Local Truckload Vehicle Routing and Scheduling with Strict Time Window Constraints

UCI-ITS-LI-WP-99-5

Xiubin Wang 1
Amelia C. Regan 2

1 Institute of Transportation Studies, University of California, Irvine
tgolob@uci.edu

2 Department of Civil and Environmental Engineering
and Institute of Transportation Studies, University of California, Irvine
aregan@uci.edu

June 1999
Institute of Transportation Studies
University of California, Irvine
Irvine, CA 92697-3600, U.S.A.
http://www.its.uci.edu
Local Truckload Vehicle Routing and Scheduling with Strict Time Window Constraints

Xiubin Wang and Amelia C. Regan
Department of Civil and Environmental Engineering and Institute for Transportation Studies
University of California, Irvine 92697

Abstract

This paper describes a solution method for time constrained vehicle routing and scheduling tailored to local truckload trucking operations such as those supporting rail or maritime intermodal operations. The problem requires a solution method that works quickly, allowing dispatchers time to examine solutions, and in some cases to make changes and re-solve. In addition, the problem must be solved several times during the course of a day as new demands become known and either traffic or transfer center delays force the reassignment of loads from one driver to another. We present a mixed integer programming problem in which explicit time constraints are replaced by binary flow variables. Two versions of the problem, one over-constrained and another likely under-constrained are solved at each step. The solutions to these problems provide upper and lower bounds on the value of the optimal solution. This method appears to be suitable for real-time or quasi real-time implementation because an initial feasible solution can be generated in well under one minute for problems of reasonable size. Improvements are made to this initial solution if more computational time is allowed. Test results are presented to illustrate the trade-offs between solution quality and computation time.

Key words: Intermodal Operations, Truckload Trucking, Information Technology, Commercial Vehicle Operations, Real-Time Fleet Management
INTRODUCTION

In the twenty years since deregulation the trucking industry in the United States has become increasingly competitive. The need to raise service levels and at the same time reduce operational costs, coupled with opportunities to integrate information technologies into day-to-day operations has led to increased interest among both the academic research community and practitioners in the development of computer and communication based tools to increase operational efficiency. A significant body of work has been developed in this area, incorporating present and future (forecast) demands, deterministic and stochastic factors and involving both optimization-based and heuristic algorithms. Powell (1996) presents various formulations and solution methods for the truckload carrier dynamic assignment problem. Most of that work has involved a network with an explicit number of nodes (geographic regions or zones) and a set of discrete time stages. This research differs in that we restrict ourselves to local truckload moves such as those in and around intermodal facilities and to a set of work that must be completed during a single 24-hour period. Demands arise within a compact geographic region near one or more intermodal facilities (rail or maritime facilities or both). These problems are simpler in some respects than the traditional dynamic vehicle allocation problems in that we do not consider repositioning moves made in anticipation of future demands. Vehicles are busy, waiting at an intermodal facility or waiting at the depot for an assignment. In certain respects however, these problems are more complicated than the traditional vehicle allocation problems. Loads have strict time window constraints, and dock times (loading, unloading and waiting times) may be unpredictable as can service times at intermodal facilities. These stochastic elements, combined with travel times subject to recurring and non-Recurring congestion have a significant affect on the ability of dispatchers to assign a driver a full day's work, even if one hundred percent of the days' demands are known at the start of the day.

This problem lends itself to formulation as a vehicle routing problem with time windows
but differs from typical VRPTW problems in that the vehicles are assigned only a small number of moves during the planning horizon (a single work day).

These problems fall into a class of those in which it is difficult to accurately describe the fluctuation of demands and service times and in which it is not cost effective to make the effort to characterize and explicitly include stochastic elements in the solution. In real time applications, trade-offs between computational complexity and solution quality exist. The complexity of accurately modeling uncertainty and the complexity of algorithms which explicitly consider stochastic elements justifies the use of a deterministic vehicle routing model as an important part of the strategies used to make online (real-time) assignments. Regan, Mahmassani and Jaillet (1996, 1998) presents a set of heuristics for real-time assignment and routing for dynamic carrier fleet operations. That work assumes that no demands are know a priori and that loads must be assigned (in order to ensure time-window feasibility) immediately after they are requested. Furthermore, the assignment rules rely on purely local optimization techniques, which miss out on system-wide assignment opportunities (Regan, Jagannathan and Wang, 1999). The approach described in this research assumes that a significant fraction of demands are known at the beginning of the assignment period and seeks to take full advantage of this information while at the same time retaining flexibility to react to changes if need be. Yang, Jaillet and Mahmassani (1999) extended earlier analysis considerably, and developed a global optimization-based formulation of the real-time truckload pickup and delivery problem which they call myopic because it involves only information known at the time of solution in a highly dynamic environment. The problem they solve corresponds to our problem. However they examine systems which are even more dynamic. Because this method is intended for eventual implementation in operations, we assume, as is typically the case, that a large fraction of demands are know at the start of day. In addition, the research described in Yang et al (1999) has as its focus much smaller problems as it was intended to develop new insight into dynamic problems rather than lead to an operational system.

The real time local truckload pick up and delivery problem has attracted relatively little
attention from the research community. Until recently, few carriers had intermodal operations of the size inviting the development of automated routing and scheduling systems and the large local pickup and delivery problems faced by private fleets typically involved primarily fixed routes. The motivation for this research is the intermodal operations of one of the largest truckload carriers in the US. The company uses optimization software to assist with the development of schedules for their long haul (over-the-road) drivers but not for their local operations. Local operations are driven by many somewhat intangible factors including customer service and safety constraints that favor sub-fleets of relatively few drivers working in the same areas and with the same customer set from day to day. These operations have been historically fairly well managed by dispatchers. However a sharp increase in recent years in the use of rail intermodal transportation has led local operations to become much more complex and increasingly large, inviting the development of computer aided dispatching systems. In addition to including more than a hundred drivers and hundreds of loads everyday, these problems increasingly include more than one rail terminal and a fairly wide geographic region. In addition, problems similar to the ones described here could just as easily involve several ports or a combination of maritime ports and rail terminals. Intermodal air transportation is also of significant interest but is not addressed here as it rarely involves truckload ground movements.

In this paper, we examine an approximation method to solve the deterministic vehicle routing problem with time window constraints. This is a relatively simple method that can be implemented easily on a standard desktop computer. The idea is that this method will be implemented in a dispatching system in the following way: At the start of the day a schedule is developed for each vehicle. Then as additional demands become known or delays are incurred the problem is solved again either with the same method or with local optimization techniques that seek to preserve the original schedules to the greatest extent possible. The start of day problem must be solved in no more than fifteen minutes on a standard desktop computer like the ones typically found in local dispatching operations. In practice operational considerations (mainly customer service and driver safety and
efficiency issues) favor the partitioning of large problems into those containing roughly twenty drivers and less than one hundred loads. In this paper we focus on problems of that size. A longer-term goal of this work is to be able to solve off-line problems including as many as one hundred drivers and five hundred loads so that a comparison can be made between operations with sub fleets of various sizes. Walker (1992) demonstrated that significant economies of scale exist for short haul truckload operations by examining ground operations providing service to a rail yard. It remains to be seen if those results apply to single companies, who may take advantage of economies of scale despite sub-fleet and sub-area partitioning. That question, of considerable interest, can be examined using the methods originally developed for real-time implementation. In this paper we examine the method for generating the start of day solution. Implementation plans and an examination of the trade-offs of using local rather than global techniques to change schedules as new demands unfold or changes occur in the system may be found in Regan et al., (1999).
Related Work

A general formulation for the vehicle routing problem with time windows, presented in Desrosiers, Dumas, Solomon and Soumis (1995, p 86), is given here:

\[
\text{Min } \sum_{k \in K} \sum_{(i,j) \in A^k} C_{ij} X_{ij}^k
\]

(1.0)

\[
\sum_{k \in K} \sum_{j \in N \cup \{d(k)\}} X_{ij}^k = 1, \quad \forall i \in N
\]

(1.1)

\[
\sum_{k \in K} X_{o(k)j}^k \leq U,
\]

(1.2)

\[
\sum_{j \in N \cup \{d(k)\}} X_{o(k)j}^k = 1, \quad \forall k \in K
\]

(1.3)

\[
\sum_{i \in N \cup \{o(k)\}} X_{i}^k - \sum_{i \in N \cup \{d(k)\}} X_{i}^k = 0, \quad \forall j \in V^k \setminus \{o(k), d(k)\}, \forall k \in K
\]

(1.4)

\[
\sum_{i \in N \cup \{o(k)\}} X_{i}^k = 1, \quad \forall k \in K
\]

(1.5)

\[
X_{ij}^k (T_i^k + t_{ij} - T_j^k) \leq 0 \quad \forall k \in K, \forall (i,j) \in A^k
\]

(1.6)

\[
a_i \leq T_i^k \leq b_i \quad \forall k \in K, \forall i \in V^k
\]

(1.7)

\[
X_{ij}^k (L_i^k + l_j - L_j^k) \leq 0 \quad \forall k \in K, \forall (i,j) \in A^k
\]

(1.8)

\[
\ell_i \leq L_i^k \leq Q^k \quad \forall k \in K, \forall i \in N \cup \{d(k)\}
\]

(1.9)

\[
L_{o(k)}^k = \ell_{o(k)}
\]

(1.10)

\[
X_{ij}^k \text{ is binary, } \quad \forall k \in K, \forall (i,j) \in A^k
\]

(1.11)

Where \(N = \{1 \ldots n\}\) is the set of customers and \(K\), indexed by \(k\), be the set of available vehicles to be routed and scheduled. \(\nu\) is the number of vehicles that can be assigned. The graph \(G^k\) is consisted of the set of nodes \(V^k\) and the set of arcs \(A^k\). \(\ell_j\) is the load at location \(j\). \(o(k)\) is the origin of vehicle \(k\) while \(d(k)\) is the destination of vehicle \(k\). \(T_i^k\) is the start time of service at location \(i\) by vehicle \(k\).
Solution approaches for this problem have typically relied on Lagrangian relaxation or decomposition methods. These methods rely on the solution of shortest path sub-problems with time window or in more general terms, resource constraints. According to Desrosiers, Soumis and Sauve (1983), in order to obtain satisfactory bounds with Lagrangian relaxation when the gap between $Z_{lp}$ and $Z_{lp}$ is large, integer sub-problems obtained without relaxing time window constraints must be solved, where $Z_{lp}$ is the objective value of the linear relaxation of the integer problem. One of the requirements of these algorithms is the solution of the elementary shortest path problem with time window (ESPPTW) constraints. The ESPPTW is NP-hard in the strong sense, as has been shown by reduction from the problem denoted sequencing within interval (Dror, 1994). Shortest path problems with time window constraints are typically solved using dynamic programming. When using dynamic programming, non-elementary shortest paths exist and deteriorate the quality of the lower bound obtained from the coordinating master problem (Desrosiers et al., 1995). In addition, dynamic programming usually involves a very large state space for any real-life transportation system (Levin, 1971). All this suggests that while dynamic programming is one possible method to solve the SPPTW, it is not a perfect method. In this work we search for an alternative.

An alternative way to deal with the time window constraints is to discretize the time windows. This method is not a new one, in fact, early work applying this method can be seen in Appelgren (1969) and Levin (1971). Appelgren (1969) solved a ship scheduling problem using time window discretization method. According to Levin (1971), who presents and directly solves an integer programming formulation including discretized time windows for minimization of aircraft fleet size, the method of discretization of the time windows achieves good results. However, neither author gives an indication of how far away their solutions are from the optimal value of the original problem. Most likely this is because at that time the original problem could not be solved. Nor do they discuss the effect of the discretization of the time windows using different intervals on the gap between the optimal value associated with the discretized formulation and the optimal objective value of the original problem. More recent application of time window
discretization can be seen in Swersey and Ballard (1984), where a school bus scheduling problem is solved using a time window discretization method to minimize fleet size, linear programming relaxation of the resulting integer programming problem is solved. For most instances, the solutions are integral; otherwise they are able to modify the solution so as to obtain integrality without increasing the fleet size.

In this paper, we also present a method based on the discretization of time windows. We revisit this general approach for two reasons. The first is that alternative methods in which time constraints are linearized have not been shown to produce good solutions. The second is that recent advances in computing allow us to implement a high level of discretization, thereby obtaining good solutions.

**Problem Description**

We examine a local truckload pick up and delivery problem. Without loss of generality, we assume a homogeneous fleet of vehicles and containers. Test problems mimic intermodal operations near a rail facility and assume that all loads either begin or end at the intermodal facility. This too is without loss of generality as the method used is suitable for any local truckload pickup and delivery problem. Each load that must be moved has a strict time window for service. Loads may have both pickup and delivery time windows but we consider only pickup time windows. We assume that the first available time of the load can be modified to take delivery time windows into account or that if only delivery time windows exist that these can be modified to be pickup time windows. After the start of the day, vehicles are scattered throughout the service region. They need not return to the depot after providing service. A vehicle can only serve one load at a time. After moving a load it turns to another demand directly or remains idle. The objective is to minimize the total cost of providing service to all loads within their time constraints. In the problems described in this paper the size of the fleet is fixed though the number of vehicles used could also be an endogenous rather than exogenous variable. The objective
is to serve as many demands as possible and at the least cost. We do not require that every load be served since we usually do not know prior to solving the problem if the fleet has sufficient capacity to serve all customers. This is operationally feasible since in practice, trucking companies regularly contract out some loads to dray operators. A large bonus is added to each assignment to guarantee that as many loads as possible are served while minimizing the total cost.

**Problem Formulation**

Notation: Let \( N = \{1, \ldots, n\} \) be the set of loads and \( K \), indexed by \( k \), be the set of available vehicles to be assigned. Consider the graph \( G = (V, A) \) consisting of the set \( V \) of nodes and \( A \) of Arcs. For each arc \( (i,j) \in A \), \( C_{ij} \) represents the cost of serving demand \( j \) directly after demand \( i \). In a more general form the cost would be the revenue gained by moving the load minus the cost of making the move. In our case only the cost is considered since most of the revenue-generating portion of the intermodal moves are not relevant to the local operations. Let \( t_{ij} \) represent the travel time from the destination of the load \( i \) to the origin of load \( j \) plus dock time, and loaded travel time associated with load \( i \). That is, it represents the time between the start of service to load \( i \) and the start of service to load \( j \). Let \( y_{ij} \) represent the idle time associated with assignment of a vehicle to load \( j \) after load \( i \). The parameter \( \beta \) represents the rate assigned to the cost of this idle time. Let \( \alpha \) be the weight assigned to the time of service. If there is a perception that better customer service includes not only serving loads within their time windows but also early in their time windows then this coefficient is assigned a non-zero positive value this parameter can also be assigned a different value for different customers (in that case it would be indexed as \( \alpha_i \)). The parameters \( [a_i, b_i] \) represent the beginning and ending of the service time window for load \( i \). \( M \) is a large positive number.

Formulation: The problem of finding the minimum cost set of routes can be formulated as follows:
Max \[ \sum_{i \in N} \sum_{j \in N(i)} (M - c_{ij})x_{ij} - \alpha \sum_{i \in N} T_i - \beta \sum_{i \in N} \sum_{j \in N} y_{ij} \] (2.0)

\[ \sum_{j \in N} x_{O_i,j} \leq 1 \quad \forall i \in K \] (2.1)

\[ \sum_{i \in N \cup O_k | k \in K} x_{i,j} \leq 1 \quad \forall j \in N \] (2.2)

\[ \sum_{i \in N \cup O_k | k \in K} x_{i,j} - \sum_{m \in N} x_{j,m} \geq 0 \quad \forall j \in N \] (2.3)

\[ x_{ij}(T_i + t_{ij} - T_j) \leq 0 \] (2.4)

\[ a_i \leq T_i \leq b_i \quad \forall i \in N \] (2.5)

\[ y_{ij} = x_{ij} \left( T_j - T_i - t_{ij} \right) \quad (i,j) \in A \] (2.6)

\[ x_{ij} \text{ is binary; for all } i \in N + \{0\}, j \in N, i \neq j \] (2.7)

**SOLUTION APPROACH**

The time constraints make it very difficult to solve this formulation for problems of reasonable size. For this reason, the explicit time constraints are eliminated in our algorithm. Instead, time constraints are taken into account in a pre-processing step in which two versions of the problem are constructed. The first is over constrained and the second is likely under constrained.

The flow variable \( x_{ij} \) represents links between tasks (loads). Feasible links are determined by the time window constraints associated with each load. If the time constraints are points rather than windows then the problem is reduced to a fixed schedule problem, whose starting time for the task is just a time point. This kind of problem has a clear and exact network representation and may be solved very quickly.

Suppose we consider only the end points of the time window of both the first and the second task when we determine the accessibility between the two tasks; then we will obtain a set of links that ignores some possible connections. For example, in the cases in
which the vehicle arrives at the first task earlier than the latest point of the time window, the vehicle is still possibly able to reach the second task. We refer to this as the over constrained network. Suppose now that we only consider the starting point of the time window of the first task and the end point of the time window of the second task when we determine the accessibility between the two tasks. Then we will get a link that includes some infeasible accessibility. We refer to the network created in this way as the under constrained network. The over constrained method produces a network from which we can obtain a feasible solution. However, the solution is not optimal with respect to the original problem.

Figure 1) shows how impossible links are included in the under constrained problem and how possible links are excluded in the over constrained problem.

Figure 1) Feasible and infeasible regions in over and under constrained representations
load i)  

load j)  

\[ t_j \quad \text{time} \]

Figure 2) Diagram showing the possibility of cycles in the under constrained network

Figure 1 a) illustrates the fact that many feasible connections are excluded by over constrained method. In fact it is feasible for vehicle to start out at a time within the gray area and serve load \( j \). Figure 1 b) illustrates the converse case. Many infeasible links may be included by the under constrained method. Wherever the vehicle leaves at a time within the gray area, load \( j \) is not accessible.

Figure 2) illustrates another point. The network in the under-constrained method could contain cycles (sub-tours) while the network structure of the over constrained method prevents sub-tours. It is never optimal for the same vehicle to return to a node it has already served.

In the over constrained method we replace constraints (2.4) and (2.5) with the following:

\[ X_{ij} (b_i + t_j - b_j) \leq 0 \]  \hspace{2cm} (2.8)

In the under constrained method we replace constraints (2.4) and (2.5) with the following:

\[ X_{ij} (a_i + t_j - b_j) \leq 0 \]  \hspace{2cm} (2.9)
Constraints (2.8) and (2.9) determine the links that decision variables can choose from. So the formulation becomes a network problem without explicit time constraints. If the coverage constraints (1) are relaxed, the formulation of the over constrained problem is a network flow problem on an a-cyclic network. There are very efficient methods to solve such problems. However the network generated using the under constrained formulation is likely to contain cycles as can be seen in figure 2.

If we use the formulation from the under constrained method, there is some infeasible space included in the solution space if the time window is large. While for over constrained problem, there will be some feasible space excluded. As a result, the optimal solution of the linear relaxation of the formulation by the least conservative method $Z_t^{LP}$, the magnitude of the integer solution from the least conservative method $Z_t^{IP}$, most conservative method $Z_m^{IP}$ and the global optimal integer solution $Z_o^{IP}$ can be placed in the following order:

$$Z_t^{LP} \geq Z_t^{IP} \geq Z_o^{IP} \geq Z_m^{IP}$$

The over constrained method provides a feasible solution. The pure network structure leads to fast solution times. However, the under constrained method only provides us a sense of the maximum amount by which our solution could be from the optimal solution value.

The reduction of the time window to a single latest time point to consider the linkages means the loss of possible linkages between tasks. The vehicle arrives at point a task, but can only be assigned a subsequent task that is accessible from the last possible point in the time window of the current task. This is equivalent to making the vehicle wait for some time before it leaves for the next task. This results in a loss of opportunity. The larger the time windows and the longer the assignment chain, the greater the lost opportunities and the bigger the gap between lower and upper bounds. For this reason, this method is more suitable for problems in which only a small number of tasks are assigned to each vehicle.
than for the typical VRPTW. It appears ideally suited to the problem we wish to solve.

If the time windows are fairly tight the over constrained method should be able to provide an acceptable solution for the truckload pick up and delivery problem, in which each driver is typically assigned three to four loads per day and the number of loads assigned to a single vehicle is typically less than twice that number (a single vehicle may be used by more than one driver).

**Time Window Discretization**

In general, the bigger the time windows, the bigger the gap between $Z_t^{IP}$ and $Z_o^{IP}$, as well as $Z_o^{IP}$ and $Z_m^{IP}$. Sometimes the gap between the two methods is so large that we cannot determine if we have reached an acceptable solution with respect to the optimal value.

The discretization method is based on the idea that if the time windows are smaller, the gap between $Z_m^{IP}$ and $Z_t^{IP}$ will be reduced. To do this we duplicate each task and require that only one of the duplicate tasks be served. If we divide the time window properly, we will get as an exact result as the global optimum through the over constrained method. In the time splitting method, the under constrained method can only provide an upper bound for us to determine the scope of the distance from the current feasible solution to the optimum.

We can prove that to any single problem, there exists one method of splitting the time windows with the least number of the subtasks, which can eliminate the gap between $Z_m^{IP}$ and $Z_o^{IP}$. The subtasks are those tasks whose time windows are part of the time window of the original task. The service of any of the subtasks causes the fulfillment of the original demand. Suppose there is an optimal assignment for the original problem with the vehicle arrival time $t_j, j \in \mathbb{N}$, then split the time window at the point $t_i$, each task being divided into two parts (if $t_i$ doesn't coincide with $a_i$ or $b_i$). The optimal assignment must be included in the feasible space defined by the formulation of the over constrained
method. In this case, the minimum number of the subtasks is less than or equal to \(2|N|\). This means that the minimum number sub-tasks that contain the original optimal solution is \(2|N|\).

If we split the time windows with the same interval, there must be an interval of such a unique maximum width that will contain the optimal result by the formulation from the under constrained method. This length can be as following:

\[
\text{Max}\{ \frac{t_i - a_i}{\sigma}, \frac{b_i - t_i}{\sigma} \in Z, \forall i \}, \quad Z \text{ is the set of natural numbers,}
\]

\(t_i\) is the service time for task \(i\) at optimal assignment.

If this number is too small, the extended network is too large to solve the problem efficiently. In order to guarantee that the optimal solution will be found for every problem, this interval would need to be infinitesimal. In that case our formulation would be much less efficient than the ones it was intended to replace. However, we are not pursuing the optimal solution directly. Our goal is to very quickly obtain an acceptable solution.

The problem is how to determine the magnitude of the interval to split the time window for an acceptable result relative to the optimal solution. The proper width varies according to characteristics of the problems. We present results based on time-window splitting using an interval from two hours to 0.2 hours.

This method can be further improved by employing time window reduction methods commonly used for preprocessing VRPTW problems. There may be some part of the time window which is of no use to the assignment since no vehicle is able to reach the demand within that time interval. We should eliminate that part of the window in order to reduce the size of the problem. This method is described in Desrochers, Desrosiers and Solomon (1992). For simplicity, we just use \(a_i, b_i\) as the ends of reduced time windows. Then the formulation after window split can be modified as following:
\( \omega \) = set of all nodes of subtasks;  
\( \delta(j) \) is the original node that has the subtask \( j \); \( \delta(j) \in \mathbb{N} \)

in the same way, for under constrained problem, we replace equation (4) with

\[
X_{ij}(a_i + t_{ij} - b_j) \leq 0 \quad \forall \delta(i) \neq \delta(j), (i,j) \in A \quad (4')
\]

for over constrained problem, we replace equation (4) with

\[
X_{ij}(b_i + t_{ij} - b_j) \leq 0 \quad \forall \delta(i) \neq \delta(j), (i,j) \in A \quad (4'')
\]

such that we can eliminate the time window constraints in the formulation explicitly.
Correspondingly the waiting time should become, for under constrained case:

\[
y_{ij} = \text{MAX}(a_i - b_i - t_{ij}, 0) \quad \forall \ (i,j) \in \omega, \ \delta(i) \neq \delta(j), \text{ and } x_{ij} \geq 0 \quad (6')
\]

for over constrained case:

\[
y_{ij} = b_i - b_i - t_{ij} \quad \forall \ (i,j) \in \omega, \ \delta(i) \neq \delta(j), \text{ and } x_{ij} \geq 0 \quad (6'')
\]

in the objective function, the service time \( T_i \) becomes \( a_i, b_i \) respectively for under constrained and over constrained cases.
\[
\text{obj Max } \sum_{i \in \mathcal{N} + \{0\}} \sum_{j \in \mathcal{N}[i]} (M - c_{ij})x_{ij} - \alpha \sum_i T_i - \beta \sum_{i,j} y_{ij}
\]

\[
\sum_{j \in \omega} x_{ij} \leq 1 \quad \forall i \in \mathcal{K} \quad (1)
\]

\[
\sum_{i \in \omega + \{0\} \setminus \mathcal{K}} x_{ij} \leq 1 \quad \forall j \in \omega \quad (2)
\]

\[
\sum_{[j, \delta(0) = \xi]} \sum_{i \in \omega + \{0\} \setminus \mathcal{K}} x_{ij} \leq 1 \quad \forall \xi \in \mathcal{N} \quad (2')
\]

\[
\sum_{i \in \omega + \{0\} \setminus \mathcal{K}} x_{ij} - \sum_{m \in \omega \setminus \{0\}} x_{jm} \geq 0 \quad \forall j \in \omega \quad (3)
\]

\[
x_{ij}(T_i + t_{ij} - T_j) \leq 0 \quad \forall \delta(i) \neq \delta(j), (i,j) \in \mathcal{A} \quad (4)
\]

\[
a_i \leq T_i \leq b_i \quad \forall i \in \omega \quad (5)
\]

\[
y_{ij} = x_{ij}(T_j - T_i - t_{ij}) \quad (i,j) \in \mathcal{A} \quad (6)
\]

\[X_{ij}\text{ is binary; for all } i \in \omega + \{0\} \setminus \mathcal{K}, j \in \mathcal{N}, i \neq j\]

**Problem Testing**

Using the GIS package TransCAD, we generated a set of problems to test the algorithm using real data to generate a set of representative problems. The problem generation package is part of a larger GIS based fleet management simulation model described in Jagannathan, (1999). The demands are generated by selecting randomly from among known customers in the service area. Time windows are randomly assigned based on the distribution shown in table 1, which roughly corresponds to the time windows associated with loads known at the start of day.
Table 1. Probabilities associated with time windows of varying length

<table>
<thead>
<tr>
<th>Time window</th>
<th>7:00-7:30 AM (0.5 hours)</th>
<th>8:00-9:30 AM (1.5 hours)</th>
<th>8:00AM-12:00PM (4 hours)</th>
<th>12:00-5:00PM (5 hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probability</td>
<td>0.1</td>
<td>0.15</td>
<td>0.35</td>
<td>0.4</td>
</tr>
</tbody>
</table>

For the problems in the test set, we begin with the vehicles at the depot (which in this problem is very near the rail yard) and make all vehicles available for the duration of the day. Vehicles are not required to return to depot after each service. Travel distances correspond to the shortest network travel distance. Travel time is assumed to be 35 miles per hour, reflecting congestion levels in the test region. The average loaded distance is quite short, less than twenty miles long. A handling (typically dock time) of forty minutes is assumed for each load though in an actual operation the handling time would be customer specific. We present here results related to solving 30 problems of 20 vehicles and 75 loads. Twenty was selected because is the typical maximum size of a local sub-fleet handled by a single dispatcher.

Results presented in this paper focus on a restricted formulation in which the objective function includes only the cost of empty travel. So, the objective function becomes the following:

\[
\text{obj} \quad \max \sum_{i \in N^+ \cup \{0\}, j \in N} \left( M - C_{ij} \right) X_{ij}
\]

The reason for this restriction is to facilitate an examination of the gap between the solution values of the over and under constrained problems. These values are less stable when the multiple objective formulation is used.

The interval to split the time window at each iteration is as follows:

Table 2. Interval for time window splitting

<table>
<thead>
<tr>
<th>Iteration</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length in</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>hours</td>
<td>2</td>
<td>1.75</td>
<td>1.5</td>
<td>1.25</td>
<td>1.00</td>
<td>0.75</td>
<td>0.7</td>
<td>0.6</td>
<td>0.5</td>
<td>0.4</td>
<td>0.3</td>
<td>0.2</td>
</tr>
</tbody>
</table>
We implement the interval in the following way. Divide the time window by the interval. If the result is integer, split the time window into this number of pieces. If not, divide the time window by the interval and raise the result to the nearest integer; split the time window into this number of pieces.

We solve the problems using CPLEX with no special modification. Naturally, the solution times could be improved significantly by the addition of a more sophisticated decomposition or relaxation method.

**Test Results**

In these tests we examine the quality of the solutions found relative to the length of solution times. All tests are run on an inexpensive desktop computer, a 233 MHZ Pentium II machine with 128 MB of ram. Several rules are used to terminate calculation. If, after sufficient discretization of the time windows the problem becomes too large to solve using CPLEX we terminate and execute the best feasible solution found so far. If the ratio of the lower and upper bounds is above 95% then we terminate and execute the solution. Or, if we have attempted to solve a problem with time window reduced to 0.20 hours, then we terminate and execute the solution. Figure three shows how the average value of the gap fluctuates over the first eight iterations for those problems considered still unsolved after eight iterations. Most of the problems are considered solved prior to the eighth iteration so those shown represent the problem instances that are most difficult to solve. It may observed that for those problems not easily solved that the ratio of the upper to lower bound solution changes very little after the first four iterations. For this reason, for real-time implementation we recommend implementing the solution found after the fourth or fifth iteration rather than waiting to find an improvement. Table 3 shows that the average time required for the first four iterations is around four minutes while the average time required for the first six iterations is around eighteen minutes, an unacceptable length of time for the application at hand.
Table 3. Ratio of lower and upper bound values after each iteration

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Mean</th>
<th>Range</th>
<th>Fraction of Problems remaining</th>
<th>Average Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.84</td>
<td>(0.590, 0.996)</td>
<td>1.00</td>
<td>37.93</td>
</tr>
<tr>
<td>2</td>
<td>0.86</td>
<td>(0.658, 0.988)</td>
<td>0.87</td>
<td>92.10</td>
</tr>
<tr>
<td>3</td>
<td>0.86</td>
<td>(0.658, 0.978)</td>
<td>0.83</td>
<td>141.26</td>
</tr>
<tr>
<td>4</td>
<td>0.87</td>
<td>(0.658, 0.975)</td>
<td>0.70</td>
<td>256.81</td>
</tr>
<tr>
<td>5</td>
<td>0.87</td>
<td>(0.735, 0.942)</td>
<td>0.63</td>
<td>410.48</td>
</tr>
<tr>
<td>6</td>
<td>0.88</td>
<td>(0.746, 1.000)</td>
<td>0.60</td>
<td>1096.22</td>
</tr>
<tr>
<td>7</td>
<td>0.88</td>
<td>(0.771, 0.954)</td>
<td>0.43</td>
<td>1644.11</td>
</tr>
<tr>
<td>8</td>
<td>0.86</td>
<td>(0.775, 0.918)</td>
<td>0.37</td>
<td>2592.93</td>
</tr>
</tbody>
</table>

Figure 3. The ratio of the lower to upper bound solution for those problems most difficult to solve.

Test results also show that the upper bound value does not change as much as the lower
bound value does as the problem is expanded. In fact it changes so little that it could be left out of the process with little or no impact. That is, rather than try to find a tighter and tighter upper bound we would instead use that time to generate a tighter lower bound, since the lower bound problem supplies us with a feasible solution.

From the test results presented in table 3, we can see that on average, the ratio at the first step is above 80% but that in some cases it many be as low as 60%. The interval to split the window is two hours at the first iteration. Included in the thirty problems was one in which three loads are rejected by the assignment made at the first iteration. Corrected in the second iteration, large intervals to split the time windows may lead to infeasibilities. However, for most of the problems tested two hours seems a reasonable interval to start with.
CONCLUSION

In this paper, a method to develop vehicle assignments for local truckload operations is presented. A set of problems based on real data has been examined. Test results presented in this paper focus on relatively small, but operationally realistic problems. Several obvious improvements may be made to this simple approach in order to be able to solve larger problems quickly. The structure of the method examined is such that it results in sub-problems that are simply network flow problems. Therefore, decomposition techniques may be applied along with time window discretization method discussed in order to solve larger problems in real-time. That extension is the subject of on-going work.

The development of relatively simple methods for solving time-window constrained truckload carrier fleet problems in real-time remains the goal of this work. Initial results are promising and we expect to reduce solution times and increase problem sizes significantly. However, in the method developed, in order to develop solutions within the time constraints of five minutes, the ratio of the upper to lower bounds can sometimes still be over 85%. In these cases we do not know how far we are from the optimal solution because the optimal results have been calculated for only a few instances. This too is a part of our on-going research. If we assume for now that the optimal solution is roughly the average of the upper bound and lower bound, we estimate that that usually we can get to within 90% of the optimal even with a very large interval for window splitting (and solution times of less than two minutes). It may be that in a highly stochastic environment in which we will need to re-solve the problem repeatedly during the day that an on-line algorithm that performs roughly at 90% of the optimal algorithm is acceptable.
Acknowledgements

The authors gratefully acknowledge the financial support of the TRB ITS-IDEA program, the University of California Transportation Center and JB Hunt Transportation, Inc. In addition we wish to thank Robert Logan, Gary Whicker Frank Broadstreet and Michael Hinkle of JB Hunt for their input to our research as well as that of Professor G. Don Taylor of the University of Arkansas. The research has also benefited from continued interaction with University of Texas researchers who are examining related problems. Finally we wish to thank Sreeram Jagannathan, for his assistance with many parts of this project, and Xiangwen Lu for his constructive suggestions. Any errors or omissions remain the sole responsibility of the authors.
REFERENCES


Transportation Research Board and for publication in the Transportation Research Record.

