Final Report

Building a Model-Based Decision Support System for Solving Vehicle Routing and Driver Scheduling Problems under Hours of Service Regulations

by

Hokey Min*

Affiliation: James R. Good Chair in Global Supply Chain Strategy
Department of Management, BAA 3008C
College of Business Administration
Bowling Green State University
Bowling Green, OH 43404
Telephone: 419-372-3442 (office)
419-372-6057 (fax)
E-mail: hmin@bgsu.edu

Prepared on August 27, 2011

* Principal Investigator

Executive Summary

In principle, hours-of-service (HOS) regulations are intended to help truck drivers ensure get adequate rest and perform safe operations. The new HOS regulations, however, may lead to substantial cost increases for regional common carriers which have already been hit hard by rising fuel prices and declining shipping demands. In addition, the new HOS regulations complicate driver schedules by not only restricting the driver's consecutive driving hours, but also expanding off-duty hours. To deal with this complex challenge, we propose a model-based decision support system (DSS) that helps to determine the truck driver's working hours, rest periods, and his/her assigned truck's schedules and routes under HOS regulations. As a core of this model-based DSS, we developed a mixed-integer programming model and a simulated annealing (SA) meta-heuristic for solving it. This model was also integrated with a geographical information system (GIS) and relational database management system (RDBMS) to enhance interfaces between the model and its parametric data using spatial, graphical displays. The practicality and robustness of the proposed model-based DSS were validated with its successful application to the real-world problem encountering a regional common carrier in the United States.

1 BACKGROUND

The leading cause of truck accidents is driver fatigue. In fact, more than 750 people die each year and more than 20,000 are injured each year because of fatigued truck drivers operating commercial trucks (Carpey, 2010). Reissman (1997) discovered that drowsy drivers were responsible for 50% of the fatal vehicle crashes on the Pennsylvania Turnpike and New York Thruway. This alarming statistic is not surprising given that drivers who have been awake for 24 hours have an equivalent driving performance to a person who has a BAC (blood alcohol content) of 0.1 g/100ml and thus are seven times more likely to have an accident (http://www.smartmotorist.com/traffic-and-safety-guideline/driver-fatigue-is-an-important-cause-of-road-crashes.html, 2008). Recognizing the seriousness of driver fatigue to highway safety, Federal Motor Carrier Safety Administration (FMCSA, 2005) has attempted to implement new hours-of-service (HOS) regulations. The new HOS regulations effective on October 1st of 2005, however, may lead to substantial cost increases for the trucking industry which will in turn hurt shippers and ultimately customers. For instance, the trucking industry may need to hire 84,000 additional drivers to comply with the new HOS rules requiring that drivers be placed out-of-service until they accumulated enough off-duty time (Min, 2009). To elaborate, off-duty breaks required to refresh driving hours were increased to 10 consecutive hours from the old rule of 8 cumulative hours. A chronic shortage of truck drivers coupled with new HOS regulations could further complicate the driver scheduling problems. In addition, due to potential loading/unloading delays and stiffer fines (between $550 and $11,000 per HOS violation depending on severity), trucking firms will be faced with daunting challenges of controlling mounting costs while complying with new HOS regulations.

In an effort to help trucking firms cope with these challenges, this paper aims to develop a model-based decision support system (DSS) as a useful decision-aid tool. The heart and soul of this DSS is a mathematical model and its solution procedure that can minimize transportation costs and avoid driver schedule conflicts due to HOS regulations. The proposed model takes the form of the combined time dependent vehicle routing and driver scheduling problem that is concerned with finding an optimal route/schedule of capacitated vehicles and an optimal assignment of drivers to such vehicles over a number of designated delivery points within pre-specified time windows. The

combined time dependent vehicle routing and driver scheduling problem may arise in many real-world situations such as just-in-time (JIT) delivery services, overnight trucking services, express parcel delivery services, and school bus services. Similar to other well-known vehicle routing and scheduling problems, the combined time dependent vehicle routing and driver scheduling problem is extremely difficult to solve due to its combinatorial nature and added complications, such as time windows and simultaneous scheduling of both drivers and their vehicles. As such, the problem calls for heuristic solution procedures that can handle practical-size problems faced by many trucking firms. In this paper, we propose simulated annealing (SA) meta-heuristics as a way to solve the combined time dependent vehicle routing and driver scheduling problem.

2 PRIOR LITERATURE

Although there exists abundant literature dealing with various forms of vehicle routing and scheduling problems, relatively few attempts have been made to solve time dependent vehicle routing and scheduling problems (TDVRSP) that are concerned with the determination of optimal routes by considering the time it takes to traverse each given arc depending on the time of the day. Some notable examples of these attempts include: Malandraki and Daskin (1992), Ichoua et al. (2003), Fleischmann et al. (2004), and Donati et al. (2006). To elaborate, Malandraki and Daskin (1992) proposed a mixed integer programming model and cutting plane heuristics that took into account time windows and the maximum allowable duration for each route (e.g., work schedules of the driver). The travel distance of arc (i, j) is a time dependent step function as the speed of the vehicle does not remain constant due to the variable traffic density. Their work was extended by Donati et al. (2006) who developed ant colony optimization meta-heuristics to improve the computational efficiency and accuracy of the solution procedure. Similar to most of the other studies dealing with TDVRSP, these two studies, however, did not take into account simultaneous vehicle routing and driver scheduling problems. For the further details of TDVRSPs that were solved in the past, the interested readers should refer to Bodin et al. (1983) and Solomon and Desrosiers (1988).

Xu et al. (2003) were among the first to consider drivers' work rules for a multiple vehicle routing and scheduling problem with time windows. In addition, they imposed a set of compatibility constraints between orders and vehicles as well as precedence constraints for loading and unloading

orders. Each vehicle trip must satisfy legal working hours of a driver as prescribed by the Department of Transportation (DOT). To solve this complicated problem, they formulated a set partitioning model and developed a column generation procedure. However, their formulation did not explicitly take into account HOS regulations and a varying speed of the vehicle. Similar to Xu et al. (2003), Goel and Gruhn (2006) considered drivers' labor rules in the European Union. They formulated a vehicle routing problem with time windows to incorporate driver scheduling issues into the vehicle routing/scheduling problem. Their study, however, did not explicitly consider the variable vehicle speed during the day. More recently, Archetti and Savelsbergh (2007) took into account HOS rules in the trip scheduling problem. Given a sequence of n transportation requests with dispatch windows at the origins, they determined in $O(n^3)$ polynomial time a driver's schedule (if it exists), i.e., driving times and rest times, so that the origins could be visited in the given sequence and within their dispatch windows, or otherwise it was found that such a feasible schedule did not exist. However, their study neglected time dependent travel times and loading/unloading times at customer locations. In addition, their model did not consider complex HOS regulations, such as scheduling of restorative breaks during the day, although they could handle HOS regulations concerning night time rest.

To go beyond these prior studies, we attempted to solve a vehicle TDVRSP under the most recent HOS rules. The main features of the proposed model and solution procedure are:

- Until recently, most of the existing vehicle routing and scheduling literature focused solely on the minimization of travel distances, travel time or transportation cost under the premise of constant vehicle speed. Similar to Malandraki and Daskin (1992), Ichoua et al. (2003) and Donati et al. (2006), we developed a step function with consecutive time intervals that took into account changes in vehicle speed due to traffic congestions and road accidents. In other words, the proposed model and solution procedure can factor variable vehicle speed into driving time and thus prevent sub-optimal or infeasible vehicle and driver schedules.
- Considering that driver's working hours may not match the available vehicle schedules, we attempted to simultaneously coordinate both driver and vehicle schedules. These attempts are very rare in the literature due to inherent computational complexity involved in simultaneous driver and vehicle scheduling.
- As in the earlier works of Xu et al. (2003), Goel and Gruhn (2006), and Archetti and

Savelsbergh (2007), we incorporated HOS regulations into the proposed model and solution procedure. Especially, unlike those earlier attempts, the proposed model explicitly considered the most recent HOS regulations enacted on October 1st of 2005.

As specified above, the proposed model and solution procedure are capable of capturing the aforementioned realistic dimensions, while other existing models could not. More importantly, a vast majority of these prior modeling efforts aimed to develop a stand-alone mathematical model whose efficiency relies on the accuracy of available data. In other words, these stand-alone models lacked interfaces between databases and models and subsequently posed difficulty in developing what-if scenarios in the case that model parameter values changed over time. To overcome such a weakness, we propose a model-based decision support system (DSS) linked to geographic information systems (GIS). In particular, the model-based DSS is proven to be useful for handling complex transportation problems as evidenced by some prior studies (e.g., Basnet et al., 1996; Benyahia and Potvin, 1998; Min, 1998; Gayialis and Tatsiopoulos, 2004; Ruiz et al., 2004).

3 MODEL-BASED DSS ARCHITECTURE

To assist truck dispatchers in developing the most desirable driver work plan and truck routes/schedules under HOS regulations, we propose a model-based DSS whose architecture is graphically displayed in Figure 1. In general, a decision support system (DSS) is referred to as an interactive, computer-based system that helps decision makers improve the quality of decisions and solve unstructured or semi-structured problems with an aid of timely data, computerized models, and expert knowledge (see Keen and Scott Morton, 1978; Turban and Aronson, 2001, for definitions of DSS). Extending the DSS framework of Sprague and Carlson (1982), the proposed model-based DSS is comprised of three components: (1) database enhancing the access of accurate, timely data necessary for model development; (2) model base developing a computerized optimization model and its solution procedure to determine the driver work plan, rest periods, and the truck's specific routes/schedules in compliance with HOS regulations; (3) dialogue base generating "if-then" or "what-if" rules for changes in delivery schedules, traffic congestion, customer bases, and HOS rules.

3.1 Relational Database Management Subsystem

A model is only as good as the quality of the data that support it (Napolitano, 1998). To enhance data quality and avoid data redundancy, we developed a relational database that contains two data sources: public and private. The relational database management subsystem (RDBMS) is designed to supplement standard operating systems by allowing greater integration of data, complex file structure, quick retrieval and changes, and better data integrity/security, while presenting data to the user in tabular forms after its normalization (Turban and Aronson, 2001; Ramakrishnan and Gehrke, 2003). Public sources include regulatory guidelines and reports issued by federal (e.g., U.S. DOT, FMCSA, Federal Highway Administration, The Bureau of Transportation Statistics), International Road Traffic and Accident Database, and the American Trucking Association. Private sources include a trucking firm's proprietary data files (e.g., customer files, manifests, logbooks, and bills of lading) available from its traffic divisions. In addition to raw data that were obtained from the above sources, we identified more specific data categories that are relevant to truck dispatch/routing and driver scheduling. These categories are:

3.1.1 Temporal data

Time is one of the primary concerns of truck and its driver scheduling. Specific elements of time that are significant to truck and its driver scheduling include: total vehicle travel (transit) time, loading/unloading time, off-duty hours including meal time at the rest stops, cumulative hours at the sleeper berth, cumulative driving hours, vehicle down time due to breakdown and delivery/pickup time windows.

3.1.2 Traffic data

Since unexpected traffic congestion can delay the delivery/pickup schedules, dispatchers should compile necessary traffic data before scheduling trucks and assigning drivers to those trucks. These data include: severity of traffic congestions on major highways (both interstate and local) during a given time of the day; locations of ongoing road construction and repairs; alternate routes that allow truckers avoid chronic traffic jams; pavement conditions of the roads that affect driving hours; toll roads; availability of rest stops; and number of ton-miles travelled by the truck without accidents.

3.1.3 Work schedule data

Since the driver's work schedule is affected by a multitude of factors, specific data that are associated with these factors should be collected. These data include: driver's request for home visits

given a weekly shift; driver's physical health and age; delivery/pickup schedules; needs for expedited shipments; relay driving; the number of available drivers; and the vehicle maintenance schedule.

3.1.4 Regulatory data

Since truck safety rules including HOS regulations have been constantly changing, dispatchers should update regulatory databases. For instance, the National Conference of State Legislatures work in cooperation with the National Highway Traffic Safety Administration to bring the motor carriers up to date, real time information about traffic safety bills and regulations. Such information should be stored and collected systematically to comply with changing traffic safety rules and regulations. Also, trucking firms should be aware of changes in FMCSA Standards so that they may avoid the use of trucks that are more prone to incur safety failures and subsequently cause accidents.

3.2 Model Management Subsystem

The complexity involved in determining the optimal routes of vehicles and work schedules of their drivers call for systematic decision-aid tools. As such tools, we propose a mixed integer programming model complemented with metaheuristics and a geographic information system (GIS). The main focus of the model management subsystem is the incorporation of the mixed integer programming model (problem solver) into the DSS. In particular, GIS can help truck dispatchers visualize traffic flows, logistics infrastructure (e.g., terminals, rest stops) available in the nationwide road network, and delivery/pickup locations. In general, GIS aims to simplify the data display mechanism by separating data representation from data storage and then allowing the decision maker to visualize how distinctive one geographic site is from another by superimposing spatial information, such as customer locations, highway networks, weather, and road structures on a map. Combined with relational data base management systems, GIS may provide a friendly platform for the enhancement of dialogue among truck dispatchers and model builders (Min and Zhou, 2002).

3.3 Dialogue Management Subsystem

At best, the model is an abstraction of real-world situations. Consequently, it cannot capture reality without running it more than once (Dyer and Mulvey, 1983; Min, 1989). Thus, the model should enable truck dispatchers to evaluate "what-if" scenarios associated with modifications in the HOS

rules and the truck firm's work and delivery plans (e.g., changes of company priorities from cost minimization to on-time delivery services, responses to truck driver shortages). In other words, the model's successful implementation depends on its flexibility for contingency planning. To enhance the model flexibility, the results of the model runs should be reported in user-friendly formats. These formats include standardized reports such as spreadsheets and tables summarizing cost saving opportunities and figures depicting customer zones, transportation networks, and road maps and structures. These formats also can be posted on the web-sites through hypertext links. The web-site posting of the model results will increase interaction among the users in remote locations. To further enhance interactions with the model and the data, the dialogue base is designed to answer ad hoc queries through an application programming interface and structured query language (SQL).

4 PROBLEM SCENARIO

Consider a truck (tractor-trailer) with a full truckload of goods that should be delivered to a number of customer locations across the United States. Since driver fatigue is one of the leading causes of truck crashes, FMCSA stipulated a series of hours of service regulations that restrict consecutive hours of driving and mandate minimum hours of restorative breaks and sleeps. For example, revised HOS regulations of 2005 require the driver to:

- Drive a maximum of 11 hours after 10 consecutive hours off duty;
- Not exceed 14 hours on duty, following 10 consecutive hours off duty;
- Not drive after 60/70 hours on duty in 7/8 consecutive days. A driver may restart a 7/8 consecutive day period after taking 34 or more consecutive hours off duty;
- Take 10 hours off-duty at the sleeper-berth, but may split sleeper-berth time into two periods provided neither is less than 2 hours.

The complexity of these regulations coupled with chronic driver shortages creates a scheduling nightmare for truck dispatchers who are responsible for scheduling the driver's working hours in such a way that he/she can spend more time at home, while meeting their customers' delivery deadlines. Since driver schedules cannot be completed without assigning each driver to an available truck, both driver and truck schedules should be coordinated together and developed simultaneously. In other words, a driver is assigned to the same truck during his entire duty hours

and his/her schedules are tightly dependent on predetermined truck schedules and routes or vice versa. In a nutshell, the combined truck routing and driver scheduling problem under HOS regulations (CTRDSP-HOS) is primarily concerned with the minimization of the total working hours of drivers and travel time of trucks given a set of routes within a fixed time horizon. Each route has a fixed starting time and is assigned to a truck and a driver from a certain set of domiciles. Each route contains a number of delivery nodes (i.e., customer locations), whose visiting order is to be determined and where partial shipments will be dropped off within specific time windows set by customer delivery requirements. Herein, notice that the arrival time at a customer location depends on the departure time from the preceding customer location. It is also affected by a myriad of factors: (1) local traffic congestion and speed limits on the roadway network between customer locations; (2) required restorative breaks during the trip between those two locations; (3) unexpected unloading delays at the preceding customer location. Furthermore, in an effort to increase more time at home for drivers, remote domiciles can be used to increase the drivers' access to their domiciles. These unforeseen circumstances and realities of HOS regulations may necessitate a modification of truck routes in such a way that driver waiting is minimized while early and/or late arrivals are avoided.

In particular, to reflect a variable vehicle speed during the day, we develop a step function that can create speed distribution with 24 time intervals where each time interval corresponds to a particular hour of the day. The step function can be mathematically expressed as: $c(t) = c_{ij}^{h}$, $h \leq t \leq h+1$, where $c(t)$ is the speed distribution at time t of the day and c_{ij}^{h} is the speed of the vehicle during the h^{th} hour (i.e., h = 0, 1, 2, 3…, 23) for a particular transportation link (i, j). For higher speed accuracy smaller than an hour time interval can be selected. However, higher accuracy has to be traded with increasing data collection and computation effort. Using the variable vehicle speed function, the travel time on a link (i, j) can be calculated given departure time from node i. The derived travel times satisfy the FIFO property (Ichoua et al., 2003). As described above, a determination of what sequence a vehicle should traverse is complicated by customer requests for timely delivery, varying travel speed, and compliance with HOS regulations. Thus, the CTRDSP-HOS is considered a special case of the time-dependent, multiple vehicle routing/scheduling problem with an added complexity of a driver scheduling problem subject to HOS regulations.

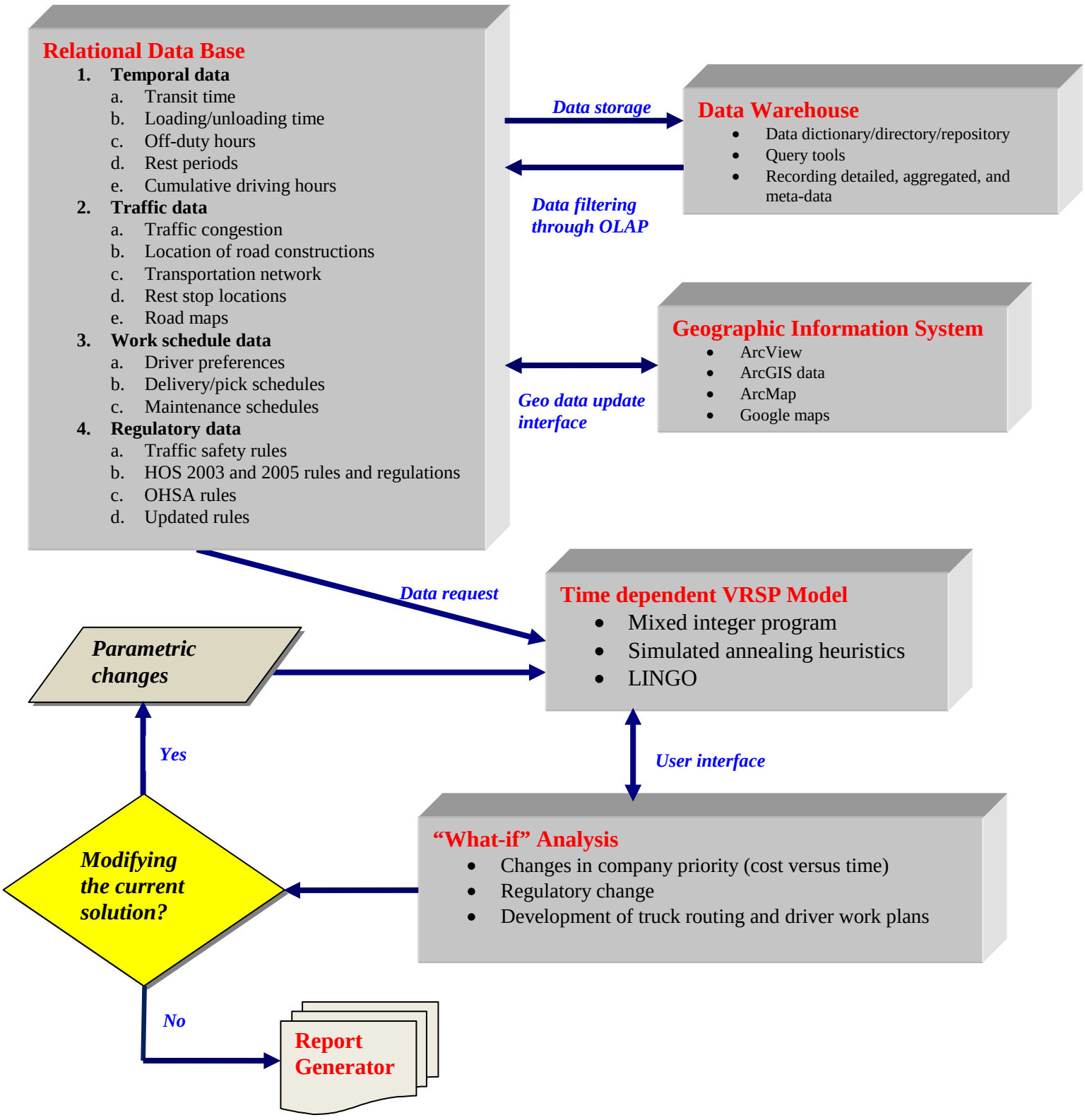

Figure 1. The Basic Architecture of a Model-based Decision Support System

5 MODEL DESIGN

The truck starts its tour at the depot, visits all customers, and returns to the depot. Each customer is associated with a node on the transportation network. There are many paths a truck can take from one customer node to another. Path traversal times depend on the time of departure from the customer node and the state of the driver regarding accumulated driving and duty times. Although it may be the same geographical location, the depot is associated with two nodes, the origin depot (k = 0) and the destination depot ($k = n+1$). The objective function is to minimize total time to complete a tour, starting at node 0, visiting n customer nodes and terminating at node $n+1$. It consists of travel times, waiting times and service times at the customer nodes, and needed break times during the trip.

5.1 Model Assumptions

Prior to developing a mathematical model, we made the following assumptions:

1. The driver can be assigned to only one truck. Also, relay driving is not considered.
2. Service (unloading) time at each customer node is known a priori, and once it starts, is not allowed to be interrupted by a driver's restorative break.
3. Daily restorative breaks can be taken at any locations during the designated trip.
4. Travel distances/times between two nodes are non-Euclidean and asymmetric.

5.2 Model Formulation

5.2.1 Indices and sets

$G(V, E)$ = Graph representing the transportation network with a set of nodes V and a set of edges E

A = set of customer nodes, $A = \{1, \ldots, n\}$, $A \subset V$

P_{kl} = set of paths from node $k \in A$ to node $l \in A$, $k \neq l$, on $G(V, E)$

P_{kl}^r = the r^{th} path from customer node k to customer node l, $P_{kl}^r \in P_{kl}$

5.2.2 Model Parameters

S_i = state of the driver at node i (a vector with three components, i.e. accumulated driving time, accumulated duty time (after the most recent daily break), and accumulated weekly duty time (after the most recent weekly break); initial state of the driver at the origin depot is $S_0 = (0, 0, 0)$.

$t_{ij}(t_i, S_i)$ = travel time on arc (i, j), when vehicle leaves node i at time t_i with driver at state S_i

$t_{P_{kl}^*}(t_k, S_k)$ = shortest time to reach customer node l from customer node k when vehicle leaves node k at time t_k and the driver is at state S_k, where P_{kl}^* is the shortest time path among all $P_{kl}^r \in P_{kl}$

s_l = service time at customer node l (s_l =0, for l = 0 and n+1)

M = arbitrarily large number

$[a_l, b_l]$ = time window within which the vehicle can start service at node l

5.2.3 Decision Variables

$$y_{kl} = \begin{cases} 1, \text{if vehicle visits customer } l \text{ immediately after customer } k \\ 0, \text{otherwise} \end{cases}$$

$$x_{ij} = \begin{cases} 1, \text{if vehicle traverses arc } (i, j) \text{ of the transportation network} \\ 0, \text{otherwise} \end{cases}$$

t_j = departure time of vehicle from node $j \in V$. The starting time of the trip from node 0, t_0, is given.

5.2.4 Mathematical Formulation

Minimize $t_{n+1} - t_0$ (1)

Subject to

$$t_{P_{kl}^*}(t_k, S_k) = \min_{P_{kl}^r \in P_{kl}} \sum_{(i,j) \in P_{kl}^r} t_{ij}(t_i, S_i) x_{ij} \qquad k \in A \cup \{0\},\ l \in A \cup \{n+1\},\ k \neq l \qquad (2)$$

$$t_l - t_k - M y_{kl} \geq t_{p_{kl}^*}(t_k, S_k) + s_l - M \qquad k \in A \cup \{0\},\ l \in A \cup \{n+1\},\ k \neq l \qquad (3)$$

$$a_l \leq t_l - s_l \leq b_l \qquad l \in \mathrm{A} \qquad (4)$$

$$\sum_{j \in V} x_{kj} = 1 \qquad k \in A \cup \{0\} \qquad (5)$$

$$\sum_{j \in V} x_{jl} = 1 \qquad l \in A \cup \{n+1\} \qquad (6)$$

$$\sum_{i \in V, i \neq j} x_{ij} - \sum_{k \in V, k \neq j} x_{jk} = 0 \qquad j \in V \qquad (7)$$

$$t_j \geq 0 \qquad j \in \mathrm{A} \qquad (8)$$

$$x_{ij} \in \{0,1\} \qquad (i, j) \in E \qquad (9)$$

$$y_{kl} \in \{0,1\} \qquad k \in A \cup \{0\},\ l \in A \cup \{n+1\}, k \neq l \qquad (10)$$

The objective function (1) is to minimize total tour time i.e., the sum of travel time, traffic delays, break time, waiting time and service time to complete a tour. Given the departure time from node k and the state of the driver (t_k, S_k), constraint (2) states that if customer l is visited after k, then the shortest time path from node k to node l is selected. The total time to traverse that path is the sum of time dependent travel times on the arcs of the path, including break time, if necessary. Constraint (3) determines the departure time from customer node l if visited after customer node k ($y_{kl} = 1$). It is the sum of arrival time at customer node l plus the service time at node l plus the slack which is potential break time and/or waiting time if the driver arrives before the beginning of the time window. Note that when y_{kl}=0, this constraint is always satisfied. Constraint (4) ensures that the start of service at customer node l should be within the time window. Constraints (5) and (6) ensure that there is only one outgoing arc from customer node k and only one incoming arc into customer node l respectively, or equivalently, a feasible tour should include all customer nodes. Constraint (7) describes the balance equations for all nodes j of the transportation network. Constraint (8) states that departure time from customer node j should be non negative. Constraints (9) and (10) designate x_{ij} and y_{kl} as binary variables.

6 SOLUTION PPOCEDURE AND MODEL TESTING

The problem described above is very complex and cannot be solved by standard optimization algorithms even for very small-sized problems. It resembles the traveling salesman problem in that certain (customer) nodes have to be visited once but the resemblance stops there. The shortest time path between consecutive customer nodes in the route should be used in addition to providing required breaks for the driver and considering possible waiting to start service within stipulated time windows (Constraints 2, 3 and 4). The problem is further complicated by considering realistic travel times on arcs (i, j) that depend on the time of departure from node i and the state of the driver S_i, which is a vector with three continuous components.

To overcome these difficulties, we developed two-phase solution procedures comprising an initialization and a simulated annealing improvement meta-heuristic. Since the travel time along each arc is not fixed, a time dependent travel function with HOS regulations is incorporated into Dijkstra's algorithm to find the travel time along each arc (TDD-HOS). To elaborate, the

initialization phase begins with a greedy heuristic which was designed to develop a Hamiltonian tour for visiting all customers and then returning to the depot. After the arrival time at each customer node l is found, the departure time t_l is computed by adding service time, waiting time if arrival occurs outside the service time window of the customer, and break time if necessary, according to constraint (3). The customer with the earliest departure time t_l, among all yet to be visited customers, is selected to be visited next. The above process is repeated until an initial tour is found. In the improvement phase, we use a simulated annealing schedule to improve the solution iteratively until a certain halting criterion is met. TDD-HOS is used again to complete tours in the improvement phase which terminates with a near optimal solution.

6.1 Greedy Heuristic and Time Dependent Dijkstra's Algorithm

The greedy heuristic finds an initial tour starting at the origin depot and ending at the destination depot. The starting time of the trip is set to time t_0. In each iteration, the greedy heuristic finds the next customer to be visited. Consider a partial tour from the origin depot to a customer k. Let the set of nodes in the partial tour be T. The next customer is selected among all unvisited yet customers $l \in (A\text{-}T)$. The set of customers/depot nodes are connected through the highway arcs (set E) of the transportation network. Actually there are many paths linking any two customer nodes k, $l \in A$ (set P_{kl}). In finding the shortest time path P_{kl}^{*}, TDD-HOS is employed. TDD-HOS is the ordinary Dijkstrar's algorithm which instead of using a constant travel time for each arc (i, j) it calculates it by a *Time Dependent Travel Function with HOS Regulations*. This is because the vehicle speed may be changing during the arc traversal or the driver may need a break. For Dijkstrar's algorithm adaptation to account for time dependent arc lengths, see Dreyfus (1969). The following notation is used:

dut: accumulated duty time (driving time, service time and waiting time) since last daily break

daily_max: maximum daily driving time between consecutive daily break periods (11 hours)

duty_max: maximum duty time between two consecutive daily break periods (14 hours)

daily_break: time required for a daily break period (10 hours)

res_ddr: residual driving time before the next daily break period (i.e. *daily_max – drt*)

res_dut: residual duty time before the next daily break period (i.e. *duty_max – dut*)

drt: accumulated driving time since last daily break

Given the departure time and the state of the driver (t_i,S_i) at node *i*, the Time Dependent Travel Function with HOS Regulations for arc (*i*, *j*) calculates the arrival time and the state of the driver (t_j,S_j) at node *j*. Assuming that the length of the trip does not exceed a week, we can simplify the notation by ignoring the third component of the state of the driver, weekly duty time. The function can be easily extended to the case where a trip may take longer than a week. The flowchart of the time dependent travel function with HOS regulations on arc (*i*, *j*) is shown in Figure 2.

As in the ordinary Dijkstra's algorithm, TDD-HOS identifies the next earliest to reach network node from customer *k* and labels it. After some iterations, consider such a labeled node *i* and an incident arc (*i*, *j*), where node *j* is not labeled yet. In Figure 2, the vehicle leaves node *i* at time t_i with state of the driver S_i = (*drt*, *dut*). The travel speed of the vehicle depends on the time of the day *t* (i.e. $v(t) = c_{ij}^{h} \mid h \leq t \leq h+1$). The residual daily driving *res_ddr* and daily duty *res_dut* times are computed next. Then, the earliest time δt is found until one of the following events occurs: (a) speed on arc (*i*, *j*) changes, (b) node *j* is reached, (c) maximum daily driving time is reached, or (d) maximum duty time is reached. Whichever event occurs first, the current time and the state of the driver are updated. If the residual driving time or duty time is not sufficient to reach node *j*, the driver takes a daily break. Time of departure after taking the break is updated and daily driving (*drt*) and duty time (*dut*) are reset to zero. The speed to traverse the remaining distance depends on the time of the day the driver departs after taking the break. The above mentioned steps are repeated until node *j* is reached. The earliest time to reach node *j* and the state of the driver (t_j,S_j) are recorded.

After the travel function is processed for arc (*i*, *j*), control passes from the travel function to the main body of the TDD-HOS algorithm which considers other arcs emanating from node *i* (if any are left) and the travel function is processed again. The earliest adjacent node to reach from *i* is identified and the Dijkstra's algorithm repeats the above steps for all labeled nodes. The earliest to reach node from any labeled node is identified and labeled. If that node is a customer node, say *l*, backtracking yields the shortest time path P_{kl}^{*}. TDD-HOS continues until every yet to be visited customer $l \in (A\text{-}T)$ is reached from customer node *k* and its truck arrival time and the state of the driver is determined.

To find the customer node that should be visited next in the initial tour, service times and possible waiting and break times have to be added. Several cases arise depending on the arrival time

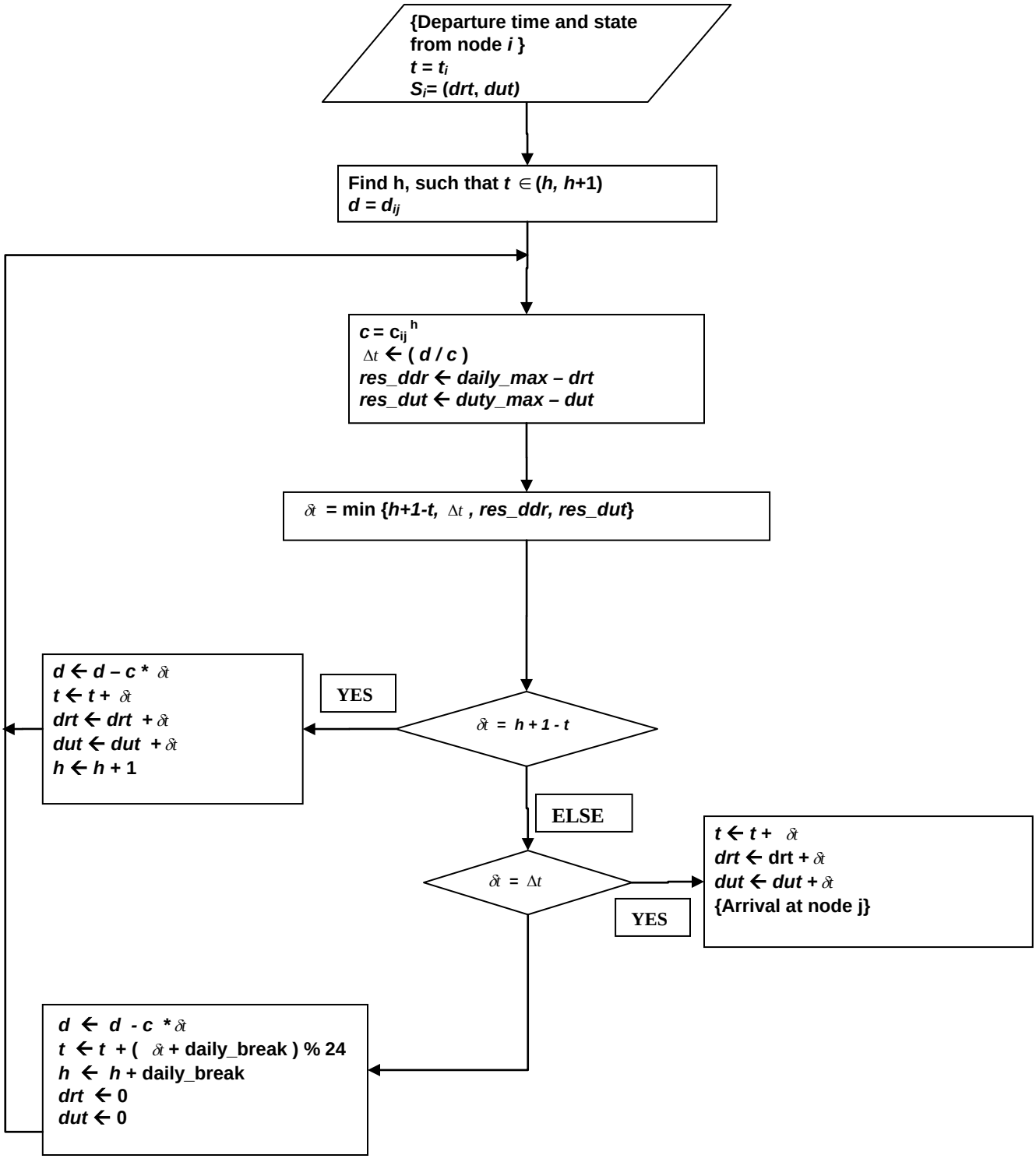

Figure 2. Time dependent travel function with HOS regulations

at node l and the state of the driver. A "service function" similar to the time dependent travel function of Figure 2 was developed (see Shah, 2008) that finds the departure time t_l from node l and the state of the driver, as summarized below. If the truck arrives during the service window $[a_l, b_l]$ and with the added service time the driver does not exceed the maximum duty time ($dut + s_l \leq duty\_max$), service can start immediately at customer node l and the departure time t_l is computed by adding service time. If the truck arrives during the service window and $dut + s_l > duty\_max$, the driver needs to take a break and start service at time a_l the next day; t_l is computed by adding break time, waiting time and service time. In case the driver arrives outside the service window, he/she can wait and start service at the beginning of the service window a_l if after adding waiting time and service time to dut, the accumulated duty time does not exceed $duty\_max$; otherwise, the driver takes a break and departure time t_l and driver's state S_l are adjusted accordingly. The service function is executed for all $l \in (A\text{-}T)$. The customer node with the earliest departure time t_l, $l \in (A\text{-}T)$, is visited next in the greedy heuristic which continues until the destination depot is reached and the initial tour is completed.

6.2 Simulated Annealing Heuristic

Annealing is a process in which a solid in a heat bath is heated by increasing its temperature to a value at which all particles of the solid randomly arrange themselves in the liquid phase, followed by a series of slowly decreasing temperatures of the heat bath. In thermodynamics terms, slow cooling allows the solid to reach thermal equilibrium, i.e. a stable state of the solid. Metropolis et al. (1953) developed an algorithm, known as simulated annealing (SA), for efficiently simulating a collection of atoms in equilibrium at a given temperature T. Skillfully inventing an analogy to Metropolis's SA algorithm, thirty years later, Kirkpatrick et al. (1983) developed a SA algorithm for finding a good solution in complex minimization problems that exhibit many local minima. For an extensive discussion of the theory of SA and its applications, see Kirkpatrick et al. (1983) and Laarhoven et al. (1987). The relevant elements that have to be selected in a new implementation of the SA algorithm are: (1) how random problem configurations are generated and evaluated (transition mechanism), (2) initial value of the control parameter (initial temperature), (3) how temperature is lowered in stages (cooling schedule), (4) what is the condition for reaching a steady state at each temperature (equilibrium condition), and (5)

when the SA terminates (stopping rule). The selection of these elements for our vehicle routing and scheduling problem under HOS regulations is described below.

6.2.1 Transition mechanism

SA heuristic starts with the solution (tour) generated by the greedy heuristic. Originally suggested by Lin (1965), sub-tour reversal is used to generate new tours. A tour section is removed and then replaced by the same customer nodes in the opposite order. A sub-tour reversal requires therefore selection of both the beginning and ending slots from a given sequence of customer nodes. Random numbers are generated to select the beginning and ending slot. Let the set of nodes from the origin depot to the beginning slot in the current tour be T. In the new tour, the departure times from all nodes $l \in T$ need not to be changed. The departure times from the remaining nodes $l \in (A\text{-}T)$ however need to be recalculated using the TDD-HOS algorithm. The generated new tour is evaluated and total time to complete it (E_1) is compared with the total time (E_2) of the current tour. If $E_1 \leq E_2$, the new solution is always accepted and becomes the current solution. If $E_1 > E_2$, the new solution is accepted with probability $p = \exp\left((E_2 - E_1)/T\right)$, thus the probability of accepting a non-improving move decreases with decreasing temperature (T) and increasing objective value deterioration (E_1 - E_2). By accepting occasionally non-improving solutions, SA avoids getting stuck at local minima.

6.2.2 Initial temperature

Following the suggestion made by Connolly (1990), we generated initially M = 50k random tours, where k = 0.5 (n) (n-1), to find $\delta_{\max}$ (largest uphill step) and $\delta_{\min}$ (smallest uphill step). The initial temperature T_0 was set to $\delta_{\max}$, thereby giving an initial probability p greater than 0.4.

6.2.3 Cooling schedule

The cooling schedule controls the rate at which the temperature changes and helps the heuristic to avoid local minima. It causes the heuristic to act more erratically when the temperature is high and consequently, at higher temperatures, the probability of accepting the worse solution is much higher than that at a lower temperature. The cooling schedule can be linear, exponential or polynomial. It can be determined by trial and error based on the trade-off between solution quality and computation time. If we decrease the temperature too quickly, the system will get "quenched." That is to say, the system is still in the higher energy state (higher objective function value) and the temperature is too low to find a tour with lower energy state (lower objective function value). Thus, we may end up

with local minima and the algorithm may not be able to find a good solution. After equilibrium is reached (see below) at each stage k, associated with temperature t_k, we reduce the temperature using the adjustable cooling rate γ. Thus, $t_{k+1} = \gamma * t_k$. The range of γ is [0.5, 0.9]. Our SA heuristic was run with different cooling schedules, $\gamma = 0.7, 0.8, 0.85, 0.9$, and the best resulting solution was selected.

6.2.4 Equilibrium condition

A number of iterations are made at each temperature to reach an equilibrium condition. In our experiments, we used Burkard and Rendl's formula (1984) that requires $L = 0.5 * n^2$ iterations at each stage to reach an equilibrium. As suggested, we also multiplied L by 1.1 to improve solution quality. Through a series of experiments, we found that a multiplicative factor of 1.03 produced results as good as those obtained by using a multiplicative factor of 1.1. The decrease in the multiplicative factor tended to reduce computation time considerably.

6.2.5 Stopping rule

The SA heuristic stops when the halting criterion (minimum temperature) is reached. The minimum temperature is taken as $\delta_{\min}$, the smallest uphill step found in the random tours initially generated. The heuristic can be stopped earlier after a certain number of iterations (approximately after 3200 iterations), if the gap between the objective value at the current solution and the best solution obtained in the last 5 consecutive iterations is less than a minimum reduction rate (ε=0.01). In our experiments, either a minimum reduction rate or a halting criterion, whichever occured first, was used as the stopping threshold.

6.3 Test Results

To evaluate the accuracy of the heuristic, we compared its solutions to the optimum ones obtained by exhaustive enumeration for relatively small sized problems. In each experiment, customers were generated at cities in the Northeast USA. For a given number of customers, n, all possible asymmetric $n!$ customer sequences were generated. In each sequence of customers from the origin to the destination depot, the path between consecutive customers and the driver's schedule were computed using the TDD-HOS algorithm. Both exhaustive enumeration and the initialization/SA metaheuristic described earlier were coded in Java using Eclipse 3.4 and run on an Intel processor 2.0 GHz. The lowest time of the $n!$ tours from the exhaustive emumeration was compared to the one

obtained by the metaheuristic.

We increased the number of customers gradually from n = 4 until the computation time required for exhaustive enumeration was too long to continue (one day). For small size problems, up to 6 customers, the two approaches found identical (optimal) solutions and took a reasonable amount of time, e.g. for n = 6 the exhaustive enumeration took an average of 150 secs while the metaheuristic took 17 secs. However, the time it takes to find the optimum by exhaustive enumeration increases exponentially with n and the limit (1 day) is reached at n = 10. At that problem size we conducted 16 experiments using different problem instances and cooling schedules γ (described earlier). The results of the metaheuristic compared very favorably to the optimal. In 70 % of the cases the solution was optimal while in the remaining 30 % of the non-optimal cases the objective function deviated by less than 1% from the optimal objective value. The time it took the metaheuristic to find the optimal or near optimal solution in the experiments was under 7 minutes. In general, we found that the accuracy of SA mainly depended on the initial solution, initial temperature, number of iterations performed at each stage and the cooling schedule. Slower cooling requires more computational time but gives better quality solutions. Thus, by varying the cooling schedule, we can make trade-offs between solution accuracy and computation time. By trial and error, we determined the optimal cooling schedule for a problem with n = 10 customers. According to the formula given earlier, the initial temperature was obtained in 4500 replications. The initial temperature was set to the largest uphill step (δ_{max} = 180) and the halting criterion was set to the lowest uphill step (δ_{min}=3). At each stage (temperature), 1.03 L iterations were performed, where L = 0.5 n^2 is the Burkard and Rendl's (1984) formula and the multiplier for improving quality solution (1.03) was found experimentally.

To evaluate the average computational efficiency of the metaheuristic in solving larger size problems, n was gradually increased to 50 customers in a step of 5 with 10 randomly generated problems at each value. All the experiments were performed with γ = 0.9 cooling schedule and with L as described above. A nonlinear fitting of the average computation time t (secs) that it took to solve the problems as a function of n yielded $t = 2.506 * n^{2.257}$. Thus, the average complexity of the heuristic is polynomial (a little higher than quadratic) in the number of customers.

6.3.1 An Example

To illustrate the usefulness of the model-based DSS and further test the solution procedure we applied it to real-world problems encountered by a regional truckload (TL) carrier which primarily serves customers in the states east of Mississippi. To keep the confidentiality of this carrier, it will be called "Delta." Delta has a fleet of refrigerated, temperature-controlled trucks which haul consumer retail products including foodstuff commodities (e.g., frozen ice cream, fresh meats, various produce). To illustrate Delta's current operations, the test problems were generated based on key customer bases in the Northeastern United States. These customers are served by the central warehouse (depot) located in Wilbraham, MA. From this depot, the truck departs at 7:00 am on Monday and returns back after serving customers located in 13 different towns. The domicile of the driver is at Enfield, MA, which is also visited during the tour and the driver stays for a period of at least four hours starting at some time during the time window 2pm – 10 pm. The time window for starting service at the customers is 9 am – 5 pm and service time is fixed to two hours.

A map of the New England States that displays the 15 cities or towns with the underlying transportation network is shown in Figure 3. The 15 cities or towns where the customers are located are: Assonet MA, Brattleboro VT, Cheshire CT, Ellington CT, Hartford CT, Long Meadow MA, Methuen MA, New Britain CT, New Haven CT, Revere MA, Sturbridge MA, Westfield MA, and Worcester MA. Only highway segments that may be traversed during the tour have been considered. The resulting network consists of 52 nodes and 138 arcs. Of the 52 nodes, 15 represent cities or towns and the remaining nodes represent highway intersections. The arcs represent highway segments between nodes. The arcs are directed, i.e. each highway segment is represented by two arcs.

Highway time dependent speed data for each arc and each hour of the day starting at 12 o' clock midnight were estimated from Google Maps (see Melachrinoudis, 2009). The Google's *traffic* feature (*http://maps.google.com/*) displays for each hour of the day the highway speed with different colors, corresponding to different speed levels from slow to fast. Three different speed levels were estimated at 20, 40 and 60 miles per hour. When parts of a highway segment (arc) were colored differently in a given hour of the day, the different speeds were weighted by the respective lengths of the highway sub segments for an overall effective speed of the arc.

The network topology, arc distances, arc time dependent speeds and the set of customers with the associated service windows and service times were used as input into the *Time Dependent Truck*

Figure 3. Underlying transportation network connecting customers and depot

Routing and Driver Scheduling Model and the *Simulated Annealing* (SA) *Metaheuristic*. The greedy heuristic yielded a tour of 95.1 hours, which was used as an initial solution to the SA metaheuristic. SA was performed using four different schedules with γ =0.7, 0.8, 0.85 and 0.9. The results were four tours with total trip time 82.59, 82.46, 81.40 and 82.22 hours, respectively. It should be noted that the solutions obtained by the four cooling schedules were not very different (within a 1.19 hour interval) which is reassuring of the quality of the final solution. Also the initial solution obtained by the greedy heuristic was fairly good, with tour time only 16.8 % higher than the best trip time obtained by SA. Each SA run took on the average 12-13 minutes for γ = 0.85.

The tour associated with the best objective value of 81.40 hours was selected as the near-optimal solution. The tour includes warehouse and customers in the following order: Willbraham –

Sturbridge – Worcester – Brattleboro – Enfield – Cheshire – New Haven – New Britain – Hartford – Revere – Methuen – Assonet – Westfield – Long Meadow – Ellington – Wilbraham. In the map of Figure 3, the highways used in the tour have been highlighted and the truck route has been drawn with thinner lines parallel to them with arrows indicating the route direction. Different symbols have been used to indicate the warehouse, the customers, the driver's home, the places the driver takes daily break and those at which he/she is waiting before service starts. The detailed truck route and driver's schedule is shown in Appendix A. Note that for notational simplicity and computational efficiency, fractional times are used with rollover from one day to the next (cumulative time). The truck leaves the depot at Wilbraham at time 7.00 and returns back at time 88.40. This translates to departure time from Wilbraham at 7:00 am on Monday and arrival back at 4:24 pm on Thursday for a total tour time of 81 hours and 24 minutes. The driver takes off-duty extended break (more than 10 hours long combining it with waiting until service starts) on each one of the three nights, on Monday night at his home in Enfield, CT, on Tuesday night at Revere, MA, and on Wednesday night at Westfield, MA. The truck waits before service starts in Sturbridge on Monday morning. The driver's schedule looks very reasonable, having all off-duty breaks overnight including one at home.

To illustrate the importance of "what-if" analysis in the model-based DSS, we assessed the impact of time dependent arcs on total trip time (TTT) and time of arrival at the destination depot (TAD) as departure time from the origin depot t_0 changes. We varied the departure time by half hour intervals before and after the base case scenario (7 am departure time on Monday). The results obtained are depicted in Figure 4 where the left scale and the curve with the square bullets represent TTT and the right scale and the curve with the triangular bullets represent TAD. The "0" in the TAD scale denotes the beginning of Thursday.

In addition to presenting the what-if analysis results, we provide some explanation that renders valuable insights. Departing earlier than 7 am improves neither TTT nor TAD. Earliest departures (5 and 5:30 am) result in higher TTT and TAD due to extra waiting at customers before service starts (9 am). Departures between 6 and 7:30 am have identical TADs while the TTTs are decreasing by the time of late start. The time saved in TTT as departure time increases is due to less waiting at the customers. Therefore, departure at 7:30 is better than departure at 7 am. A departure change from 7:30 to 8 am and 8:30 to 9 am results in both higher TTT and TAD. To explain this, we

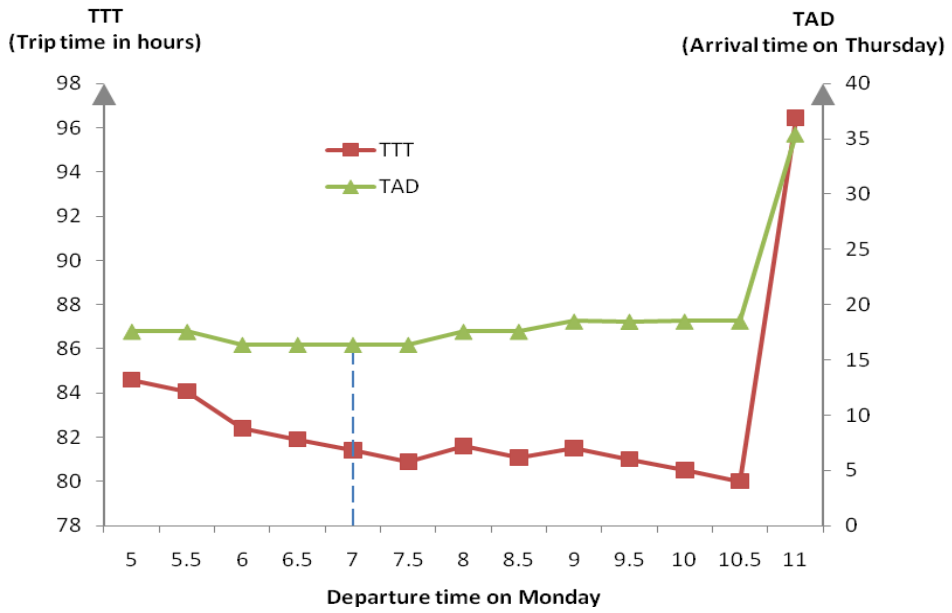

Figure 4. What-if analysis as departure time changes

investigated the reason for the route change and found that due to traffic and/or later departure a service window is missed. For example, using the vehicle speed data on the route obtained for the 8:30 am departure, we calculated the driver's schedule if the truck had departed at 9 am and found that the truck arrives at the customer in Revere at 5:10 pm, missing the time window by 10 minutes and thus having to wait until the beginning of the service window the next day. The heuristic of course finds a better route for a 9 am departure which, however, is about half an hour higher in TTP and close to an hour in TAD compared to the 8:30 am departure. As with the time interval 6-7:30 am, departure time at 9 – 10:30 am results in constant TAD but in decreasing TTT by the amount of late start, again due to decreasing waiting time at the customers. A departure at 11 am results in an additional overnight break with arrival at the destination depot at 11:45 am on Friday. Compared to 7 am, the 10:30 am departure provides a route that is 1 hour and 23 minutes shorter in TTT but 2 hours and 7 minutes longer in TAD (a tradeoff between TTT and TAD). Thus, as input parameters or operating scenarios change, what-if analysis provides routes and schedules that best align the timing of arrivals at customer locations and drivers' breaks with service time windows, by directly considering time dependent speeds in route selection.

7 CONCLUDING REMARKS

Through a series of computational experiments, we found that the computational efficiency and accuracy of the proposed SA heuristic depended on the quality of its initial solution, equilibrium conditions, and the cooling schedule used to solve the problem. The solution quality of the heuristic was verified by comparing its results to those of the exhaustive enumeration algorithm for problems up to 10 customers. The vast majority of the solutions obtained by SA were within 1 % of the optimal solutions while 70% were optimal. For 10 customers SA took under 7 minutes while exhaustive enumeration took more than a day. Further experiments for larger size problems indicated that the average computational efficiency of the SA heuristic is a little higher than quadratic in the number of customers. A hypothetical but realistic example of the truck routing and driver scheduling problem with HOS regulations encountered by a typical long-haul carrier in the U.S. was presented.

Despite the aforementioned proven efficiency and practicality, the proposed model and solution procedure point to a number of directions for future work:

- The model-based DSS can be expanded to include the element of uncertainty (stochasticity) involved in the time windows as well as travel times between nodes.
- The model and solution procedure can be modified to consider the multiple objective aspect (e.g., trade-off between driver preferences and customer delivery requests) of the vehicle routing and scheduling problem similar to the one solved by Min (1991).
- Future research may accommodate delivery schedule changes dynamically through real time communication between the dispatcher and the drivers, i.e., for adding new delivery requests or canceling existing delivery requests.
- The comparison of the simulated annealing meta-heuristic to other proven heuristics such as ant-colony optimization, genetic algorithm, Tabu search, set partitioning, or *k*-opt exchange heuristics are worth investigating in the future.

Acknowledgments: The authors would like to thank the Intermodal Transport Institute at the University of Toledo and the U.S. Department of Transportation for funding this study through a research grant. Also, the authors are grateful to Mr. Ed Nagle, CEO of Nagle Trucking Inc. and Mr. Charles Harrett, CEO of Northern Continental Logistics for providing the authors with practical insights into hours-of-service regulations and their trucking operations in the United States.

REFERENCES

1. Archetti C. and Savelsbergh M. (2007). The trip scheduling problem. Working paper, Atlanta, GA: Georgia Institute of Technology, www.isye.gatech.edu/~mwps/publications/.
2. Basnet, C., Foulds, L., and Igbaria, M. (1996). FleetManager: a microcomputer-based decision support system for vehicle routing. *Decision Support Systems*, 16, 195-207.
3. Benyahia, I. and Potvin, J-Y (1998). Decision support for vehicle dispatching using genetic algorithm. *IEEE Transactions on Systems, Man, and Cybernatics-Part A*, 28(3), 306-314.
4. Bodin, L, Golden, B, Assad, A, and Ball, M. (1983). Routing and scheduling of vehicles and crews: The state of the art. *Computers and Operations Research*, 10, 62-212.
5. Burkard R.E., Rendl F. (1984). A thermodynamically motivated simulation procedure for combinatorial optimization problems. *European Journal of Operational research*, 17, 169-174.
6. Carpey, S. (2010). Truck accidents and driver fatigue - Accidents waiting to happen, accidents that can be avoided. http://ezinearticles.com/?Truck-Accidents-and-Driver-Fatigue---Accidents-Waiting-to-Happen,-Accidents-That-Can-Be-Avoided&id=2121918.
7. Connolly D.T. (1990). An improved annealing scheme for QAP. *European Journal of Operational Research*, 46, 93-100.
8. Donati, A., Montemanni, R., Casagrande, N., Rizzoli, A., and Gambardella, L. (2006). Time dependent vehicle routing problem with a multi ant colony system. *European Journal of Operations Research*, 185, 1174- 1191.
9. Dyer, J.S. and Mulvey, J.M. (1983). Integrating optimization methods with information systems for decision support. In *Building Decision Support Systems*, J.L. Bennett Ed., Reading, MA: Addison-Wesley, pp. 89-109.
10. Dreyfus, S.E. (1969). An appraisal of some shortest-path algorithms. *Operations Research*, 17, 395-412.
11. Federal Motor Carrier Safety Administration, (2005). *Hours-of-Service Regulations.* http://www.fmcsa.dot.gov/rules-regulations/topics/hos/hos-2005.htm.
12. Fleischmann, B., Gietz, M., and Gnutzmann, S. (2004). Time-varying travel times in vehicle routing. *Transportation Science*, 38, 160-173.
13. Gayialis, S.P. and Tatsiopoulos, I. P. (2004). Design of an IT-driven decision support system for vehicle routing and scheduling. *European Journal of Operational Research*, 152, 382-398.
14. Goel, A. and Gruhn, V., (2006). Driver's working hours in vehicle routing and scheduling. Proceedings of the 9th IEEE International Conference on Intelligent Transportation Systems, Toronto, Canada, September 17-20, pp. 1280-1285.
15. Ichoua, S., Gendreau, M. and Potvin, J-Y (2003). Vehicle dispatching with time-dependent travel times. *European Journal of Operational Research*, 144, 379-396.
16. Keen, P.G.W., and Scott Morton, M.S. (1978). *Decision Support Systems: An Organizational Perspective.* Reading, MA: Addison-Wesley.
17. Kirkpatrick, S., Gelatt Jr. C.D., and Vecchi, M.P. (1983). Optimization by simulated annealing. *Journal of Operations Research*, 220, 671-680.

18. Laarhoven, V.P.J.M and Aarts, E.H.L. (1987). *Simulated Annealing: Theory and Applications.* Norwell, Massachusetts: Kluwer Academic Publishers.
19. Lin, S. (1965). Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, 44, 2245-2269.
20. Malandraki, C. and Daskin, M., (1992). Time dependent vehicle routing problems: Formulations, properties, and heuristic algorithms. *Transportation Science*, 26(33), 185-200.
21. Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller A. and Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21, 1087-1092.
22. Min, H. (1989). A model-based decision support system for locating banks. *Information and Management*, 17(4), 207-215.
23. Min, H. (1991). A multiobjective vehicle routing problem with soft time windows: the case of public library distribution system. *Socio-economic Planning Sciences*, 25(3), 179-188.
24. Min, H. (1998). A personal computer assisted decision support system for private versus common carrier selection. *Transportation Research E*: *Logistics and Transportation Review*, 34(3), 229-241.
25. Min, H. (2009). The impact of hours-of-service regulations on transportation productivity and safety: A summary of findings from the literature. *Journal of Transportation Management*, 21(2), 49-64.
26. Min, H. and Zhou, G. (2002). Supply chain modeling: past, present and future. *Computers and Industrial Engineering*, 43(1), 231-249.
27. Melachrinoudis, E. (2009). Final Report on UTCTC-SC-1 Grant. Department of Mechanical and Industrial Engineering, Northeastern University, August 2009.
28. Napolitano, M. (1998). *Using Modeling to Solve Warehouse Problems: A Collection of Decision-Making Tools for Warehouse Planning and Design.* Oak Brook, IL: Warehousing Education and Research Council.
29. Ramakrishnan, R. and Gehrke, J. (2003). *Database Management Systems.* 3rd edition, New York, NY: McGraw-Hill.
30. Reissman, C.J. (1997). *The Alert Driver: A Trucker's Guide to Sleep, Fatigue, and Rest in our 24-Hour Society.* Alexandria, VA: American Trucking Associations, Inc.
31. Ruiz, R., Maroto, C., and Alcaraz, J. (2004). A decision support system for a real vehicle routing problem. *European Journal of Operational Research*, 153, 593-606.
32. Shah, V.D. (2008). Time dependent truck routing and driver scheduling problem with hours of service regulations. MS Thesis, Northeastern University, Boston, Massachusetts, USA.
33. Solomon, M and Desrosiers, J. (1988). Time window constrained routing and scheduling problems. *Transportation Science*, 22, 1-13.
34. Sprague, R.H. and Carlson, E.D. (1982). *Building Effective Decision Support Systems.* Englewood Cliff, New Jersey: Prentice-Hall.
35. Turban, E. and Aronson, J.E. (2001). *Decision Support Systems and Intelligent Systems*, Upper Saddle River, New Jersey: Prentice-Hall.
36. Xu, H., Chen, Z.L., Rajagopal, S., and Arunapuram, S. (2003). Solving a practical pickup and delivery problem. *Transportation Science*, 37(3), 347-364.

APPENDIX A

1. At 7.00 (7 am Monday) travel from **Wilbraham depot** to I90_R32 to Sturbridge for 0.74 hours.
   Arrival at **Sturbridge** customer: 7.74. **Wait** until 9:00 to start service; service lasts 2 hours
2. At 11.00 travel from Sturbridge to I90_I395 to Worcester for 0.58 hours
   Arrival at **Worcester** customer: 11.58. Service lasts 2 hours
3. At 13.58 travel from Worcester to I190_R2 to I91_R2 to Brattleboro for 2.74 hours
   Arrival at **Brattleboro** customer: 16.32. Service lasts 2 hours
4. At 18.32 travel from Brattleboro to I91_R2 to I90_I91 to I91_R5 to Enfield for 2.18 hours
   Arrival at **Enfield (home of the driver)**: 20.50 (8:30 pm)
   Stay overnight at home taking an extended **off-duty break** until 31.70 (7:42 am Tuesday)
5. At 31.70 travel from Enfield to Hartford to R6_R9 to I84_I72 to R10_I691 to Cheshire for 1.3 hours
   Arrival at **Cheshire** customer: 33.00. Service lasts 2 hours
6. At 35.00 travel from Cheshire to NHV to New Haven for 0.48 hours
   Arrival at **New Haven** customer: 35.48. Service lasts 2 hours
7. At 37.48 travel from New Haven to NHV to I91_I691 to I91_R9 to New Britain for 1.07 hours
   Arrival at **New Britain** customer: 38.55. Service lasts 2 hours
8. At 40.55 travel from New Britain to R6_R9 to Hartford for 0.38 hours
   Arrival at **Hartford** customer: 40.93. Service lasts 2 hours
9. At 42.93 travel form Hartford to I84_I83 to Sturbridge to I90_I395 to I90_I146 to I90_I495 to I90_I95 to BOS to Revere for 2.99 hours. Arrival at **Revere** customer: 45.92 (9:55 pm Tuesday)
   Stay overnight taking an extended **off-duty break** until 9:00 am to start service
10. At 59.00 travel from Revere to I93_I95 to I495_I93_via_I93 to Methuen for 0.87 hours
    Arrival at **Methuen** customer: 59.87. Service lasts 2 hours
11. At 61.87 travel from Methuen to I495_I93_via_I93 to I93_I95 to BOS to I93_R24 to I495_R24 to Assonet for 1.91 hours. Arrival at **Assonet** customer: 63.78. Service lasts 2 hours
12. At 65.78 travel from Assonet to PRVD to I90_I146 to I90_I395 to Sturbridge to I90_R32 to I90_R21 to I90_I91 to Westfield for 3.33 hours. Arrival at **Westfield** customer: 69.11 (9.11 pm)
    Stay overnight taking an extended **off-duty break** until 9:00 am. Service lasts 2 hours
13. At 83.00 travel from Westfield to I90_I91 to I91_R5 to Long_Meadow for 0.49 hours
    Arrival at **Long_Meadow** customer: 83.49. Service lasts 2 hours
14. At 85.49 travel from Long_Meadow to I83_H_road to Ellington for 0.38 hours
    Arrival at **Ellington** customer: 85.87. Service lasts 2 hours
15. At 87.87 travel from Ellington to I83_H_road to Wilbraham
    Arrival at **Wilbraham depot**: 88.40 (4:24 pm Thursday)