

Principal Investigator/Project Director: Randolph W. Hall

Institution: University of Southern California

Award Number: DMI-0127965

Program: OR/SEE operations research/service enterprise engineering

Project Title: Dynamic Cargo Assignment and Route Planning in the Trucking Industry

FULL-TRUCK-LOAD ASSIGNMENT AND ROUTE PLANNING IN DETERMINISTIC AND STOCHASTIC ENVIRONMENTS¹

Hossein Jula^{*}, Maged Dessouky^{**}, Petros Ioannou^{*†}, and Randolph Hall^{**}

^{*} *Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90089-2562*

^{**} *Department of Industrial and Systems Engineering, University of Southern California, Los Angeles, CA 90089-0193*

[†] *Corresponding author: Email: ioannou@rcf.usc.edu, Tel: (213) 740 4452*

Abstract: The full-truck-load route planning and scheduling with time appointment system falls in the class of Traveling Salesman Problem with Time Windows (TSPTW). In this paper, methods for improving the scheduling of intermodal trucks in deterministic and stochastic environments, where ISO containers need to be transferred between marine terminals, intermodal facilities, and end customers are investigated. The objective is to reduce empty miles, and to improve customer service. Computational experiments are used to demonstrate the efficiency of the proposed methods.

1. INTRODUCTION

The substantial increase in international volume of cargo arriving at U.S. ports together with the growth in the national freight have introduced congestion at many traffic networks and led to new dynamics in surface transportation. Worldwide container trade is growing at a 9.5% annual rate, and the U.S. growth rate is around 6% (Vickerman 1998). Every major port is anticipated to at least double its cargo by 2020 (Ryan 1998). As a consequence, truck traffic in the vicinity of the nation's ports is likely to grow substantially, leading to increased roadway congestion, and delays for both ordinary drivers and for the import and export of goods.

In today's trucking industry, there is a lot of discussion about the appointment window system (which will be called time window, hereafter) to manage the flow of trucks in traffic networks and at customers' locations. This system requires that freight carriers deliver/pick-up their cargo within a

¹ This work is supported in part by the National Science Foundation under grant DMI-0127965, and in part by METTRANS located at the University of Southern California and the California State University at Long Beach. The contents of this paper reflect the views of the authors who are responsible for the facts and the accuracy of the data presented herein.

specified time period due to the commitments made to the end customers, and due to the limited availability of certain resources at terminals.

In this paper methods are investigated to improve the scheduling of trucks in deterministic and stochastic environments, where ISO containers need to be transferred between marine terminals, intermodal facilities, and end customers. The containers need to be delivered/picked up within a specified time period. The objective is to reduce empty miles, and to improve customer service. The problems studied fall in the class of full-truck-load vehicle planning, routing and scheduling, which can be modeled by the Traveling Salesman Problem with Time Windows (TSPTW) (Savelsbergh and Sol 1995, Jula et al. 2002).

It should be noted that the TSPTW problem is an interesting special case of the Vehicle Routing Problem with Time Windows (VRPTW) in which the capacity constraints are relaxed. Recently the VRPTW, and a variety of its practical applications have been the subject of a wide body of research (Golden and Assad 1988, Laporte 1992, Desrochers et al. 1992, Ball et al. 1995, Dumas et al. 1995, Kohl et al. 1999). In VRPTW a number of vehicles, each with a given capacity, is located at a single depot and must serve a number of geographically dispersed customers. Each customer has a given demand and must be served within a specified time window. The objective is to minimize the total cost of travel (Desrochers et al. 1988).

In this paper, we study the full-truck-load routing, planning and scheduling both in the deterministic and stochastic networks. For the deterministic network, the multi-vehicle TSPTW is considered, and the following three methodologies are developed and compared.

- a. An exact method based on Dynamic Programming (DP),
- b. A hybrid methodology consisting of dynamic programming in conjunction with genetic algorithms (GA), and
- c. A heuristic insertion method.

For the stochastic environment, we develop a methodology to address the existing non-linearity formed by hard time window. Furthermore, we propose an approximate solution method for non-stationary stochastic TSPTW with random travel and service times.

2. DETERMINISTIC ENVIRONMENT

2.1. Problem Description

Let $G=(ND,A)$ be a graph with node set $ND=\{o,d,N\}$ and arc set $A=\{(i,j) \mid i,j \in ND\}$. The nodes $\{o\}$ and $\{d\}$ represent the single depot (origin-depot and destination-depot), and $N=\{1,2,\dots,n\}$ is the set of customers. To each arc $(i,j) \in A$, a cost c_{ij} and a duration of time t_{ij} are associated representing the cost and time of traveling between nodes i and j , respectively. In addition, to each node $i \in ND$ a service time s_i and a time window $[a_i, b_i]$ are associated. The service time s_i is the duration of time for a vehicle to be served at node i , and a_i and b_i are the earliest and latest time to visit node i , respectively. An arc $(i,j) \in A$ is feasible iff $a_i + s_i + t_{ij} \leq b_j$. Let V be the set of vehicles v . A route in the graph G is defined as assigning a set of nodes $r^v=\{o, w^v_1, w^v_2, \dots, w^v_k, d\}$ to vehicle v such that each arc in r^v belongs to A , and the time that service begins at node $j \in r^v$ is within the time window of that node.

Let's also define: $x_{ij}^v=1$ if arc $(i,j) \in A$ is traveled by vehicle v and is in the optimal path. $x_{ij}^v=0$ otherwise. T_i^v is the time when service begins at node i by vehicle v .

The m-TSPTW can be formulated as follows:

$$\text{Min} \quad \sum_{v \in V} \sum_{(i,j) \in A} c_{ij} x_{ij}^v \quad (1a)$$

$$\text{Subject to} \quad \sum_{i \in V} \sum_{j \in N \cup \{d\}} x_{ij}^v = 1 \quad \forall i \in N \quad (1b)$$

$$\sum_{i \in V} \sum_{j \in N} x_{oj}^v \leq |V| \quad \forall i \in N, v \in V \quad (1c)$$

$$\sum_{j \in N \cup \{d\}} x_{ij}^v - \sum_{j \in N \cup \{o\}} x_{ji}^v = 0 \quad \forall i, j \in ND, v \in V \quad (1d)$$

$$x_{ij}^v (T_i^v + s_i + t_{ij} - T_j^v) \leq 0 \quad \forall i, j \in ND, v \in V \quad (1e)$$

$$a_i \leq T_i^v \leq b_i \quad \forall i \in ND, v \in V \quad (1f)$$

$$T_d^v - T_o^v \leq T \quad \forall v \in V \quad (1g)$$

$$x_{ij}^v \in \{0, 1\} \quad \forall (i,j) \in A, v \in V \quad (1h)$$

Constraints (1b) require that only one vehicle visit each node in N . Constraints (1c) ensure that at most $|V|$ number of vehicles are used. To fix the number of vehicles, the inequality should be replaced by equality. Constraints (1d) guarantee that the number of vehicles leaving node j is the same as the number of vehicles entering the node. Therefore, constraints (1b) to (1d) together enforce that at most $|V|$ number of vehicles visit all nodes in N only once. Constraints (1e) enforce the time feasibility condition on consecutive nodes. Constraints (1f) specify the time window constraints at each node. Constraints (1g) require that each vehicle shall be used less than a certain number of hours per day, and finally constraints (1h) are the binary constraints.

2.2. Proposed Solution Methods for m-TSPTW

It should be noted that in contrast to the relation between the TSP and the m-TSP (e.g., see Reinelt (1994)), the m-TSPTW cannot be transformed into an equivalent TSPTW. In the following, we have developed three methodologies to solve the m-TSPTW.

2.2.1. Exact method for m-TSPTW

A two-phase Dynamic Programming (DP) based method is proposed for solving the m-TSPTW. The method is an extension of the algorithm used for the TSPTW and proposed in Dumas et al. (1995). Partial paths, however, which cannot be extended to other nodes, will not be eliminated, as it may be on the optimum set of solutions. Moreover, at each node tests are conducted to ensure the reachability of the depot according to the constraint (1g). The outcome of the first phase will be sets of feasible solutions. The sets, then, will be fed to another DP algorithm in order to find a set of routes with minimum total cost which covers all nodes.

Phase One: We require that the triangular inequality holds for both the travel costs and the travel times between each two nodes of the graph G . That is, for any $i, j, k \in ND$, we have $c_{ik} \leq c_{ij} + c_{jk}$ and $t_{ik} \leq t_{ij} + t_{jk}$. Let $S \subseteq N$ be an unordered set of visited nodes, $i \in S$ be the last visited node by vehicle v (i.e., on path v); and t be the time at which service begins at node i . Associated to each state (S, i, t) , defined above, is a cost denoted by $F(S, i, t)$ and defined as the least cost with minimum spanned time of a path starting at node $\{o\} \in ND$ passing through every node of S exactly once and ending at node i . Note that, there are several paths that visit set S and end at node i . Among them, we choose the one with minimum cost and minimum spanned time (see the state elimination test 2, below). Let also $fr(S) \in N$ be the first node in set S . The starting time of set S is denoted by $t_{fr(S)}$ and is defined as $t_{fr(S)} = \max(a_o, a_{fr(S)} - s_o - t_{o,fr(S)})$. This time indicates the earliest possible vehicle dispatching time from the depot such that no waiting time is necessary at node $fr(S)$.

In order to reduce the computational time, two types of elimination tests are performed: arc elimination, and state elimination.

1) *Arc elimination tests:* These tests are performed before and during running the DP algorithm.

- a. Arc elimination before running the algorithm. Let $EAT(i, j)$ be the earliest arrival time at node j from node i . $EAT(i, j)$ is defined as follows:

$$EAT(i, j) = a_i + s_i + t_{ij} \quad (2)$$

Let also $BEFORE(j)$ be the set of all nodes that should be visited before node j , and is defined as follows:

$$BEFORE(j) = \{k \in ND \mid EAT(j, k) > b_k\} \quad (3)$$

Nodes which can not be covered by set $BEFORE(j)$ will be added to set $FORBID(j)$.

- b. Arc elimination during running the algorithm. Given state (S, i, t) , $a_i \leq t \leq b_i$, the time to visit node $j \in N$ after node i is $t + s_i + t_{ij}$. The reachability time of the depot, $\{d\} \in ND$, after visiting node j is denoted by t' and is defined as $t' = \max(a_j, t + s_i + t_{ij}) + s_j + t_{j,d}$. Node j can be added to set S if all of the following tests are satisfied.

$$\begin{aligned} t + s_i + t_{ij} &\leq b_j \\ t' &\leq b_d \\ t' - t_{fr(S)} &\leq T \end{aligned} \quad (4)$$

Note that, In the arc elimination tests, as soon as the algorithm reaches a node j , which can be added to set S of state (S, i, t) , the set $S/BEFORE(j)$ will not be explored at any other state generated from that state. These nodes will be kept in the set $U(S)$. However, if node j can not meet any of the tests in (4), only node j will be kept in the set $U(S)$ and will not be explored at any other states generated from S .

2) *State elimination tests.* These tests are implemented during and after performing phase one.

- a. Given states (S,i,t_1) and (S,i,t_2) , the second state is eliminated if $t_1 \leq t_2$ and $F(S,i,t_1) \leq F(S,i,t_2)$.
- b. Given states (S,d,t_1) and (S,d,t_2) , the second state is eliminated if $F(S,d,t_1) \leq F(S,d,t_2)$. Test 2-b reduces the number of states passing to the next phase in order to reduce the computational time in phase two.

Phase Two: The outcome of the first phase is sets of feasible solutions (routes) with their associated costs. Let's assume that R routes were generated in Phase I. Let x_r be one if the route $r \in R$ is selected among the optimum routes, and zero otherwise. Associated with route r is the cost F_r which was calculated in the previous phase. Let also \mathbf{b}_r be equal to one if route r visits node $i \in N$, and zero otherwise. The set-covering problem can be formulated as follows:

$$\begin{aligned}
 \text{Min} \quad & \sum_{r=1}^R F_r x_r \\
 \text{Subject to} \quad & \sum_{r=1}^R \mathbf{b}_{ir} x_r = 1 \quad \forall i \in N \\
 & x_r = \{0,1\}
 \end{aligned} \tag{5}$$

In order to find the set of routes in (5) which covers all nodes in G exactly once with minimum total cost, the routes generated in Phase I are fed to another DP algorithm. That is, the second DP solves a set-covering problem in order to extract the optimum set of solutions among all sets of solutions. It should be mentioned that the set covering problem has also been proven to be NP-hard (Garey, and D.S. Johnson 1979).

In phase II, the state (St, v) is defined as follows: $St \subseteq N$ is an unordered set of visited nodes, and v is the number of routes (vehicles) forming the state St . Assigned to each state is a cost denoted by $g(St, v)$ and is defined as the least total cost of routes forming St (covering every node of St). Given states (St, v_1) and (St, v_2) , the second state is eliminated if $g(St, v_1) < g(St, v_2)$. Furthermore, if $g(St, v_1) = g(St, v_2)$ the second state is eliminated if $v_1 \leq v_2$.

The outcome of Phase II will be a set of routes covering all nodes in G exactly once with total minimum cost.

2.2.2. Genetic algorithms for m -TSPTW

Although the exact method in Subsection 2.2.1 is capable of finding the exact solution, the algorithm becomes computationally very costly for problems of medium or larger size. Note that the set covering problem is NP-hard. What we observed in our computational experiments is that the DP algorithm for the set covering problem (phase II) is computationally slow for problems involving around 20 or more nodes.

Hence, it is desirable to find a mechanism that results in a compromise between the quality of the solution and the computational time needed to obtain that solution. Here, we use genetic algorithms (GA) for solving the second phase in order to find an approximate solution for large size problems. The results are demonstrated and compared in Table 1.

It is worth mentioning that it is generally believed that GA is slow and would take time to find a high quality solution. The amount of computational effort required by this algorithm depends on the size of

the problem, i.e., the number of the nodes in graph G as well as the number of the generated feasible solutions in Phase I.

2.2.3. Insertion method

We also develop a sequential insertion heuristic methodology for large size m-TSPTW problems. In the insertion method, routes are generated one at a time. That is, a new route is initialized when all active routes are full. The developed algorithm searches for *feasible* insertions of customers into active routes, and then an *optimization* procedure is used to find the best feasible insertion location.

Feasible insertion procedure: Let $r = \{o, 1, \dots, i, j, j+1, \dots, n, d\}$ be a feasible solution (route) starting from origin $\{o\}$, visiting nodes in route r only once, and ending at destination $\{d\}$. Figure 1 illustrates a typical route r .

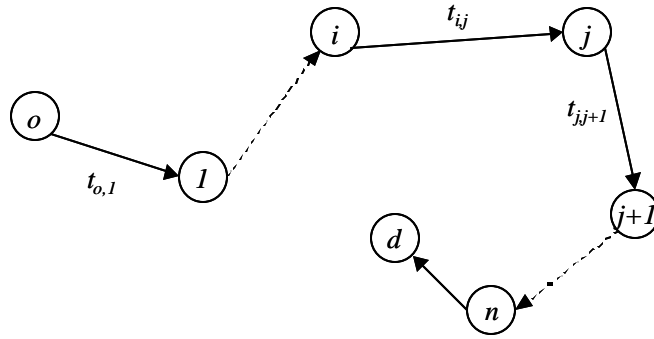


Figure 1: A graphical representation of a typical route.

Let A^r be the arc set associated with route (solution) r defined as $A^r = \{(i,j) / i,j \in r, \text{ and node } j \text{ is visited immediately after node } i\}$. Let also T_i be the time when service begins at node i . Recall that associated to each node $i \in r$ are a service time s_i and a time window $[a_i, b_i]$, and to each arc $(i,j) \in A^r$ are a cost c_{ij} and a time t_{ij} representing the cost and the time of traveling between nodes i and j , respectively.

Assuming that node j is visited after node i , the *waiting time* at node j denoted by w_j is defined as the duration of time at node j the vehicle has to wait before being served, and is given by

$$w_j = \max(0, a_j - (T_i + s_i + t_{ij})) \quad (6)$$

Similarly, the *excess time* at node i denoted by Δ_i is defined as the portion of the time-window (at node i) between the latest time to visit node i , b_i , and the time that the service has started at node i , and is given by:

$$\Delta_i = b_i - \max(T_i, a_i) \quad (7)$$

Let $\mathbf{d}t_{ij}$ be the disturbance (changes from the nominal value) occurs in the traveling time between nodes i and j . The *feasibility margin* of the solution r to the changes in traveling time in link (i,j) is

denoted by \mathbf{j}_{ij}^r and can be defined as the maximum value of $\mathbf{d}t_{ij}$ such that the solution r is still feasible. The feasibility margin of solution r with respect to the changes in the traveling time of the link (i,j) can be computed recursively as follows

$$\begin{aligned} \mathbf{j}_{ij}^r &= w_j + \text{Min}(\Delta_j, \mathbf{j}_{j,j+1}^r) \quad \forall (i,j) \in A^r \\ \mathbf{j}_{d,d+1}^r &= M \end{aligned} \quad (8)$$

where M is a big number. Node l can be inserted between each two consecutive nodes i and j in route r if the following inequalities are satisfied:

$$t_{il} + w_l + s_l + t_{lj} - t_{ij} \leq \mathbf{j}_{ij}^r \quad (9)$$

$$T_i + s_i + t_{il} \leq b_l \quad (10)$$

Equation (9) indicates that the time changes in route r caused by inserting node l between nodes i and j should be less than or equal to the \mathbf{j}_{ij}^r . Equation (10) ensures that node l is visited before the latest time b_l .

Optimization procedure: To each route r a cost C^r is associated which is the summation of the traveling costs between each two consecutive nodes on route r . In other words, C^r is the total cost of the traveling throughout all nodes of route r in the specified order. Let's assume that node l can be inserted between nodes i and j on route r , i.e., the inequalities (9) and (10) are met. The changes in the cost C^r due to insertion of node l between nodes i and j is denoted by ΔC_{ij}^r and is given by

$$\Delta C_{ij}^r = c_{il} + c_{lj} - c_{ij} \quad (11)$$

To insert a node l in route r , we first find all feasible insertion locations according to (9) and (10). Among all feasible locations (arcs) in route r , a candidate arc $(i,j) \in A^r$ is one which results in minimum changes in cost C^r . In other words, we are interested in finding

$$\text{Min}_{(i,j) \in A^r} \Delta C_{ij}^r \quad (12)$$

We then examine all routes r to find the best locations among all feasible locations to insert node l . That is we are interested in finding the best candidate among all candidates with minimum cost of insertion, i.e.,

$$\text{Min}_r \left\{ \text{Min}_{(i,j) \in A^r} \Delta C_{ij}^r \right\} \quad (13)$$

If such a candidate could not be found, a new route will be initialized. The node l , then, will be added to the newly generated route. The algorithm will be terminated when no more nodes left to be inserted.

2.3. Computational Experiments

The experimental tests consisted of a Euclidean plane in which customer coordinates were uniformly distributed between 0 and 7 hours, and travel times and costs equal distances. The coordinates of the depot were generated randomly between 3 and 4 hours using the uniform distribution function. The dimensions of the Euclidian plane and the location of the depot were selected such that every customer in the Euclidian plane can be reached and served by at least a truck, and the truck can go back to depot, within a working day, i.e., $T=10$ hours.

The 'time to start service' at each node was generated as a uniform random variable between 9:00 a.m. to 5:00 p.m. The time window interval length was generated as a uniform random variable in the interval $[0, w]$, where $w=2$ hours. The service time at each node was assumed to be a uniform random variable generated between 30 minutes to 2 hours. The time window at the depot is set between 6:00 a.m. till 20:00 p.m. and the service time at the depot is assumed to be zero.

Table 1: Comparing exact, hybrid GA, and insertion methods, $w = 2$ hours

No of nodes	Dynamic Programming		Genetic Algorithm ^a		Insertion method	
	Cost	CPU time	Cost	CPU time	Cost	CPU time
7	13.69	0.38	13.69	0.22	17.69	0.11
10	23.88	1.70	23.88	0.50	26.96	0.11
15	35.38	326.4	35.38	0.76	42.25	0.16
20	NA ^b	NA	52.44	15.86	59.91	0.27
30	NA	NA	98.97	61.61	107.0	0.49
50	NA	NA	179.4	191.91	194.45	1.45
100	NA	NA	338.9	1876	359.08	4.67

a) In average (based on the results of 10 trials).

b) NA: The result couldn't be obtained.

Table 1 summarizes and compares the cost and the CPU time of the experimental results for three different methodologies: the exact method, the hybrid Genetic Algorithm (GA) method, and the insertion method. The experiments were tested on an Intel Pentium 4, 1.6 GHZ, and were coded in Matlab 5.3 developed by Math Works, Inc. In Table 1 each set of customers (row) is built upon the previous row. For instance, for the number of customers (nodes) equal to 10, we used the same randomly generated customers for $N=7$ and added 3 newly generated customers. As shown in Table 1, exact method is efficient for relatively small size problems consisting of a few nodes. GA is capable of finding optimum solution for small size problems, and able to find 'good' solution for medium to large size problems (more than 30 nodes). Finally, The heuristic insertion method was able to find relatively good solutions for medium to large size problems (more than 50 nodes); furthermore, the method is computationally very fast.

3. STOCHASTIC ENVIRONMENT

3.1. Problem Description

Let's consider the graph $G=(ND,A)$ described in Subsection 2.1 again. In stochastic TSPTW associated with each arc $(i,j) \in A$, is a stochastic process $X_{ij}(t)$ with argument t representing the travel time on that arc. The argument t indicates the time when a vehicle enters the arc (i,j) . We assume that travel times on individual links at any particular time t are statistically independent.

A cost coefficient denoted by c_{ij} is associated to each arc $(i,j) \in A$. The cost c_{ij} can be a deterministic number (e.g., the distance between nodes i and j), or a random variable (e.g., the travel time on the arc (i,j)). Associated with each node $i \in ND$ is a service time s_i representing the duration of time for a vehicle to be served at node i . We assume that the service time s_i is a random variable, which is independent of the time that the service starts at node i . As before, to each node $i \in ND$ a hard time window $[a_i, b_i]$ is associated where a_i and b_i are the earliest and the latest time to visit node i , respectively.

3.2. Estimating the first two moments of the arrival time

Let $r=\{o,1,\dots,i,j,l,\dots,n,d\}$ be a route in graph G , as shown in Figure 1, and A^r be the arc set associated with route r defined as $A^r=\{(i,j) \mid i,j \in r, \text{ and } j \text{ is visited immediately after } i\}$.

3.2.1. Without service time and time windows

Let's start by assuming that there are no time windows and service times associated with nodes of route r , i.e., $[a_i, b_i] = [0, \infty)$ and $s_i = 0 \quad \forall i \in r$. In other words, we assume that as soon as a vehicle arrives at node i it immediately continues its travel toward node j . Let $\mathbf{h}_j(t)$ and $\mathbf{s}_{ij}^2(t)$ denote the mean and variance of the stochastic process $X_{ij}(t)$ corresponding to its first-order density function, respectively. Given the mean of the arrival time at node i , the mean of the arrival time at node j , the immediate node after i can be computed by (Papoulis, 1991).

$$E[Y_j^r] = E[Y_i^r] + E[\mathbf{h}_{ij}(Y_i^r)], \quad (14)$$

Using Taylor's series expansion, and assuming that the function $\mathbf{h}_{ij}(t)$ is differentiable at and around $E[Y_i^r]$, the first order approximation of (14) can be obtained by (Fu and Rilett, 1998)

$$E[Y_j^r] \cong E[Y_i^r] + \mathbf{h}_{ij}'(E[Y_i^r]) \quad (15)$$

Similarly, by using Taylor's series expansion, the first order approximation of the second moment of Y_j^r can be obtained by (see also, Fu and Rilett 1998, and Papoulis 1991)

$$\text{var}(Y_j^r) \cong (1 + \mathbf{h}_{ij}'(E[Y_i^r]))^2 \cdot \text{var}(Y_i^r) + \mathbf{s}_{ij}^2(E[Y_i^r]), \quad (16)$$

where $\mathbf{h}'_j(\cdot)$ is the first derivative of $\mathbf{h}_j(\cdot)$. Therefore, given the departure time Y_o and the mean and variance of the travel time on each arc $(i, j) \in A^r$, the expected value and variance of the arrival time at each node $i \in r$ can be calculated using equations (15) and (16), recursively.

3.2.2. With service time

Let random variable S_i denote the service time at node i . Given arrival time at node i taking route r , Y_i^r , the first moment of the departure time from node i , W_i^r , is obtained by

$$E[W_i^r] = E[Y_i^r] + E[S_i]. \quad (17)$$

Since random variable S_i is independent of the arrival time Y_i^r , we have $\text{cov}(Y_i^r, S_i) = 0$. Hence, the second moment of W_i^r can be computed as follows:

$$\text{var}(W_i^r) = \text{var}(Y_i^r) + \text{var}(S_i), \quad (18)$$

Given the mean and variance of the departure time W_i^r in (17) and (18), the first order approximation model of the mean and variance of Y_j^r can be obtained according to equations (15) and (16) by

$$E[Y_j^r] \cong E[W_i^r] + \mathbf{h}_{ij}(E[W_i^r]) \quad (19)$$

$$\text{var}(Y_j^r) \cong (1 + \mathbf{h}'_{ij}(E[W_i^r]))^2 \cdot \text{var}(W_i^r) + \mathbf{s}_{ij}^2(E[W_i^r]), \quad (20)$$

Hence, given the departure time Y_o , the mean and variance of the travel time on each arc $(i, j) \in A^r$, and the mean and variance of the service time at each node $i \in r$, the first two moments of the arrival time at each node i can be calculated using equations (17) to (20), recursively.

3.2.3. With hard time windows

Without loss of generality and for the sake of simplicity, we assume that there is no service time associated with nodes of route r . Let $g(y_i)$ denote the departure time as a function of the arrival time y_i in the case of hard time windows at each node i . Figure 2 illustrates the non-linear function $g(\cdot)$ at node i , graphically.

As shown in Figure 2, given the arrival time Y_i^r at node i taking route r , the departure time W_i^r is given by $W_i^r = g(Y_i^r)$. If a vehicle arrives at node i earlier than a_i , the vehicle waits till the time $t = a_i$ before departing. However, if the arrival time at node i is later than b_i the departure time will be

$M \cdot y_i^r$ where M is a large number. Note that the variable M , in Figure 2, is considered to make sure that the departure time W_i^r is a random variable, see also Papoulis (1991).

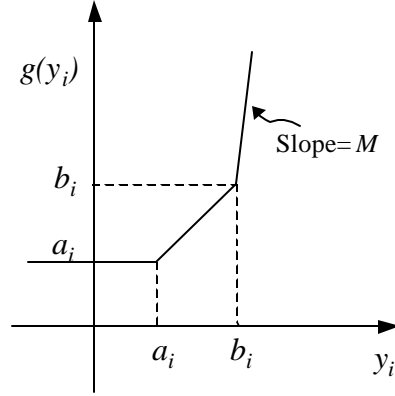


Figure 2: Model of the hard time window at node i , the graphical representation.

To estimate the first and second moments of random variable W_i^r in terms of function $g(\cdot)$ and moments of random variable Y_i^r , we approximate the function $g(Y_i^r)$ at or around $E[Y_i^r]$. Using Taylor's series expansion, the first and second central moment of random variable W_i^r can be obtained by

$$E[W_i^r] = g(E[Y_i^r]). \quad (21)$$

$$\text{var}(W_i^r) = (g')^2(E[Y_i^r]) \cdot \text{var}(Y_i^r). \quad (22)$$

Equations (21) and (22) lead to estimating the mean and variance of the departure time W_i^r at node i in terms of function $g(\cdot)$ and the mean and variance of the arrival time Y_i^r . It should be noted that Equation (22) is valid as long as

$$\begin{aligned} |E[Y_j^r] - a_j| &\gg \mathbf{s}(Y_j^r), \text{ and} \\ |E[Y_j^r] - b_j| &\gg \mathbf{s}(Y_j^r), \end{aligned} \quad (23)$$

hold, where $\mathbf{s}(Y_j^r)$ denotes the standard deviation of the arrival time at node j . For the cases in which the inequalities in (23) don't hold, we approximate the value of the standard deviation $\mathbf{s}(W_j^r)$ by

$$\mathbf{s}(W_j^r) \approx \frac{1}{2} \cdot \int_{E[Y_j^r] - \mathbf{s}(Y_j^r)}^{E[Y_j^r] + \mathbf{s}(Y_j^r)} g'(x) dx \quad (24)$$

3.3. Proposed solution method for the Stochastic TSPTW

Let's consider route r in Figure 1, and let Y_i^r denote the arrival time at node i taking route r .

Definition (confidence coefficient at node i on route r): The confidence coefficient at node i on route r , denoted by g_i^r , is defined as the probability of arriving at node i before the latest time to visit node i , b_i , i.e.

$$g_i^r = P\{Y_i^r < b_i\} \quad (25)$$

We say that the arrival time at node i taking route r meets the required confidence level at this node, say j_i , if the confidence coefficient at node i , g_i^r , is greater than j_i , i.e., $g_i^r \geq j_i$. Without loss of generality, hereafter, we assume that the confidence level at all nodes are equal, i.e., $j_i = j \quad \forall i \in ND$. The route r is said to be 'feasible' if the arrival time at all nodes $k \in \mathcal{I}_r$ meet the confidence level j .

In the stochastic TSPTW problem, the objective is to find the optimum, the least-cost, feasible route starting from $\{o\}$, visiting all nodes in N and ending at destination $\{d\}$, such that each node is visited exactly once.

Here, we propose an approximate solution method for the stochastic TSPTW. We assume that the mean and variance of the traveling time $X_{ij}(t)$ on arc $(i,j) \in A$, denoted by $h_{ij}(t)$ and $s_{ij}^2(t)$, are given.

We also assume that the mean and variance of the service time s_i at each node $i \in ND$ denoted by $E[s_i]$ and $var(s_i)$, respectively, are known a priori. Without loss of generality, here we assume that the cost of traveling on a route equals to the mean travel time on that route.

Let the state (S,i) be defined as $S \subseteq N$ is an unordered set of visited nodes, and $i \in \mathcal{I}_S$ is the last visited node. Associated to each state are:

- The mean and variance (denoted by $E[Y_i^S]$ and $var(Y_i^S)$, respectively) of the arrival time at node i taking the path starting at origin $\{o\}$ passing through every node of S exactly once and ending at node i ,
- A cost denoted by C_i^S , defined as the cost of traveling on the path described above, and
- A confidence coefficient g_i^S , defined in (25).

Definition (acceptable arc): Let the arrival time at node i meet the required confidence level, i.e., j . An arc $(i,j) \in A$ is said to be 'acceptable' if the arrival time at node j traveling on arc (i,j) also meets the required confidence level.

Definition (state extendibility): Let node $i \in S$ be the last visited node of the state (S,i) . The state S is said to be expandable to node j , $j \in ND$ and $j \notin \{S,o\}$, if the arc $(i,j) \in A$ is 'acceptable'.

In order to reduce the computational time, two types of elimination tests are performed: arc elimination, and state elimination.

- 1) *Arc elimination test:* The arc elimination test looks one step ahead to reduce the number of states. Let's assume that the state (S,i) is extendible to node j . The state (S,i) will be

extended to node j if all arcs (j,k) , $k \in ND/\{S, j, o\}$, are acceptable. Otherwise the state will be eliminated.

- 2) *State elimination test.* This test implements the dynamic programming algorithm to reduce the number of states. Given states (S_1, i) and (S_2, i) , where S_1 and S_2 cover the same set of nodes in ND , the second state is eliminated if $E[Y_i^{S_1}] \leq E[Y_i^{S_2}]$, $\text{var}(Y_i^{S_1}) \leq \text{var}(Y_i^{S_2})$, and $C_i^{S_1} \leq C_i^{S_2}$.

3.4. Computational Experiments

The experimental test consists of a Euclidean plane in which node coordinates are uniformly distributed between 0 and 50, and the coordinates of the depot is at 0 and 0. The mean travel time equals distance, and the standard deviation of the travel time is set one tenth of the mean travel time. We assume no service time associated with each node.

The time window at each node is generated around the time to begin service at that node according to the first nearest neighbor TSP tour (Reinelt, 1994). That is, assuming travel time equals distance, we found the best deterministic TSP tour based on the first nearest neighbor heuristic algorithm. Accordingly, the time to reach node i taking the generated tour, say T_i , is calculated and the time window $[a_i, b_i]$ is generated around this time by

$$a_i = \max\left(0, T_i - \frac{w}{2}\right), \quad (26)$$

$$b_i = T_i + \frac{w}{2}, \quad (27)$$

where $w=20, 30, 40, 60,$ and 80 minutes.

The proposed solution method is applied to the generated graph for confidence level, $\beta=90\%$ and by using the Tchebycheff bound. Table 2 presents the experimental results with a different number of nodes (customers), N , and different time window widths, w . The cost is equal to the mean travel time. The second column in Table 2 shows the deterministic cost of taking the first nearest neighbor TSP solution. This solution, however, may not be feasible for the stochastic TSPTW. The algorithm was coded in Matlab 5.3 developed by Math Works, Inc.

In Table 2, each set of nodes (row) is built upon the previous row. For instance, for the number of nodes equal to 30, we kept the same randomly generated nodes for $N=20$ and added 10 newly generated ones. A new first nearest neighbor TSP tour is then found and a time window is assigned to each node according to (26) and (27).

The results in Table 2 indicate that the approximate algorithm is successful in solving problems up to 80 nodes with fairly wide time windows. As expected, the CPU time increases with time window width and with problem size.

Table 2: Computational results for stationary, stochastic TSPTW using Tchebycheff bound with $\beta=90\%$.

No of nodes	Init ^a	w=20 Min		w=30 Min		w=40 Min		w=60 Min		w=80 Min	
		Best ^b	CPU ^c	Best	CPU	Best	CPU	Best	CPU	Best	CPU
20	180.6	178.7	2.1	178.7	8.1	178.7	8.1	178.0	17.8	178.0	97.6
30	267.6	265.2	10.6	262.8	28.3	262.8	40.3	255.9	120.5	254.6	477.5
40	289.9	288.8	42.4	281.1	79.04	280.6	137.4	274.6	515.5	270.7	1511
50	309.1	306.3	69.4	304.1	135.4	304.1	256.8	301.0	911.0	NA ^d	NA
60	382.2	376.6	123.5	374.3	253.7	374.3	493.9	NA	NA	NA	NA
80	455.3	445.7	1042	440.3	2739	NA	NA	NA	NA	NA	NA

- a) The cost of initial route generated by deterministic first nearest neighbor TSP heuristic.
- b) The cost of the best solution using the proposed approximate STSPTW method.
- c) The CPU time in seconds on a Pentium4 (1.6 GHz) personal computer.
- d) NA: Not Available, i.e., no route could be found.

4. CONCLUSIONS

In this paper, we investigated the methods for truck scheduling and route planning, where ISO containers need to be transferred between marine terminals, intermodal facilities, and end customers, and need to be delivered/picked up within a specified time period. The problems studied fall in the class of full-truck-load vehicle planning, routing and scheduling with appointment window system. The problem can modeled by the Traveling Salesman Problem with Time Windows (TSPTW). In this research, we developed three methodologies for solving the deterministic multi-TSPTW:

- An exact two-phase Dynamic Programming (DP). The method consists of two phases: 1) generating feasible solutions, and 2) finding the optimum solution among all feasible solutions (set-covering problem). Computational experiments show that the proposed exact method can optimally solve problems with up to 15-20 nodes on randomly generated problems.
- A hybrid methodology consisting of DP in conjunction with genetic algorithms (GA). The GA algorithm is used to find a 'good' solution among all feasible solutions. Experimental results show the efficiency of the GA set-covering algorithm for medium to large size problems (about 50 nodes).
- A heuristic insertion method. The method involves inserting nodes into the routes sequentially. Experimental results show that the method is computationally very fast and it is fairly efficient for medium to large size problems (more than 50 nodes). The heuristic insertion method was able to find relatively good solutions for medium to large size problems; furthermore, the method is computationally very fast.

In the second part of this paper, we considered the Stochastic Traveling Salesman Problem with Time Windows (STSPTW) in which the travel times along the arcs and the service times at nodes of the

network are non-stationary stochastic processes. One of the major difficulties in solving this STSPTW problem is the existence of non-linearity formed by the hard time windows at nodes. This non-linearity is thoroughly investigated and an approximate methodology is developed to address the existing hard time windows in the STSPTW problem. We proposed an approximate solution method algorithm for the STSPTW problem with stochastic travel and service times. The results also show that the approximate algorithm was successful in solving stationary STSPTW problems up to 80 nodes with fairly wide time windows.

References

- M.O. Ball, T.L. Magnati, C.L. Monma, and G.L. Nemhauser (eds), *Network Routing*, Vol. 8, *Handbooks in Operations Research and Management Science*, pp. 35-130, Elsevier Science, Amsterdam (1995).
- M. Desrochers, J.K. Lenstra, M.W.P. Salvelsbergh, and F. Soumis, "Vehicle routing with time windows: optimization and approximation," in *Vehicle Routing: Methods and Studies*, B.L. Golden and A.A. Assad (eds), 65-84, North Holland Publication, Amsterdam, (1988).
- M. Desrochers, J. Desrosiers, and M. Solomon, "A new optimization algorithm for the vehicle routing problem with time windows," *Operations Research*, vol. 40, 342-354 (1992).
- Y. Dumas, J. Desrosiers, E. Gelinas, and M.M. Solomon, "An optimal algorithm for the traveling salesman problem with time windows," *Operations Research*, vol. 43, no. 2, 367-371 (1995).
- L. Fu and L. R. Rilett, "Expected shortest paths in dynamic and stochastic traffic networks," *Transportation Research – Part B*, vol. 32, no. 7, 499-516 (1998).
- M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco: W.H. Freeman, 1979.
- B.L. Golden and A.A. Assad (eds), *Vehicle Routing: Methods and Studies*, North Holland Publication, Amsterdam (1988).
- H. Jula, M. Dessouky, P. Ioannou, and A. Chassiakos, "Container movement by trucks in metropolitan networks: modeling and optimization," Working Paper (2002).
- N. Kohl, J. Desrosiers, O.B.G. Madsen, M.M. Solomon, and F. Soumis, "2-path cuts for the vehicle routing problem with time windows," *Transportation Science*, vol. 33, no. 1, 101-116 (1999).
- G. Laporte, "The vehicle routing problem: an overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, 345-358 (1992).
- A. Papoulis, *Probability, random variable, and stochastic processes*, McGraw-Hill, New York, Third Edition (1991).
- G. Reinelt, *The Traveling Salesman: Computational Solutions for TSP Applications*, Vol. 840, *Lecture Notes in Computer Science*, Springer-Verlag, New York (1994).
- N.K. Ryan, "The future of maritime facility designs and operations," *Proceeding of 1998 Simulation Conference*, pp. 1223-1227, (1998).
- M.W.P. Savelsbergh and N. Sol, "The general pickup and delivery problem," *Transportation Science*, vol. 29, no. 1, 17-29, (1995).
- M.J. Vickerman, "Next-Generation Container Vessels: Impact on transportation Infrastructure and Operations," *TR News*, vol. 196, pp. 3-15, May-June, (1998).