A Method of Filtered Beam Search Based Delivery Scheduling*

Member Heloisa Martins Shih (Yokohama National University)
Member Yigang Cai (Yokohama National University)
Member Takashi Sekiguchi (Yokohama National University)

In this paper we propose a heuristic procedure to solve the routing phase of delivery scheduling problem. Aiming at fast response, a beam search based decision procedure is utilized. In this research, a new evaluation rule named as "the most willing neighbor" is applied. The new rule retains the simplicity of the nearest neighbor rule, and overcomes its weak points. Three applications are presented and compared with the results obtained in 20 runs of the well-known 3-opt procedure. The proposed system offers flexibility and high quality solutions with very fast response.

Key words : TSP, Scheduling, Beam search, Filtered beam search, The most willing neighbor rule

1. Introduction

Delivery scheduling (or vehicle routing) problem is defined as the scheduling of one or more vehicles (trucks) with fixed capacities for transporting certain objects between specified pickup and delivery points. The delivery scheduling problem may be divided into two phases: assignment, determining which of the delivery points will be visited by each vehicle, and routing, determining which route each vehicle will take. In this paper, we propose an Artificial Intelligence (AI) based system to solve the routing phase of this problem.

In recent years, advances in hardware and software technology and availability of computerized geographical database, have made followings possible: the construction of enhanced graphics, remote data entry, sophisticated tracking of the vehicles, detailed road maps and address location systems, speed up the demand for computer-assisted real-time delivery scheduling systems(9).

These factors allied to the promising results obtained by beam search based FMS (Flexible Manufacturing System) scheduling system presented in Ref. (3), (4) constitute the initial motivation for developing the beam search based delivery scheduling system presented in this paper. Beam search is a search technique borrowed from AI theory.

The aimed characteristics of the proposed system are: possibility to provide real time response and fast rescheduling; high grade of adaptability, e.g., easy introduction of new delivery points, modification of distance matrix, etc.; and possibility of utilization under different scheduling conditions.

In Section 2 we present a brief review of the delivery scheduling problem. Section 3 presents an introduction of beam search technique. In Section 4 the proposed delivery scheduling system is described, and 3 applications are presented in Section 5. Section 6 contains our conclusions.

2. Delivery scheduling review

The mathematical formulation of the routing phase of the delivery scheduling problem is practically the same of the traveling salesperson problem (TSP) (5). A feasible solution to TSP is called a route. In a route, if $j$ is the next node to be visited after node $i$, it is said that the route contains an edge linking $i$ to $j$. One of the characteristics of the
TSP is its cost structure. In most of the problems, the locations to be visited all lie in the same geometric plane and the cost of traveling between any pair of locations is modeled as the Euclidean distance between the locations. These problems are called Euclidean TSP. Otherwise, TSP is said to be non-Euclidean. A TSP is said to be symmetric if the traveling cost from one location to another is equal to the cost of the reverse trip for all pairs of locations. Otherwise, TSP is asymmetric. Most of the proposed solution procedures are focused on Euclidean symmetric TSP.

TSP is NP-complete, and the computation time required to achieve optimal solution increases exponentially with the size of the problem. Integer programming, dynamic programming and branch and bound method have been proposed to solve TSP, but the time necessary to reach optimal solutions and, specially in the case of dynamic programming, enormous memory requirements, have limited the applications to very small problems. The most frequently chosen solution procedures for TSP are heuristics.

The heuristic methods may be divided into those: construction procedures, which aim to build a near optimal route starting from the original distance matrix; improvement procedures, which start with a feasible initial solution and seek to improve it via a sequence of interchanges; composite procedures, that apply construction procedures to find an initial solution, and then improve it by utilizing improvement procedures.

In many of the route construction procedures, the initial route is a randomly chosen node called "self-loop". Then, at each cycle following a defined rule, one node is added to the route constructed in the previous cycle.

The best-known improvement procedures are the edge exchange procedures proposed by Lin and Kernighan. In these procedures, starting from random initial solutions, in each improvement cycle a number of edges \( r \), in a feasible route are exchanged by \( r \) edges not in the solution, as long as the result remains a route and the length of the new route is less than the length of the previous route. Exchange procedures are referred as \( r \)-opt procedures, where \( r \) is the number of edges exchanged at each iteration. In Section 5 we utilize 3-opt procedure for comparison with the results obtained by the delivery scheduling system proposed in this research.

The beam search based on heuristic method, proposed in this paper, is a construction procedure. However, aiming at better solutions, a route is appended to the proposed procedure which tries to improve the final solution, by testing the possibility of exchanging the location each of the points in the final route. This is a simplified case of the improvement procedure developed by Lin. We call this routine as single point exchange. Therefore, what we proposed is a composite procedure.

3. Filtered beam search

3.1 Beam search method

Beam search is a well-known AI technique for efficient searching in decision trees, particularly where the solution space is vast. Some applications to scheduling problems have been reported. In beam search, the decision tree is constructed level by level. In each level \( l \), it applies an evaluation function to appraise all nodes. Only the best \( w \) nodes are selected as beams and expanded to next level. In other words, from all nodes of each level \( l \) only \( w \) nodes will be "saved" to compose possible solutions. The other nodes of level \( l \) are ignored forever. The number of beams \( w \) to be selected at each level is controlled by a user defined variable called beamwidth.

We will illustrate the concepts of beam search by a simple delivery scheduling example. There are one delivery vehicle and 5 requests to be delivered at 5 different spots (R1 to R5). We will select the one spot at a time in delivery order. Fig. 1 shows a decision tree for this problem. The nodes (circles) of the tree represent partial schedules: level 0 means no spot scheduled, level 1 means first spot scheduled, level 2 means second spot scheduled, and so on. Should the complete decision tree be constructed, the number of nodes at level 0, 1, 2, 3, 4 and 5 would be 1, 5, 20, 60, 120 and 120 respectively. In general, it is prohibitive to consider the complete tree, and the decision system includes methods to...
reject the “unpromising” nodes. In beam search this task is realized by evaluating the nodes of each level, and only “saving” the beam nodes. When level 5 is reached there are only two fully expanded branches corresponding to two alternative delivery schedules. The better one is chosen as solution of the problem.

To estimate if a node is promising, beam search utilizes an evaluation function. Ideally this evaluation function should be able to make a good estimation of the minimum total cost of the best solution available, starting from the partial solution represented by the node. Sometimes the estimation can not be of the optimal total cost itself, but simply some urgency or priority rating. Error-free evaluation functions are prohibitive in cost. On the other hand, errors in it may cause the optimal solution to be pruned away and never be recovered. Beam search method recognizes this danger, by selecting not only one but a number of “promising” nodes. The wider the beamwidth the greater the necessary computational effort will be, and the higher the safety is. As computational cost increases approximately linearly with beamwidth(4), while errors are eventually decreased, there is a tradeoff between costs and benefits.

3.2 Filtered beam search method

If the evaluation function fails to select a potentially good node as beam, the alternative solution represented by that node is lost forever. Ow and Morton(15) proposed a less aggressive extension of beam search, called filtered beam search. In filtered beam search the evaluation of the nodes at each level is realized in two phases:

(i) Filtering phase: During the filtering phase a fast and cheap procedure is utilized to appraise all the nodes at the level. The most promising nodes are selected as filtered nodes.

(ii) Beam selection phase: During this phase, the filtered nodes are evaluated by a more precise and frequently more costly or time consuming function to select the beam nodes.

Filtered beam search method utilizes two search control parameters: filterwidth, representing the number of filtered nodes at each level and beamwidth, representing the number of beams.

In filtered beam search, the utilization of costly procedures, like a partial or total expansion of the filtered nodes, is made possible, as it is applied only to a controlled number of “somehow promising” nodes. In general, filterwidth and beamwidth are selected by simulation or a user(4).

4. Beam search based delivery scheduling

4.1 The most willing neighbor rule

The objective of the proposed system is the generation of a minimum traveling distance route starting from a distribution center, visiting all delivery nodes exactly once and returning to the distribution center. Aiming at the development of a more general procedure, we do not limit the proposed system to Euclidean symmetric cases. Therefore, for the Euclidean symmetric case, the input data is the (x, y) coordinates of the delivery nodes and the distribution center. Alternatively, the distance (cost) matrix may be input directly, for any kind of problems. In addition, from the (x, y) coordinates, instead of Euclidean distances, the matrix of distances may be constructed with Cartesian distances. This would be recommended, for example, when the delivery problem is within a neighborhood and the vehicle must travel around the blocks (as in the mail delivery problem).

We propose a construction procedure based on filtered beam search. At each level of the decision.

Fig. 1. Beam search decision tree (beamwidth=2) for the scheduling of 5 deliveries (R1 to R5).
tree, the selection of the beams will be made in two
phases: selection of the filter nodes by a quick,
cheap rule; and then, selection of the beam nodes
by applying a global evaluation function to the filter
nodes. The evaluation function for the filtering
phase is adapted from the nearest neighbor rule.
The new rule is called the most willing neighbor.
According to it, the selection of node $j$, posterior to
a given node $i$, is realized as:

$$
\min_{j}(\text{dist}(i,j) - \max_{i}\min_{k}(\text{dist}(i,l)))
$$

where:

- $i$: last node added to the route (beam of previ-
ous level)
- $j$: filter node candidate, selected among the
nodes not allocated yet
- $\text{dist}(i,j)$: distance from node $i$ to node $j$

In other words, to evaluate the possibility of
traveling from the last node added to the route ($i$)
to a node not allocated yet ($j$), we subtract from the
distance ($i,j$) the maximum between the minimum
of line $i$ (minimum distance to leave node $i$) and
the minimum of column $j$ (minimum distance to reach
node $j$ from any other node), removing (or trying
to) the disparities caused by absolute distances. For
example, let's imagine that we have a somehow
isolated node $j$ whose nearest neighbor is $i$. The
nearest neighbor rule will not construct a connec-
tion from $i$ to $j$, if the distance ($i,j$) is greater than
the distances from $i$ to its other close neighbors,
even if $i$ is the nearest node to reach $j$. In the
proposed rule the priority is proportional to the
"willingness" of node $i$ to be the anterior neighbor
of $j$ or the "willingness" of $j$ to be the posterior
neighbor of $i$, which is greater.

4.2 Scheduling algorithm

The evaluation rule for the selection of the beams
tries to visualize the potential of each of the filter
nodes. Each of the filter nodes is expanded to a
complete route, utilizing the most willing neighbor
rule of Eq. (1). The total distance of the expanded
route is associated to the filter node. A beamwidth
controlled number of nodes with lower estimated
costs are selected as number of beams of the level.

The proposed algorithm is composed of:

(Step 1) Utilizing filtered beam search, con-
struct the routing.

Starting from the distribution center (level zero),
for each level (level 1 to level $n$, where $n$ is the
number of nodes to be visited):

(i) Select the filter nodes of the level, favoring
the most willing neighbors (Eq. (1)) of the beam
nodes of the previous level.

(ii) Select beam nodes at the level by evalu-
ating the potential travelling distance of the filter
nodes with the most willing neighbor rule.

(iii) The route with minimum total travelling
distance at level $n$ is the suggested solution.

(Step 2) Analyze the suggested solution. If it
is satisfactory, stop the procedure. If it is not ac-
ceptable, divide the nodes into 2 to 4 groups.

(Step 3) Alter the most willing neighbor rule
to include the condition: "once traveling in a group,
the vehicle must visit all the nodes within the group
before leaving the group".

(Step 4) Return to Step 1.

Step 1 is processed by a Personal Computer
Turbo C program. The input data are filterwidth,
beamwidth, number of delivery nodes and, alterna-
tively, either $(x,y)$ coordinates of the delivery nodes
and distribution center, and the kind of distance
function (Euclidean or Cartesian) or distance (cost)
matrix.

Step 2 is executed by a user. In this Step, a route
is considered not acceptable, for example, when it
includes a number of crossing paths. The objectives
of the division in groups are: (a) to force the
algorithm to visit all the nodes in each group, be-
fore visiting other groups, and (b) to try to prevent
the crossing paths. We believe that most real cases
have natural divisions provided by, for example, the
disposition of the roads network, rivers, traffic
restrictions. As will be seen in next section, in the
applications studied we utilized simple geographic
divisions-north/south, east/west-which caused a
great impact in the quality of the final solution.

The rule modification contained in Step 3 is al-
ready presented in the developed computer pro-
gram. By only inputting the decisions from Step 2
and restarting the program, a new solution is con-
structed.
Having observed that the composite procedures produce better solutions than pure constructive methods (Section 2), a “single point exchange” routine that tries to improve the final solution, by testing the possibility of exchanging the location each of the points in the final route, is appended to our program. To achieve an optimal route relative to single point exchange, each time the route is improved, the testing process must be started from the beginning again, testing the new route for optimality relative to single point exchange. This is done until the current route can no longer be improved.

5. Application examples

The proposed algorithm was tested utilizing random generated problems and problems reported in the literature. We will present 3 of these applications. The reported computation times were obtained in a Personal Computer NEC PC-9801RX. To compare the obtained results, we developed also a Turbo C program for the 3-opt algorithm proposed by Lin\(^{[22]}\) as explained in Section 2.

5.1 Random problem with 20 nodes

Fig. 2 shows a delivery problem obtained by random generation of 20 nodes and one distribution center (node C in the figure) in the plane.

Part (a) of the Fig. 2 is the route obtained by the first application of Step 1 of the proposed algorithm, with filterwidth 4 and beamwidth 2. We use Cartesian distances, and the length of the obtained route is 137. As can be observed, there is a crossing of paths 20-1 and 19-8, caused apparently by the fact that some nodes of the southern part of the problem (nodes 8, 10 and 19) were skipped when visiting the other southern nodes. We decided to divide the problem into two groups of nodes: north (nodes 13 to 18) and south (nodes 1 to 12 plus nodes 19 and 20).

The solution obtained by the new application of Step 1 of the proposed procedure is shown in Fig. 2 (b). The total computation time is approximately 2 seconds. The new route has a total traveled length of 129 and is optimal to single point exchange, as no improvement is obtained by the single point exchange routine.

In 20 runs of the 3-opt procedure starting from random initial solutions, the best solution found is identical to the route shown in Fig. 2 (b). The total computation time for 20 runs was 49 seconds.

5.2 Random problem with 30 nodes

The 30 nodes/1 distribution center delivery problem, shown in Figure 3, was constructed by random generation of the nodes in a 100 X 100 plane.

Initially, applying filtered beam search with filterwidth 4 and beamwidth 2 we obtained a route with total traveled distance of 672. There were a few crossing paths caused, by overlooking of some of the nodes in the first north/south part of the
route. Hence, we divided the nodes in two groups: north (nodes 3 to 8, 10, 16, to 19, 21, to 27) and south (nodes 1, 2, 9, 11 to 15, 20, 28, 29, 30). The length of the new best route is 614 and there are no more crossing paths. The computation time to find this solution is around 4 seconds. Fig. 3 (a) shows the solution obtained after the application of the single point exchange routine (obtained after only two changes). The optimal to single point exchange route has a length of 608.

In 20 runs of the 3-opt procedure for this problem, we obtained the best solution with length 606 in 174 seconds of computation time. The best route obtained by the 3-opt procedure is shown in Fig. 3 (b).

5.3 33-city problem of Karg and Thompson

Karg and Thompson\cite{Karg} presented this problem in their paper, including the distances matrix. Due to the easy access to the utilized data, it has been largely employed as a "classical example" with known optimal solution. It is composed of a random choice of 33 cities of the U. S. A. An outline of the disposition of the cities is presented in Fig. 4. As there is no pre-defined distribution center, node 15 is adopted as center.

The route displayed in Fig. 4 (a) is obtained by the application of Step 1 of the proposed algorithm, utilizing filterwidth 4, beamwidth 2 and the distance matrix from Karg and Thompson, after 4 seconds of computation time. The total length of the route is 11032, less than 1.6% longer than the known optimal route, that has total length of 10861. As there are no crossing paths, we decide that the solution is satisfactory and no group division is made. The application of the single point exchange routine to the obtained solution resulted in the optimal route after

\begin{figure}
\centering
\includegraphics[width=\textwidth]{fig3.png}
\caption{Random problem with 30 nodes.}
\end{figure}

\begin{figure}
\centering
\includegraphics[width=\textwidth]{fig4.png}
\caption{33-City Problem of Karg and Thompson.}
\end{figure}
Table 1. Comparison of results obtained by the proposed system and 3-opt procedure.

<table>
<thead>
<tr>
<th></th>
<th>20 nodes rnd</th>
<th>30 nodes rnd</th>
<th>33-city prob.**</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>cost</td>
<td>time**</td>
<td>cost</td>
</tr>
<tr>
<td>Proposed system/plain**</td>
<td>129</td>
<td>2</td>
<td>614</td>
</tr>
<tr>
<td>Proposed system/improv**</td>
<td>129</td>
<td>3</td>
<td>606</td>
</tr>
<tr>
<td>3-opt procedure/best**</td>
<td>129</td>
<td>49</td>
<td>606</td>
</tr>
<tr>
<td>3-opt procedure/mean**</td>
<td>130.3</td>
<td>2.5</td>
<td>622.0</td>
</tr>
</tbody>
</table>

- * 1 problem presented by Karg and Thompson
- * 2 seconds on Personal Computer NEC PC-9801 RX
- * 3 without single point exchange routine
- * 4 with single point exchange routine
- * 5 cost of best solution from 20 runs starting with random initial routes and total time of 20 runs
- * 6 mean cost of solution of 20 runs and mean time of 1 run

only one exchange. The optimal route is shown in Fig. 4 (b).

20 runs of the 3-opt procedure for this problem, utilizing a total of 229 seconds of computation time, provides solution the optimal solution displayed in Figure 4 (b).

Table 1 contains a synopsis of the results obtained by utilizing the proposed method without single point exchange routine, the proposed method with single point exchange routine and the best and mean solutions of 3-opt procedure. By analyzing the table we may conclude that the proposed method produces solutions equivalent to the best solution in 20 runs of the comparison procedure with computation time at least equivalent to one run of the 3-opt procedure.

5.4 About large scale problems

In general, a large scale of delivery scheduling will have 1,000 or more nodes. The number of routes R for n nodes at each level l of the complete decision tree is

\[ R(l) = n! / (n-l)! \] ............................. (2)

That is, to obtain the optimal solution, \( n! (\geq 2^{n-1}) \) routes should be examined. On the other hand, if we construct a filtered beam search decision tree with filterwidth \( w_f \) and beamwidth \( w_b \) for the same problem, the number of routes \( R_b \) is

\[ R_b(l) = \begin{cases} \frac{w_f + n}{w_f + w_b(n-l+1)} & (l=1) \\ \frac{w_f + n}{w_f + w_b(n-l+1)} & (l \geq 2) \end{cases} \] ................................. (3)

The number of routes which must be finally examined is \( w_b \). This method cannot be compared directly with the complete decision tree. In this case, we compare the computer cost by the number of evaluated times of evaluation function \( N \). \( N \) can be shown as

\[ N = \sum_{l=1}^{n} R_b(l) = w_f(n+1) + \frac{1}{2} w_b n(n-1) \] ................................. (4)

So, we conclude that \( N \) increases linearly with \( o(n^2) \).

6. Conclusions

In this paper we proposed a filtered beam search based procedure to solve the routing phase of the delivery scheduling problem. A new evaluation rule called the most willing neighbor was also proposed. The new rule retains the simplicity of the nearest neighbor rule, and overcomes its weak points. Three applications were presented and compared with the results obtained in 20 runs of the 3-opt procedure presented by Lin. The proposed system presented:

(1) Flexibility, as it may be utilized indifferently with Euclidean, Cartesian or non-Euclidean and non-Cartesian distances (costs), and with symmetric or asymmetric distance (cost) matrix.

(2) High quality solutions, enhanced by the single point exchange routine. The proposed procedure provides the best solution of the 3-opt procedure for the 20-node problem and route 0.33% longer than the best solution of the 3-opt algorithm for the 30-node problem. For the 33-city problem, with known optimal solution, the algorithm provides a solution with total travelling distance 1.6% longer than the optimal solution without the single point exchange routine. By applying the routine the optimal solution itself was obtained after only one exchange.

(3) Very fast response. The computation time of the proposed procedure with single point exchange is approximately equivalent to a single run.
of the 3-opt procedure for the 20-node problem, and less than a single run of the 3-opt procedure for the 30-node and the 33-city problems. We may observe also that the rate of increase of the computation time with the size of the problem is smaller for the proposed system.

From now on, we intend to incorporate to the system time windows, restricting the time interval during which delivery to some of the nodes has to be completed, and capacity constraint for the delivery vehicles. We are studying also the inclusion of the assignment phase to the algorithm.

(Manuscript received March, 4.'92)

References


Heloisa Martins Shin (Member)

She was born in Brazil in February 1961. She graduated and received the Master of Engineering degree from University of Sao Paulo, Brazil-Department of Industrial Engineering in 1982 and 1988 respectively. She received her PhD form the Division of Electrical and Computer Engineering of Yokohama National University in 1992. Since 1983 she is Assistant Professor at the Department of Industrial Engineering of University of Sao Paulo, Brazil. Her main areas of research are production planning, scheduling and management in a factory automation environment. She is a member of the Brazilian Society of Industrial Engineers.

Yigang Cai (Member)

He was born in China in March 1960. He graduated and received the Master of Engineering degree from the Department of Computer of Yamanashi University, Japan, in 1984 and 1986 respectively. He received the Doctor of Engineering degree from the Division of Electrical and Computer Engineering of Yokohama National University in 1991. Now he is Assistant Researcher at the same Division. His main areas of research are expert system, knowledge acquisition, sequential control systems and Petri net theory. He is a member of the Society of Instrument and Control Engineers of Japan.

Takashi Sekiguchi (Member)

He was born in Japan in June 1935. He graduated from Yokohama National University (Department of Electrical Engineering) in 1959 and joined, in the same year, Yaskawa Electric Co. Since 1961 he joined the Division of Electrical and Computer Engineering of Yokohama National University, where he is now a Full Professor. He holds a Doctor of Engineering degree. His main areas of research are motor control, sequential control systems and CIM/FA systems. He is a member of the Society of Instrument and Control Engineers of Japan and the Industrial Electronics Society of IEEE.

1068 T. IEE Japan, Vol. 113-D, No. 9, '93