Dynamic routing for milk-run tours with time windows in stochastic time-dependent networks

Ali R. Güner a,*, Alper Murat b, Ratna Babu Chinnam b

a Sahintepe Mah., 133374 nolu Sok, No:6/E, Gaziantep, Turkey
b Department of Industrial and Systems Engineering, Wayne State University, 4815 Fourth St., Detroit, MI 48202, USA

A B S T R A C T

We consider finding static yet robust recurring milk-run tours while dynamically routing the vehicle between site visits. The network arcs experience recurrent congestion, leading to stochastic and time-dependent travel times. Based on vehicle location, time of day, and current and projected network congestion states, we generate dynamic routing policies (DRP) for every pair of sites using stochastic dynamic programming (SDP). By simulating DRP we find travel time distributions for each pair of sites which is used to build the robust tour using another SDP formulation. Results are very promising when the algorithms are tested in a simulated network using historical traffic data.

1. Introduction

Just-in-time (JIT) production requires frequent small-batch pickups and deliveries subject to fixed time windows. Since the shipments are usually less than a truckload, the freight carrier planners develop milk-run tours (e.g., a visiting sequence of pickup and delivery sites). In a milk-run tour, for example, the vehicle departs from a distribution center (DC), picks up goods from several supplier sites, and returns to the DC for another delivery. In planning milk-run tours, managers also consider heijunka (production smoothing or workload leveling) and muda (waste) philosophies of JIT production. Whereas the former can be achieved by equally spacing the delivery time windows over the suppliers’ operating hours, the latter can be achieved by visiting the supplier sites at an optimal frequency, balancing transportation and inventory costs. The recurrent and non-recurrent congestion on road networks increase the travel time variability thus rendering it difficult to make delivery and pickup visits within the established time windows, which can be as narrow as 15–30 min (Chen et al., 2003; Groenevelt, 1993). Some industries allow early or tardy delivery and/or pickups with a penalty (soft time windows). However, there are many practical settings (e.g., JIT production) with hard time windows where vehicles may pick up or deliver only during fixed times without exception (Cordeau et al., 2000).
In this paper, we address the problem of planning milk-run tours for JIT production subject to hard time windows in congested road networks. We model the milk-run tours as Traveling Salesman Problems (TSP) with hard time windows. The road network congestion is represented through random network arc travel times and time-dependent congestion states.

The classical TSP is concerned with finding the least cost tour that visits each site exactly once given the set of sites. The travel between any pair of sites is a path which can be static (e.g., a fixed sequence of arcs) or can be determined through a dynamic policy. The cost of travel between pairs of sites can be measured in time, distance or a function of both, be deterministic or probabilistic, and be time-dependent or independent. In STD-TSP setting, we consider a TSP with hard time windows under stochastic time-dependent (STD) arc travel times. The changes in arc congestion states represent the traffic dynamics and are modeled as Markov processes.

Recurrence of congestion in any arc is considered recurrent congestion. The recurrent congestion is more dominant for milk-run tours in STD networks. We model the recurrent congestion by defining congestion states of arcs based on historical ITS traffic data using Gaussian Mixture Model (GMM) based clustering. We consider only the recurrent congestion (e.g., rush hour) and exclude the non-recurrent (e.g., traffic incidents and inclement weather). This is necessary since the milk-run TSP tours are established for longer periods where the recurrent congestion is more dominant. We model the recurrent congestion by defining congestion states of arcs based on historical ITS traffic data using Gaussian Mixture Model (GMM) based clustering (Verbeek et al., 2003). The changes in arc congestion states represent the traffic dynamics and are modeled as Markov processes.

Accordingly, the optimal dynamic routing problem is then cast as a Markov decision process (MDP) where the states space consists of the position of the vehicle, the time of the day, and the current and projected congestion states of arcs with limited look ahead (examining the state of the full network is computationally prohibitive and even unnecessary, see Kim et al., 2005b). In optimal dynamic routing between pairs of sites, we consider only the recurrent congestion (e.g., rush hour) and exclude the non-recurrent (e.g., traffic incidents and inclement weather). This is necessary since the milk-run TSP tours are established for longer periods where the recurrent congestion is more dominant. We model the recurrent congestion by defining congestion states of arcs based on historical ITS traffic data using Gaussian Mixture Model (GMM) based clustering (Verbeek et al., 2003). The changes in arc congestion states represent the traffic dynamics and are modeled as Markov processes.

By simulating the optimal DRPs, we estimate the travel time distributions between every pair of sites. We then use these distributions to determine the optimal TSP tour by solving a stochastic dynamic programming formulation for TSP. Since the travel times are STD, we employ the convolution approach in Chang et al. (2009) to estimate the distribution of site arrival times for pickup and delivery. Whereas the routes between pairs of sites are dynamic, the TSP tour is static. This is because, in JIT production systems, the tours for pickups and deliveries support such objectives as production smoothing and workload leveling and remain fixed for extended periods (e.g., months). An optimal TSP tour can be obtained by minimizing the mean criteria combination (e.g., travel time, mileage, and truck utilization) or a mean-variance objective that also accounts for the variability of criteria. Although our methodology could accommodate a wide range of these objectives, we select a mean-variance objective based on the trip time which accounts for the transportation cost and service level (i.e., on-time performance) trade-offs in JIT production systems. We define the most robust TSP tour as the tour with minimum trip time mean-variance objective.

Our study contributes to the stochastic TSP literature in several ways. First, different than existing stochastic TSP studies, by dynamically routing between pairs of sites, we determine the path travel cost distributions between pairs of sites endogenously. With time-dependent travel times, our results show that this distinction results in selecting optimal tours that are superior than those found by considering static paths between pairs of sites in terms of cost and on-time performances. Second, we propose a robust tour formulation that selects tours based on a preset trade-off between mean and variance of the cost performance. Third, we propose a procedure to set time-windows for milk-run tours to support JIT deliveries which improves the on-time performance. Lastly, we demonstrate the proposed model and methodology using real-world network and traffic data sets.

These contributions can be summarized as follows:

- Integrating dynamic routing within stochastic TSP to determine robust milk-run tours on STD networks and the analysis of cost and on-time performance improvement of dynamic routing over static routes between milk-run sites.
- Introduction of robustness measure for selecting tours in stochastic networks and a procedure to set time-windows for milk-run tours in STD networks to improve the on-time performance.
- Using real network and traffic data, simulating the results of the proposed integrated approach and demonstrate the transportation cost and delivery service level improvement based on optimal dynamic routing between sites.

The rest of the paper is organized as follows. A selective survey of the related literature is given in Section 2. In Section 3, the modeling of the stochastic time-dependent TSP is described. Section 4 presents the experimental results of a case study application to show the effectiveness of the proposed approach. Section 5 concludes the study and suggests directions for future research.

---

3 According to Research and Innovative Technology Administration (RITA) of U.S. Department of Transportation (US DOT), “Intelligent transportation systems (ITS) encompass a broad range of wireless and wire line communications-based information and electronics technologies. When integrated into the transportation system's infrastructure, and in vehicles themselves, these technologies relieve congestion, improve safety and enhance productivity.” ITS technology and coverage is expanding quickly in the U.S. and is widely used in many developed and developing nations around the world. For more information about U.S. ITS, see: http://www.its.dot.gov.
2. Literature survey

In JIT production systems, the pickup and delivery tours are constructed while accounting for logistics drivers such as leveling the workload and decreasing inventory levels. One approach for determining pickup and delivery tours in JIT systems is the common frequency routing (CFR) method, where the suppliers are grouped into subsets and each subset of suppliers is served in a single tour (Chuah and Yingling, 2005). The CFR method considers scheduling and routing decisions jointly while accounting for transportation and inventory costs. For computational tractability, the CFR method assumes fixed routes and identical visit frequency for suppliers in the same subset. Another approach is the generalized frequency routing (GFR) where a supplier’s visit frequency is not required to be the same as other suppliers in the subset (Ohlmann et al., 2008). One of the goals in scheduling and routing decisions is to achieve production smoothing through uniformly spaced pickup and delivery visits. These “lean” routing studies consider a more general problem (e.g., vehicle routing problem -VRP) than the TSP studied in this paper but assume that the travel times on the transportation network are deterministic and time-independent. Accordingly, our focus is on selecting robust tours for a given subset of suppliers with uniformly spaced hard time windows.

The body of literature to which this study is related is the stochastic time-dependent traveling salesman problem (TSP) with time windows. In the classical TSP, given a set of sites and the cost matrix relating pairs of sites, the goal is to find the shortest tour starting from the origin site, visiting each site exactly once, and returning to the origin site. TSP and its generalization VRP have been studied for more than five decades and a wide variety of exact and heuristic algorithms have been developed (Johnson and McGeoch, 1997; Junger et al., 1995; Laporte, 2009, 2010). There are many variants of the classical TSP but we restricted our review to those studies with time-dependent and stochastic travel times. Malandraki and Dial (1996) presented a dynamic programming (DP) procedure and a “restricted” DP procedure that uses the nearest-neighbor heuristic approach to solve the time-dependent TSP (TD-TSP). They modeled the time dependency by discrete step functions such that the planning horizon had a number of different time zones and the travel times differed only at different time zones. Ichoua et al. (2003) recognized the limitation of using such step functions which violates the first-in-first-out (FIFO) principle by causing a later departure time leading to an earlier arrival time if steep speed increases occur. Accordingly, they emphasized the need to explicitly model time-dependent travel times and proposed a model to determine TSP tours in compliance with the FIFO principle.

Another variant of the classical TSP is the TSP with stochastic travel times between sites. This variant is most studied in the more general form of the vehicle routing problem (Laporte et al., 1992; Lambert et al., 1993). 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) solved the TSP through a dynamic programming approach applied to a reduced state space. They employed two-state space reduction strategies to reduce the computational complexity. Initially they estimated the mean and variance of the arrival time of the vehicle at each site based on the first (or second) order Taylor approximation. In the first strategy, they defined a service level based on the arrival times to sites and eliminated routes that did not satisfy those service levels. The other strategy eliminates states based on expected travel times. 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 in Houck et al. (1980) to solve the problem. Although the TSP problem we considered is similar to those in Jula et al. (2006) and Chang et al. (2009), the travel time distributions between pairs of sites are endogenous in our study. In particular, we integrated the construction of a TSP tour among sites with the road network routing between pairs of sites in the TSP tour. The dynamic routing between sites accounts for the time-dependent stochastic congestion states by using real-time traffic information and by anticipating congestion states with limited look ahead. To the best of our knowledge, there is no prior study proposing and integrating dynamic routing between sites for the stochastic time-dependent TSP problem. In addition, whereas Jula et al. (2006) and Chang et al. (2009) identified tour(s) with least expected tour times, we select tour(s) with minimum mean-variance objective of the trip times.

Dynamic routing and modeling of real time information have mostly been studied in the shortest path problem literature. Psaraftis and Tsitsiklis (1993) conducted the first study to consider the stochastic temporal dependence of arc costs and suggested 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 dynamic programming procedure to determine the optimal DRP. Kim et al. (2005a) studied a similar problem with assuming that the information on all of the arcs was available in real-time. They proposed a dynamic programming (DP) formulation where the state space included the states of all arcs, time, and the current node. They noted that the state space of the proposed formulation became quite large making the problem intractable. To address the intractable state-space issue, Kim et al. (2005b) proposed state space reduction methods. The modeling and partitioning of travel speeds information in Kim et al. (2005a) for the determination of arc congestion states had some limitations. As a result, the value of real-time information was compromised rendering the loss of performance of the dynamic routing policy (DRP). Güner et al. (2012) addressed these limitations by another dynamic routing approach. Moreover, they considered both recurrent and non-recurrent congestion as sources of the stochasticity on road network. For state space reduction concerns, they limit the network state information to limited-look-ahead where they only see the state information for two links ahead of the current location. Sever et al. (2013) also...
follows:

describe a procedure to set time-windows given a milk-run tour that maximizes the on-time performance.

Next, by simulating this dynamic routing policy, we estimate the travel time distributions between every pair of sites and

stochastic time-dependent dynamic programming (STD-DP) methodology to determine the robust tour. In Section 3.3, we

generate optimal dynamic routing.

sites in STD networks where the stochastic path travel times between pairs of pickup and delivery sites are estimated

through dynamic routing on the road network and thus are dependent on the site departure times. We select

the tours based on a robust tour objective that captures the tradeoff between transportation efficiency and on-time delivery

service level.

We use a sequential method to determine the robust tour. First, we determine the travel time distributions between every

pair of sites in the milk-run tour. Second, we determine the tour minimizing the robustness cost objective, i.e. mean-variance

objective of the trip time. In what follows, we first describe the procedure to estimate time-dependent travel time distributions

between the milk-run sites in Section 3.1. Given these travel time distributions between sites, Section 3.2 presents a

stochastic time-dependent dynamic programming (STD-DP) methodology to determine the robust tour. In Section 3.3, we

describe a procedure to set time-windows on a milk-run tour that maximizes the on-time performance.

3. STD-TSP with dynamic routing

The stochastic time-dependent traveling salesman problem (STD-TSP) with dynamic routing is to find a tour of a given set of

sites (i.e., DC and suppliers) while dynamically routing between sites’ visits on a stochastic time dependent (STD) network to

meet the time windows requirements. It differs from the TSP with stochastic travel times in that the travel time distributions

are obtained through dynamic routing on the road network and thus are dependent on the site departure times. We select

the tours based on a robust tour objective that captures the tradeoff between transportation efficiency and on-time delivery

service level.

In this section, we first develop a dynamic routing policy between every pair of sites in the milk-run delivery problem. Next, by simulating this dynamic routing policy, we estimate the travel time distributions between every pair of sites and at all possible departure times.

<table>
<thead>
<tr>
<th>Notation: dynamic routing</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s_a(t) )</td>
</tr>
<tr>
<td>( \pi^j_a(t) )</td>
</tr>
<tr>
<td>( TP_a(t, t + 1) )</td>
</tr>
<tr>
<td>( K )</td>
</tr>
<tr>
<td>( t_k )</td>
</tr>
<tr>
<td>( n_k )</td>
</tr>
<tr>
<td>( \text{CrAS}(n_k) )</td>
</tr>
<tr>
<td>( \text{ScAS}(a_k) )</td>
</tr>
<tr>
<td>( \text{PScAS}(a_k) )</td>
</tr>
<tr>
<td>( \Omega_k )</td>
</tr>
<tr>
<td>( \pi_k(\Omega_k) )</td>
</tr>
</tbody>
</table>

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 select next. A directed arc, \( a \), is represented by an ordered pair of nodes \( a = (n, n') \in A \) in which \( n \) is called the origin and \( n' \) is called the destination of the arc. We assume that only some of the arcs real-time traffic data (e.g., speed) is available through the ITS. A directed arc \( a \) is labeled as observed if its data is available. An arc can be in \( r + 1 \in \mathbb{Z} \) different congestion levels, with level 0 representing “Uncongested” state and the other \( r \) levels representing distinct “Congested at level \( r \)” on the basis of the arc’s average speed. We denote the congestion state of arc \( a \) at time period \( t \) with \( s_a(t) \), i.e. \( s_a(t) = \{\text{Congested at level } i \} = \{i\} \) for \( i = 1, 2, \ldots, r \) and it is determined as follows:

\[
s_a(t) = \{i; \text{ if } c^a_i(t) > v_a(t) > c^a_{i-1}(t)\} \text{ for } i = 1, 2, \ldots, r
\]

where \( c^a_i(t) \) denote the maximum (cut-off) speed for level \( i \) and \( v_a(t) \) denote the average speed on arc \( a \) at time \( t \). Assume that \( c^a_0(t) \) is the maximum possible speed on the arc \( a \) and \( c^a_r(t) = 0 \) mph as the lowest speed possible. For instance, \( s_a(t) = \{\text{Uncongested}\} = \{0\} \) and \( s_a(t) = \{\text{Congested}\} = \{1\} \) if there are two congestion levels (e.g., \( r + 1 = 2 \)). Let \( S(t) \) denote the traffic congestion state vector for the entire network with all arcs observed, i.e., \( S(t) = \{s_1(t), s_2(t), \ldots, s_K(t)\} \) at time \( t \). For presentation clarity, we will suppress \( t \) in the notation whenever time reference is obvious from the expression. Let \( s(t) \) be the state realization of \( S(t) \).
We assume that the state of an arc evolves according to a non-stationary Markov chain and arc states are independent from each other and have the single-stage Markovian property. To estimate the state transitions for each arc, we jointly model the velocities of two consecutive periods. Accordingly, the time-dependent single-period state transition probability from state \(s_0(t) = i\) to state \(s_0(t + 1) = j\) is denoted by \(P\{s_0(t + 1) = j|s_0(t) = i\} = \pi_d^j(t)\). We estimate the transition probability for arc \(a\), \(\pi_d^j(t)\) from the joint speed distribution as follows:

\[
\pi_d^j(t) = \begin{cases} 
\pi_a(t) > \pi_d^{j+1}(t) \cap \pi_a(t) > \pi_d^{j+1}(t+1) \\
\pi_d^j(t) > \pi_a(t) > \pi_d^{j+1}(t) 
\end{cases}
\]

where the \(|\cdot|\) operator corresponds to the frequency count of event \(e\). Let \(TP_a(t, t + 1)\) denote the matrix of state transition probabilities from time \(t\) to time \(t + 1\), then, we have \(TP_a(t, t + 1) = [\pi_d^j(t)]_{ij}\). 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). Given the independence assumption of the arcs’ congestion states, we can find the probability of the network state \(S(t + \delta)\) as follows:

\[
P(S(t + \delta)|S(t)) = \prod_{a=1}^{\mid A \mid} P(s_a(t + \delta)|s_a(t))
\]

where \(\delta\) is a positive integer number. Then, we can find the congestion state transition probability matrix for each arc in \(\delta\) periods from Kolmogorov’s equation:

\[
TP_a(t, t + \delta) = [\pi_d^j(t)]_{ij} \times [\pi_d^j(t + 1)]_{ij} \times \ldots \times [\pi_d^j(t + \delta)]_{ij}
\]

We assume that the distribution of an arc travel time is normally distributed and depends on the congestion state of the arc, \(s_a(t)\), at the time of departure \((t)\):

\[
\delta(t, a, s_a) \sim N(\mu(t, a, s_a), \sigma^2(t, a, s_a))
\]

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 \(a\). For clarity of notation, we hereafter suppress the arc label from the parameter space wherever it is obvious, i.e. \(\delta(t, a, s_a)\) will be referred as \(\delta_a(t, s)\).

In order to find the optimal dynamic routing policy between every pair of sites, we employ the backward recursion to minimize the expected travel time based on real-time information such as the path originates at the origin node \(n_0\) and ends at the destination node \(n_k\). Assume that there is a feasible path between \((n_0, n_j)\) where a path \(p = (n_0, n_k)\) is defined as the sequence of (decision) nodes where \(k = 0, \ldots, K - 1\) and \(K\) is the number of decision stages (e.g., number of nodes visited on a path). We define arc set \(a_0 = (n_0, u) \in A\) as the current arc set of node \(n_0\), denoted with \(CrAS(n_0)\). That is, \(CrAS(n_0) = \{a_0 : a_0 = (n_0, u) \in A\) with \(u \in N\}\) is the set of arcs emanating from node \(n_0\). Routing decisions (which node to select next) are made at each node on a path. Let \(n_k \in N\) be the location of \(k\)th decision stage, \(t_k\) be the time at \(k\)th decision stage where \(t_k \in \{1, \ldots, T\}\) \(T > t_{k-1}\). In our model we discretize the planning horizon, \(T\). \(T\) is an arbitrarily large number and is used to limit the planning horizon for modeling purposes.

An optimal dynamic routing policy requires the projection of the traffic states of the complete network, however this approach renders the state space excessively large. Since the projected information is not very different from the steady state probabilities of the arc congestion states, there is little value in projecting the congestion states well ahead of the current location. Thus, there is a tradeoff between efficient but practical approach and the degree of look-ahead (e.g., the number of arcs to monitor) with the resulting projection accuracy and routing performance. This concept was very well illustrated in Kim et al. (2005b). Accordingly, we limit our look-ahead to a finite number of arcs that can vary by the vehicle location (current node). The selection of the arcs to take into account would depend on factors such as arc lengths, the value of real-time information, time of the day, and the congestion state transition characteristics of the arcs. For presentation purposes and without loss of generality, we choose to monitor only two arcs ahead of the vehicle location and model the rest of the arc congestion states through their steady state probabilities. Consequently, we define the so-called successor and post-successor set for all the arcs in the network: (1) \(ScAS(a_k)\) is the successor arc set of arc \(a_k\): \(ScAS(a_k) = \{a_{k+1}, n_{k+1}\}\) is the successor arc set of \(a_k\) with \(u \in N\), i.e., the set of outbound arcs from the destination node \((n_{k+1}, a_{k+1})\) of arc \(a_k\). (2) \(PScAS(a_k)\) is the post-successor arc set of arc \(a_k\): \(PScAS(a_k) = \{a_{k+2}, n_{k+2}\}\) is the set of outbound arcs from the destination node \((n_{k+2}, u) \in ScAS(a_k)\) and \(u \in N\), i.e., the set of outbound arcs from the destination nodes of \(ScAS(a_k)\).

Since the total path 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. \(\Omega_k\) denotes the state \((n_k, t_k, s_{k+1}, a_{k+1}, k)\) of the system at the \(k\)th decision stage. The state vector has the information of the current node \((n_k)\), arrival time to the current node \((t_k)\), and \(s_{k+1}, a_{k+1}, k\), the congestion state of arcs set of \(a_{k+1} \cup a_{k+2}\) where \(\{a_{k+1} \in ScAS(a_k)\}\) and \(\{a_{k+2} \in PScAS(a_k)\}\) at \(k\)th decision stage. The set of current arcs of node \(n_k\), \(CrAS(n_k)\) is the action space for the state \(\Omega_k\).
Alternative arcs from the action space are evaluated based on the remaining expected travel time at every decision stage. With the selection of an outgoing arc, the expected travel time at a given node is the total of expected travel time on the chosen arc and the expected travel time of the chosen arc’s next node. Let \( \pi_{n_kn_j} = \{\pi_0, \pi_1, \ldots, \pi_{K-1}\} \) be the dynamic routing policy (DRP) of the trip that is composed of policies for each of the \( K-1 \) decision stages. For a given state \( \Omega_k = (n_k, t_k, s_{n_k-1}, n_{k+1}, k) \), the policy \( \pi_k(\Omega_k) \) is a deterministic Markov policy that chooses the outgoing arc from node \( n_k \), i.e., \( \pi_k(\Omega_k) = a \in CAAs(n_k) \). Thus, for a given policy vector \( \pi \), the expected travel cost can be found as follows:

\[
F^*(\Omega_k) = \min_{\pi} \left\{ \sum_{k=0}^{K-2} g(\Omega_k, \pi_k(\Omega_k), \delta_k) + g(\Omega_{K-1}) \right\}
\]

where \( \Omega_k = (n_k, t_k, s_k) \) is the starting state of the system. \( \delta_k \) is the random travel time at decision stage \( k \), i.e., \( \delta_k \equiv \delta(t_k, \pi_k(\Omega_k), s_k(t_k)) \). \( g(\Omega_k, \pi_k(\Omega_k), \delta_k) \) is cost of travel on arc \( \pi_k(\Omega_k) = a \in CAAs(n_k) \) at stage \( k \), i.e., if travel cost is a function \( (\phi) \) of the travel time, then \( g(\Omega_k, \pi_k(\Omega_k), \delta_k) \equiv \phi(\delta_k) \) and \( g(\Omega_{K-1}) \) is terminal cost of earliness/tardiness of arrival time to the destination node under state \( \Omega_{K-1} \). Note that \( K \) represents the number of decision stages and, accordingly, \( k = 0, 1, \ldots, K-1 \) corresponds to each of the decision stages. Since the decisions are made at each node, the value of \( K \) is bounded by the total number of nodes in the network, i.e., \( K = |N| \). While this number could be large, the dynamic programming approach (described next) selects much fewer number of nodes to be visited than the maximum number possible in most practical scenarios. Nonetheless, we initialize the \( K \) at its upper bound since, under very rare scenarios, it is possible that the optimal DRP could be attained by visiting all the nodes of the network. Hence, with the exception of such rare scenarios, the optimal DRP will have effective number of stages that are less than the upper bound, i.e., for a given starting time and congestion state at the origin node \( n_0 \), the number of decision stages (a.k.a. number of nodes to be visited) to arrive to the destination node \( n_T \) is less than \( |N| \). Accordingly, the optimal DRP’s effective number of decision stages would then correspond to the actual number of nodes to be visited for a given starting time and congestion state at the origin node \( n_0 \) to arrive to the destination node at the least cost possible. In order to find the minimum expected travel time, we minimize \( F(\Omega_0) \) over the policy vector \( \pi \) as follows:

\[
F(\Omega_0) = \min_{\pi_{n_0n_j}} \{ F(\Omega_0) \}
\]

The corresponding optimal policy is then:

\[
\pi^*_{n_0n_j} = \arg\min_{\pi_{n_0n_j}} \{ F(\Omega_0) \}
\]

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

\[
F(\Omega_k) = \min_{\pi_k} F(g(\Omega_k, \pi_k(\Omega_k), \delta_k) + F(\Omega_{k+1}))
\]

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:

\[
F(\Omega_k) = \sum_{\delta_k} P(\delta_k|\Omega_k, a_k) \left[ g(\Omega_k, a_k, \delta_k) + \sum_{s_{n_k-1,k+1}} P(s_{n_k-1,k+1}(t_{k+1})|s_{n_k-1,k}(t_k)) \sum_{s_{n_k+1,k+1}} P(s_{n_k+1,k+1}(t_{k+1})|s_{n_k,k}(t_k)) \right]
\]

where \( P(\delta_k|\Omega_k, a_k) \) is the probability of traveling arc \( a_k \) in \( \delta_k \) periods. \( P(s_{n_k-1,k+1}(t_{k+1})) \) is the long run probability of arc \( a_{k+1} : n_k \rightarrow n_{k+1} \in CAAs(a_k) \) being in state \( s_{n_k-1,k+1} \) in stage \( k+1 \). This probability can be calculated from the historical frequency of a state for a given arc and time.

To solve \( F(\Omega_k) \), \( k = K-1, K-2, \ldots, 0 \) we use the backward dynamic programming algorithm. In the backward induction, we initialize the final decision epoch such that \( \Omega_{K-1} = (n_{K-1}, s_{n_{K-1}-1}, n_{K-1}) \). \( n_{K-1} \) is the destination node, and \( F(\Omega_{K-1}) = 0 \) if \( t_{K-1} > T \). Accordingly, if there is delivery tardiness, e.g., \( t_{K-1} > T \) we add a penalty cost. Note that \( s_{K-1} = \emptyset \), since the destination node does not have any current and successor arc states, e.g. the travel terminates at the destination node.

### 3.1.1. Estimating travel time distributions between sites

Let \( M \) be the set of sites to be visited (e.g. DC and suppliers), given a pair of sites, origin \( j \in M \) and destination \( k \in M \), we formulated the dynamic programming formulation for all feasible departure times from \( j \) and obtain the optimal routing policy, \( \pi_k \). Next, for each departure time alternative \( (t_j) \), we sample a congestion state \( s(t_j) \) for current and successor arcs of \( j \), and simulate the policy corresponding to the sample state \( \Omega = (j, t_j, s(t_j)) \). Note that the sampling probabilities of the congestion state \( s(t_j) \) are based on the steady-state probabilities of the states of current and successor arcs of \( j \). Following sufficient sampling for \( t_j \), we estimate the distribution of the mean travel times obtained by simulating corresponding policies for each sampled state \( \Omega \). We then calculate the expectation and variance of travel time from \( j \) to \( k \) at time \( t_j \) and respectively denote them with \( E(\Delta_k(t_j)) \) and \( Var(\Delta_k(t_j)) \). Note that, with slight abuse of notation, \( \Delta_k(t_j) \) corresponds to the random travel time between \( j \) and \( k \) departing at \( t_j \).
3.2. Dynamic programming for STD-TSP

In this section, we describe the stochastic time-dependent dynamic programming (STD-DP) approach for selecting a robust tour of a given set of sites (i.e., DC and suppliers) while dynamically routing between sites’ visits to meet the time windows requirements. The time window requirements are strict (e.g., hard time windows) and each site has a deterministic service time for loading/unloading. This STD-DP approach integrates and builds on the results of earlier studies. Specifically, it integrates the stochastic tour search procedure from Malandraki and Dial (1996) and Jula et al. (2006) and the convolution idea from Chang et al. (2009). However, the proposed STD-DP approach uses the travel time distributions obtained in the preceding section by dynamically routing on the road network. Further, the approach selects the most robust tour by trading off the expected duration of the tour with its variability as follows:

\[ T_{C00} = E[T(M, \tau, 0)] + b \sqrt{Var(T(M, \tau, 0))}, \]

(11)

where \( \tau \) is the tour, \( E[T(M, \tau, 0)] \) and \( Var(T(M, \tau, 0)) \) are the expected value and variance of the round trip duration departing from site 0 (DC) at time \( t_0 \), visiting all sites in \( M \) once, and returning back to site 0; \( b \) is a user defined risk-parameter for balancing the transportation efficiency with on-time delivery performance.

### Notation: dynamic programming for STD-TSP

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( M )</td>
<td>Set of sites to be visited including DC, i.e., ( 0, 1, \ldots, m-1 \in M )</td>
</tr>
<tr>
<td>( (C, k) )</td>
<td>Unordered set of visited sites, i.e., ( (C, k) \subseteq M/(0) ) where ( k \in C ) is the last visited site</td>
</tr>
<tr>
<td>( \tau )</td>
<td>Partial tour that starts from the DC, visits all sites in ( (C, k) ) and ends the tour at site ( k ), i.e. ( \tau \in \Gamma(C, k) )</td>
</tr>
<tr>
<td>( T(C, \tau, k) )</td>
<td>Random variable of arrival time at site ( k ) taking the partial tour ( \tau ) of set ( (C, k) )</td>
</tr>
<tr>
<td>( s_j )</td>
<td>Deterministic service time at site ( j )</td>
</tr>
<tr>
<td>( \Delta_{jk}(t_j) )</td>
<td>Travel time from site ( j ) to site ( k ) at the departure time ( t_j = T(C, \tau, j) + s_j )</td>
</tr>
<tr>
<td>( p_{t_j} )</td>
<td>Probability of departing at time ( t_j ) from node ( j )</td>
</tr>
</tbody>
</table>

We first describe the STD-DP approach without the time-windows and present its extension to time window case. There are \( m-1 \) sites (other than the DC, assuming the vehicle at the DC) to be visited, represented by nodes \( 1, \ldots, m-1 \in M \). Let \( (C, k) \subseteq M/(0) \) be an unordered set of visited sites where \( k \in C \) is the last visited site. Define partial tour \( \tau \) as a tour that starts from site 0 (DC), visits all sites in \( (C, k) \) only once and ends the tour at site \( k \). Note that there may be more than one partial tour corresponding to \( (C, k) \) and we denote the set of partial tours with \( \tau \in \Gamma(C, k) \). For brevity, we do not repeat the membership of partial tours in the remainder and assume \( (C, \tau, k) \) implies \( \tau \in \Gamma(C, k) \). Let \( T(C, \tau, k) \) be the random variable of arrival time at site \( k \) taking the partial tour \( \tau \) of set \( (C, k) \) after departing site 0 at time \( t_0 \). Let also \( E(T(C, \tau, k)) \) and \( Var(T(C, \tau, k)) \) be the mean and variance of arrival time to site \( k \), \( T(C, \tau, k) \), after taking the partial tour \( \tau \), respectively.

**Step 1. Initialize:** For all \( |(C, k)| = 1 \) where \( (C, k) = \{k\}, k \in M/(0) \), we initialize \( E[T(C, \tau, k)] = T(0) + S_0 + E[\Delta_{tk}(t_0)] \) and \( Var(T(C, \tau, k)) = Var[\Delta_{tk}(t_0)] \), where \( T(0) \) is the arrival time to site 0 (DC), \( S_0 \) is the service (e.g., loading/unloading) time at site 0, and \( E[\Delta_{tk}(t_0)] \) is the expected travel time from site 0 to site \( k \) as a function of the departure time, \( t_0 \). Note that if there is no waiting at site 0 then \( t_0 = T(0) + S_0 \). Also note that the expectation value \( E[\Delta_{tk}(t_0)] \) is over the congestion states of current and successor arcs of site 0.

**Step 2. Main:** For all \( |(C, k)| > 1 \), there are partial tours of set \( (C, k) \), where we visit \( k, k \in M/(0,j) \) immediately after \( j \) (for all \( j \in (C/k) \)). The mean and variance \( T(C, \tau, k) \) for the partial tour \( \tau \) is calculated through the following convolution propagation approach adapted from Chang et al. (2009):

\[
E[T(C, \tau, k)] = E[T(C, \tau, j)] + s_j + \sum_{t_j} E[\Delta_{tk}(t_j)]p_{t_j},
\]

(12)

\[
Var(T(C, \tau, k)) = Var(T(C, \tau, j)) + \sum_{t_j} p_{t_j} Var[\Delta_{tk}(t_j)] + \sum_{t_j} p_{t_j} E[\Delta_{tk}(t_j)]^2 - \left[ \sum_{t_j} p_{t_j} E[\Delta_{tk}(t_j)] \right]^2 - 2 \sum_{t_j} E[\Delta_{tk}(t_j)]
\]

\[
\times \sqrt{Var(T(C, \tau, j))}(\phi_{t_{j}} \phi_{t_{j-1}})
\]

(13)

where \( s_j \) is the deterministic service time at site \( j \); \( \Delta_{tk}(t_j) \) is the travel time from site \( j \) to site \( k \) at the departure time \( t_j = T(C, \tau, j) + s_j \); \( p_{t_j} \) is the probability of departing at time \( t_j \) from node \( j \). Note that the expectation value \( E[\Delta_{tk}(t_j)] \) is over the congestion states of current and successor arcs of site \( j \). Let \( z_j = \frac{t_j - E(T(C, \tau, j))}{\sqrt{Var(T(C, \tau, j))}} \), we calculate \( p_{t_j} \) as \( p_{t_j} = \Phi(z_j) - \Phi(z_{j-1}) \), where \( \phi(\cdot) \) and \( \Phi(\cdot) \) are the density and cumulative distribution functions of the standard normal distribution, respectively. Once \( T(C, \tau, k) \) is calculated for all \( |(C, k)| > 1 \), we decrease the number of partial tours under investigation by performing the following partial tour elimination test adapted from Jula et al. (2006).
Dominancy test: There may be more than one partial tour for a set \((C, k)\). Let us assume \((C, \tau_1, k)\) and \((C, \tau_2, k)\) are two partial tours of set \((C, k)\) that cover the same sites. We eliminate the partial tour \((C, \tau_1, k)\) if \(T(C, \tau_2, k)\) dominates \(T(C, \tau_1, k)\), e.g., \(E[T(C, \tau_2, k)] \leq E[T(C, \tau_1, k)]\) and \(Var(T(C, \tau_2, k)) \leq Var(T(C, \tau_1, k))\).

Additional partial tour elimination tests based on time windows are described in the next section. After testing all pairs of partial tours, we repeat the main step until \(C = M - \{0\}\).

Step 3. Termination: To complete the tour at site 0 (DC), we set \(k = 0\) and perform the main step one last time and obtain the expectation and variance of the total tour time \(T(C, \tau, 0)\) for all remaining tours \(\tau\) of \((C, 0)\) where \(C = M\). For each of the remaining tours we calculate \(TC_{oo}\) by Eq. (11) and select the tour with minimum mean-variance as the robust tour solution.

3.2.1. STD-TSP with time windows

In the preceding section, we presented STD-DP for solving the STD-TSP without time windows. This subsection extends it to cases with hard time windows. When there is a time window requirement at a site, there are three possible arrival scenarios to that site with regard to the time window: early, late, and on-time arrival. In our model, we allow early arrivals, if earliness is not greater than a pre-specified value, by requiring the vehicle to wait until the beginning of the time window. We do not allow late arrivals by eliminating partial tours with the possibility of tardiness greater than a pre-specified probability. Note that strictly enforcing hard time windows (i.e. through zero threshold probability) would result in inefficient tours. Hence, by using a non-zero probability threshold, we are in essence treating the hard time windows in a non-strict manner.

Let us assume the vehicle arrives at site \(j\) with a random arrival time of \(T(C, \tau, j)\) with partial tour \(\tau\) and does not violate any time window requirement. Let \((e_j, l_j)\) be the time window at site \(j\), where \(e_j\) is the earliest time and \(l_j\) is the latest time to start service at site \(j\).

- **Early Arrival**: The vehicle arrival is assumed to be early if probability of arriving later than \(e_j\) is less than the early arrival probability \(\gamma\): \(P(T(C, \tau, j) \geq e_j) \leq \gamma\). The vehicle can wait only if the mean arrival time is greater than \(e_j - \varepsilon\), where \(\varepsilon\) is maximum allowable waiting time at the site; otherwise the vehicle is assumed to be too early and the partial tour is discarded. Note that if a particular vehicle arrival is accepted, the start time to service is \(\max(T(C, \tau, j), e_j)\).
- **Late Arrival**: The vehicle arrival is assumed to be late and the partial tour is discarded if probability of arriving later than \(l_j\) is greater than the maximum allowable tardiness probability \(\bar{\gamma}\): \(P(T(C, \tau, j) \geq l_j) > \bar{\gamma}\).
- **On-time Arrival**: The vehicle arrival is assumed to be on-time and is accepted if both \(P(T(C, \tau, j) \geq e_j) > \gamma\) and \(P(T(C, \tau, j) \geq l_j) \leq \bar{\gamma}\).

Given these definitions, Eqs. (12) and (13) can be replaced by:

\[
E[T(C, \tau, j)] = E[\max(T(C, \tau, j), e_j)] + s_j \tag{14}
\]

\[
Var(T(C, \tau, j)) = E[\max(T(C, \tau, j), e_j)^2] - E^2[\max(T(C, \tau, j), e_j)] \tag{15}
\]

Note that the maximization operator is due to the waiting upon early arrival. For late arrivals, the maximum operator in (14) and (15) does not exist since there is no waiting with late arrivals. In both early and late arrival cases, we eliminate those partial tours according to the corresponding pre-defined parameters \((\gamma, \varepsilon, \bar{\gamma})\). Note that, different than the stochastic dominance elimination, time window eliminations are used in the initialization step and at the termination step if there are also DC time windows applicable to the tour completion time. In this study, we assume that the suppliers are ready for the pickup/delivery of the load upon the vehicles arrival and that this loading/unloading service time is deterministic. An extension for the deterministic service time is to account for the uncertainty which can be caused for supplier tardiness in loading/unloading upon the vehicle’s arrival to the supplier site. Clearly, in the presence of such uncertainties, the effectiveness of the routing optimization would be impacted. Depending on the severity of such pickup/delivery site delay uncertainty, the aforementioned methods should be modified to account for this additional delay uncertainty.

3.3. Determining time windows for a given tour to improve on-time performance

We described how the STD-DP approach is extended for problems with hard-time windows, however, in most just-in-Time (JIT) production systems, the time window requirements affect different parties differently. For instance, the carriers are penalized for late deliveries either by charges associated with contracted service levels or by their reduced ranking as a transportation service supplier. In comparison, early arrivals correspond to lower utilization of assets and drivers. The suppliers (pickup sites), on the other hand, need to stock more safety inventory and allocate more material handling resources if time windows are relaxed (e.g., width of the window is increased). The width of the time window and their positioning constitute two features of most logistics contracts and are often re-adjusted due to changing production volumes and routes. The time window setting process differs from industry to industry. In JIT environments, it is common that the time windows are set by trucking and/or manufacturer companies according to JIT principles and are usually accepted by the suppliers as
part of the sourcing contract. In such a setting, the trucks visit the supplier sites several times per day subject to the tight time windows spaced as much evenly as possible within the supplier’s operating hours (even spacing is generally key to supplier efficiency; reduces finished goods inventory levels). Similar to supplier’s time windows, the OEM might have commit-to-delivery time requirements for the transportation service providers. Clearly, given the travel time variability, the route travel time does not necessarily always meet the OEM’s imposed commit-to-delivery time. In this paper, we assume that the OEM’s DC maintains sufficient safety stock of components to account for any impact due to the mismatch, i.e., between the inbound (of components from the suppliers) and outbound (deliveries to OEM’s manufacturing plants) shipments. Clearly, the cost of maintaining such inventory has to be traded off with the transportation cost of having fewer supplier sites per tour and increased number of tours/vehicles. This consideration where the OEM contracts a transportation service provider, requires commit-to-deliver time requirement, and trades of the cost of transportation with that of safety inventory constitutes a future research direction of this work.

We now describe a procedure for carriers to position the time windows such that the on-time delivery performance is improved. We assume that the width of time windows (w) is determined beforehand by the supplier and manufacturer and they are indifferent to the positioning of the time windows as long as they are uniformly distributed during the delivery horizon. The procedure uses the result that the site arrival times follow Gaussian distribution when the arc travel times are also Gaussian (Chang et al., 2009). Therefore, centering the time windows at the expected site arrival times maximizes the on-time delivery performance, if, there is no waiting allowed at the site for early deliveries. This is indeed the case practiced by carriers even if there is some flexibility in early arrival acceptance. Let \( \tau_k \) be the selected ordered tour that starts from DC, visits all sites once and ends the tour at site \( k \). Accordingly, \( T(C, \tau_k, k) \) is the random variable of arrival time at site \( k \) by following the partial tour \( \tau_k \). Let also \( E[T(C, \tau_k, k)] \) and \( \text{Var}(T(C, \tau_k, k)) \) be the mean and variance of arrival time \( T(C, \tau_k, k) \), respectively.

Procedure for Setting Time Windows:

For \( k = 1, \ldots, m - 1 \), Repeat:

If \( k \) is the first site visited by tour \( \tau \),

\[
E[T(C, \tau_k, k)] = T(0) + s_0 + E[A_{0k}(t_0)] \quad \text{and} \quad \text{Var}(T(C, \tau_k, k)) = \text{Var}(A_{0k}(t_0))
\]

where \( T(0) \) is the arrival time to site 0 (DC), \( s_0 \) is the service time at site 0, and \( E[A_{0k}(t_0)] \) is the expected travel time from site 0 to site \( k \) as a function of the departure time, \( t_0 \).

else,

Assume visiting \( k \) immediately after \( j \) and look up the updated \( E[T(C, \tau_j, j)] \) from the previous step. Calculate \( E[T(C, \tau_k, k)] \) from Eq. (12).

Set \( e_k = E[T(C, \tau_k, k)] - w/2 \) and \( l_k = E[T(C, \tau_k, k)] + w/2 \).

Update \( E[T(C, \tau_k, k)] \) and \( \text{Var}(T(C, \tau_k, k)) \) according to Eqs. (14) and (15).

Return.

The above procedure is an iterative procedure where we visit sites according to the tour \( \tau \) and set time windows for each site one at a time. At each step, we calculate the expected arrival time to that site based on the time windows set at the previously visited sites. We account for the previously set time windows because they affect the site arrival time of the subsequent visited sites through the waiting at early arrivals. Note that the centered placement of time windows is an assumption. It is possible to shift the time windows to the right of the center (expected site arrival time) such that the likelihood of late arrivals decreases. Clearly, this modification is contingent upon the maximum allowable waiting time imposed for early arrivals. In the case of unrestricted waiting, it can be shown that, by shifting the time window to the right, one can turn time window constraints into redundant constraints.

3.4. Relationship to integrated production and operational transportation planning

The proposed problem and solution approach, while focused on the cost efficiency and service level considerations associated with the operational transportation planning, is closely related to the integrated planning of productions and transportation operations. Today’s effective supply chains plan for the manufacturing operations in coordination with transportation operations by scheduling production runs and logistics pickup and deliveries while optimizing the inventory levels (cycle and safety) to align supply and demand for service level targets. In this work, we focused primarily on the transportation planning aspects of the integrated supply chain planning.

In addition to the criticality of optimizing the transportation routes for routing cost efficiency and on-time delivery performance, the optimal planning of the visit frequency of each supplier site in coordination with the supplier and manufacturer inventory management is also important for the integrated planning. While our study concerns route planning given a set supplier sites to visit, a truly integrated supply chain planning will determine the frequencies and routes of supplier visits as well as fleet size while optimizing the inventory levels (both cycle and safety), time windows, and routes. To achieve this integration, the present study’s scope needs to be extended in three directions to include supply and inventory management, which requires revisiting the time-windows setting and routing to achieve the service level targets.
• **Inventory Management – Cycle Inventory**: Integrated planning of cycle inventory management and route planning requires determining periodic (e.g., weekly) visit schedules which include optimal visit frequencies, allocations of visits in addition to route planning. This integrated problem is even more challenging than the dynamic routing problem studied in this study since it extends the static periodic vehicle routing problem (Toth and Vigo, 2014). One way that the integrated problem can be tackled for cycle inventory management, albeit heuristically, is to first solve an inventory optimization problem to determine the frequencies of visits to supplier sites. Given these visit frequencies, next task is to assign multiple supplier visits into multiple vehicle routes throughout the day. For a given assignment, the routing problem solved herein could be employed. This is a heuristic approach since the ensuring of the optimality of the visit frequencies necessitates the knowledge of the supplier visit-trip allocation and routing solutions. Nonetheless, an iterative procedure where the frequencies are determined optimally (at the master problem), visit and trip allocations (at the first level sub-problem) and route optimization (at the second level sub-problem) can be designed.

• **Inventory Management – Safety Inventory**: The determination of the visit frequencies and assigning them to periodically repeated tours resolves the cycle inventory levels at the supplier sites. However, another critical element of the integrated planning problem is the impact of the supplier site arrival time uncertainty on the safety stock levels required at the supplier sites. As shown above, the proposed modeling and solution approach allows for estimating the arrival time distribution information required in calculating the safety stock requirements at the supplier sites. Accordingly, once the visit schedules are identified, the route planning can be optimized while setting the safety stock levels to satisfy the service level requirements. This last step requires modifying the objective function in (11) to account for the arrival time variability to each of the sites (i.e. not just the variability of the total tour completion time) in a given tour.

• **Production Scheduling**: Lastly, another layer of decision making in the integrated supply chain planning of manufacturing and logistics operations is the production scheduling. The integrated problem of production, inventory and static routing is a challenging problem and our consideration of dynamic routing between sites would further increase the computational complexity (Bard and Nanananukul, 2009). One heuristic approach is to first solve the integrated problem for the inventory management and route planning and then make the production scheduling decisions. However, this heuristic approach can result in significant suboptimality if the production costs are subject to economies of scale and inventory costs are high.

4. Experimental study

In this section, we test the proposed methodology on a real case study application using the road network from Southeast Michigan, U.S.A. (Fig. 1). We consider an automotive JIT production system where an original equipment manufacturer’s (OEM) distribution center is replenished by milk-run pickup and deliveries from multiple suppliers. The case study road 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 is collected by the Michigan ITS Center and Traffic.com. In this application, we used data from 66 weekdays of May, June, and July 2009, for the full 24 h of each day. The raw speed data was aggregated at a resolution of 5 min intervals. For the experimentation, we increased the resolution of data to one data-point per minute through linear interpolation (see Kim et al., 2005a). Since the collected speed data is averaged across different vehicle classes (i.e., automobile, trucks) and no data was available for individual classes of vehicles, we assumed that the truck being routed could also cruise at the collected average speeds. We implemented all of our algorithms and methods in Matlab 7 and executed them on a machine (with CPU 2.2 GHz and 4 GB RAM) running Microsoft Windows 7 operating system.

Our experimental study is outlined as follows: Section 4.1 describes the estimation and modeling process for recurrent congestion and illustrates through a sample arc of the network. Section 4.2 explains the steps of generating DRPs and estimating travel time distributions between sites. Section 4.3 presents experimental results of identifying and selecting robust STD-TSP tours without time windows and reports savings from employing the dynamic routing policy over the static routing policy between pair of sites. Section 4.4 evaluates the performance of routing policies identified in Section 4.3 after setting the sites’ time windows as described in Section 3.3.

4.1. Estimating congestion states

The proposed dynamic routing algorithm calls for identification of different congestion states, estimation of their state transition rates, and estimation of arc traverse times by time of the day. To better illustrate the modeling of congestion states, we present the data and congestion state identification and separation procedures for an example arc (7, 8). The speed data for arc (7, 8) for the weekdays is 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 clearly seen that the traffic speeds follow a non-stationary distribution that vary highly with time of the day.

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 the day. In particular, we used the greedy learning GMM clustering method of Verbeek et al. (2003) for its computational efficiency and performance. After obtaining the state clusters for each time interval t, we first estimate the time-dependent cut-off speeds if GMM yields more than one congestion state at t. Next, given cut-off speeds, we then estimate the parameters of the Gaussian distributions for state transitions for congestion
state $i$ from $t$ to $t+1$ for all $t$, i.e., $(\mu^i_{t+1}, \Sigma^i_{t+1})$. Applying GMM for arc (7, 8), for instance, recommended two clusters of congestion states for almost all time intervals. Fig. 3a illustrates the transition rates for arc (7, 8) with a 15 min time interval resolution during the day. Note that we are using two clusters for arc (7, 8) in all time intervals for presentation purpose (other than increasing computational burden, there are no other consequences). In Fig. 3a, the $\alpha_i$ denotes the probability of state transition from congested state to congested state and $\beta_i$ denotes the probability of state transition from uncongested state to uncongested state. The mean travel time of arc (7, 8) for congested and uncongested traffic states is given in Fig. 3b.
4.2. Estimating travel time distributions between sites

Using the previous section’s results, e.g., time and congestion state dependent distribution of arc travel times and congestion state transition probabilities, we employed the dynamic routing algorithm in Section 3.1 to determine the dynamic routing policy \( p_{jk} \) between every pair of customer sites \((j, k)\) at different departure times. Next, we estimate the travel time distribution between every pair of sites. This can be achieved by simulating the optimal dynamic policies in two different ways: using estimated arc travel time distributions as described in Section 3.1 or using the available historical data for 66 weekdays. We choose to use the historical data because of the link interactions and dependencies not captured through the estimation of arc travel time distributions.

In most real transportation networks, the congestion states among the arcs are highly correlated. As a result, independent simulation of each arc’s congestion states leads to uncorrelated arc states and might cause incorrect travel time distributions. To avoid such problems, we simulated the network with historical data one day at a time. Specifically, we routed the vehicle from origin site to the destination site; at each decision epoch (e.g. node), the historic arc speed data was used to identify the congestion state and determine which arc to traverse next. We ran the simulations for 66 weekdays of May, June, and July 2009 and obtained 66 samples for all pairs of sites at different departure times. Although the number of runs was small, we believe it captured the dependency of arc congestion states better and that it accurately predicts the routing scenario’s outcome. In addition, due to weather patterns/seasonality, traffic dynamics do change over extended periods. Hence, it is generally inappropriate to use data from extended periods (e.g., a year) to establish the tours and the dynamic routing policies. For these reasons, it might be best to re-optimize the tour and the dynamic routing policies at regular intervals (e.g., monthly or quarterly).

4.3. Building STD-TSP tours

In this section, we construct the robust STD-TSP tours using the effective travel time distribution resulting from dynamic routing between every pair of sites (as explained in Section 4.2). To quantify the benefits of using a dynamic routing policy, we also identify and select the robust STD-TSP tours with a static routing policy between each pair of sites.

In milk-run tours, the number of tour stops in urban areas is generally 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). Our case study application also conforms to these estimates as there are 5 stops (i.e., one DC and four supplier sites). Although there are hundreds of suppliers replenishing the same DC, we only consider the subset of suppliers that were part of the same TSP tour. The determination of such supplier clusters is beyond the scope of this study and is assumed to be performed a priori based various factors (e.g., geographical supplier locations, nature of cargo) as in common frequency routing. There were no pre-established requirements on the sequence of site visits and the truck had enough capacity to visit all sites in a single tour. As in most JIT environments, the time windows in this case study were set by trucking and OEM’s logistics division and accepted by the suppliers as part of the sourcing contract. Therefore, we herein consider the case without time windows and then set the time windows for on-time performance in Section 4.4.

In the STD-TSP of the case study application, we have node 80 as the DC (origin site) and nodes 61, 103, 51, and 132 as the supplier sites (Fig. 1). Accordingly, there are \((5-1)! = 24\) possible dominated and non-dominated tours. To capture the effect of traffic congestion, we consider 48 trip start times evenly spaced every half an hour and determine tours for each of them separately (Fig. 4). We assume all the sites’ service times are 15 min. Since there are 4 sites other than the DC, the total service time is 60 min for each trip. To compare the results we define STD-TSP tours with the following two site-to-site routing policies:

![Fig. 3. For arc (7, 8) (a) congestion state-transition probabilities: \( \alpha \), congested to congested transition; \( \beta \), uncongested to uncongested transition probability (b) mean travel time(min) for congested and uncongested congestion states.](image-url)
1. **STD-TSP tour with static routing policies (Static policy):** In practice, almost all commercial logistics software aims to identify TSP tours based on a static path between a pair of sites. First, for a given site pair and departure time, all paths are identified and then their expected path travel times are calculated according to the travel time distributions of paths’ arcs. Next, the path with the least expected cost is selected as the static path to be used in the TSP tour. Then, for every trip start time, we select a robust TSP tour by solving STD-TSP using travel time distributions between pairs of sites estimated through the static paths.

2. **STD-TSP tour with dynamic routing policies (Dynamic policy):** In this policy, the paths between pairs of customers are dynamic routing policies (DRP). Based on the arc travel time distributions, congestion states and transition probabilities, we first generate DRPs between every pair of sites as described in Section 3.1. Then, these DRPs are simulated to find the site-to-site travel time distributions as described in Section 4.2. Finally, for every trip starting time, the robust TSP tour is selected using the DP algorithm for STD-TSP based on the simulated travel time distributions between pair of sites.

In identifying and selecting the robust tour, we set standard deviation coefficient in the cost function $b = 1.65$ such that the robust tour’s trip duration is less than the mean-variance objective 97.5% of the time. We calculated the mean and standard deviations of trip times for all static and dynamic policy tours for evenly spaced 48 trip starting times beginning at 00:00am. The results revealed that 4 out of the 24 possible tours dominate the other tours for all 48 trip starting times for both static and dynamic policies. These **dominant** tours are:

- **tour 1:** 80 → 132 → 103 → 51 → 61 → 80;
- **tour 2:** 80 → 132 → 51 → 103 → 61 → 80;
- **tour 3:** 80 → 61 → 103 → 51 → 132 → 80;
- **tour 4:** 80 → 61 → 51 → 103 → 132 → 80.

Among these four tours, **tour 1** is the most selected tour by both static (40 out of 48 starting time scenarios) and dynamic (41 out of 48 starting time scenarios) policies. We report **tour 1** mean travel time and standard deviations in Fig. 4 for every starting time during the day. Note that these results are obtained by simulating **tour 1** using the historic data (66 weekdays of May, June, and July 2009).

As expected, Fig. 4 illustrates that the savings with dynamic routing are higher, as well as significant, during the peak traffic times (e.g., around 8:00 and 17:00) and insignificant during the uncongested periods. These results clearly illustrate the importance of using dynamic routing between pairs of sites. To further illustrate the savings, we present the robust tours and their mean and standard deviation of travel times identified by the two policies for two particular departure times in Table 1.

We also report the CPU times in the last column of Table 1 (for two departure times) and in Table 2 (for all 48 departure times). The computational time difference with static and dynamic policies is mainly because of the generation of dynamic routing policies in dynamic scenario. As noted earlier, the vehicle is guided through pre-calculated DRPs when traveling from one site to another site. Thus, once the DRP is optimized offline for a given departure time and set of pickup/delivery stops, the real time vehicle routing on a daily basis is simply looking up from an offline generated DRP lookup table. Clearly, the CPU times effort to prepare these lookup tables could be excessive considering larger network scenarios. For instance, the total number of DRP tables to be calculated for 48 starting time scenarios is $48 \times (N + 1)/N$ considering the depot. While for the current experiments with four sites this corresponds to offline calculation of 960 DRP tables, for a 10 site problem the effort is 5580 DRP tables (or 220 days of uninterrupted running of single core CPU). However, the practice of JIT deliveries in most manufacturing environments, not only require visiting few number of sites but also restricted in terms of trip starting times. Specifically, the number of trip starting times to traverse between each site pair will be less than 48. Lastly, current
cloud-based cluster computing technologies would help alleviate the computational burden and make it practical to implement this offline optimization of DRPs.

4.4. Evaluation of STD-TSP tours with time windows

In the previous section, we selected the robust tours associated with static and dynamic routing policies across 48 starting times. We originally assumed no time windows. In this case study application, the determination of the TSP tour and the setting of time windows are sequential tasks. Specifically, the carrier first determines the tours for transportation efficiency and then the carrier and OEM’s logistics division jointly set the spacing of time windows so as to maximize the on-time delivery performance. Next, we present and compare the trip duration results of using static and dynamic routing policies in a scenario where there are 4 DC replenishment shifts each day and the shift starting times (ST) are ST = {0:00; 6:00; 12:00; 18:00}. We then present the results after setting time windows.

According to the results in the preceding section, tour 1 is the most selected tour by both static and dynamic policies across different trip start times. The other robust tours identified are tours 2, 3, and 4 in decreasing order of selection frequency. In Tables 3 and 4, we provide the mean and standard deviation of trip times (tour travel time + service times) of these four dominant tours and their associated standard deviations at shift starting times when following static and dynamic policies between pair of sites, respectively. These results are obtained by simulating the corresponding tours using the historic data (66 weekdays of May, June, and July 2009).

Table 3 results indicate that the mean tour trip time savings associated with dynamic routing are highest in the two congested start times, namely 6:00 and 18:00, which are close to the urban area peak traffic times. We further note that the savings with start time at 12:00 is also as high as the congested periods (i.e., 6:00 and 18:00). The results in Table 4 for the standard deviation of tour trip times demonstrate the reduction in variability. Eventually, this can be regarded as minimizing the risk of late/early arrival.

The robust tour for each starting time is selected according to the mean-variance objective using the results in Tables 3 and 4. These mean-variance objectives for the four dominant tours are presented in Table 5 along with that of the selected robust tour in the last row. The selected robust tours corresponding to static and dynamic policies are highlighted in bold for each start time. The dynamic policy’s robust tour achieves the most savings over that of the static policy for trips starting at 12:00 and the mean-variance objective savings range from 2.6% to 12.0% with an average of 9.2%. The mean tour trip time savings based on the robust tours range from 1.5% to 11.0% with an average of 8.1% as can be calculated from Table 3. These tour trip duration savings correspond to the improvement in transportation efficiency. Similarly, the savings in the standard deviation of tour trip times based on the robust tours range from 16.5% to 23.7% with an average of 21.6% as can be calculated from Table 4. These savings correspond to the improvement in tour trip time reliability affecting the on-time delivery performance.

Table 5 results indicate that tours 1 and 2 are dominant tours for the four start times. In the remainder of this section, we assume that tour 1 is selected for both static and dynamic policies. In fact, tour 1 is indeed the selected robust tour for start times 6:00 and 12:00 and its performance difference from the selected robust tour is small for starting times of 0:00 and 18:00.

Next, we set the time windows according to the procedure described in Section 3.2. Here, we assume the width of the time windows to be 30 min for all supplier sites. Further, we allow unrestricted waiting for early arrivals at all sites. We illustrate the time windows through their centers (mean site arrival times) and deviations around centers (standard deviation of site arrival times) in Table 6 for the selected robust tour 1.

Table 1
At two different departure times, robust tours comparison for the static and dynamic policies.

<table>
<thead>
<tr>
<th>Policy</th>
<th>Robust tour</th>
<th>Departure time</th>
<th>Mean trip time (min)</th>
<th>Mean tour travel time (min)</th>
<th>Std. Dev. of tour travel time (min)</th>
<th>CPU time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static</td>
<td>Tour 1</td>
<td>7:00</td>
<td>253.8</td>
<td>193.8</td>
<td>13.08</td>
<td>33.2</td>
</tr>
<tr>
<td>Dynamic</td>
<td>Tour 1</td>
<td>7:00</td>
<td>224.5</td>
<td>164.5</td>
<td>10.37</td>
<td>3724.2</td>
</tr>
<tr>
<td>Static</td>
<td>Tour 1</td>
<td>7:30</td>
<td>242.4</td>
<td>182.4</td>
<td>13.27</td>
<td>317.</td>
</tr>
<tr>
<td>Dynamic</td>
<td>Tour 2</td>
<td>7:30</td>
<td>216.1</td>
<td>156.1</td>
<td>10.19</td>
<td>3542.5</td>
</tr>
</tbody>
</table>

Table 2
CPU time statistics for the static and dynamic policies for 48 starting time scenarios.

<table>
<thead>
<tr>
<th>Policy</th>
<th>Min (s)</th>
<th>Max (s)</th>
<th>Average (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static</td>
<td>27.5</td>
<td>43.6</td>
<td>32.4</td>
</tr>
<tr>
<td>Dynamic</td>
<td>3445.6</td>
<td>3936.7</td>
<td>3672.6</td>
</tr>
</tbody>
</table>

The mean and standard deviation of return times to DC (node #80) corresponds to the mean and standard deviation of the tour 1 trip times. Note that the means and standard deviations of DC return times in Table 6 are different than those of tour trip times without time windows reported in Table 3. These differences are due to the waiting at the sites upon early arrival.
The waiting due to early arrival increases (decreases) the mean (standard deviation) of the tour trip time. Table 7 presents the service level performance (on-time delivery) of static and dynamic policies for tour 1 at different start times. These results are based on simulating tour 1 using dynamic and static policies between sites subject to the time windows set for each policy in Table 6. Table 7 results show that as congestion increases, the dynamic policy taking real-time traffic information into account becomes increasingly superior to the static policy planning methods. The on-time delivery performance can be increased up to 8% for a site and up to 4% for a tour (starting at 18:00). We conclude that the dynamic policy not only decreases transportation cost (measured by trip time), but also increases the delivery service level performance (measured by on-time delivery).

The results in Table 7 are obtained with the assumption that there is unrestricted waiting for early arrivals at all sites. Further, the time windows are centered on the mean site arrival times depending on whether static or dynamic routing policy is used between pairs of sites. As explained in Section 3.3, one could shift the time windows to the right of the center (expected site arrival time) to reduce the late arrival occurrences. However, the effectiveness of this modification relies

---

**Table 3**

<table>
<thead>
<tr>
<th></th>
<th>Mean tour trip times</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>00:00</td>
<td>06:00</td>
<td>12:00</td>
<td>18:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tour 1</td>
<td></td>
<td>178.7</td>
<td>174.5</td>
<td>2.4%</td>
<td>238.2</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>177.2</td>
<td>174.0</td>
<td>1.8%</td>
<td>241.6</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>181.2</td>
<td>179.0</td>
<td>1.2%</td>
<td>236.4</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>183.6</td>
<td>181.1</td>
<td>1.4%</td>
<td>248.3</td>
</tr>
</tbody>
</table>

**Table 4**

<table>
<thead>
<tr>
<th></th>
<th>Standard deviation of tour trip times</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>00:00</td>
<td>06:00</td>
<td>12:00</td>
<td>18:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tour 1</td>
<td></td>
<td>7.8</td>
<td>7.0</td>
<td>10.3%</td>
<td>13.0</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>8.3</td>
<td>7.5</td>
<td>9.5%</td>
<td>13.6</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>7.8</td>
<td>7.7</td>
<td>1.0%</td>
<td>14.5</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>9.8</td>
<td>8.6</td>
<td>12.0%</td>
<td>15.2</td>
</tr>
</tbody>
</table>

**Table 5**

<table>
<thead>
<tr>
<th></th>
<th>Mean-variance tour trip time objectives</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>00:00</td>
<td>06:00</td>
<td>12:00</td>
<td>18:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tour 1</td>
<td></td>
<td>191.5</td>
<td>186.0</td>
<td>2.9%</td>
<td>259.7</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>190.9</td>
<td>186.4</td>
<td>2.4%</td>
<td>264.1</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>194.1</td>
<td>191.8</td>
<td>1.2%</td>
<td>260.3</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>190.7</td>
<td>195.3</td>
<td>2.2%</td>
<td>273.4</td>
</tr>
</tbody>
</table>

**Table 6**

<table>
<thead>
<tr>
<th></th>
<th>Simulated mean arrival times (with time windows) to the sites in the sequence of tour 1 based on static and dynamic policies.</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean site arrival times</td>
<td>Std. Dev. of site arrival times</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Site 1</td>
<td>132</td>
<td>18.7</td>
<td>18.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>103</td>
<td>67.3</td>
<td>66.6</td>
<td>87.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>51</td>
<td>98.7</td>
<td>97.9</td>
<td>131.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>61</td>
<td>147.0</td>
<td>143.9</td>
<td>197.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>80</td>
<td>179.2</td>
<td>175.1</td>
<td>240.1</td>
</tr>
</tbody>
</table>

The results in Table 7 are obtained with the assumption that there is unrestricted waiting for early arrivals at all sites. Further, the time windows are centered on the mean site arrival times depending on whether static or dynamic routing policy is used between pairs of sites. As explained in Section 3.3, one could shift the time windows to the right of the center (expected site arrival time) to reduce the late arrival occurrences. However, the effectiveness of this modification relies...
on the maximum allowable waiting time imposed for early arrivals. To understand the effect of shifting time windows, we adapted time windows of the static policy as the time windows of the dynamic policy. This allows us to retain the assumption of unrestricted waiting for early arrivals and compare the on-time delivery results of dynamic policy with those in Table 7. The results of on-time delivery with dynamic policy using the time windows of the static policy are presented in Table 8. With this setting, the on-time delivery performance of the truck following the dynamic policy is 100 percent for all starting times and for all sites based on historic data (66 weekdays of May, June, and July 2009). Clearly, this improvement in on-time performance is attained with increased waiting at sites. Table 8 also presents the mean waiting times at sites.

### 5. Conclusions

In this work, we studied the STD-TSP with dynamic routing problem. It is an extension of the stochastic TSP and aims to find a robust milk-run tour of a given set of sites (i.e., DC and suppliers) while dynamically routing on a stochastic time-dependent road network between sites’ visits to meet the time windows requirements. The solution is comprised of a static TSP tour of sites that remains fixed for extended periods (e.g., months) and a dynamic routing policy between pairs of sites. The static tour is motivated by the fact that tours cannot be changed on a regular basis (e.g., daily) for milk-run pickup and delivery in routine JIT production. The objective trades off the expected duration of the tour with its variability, capturing the tradeoff between transportation efficiency and on-time delivery service level.

We proposed a sequential solution approach. We first determined the travel time distributions between each pair of sites by formulating and solving a stochastic dynamic programming formulation for the dynamic routing problem on a stochastic time-dependent road network. The dynamic routing model exploits the real-time traffic information available from ITS. We proposed effective data driven methods for accurate modeling and estimation of recurrent congestion states and their state transitions. Whereas we assumed arcs are independent in generating dynamic routing policies, we simulated dynamic routing policies using historical data to capture the arc dependencies in all our experiments. Using simulation results, we estimated the site-to-site travel time distributions. Once the travel time distributions were estimated for every pair of sites at different departure times, we employed a stochastic time-dependent dynamic programming method (STD-DP) to solve the problem and selected the robust tour minimizing the mean-variance objective of the trip time. We also provided a time window setting procedure to increase on-time delivery performance and support workload leveling.

We tested the proposed methodology on a real case study application using the road network from Southeast Michigan. This study corresponded to an automotive JIT production system where an OEM’s DC is replenished by milk-run pickup and deliveries from multiple suppliers. The study road network covered major freeways and highways in and around the Detroit metropolitan area. To quantify the benefits of using dynamic policy, we compared the selected robust STD-TSP tours with those of the static routing policy between pair of sites. We first experimented without time windows for both static and dynamic policies. The results showed that the dynamic policy saves 8.1% in trip duration on average and reduces standard deviation of trip duration by 21.6% on average. After setting the time windows according to the expected site arrival times, we showed that the on-time delivery performance can be increased up to 8% for a site and up to 4% for a tour by using dynamic routing policy. Lastly, we showed that it is possible to further increase the on-time performance by setting the time windows of dynamic routing policy according to those of the static policy. We concluded that the dynamic policy not only

<table>
<thead>
<tr>
<th>Table 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-time delivery performances (in percentages) of the policies with time windows.</td>
</tr>
<tr>
<td>ST</td>
</tr>
<tr>
<td>Site 132</td>
</tr>
<tr>
<td>103</td>
</tr>
<tr>
<td>51</td>
</tr>
<tr>
<td>61</td>
</tr>
<tr>
<td>80</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-time delivery performances (in percentages) and average waiting times (in minutes) for dynamic policy when setting time windows of dynamic policy as the time windows of static policy in Table 7.</td>
</tr>
<tr>
<td>ST</td>
</tr>
<tr>
<td>Site 132</td>
</tr>
<tr>
<td>103</td>
</tr>
<tr>
<td>51</td>
</tr>
<tr>
<td>61</td>
</tr>
<tr>
<td>80</td>
</tr>
</tbody>
</table>
There are several promising extensions of this research. The dynamic routing policies are generated by assuming arc independence. While we have partly compensated for this by simulating the policies using actual historical data from the ITS network, the policies themselves are not guaranteed to be optimal if there are significant arc interactions. Hence, a future study is to account for the link interactions in modeling congestion and generating dynamic routing policies. Another future study is to integrate the proposed approach within the more general problem of VRP, where the supplier-route assignment decisions are made in addition to the routing of individual vehicles. This last direction especially opens up further possibilities in terms of accessing and leveraging more high fidelity real-time traffic information in which vehicles dynamically act as collaborative agents exchanging local traffic information. Furthermore, this collaboration would also enable scenarios where vehicles receive auxiliary help from other vehicles and from the distribution center in case of such exceptions as severe non-recurrent incidents. Lastly, our exact approach in optimizing DRPs between pairs of sites could become computationally burdensome. This is especially true for those implementations where the number of sites is large and/or sites to be visited are changing and do not lend themselves for pre-optimization (e.g. subsets from a very large set of sites). For such applications, there are unique research opportunities for designing novel and efficient heuristic procedures to optimize DRPs more effectively. One such contribution could be to develop preprocessing and filtering techniques that would effectively eliminate the consideration of all start-time combinations and/or pairs of sites that would not be traversed in an optimal tour.

Acknowledgments

We are thankful to the editor and three reviewers’ comments and suggestions which dramatically improved the technical clarity and presentation of the paper. This work was supported by funds from the U.S. Department of Transportation and the Michigan Department of Transportation through the MIOH University Transportation Center.

References