VALUE OF ITS INFORMATION FOR CONGESTION AVOIDANCE IN INTER-MODAL TRANSPORTATION SYSTEMS[1]

Phase II

FINAL REPORT

March 2010

Principal Investigators:

Alper E. Murat (PI),
Assistant Professor,
Wayne State University

Ratna Babu Chinnam (Co-PI),
Assistant Professor,
Wayne State University

Snehamay Khasnabis (Co-PI),
Professor,
Wayne State University

[1] Throughout this document "Inter-modal Freight" refers to the "Shipment of freight involving more than one mode of transportation (road, rail, air, and sea) during a single, seamless journey".

Table of Contents **Page No**

SUMMARY OF RESULTS

Our project has four major mile-stones for the second year:

- **Mile-stone #1:** Develop Dynamic Inter-modal Transportation Optimization Models: For mostly air-road network and inter-modal networks significant to OHIO MICHIGAN regions and our collaborators
- **Mile-stone #2:** Develop Dynamic Operational Response Optimization Models: For supply chain operations identified in the case studies with collaborator Ford
- **Mile-stone #3:** Develop efficient heuristic solution algorithms for the above Dynamic Optimization Models
- **Mile-stone #4:** Validate Dynamic Optimization Models and heuristics using case study data

The Research Team, made up of Dr. Alper Murat (Project PI), Dr. Ratna Babu Chinnam (Project Co-PI), Dr. Snehamay Khasnabis (Project Co-PI), doctoral student – Farshid Azadian, has made very good progress with respect to all these four milestones over the second year.

In what follows, we summarize our achievement with respect to these milesones.

Mile-stones #1 & #2:

We have concentrated the majority of our efforts in developing stochastic models and algorithms for dynamic routing on intermodal networks in the form of Stochastic Dynamic Programming (SDP). We have met and exceeded our goals in terms of the practicality of the models and algorithms developed.

In particular, we have expanded the scope of the models developed in the first year, where we only considered a single inter-modal shipment. While this approach is suitable in most part for the shippers (e.g. for our collaborator Ford) and to some extent for the freight forwarders (e.g. C.H.Robinson), multiple shipments with milk run deliveries is the most common scenario in the freight forwarding industry and to some extent for the shippers with multiple facilities. For instance, Ford often receives inbound containers full of parts and then break-bulks them and delivers to different parts assembly plants. Therefore we have expanded the scope of our modeling approach and solution algorithms based on SDP to account for the multi-leg delivery and pick up aspects. The consideration of the multiple shipments (pickup and delivery) not only increases the complexity of the models but also makes the solution intractable.

In order to leverage our earlier work, we have again considered the *air-road inter-modal problem* and developed models and algorithms for solving this problem with alternative access airports. We developed a case study which considers Detroit and Toledo airports as the alternative access

airports. The problem, called ***dynamic air-cargo routing with milk run pickups and deliveries***, is to pickup multiple customer air-cargo shipments, deliver to different airports and airship them on the air network with dynamic routing using real-time flight information. In the case study, we consider a set of customers dispersed geographically in the Ohio-Michigan region. Each customer's shipment vary in terms of size (weight, volume) and has different shipment requirements (e.g. delivery time). The shipments are picked up by trucks and delivered to the selected airports for air shipment. The decisions are the customer-truck-trip assignments, truck-trip-airport assignments, truck-trip routes, and customer-flight assignments. This problem is a novel problem and extends the existing literature on vehicle routing problems. In the existing literature, the customer-flight assignment (and hence airport assignment) decisions are assumed to be made a priori. The routing decisions are then made given the customer-airport assignments. However, as we illustrated in our earlier work, it is advantageous to consider dynamic routing with real-time congestion information, especially for time sensitive air-cargo shipments originating from urban areas with multiple access airports. This work is described in Section A.

A limitation of the milk run air-cargo pickup and delivery is the assumption of deterministic travel times on the road network. Whereas the stochastic congestion modeling is important for the air network, the road network congestion is also important for dropping off the cargo shipments on time for loading on a particular flight. We have addressed this limitation by developing ***robust tour selection*** models and algorithms for truck milk run trips where the trucks are dynamically routed in the stochastic congested road network. This model considers a single truck (e.g. single tour) and a given set of customers whose loads are to be picked up. Hence this model serves as an accurate estimator of the tour delivery performance (on time and variability) for the milk run air-cargo pickup and delivery problem described above. In this work, we leveraged the synergies between this project and another project funded by DOT's University Transportation Centers Program (MI-OH UTC). The results are presented in Section B. While we have developed and tested the robust tour selection model formulations and algorithms, we have not integrated it with the milk run air-cargo pickup and delivery. This integration will be accomplished in the final year of the project.

As for the operational response to remedy the effect of congestion at intermodal facilities, we have developed dynamic operational response models which extend the previous year's static models.

Mile-stones #3 & #4:

We have approached the development of heuristic methods in two alternative ways. First, we have reduced the state space of our SDP formulations through state space filtering and state aggregation. Secondly, we have developed and tested the AO* heuristic algorithm for routing on the stochastic-time dependent air and road networks. While the preliminary results obtained with AO* is encouraging in the sense that the state space searched to a lesser extent than the SDP method, we found that its overhead calculations are time consuming and eliminates other

advantages. The benefits of the AO* is most notable whenever there is accurate lower bound information on the objective function. This information is seldom available given the sparsity of the data regarding dynamic air cargo routing. Hence we have primarily modified our exact SDP algorithms and developed heuristic versions through the state space filtering and state aggregation to reduce the computational burden. In fact, our algorithms can now find optimal or close to optimal solutions in minutes which is acceptable for practical applications.

For the dynamic operational response models, we have refined the Progressive Hedging Algorithm (PHA). We further developed hybrid heuristic algorithms combining the Sample Average Approximation with PHA. Our tests with these heuristic algorithms are still ongoing and will be completed in the first half of the final year.

Description of Sections A, B, and C

In Section A, we propose a modeling and solution framework for the dynamic air cargo routing on the air-road intermodal network with milk run customer pickups and airport deliveries. We show that the routing problem on the air-network can be separated from the routing on the road network after accounting for the cost-to-go profiles at the airport for each customer-flight assignment. We developed a stylized experimental study to illustrate the effect of number of trucks, number of customers, and number of alternative access airports on the performance. In addition, we present a case study in the Ohio-Michigan region where the Toledo Express airport (TOL) and the Detroit Metro Wayne County airport (DTW) are considered as the alternative access airports.

In section B, we relax the deterministic travel time assumption and consider the stochastic congestion on the road network. The trucks picking up customers' air cargo shipments and delivering to alternative access airports are subject to recurrent and non-recurrent traffic congestion en route. Hence, the routing decisions need to account for the delivery reliability to the airport. We developed an approach to determine robust tours for trucks where the truck picks up customers' air cargo shipments in such a way (e.g. sequence of visits) that the delivery reliability at the airport is increased. We report on experiments with and without time-windows at the customer locations. Results show that by dynamically routing between customer pairs, the trucks can avoid the congestion and increase the reliability of delivery to the airports.

In section C, we briefly summarize our efforts in developing heuristic methods for dynamic routing models on intermodal networks and operational response models. These heuristic methods are complementary to state space filtering and state aggregation heuristics that are applied in conjunction with the stochastic dynamic programming algorithm. For the dynamic routing on stochastic and time-dependent intermodal networks, we outline the AO* heuristic algorithm developed and tested. For the stochastic programming formulations obtained from the operational response models, we summarize our efforts in developing hybrid methods which combine the Sample Average Approximation with the Progressive Hedging Algorithm.

A. Dynamic Air-Cargo Routing on Air-Road Intermodal Networks with Milk Run Pickups and Deliveries

1.1 Problem Statement

Air-road intermodal routing consists of picking up shipments from a set of customers by a fleet of vehicles (e.g. trucks) and delivering them to the airports for air-shipment. The air-shipment includes routing on the air-network by selecting a flight sequence that carries the cargo from origin airport to destination either directly or by passing through connecting airports. Accordingly, this problem integrates two sub-problems: the road-network routing and the air-network routing. In the road-network routing, decisions are customer to truck assignment and customer visit schedule for each truck. On the other hand, in the air-network routing, a network of flights and transit airports are available. The operational decisions include selecting a sequence of flights for each customer's order. The connection point between these two sub problems is the customer-flight assignment, which dictates which airport a truck must visit to deliver orders and in what time window.

In this project, we focus on time-sensitive cargo delivery. According to the Bureau of Transportation Statistics, time-sensitive shipments, which are mostly characterized by light-weight expensive packages, are responsible for the 80% growth in air-freight shipping industry. The primary characteristics of the air-freight shipping are the agility and reliability of delivery. Accordingly, customers are willing to pay extra fare for these service characteristics. In this work, our main objective is to minimize the overall shipping cost that only includes flights' expenditures and delivery tardiness penalties. We ignore the vehicle costs for two reasons. First, the road transportation cost is not as significant as the cost of air transportation. Second, the vehicle expenditures that can be incorporated in the objective are merely variable costs directly related to the routing decisions. The fixed cost of acquiring and operating the trucks are in most part independent of the routing decisions.

The routing problem on air-network has been extensively studied in the earlier work. In that work, we present an efficient algorithm to construct routing policy on a given air-network in both deterministic and stochastic settings. As demonstrated, the routing policies characterize the optimum flight choice at each airport based on the arrival time of the cargo to the airport and flight network status. The objective is to minimize the expected overall shipping cost based on the real-time flight delay information available.

Since the routing on the air-network is extensively presented in the previous report, we will not repeat herein. Rather, we briefly discuss the relevant information whenever needed in the remainder of this work. In the earlier work, we determined the routing policy based on the expected cost-to-go of choosing an available flight from an airport given the real-time flight

departure information. Clearly, choosing a flight affects the future choice of flights through the flight path. Hence, the expected cost-to-go is based on both the immediate cost of using a flight and the cumulative cost of subsequent flight choices until the arrival to the destination airport. The cost-to-go of selecting a flight is also dependent on the time the decision is made.

An illustration of the time dependence of the cost-to-go of a flight is presented in Figure 1. In the deterministic setting (e.g., no stochastic flight departure delays at the origin airport), the cost-to-go of the flight is constant prior to the scheduled departure time. After the flight departure, the cost increases instantaneously to a failure penalty (e.g. not able to deliver order). On the other hand, in the stochastic setting (e.g., with stochastic departure delays at the origin airport), actual departure may happen later than that of the scheduled departure time. Accordingly, in contrast with deterministic setting, the cost-to-go increases gradually to the failure penalty. The difference, as illustrated in Figure 2, is a nonlinear curve for cost-to-go which can be approximated with piece-wise linear functions. With this approximation, the modeling and solution algorithm aspects for both the stochastic and deterministic cases are identical. Hence, we herein consider only the simple case (e.g. deterministic) setting as far as the departure delays at the origin airport are concerned. Note that the we still account for the stochastic departure delays in the remainder of the air network and the cost-to-go step functions are determined accordingly.

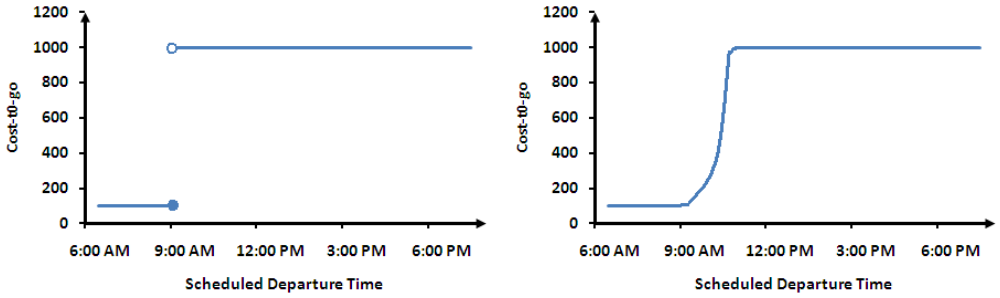
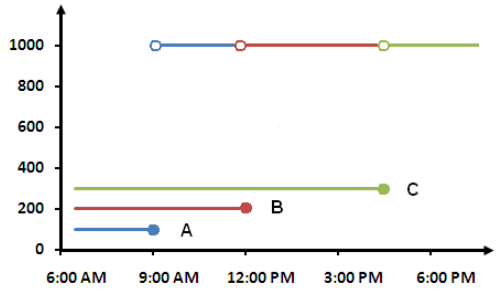


Figure 1. Schematic flight cost-to-go based on cargo arrival time for scheduled departure time at 9:00 am for deterministic (left) and stochastic (right)

Each airport has a set of flights available with varying scheduled departure times and destination airports. Given the arrival time to the airport, the optimum policy selects the flight with the minimum cost-to-go. This recommendation is time-dependent and thus will change as flights depart. Figure 2 illustrates a typical cost-to-go profile for an airport with three flight alternatives based on the cargo arrival time. As can be observed, flight A has the lowest cost-to-go before its departure time. After that, flight B would be the best choice, which in turn will be preferred over flight C. Accordingly, the flight selection policy for this example suggests choosing flights A, B and C at each marked time-window, respectively (policy table in Figure 2). It should be noted that, in practice, there are a large number of flights at each airport which can be chosen. However, given that the cargo is time-sensitive and thus has delivery due date, it is possible to filter out the set of flights with cost-to-go greater than or equal to the failure penalty. In other words, at each

airport, we only need to consider a finite set of flights when the cost-to-go is less than equal to the delivery failure penalty.



Policy	
Time Table	**Cost-to-go (flight)**
6:00 – 9:00	100 (A)
9:00 – 12:00	200 (B)
12:00 – 16:00	300 (C)
16:00 – 24:00	1000 (-)

Figure 2. Typical flights cost-to-go at an airport (left) and relative policy table (right)
Flight departure times: A=9:00 am, B=12:00 pm, C 4:00 pm

For the case with deterministic departures from the origin airport, the cost-to-go profile for an airport is a time dependent step function based on the best choice of flight at a given cargo arrival time to the airport. Accordingly, the optimum cost-to-go from an airport only depends on the cargo arrival time to the airport. Therefore, the air-network routing problem is dependent on the solution of the road-network routing problem through the cargo arrival time. By characterizing the cost function of the road network routing problem as the cost-to-go at the airport, we can separate these two problems. This is the key distinction from the previous research.

Based on the separability of the air-network routing and road-network routing problems, in the rest of this section, we focus on the modeling and solution of the road-network routing of the intermodal time-sensitive cargo shipments. First, we present the problem formulation and discuss different approaches to improve its solution efficiency. Next, we study the benefit of alternative access airports and analyze the effect of problem parameters on the objective function. We perform this sensitivity analysis via a simulation study on a sample set of problems. Next, we present and discuss the case study results of the problem on the southeast Michigan northwest Ohio region. Finally, we conclude this section by summarizing the findings.

1.2 Problem Formulation

Let $G \equiv (V, A)$ be the directed graph that represents the problem network. Let V be a set of nodes (e.g. depot, customer, and airports) and A a set of arcs connecting the nodes. The set of nodes consists of a depot, a set of customers (denoted as C), and a set of airports (denoted as H). The customer pickup and airport deliveries are performed by a set of trucks (K) each of which departs from the depot, visits a set of customers to pick up their orders and then delivers them to an airport. This cycle can be repeated until all the orders are picked up and delivered to an airport, e.g. a truck can make multiple trips. From now on, we refer each path that starts from the depot or

an airport and ends at an airport as a *trip*; let T denote the set of trips. For each airport $h \in H$, let R_h denote the set of flights available. Each flight r has a scheduled departure time noted as Q_{hr}. Each customer $i \in C$ has an order of size u_i. Each customer i has a cost-to-go of F_{ihr} at airport h if shipped by flight r which is calculated based on its delivery destination and due date. Since order splitting is not allowed, a customer's order must be shipped on the same flight as a whole. The flight time on each air-network arc, connecting node i to j, is noted by l_{ij}. The depot opening and closing times are θ_d^O and θ_d^C, respectively. All trucks can leave and must return during the depot's operating hours. Let a_i be the arrival time at customer i and z_h^{kt} be arrival time of truck k at airport h in t^{th} trip. Besides a_i and z_h^{kt}, other decision variables are x_{ij}^{kt}, which indicates whether node i is visited right after node j by truck k in trip t,, and y_{ihr}^t, which indicated whether the order of customer i is picked up in the t^{th} trip to be shipped by flight i departing from airport h.

Our problem is formulated as follows:

$$Min \quad \sum_{i\in C}\sum_{h\in H}\sum_{r\in R_h}\sum_{t\in T} u_i y_{ihr}^t F_{ihr} \tag{1}$$

$$\begin{cases} x_{hj}^{1t} = 0 & \forall j \in V \quad \forall h \in H \quad \forall t \in T \\ x_{dj}^{kt} = 0 & \forall j \in V \quad \forall k \in K \mid k > 1 \quad \forall t \in T \\ x_{jd}^{kt} = 0 & \forall h \in H \quad \forall k \in K \quad \forall t \in T \\ x_{dh}^{kt} = 0 & \forall h \in H \quad \forall k \in K \quad \forall t \in T \end{cases} \tag{2}$$

$$\sum_{t\in T}\sum_{k\in K}\sum_{j\in V\setminus i} x_{ij}^{kt} = 1 \quad \forall i \in C \tag{3}$$

$$\sum_{j\in V\setminus d} x_{dj}^{1t} \le 1 \quad \forall k \in K$$
$$\sum_{h\in H}\sum_{j\in V\setminus h} x_{jh}^{1t} = \sum_{j\in V\setminus i} x_{dj}^{1t} \tag{4}$$

$$\sum_{j\in V\setminus i} x_{ji}^{kt} - \sum_{j\in V\setminus i} x_{ij}^{kt} = 0 \quad \forall i \in C; \quad \forall k \in K; \quad \forall t \in T \tag{5}$$

$$\sum_{j\in V\setminus i} x_{ji}^{kt} - \sum_{j\in V\setminus h} x_{hj}^{k(t+1)} = 0 \quad \forall h \in H; \quad \forall k \in K; \quad \forall t \in T \mid t < |T| \tag{6}$$

$$x_{ij}^{kt}\left(a_i + l_{ij}\right) \le a_j \quad \begin{matrix} \forall (i,j) \in C, j \ne i; \\ \forall k \in K; \quad \forall t \in T \end{matrix} \tag{7}$$

$$\sum_{j\in V\setminus h} x_{jh}^{kt}\left(\theta_d^O + \sum_{\substack{p\in T \\ p\le t}}\sum_{\substack{(i,j)\in V \\ i\ne j}} x_{ij}^{pt} l_{ij}\right) \le z_h^{kt} \quad \begin{matrix} \forall k \in K; \quad \forall t \in T; \\ \forall h \in H \end{matrix} \tag{8}$$

$$y_{ihr}^{kt}\, z_h^{kt} \le Q_{hr} \quad \begin{matrix} \forall i \in C; \quad \forall k \in K; \quad \forall r \in R_h \\ \forall t \in T; \quad \forall h \in H \end{matrix} \tag{9}$$

$$\sum_{h\in H}\sum_{r\in R_h} y_{ihr}^{kt} \le \sum_{\substack{p\in K\\ p\le k}}\sum_{j\in V\setminus i} x_{ij}^{pt} \quad \forall i\in C;\quad \forall k\in K;\quad \forall t\in T \tag{10}$$

$$\sum_{h\in H}\sum_{t\in T}\sum_{k\in K}\sum_{r\in R_h} y_{ihr}^{kt} = 1 \quad \forall i\in C \tag{11}$$

In this model, the objective function is to minimize the overall shipping cost; shipping cost is calculated based on summation of cost-to-go of the selected flights. The cost-to-go of each flight includes both the flight cost and any delivery tardiness penalties. Constraints set (2) prevents infeasible connections. For instance, any direct connection from depot to airport is not allowed since there is no reason for an empty truck to visit an airport. Constraints (3) indicate that all customers must be visited exactly once. Constraints (4) allow trucks to leave the depot only in the first trip. Moreover, all the trucks that left the depot in their first trip must end their first tour in one of the airports. Constraints (5) and (6) impose flow conservation for customers and airports respectively; if a truck arrives at a customer during a trip, it must leave that customer in the same trip. As for the airports, if a truck arrives at an airport during a trip, it must leave that airport in the next trip. Constraints (7) calculate the arrival time at customers. The main purpose of these constraints is to prevent the sub-tours by preventing cycles consisting of only the customers. In this aspect, these constraints are similar to the Miller-Tucker-Zemlin sub-tour elimination constraints for the traveling salesman problem (Miller et al., 1960). Constraints (8) calculate the arrival time at the airports for each truck and for each trip. Constraints (9) prevent assigning a customer's cargo shipment to a flight if the airport arrival time for the truck carrying that customer's cargo is later than the flight's scheduled departure time. Similar to (9), constraints (10) prohibit assigning customers' cargo shipments to flights departing from unvisited airports. Constraints (11) imply that all the customer shipments must be assigned to a flight.

It is possible to improve the efficiency of this model by tightening the constraints via introducing aggregated variables. The following constraints (12) and (13) define decision variables $s_i^{kt}$and w_i^{kt}that play as aggregated variables for connecting decision variables x_{ij}^{kt}; for instance, s_i^{kt}indicates whether node *i* has any outgoing in trip *t* by truck *k*. Empirical study supports the efficiency of this approach of formulation.

$$s_i^{kt} = \sum_{j\in V\setminus i} x_{ij}^{kt} \quad \forall i\in V;\quad \forall k\in K;\quad \forall t\in T \tag{12}$$

$$w_i^{kt} = \sum_{j\in V\setminus i} x_{ji}^{kt} \quad \forall i\in V;\quad \forall k\in K;\quad \forall t\in T \tag{13}$$

This model is nonlinear as a result of constraints (9), (10) and (11). By defining *M* as a very large number, we can rewrite these constrains in a linear form so as to make the model linear.

$$\left(x_{ij}^{kt}-1\right)M+a_i+l_{ij}\le a_j \quad \begin{matrix}\forall (i,j)\in C,\ j\ne i;\\ \forall k\in K;\quad \forall t\in T\end{matrix} \tag{14}$$

$$\theta_d^O+\sum_{\substack{p\in K\\ p\le k}}\sum_{\substack{(i,j)\in V\\ i\ne j}}x_{ij}^{pt}l_{ij}+(w_h^{kt}-1)M\le z_h^{kt} \quad \begin{matrix}\forall k\in K;\quad \forall t\in T;\\ \forall h\in H\end{matrix} \tag{15}$$

$$z_h^{kt}\le\sum_{r\in R_h}Q_{hr}y_{ihr}^{kt}+\left(1-\sum_{r\in R_h}y_{ihr}^{kt}\right)M \quad \begin{matrix}\forall i\in C;\quad \forall k\in K;\\ \forall h\in H;\quad \forall t\in T\end{matrix} \tag{16}$$

The validity of these constraints is ensured by setting $M\ge z_h^{kt}\ \forall h\in H,\forall k\in K,\forall t\in T$. Linearization of the model is important since most of the available algorithms can handle linear models far better than non-linear models, especially when it comes to finding the optimal solution. Accordingly, the revised formulation for the problem is as follows.

$$Min\quad \sum_{i\in C}\sum_{h\in H}\sum_{r\in R_h}\sum_{t\in T}u_i y_{ihr}^t F_{ihr} \tag{1}$$

$$s_i^{kt}=\sum_{j\in V\setminus i}x_{ij}^{kt}\quad \forall i\in V;\quad \forall k\in K;\quad \forall t\in T \tag{12}$$

$$w_i^{kt}=\sum_{j\in V\setminus i}x_{ji}^{kt}\quad \forall i\in V;\quad \forall k\in K;\quad \forall t\in T \tag{13}$$

$$\begin{cases} x_{hj}^{1t}=0 & \forall j\in V \quad \forall h\in H \quad \forall t\in T\\ s_d^{kt}=0 & \forall k\in K\mid k>1 \quad \forall t\in T\\ x_{jd}^{kt}=0 & \forall h\in H \quad \forall k\in K \quad \forall t\in T\\ x_{dh}^{kt}=0 & \forall h\in H \quad \forall k\in K \quad \forall t\in T \end{cases} \tag{2}$$

$$\sum_{t\in T}\sum_{k\in K}s_i^{kt}=1\quad \forall i\in C \tag{3}$$

$$\begin{aligned} &s_d^{1k}\le 1\quad \forall k\in K\\ &\sum_{h\in H}w_h^{1k}=s_d^{1k}\end{aligned} \tag{4}$$

$$w_i^{kt}-s_i^{kt}=0\quad \forall i\in C;\quad \forall k\in K;\quad \forall t\in T \tag{5}$$

$$w_h^{kt}-s_h^{k(t+1)}=0\quad \forall h\in H;\quad \forall k\in K;\quad \forall t\in T\mid t<\left|T\right| \tag{6}$$

$$\left(x_{ij}^{kt}-1\right)M+a_i+l_{ij}\le a_j \quad \begin{matrix}\forall (i,j)\in C,\ j\ne i;\\ \forall k\in K;\quad \forall t\in T\end{matrix} \tag{14}$$

$$\theta_d^O+\sum_{\substack{p\in K\\ p\le k}}\sum_{\substack{(i,j)\in V\\ i\ne j}}x_{ij}^{pt}l_{ij}+(w_h^{kt}-1)M\le z_h^{kt} \quad \begin{matrix}\forall k\in K;\quad \forall t\in T;\\ \forall h\in H\end{matrix} \tag{15}$$

$$z_h^{kt} \le \sum_{r \in R_h} Q_{hr} y_{ihr}^{kt} + \left(1 - \sum_{r \in R_h} y_{ihr}^{kt}\right) M \qquad \begin{matrix} \forall i \in C; \quad \forall k \in K; \\ \forall h \in H; \quad \forall t \in T \end{matrix} \tag{16}$$

$$\sum_{h \in H} \sum_{r \in R_h} y_{ihr}^{kt} \le \sum_{\substack{p \in K \\ p \le k}} s_i^{pt} \quad \forall i \in C; \quad \forall k \in K; \quad \forall t \in T \tag{10}$$

$$\sum_{h \in H} \sum_{t \in T} \sum_{k \in K} \sum_{r \in R_h} y_{ihr}^{kt} = 1 \quad \forall i \in C \tag{11}$$

This model is a mixed integer linear program that can be solved using conventional methods for integer programming problems. We can improve the efficiency of the formulation further by lifting the constraints. As presented by Desrochers and Laporte (1991), the linearized form of constraints (14) can be lifted as follows by taking the reverse arc (*j, i*) into account:

$$x_{ji}^{kt}\left(M - l_{ij} - l_{ji}\right) + \left(x_{ij}^{kt} - 1\right) M + a_i + l_{ij} \le a_j \quad \forall (i, j) \in C \setminus D, i \ne j \quad \forall k \in K \quad \forall t \in T \tag{17}$$

In the next section, we demonstrate the solution method for this model using ILOG CPLEX platform.

1.3 Solution Method

The model presented in the previous section is a mixed integer linear programming (MILP) model. In the field of integer programming and combinatorial optimization, various solution algorithms are available for tackling MILP models. We decided to use an efficient Branch-and-Cut (B&C) algorithm to solve our model. The method solves the linear program without the integer constraint using the regular simplex algorithm. When an optimal solution is obtained, and this solution has a non-integer value for a variable that is supposed to be an integer, a cutting plane algorithm is used.

The cutting plane method for MILP works by solving a non-integer linear program, which is the linear relaxation of the given integer program. The obtained optimum is tested for being an integer solution. If it is not, there is guaranteed to exist a linear inequality that separates the optimum from the convex hull of the true feasible set. Finding such an inequality is the separation problem, and such an inequality is a cut. A cut can be added to the relaxed linear program to cut off the current non-integer solution.

At this point, the branch and bound part of the B&C algorithm starts. The problem is split into two versions, one with the additional constraint that the variable is greater than or equal to the next integer greater than the intermediate result, and one where this variable is less than or equal to the next lesser integer. In this way new variables are introduced in the basis according to the number of basic variables that are non-integers in the intermediate solution but which are integers according to the original constraints. The new linear programs are then solved using the simplex

method and the process repeats until a solution satisfying all the integer constraints is found. During the branch and bound process, further cutting planes can be separated, which may be either global cuts, i.e., valid for all feasible integer solutions, or local cuts, meaning that they are satisfied by all solutions fulfilling the side constraints from the currently considered branch and bound sub-tree.

If such an inequality is found, it is added to the linear program, such that resolving it will yield a different solution which is hopefully *less fractional*. This process is repeated until either an integer solution is found, which is then known to be optimal, or until no more cutting planes are found.
To implement the solution algorithm, we developed a Matlab program that acts as our main platform and manages the data files, runs solution enquiries to ILOG CPLEX and processes the solutions. As for the solution engine, we used the ILOG CPLEX platform version 5.5. ILOG engines are commonly used to solve linear and IP problems. This engine is equipped with advanced sub-algorithms to solve models efficiently and quickly. As far as our B&C algorithm is concerned, ILOG is equipped with clique cuts, cover cuts, implied bound cuts, flow cuts, zero-half cuts, and Gomory fractional cuts, which are used to cut non-integer solutions.

Briefly, a *clique* is a relationship among a group of binary variables such that at most one variable in the group can be positive in any integer feasible solution. Before optimization starts, ILOG CPLEX constructs a graph representing these relationships and finds maximal cliques in the graph. On the other hand, if a constraint takes the form of a knapsack constraint (that is, a sum of binary variables with nonnegative coefficients less than or equal to a nonnegative right-hand side), then there is a minimal cover associated with the constraint. A minimal cover is a subset of the variables of the inequality such that if all the subset variables were set to one, the knapsack constraint would be violated, but if any one subset variable were excluded, the constraint would be satisfied. ILOG CPLEX can generate a constraint corresponding to this condition, and this cut is called a *cover cut*. As for *implied bound cuts*, in some models, binary variables imply bounds on continuous variables. ILOG CPLEX generates potential cuts to reflect these relationships. *Flow covers* are generated from constraints that contain continuous variables, where the continuous variables have variable upper bounds that are zero or positive depending on the setting of associated binary variables. The idea of a flow cover comes from considering the constraint containing the continuous variables as defining a single node in a network where the continuous variables are in-flows and out-flows. The flows will be on or off depending on the settings of the associated binary variables for the variable upper bounds. The flows and the demand at the single node imply a knapsack constraint. That knapsack constraint is then used to generate a cover cut on the flows (that is, on the continuous variables and their variable upper bounds). *Zero-half cuts* are based on the observation that when the left-hand side of an inequality consists of integral variables and integral coefficients, then the right-hand side can be rounded down to produce a zero-half cut. Finally, *Gomory fractional cuts* are generated by applying integer rounding on a pivot row in the optimal LP tableau for a (basic) integer variable with a fractional solution value.

To be able to use ILOG CPLEX, we developed a program based on ILOG CPLEX scripting language that performs data processing using the data files generated by the Matlab codes. The results are then saved on the disk for Matlab to retrieve and process them.

1.4 Experimental Study

In this section, we study the effect of different parameters by performing sensitivity analysis using simulation. In our experimental study, we analyze the effect of the number of trucks available, the number of alternative access airports, and the number of customers. For this purpose, we generate a set of problem instances and solve each problem instance using the algorithm described in the preceding section. The results are then recorded and compared for assessing the effect of the number of trucks, alternative access airports and customers. The model developed in the preceding section is suitable for testing the effect of the number of trucks since the model formulation does not enforce utilization of all the available trucks. In contrast, we need to introduce additional constraints to be able to impose limitations on the number of airports chosen. Accordingly, we add the following constraints to the model:

$$\sum_{h \in H} \eta_h \leq N \tag{18}$$

$$w_h^{kt} \leq \eta_h \quad \forall h \in H; \quad \forall k \in K; \quad \forall t \in T \tag{19}$$

In these constraints, η_h indicates whether the airport h is used or not; N is the total number of airports that are allowed to be used.

In this section, we first describe the data generation process. Next, we present and discuss the sensitivity analysis results.

1.4.1 Data generation

For our experimental study, we first define a region based on the number of customers and then randomly generate customer nodes within this region. By relating the number of customers to the size of the region, we are able to control the density of customer nodes throughout the region. The locations of nodes (e.g. a depot, customers and airports) are generated randomly and independently based on uniform distribution over the region. The customer order sizes are randomly selected between 1 and 10 based on uniform distribution. The travel time between each pair of nodes is calculated based on Manhattan distance[2] (i.e. City Block Distance). Compared to

[2] The Manhattan distance or City Block Distance computes the distance that would be traveled to get from one point to the other if a grid-like path is followed. The Manhattan distance between two items is the sum of the differences of their corresponding components.

the Euclidean distance, the Manhattan distance approach better represents the travel time by going through city blocks (see figure 3). Moreover, this approach conserves the distance triangularity property that is needed for transportation problems. This implies that the direct travel time between two nodes is always equal to or less than the travel time of an indirect path; in other words $l_{ij} \leq l_{ik} + l_{kj}$.

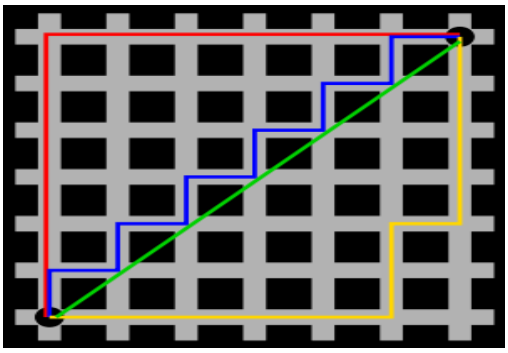


Figure 3. Euclidean distance (green) vs. Manhattan or City Block distance (blue, yellow or red)

In any feasible solution to this problem, each truck completes its trip at the depot by returning from an airport. Accordingly, to ensure the existence of a feasible solution, each airport should have a flight with a scheduled departure time late enough for shipping out the customers' cargo picked up in the truck's last trip. A simple way to achieve this is to construct a single trip that visits all the customers in an arbitrary sequence and then deliver the cargo to the airport. The truck arrival time in this solution can then be used as the departure time for the last flight at that airport. Let Q^{max} be this departure time. Accordingly, the flight departure time is randomly and uniformly generated in $[Q^{min}, Q^{max}]$, where Q^{min} is calculated as follows.

$$Q^{min} = Q^{max}/\rho \qquad (20)$$

In this formula, $\rho \geq 1$ represents the tightness of the departure times. Increasing the value of ρ advances the mean of the flight departure times and thus makes the flights more difficult to get on. Our empirical studies show that the reasonable value for ρ lies in [2,4]. In our experimental study, we use the Next-Best Sequence[3] for the Q^{max} calculation with $\rho = 3$. The flight departure times are always sorted in increasing order such that the final flight departs latest.

The cost-to-go of every flight for every customer is generated randomly and uniformly between [100, 300]. The cost-to-go of flights is then sorted in increasing order as in the case of flight departure time. Therefore, the first flight at an airport departs first and has the cheapest cost-to-go (e.g. most desirable). Moreover, the latest flight at each airport acts as the safety flight to ensure the existence of a feasible solution. The cost-to-go of this flight serves as an upper bound of all flight costs outgoing from the same airport. We generated 4 flights per airport in our experiments.

[3] In Next-Best Sequence, the trip starts at the depot and at each step the closest node is visited next.

In the experimental study, we used three problem sizes: small, medium, and large with 7, 15 and 20 customers, respectively. For each problem size, we generated 100 problem instances and solved them with different a number of trucks and number of alternative access airports.

1.4.2 Results of the Experiments

In our problem setting, we included at most three airports since using more than three airports in an alternative access airport approach is rare (Hall 2004). As for the number of available trucks, we allow at most 4 trucks. The results of our simulations are presented in Table 1. In this table, for each class of the problem (e.g. small, medium and large), we regard the one truck and one airport combination as our baseline and measure the performance of the other combinations in comparison with this baseline scenario. We report on the average performances on the solution of 100 problem instances generated for each combination of customer set, airport set, and number of trucks.

Table 1. Sensitivity results for different problem scenarios. Results present the optimal total shipment cost of a particular scenario as a percentage of the baseline scenario (for the same number of customers)

4 Flights per Airport		**1 truck**	**2 trucks**	**3 trucks**	**4 trucks**
7 customers	**1 airport**	100 %	92.1%	91.3%	91.0%
	2 airports	93.5%	90.7%	90.2%	90.2%
	3 airports	93.2%	88.7%	88.3%	88.1%
15 customers	**1 airport**	100 %	87.8%	85.2%	84.8%
	2 airports	85.9%	81.3%	79.0%	75.3%
	3 airports	82.6%	77.7%	73.5%	72.2%
20 customers	**1 airport**	100 %	85.5%	81.8%	80.9%
	2 airports	79.0%	72.4%	70.6%	64.5%
	3 airports	73.5%	67.2%	63.7%	62.5%

In small size problems (e.g. 7 customers) with only one airport, an increase in the number of available trucks at first improves the objective function. However, further increase in the number of trucks has no significant affect and the objective function remains almost constant. When two airports are allowed to be used, we observe a significant improvement in the objective function indicating the benefits of having alternative access airports. The benefit of having alternative access airports increases as the number of trucks increase (from one truck to two trucks). However, similar to the pattern observed in the single airport case, increasing the number of trucks soon becomes ineffective. Finally, in the case of three airports, we still observe improvement in comparison with the two airport case. However, this improvement is less significant compared to increasing the number of airports from one to two. In other words, although the alternative access

airports can notably improve the objective function, increasing the number of airports beyond two airports provides diminishing improvements in the objective function.

The results of the medium sized problems (e.g. 15 customers) show a similar pattern as in the case of the small sized problem. However, the superiority of having at least one alternative access airport option is more pronounced. As can be observed in the results, with a single truck, increasing the number of airport choices from one to two improves the objective function more than in the case of the small sized problem. This can be explained by the fact that in medium size problems, due to the greater number of customers, there is more opportunity to benefit from the extended choice of flights. The effect of increasing the number of trucks for a given number of airport choices diminishes slower than in the small sized problem. For instance, with two airport choices and two trucks, increasing the number of trucks still improves the objective function.

With large sized problems (e.g., 20 customers), we observe similar patterns of the objective-function responses to different parameter combinations (e.g. number of trucks and airports). However, these effects are slightly more pronounced compared to the medium and small sized problems. The main reason is that a large number of customers provide a better chance of exploiting the additional flight options as the number of alternative access airports is increased.

As described above, for each of the 100 problems instances, we generated the customer node locations randomly and independently over the region. After studying the individual problem instances, we noticed a relationship between the distribution of the customer nodes and the effect of alternative access airports and the number of trucks. In particular, we observed that when customers are clustered (e.g. close to each other), the effect of increasing the number of trucks and the number of airport choices (beyond 2) become less significant. To further study this observation, we designed another simulation experiment. In this experiment, we use a medium sized problem (with 15 customers) and generated all the parameters as before except for the customer locations. The locations of depot and airports are randomly and independently generated as before. We generated customers in three clusters with five customers in each cluster. We first randomly generate three locations as the cluster centers. Then, for each cluster we generate five customers based on the following expression.

$$\begin{cases} x_i = x_c + \gamma . \varphi(0,1) \\ y_i = y_c + \gamma . \varphi(0,1) \end{cases} \quad (21)$$

where $\varphi(0,1)$ is a standard normal distribution and γ is a cluster expansion factor; $[x_c, y_c]$ and $[x_i, y_i]$ denote the coordinates of the cluster c center and customer i location, respectively. The travel times are calculated based on the Manhattan distance as before. Figure 4 illustrates the distribution of the customer locations with the clustering based method (on the left) and with that of the previous method based on uniform sampling (on the right). Clearly, in the cluster-based

method, the customers are more likely to be close to each other and dispersed around respective cluster centers.

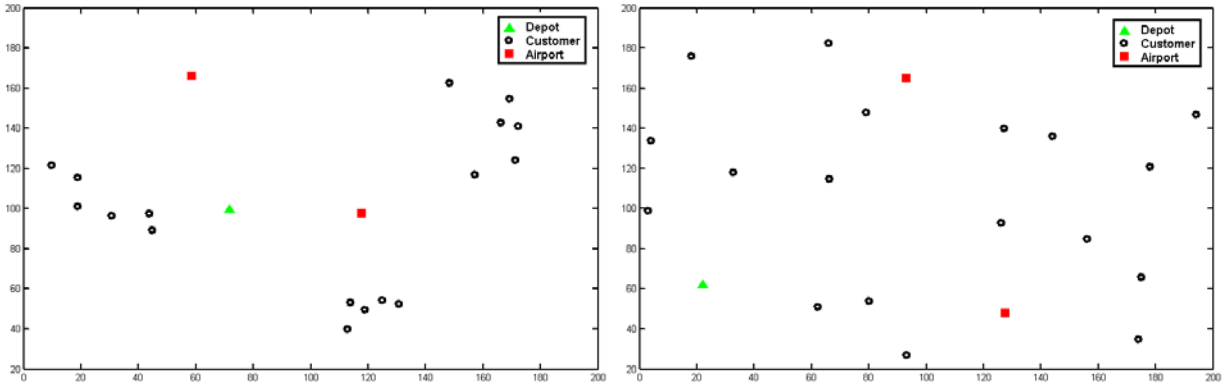


Figure 4. Typical problem instance for clustered (left) and random customers' locations

Next, we study the effect of parameter combinations on the objective function. The results are presented in Table 2. As before, we regard the one truck and one airport combination as our baseline and measure the performance of the other combinations in comparison with this baseline scenario. We report on the average performances on the solution of 100 problem instances generated for each combination of the airport set and number of trucks.

Table 2. Sensitivity results for different problem scenarios. Results present the optimal total shipment cost of a particular scenario as a percentage of the baseline scenario (in the same customer clustering category)

Medium Size Problem		**1 truck**	**2 trucks**	**3 trucks**	**4 trucks**
Non-clustered customers	**1 airport**	100 %	88.9%	86.4%	85.7%
	2 airports	86.3%	82.5%	81.1%	74.6%
	3 airports	82.3%	78.0%	73.5%	70.2%
Clustered customers	**1 airport**	100 %	87.2%	85.9%	85.1%
	2 airports	87.9%	83.0%	81.8%	80.3%
	3 airports	85.1%	79.9%	72.7%	71.9%

As can be observed in Table 2, the effect of increasing the number of trucks and airport alternatives is similar to that shown in Table 1. While an increase in the number of trucks improves the objective function, this effect is less notable in the clustered problem. This is attributable to the fact that the increased proximity of customers in the clustered scenario reduces the effect of additional trucks since, in most cases, a single truck can service an entire cluster. In contrast, in the problems with non-clustered customers, increasing the number of trucks can be beneficial for customers located far away from other customers.

The effect of increasing the number of alternative access airports improves the performance in both the clustered and non-clustered customers cases. However the significance is different in these two cases. In the clustered customer problems, increasing the number of airports has lesser impact on the objective function improvement compared to the non-clustered customer problems. This can be explained by the fact that the proximity of the clustered customers reduces the airport delivery times and thus customers are not able to benefit from the extended choice of flights.

1.4.3 Experimental Study Conclusion

In this experimental study, we setup a series of controlled experiments to investigate the effect of problem parameters on the solution performance. We studied the influence of the number of trucks and the number of alternative access airports. We also carried out a separate experiment to study the effect of customer clustering on the performance. It was observed that the existence of alternative access airports is always beneficial. This effect further increases with the number of trucks and is more significant for larger problems with more customers. This observation can be used to estimate the optimum number of trucks for a given scenario.

Moreover, our experimentation with clustered customers shows that the effect of parameters is reduced with the clustering. Particularly, the effect of the number of trucks becomes less significant, especially when it exceeds the number of clusters. This observation confirms the common practice of integrated carriers (e.g. UPS) in dividing a market area into different regions and assigning exclusive trucks to service the customers in each region.

1.5 Case study

In this section, we present the case study application of the proposed model in a real world scenario. To study the regional benefit of our proposed model, we generated a case study over the southeast Michigan and northwest Ohio region. The case study problem is concerned with the collection of customers' air cargo in the region and delivering them to the local airports for air-shipment. We use the most demanded airports in this region for freight transportation: Detroit Metropolitan Wayne County airport (DTW) and Toledo Express airport (TOL). We performed a similar case study in our previous work that mainly focuses on the air-network routing. Our main goal in this case study, however, is to study the road-network routing and road-air intermodal connectivity. In what follows, we first discuss the data gathering approach and then present the results including the results of a sensitivity analysis.

1.5.1 Case Study Data

In this case study, the main goal is to analyze the routing on the road network for picking up orders and delivering them to the airports. We use the DTW and the TOL airports. While the DTW is a mixed freight-passenger airport, the TOL is mostly a freight shipping airport. Since passenger flights are legally obligated to report their operational details, comprehensive

operational information is available through the BTS database for passenger flights at the DTW. On the other hand, at TOL, most of the freight flights are privately operated or chartered and are not obligated to report their operational details. Accordingly, although we have very detailed information on DTW flights, the TOL flight information is not as complete. Further, an analysis based on the historical flight information (e.g. number of departing flights, flight connection and destinations, departure schedule) would not serve well for the case study's purpose of evaluating alternative access airports for dynamic inter-modal routing with milk run pickups and deliveries. Since the air-network routing is not the focus of this analysis, we use randomly generated flight data (scheduled departure time and cost-to-go) for both airports. We generate the same number of flights at each airport. These flights are randomly and independently generated.

We used the Google API to generate the geographical locations for customer originations of the air cargo. In this approach, our computer program first generates a random geographical coordination (latitude & longitude) on the case study region (southeast Michigan and northwest Ohio). Next, we sent an inquiry to Google map using Google API protocol to verify that this location is a valid address and accessible through the road network. If the location is a valid address, it is accepted; otherwise, it is rejected and replaced by a another randomly generated location. This approach excludes all unrealistic locations including, but not limited to, empty fields, out of road locations, lakes and parks. One outcome of this approach is that locations with higher road connectivity (e.g. cities) have more chance of being selected as customer locations.

Another key element of the case study is the location of the depot. Since customer locations are generated randomly and independently of the location of airports and the depot, the proximity of the depot to any of the airports does not necessarily imply that the closest airport to the depot will be chosen for all air cargo shipments. However, it can favor one airport over the other rather indirectly. Accordingly, we consider the location of the depot as one of the parameters in this case study and study its effect on the performance. In practice, freight forwarders are more likely to locate their deports close to the airports. Therefore, we define three scenarios for depot locations: in the vicinity of DTW, in the vicinity of TOL, and in between TOL and DTW (around Monroe, Michigan).

The next step in the data generation for the case study is to calculate the travel time for each pair of nodes (customers, airports, and the depot). We used the Google API to calculate the travel time between each pair of nodes based on the actual road network travel time. In this approach, we send an enquiry to the Google API for the travel time between a pair of locations. Then, the Google API finds the path with the shortest travel time between the requested locations using the real road network considering the speed limits and report back the estimated travel time with turn-by-turn direction (see figure 5). It should be noted that travel times calculated in this approach are not necessary symmetric since the same routes may not be used in a reverse trip. However, since the

Google API always returns the shortest path, the triangularity property is still valid for the calculated travel times ($l_{ij} \leq l_{ik} + l_{kj}$).

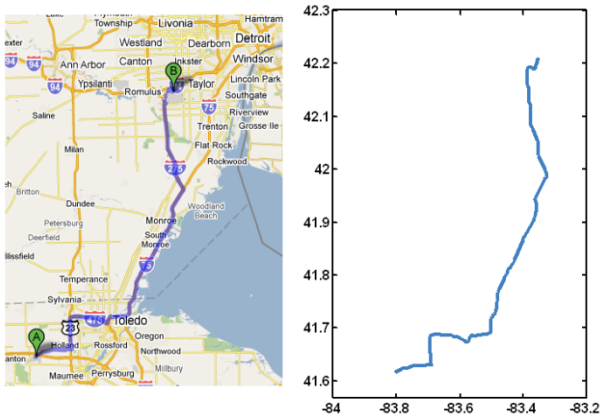


Figure 5. Google API travel time and turn-by-turn direction demonstration; snap shot from original Google map (left) and graph generated by our Matlab code for direction inquiry from the Detroit airport to the Toledo airport; estimated travel time is 72 minutes

In the case study, we consider 20 customers and generated 300 geographical distribution instances for these customers.

1.5.2 Case Study Results

In our case study, we study the effect of the number of trucks and having an alternative access airport on the performance. We consider up to 3 trucks. The results of our simulation are presented in Tables 3 and 4. In Table 3, we present the effect of the parameters on the objective function. In this table, for each depot location scenario, we consider one truck and one airport combination as our baseline and then measure the performance of other combinations relative to this baseline scenario. Table 4 presents the percent of the time the TOL airport is selected over the DTW airport when only one airport choice is allowed.

Table 3. Sensitivity results for different problem scenarios. Results present the optimal total shipment cost of a particular scenario as a percentage of the baseline scenario (for the same depot location)

Depot Location	***4 Flights per Airport***	**1 truck**	**2 trucks**	**3 trucks**
DTW	**1 airport**	100%	85.4%	84.8%
	2 airports	89.7%	82.1%	81.0%
Monroe	**1 airport**	100%	82.6%	80.4%
	2 airports	91.2%	78.9%	77.1%
TOL	**1 airport**	100%	83.8%	83.0%
	2 airports	86.1%	80.2%	78.5%

Table 3 results demonstrate that with only one truck the alternative access airport improves the objective function. This effect, however, is not the same for different depot locations. The best results are achieved when the depot is in the vicinity of one of the airports. It appears that the proximity to an airport allows the truck to quickly deliver the customer orders in the vicinity of the depot to the nearby airport and then deliver other orders to the alternative access airport. Moreover, the performance improvement is slightly more if the depot is closer to TOL than to DTW. This can be explained by considering the final distribution of customers. As mentioned in the data generation section, since the DTW airport is closer to population centers, it is more likely to have more customers closer to the DTW airport than the TOL airport. The worst result is achieved when the depot is in Monroe (between the two airports). In this scenario, the freight forwarder cannot fully benefit from the proximity to any of the airports. The observed phenomenon confirm the freight forwarders' practice of establishing their depot close to major airports.

As the number of trucks increase, an interesting change in the performance of depot locations is observed. While the performance improvement profile remains as before for the single airport alternative, the Monroe depot becomes the best option with the alternative access airport. This can be explained by the fact that, when two trucks are available, a depot can potentially send one truck to each airport while splitting the orders among the airports. Clearly, under this scenario, the trucks departing from a depot that is located at a balanced distance from either airport can reach the customers faster and therefore have better performance.

Table 4. Percent of the time the TOL airport is selected over the DTW airport when only one airport choice is allowed

Depot Location	1 truck	2 trucks	3 trucks
DTW	40.8%	40.8%	40.9%
Monroe	43.4%	44.1%	42.5%
TOL	45.7%	45.5%	45.8%

Table 4 presents the choice between two airports when only one is allowed to be used for air-cargo shipment. Results indicate that DTW seems to be a more desirable choice in this particular example. This can be explained by the fact that the majority of the customers are likely to be closer to the DTW airport. Although the DTW airport has a slight advantage over the TOL airport, the overall difference is not significant across different depot locations. After reviewing the solutions to the individual problem instances, we notice two typical solution outcomes with the single airport alternative. These solution outcomes are dependent on the flight departure times and the cost-to-go functions and occur when the depot is located close to one of the airports (Figure 6).

Note that customers are visited according to the numbered ordering shown in Figure 6. These trips do not necessarily correspond to the shortest tour, e.g. customers located close could be apart in the visit sequence. The rationale is that when the customers can be visited and still have their cargo loaded on the desirable flight, then the shortest distance traveled is not a requirement.

One solution outcome is the case when the trucks first pickup orders near the home airport and then head toward the distant airport to pickup distant orders and drop them off them at the distant. The other case is when the trucks perform multiple trips ending at the home airport and then go for a long trip to collect orders closer to the distant airport and then return to the home airport for delivery. These observations, particularly the first solution outcome, are counterintuitive.

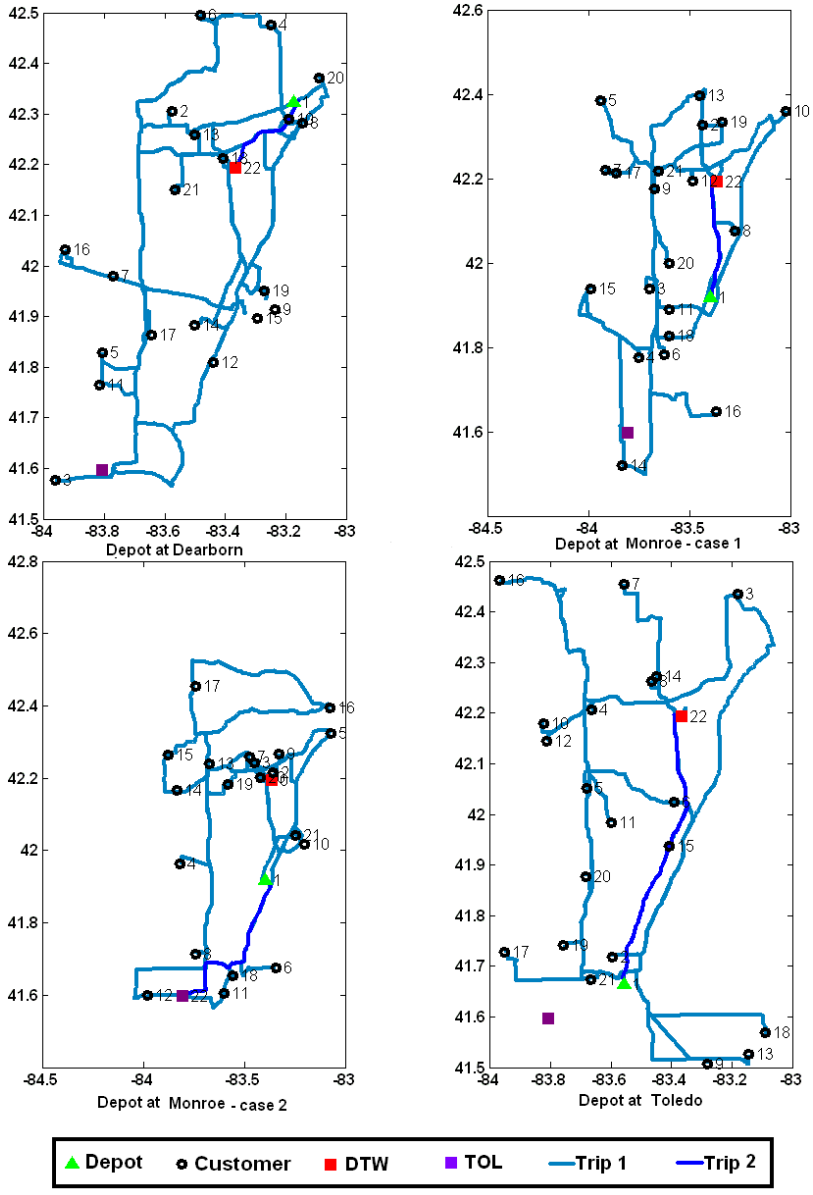


Figure 6. Solution paths for single truck and single choice of airport in different depot location scenarios (customers visited according to the numbered ordering)

In the case of the Monroe depot, there is no specific pattern in the solutions. The trucks are more likely to choose the airport with higher density of customers. This reconfirms the tendency of the freight forwarders to locate their depot close to the airports airport.

1.5.3 Case study Conclusion

In this case study, we consider dynamic intermodal routing on the air-road network with milk runs in the southeast Michigan and northwest Ohio region. We consider two major airports in the region (i.e. DTW & TOL). We study the effect of the number of trucks and airport choices for three different locations of depots on the performance. Results indicate that the alternative access airport is always beneficial. This benefit improves as the number of trucks increase.

Moreover, we also observe that the location of the depot can have a significant effect on the performance of a freight forwarder. Results favor depot locations close to an airport and in the areas with higher customer density.

B. Dynamic Routing in Stochastic Time-Dependent Networks for Milk Run Tours with Time Windows

2.1 Introduction

Freight forwarders often receive their orders in less than truck load and need to pick up these orders from several customers by sending a truck from the depot. In determining which customers to visit in a given tour and the visit sequence in the tour, the most important consideration is the flight schedules at the airport. However, the congestion on road networks causes high variability in travel times (Chen et al.) that makes it difficult to catch flights. For example, in a survey in California 85% of trucking company managers state that they are missing their schedules at some levels because of congestion. Further only 22% of the managers state that the time-windows for pickup and deliveries do not force their drivers to work in congested conditions (Golob and Regan, 2003). Different than the earliness and tardiness in the manufacturing environments which are subject to soft time windows, the earliness and tardiness in scheduled freight flights have *hard* time windows and result in stepwise penalties. In this work, we address the milk run tour of a truck picking up air cargo shipments from a known set of customer locations and delivering them to the airport. We model the problem as a Traveling Salesman Problem (TSP) with hard time windows. In addition we consider the road network congestion and model the network arcs having random travel times conditional on the time-dependent congestion states.

TSP is the problem of finding the least cost tour that visits each site exactly once with a given set of customers and the cost (time/ distance) between each pair of customers. Based on the treatment of arc costs (e.g., deterministic, stochastic time-dependent travel times, etc.), the nature of the network (e.g., dynamic customer arrival), and the presence of time windows (e.g., hard, soft) the

TSP takes different forms. Even the simplest forms of the TSP are known to be NP-hard (there are no solution algorithms with complexity that grow polynomially as a function of the size of the network). In our problem setting we are dealing with TSP problems with time windows under stochastic time-dependent arc travel times, as well as congestion states that require "dynamic" routing policies. Since these settings make the problem impractical to seek "optimal" routing algorithms, we developed effective heuristics to build "robust" TSP tours. The methods from the literature assume a direct arc (or a static path) between sites and/or the availability of travel time distribution. However with time varying structure of the network it might happen that taking other paths requires less travel time. There is also no modeling of congestion (either recurrent or non-recurrent) in TSP literature.

The congestion is classified as recurrent (e.g., experienced during rush hours) and non-recurrent (e.g., occurs during incidents, inclement weather, etc.). In this study we only model recurrent congestion. To find an optimal policy between customer locations under recurrent congestion, we assume the traffic dynamics follows a Markov process. Namely, the state of the next time period depends only on the state of the previous time period. Then recurrent congestion may be modeled based on the Markov decision process (MDP). We assume that the state set of the MDP is based on the position of the vehicle, the time of the day and the congestion states of the arcs. The congestion state classes (i.e.: congested, uncongested, etc.) of the roads are a function of the time of the day and determined with historic traffic data based on a Gaussian Mixture Model (GMM). Since not all network information affects an optimal decision, we assume the arc set of a state such that only the arcs close to the vehicle affect the decision. We also assume that the traffic data for some of the arcs may not be available.

In this work we do not assume a fixed route, but rather employ a stochastic time-dependent (STD) routing policy between depot, airport, customer sites (for consistency we will use "site" to refer to depot, airport, customer, etc.). Since we need travel time distributions between sites to identify the TSP tour, we first derive dynamic routing policies (for each pair of sites and by time of day) and estimate the time-dependent travel time distributions from the derived policy through simulation. Then we identify the robust TSP tours by applying a dynamic programming (DP) for the problem. Since the travel times are STD, we exploit a convolution method from Chang et al. (2009) to estimate the distribution of site arrival times in stochastic time-dependent networks during the execution of DP. The tours are "robust" in the sense that they are generally effective given normal congestion patterns. However, the actual routing of the truck between sites will be based on real-time traffic flow information guided by dynamic routing policies. A key aspect of the proposed approach is the "robust" tour that will trade-off catching earlier flights with the risk of on time delivery to the airport. While we commit to the robust tour, we still dynamically route the truck between sites.

The rest of the section is organized as follows. A survey of literature is given in section 2. In section 3, the modeling of the stochastic time-dependent TSP is described. Section 4 presents the experimental studies to show the robustness of the proposed model. Section 5 concludes the study and suggests some future direction.

2.2 Literature Survey

In the literature this kind of problem is studied as a stochastic time dependent traveling salesman problem with time windows. In the standard version of the TSP, we are given a set of sites and the distances between each pair of sites, and it is asked to find the shortest tour that originates from an origin site, visits each site exactly once, and returns to the origin site. The TSP has been studied for more than 50 years and a wide variety of exact and heuristic algorithms have been developed. Johnson and McGeoch (1997) and Junger et al. (1995) are two of the sources from literature on algorithmic and computational aspects of the TSP.

Different variants of the TSP have been studied in the literature. Malandraki and Dial (1996) presented a DP and a "restricted" DP (that also exploits nearest-neighbor heuristic) approach to solve time-dependent TSP (TD-TSP). They modeled the time dependency in travel time by discrete step functions such that the planning horizon has a number of different time zones and travel times differ only at different time zones. The limitation of using such step functions, i.e. a later depart time might lead to an earlier arrival time where steep speed increases occur, was emphasized later by Ichoua et al. (2003) as being essential in modeling time-dependent travel times. They also proposed alternative models that comply with the FIFO (first in, first out) principle and constructing models to build the TSP tour.

Another variant of the TSP is stochastic TSP where travel times between sites are random. This variant is mostly studied as the vehicle routing problem (e.g. Laporte et al., (1992) and Lambert et al., (1993)) where the TSP is a special vehicle routing problem in which there is one vehicle and the capacity constraints of the vehicle are relaxed. Jula et al. (2006) and Chang et al. (2009) studied the stochastic time-dependent TSP with time windows (STD-TSP-TW). Jula et al. (2006) estimated the first two moments of arrival time of the vehicle at each site based on the first (or second) order Taylor approximation of arrival times. They defined a service level based on arrival time to sites and eliminate those routes which the service levels do not satisfy. They also performed a state elimination test based on expected travel times. These two eliminations were used to reduce the state space of dynamic programming and eventually reduce the computational time. Chang et al. (2009) developed a convolution–propagation approach (CPA) to estimate the mean and variance of arrival times at sites assuming the arc travel times are normally distributed. They proposed a heuristic algorithm that uses the n-path relaxation of deterministic TSP by Houck et al., (1980) to solve the problem.

Although the problem that we are considering is the same as that in Jula et al. (2006) and Chang et al. (2009), the modeling concept is very different. The main difference of our study is the existence of dynamic routing between sites and modeling congestion that takes into account real time traffic information and projects the future traffic. Jula et al. (2006) and Chang et al. (2009) also assume a direct arc between sites and/or the availability of travel time distribution while we allow dynamic routing between sites. To the best of our knowledge, there is no earlier study on the dynamic routing for the stochastic time-dependent TSP problem.

Dynamic routing and modeling real time information is mostly studied under shortest path problems in the literature. Polychronopoulos and Tsitsiklis (1993) is the first study to consider the stochastic temporal dependence of arc costs and to suggest using online information en route. They defined the environmental state of nodes that is learned only when the vehicle arrives at the source node. They considered the state changes according to a Markovian process and employed a DP procedure to determine the optimal policy. Kim et al. (2005a) studied a similar problem as in Psaraftis and Tsitsiklis (1993) except that the information of all of the arcs is available in real-time. They proposed a DP formulation where the state space includes states of all arcs, time, and the current node. They stated that the state space of the proposed formulation becomes quite large making the problem intractable. They reported substantial cost savings from a computational study based on a southeast Michigan road network. To address the intractable state-space issue, Kim et al. (2005b) proposed state space reduction methods. A limitation of Kim et al. (2005a) is the modeling and partitioning of travel speeds for the determination of arc congestion states. They assumed that the joint distribution of velocities from any two consecutive periods follows a single unimodal Gaussian distribution, which cannot adequately represent arc travel velocities for arcs that routinely experience multiple congestion states. Moreover, they also employed a fixed velocity threshold *(50 mph)* for all arcs and for all times in partitioning the Gaussian distribution to estimate state-transition probabilities (i.e., transitions between congested and uncongested states). As a result, the value of real-time information is compromised rendering the loss of performance of the dynamic routing policy.

2.3 Modeling Dynamic Routing for Milk Runs

The dynamic routing model for milk runs requires estimating time-dependent travel time distributions between every pair of sites. These travel time distributions are estimated through the following steps:

- Develop a dynamic routing policy (action set for every state) between every pair of sites and possible departure times.
- Estimate the travel time distribution through simulation for every possible departure time.

Once the travel time distributions are estimated for every pair of sites and possible departure times, we then employ stochastic time dependent dynamic programming (STD- DP) to identify the

most robust tour. We also exploit the convolution approach of Chang et al. (2009) to identify arrival time distributions at sites for partial tours during the execution of STD-DP.

Let $G=(N,A)$ be a directed graph in which N is the set of nodes and $A \subseteq N \times N$ is the set of directed arcs. The (decision) node $n \in N$ represents an intersection where the driver can decide which arc to take next. A directed arc is represented by an ordered pair of nodes $(n,n') \in A$ in which n is called the origin and n' is called the destination of the arc. Let $M \subseteq N$ be the set of (e.g. depot, supplier, or customer) sites. Note that sites are also decision nodes, while the opposite statement needs not to be true. Assume 0 is the origin site (depot) and there are m-1 sites to be visited, represented by nodes $1,\ldots,m-1$ which represent customer/supplier sites. For the time being, let's assume that the vehicle at site j will visit site k next, where $j,k \in M$ and assume that there is an optimal policy, π_{jk}, between every sites (j,k). Let $\delta_{j,k}(t_j)$ and $Var\left(\delta_{j,k}(t_j)\right)$ be the expected travel time and variance when following the optimal policy.

Section 3.1 explains how to find optimal policy π_{jk} to estimate $\delta_{j,k}(t_j)$ and $Var\left(\delta_{j,k}(t_j)\right)$. And section 3.2 describes building milk run tours with using these parameters by STD-DP algorithm.

2.3.1 Estimation of travel times between sites

A directed arc $(n,n') \in A$ is labeled as observed if its real-time traffic data (e.g., velocity) is available through the intelligent transportation system. An observed arc can be in $r+1 \in Z^+$ different states that represent the arc's traffic congestion level at a given time. Let $s_a(t)$ be the congestion state of arc a at time t, i.e. $s_a(t)=\{\text{Congested at level } i\}=\{i\}$ for $i=1,2,\ldots,r+1$ and be determined as follows:

$$s_a(t)=\left\{i, \text{if } c_a^{i-1}(t) \le v_a(t) < c_a^i(t)\right\} \quad (1)$$

where $c_a(t)$ denotes the cut-off velocity. For instance, if there are two congestion levels (e.g., $r+1=2$), then the states will be $s_a(t)=\{\text{Uncongested}\}=\{0\}$ and $s_a(t)=\{\text{Congested}\}=\{1\}$.

We assume that the state of an arc evolves according to a non-stationary Markov chain. In a network with all arcs observed, $S(t)$ denotes the traffic congestion state vector for the entire network, i.e., $S(t)=\left\{s_1(t),s_2(t),\ldots,s_{|A|}(t)\right\}$ at time t. For presentation clarity, we will suppress (t) in the notation whenever time reference is obvious from the expression. Let the state realization of $S(t)$ be denoted by $s(t)$. It is assumed that arc states are independent from each other and have the single-stage Markovian property. In order to estimate the state transitions for each arc, the velocities of two consecutive periods are modeled jointly. Accordingly, the time-dependent single-

period state transition probability from state $s_a(t)=i$ to state $s_a(t+1)=j$ is denoted by $P\{s_a(t+1)=j \mid s_a(t)=i\}=\alpha_a^{ij}(t)$. The transition probability for arc a, $\alpha_a^{ij}(t)$ is estimated from the joint velocity distribution as follows:

$$\alpha_a^{ij}(t)=\frac{\left|c_a^{i-1}(t)\le V_a(t)<c_a^i(t)\cap c_a^{j-1}(t+1)<V_a(t+1)<c_a^j(t+1)\right|}{\left|c_a^{i-1}(t)\le V_a(t)<c_a^i(t)\right|} \quad (2)$$

Let $TP_a(t,t+1)$ denote the matrix of state transition probabilities from time t to time $t+1$ then we have $TP_a(t,t+1)=\left[\alpha_a^{ij}(t)\right]_{ij}$. Note that the single-stage Markovian assumption is not restrictive in our approach as we could extend our methods to the multi-stage case by expanding the state space (Bertsekas, 2001). Let the network be in state $S(t)$ at time t, and we want to find the probability of the network state $S(t+\delta)$, where δ is a positive integer number. Given the independence assumption of the arcs' congestion states, this can be formulated as follows:

$$P\left(S(t+\delta)\mid S(t)\right)=\prod_{a=1}^{|A|}P\left(s_a(t+\delta)\mid s_a(t)\right) \quad (3)$$

Then the congestion state transition probability matrix for each arc in δ periods can be found by the Kolmogorov's equation:

$$TP_a(t,t+\delta)=\left[\alpha_a^{ij}(t)\right]_{ij}\times\left[\alpha_a^{ij}(t+1)\right]_{ij}\times\ldots\times\left[\alpha_a^{ij}(t+\delta)\right]_{ij} \quad (4)$$

With the normal distribution assumption of velocities, the time to travel on an arc can be modeled as a non-stationary normal distribution. We further assume that the arc's travel time depends on the congestion state of the arc at the time of departure (equivalent to the arrival time whenever there is no waiting). It can be determined according to the corresponding normal distribution:

$$\delta(t,a,s_a)\sim N\left(\mu(t,a,s_a),\sigma^2(t,a,s_a)\right) \quad (5)$$

where $\delta(t,a,s_a)$ is the travel time; $\mu(t,a,s_a)$ and $\sigma(t,a,s_a)$ are the mean and the standard deviation of the travel time on arc a at time t with congestion state $s_a(t)$.

2.3.2 Dynamic routing model with recurrent congestion

Assume that the objective of the dynamic routing model is to minimize the expected travel time based on real-time information such as the trip originates at node n_0 and ends at node n_d. Let's assume that there is a feasible path between (n_0,n_d) where a path $p=(n_0,..,n_k,..,n_{K-1})$ is defined as the sequence of (decision) nodes such that $a_k\equiv(n_k,n_{k+1})\in A,\ k=0,..,K-1$ and K is the number of nodes on the path. Note that while all nodes are decision nodes, only origin and destination nodes are (depot, airport, or customer) sites for the problem.

We define set $a_k \equiv (n_k, n_{k+1}) \in A$ as the current arcs set of node n_k, denoted with $CrAS(n_k)$. That is, $CrAS(n_k) \equiv \{a_k : a_k \equiv (n_k, n_{k+1}) \in A\}$ is the set of arcs emanating from node n_k. Each node on a path is a decision stage (or epoch) at which a routing decision (which node to select next) is to be made. Let $n_k \in N$ be the location of k^{th} decision stage, t_k is the time at k^{th} decision stage where $t_k \in \{1,...,T\}$ $T > t_{K-1}$. T is an arbitrarily large number and is used to limit the planning horizon for modeling purposes. Note that we are divide the planning horizon into uniformly spaced discrete time epochs.

While the optimal dynamic routing policy requires real-time consideration and projection of the traffic states of the complete network, this approach makes the state space prohibitively large. In fact, there is little value in projecting the congestion states well ahead of the current location. This is because the projected information is not different from the long run average steady state probabilities of the arc congestion states. Hence, an efficient but practical approach would trade off the degree of look-ahead (e.g., the number of arcs to monitor) with the resulting projection accuracy and routing performance. This has been very well illustrated in Kim et al. (2005b). Thus, we limit our look-ahead to a finite number of arcs that can vary by the vehicle location on the network. The selection of the arcs to monitor would depend on factors such as arc lengths, the value of real-time information, and the congestion state transition characteristics of the arcs. For ease of presentation and without loss of generality, we choose to monitor only two arcs ahead of the vehicle location and model the rest of the arcs' congestion states through their steady state probabilities. Accordingly, we define the following two sets for all of the arcs in the network. $ScAS(a_k)$, the successor arc set of arcs a_k, $ScAS(a_k) \equiv \{a_{k+1} : a_{k+1} \equiv (n_{k+1}, n_{k+2}) \in A\}$, i.e., the set of outgoing arcs from the destination node (n_{k+1}) of arc a_k. $PScAS(a_k)$, the post-successor arc set of arc a_k, $PScAS(a_k) \equiv \{a_{k+2} : a_{k+2} \equiv (n_{k+2}, n_{k+3}) \in A\}$ i.e., the set of outgoing arcs from the destination nodes (n_{k+2}) of arcs a_{k+1}.

Since the total trip travel time is an additive function of the individual arc travel times on the path plus a penalty function measuring earliness/tardiness of arrival time to the destination node, the dynamic route selection problem can be modeled as a dynamic programming model. The state $\left(n_k, t_k, s_{a_{k+1} \cup a_{k+2}, k}\right)$ of the system at k^{th} decision stage is denoted by Ω_k. This state vector is composed of the state of the vehicle and network and thus is characterized by the current node (n_k), the current node arrival time (t_k), and $s_{a_{k+1} \cup a_{k+2}, k}$, the congestion state of arcs $a_{k+1} \cup a_{k+2}$ where $\{a_{k+1} : a_{k+1} \in ScAS(a_k)\}$ and $\{a_{k+2} : a_{k+2} \in PScAS(a_k)\}$ at k^{th} decision stage.

The action space for the state Ω_k is the set of current arcs of node n_k, $CrAS(n_k)$. At every decision stage, the trip planner evaluates the alternative arcs based on the remaining expected

travel time. The expected travel time at a given node with the selection of an outgoing arc is the summation of expected arc travel time on the arc chosen and the expected travel time of the next node. Let $\boldsymbol{\pi}_{n_0 n_d} = \{\pi_0, \pi_1, ..., \pi_{K-1}\}$ be the policy of the trip that is composed of policies for each of the K-1 decision stages. For a given state $\Omega_k = \left(n_k, t_k, s_{a_{k+1} \cup a_{k+2}, k}\right)$, the policy $\pi_k(\Omega_k)$ is a deterministic Markov policy which chooses the outgoing arc from node n_k, i.e., $\pi_k(\Omega_k) = a \in CrAS(n_k)$. Therefore, the expected travel cost for a given policy vector π is as follows:

$$F^{\pi}(\Omega_0) = \underset{\delta_k}{E}\left\{\sum_{k=0}^{K-2} g\left(\Omega_k, \pi_k(\Omega_k), \delta_k\right) + \bar{g}\left(\Omega_{K-1}\right)\right\} \tag{6}$$

where $\Omega_0 = (n_0, t_0, S_0)$ is the starting state of the system. δ_k is the random travel time at decision stage k, i.e., $\delta_k \equiv \delta\left(t_k, \pi_k(\Omega_k), s_a(t_k)\right)$. $g(\Omega_k, \pi_k(\Omega_k), \delta_k)$ is the cost of travel on arc $\pi_k(\Omega_k) = a \in CrAS(n_k)$ at stage k, i.e., if travel cost is a function (ϕ) of the travel time, then $g(\Omega_k, \pi_k(\Omega_k), \delta_k) \equiv \phi(\delta_k)$ and $\bar{g}(\Omega_{K-1})$ is the terminal cost of earliness/tardiness of the arrival time to the destination node under state Ω_{K-1}. Then, the minimum expected travel time can be found by minimizing $F(\Omega_0)$ over the policy vector π as follows:

$$F^*(\Omega_0) = \min_{\boldsymbol{\pi}_{n_0 n_d} = \{\pi_0, \pi_1, ..., \pi_{K-1}\}} F(\Omega_0) \tag{7}$$

The corresponding optimal policy is then:

$$\boldsymbol{\pi}^*_{n_0 n_d} = \arg\min_{\boldsymbol{\pi}_{n_0 n_d} = \{\pi_0, \pi_1, ..., \pi_{K-1}\}} F(\Omega_0) \tag{8}$$

Hence, the Bellman's cost-to-go equation for the dynamic programming model can be expressed as follows (Bertsekas, 2001):

$$F^*(\Omega_k) = \min_{\pi_k} \underset{\delta_k}{E}\left\{g(\Omega_k, \pi_k(\Omega_k), \delta_k) + F^*(\Omega_{k+1})\right\} \tag{9}$$

For a given policy $\pi_k(\Omega_k)$, we can re-express the cost-to-go function by writing the expectation in the following explicit form:

$$\begin{aligned} F(\Omega_k \mid a_k) = \sum_{\delta_k} P(\delta_k \mid \Omega_k, a_k)\Big[g(\Omega_k, a_k, \delta_k) + \\ \sum_{s_{a_{k+1}, k+1}} P\left(s_{a_{k+1}, k+1}(t_{k+1}) \mid s_{a_{k+1}, k}(t_k)\right) \sum_{s_{a_{k+2}, k+1}} P\left(s_{a_{k+2}, k+1}(t_{k+1})\right) F(\Omega_{k+1})\Big] \end{aligned} \tag{10}$$

where $P(\delta_k \mid \Omega_k, a_k)$ is the probability of travelling arc a_k in δ_k periods. $P\left(s_{a_{k+2}, k+1}(t_{k+1})\right)$ is the long run probability of arc $a_{k+2} : a_{k+2} \in PScAS(a_k)$ being in state $s_{a_{k+2}, k+1}$ in stage $k+1$. This probability can be calculated from the historical frequency of a state for a given arc and time.

We use the backward dynamic programming algorithm to solve $F^*(\Omega_k)$, $k = K-1, K-2, .., 0$. In the backward induction, we initialize the final decision epoch such that, $\Omega_{K-1} = (n_{K-1}, t_{K-1}, s_{K-1})$, n_{K-1} is the destination node, and $F(\Omega_{K-1}) = 0$ if $t_{K-1} \leq T$. Accordingly, a penalty cost is accrued whenever there is delivery tardiness, e.g., $t_{K-1} > T$. Note that $s_{K-1} = \varnothing$ since the destination node's current and successor arcs states congestion state information.

2.3.3 Estimating travel time distributions between sites

For the time being, let's assume that the vehicle at site j will visit site k next, where $j,k \in M$ and assume that there is an optimal policy, $\boldsymbol{\pi}_{jk}$, between every sites (j,k). Simulating the optimal policy $\boldsymbol{\pi}_{jk}$ at departure time, t_j yields the expected travel time and variance denoted with $\delta_{j,k}(t_j)$ and $Var\left(\delta_{j,k}(t_j)\right)$, respectively.

2.4 STD-DP algorithm for STD-TSP tours

We describe a DP algorithm to find a robust STD-TSP tour for a given starting time in this section. The algorithm is adapted from Malandraki and Dial (1996) and Jula et al. (2006) and uses a convolution approach from Chang et al. (2009):

There are *m*-1 sites (other than the depot, assuming the vehicle is at the depot) to be visited, represented by nodes $1, \ldots, m-1 \in M$. Let $(C,k) \subseteq M / \{0\}$ be an unordered set of sites where $k \in C$ is the last visited site. Define *partial tour* as a tour that starts from the depot, visit all the sites in (C,k) only once and ends the tour at site k. Note that there may be more than one *partial tour* for a state (C,k). Let $T(C,k)$ be the random variable of arrival time at site k taking the *partial tour* (C,k). Let also $E[T(C,k)]$ and $Var(T(C,k))$ be the mean and variance of arrival time $T(C,k)$, respectively.

Initialization step: For all $|(C,k)| = 1$ where $(C,k) = \{k\}, k \in M / \{0\}$, we initialize $E[T(C,k)] = T(0) + s_0 + \delta_{0k}(t_0)$ and $Var(T(C,k)) = Var(\delta_{0k}(t_0))$, where $T(0)$ is the arrival time to the site 0 (depot), s_0 is the service (or preparation) time at the site 0, and $\delta_{0k}(t_0)$ is the expected travel time from site 0 to site k as a function of the departure time , t_0. The departure time of a site (i.e. t_0 for site 0) is the sum of the arrival time to the site and the service time at the site; in this example $t_0 = T(0) + s_0$. Without the loss of generalization we assume the service time at sites are deterministic.

Main step: For all $|(C,k)|>1$ consider visiting k, $k \in M/\{0,j\}$ immediately after j $\left(\text{for all } j \in C/\{k\}\right)$ and look up $E\left[T(C,j)\right]$ and $Var\left(T(C,j)\right)$ from the previous step. So we have from Chang et al. (2009):

$$E\left[T(C,k)\right] = E\left[T(C,j)\right] + s_j + \sum_{t_j=1}^{T} \delta_{jk}\left(t_j\right) * p_{t_j} \tag{11}$$

$$\begin{aligned} Var\left(T(C,k)\right) &= Var\left(T(C,j)\right) + \sum_{t_j=1}^{T} p_{t_j} * \sigma_{t_j}^2 + \sum_{t_j=1}^{T} p_{t_j} * \delta_{jk}\left(t_j\right)^2 \\ &- \left[\sum_{t_j=1}^{T} p_{t_j} * \delta_{jk}\left(t_j\right)\right]^2 - 2\sum_{t_j=1}^{T} \delta_{jk}\left(t_j\right) * \sqrt{Var\left(T(C,j)\right)} * \left(\varphi_{Z_t} - \varphi_{Z_{t-1}}\right) \end{aligned} \tag{12}$$

where s_j is the service time at node j; $\delta_{jk}\left(t_j\right)$ is the travel time from site j to site k at the departure time $t_j = E\left[T(C,j)\right] + s_j$; p_{t_j} is the probability of departing at time t_j from node j. To determine p_{t_j} let $\Phi(\cdot)$ be the cumulative distribution function of the standard normal distribution and $\varphi(\cdot)$ be the corresponding density. Define $z_{t_j} = \frac{t_j - E\left[T(C,j)\right]}{\sqrt{Var\left(T(C,j)\right)}}$, then we have $p_{t_j} = \Phi\left(z_{t_j}\right) - \Phi\left(z_{t_j-1}\right)$. Perform the elimination of partial tour tests defined in Jula et al. (2006). Keep only states (C,k) those pass the tests. Do main step until $C = M - \{0\}$.

Termination Step: To complete the tour at the depot, set *k*=**0** and perform the main step. Select the tour with the minimum cost as the robust tour.

2.4.1 Time windows

When there is a time window at a site, there are three possible cases of arrival at that site with regard to the time window: early, late, and in-time arrival. In our model, we allow early arrivals (if earliness is not greater than a pre-specified value) by forcing the truck to wait until the beginning of the time window and we do not allow late arrivals (e.g. if the possibility of tardiness is greater than a pre-specified probability, we discard that partial tour).

Assume a truck traveled up to site j with a random arrival time of $T(C,j)$ and without violating any time windows. Let the time window at site j be $\left(e_j, l_j\right)$, where e_j is the earliest time and l_j is the latest time to start service at site j.

- A vehicle is assumed to be early if the probability of arriving later than e_j is less than the early arrival probability $\underline{\gamma}$: $P\left(T(C,j) \geq e_j\right) \leq \underline{\gamma}$. A vehicle can wait only if $T(C,j) \geq \left(e_j - \varepsilon\right)$,

where ε is the maximum allowable waiting time at a site; otherwise the vehicle is assumed to be *too early* and the partial tour is discarded not to waste time. Note that if a vehicle is accepted, then the start time to service is $\max\left(T(C,j),e_j\right)$.

- A vehicle is assumed to be late and the partial tour is discarded if the probability of arriving later than l_j is greater than the maximum allowable tardiness probability $\overline{\gamma}$: $P\left(T(C,j)\geq l_j\right)>\overline{\gamma}$.
- A vehicle is assumed to be in-time and is accepted if $P\left(T(C,j)\geq e_j\right)>\underline{\gamma}$ and $P\left(T(C,j)\geq l_j\right)\leq\overline{\gamma}$.

Given these definitions $E\left[T(C,j)\right]$ and $Var\left(T(C,j)\right)$ in equations (9) and (10) can be calculated with the following formulas from Chang et al. (2009):

$$E\left[T(C,j)\right]=E\left[\max\left(T(C,j),e_j\right)\right]+s_j \tag{13}$$

$$Var\left(T(C,j)\right)=E\left[\max\left(T(C,j),e_j\right)^2\right]-E^2\left[\max\left(T(C,j),e_j\right)\right] \tag{14}$$

2.4.2 Elimination of partial tours

There are some opportunities to decrease the number of partial tours under investigation and eventually to reduce the number of states and computational burden during the execution of the algorithm. The eliminations can be done through time windows and dominancy tests:

Time window test: A *partial tour* is eliminated if it does not satisfy the time windows (i.e. a too early or a late vehicle) at any stage of the execution of the algorithm.

Dominancy test: As noted earlier, there may be more than one *partial tour* for a state (C,k). Let's assume (C_1,k) and (C_2,k) are two partial tours of state (C,k) that cover same sites. We eliminate the partial tour (C_1,k) if $T(C_2,k)$ dominantes $T(C_1,k)$,(e.g. $E\left[T(C_2,k)\right]\leq E\left[T(C_1,k)\right]$ and $Var\left(T(C_2,k)\right)\leq Var\left(T(C_1,k)\right)$ (Jula et al., 2006).

2.5 Experimental Studies

We tested our procedure on a road network from Southeast Michigan (Fig. 1). The sample network covers major freeways and highways in and around the Detroit metropolitan area. The network has 140 nodes and a total of 492 arcs with 140 observed arcs and 352 unobserved arcs. Real-time traffic data for the observed arcs was collected by the Michigan ITS Center and Traffic.com for 66 weekdays of May, June, and July 2009, for the full 24 hours of each day at a resolution of one observation per 5 minutes. To better illustrate the methods and results, we first

present the data and congestion state separation and identification procedures for a sample road leg, arc between nodes 7 and 8, as an example.

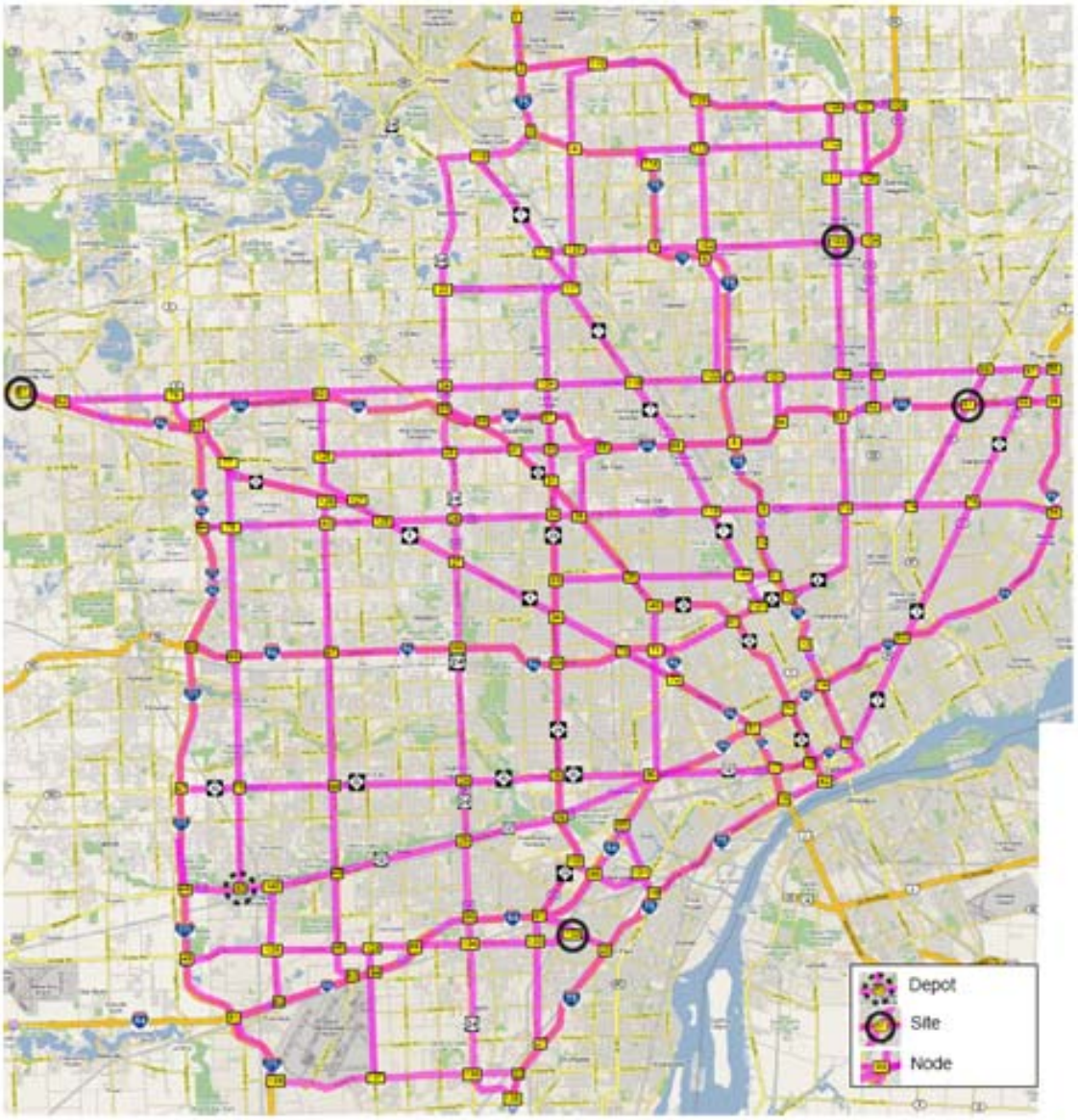


Fig. 1 Southeast Michigan road network considered for experimental study.

The speed data for arc (7, 8) for the given weekdays are illustrated in Fig. 2a. The mean and standard deviations of speed for the arc (7, 8) are plotted in (Fig. 2b). From Fig. 2a and b, it can be seen clearly that the traffic speeds follow a highly stochastic and non-stationary distribution that vary with the time of the day.

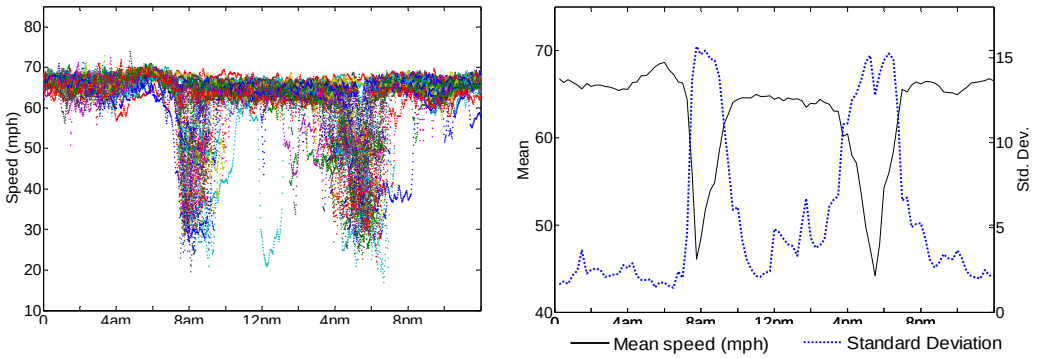


Fig. 2 For arc (7,8), (a) raw traffic speeds for 66 weekdays; (b) mean (mph) and standard deviations of speeds by the time of day with 15 minute time interval resolution.

Given the traffic speed data, we employed the Gaussian Mixture Model (GMM) clustering technique to determine the number of recurrent-congestion states for each arc by time of day. In particular, we employed the greedy learning GMM clustering method of Verbeek et al. (2003) for its computational efficiency and performance. The parameters of the traffic state joint Gaussian distributions (i.e., $\boldsymbol{\mu}^{i}_{t,t+1}; \boldsymbol{\Sigma}^{i}_{t,t+1}$) along with the computed cut-off speeds (if GMM yields more than one state) are employed to calculate travel time distribution parameters and the transition matrix elements as explained in section 3. In the event that two states are identified by GMM, let α_t denotes the probability of state transition from congested state to congested state whereas β_t denotes the probability of state transition from uncongested state to uncongested state. Fig. 3a illustrates the transition rates for arc (7, 8) with a 15 minute time interval resolution during the day. The mean travel time of arc (7, 8) for congested and uncongested traffic states is given in Fig. 3b.

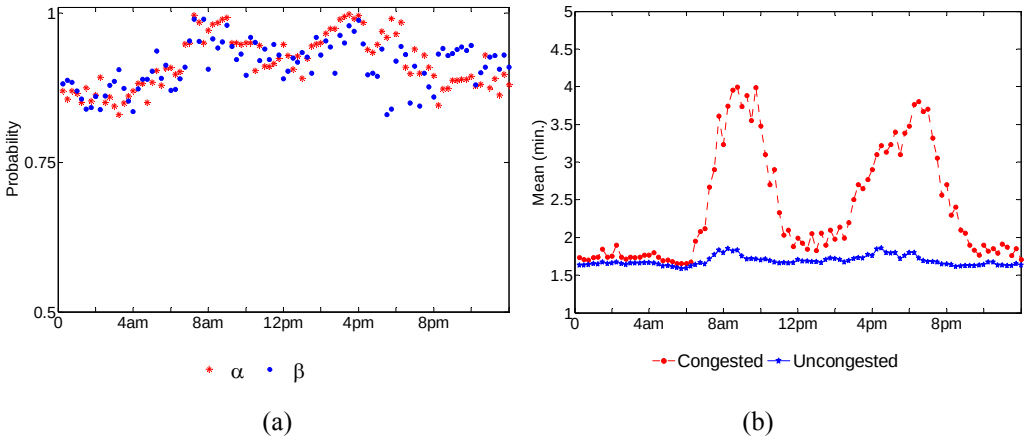


Fig. 3 For arc (7, 8) (a) congestion state-transition probabilities: α, congested to congested transition; β, uncongested to uncongested transition probability (b) mean travel time(min.) for congested and uncongested congestion states.

After calculating the parameters for every arc in the network we employed the dynamic routing algorithm to find the dynamic routing policy π_{jk} between every pair of customer sites (j,k). We then estimated the travel time distribution between sites with simulation of the policy.

We want to highlight that it is a fact that in real transportation networks the congestion states among the arcs are highly correlated. However during the simulation each arc congestion states are simulated independently. This leads to uncorrelated arc states and misleading the results. To overcome this problem we simulate the network with historical data one day at a time. Explicitly, in the first run of the simulation all arcs on the network followed the historic data of day one and so on. We ran the simulations for 66 weekdays in May, June, and July 2009. Although with this kind of simulation the number of runs is small, we believe it captures the dependency and represents the real situation better.

2.5.1 Experiment 1: Stochastic Time-Dependent TSP

In this experiment we assume there are no time windows at customer sites. Although this case is not common in the freight forwarding industry, we use the results of this experiment to define the time windows in experiment 2. This experiment also illustrates (if any) the savings better, because it examines all possible tours.

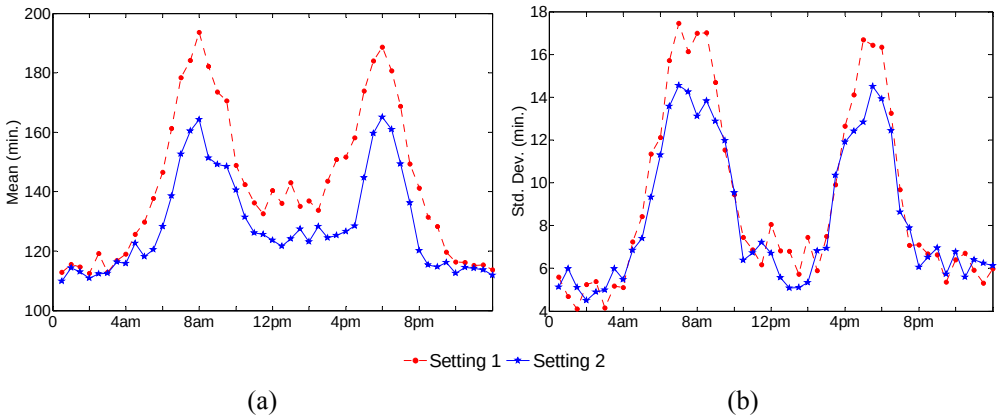
In milk runs the number of stops per tour in urban areas for a vehicle is equal to or greater than 5 stops per tour: approximately 5.6 in Denver (Holguin-Veras and Patil, 2005), 6 in Calgary (Hunt and Stefan, 2005), and 6.2 in Amsterdam (Vleugel and Janic, 2004). Conforming to the literature we also assume that there are 5 stops per tour one of which is the airport site and the other 4 are supplier sites. We also assume there is not any prerequisite about the sequence of site visits and the truck has enough capacity to visit all sites in a tour. Also, we assume that the truck begins its tour from the airport.

Node 80 is considered as the origin site (airport) while the nodes 61, 103, 51, and 132 are considered as the customer sites (Fig. **1**) of the tour. To illustrate the behavior of traffic and consequently the total travel times during the day, the trip starting times are taken as every half hour during the day. Since there is no time window none of the tour was eliminated as a result of time window constraints. Thus, there are (5-1)!=24 possible dominated and non-dominated tours. We assume the service times are 15 minutes at sites. Since there are 4 sites, the total of service time is 60 minutes for each trip.

We give the results of *total travel times* (total trip time - total service time) in (a)
(b)
Fig. **4** for the following two settings.

Setting 1: In practice, almost all commercial logistics software aims to identify TSP tours based on the given (static) path (between a pair of sites) costs which are best on average. In this context, given the traffic histories for the arcs of the network, the best path between every pair of sitesis identified first. Then, for every trip the starting time of the best tour is found by the DP algorithm for STD-TSP based on the given costs. Note that although the paths between sites are static, the milk run tour is time-dependent and might change over the time of the day.

Setting 2: In this setting, unlike the commercial logistics software there is no given static path (as a result cost) between customers. Based on the given traffic data of the network, first policies are generated between every pair of sites based on section 2.3.1. Then, these policies are used to find the travel time distribution as described in section 2.3.1. Finally, for every trip starting time the most robust tour is found by the DP algorithm for STD-TSP.



(a) (b)

Fig. 4 (a) Mean travel time (b) Standard deviation for 48 trip starting times during the day for setting 1and setting 2.

To illustrate the results better we highlighted the mean and standard deviation of tour travel times identified by the two settings for two particular departure times in Table 1. Table 1Table 1 Tour, mean travel time, and standard deviation for a sample of trip starting time for the two settings. Tours are represented with the visit sequence of nodes.

Setting	Tour	Departure Time	Mean Trip Time (min.)	Mean Travel Time (min.)	Std. Dev. of Travel Times
1st	80→132→103→51→61→80	7:00 am	244.1	184.1	16.1
2nd	80→132→103→51→61→80	7:00 am	218.4	168.4	14.2
1st	80→132→103→51→61→80	7:30 am	253.4	193.4	16.8
2nd	80→61→103→51→132→80	7:30 am	223.2	163.2	13.2

2.5.2 Experiment 2: Stochastic time-dependent TSP with Time Windows

In this experiment we assume there are time windows at sites. We assume the service times are 15 minutes at the sites. We also assume there are 6 shifts each day and shifts starting times in hours during the day are as followings: ST= {0:00; 4:00; 8:00; 12:00; 16:00; 20:00)}. In experiment 1 with setting 2, 4 tour are found to be dominating tours (e.g., with better mean trip times) for 48 different trip starting times. To illustrate we give the mean *trip times* (travel time + service time)

to complete the tours and their associate standard deviations in Table 2 for shift starting times. From these tours, the tour (denote with 1st) *80→132→103→51→61→80* is the most robust one. The other tours are the followings: 2nd tour: *80→132→51→103→61→80*; 3rd tour: *80→61→103→51→132→80*; and 4th tour: *80→61→51→103→132→80.*

Table 2 Tours mean trip times and standard deviations at the beginning of shifts based on setting 2.

Start Time			Mean						Std. Dev.			
Tour	0:00	4:00	8:00	12:00	16:00	20:00	0:00	4:00	8:00	12:00	16:00	20:00
1st	169.8	182.6	211.3	191.7	181.7	175.4	5.1	7.3	12.4	5.6	12.4	6.5
2nd	175.9	182.9	218.9	189.4	191.7	179.3	6.0	6.8	14.2	6.8	11.5	6.5
3rd	181.4	193.5	209.3	188.4	190.9	187.3	6.2	9.4	13.8	8.2	13.5	7.2
4th	179.9	199.2	221.6	192.8	194.2	185.3	6.3	9.9	12.5	7.8	12.3	7.0

Table 3Simulated mean arrival times to the nodes based on setting 1.

Start Time			Mean						Std. Dev.			
Site	0:00	4:00	8:00	12:00	16:00	20:00	0:00	4:00	8:00	12:00	16:00	20:00
80	0	0	0	0	0	0	0	0	0	0	0	0
132	17.6	18.1	26.3	18.3	26.1	17.9	0.82	0.90	1.45	0.98	2.18	1.17
103	65.1	68.7	98.7	75.1	96.4	76.5	2.54	2.72	4.13	3.02	4.82	3.11
51	96.7	102.0	141.2	110.9	129.1	107.8	3.42	3.51	5.33	3.71	6.62	3.82
61	142.2	155.5	202.1	161.6	180.5	153.1	4.37	5.03	9.32	5.06	9.94	4.88
80	172.8	185.6	242.1	196.0	218.2	191.5	5.59	7.24	17.02	6.82	14.11	6.67

Table 4 Simulated mean arrival times to the nodes based on setting 2.

Start Time			Mean						Std. Dev.			
Site	0:00	4:00	8:00	12:00	16:00	20:00	0:00	4:00	8:00	12:00	16:00	20:00
80	0	0	0	0	0	0	0	0	0	0	0	0
132	16.4	17.3	21.2	18.2	23.6	17.9	0.71	0.82	1.12	0.94	1.78	1.04
103	63.6	67.8	87.6	74.2	79.4	66.2	2.43	2.71	3.96	2.93	4.38	3.23
51	95.5	101.3	120.7	106.5	112.5	97.4	2.95	3.17	5.17	3.25	6.24	3.35
61	140.3	151.8	171.9	151.2	156.9	142.4	3.73	4.93	9.27	4.61	9.53	4.28
80	169.8	182.6	211.3	181.7	188.5	175.4	5.14	6.3	11.8	5.58	12.43	5.74

We assume the time windows are 30 minutes and the center time point of the windows are the average arrival times during the day to the nodes in the sequence of the 1st tour. We also assume the truck starts a trip in a shift and must return to the depot in that shift. We give the site time windows in **Error! Not a valid bookmark self-reference.**.

Table 5 Sites' time windows

Site	80	132	103	51	61
Earliest	0	5	61	93	142
Latest	240	35	91	123	172

We give the results with the two settings in section 2.4.1 based on the simulation of historic data. Percentages are the performance of each setting for 66 simulated days.

Table 6 The % performance of the settings under time windows

Start time	Setting 1						Setting 2					
Site	0:00	4:00	8:00	12:00	16:00	20:00	0:00	4:00	8:00	12:00	16:00	20:00
132	100	100	100	100	100	100	100	100	100	100	100	100
103	100	100	100	100	100	100	100	100	100	100	100	100
51	100	100	100	100	100	100	100	100	100	100	100	100
61	100	100	20	100	32	100	100	100	94	100	100	100
80	100	98	8	96	18	100	100	100	88	100	100	100

Table 7 Average of waiting times at nodes over 66 days

Start Time	Setting 1						Setting 2					
Site	0:00	4:00	8:00	12:00	16:00	20:00	0:00	4:00	8:00	12:00	16:00	20:00
132	0	0	0	0	0	0	0	0	0	0	0	0
103	0.05	0	0	0	0	0	0.04	0	0	0	0	0.04
51	0.23	0.04	0	0	0	0	1.39	0	0	0	0	0.05
61	1.99	0.02	0	0	0	0.27	0.98	0	0	0	0	0.06
80	0	0	0	0	0	0	0	0	0	0	0	0

2.6 Conclusions

We developed a “robust” tour while employing dynamic routing algorithms to route the truck between sites (e.g., depot/airport/customer sites). This is practical in the sense that pickup/delivery windows are often agreed upon at the time of the pickup request. Hence, there is often little opportunity to change the visiting sequence of sites. While we cannot change the visiting sequence of sites, we could still dynamically route the truck between sites.

As a future study, we will consider extending the problem where the driver not only can change his route between two sites but also can change the visiting sequence of sites. Hence, the visiting sequence of sites will be determined by an optimal policy based on the state at the time of arrival to the nodes rather than a “robust” tour sequence.

C. Heuristic Methods

In this section we summarize our efforts in developing heuristics solution methods for dynamic routing models on intermodal networks and operational response models. These heuristic methods are complementary to state space filtering and state aggregation heuristics that are applied in conjunction with the stochastic dynamic programming (SDP) algorithm. For the dynamic routing on stochastic and time-dependent intermodal networks, we outline the AO* heuristic algorithm developed and tested in Section 3.1. In Section 3.2., we summarize our efforts in developing hybrid methods which combine the Sample Average Approximation with the Progressive Hedging Algorithm.

3.1 AO* Heuristic for Dynamic Routing

Our initial experiments based on applying the stochastic dynamic programming approach for solving dynamic routing problems exhibited limitations in terms of computational efficiency. With this limitation, we concluded that the implementation of our models for practical sized problems would not be feasible. Hence we identified an alternative solution method called AO* which is described next.

The AO* algorithm is first presented by Nilsson (1980) and is based on the idea of using AND/OR networks. AO* is solution technique which is originally applied to a decomposable production system by defining an implicit AND/OR network (Nilsson, 1980). Different than the traditional networks, where arcs connect pairs of nodes, AND/OR networks have connectors directed from a parent node to a set of connected nodes. The AO* algorithm, beginning with a start node, identifies *a solution graph* to one or more terminal nodes in a pre-specified set of terminal nodes (referred as goal set). A *solution graph* is a graph which spans from the start node to all other nodes in the goal set. AO* operates by marking AND/OR network arcs for inclusion in a partial solution graph. A partial solution graph is a solution graph where not all the leaf nodes are terminal nodes (e.g. in the goal set). AO* finds a solution graph in two major operations: a top-down graph-growing operation and a back propagation operation which performs several operations (revises costs associated with each node, labels nodes as solved, etc.). Whereas the first operation functions as a constructive solution heuristic, the second operation serves as an improvement iteration.

The key characteristic of the AO* algorithm is that there is always a solution to the routing problem. Hence, when the algorithm is terminated at any time, there is a routing policy which is implementable. This is an attractive characteristic of AO* in comparison with the SDP approach since SDP cannot provide a solution until it is executed to completion. However, the routing policies obtained from the AO* are usually inferior and thus we need to solve it to optimality. This deficiency of the AO* can be improved if there is a good lower bound information on the optimal solution to the routing problem. Hence, the benefits of the AO* is most notable whenever there is

accurate lower bound information on the objective function. However, this information is seldom available given the sparsity of the data regarding dynamic air cargo routing. Therefore, we could not leverage the algorithmic efficiency of the AO*.

Upon concluding that AO* is not efficiently applicable to our problems, we focused our efforts in improving the performance of SDP algorithms. Specifically, we have modified our exact SDP algorithms and developed heuristic versions through the state space filtering and state aggregation to reduce the computational burden. The state space aggregation corresponds to reducing the time epochs for which we generate a decision (routing) policy. Whereas, in some problem settings, the routing policies with short intervals (e.g., one minute) are necessary, our experimental results indicated that reducing the time resolution of our policies does not significantly impact the quality of the solutions obtained. Hence, we have aggregated the state space by increasing the time intervals for which we generate routing policies. Second modification to the SDP algorithms is to filter the state space. Experimental results indicated that the flight paths visited in any given optimal policy is limited to a few. Hence we developed worst-case and best-case heuristics to filter out flight paths from the state space. With these two modifications, our SDP can now find optimal or close to optimal solutions in minutes which is acceptable for practical applications.

3.2 Heuristics Methods for Operational Response

In tackling static and dynamic operational response models, we developed integer programming and stochastic integer programming models, respectively. In solving these models, we have primarily relied on off-the-shelf solver engines. Whereas the static problems are easily solvable with such tools, the dynamic operational response models are significantly large and cannot be solved efficiently with off-the-shelf solver engines. Hence we resorted to heuristic methods, e.g. the Progressive Hedging Algorithm (PHA). The PHA is efficient for single period stochastic programming problems but fail to preserve the same efficiency for multi-period models such as our operational response models. Accordingly, we have begun experimenting with alternative methods such as the Sample Average Approximation (SAA) method. This method is based on the principle of scenario sampling, e.g. taking samples from the uncertainty space, and solving the samples independently and subsequently choosing the best solution. The efficiency of SAA can be set by the analyst through the determination of sample size and number of samples. However, small sample size with fewer samples, the solution quality is compromised. For a sufficiently good quality solution, the required sample size and number of samples are large. Therefore there is a notable tradeoff between the efficiency and the solution quality.

For this we began developing hybrid methods which combine PHA with SAA. The rationale is that PHA is efficient in decomposing problem into smaller problems for each realizable scenario, e.g. disruption scenario with uncertain parts delivery schedule. On the other hand, SAA has the advantage of not considering the entire set of scenarios by sampling the uncertainty space. Hence hybrid combination of these two methods would reduce the number of samples and the sample

size required to obtain good quality solutions. We have developed two prototype hybrid algorithms. Our tests with these heuristic hybrid algorithms are still ongoing and will be completed in the final year.

D. Results Dissemination

1. Conference Activity

Conference Presentations:

1. Murat, A., Aydin, N., Rossi, G. , "Impact of Flexibility on Supply Chain Resilience," OEM Global Supply Chain, *SAE World Congress* in Detroit (April 20-23, 2009).

2. Murat, A., Azadian, F., and Chinnam, R.B., "Dynamic Freight Routing on Air-Road Intermodal Network using Real-Time Congestion Information" - *POMS Annual Conference 2009 - Global Challenges and Opportunities*, Orlando, 1-4 May 2009.

3. Azadian, F., Murat, A., and Chinnam, R.B., "Vehicle routing with alternative access airport selection for air cargo," *INFORMS Annual Meeting*, San Diego (October 11-14, 2009).

Conference Sessions Organized:

1. We organized a special session titled "OEM Global Supply Chain, *SAE World Congress* in Detroit (April 20-23, 2009). The session was co-Chaired by principal investigators Dr. Murat and Dr. Chinnam.

Conferences Planning to Attend:

1. Azadian, F., Murat, A., and Chinnam, R.B., "Dynamic routing on air-road intermodal networks with milk runs," *INFORMS Annual Meeting*, Austin TX (November 7-10, 2010).

2. Journal Publications

A journal manuscript has been dispatched to the *Transportation Research Part E* journal that reports our dynamic routing of air cargo models, algorithms and their performance. A second manuscript based on dynamic routing on the Air-Road network is currently under preparation (Section A of this report) and will be submitted to *Transportation Research Part E* for review in this semester.

E. References

1. Bertsekas, D.P., 2001. Dynamic programming and optimal control. Athena Scientific, 320 pp.
2. Chang, T.-S., Wan, Y.-W. and OOI, W.T., 2009. A stochastic dynamic traveling salesman problem with hard time windows. European Journal of Operational Research, 198(3): 748-759.
3. Chen, C., Skabardonis, A. and Varaiya, P., 2003. Travel time reliability as a measure of service, 82nd Annual Meeting Transportation Research Board, Washington, D.C.
4. Cordeau, J.-F., Desaulniers, G., Desrosiers, J., Solomon, M.M. and Soumis, F., 2000. The VRP with time windows. In: P. Toth and D. Vigo (Editors), The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications. SIAM, Philadelpia,PA.
5. Golob, T.F. and Regan, A.C., 2003. Traffic congestion and trucking managers' use of automated routing and scheduling. Transportation Research Part E.
6. Groenevelt, H., 1993. The just-in-time system. In: S.C. Graves, A.H.G. Rinnooy Kan and P.H. Zipkin (Editors), Handbooks in Operations Research and Management Science. North-Holland, Amsterdam, pp. 629-670.
7. Holguin-Veras, J. and Patil, G., 2005. Observed trip chain behavior of commercial vehicles. Transportation Research Record, 1906.
8. Houck, D., Picard, J., Queyranne, M. and Vegamunti, R., 1980. The travelling salesman as a constrained shortest path problem: Theory and computational experiment. Opsearch, 17(2-3): 93-109.
9. Hunt, J. and Stefan, K., 2005. Tour-based microsimulation of urban commercial movements, 16th International Symposium on Transportation and Traffic Theory (ISTTT16), Maryland.
10. Ichoua, S., Gendreau, M. and Potvin, J.-Y., 2003. Vehicle dispatching with time-dependent travel times. European Journal of Operational Research, 144: 379-396.
11. Johnson, D.S. and McGeoch, L.A., 1997. The traveling salesman problem: A case study in local optimization. In: E. Aarts and J.K. Lenstra (Editors), Local Search in Combinatorial Optimization. Wiley, London, pp. 215-310.
12. Jula, H., Dessouky, M. and Ioannou, P.A., 2006. Truck route planning in nonstationary stochastic networks with time windows at customer locations. IEEE Transactions on Intelligent Transportation Systems, 7(1).
13. Junger, M., Reinelt, G. and Rinaldi, G., 1995. The traveling salesman problem. In: M. Ball, T. Magnanti, C. Monma and G. Nemhauser (Editors), Network Models. North Holland, Amsterdam, pp. 225-330.
14. Kim, S., Lewis, M.E. and White III, C.C., 2005a. Optimal vehicle routing with real-time traffic information. IEEE Transactions on Intelligent Transportation Systems, 6(2): 178-188.
15. Kim, S., Lewis, M.E. and White III, C.C., 2005b. State space reduction for non-stationary stochastic shortest path problems with real-time traffic congestion information. IEEE Transactions on Intelligent Transportation Systems, 6(3): 273-284.

16. Lambert, V., Laporte, G. and Louveaux, F., 1993. Designing collection routes through bank branches. Computers & Operations Research, 20(7): 783-791.
17. Laporte, G., Louveaux, F. and Mercure, H., 1992. The vehicle routing problem with stochastic travel times. Transportation Science, 26(3): 161-170.
18. Malandraki, C. and Dial, R.B., 1996. A restricted dynamic programming heuristic algorithm for the time dependent traveling salesman problem. European Journal of Operational Research, 90: 45-55.
19. Ohlmann, J.W., Fry, M.J. and Thomas, B.W., In Press. Route design for lean production systems. Transportation Science.
20. Psaraftis, H.N. and Tsitsiklis, J.N., 1993. Dynamic shortest paths in acyclic networks with Markovian arc costs. Operations Research, 41: 91-101.
21. Verbeek, J.J., Vlassis, N. and Kröse, B., 2003. Efficient greedy learning of Gaussian Mixture Models. Neural Computation, 5(2): 469-485.
22. Vleugel , J. and Janic, M., 2004. Route choice and the impact of 'logistic routes'. In: E. Taniguchi and R. Thompson (Editors), Logistics systems for sustainable cities. Elsevier.
23. M. Desrochers and G. Laporte. “Improvements and Extensions to the Miller-Tucker-Zemlin Subtour Elimination Constraints.” Operations Research Letters, 10:27–36 (1991).
24. C.E. Miller, A.W. Tucker and R.A. Zemlin. “Integer Programming Formulations and Traveling Salesman Problems.” Journal of the Association for Computing Machinery, 7:326–329 (1960).
25. R. Hall. “Freight routing and containerization in a package network that accounts for sortation constraints and costs” Metrans Project 03-07, Final report (2004).
26. N.J. Nilsson, Problem solving Methods in Artificial Intelligence, McGraw-Hill, New York (1971)
27. Baghci, A. Mahanti, Three approaches to heuristic search in networks, J. ACM 32 (1985) 1-27
28. J.L. Bander, C.C. White III, A heuristic search approach for a nonstationary shortest path problem with terminal costs, Transportation Science 36 (2002) 218–230