement (Lawler et. al., 1985).

Tour improvement heuristic is characterized by exchanging edges to change the current configuration of the tour. After an initial tour is built, the edge-exchange process is carried out until no further improvement can be found. Examples of tour improvement heuristics include 2-opt, 3-opt and Lin-Kernighan heuristics. The quality of the solution depends on tour configuration and there is a possibility that local, rather than global, optimal is found.

In tour construction heuristics, the tour is built from scratch and the city is added one at a time until a complete tour is found. Examples of such heuristics include nearest neighbour heuristic, nearest insertion heuristic, cheapest insertion heuristic, farthest insertion heuristic and random insertion heuristic. Tour construction heuristic is a greedy approach, part of the tour that is already built remains unchanged throughout the tour construction process. No attempt is made to change or undo part of the tour that has been built. This is in contrast to the tour improvement heuristic in which the configuration of the tour is changed during the iterative improvement process until a shorter tour is found. Secondly, tour construction heuristic is myopic. It often relies on local distance information to build tour.

The greedy nature of tour construction heuristics results in tour configuration remains unchanged during the tour construction process and the myopic nature of tour construction heuristics can lead to sub-optimal solution. The research aims to investigate the application of heuristic learning algorithm to address these two problems.

This paper describes the development of a heuristic learning strategy to solve TSP. The feature of the heuristic learning algorithm is the application of heuristic learning to update and improve the initially underestimated state heuristic estimates, which will lead to the improvement of state selection decision and an optimal solution when the goal state is reached. The heuristic evaluation function of the algorithm considers both local distance and global estimated distance. Local distance is the actual cost of moving from one state to another and global estimated distance is the heuristic estimate of a state to the goal state. The heuristic learning mechanism allows the algorithm to update the heuristic estimates of visited states and thus modify the tour configuration along the search process. This way the tour configuration changes as a result of heuristic learning.

Search and Learning A* algorithm

The strategy is based on an admissible heuristic learning algorithm called Search and Learning A* (SLA*). SLA* is an algorithm which repetitively updates the heuristic estimate of a state when the search path is backtracked. The algorithm progresses from the initial state to the goal state by updating the initial heuristic estimates during the process of searching and activating the backtracking process whenever heuristic learning occurs. SLA* algorithm is given as follows (Zamani, 1995):

Let	k(x,y) be the positive edge cost from state x to a neighboring state y, and
	h(x) be the non-overestimating heuristic estimate from state x to the goal state.
Step 1:	Put the root state on the backtrack list called OPEN.
Step 2:	Call the top-most state on the OPEN list x. If x is the goal state, stop; otherwise continue.
Step 3:	Evaluate {k(x,y) + h(y)} for every neighboring state y of x, and find the state with the minimum value. Call this state x'. Break ties randomly.
Step 4:	If h(x) ≥ {k(x,x') + h(x')}, add x' to the OPEN list as the top-most state and go to step 2. Otherwise replace h(x) with {k(x,x') + h(x')}.
Step 5:	If x is not the root state, remove x from the OPEN list.
Step 6:	Go to step 2.

From a front state x, the state selection process is based on the minimum increment of the heuristic function f(y) = k(x,y) + h(y), where k(x,y) is the positive true edge cost from state x to its neighbouring state y, and h(y) is the heuristic estimate of state y. The algorithm will first identify the state with the minimum f(y), and then compare it with h(x) to decide if h(x) can be improved. If it does, heuristic learning is said to occur. This relationship can be expressed as h(x) ≥ min {k(x,y) + h(y)}. Hence, if h(x) is not smaller than minimum f(y), then this relationship is true, and the state with minimum f(y) is added to the search path as the new front state. From this new state, the algorithm continues the search operation. If this heuristic relationship is not true, then h(x) is too much underestimated and it can be updated to this minimum f(y), then the new h(x) is replaced with the minimum of {k(x,y) + h(y)}.

Due to the fact that the selection criterion of a state is based on the heuristic estimate of its neighboring states, hence the update of h(x) may invalidate the selection of its previous state (x-1). In order to reflect the effect of the new h(x) to the search path, the algorithm conducts a backtracking operation by applying the above rationale to state (x-1) to see if h(x-1) can be updated. If h(x-1) is updated, then its previous state (x-2) will be reviewed as well. This way the states of the search path will be reviewed one by one in the reverse order. Along the way, any state whose heuristic estimate is improved will be detached from the path. This is because the improvement casts doubt on the validity of its previous minimum heuristic function status. This review process continues until it reaches the state whose heuristic estimate remains unchanged or until it reaches the root state. Then the algorithm resumes the search from this state. As a result, the algorithm would have completely updated its earlier path. Hence the search path that is to be developed subsequently will be a minimum path. When the goal is reached, the path is an optimal path and represents a complete solution.

Strategy to select candidate edges

The concept of Delaunay triangulations is used to construct candidate set which consists of promising edges from which state transition operator takes edges with priority. This is achieved by only considering those edges which are likely to result in an optimal tour. Delaunay triangulation is the triangulation with the property that no point in the point sets falls in the interior of the circumcircle of any triangle in the triangulation (Aurenhammer, 1991). For n-city problem, there are at most (3n-6) Delaunay edges and (2n-4) Delaunay triangles. As a result the search space is smaller because there are at most (3n-6) edges formed in Delaunay triangulation.

State-space formulation

In order to apply the search mechanism of SLA* to solve TSP, the problem needs to be transformed into a state space problem. A state is defined as a tour consists of the ordered selected cities and unvisited cities. The selected cities will form a partial incomplete tour by connecting cities in the order of their selection. Let the city of origin be city 1. For a given state S_i, it is represented as ((1,2,…,i,1),{U}), where i is the last city added to the tour and U is (i+1,i+2,...,n).

For a given state S_i, the transition to the next state S_{i+1} is through the selection of a city from the neighboring cities of i, which are in the unvisited set U. The neighboring cities of i are identified from the candidate edges formed using Delaunay triangulation. The state is selected based on the minimum increment of tour length. The transition cost from a parent state S_i to a child state S_{i+1} is the increment in distance between states S_i and S_{i+1}, which is [d(i,i+1) + d(i+1,1) - d(i,1)] with d(i,j) = $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ being the distance between two cities $i(x_i,y_i)$ and $j(x_j,y_j)$.

The application of SLA* requires a non-overestimating heuristic estimate from current state S_i to the goal state to be determined. The heuristic estimate for state S_i is calculated using the cost of minimal spanning tree on remaining (n-i) unvisited cities.

The proposed heuristic learning algorithm

The proposed heuristic learning algorithm which is based on the above state space representation is given as follows:

Let S_i be the ith state with its tour $P_i(1,2,...,i,1)$, where 1 is the city of origin and i is the last city of the tour. Its heuristic estimate h(i) is the minimum spanning tree of the remaining (n-i) cities. S_i is the goal state when i = n.
d(i,j) be the Euclidean distance between city i and city j.

H(i) be the estimated tour length for S_i, which consists of the tour p_i and h(i).

```
Step 0: Apply Delaunay Triangulation algorithm to find neighboring edges for each city.
Step 1: Locate the city of origin as the one with the smallest x-coordinate; choose the city with the largest y-coordinate to break ties.
Step 2: Put the root state on the backtrack list called OPEN.
Step 3: Call the top-most state on the OPEN list $S_i$. If $S_i$ is the goal state, stop. Otherwise continue.
Step 4: Find the (i+1)th city with $\min\{[d(1,2) + d(2,3) + \dots + d(i-1,i) + d(i,i+1) + d(i+1,1)] + h(i+1)\}$ from neighboring cities of i; break ties randomly. If no neighboring city of i can be found, go to step 6.
Step 5: If $\{[d(1,2)+d(2,3)+\dots+d(i-1,i)+d(i,1)] + h(i)\} \geq \min \{[d(1,2) + d(2,3) + \dots + d(i-1,i) + d(i,i+1) + d(i+1,1)] + h(i+1)\}$, add $S_{i+1}$ to the OPEN list as the top-most state; otherwise replace h(i) with $[d(i,i+1) + d(i+1,1) + h(i+1) - d(i,1)]$.
Step 6: Remove $S_i$ from OPEN list if it is not the root state.
Step 7: Go to step 3.
```

Conclusion and future research

This approach demonstrates that backtracking and forward search processes of heuristic learning will lead to dynamic construction of tour. This is achieved through the consideration of both the local and global estimated distance information in the form of k(x,y) and h(x) respectively. The information is updated continuously and it leads to repetitive deletion and addition of cities from and to the tour during the search process.

Future research will be conducted to evaluate the performance of the above algorithm to solve large-size problems. A computer program consists of the following modules will be developed:

(i) Delaunay triangulation module to identify candidate edges,

(ii) state formulation module to construct tour based on the above state transformation process,

(iii) heuristic estimation module to calculate heuristic estimate of a state using minimal spanning tree, and

(iv) path module to apply the state selection criterion for expanding and keeping track of the search path.

References

Aurenhammer, F. “Voronoi Diagrams – A Survey of a Fundamental Geometric Data Structure,” *ACM Computing Surveys*, 23(3), 1991, pp. 345-405

Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G. and Shmoys, D.B., *The traveling salesman problem*. John Wiley & Sons, 1985

Zamani, M.R. *Intelligent Graph-Search Techniques: An Application to Project Scheduling Under Multiple Resource Constraints*, PhD thesis, 1995