EURO J Transp Logist (2013) 2:5–27
DOI 10.1007/s13676-012-0013-x

RESEARCH PAPER

A pickup and delivery problem using crossdocks and truckload lane rates

Kathleen A. Lindsey · Alan L. Erera ·
Martin W. P. Savelsbergh

Received: 29 November 2011 / Accepted: 12 July 2012 / Published online: 6 September 2012

Abstract This paper presents solution approaches for a shipper pickup and delivery planning problem to move freight from suppliers to distribution centers. Each shipment is moved either direct via a less-than-truckload (LTL) carrier, or possibly consolidated with other shipments and moved by one or two truckload (FTL) routes. When using a FTL carrier, the shipper takes advantage of contracted lane rates that establish prices per mile for a truck operated between two locations that are significantly less than the comparable LTL price for shipping a full truckload. The challenge for the shipper is to consolidate multiple shipments effectively to take advantage of this price differential. Consolidated FTL routes may each visit multiple shipment origins (supplier locations) and/or destinations (distribution center locations). Additionally, FTL routes may move shipments through a single crossdock facility en route. A path-based integer programming model for this planning problem is presented. The model can be solved directly with commercial integer programming software for smaller instances. For larger instances, a search scheme is developed in which restrictions to the integer program are solved sequentially. A computational study using data from a major US retailer demonstrates the effectiveness of the solution approaches. In each instance, substantial transportation cost savings are identified from baseline LTL costs.

Keywords Shipment planning · Cross-docking · Full truckload ·
Less-than-truckload · Lane rate · Integer programming

K. A. Lindsey (✉) · A. L. Erera
School of Industrial and Systems Engineering, Georgia Institute of Technology,
765 Ferst Drive, NW, Atlanta, GA 30332-0205, USA
e-mail: kate.abercrombie@gatech.edu

M. W. P. Savelsbergh
School of Mathematical and Physical Sciences, The University of Newcastle,
University Drive, Callaghan, NSW 2308, Australia

Introduction

The US retail industry continues to evolve toward a model where a few large and powerful players sell a growing number of products. In this setting, large retail chains offering products across various market segments, such as grocery, home improvement, and clothing, must transport goods from many suppliers to many retail outlets. To enable low store inventories, outlets are served with frequent shipments from distribution centers, which may span the spectrum from traditional warehouses to cross-docking facilities (crossdocks). Major retailers then have two major freight transportation subproblems: one concerned with moving product from suppliers to distribution centers, and the other with moving product from distribution centers to stores. This research focuses on models for building cost-effective transportation plans inbound to distribution centers.

When individual distribution centers maintain high inventory levels of a product, or when large suppliers deliver many products to a distribution center (DC) simultaneously, inbound transportation may be relatively simple. Full truckload (FTL) or containerload services direct from supplier to DC are economical choices in these cases. In other cases, the simplest options are more expensive. For example, public less-than-truckload (LTL) freight services can be used to move smaller quantities directly, but the rates for such shipments (per weight and per mile) are significantly higher than FTL service.

In this research, we consider a more complex inbound system organized to utilize contracted FTL services inbound to DCs for smaller shipments. To do so, the retailer no longer uses direct supplier-to-DC shipments exclusively. Instead, multiple-stop FTL routes that pick up shipments from one or more suppliers and deliver them to one or more DCs will also be considered. In addition, the retailer may operate additional inbound crossdocks that allow full truckloads to be sent from suppliers to crossdocks, and then from crossdocks to DCs.

When FTL services are used for multiple-stop routes, determining transportation charges is more complex. Retailers negotiate contracts with FTL carriers that specify a transportation charge (or lane rate) for a large number of frequently used origin–destination pairs, or lanes. Each lane rate represents a price per mile for a truck operated on this lane, and is significantly less than the comparable price per mile of shipping a full truckload of LTL freight. In addition to the lane rate, the contract with the FTL carrier will often also allow additional stops between the lane origin and destination for picking up or dropping off freight at additional cost. To discourage excessive extra travel distance in this case, the carrier may specify an upper limit on out-of-route travel either as an absolute bound or as ratio with respect to the original lane distance. If this bound is not violated, the FTL company simply charges the retailer the lane rate multiplied by the total route travel distance including the added stops, along with appropriate stopoff charges. Figure 1 gives an example of adding an additional stop to a lane with a maximum out-of-route travel distance ratio allowance of 1.40, a truckload rate of $1 per mile, and an additional stop charge of $50. The example shows that a single truckload route traveling on path $p =$ (origin, additional drop-off, destination) can replace two LTL shipments

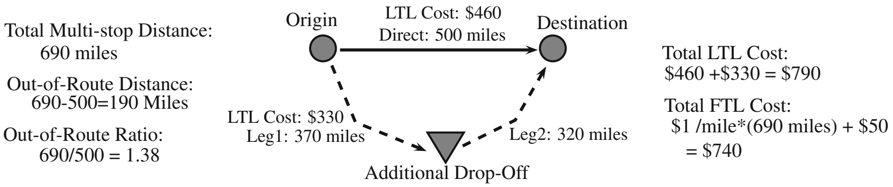

Fig. 1 Lane rate additional stops

traveling between (origin, destination) and (origin, additional drop-off), with total cost savings of $50.

This paper focuses on a pickup and delivery planning problem to move shipments from suppliers to DCs in this setting; see Savelsbergh and Sol (1995) for a survey of the pickup and delivery problem literature. A retailer needs to plan inbound moves serving a set of shipments, where each shipment is served either by an LTL route or by one or two FTL routes. FTL routes may include additional stops between origin and destination, and may originate from or terminate at a crossdock; LTL routes directly connect a shipment's origin to destination. Note that since FTL routes in this context are not cycles, this problem can be considered a type of *open vehicle routing* problem. LTL lane rates exist between each shipment's origin and destination, while FTL rates exist for a subset of network lanes. The problem is to find a feasible set of open routes that minimizes total transportation cost, defined as the sum of the minimum transportation cost for each route in the set. A feasible set of open routes is such that: (1) every shipment is served, traveling via either a single route from origin to destination or a sequence of one route from the origin to a crossdock and one route from a crossdock to the destination; and (2) the shipments simultaneously loaded into any vehicle have total weight and cubic volume (cube) that do not exceed vehicle limits.

We model the problem using a path-based integer programming (IP) formulation, and develop solution approaches tailored to solve problem instances representative of those found in practice using data from a large US retailer. Solving the IP model directly using commercial integer programming software works well on instances with a smaller number of shipments, quickly finding an optimal solution. However, for larger instances, solving the IP model becomes more difficult. In practice, routing decisions need to be made in a restricted period of time; our industry partner suggests that they allocate no more than 4 h of computation time. Since solvers struggle to close the optimality gap for larger instances given this time budget, we also develop a heuristic search scheme in which restrictions to the integer program are solved sequentially. Computational experiments use a number of instances based on data from a large US retailer and compare the performance of both the direct optimization and heuristic approaches. Experimental results demonstrate that the heuristic approach provides high quality solutions for all instances, given the time restriction. Furthermore, in all instances, substantial cost savings are identified using the model when compared against the baseline LTL costs.

The remainder of the paper is organized as follows. The "Literature review" section provides a brief literature review of related research. The "Problem definition and proposed solution approach", "Building a route set", and "Search with restricted route sets” sections formally define the problem and detail the proposed solution approaches. The “Computational results” section provides computational results from real world instances. Finally, the “Conclusions and future research” section provides conclusions and comments on future research.

Literature review

The primary focus herein is to develop approaches for planning transportation for origin–destination less-than-truckload shipments to take advantage of cost economies in truckload transportation and the consolidation opportunities provided by shipping through crossdocks. The existing literature in this area can be grouped roughly into three categories: (1) cross-docking network planning; (2) vehicle routing with cross-docking; and (3) pickup and delivery routing with transfers or transshipments.

Cross-docking network planning problems are primarily service network design problems, and the existing literature does not consider multiple-stop vehicle routing decisions. One example in this category is Chen et al. (2006), who investigated planning shipments with time windows through a network of crossdocks with handling capacities and different handling costs. They showed that the most cost-effective solutions do not always send each shipment through the crossdock that minimizes out-of-route distance (and transportation cost), due to capacities and handling costs. Unlike our research, however, their paper assumed that total transportation cost can be modeled as a linear function of individual shipment costs (similar to LTL costs) and did not explicitly model consolidation in vehicles. Li et al. (2006) addressed a crossdock routing problem for a single product type, where supplies of the product from origins need to be routed to customer demand locations. Suppliers and customers both have time windows defining when shipments must be shipped and received. Inventory may be held at crossdocks, but creates holding cost. Although their paper only considered direct shipments (origin-to-destination, origin-to-crossdock, crossdock-to-destination), constraints were used to count the number of required capacitated vehicles on each transport arc and using an additional vehicle incurs a fixed charge. Sung and Song (2003) addressed a similar problem with two types of vehicles with different operating costs and capacities. While they did not model time windows for pickup or delivery, they enforced a service time limit on the maximum duration (travel plus handling time) for each shipment. Furthermore, the formulation included variables for the selection of which potential crossdocks to open to serve demand. Jayaraman and Ross (2003) also considered a design problem of which crossdock facilities to open to serve demands for a set of products. Like Li et al. (2006), the model decided where to source demand for each product for each customer rather than planning for known shipments. They developed a two phase modeling approach, where the first phase model addressed the design of the network and determines fixed supplier-to-crossdock and crossdock-to-customer assignments. In the second phase, all supplier and crossdock location decisions were fixed, and more flexible product routing is allowed. Transportation costs were assumed to be linear in product flow.

Prior papers have addressed vehicle routing problems where a crossdock is used to serve demand from multiple origins moving to multiple destinations. The specific planning problem considered is often a variation of the so-called *vehicle routing problem with cross-docking* (VRPCD). In this problem, a homogeneous fleet of vehicles based at a crossdock performs a set of pickup tours, bringing shipments from origins into the crossdock. Then, the same fleet performs a set of delivery tours from the crossdock to destinations. All pickups and deliveries must occur within specified time windows, and shipment timing coordination is enforced at the crossdock. Lee et al. (2006) proposed a mixed integer programming formulation and a tabu search algorithm, assuming that all vehicles should simultaneously arrive back at the crossdock after pickup tours. Liao et al. (2010) developed a new tabu search algorithm with minor differences from the approach in Lee et al. (2006) that outperformed the original algorithm on test problems, finding both lower cost solutions and requiring less computation time. Wen et al. (2009) addressed a similar problem, but considered a more flexible coordination constraint at the crossdock that guaranteed that outbound tours cannot be dispatched until all of their freight has arrived inbound to the crossdock. A tabu search heuristic with an embedded adaptive memory procedure found cost-minimizing routes.

Many-to-many pickup and delivery problems that model vehicle consolidation, as extensions of one-to-many or many-to-one multiple vehicle routing problems, have received significant attention in the literature. Pickup and delivery problems with opportunities for freight or passenger transfer between vehicles have been studied less, but have not been ignored. For example, in dial-a-ride passenger transportation one research example is provided by Cortés et al. (2010). The authors extended the classical pickup and delivery problem with an option for passengers to transfer from one vehicle to another at specific locations. A Benders decomposition approach for solving the resultant formulation was proposed, and compared to a straightforward branch-and-bound strategy.

In freight settings, researchers have proposed transshipment extensions to the pickup and delivery problem with time windows. A notable example is the work of Mitrovic-Minic and Laporte (2006). Their paper presented a two-phase heuristic for the problem, and used the approach in an empirical study based on a courier company serving a large geographic region. By utilizing transshipments at fixed locations, loads could be served by two vehicles in sequence. One vehicle picks up the load and takes it to the transshipment location, where the load is transferred to the second vehicle that transports it to the delivery location. Examples with a relatively small number of loads and transshipment points were shown to gain significant benefit from transshipment when origins and destinations were clustered. Mues and Pickl (2005) considered a similar problem, where different transportation modes were available. A column generation approach was proposed for two versions of the problem, one with a single transshipment location and another with multiple transshipment locations. Results were provided for the single location version, and instances with up to 70 loads could be solved quickly.

The research we present in this paper is similar to the cross-docking network planning papers. However, we assume that the available crossdocks are known, and furthermore that known shipments with fixed origins and destinations must be served. Since the shipment quantities are small, we allow planning multiple stop pickup-and-delivery routes to serve these shipments using the best possible truckload rates, potentially using a single crossdock en route as a transfer point. To our knowledge, no existing research presented in the literature has focused on this important planning problem.

Problem definition and proposed solution approach

We now define a pickup and delivery open routing problem with crossdocks where FTL rates can be used on a subset of lanes. Let K be a set of orders that must be moved within a planning horizon. Each order $k \in K$ is to be moved from an origin location o_k to a destination location d_k, and has a weight of $w_k \leq W$ and cubic volume of $q_k \leq Q$ where W and Q are the weight and cube capacity of a truckload. Let J be a set of crossdocks. Assume that a LTL shipping rate of $r^{\mathrm{L}}(o_k, d_k)$ per pound per mile is available for all $k \in K$. Assume that a FTL shipping rate of $r^{\mathrm{T}}(i,j)$ per mile, an extra stop charge of $s^{\mathrm{T}}(i,j)$, and a truckload minimum cost $r^{\mathrm{T}}_{\mathrm{min}}(i,j)$ exist for each (i,j) in some subset A^{T} of all lanes $V \times V$, where $V = J \cup \{\cup_{k \in K} o_k\} \cup \{\cup_{k \in K} d_k\}$. Let M be a maximum out-of-route distance ratio allowed for truckload moves. Let $\ell(p)$ be a known function that returns the travel distance required to travel the sequence of locations p. The problem is then to determine a set of a feasible routes R^* where:

1. each order $k \in K$ is transported from o_k to d_k via a single route, or via two routes connecting at a single crossdock $j \in J$;
2. each route $p = \{o_{(1)}, o_{(2)}, \ldots, o_{(a)}, j\} \in R^*$ destined to a crossdock j visits only order origins, and must not exceed vehicle capacity: $\sum_{i=1}^{a} w_{(i)} \leq W$ and $\sum_{i=1}^{a} q_{(i)} \leq Q$;
3. each route $p = \{j, d_{(b)}, d_{(b-1)}, \ldots, d_{(1)}\} \in R^*$ from a crossdock j only visits order destinations, and must not exceed vehicle capacity: $\sum_{i=1}^{b} w_{(i)} \leq W$ and $\sum_{i=1}^{b} q_{(i)} \leq Q$;
4. each route $p = \{o_{(1)}, \ldots, o_{(a-1)}, o_{(a)}, d_{(a)}, d_{(a-1)}, \ldots, d_{(1)}\} \in R^*$ that does not visit a crossdock will serve one or more orders, visiting all origin locations followed by all destination locations in last-in first-out order, and must not exceed vehicle capacity: $\sum_{i=1}^{a} w_{(i)} \leq W$ and $\sum_{i=1}^{a} q_{(i)} \leq Q$;
5. each route $p_{\mathrm{O}}(j) \in R^*$ outbound from a crossdock j can only serve destinations of orders whose origins are also served on some inbound route $p_{\mathrm{I}}(j) \in R^*$; and
6. the sum $\sum_{p \in R^*} c(p)$ of the individual route transportation costs is minimized.

Each route cost $c(p)$ is defined to be the minimum cost visiting locations $p = \{v_1, v_2, \ldots, v_m\}$ given $r^{\mathrm{L}}, r^{\mathrm{T}}, s^{\mathrm{T}}$, and M; we will show that $c(p)$ can be determined for any p by solving an acyclic minimum cost path problem.

Path-based formulation

We propose solution approaches that rely on path-based binary integer programming models. Each path in the model represents a route of one of the three types:

inbound from order origins to crossdock, outbound from crossdock to order destinations, or pickup visits at origins followed by delivery visits at destinations for a set of orders. Let R_{I} be all capacity-feasible routes inbound to all crossdocks, R_{O} be all capacity-feasible routes outbound from all crossdocks, and R_{D} be all capacity-feasible last-in first-out pickup-and-delivery routes. Let $R = R_{\mathrm{I}} \cup R_{\mathrm{O}} \cup R_{\mathrm{D}}$. Furthermore, for a specific order k let $R_{\mathrm{I}}(k,j)$ be the subset of routes in R_{I} that pickup order k from o_k and deliver it to crossdock j, $R_{\mathrm{O}}(k,j)$ be the subset of routes in R_{O} that depart crossdock j with order k and deliver it to d_k, and $R_{\mathrm{D}}(k)$ be the subset of routes in R_{D} that serve order k by visiting o_k and d_k.

The decision variables are then:

$$x_p = \begin{cases} 1 & \text{if route } p \text{ is selected} \\ 0 & \text{otherwise} \end{cases},$$

and the integer programming model is:

$$\text{minimize} \quad \sum_{p \in R} c(p) x_p \tag{1}$$

$$\text{subject to:} \sum_{p \in R_{\mathrm{D}}(k)} x_p + \sum_{j \in J} \sum_{p \in R_{\mathrm{I}}(k,j)} x_p = 1 \qquad \forall k \in K \tag{2}$$

$$\sum_{p \in R_{\mathrm{O}}(k,j)} x_p = \sum_{p \in R_{\mathrm{I}}(k,j)} x_p \qquad \forall k \in K,\ j \in J \tag{3}$$

$$x_p \in \{0,1\} \quad \forall p \in R \tag{4}$$

The objective function (1) minimizes the total transportation costs by summing the costs of chosen routes. Constraints (2) ensure that each individual order k is served by exactly one pickup route. Constraints (3) further ensure that each order k that is delivered into crossdock j must also be served on an outbound route from j delivering to d_k.

Calculating route costs

The cost to fulfill an order depends on how it is shipped from its origin to its destination. The default, most costly, option is to dispatch it as an LTL shipment, which is modeled as a single-order route $p \in R_{\mathrm{D}}$ from the supplier associated with the order to the distribution center associated with the order. This option is available for all orders.

The goal of the optimization is to identify cheaper cost options for some or all orders by exploiting less costly FTL lane rates, either for an individual order or by combining orders into multi-stop routes (either direct multi-stop routes or indirect multi-stop routes visiting a crossdock).

If a route $p \in R_{\mathrm{D}}$ serves a single order k, its cost is determined by the minimum cost prescribed by either the LTL rate or the FTL rate (if it exists) on lane $p = (o_k, d_k)$, as follows:

$$c(p) = \min\{r^{\mathrm{L}}(p) * w_k * \ell(p), \max\{r^{\mathrm{T}}(p) * \ell(p), r^{\mathrm{T}}_{\min}(p)\}\}, \tag{5}$$

where $r^{\mathrm{T}}_{\min}(p)$ is the minimum truckload charge for lane p.

Determining the cost of a multi-stop route $p \in R$ is more involved. In order for such a route to be feasible, it must be covered by the set of available truckload lane rates. We have the following definition.

Definition 1.1 (*Covered Route*) A route $p = \{i_1, i_2, \ldots, i_m\} \in R$ is covered if there exists a rated path p_r from i_1 to i_m, i.e., a subset of the stops in p (in sequence) such that each $(i_j, i_{j+k}) \in p_r$ has an associated truckload lane rate $r^{\mathrm{T}}(i_j, i_{j+k})$ and the subsequence $\{i_j, \ldots, i_{j+k}\}$ of p does not exceed the maximum out-of-route distance ratio: $\frac{\ell(i_j, i_{j+1}, \ldots, i_{j+k})}{\ell(i_j, i_{j+k})} \leq M$.

A simple example of a covered route p would be one where a truckload rate exists on lane (i_1, i_m), and $\frac{\ell(p)}{\ell(i_1, i_m)} \leq M$. In this case, the cost of route p could be determined by $\max\{r^{\mathrm{T}}(i_1, i_m)\ell(p) + (m-2)s^{\mathrm{T}}(i_1, i_m), r^{\mathrm{T}}_{\min}(i_1, i_m)\}$. Note that for each of the $m-2$ additional stops visited between i_1 and i_m, the lane stop-off charge is incurred. It is possible, however that p might be also covered by an alternative rated path p_r, and using that path may result in a lower cost $c(p)$.

To determine the minimum cost $c(p)$ of any covered route $p \in R$, we use a simple acyclic shortest path problem. A network is constructed with m nodes, labeled $\{i_1, i_2, \ldots, i_m\}$. An arc connects i_j to i_{j+k} for $k \geq 1$ if and only if a truckload lane rate $r^{\mathrm{T}}(i_j, i_{j+k})$ exists and if using this lane rate would not violate the out-of-route distance ratio: $\frac{\ell(i_j, i_{j+1}, \ldots, i_{j+k})}{\ell(i_j, i_{j+k})} \leq M$. The cost of such an arc is then given by:

$$c_{i_j, i_{j+k}} = \max\{r^{\mathrm{T}}(i_j, i_{j+k})\ell(i_j, i_{j+1}, \ldots, i_{j+k}) + (k-1)s^{\mathrm{T}}(i_j, i_{j+k}), r^{\mathrm{T}}_{\min}(i_j, i_{j+k})\}$$

The cost $c(p)$ is determined by the cost of a minimum cost path connecting i_1 to i_m on this acyclic network. Note that the existence of at least one such path is guaranteed, since p is a covered route.

Building a route set

In practice, enumerating the full set of feasible routes R will likely lead to computationally intractable instances of the IP model. Therefore, we develop a pragmatic solution approach that enumerates a large subset of feasible routes $R' \subset R$ that are likely to be part of a high-quality solution.

Given a set of orders $K(p)$ to be served on route p, consider the problem of determining the best sequence of stops. There are three types of routes: (1) direct routes that visit both o_k and d_k for each $k \in K(p)$; (2) inbound crossdock routes that visit o_k for each $k \in K(p)$ before terminating at some crossdock $j \in J$; and (3) outbound crossdock routes that depart crossdock $j \in J$ and then visit d_k for each $k \in K(p)$. For each type of route, a sequential insertion approach is used to construct route p by inserting one order k at a time.

For a direct route of type (1), consider a partial route p' visiting a subset of the locations required for the orders in $K(p)$, *i.e.*, $p' = \{o_1, o_2, \ldots, o_m, d_m, d_{m-1}, \ldots, d_1\}$.

When inserting order k into p', locations o_k and d_k are inserted simultaneously with o_k immediately prior to o_1 or immediately after $o_1, o_2, \ldots, o_m$, which then fixes the position of d_k immediately after d_1 or immediately prior to $d_1, d_2, \ldots, d_m$ to maintain the LIFO ordering. For routes of type (2) into crossdock j, a partial route will have the form $p' = \{o_1, o_2, \ldots, o_m, j\}$ and inserting order k requires finding an insertion position for o_k immediately prior to any location in p'. Similarly, for routes of type (3) from crossdock j, inserting order k into $p' = \{j, d_1, d_2, \ldots, d_m\}$ requires finding an insertion position for d_k immediately after any location in p'.

Given an order k and a partial path p', all feasible insertion positions are first evaluated to find a minimum cost feasible insertion position. An insertion position is considered feasible if the new route p'' after the insertion of order k is covered. For each such feasible position, the new route cost $c(p'')$ is determined using the approach in the "Calculating route costs" section, and the cheapest position is selected. Note that if path p'' visits the same location multiple times in sequence (for example, if two or more orders are to be picked up at the same supplier location), these visits are merged into a single visit to the location before costing. There may not exist any feasible insertion position for order k in p'. If order k is not the last order to be added to p', we insert it at the position that minimizes the increase in total route distance from p' to p'' in the hope that a covered route will result after inserting additional orders. Each covered final route p is included in R' with cost $c(p)$.

Computing $c(p'')$ exactly for each insertion position is somewhat more expensive computationally than traditional insertion cost calculations. However, computational experiments (see the "Computational results" section for details) demonstrate the benefit of this strategy over a simpler one where best insertion positions are determined by minimizing the increase in the total distance of p'' over p'.

Next, we describe the strategies used to generate sets of orders to be served together on a route. To ensure that a feasible solution to the IP exists, we first generate a single-order direct route for each $k \in K$ of the form $p = (o_k, d_k)$, with a cost $c(p)$ given by (5).

To generate sets of orders to be served together on a route, a two-step procedure is used. In the first step, we create order lists that contain a reasonably large number of orders. In the second step, each of the order lists is partitioned into sets of orders to be served together on a route using a first-fit bin packing procedure. The bin-packing procedure works as follows. Let O be the order list. Orders are removed from O in sequence and placed into bins, where bin i contains the orders $K(p_i)$ for route p_i. Order $k \in O$ is placed in the first available bin p_i with $w_k \leq W - \sum_{k' \in K(p_i)} w_{k'}$ and $q_k \leq Q - \sum_{k' \in K(p_i)} q_{k'}$.

To create the order lists, we group orders by the geographic proximity of order origins and destinations. Let $L(S)$ and $L(D)$ be the sets of physical locations representing suppliers (order origins) and distribution centers (order destinations), respectively. Then, given a distance ρ, let the neighbor list $N(f, \rho)$ for each facility $f \in L(S) \cup L(D)$ be the facilities $f' \in L(S) \cup L(D)$ no further than ρ miles from f. By only generating routes that serve orders whose facilities are nearby geographically, we hope to generate low cost routes. See the "Computational results" section for a brief discussion on how ρ was chosen in this study.

Order lists for direct routes We start by creating two order lists $O_1(i)$ and $O_2(i)$ for each location $i \in L(S) \cup L(D)$. If i is a supplier, $O_1(i)$ and $O_2(i)$ contain the orders $k \in K$ with $o_k = i$. The difference between the order lists is the way in which the orders are sorted. In $O_1(i)$ the orders are sorted in order of non-increasing weight w_k. In $O_2(i)$ the orders are sorted in order of non-increasing angle θ_k of the destination location d_k when the location is represented in polar coordinates. (When there are orders with the same destination location, the orders are further sorted in non-increasing order of weight w_k.) Next, we create two order lists for pairs of suppliers i and i' such that $i' \in N(i, \rho)$ [and thus $i \in N(i', \rho)$]. Each of these two lists contains the orders $k \in K$ with $o_k = i$ or $o_k = i'$. Again, $O_1(i, i')$ is sorted in non-increasing order of weight w_k and $O_2(i, i')$ is sorted first by the angle θ_k of destination d_k and then by non-increasing order by weight w_k. The process is repeated for distribution centers. The creation of lists focused on order weights and lists focused on geographic locations provides diversification.

Order lists for crossdock routes We limit τ, the number of crossdocks, that we consider for a particular order. Therefore, we start by creating crossdock order lists $K_j(\tau)$ for $j \in J$. For each order $k \in K$, we compute the detour travel distance incurred when order k is routed through crossdock $j \in J$, i.e., $\ell(o_k, j, d_k) - \ell(o_k, d_k)$. Order k is then added to crossdock order list $K_j(\tau)$ for the τ lowest detour travel distance crossdocks j.

We use the crossdock order lists to construct the order lists that will be input to the bin packing procedure. For each crossdock $j \in J$, we first build inbound order lists $O_1(i) \subseteq K_j(\tau)$ for each unique supplier location i. As before, the orders in $O_1(i)$ are sorted in order of non-increasing weight w_k. Then, we build multiple-supplier inbound order lists by adding the orders in $K_j(\tau)$ for additional suppliers $i' \in N(i, \rho)$ one supplier at a time. For example, lists $O_1(i, i')$ and $O_1(i, i', i'')$ would be created if both i' and i'' are in $N(i, \rho)$. Orders for additional suppliers are added until a maximum number of suppliers is reached.

A similar process is used to create outbound order lists from each $j \in J$ to distribution centers. An outbound order list $O_1(i) \subseteq K_j(\tau)$ is created for each unique distribution center location i, and then additional order lists are created by adding orders bound for neighboring distribution centers $i' \in N(i, \rho)$.

Search with restricted route sets

As instances become larger, i.e., as the numbers of orders and possible routes grow, solving the integer program defined in the "Path-based formulation" section becomes increasingly difficult. To ensure that high-quality solutions are found reliably given limited available computation time, we have developed a search scheme that solves a sequence of smaller integer programs, each obtained by restricting the set of variables considered. This approach ensures that we maintain the two essential characteristics of our solution approach: (1) using heuristics focusing on local considerations to build routes, and (2) using an integer program focusing on global considerations to select the most appropriate routes. Search schemes based on restricting the set of variables considered in an integer program have become quite popular and have proven successful, see e.g., De Franceschi et al. (2006) and Hewitt et al. (2009).

The restricted integer programs always include variables corresponding to the routes in the best known feasible solution. Thus, for each shipment, the integer program chooses between the route(s) that serve that shipment in the best known solution, and the alternative routes available to serve that shipment. Since even the restricted integer program may be difficult to solve, a computation time limit is imposed. The success of the search scheme depends on choosing restrictions R'' of the route set R' that result in improved feasible solutions quickly. (It is not important to prove optimality of an improved solution to the restricted problem solved.)

Our approach for restricting the route set R' is to focus on different types of problem locations each iteration. More specifically, we build the following restricted route sets R'':

1. *Distribution Center Restriction* Given a single distribution center $f \in L(D)$, let $K' \subset K$ be the set of orders that have f as the destination. Next, let $K'' \subset K$ be the set of orders for which there exists at least one route in R' that includes a visit to either o_k or d_k for an order $k \in K'$; the orders in K'' can be served on routes with orders in K'. The restricted set R'' contains all routes that serve *only* shipments in $K' \cup K''$.
2. *Crossdock Restriction* Given a crossdock $j \in J$, let $K' \subset K$ be the set of orders for which there exists at least one route that ends and one route that starts at j. The restricted set R'' contains all routes that end or start at j and serve only orders in K'.
3. *All Crossdocks Restriction* Let $K' \subseteq K$ be the set of orders for which there exists at least one route that ends and one route that starts at some crossdock $j \in J$. The restricted set R'' contains all routes that end and start at any crossdock and serve only orders in K'.
4. *No Crossdock Restriction* Let $K' \subseteq K$ be the set of orders for which there exists at least one route that does not include a visit to a crossdock. The restricted set R'' contains all routes that do not visit a crossdock and that serve only orders in K'.
5. *Suppliers Restriction* For this restriction, we consider the set of all orders K. The restricted set R'' contains all routes in R' that include a visit to any shipment origin o_k; the only routes not included here are those that begin at a crossdock.

The restricted route sets that we use represent a balance between relatively small ones, such as the *Distribution Center Restriction*, and relatively large ones, such as the *All Crossdocks Restrictions* or the *Suppliers Restriction*. The overall search heuristic follows the scheme presented in Algorithm 1. The time limit T for the main part of the scheme is set to 3.5 h. To ensure that with high probability each *Distribution Center Restriction* is solved at least once, the time limit for these restrictions is set based on the number of DCs; the maximum time allotted to each is $\frac{T}{|L(D)|}$. All other restricted IPs are allotted 10 minutes of solve time, with one exception. The final full IP is solved with a time limit of 20 min. It has proven

Algorithm 1: IP-Based Neighborhood Search Scheme

```
Construct initial feasible solution;
while the search time has not exceeded time limit $T$ do
    for every distribution center do
        Solve Distribution Center Restriction;
    end
    Solve Suppliers Restriction;
    Solve All Crossdocks Restriction;
    if All Crossdocks Restriction does not solve to optimality then
        for every crossdock do
            Solve Crossdock Restriction;
        end
    end
    Solve No Crossdock Restriction;
    Solve IP;
end
Solve No Crossdock Restriction;
Solve IP;
```

valuable to include the solution of the full IP at the end, once a good solution and its corresponding upper bound is known, since it provides a global view of the problem and is sometimes able to find additional improvements.

An initial feasible solution is found by first solving the linear relaxation of the full IP, fixing any route variable with a value less than 0.001 to a value of zero, and then solving the resulting IP for 10 min or until a feasible solution has been found; in practice, this approach always identifies a feasible solution within 10 min of solve time.

Computational results

The solution approach described above is implemented in C++ using ILOG Concert Technology with ILOG CPLEX 12.2 as the solver; when solving IPs all parameters are set to default values, except that multithreading was disabled. The experiments were run on a cluster of workstations with clock speeds between 2.0 and 3.5 MHz and 1.5 Gb of memory. Table 1 presents the characteristics for the instances used in our computational study. The first three instances are based on real-life data from a large US retailer, and represent sets of shipments to be transported in an operating period of about a week. The remaining instances have been generated using the suppliers, distribution centers, crossdocks (CDs), shipments, and lanes with FTL rates from the first three instances.

The generated instances were created, in part, to investigate the impact, if any, of instance characteristics on the difficulty of constructing high-quality solutions and the structure of high-quality solutions. The generated instances have different spatial characteristics, different ratios of number of suppliers to number of distribution centers, and different in- and outbound shipment patterns. More specifically, in Instance 5 all facilities are in a small geographic region, in Instance 6

Table 1 Instance characteristics

Instance	No. of suppliers	No. of distribution centers	No. of CDs	No. of shipments	No. of FTL lane rates	Max ΔCost bound (%)
Instance 1	4	40	14	128	1,389	67.56
Instance 2	98	46	14	507	7,864	78.50
Instance 3	33	40	14	1,126	3,073	78.63
Instance 4	73	15	14	503	4,082	77.89
Instance 5	53	49	8	389	2,678	81.78
Instance 6	89	38	10	612	5,772	80.56
Instance 7	44	48	10	254	2,736	72.92
Instance 8	41	61	10	615	3,914	75.83
Instance 9	85	59	10	885	7,270	75.97
Instance 10	80	35	10	1,120	4,929	78.42
Instance 11	29	41	10	1,125	2,275	77.73

all facilities are within a 200 mile radius of a CD, and in Instance 7 all facilities are more than 200 miles away from a CD. The other generated instances have spatial characteristics similar to the three instances based on real-life data from a large US retailer, but varying ratios of number of suppliers to number of distribution centers, and varying numbers of in- and outbound shipments at facilities (see Table 1).

The direct distance for a shipment, i.e., from the supplier at its origin to the distribution center at its destination ranges from 13 miles to 3,450 miles, with an average distance of 1,413 miles. The average per trailer per mile FTL lane rate, over the origin destination pairs for which a LTL lane rate is specified, is $13.21, and the average per trailer per mile FTL lane rate, over the origin–destination pairs corresponding to the shipments, is $1.43.

A maximum out-of-route distance ratio of $M = 2.0$ is used and the charge for including an additional stop between i and j is set to $s^{\mathrm{T}}(i,j) = \$50$, for all lanes $(i,j) \in A^{\mathrm{T}}$. When generating routes, we restrict the number of unique supplier (origin) locations and the number of unique distribution center (destination) locations visited by any single route; in this study, no more than five locations of each type can be visited by any route. The parameter settings for the maximum out-of-route distance, stop-off charge, and number of unique stops for each location type are the ones used by a large US retailer. Furthermore, a neighbor radius of $\rho = 1,000$ miles is used. Initial experimentation showed that a larger neighbor radius consistently tended to produce greater savings. Therefore, the neighbor radius is set to a reasonably large value that is acceptable in practice.

For all experiments, we use cost savings (ΔCost) as the primary performance measure, where cost savings are measured in percentages relative to a baseline cost computed when each order is moved by LTL transportation. The value of a 1 % cost savings is different for each instance since the number of shipments can vary significantly between instances. For the instances evaluated in this study, 1 % cost savings can range from $1,000 to $16,000.

Since we do not enumerate all feasible routes, we also compute a simple upper bound on the possible cost savings for an instance that is independent of the set of routes generated. The bound determines the minimum possible cost incurred for moving each shipment using LTL or FTL transportation. Utilizing the lane rates and shipment information, the minimum possible FTL transportation cost can be defined given that a shipment served by FTL transportation must travel at least the distance between the origin and destination. Consequently, a lower bound on the transportation cost given $p = (o_k, d_k)$ is:

$$LB = \sum_{k \in K} \min\left\{ r^{\mathrm{L}}(p) * w_k * \ell(p), r^{\mathrm{T}}(p_{min}) * \frac{w_k}{W} * \ell(p) \right\},$$

where $r^{\mathrm{L}}(p) * w_k * \ell(p)$ is LTL shipment cost and $r^{\mathrm{T}}(p_{min}) * \frac{w_k}{W} * \ell(p)$ represents the minimum possible cost utilizing FTL transportation for shipment k. For the FTL transportation cost, $\frac{w_k}{W}$ represents the fraction of vehicle capacity that shipment k requires, $\ell(p)$ is the distance between the origin and destination of shipment k, and $r^{\mathrm{T}}(p_{min})$ is the minimum lane rate that shipment k can feasibly use. Shipment k can feasibly use a lane rate for (i,j) if the length of route $p = \{i, o_k, d_k, j\}$ is less than M times the distance between i and j. An upper bound on the maximum cost savings follows by subtracting this lower bound from the baseline LTL transportation cost. The maximum cost savings bound for each of the instances is reported in the last column of Table 1.

Route generation

We first assess the importance of using transportation cost-based insertion as opposed to distance-based insertion (see the "Building a route set" section). To do so, we conducted the following computational experiment. For each instance, we set $\tau = 1$ to limit each shipment to crossdock routes using only the crossdock that minimizes the out-of-route distance. We then generated two integer programs, one in which routes were generated using distance-based insertion and one where the routes were generated using transportation cost-based insertions. The final cost of each generated route for each IP is the true transportation cost computed via the approach in the "Calculating route costs" section, independent of the insertion approach. Each IP was solved with a 4 h time limit, and the resulting cost savings are presented in Table 2.

As expected, the transportation cost-based insertion procedure requires more computation time than distance-based insertions; the average time for route generation increased from 1.66 to 5.48 min. However, the additional time spent on route generation is justified by the increased cost savings, as illustrated clearly by the results in Table 2. Therefore, in all other experiments the lane rate cost information will be used when determining insertion positions during route generation.

Table 2 Comparing insertion techniques

Instance	No. of crossdocks (τ)	ΔCost	
		Trans. cost insertion (%)	Distance insertion (%)
Instance 1	1	46.11	45.34
Instance 2	1	55.14	50.60
Instance 3	1	64.79	60.95
Instance 4	1	64.79	61.19
Instance 5	1	54.78	52.81
Instance 6	1	66.74	63.23
Instance 7	1	49.02	45.75
Instance 8	1	52.02	47.61
Instance 9	1	50.99	46.42
Instance 10	1	63.14	59.40
Instance 11	1	61.28	58.20

Shipment plan construction

The size of the integer program and the quality of a shipment plan are impacted by the number of crossdocks considered for each shipment. When this number increases, consolidation opportunities increase as well which may result in lower costs. In our first experiment, we test how solutions change as we increase the number of feasible crossdocks per shipment. Instance 1 has only a small number of shipments and the full IP with the entire route set R' can easily be solved, therefore it is an ideal candidate for this experiment. Since there are 14 CDs in Instance 1, the instance was solved 14 times, each time allowing one more crossdock to be considered for each shipment (in non-decreasing order of origin–destination distance via the crossdock). Each of the 14 resulting integer programs is solved to optimality in less than 30 min. Results summarizing the characteristics of the problems and optimal solutions are presented in Table 3. The table reports the number of crossdocks considered for a shipment (τ), the number of pickup-and-delivery routes R'_{D} generated, the number of crossdock inbound routes R'_{I} generated, the total number of crossdock outbound routes R'_{O} generated, the total number of routes generated, ΔCost, the percentage of shipments using LTL routes (LTL), FTL direct routes (FTL-D), and FTL routes visiting crossdocks (FTL-CD), the number of CDs used in the solution, and the time to solve the IP.

As expected, the cost savings increase as the number of crossdocks considered for an order increases; from 46.1 % when only a single crossdock is considered for an order to 51.7 % when nine crossdocks are considered for an order (the maximum possible cost savings for this instance is bounded from above by 67.6 %). The increased cost savings arise because the percentage of orders consolidated and shipped via a crossdock increases with the number of crossdocks considered for an order. Observe that this sometimes results in an increase in orders that are shipped using an LTL route (e.g., when the number of crossdocks considered for a shipment

Table 3 Impact of the number of crossdocks considered for a shipment (Instance 1)

τ	No. of routes created				ΔCost (%)	Shipment breakdowns by %			No. CDs used	Time (s)
	R'_{D}	R'_{I}	R'_{O}	Total		LTL	FTL-D	FTL-CD		
1	808	193	696	1,697	46.1	71.1	27.3	1.6	1	0.07
2	808	434	3,132	4,374	48.6	73.4	17.2	9.4	2	0.32
3	808	697	6,795	8,300	49.1	73.4	14.8	11.7	2	1.17
4	808	983	11,394	13,185	49.2	68.0	12.5	19.5	3	2.45
5	808	1,300	18,655	20,763	50.5	58.6	5.5	35.9	3	5.71
6	808	1,626	27,180	29,614	50.7	57.8	6.3	35.9	3	14.65
7	808	1,972	36,049	38,829	51.2	58.6	5.5	35.9	2	35.73
8	808	2,324	44,983	48,115	51.2	58.6	5.5	35.9	2	27.02
9	808	2,693	56,494	59,995	51.7	57.0	6.3	36.7	4	29.87
10	808	3,042	68,754	72,604	51.7	57.0	6.3	36.7	4	34.55
11	808	3,387	81,253	85,448	51.7	57.0	6.3	36.7	4	114.68
12	808	3,740	95,479	100,027	51.7	57.0	6.3	36.7	4	133.24
13	808	4,065	110,056	114,929	51.7	57.0	6.3	36.7	4	483.30
14	808	4,280	118,876	123,964	51.7	57.0	6.3	36.7	4	1,475.74

is increased from 1 to 2). This happens when orders that were previously shipped together on a direct FTL route are no longer shipped together, because shipping some, but not all, via a crossdock creates better cost-saving consolidation opportunities. A similar phenomenon occurs when the number of crossdocks considered for a shipment increases from 6 to 7 and the number of crossdocks used in the optimal solution decreases. Because of the increase in the number of crossdocks considered for a shipment, one of the crossdocks is now a feasible option for many more orders and offers substantially better cost-saving consolidation opportunities. All of the shipments that were previously consolidated at another crossdock are now moved through this new crossdock, thus decreasing the total number of crossdocks used by one.

To see if relaxing the default parameter settings allows even larger cost savings, we performed an additional experiment using Instance 1 in which the number of crossdocks considered for each shipment is set to 14 and the neighbor radius is set to $\rho = 4{,}000$ miles, which essentially removes all restrictions for this instance. The integer program solved in 69 min with a cost savings of 52.4 %, a minor improvement. This result hints at the relative weakness of the maximum cost savings bound. It is also important to note that this result depends on the full set of routes R' produced by our route generation strategy, and that some additional cost savings may be found by considering additional feasible routes.

Instances 2 through 11 have a larger number of shipments and locations, and therefore many more routes are generated. Our second experiment focuses on finding high quality solutions to such larger instances. For these, it is rarely possible to solve the full IP using the entire set of routes R' in less than 4 h (the imposed time

Table 4 Instance results

						Full IP							Heuristic search with restricted IPs					
Inst.	τ	No. of routes created						Shipment breakdowns by %			No. CDs used	Time (h)		Shipment breakdowns by %			No. CDs used	Time (h)
		R'_{D}	R'_{I}	R'_{O}	Total	*ΔCost* (%)	Opt Gap (%)	LTL	FTL-D	FTL-CD			Δ*Cost* (%)	LTL	FTL-D	FTL-CD		
2	1	6055	6778	3077	15910	55.1	0.01	52.7	34.3	13.0	6	0.005	55.1	52.7	34.1	13.2	7	0.03
2	2	6055	24269	11103	41427	56.4	0.96	48.1	28.2	23.7	8	4.00	56.5	47.5	27.6	24.9	8	4.00
2	3	6055	57087	25040	88182	57.0	5.00	33.1	17.6	49.3	8	4.00	57.7	44.4	20.9	34.7	6	4.00
3	1	20778	13930	21736	56444	64.8	2.60	20.3	49.5	30.2	9	4.00	65.3	21.0	46.4	32.6	10	4.00
3	2	20778	43111	99518	163407	62.2	14.05	15.5	29.0	55.5	10	4.00	65.7	21.8	46.1	32.1	6	4.00
3	3	20778	88911	234269	343958	30.0	55.42	73.7	26.3	0.0	0	4.00	66.2	20.2	38.1	41.7	7	4.00
4	1	6579	7092	2028	15699	64.8	0.01	25.2	48.7	26.0	9	0.03	64.8	25.2	48.7	26.0	9	0.05
4	2	6579	22735	7170	36484	66.5	0.56	26.2	34.0	39.8	9	4.00	66.5	24.9	35.4	39.8	10	4.00
4	3	6579	46903	15455	68937	66.3	3.89	17.9	25.6	56.5	8	4.00	66.3	16.3	21.3	62.4	6	4.00
5	1	6205	3923	5263	15391	54.8	0.00	35.7	56.6	7.7	4	0.00	54.8	35.7	56.6	7.7	4	0.01
5	2	6205	15474	24482	46161	55.2	1.30	29.3	43.4	27.2	4	4.00	55.3	31.1	47.6	21.3	5	4.00
5	3	6205	34430	64540	105175	55.1	5.40	15.4	33.7	50.9	4	4.00	56.2	18.8	42.2	39.1	4	4.00
6	1	13530	12767	7117	33414	66.7	2.12	20.8	44.6	34.6	8	4.00	66.9	22.2	46.9	30.9	6	4.00
6	2	13530	41587	39577	94694	61.3	21.09	35.5	63.6	1.0	3	4.00	68.0	22.7	35.9	41.3	5	4.00
6	3	13530	91077	102901	207508	67.3	9.92	12.4	17.5	70.1	5	4.00	68.2	23.9	36.9	39.2	5	4.00
7	1	1159	916	1560	3635	49.0	0.00	48.8	37.4	13.8	6	0.0001	49.0	48.8	37.4	13.8	6	0.001
7	2	1159	2705	5491	9355	51.1	0.01	47.6	36.2	16.1	6	0.002	51.1	47.6	36.2	16.1	6	0.01
7	3	1159	5615	12258	19032	51.8	0.01	42.9	24.4	32.7	7	1.29	51.8	42.9	24.4	32.7	7	4.00
8	1	7981	8420	8765	25166	52.0	0.96	44.2	31.5	24.2	5	4.00	52.0	42.1	33.8	24.1	4	4.00

Table 4 continued

						Full IP							Heuristic search with restricted IPs					
Inst.	τ	No. of routes created						Shipment breakdowns by %			No. CDs used	Time (h)		Shipment breakdowns by %			No. CDs used	Time (h)
		R'_{D}	R'_{I}	R'_{O}	Total	$\Delta Cost$ (%)	Opt Gap (%)	LTL	FTL-D	FTL-CD			$\Delta Cost$ (%)	LTL	FTL-D	FTL-CD		
8	2	7981	28016	45954	81951	52.2	8.14	32.5	17.2	50.2	5	4.00	54.2	38.2	24.1	37.7	4	4.00
8	3	7981	62187	114736	184904	52.4	13.27	25.9	6.0	68.1	4	4.00	55.0	28.3	9.8	62.0	5	4.00
9	1	15235	17667	19597	52499	51.0	2.49	39.7	26.8	33.6	9	4.00	51.4	42.6	26.8	30.6	9	4.00
9	2	15235	69478	109479	194192	43.2	21.99	54.0	44.1	1.9	3	4.00	51.0	46.9	35.4	17.7	5	4.00
9	3	15235	158058	287729	461022	39.3	31.95	54.6	44.0	1.5	3	4.00	52.3	38.4	20.0	41.6	5	4.00
10	1	21457	20905	15193	57555	63.1	4.18	23.1	43.4	33.5	9	4.00	64.1	23.7	46.4	29.9	9	4.00
10	2	21457	72493	82750	176700	52.4	29.98	31.7	45.0	23.3	8	4.00	64.4	28.9	48.4	22.7	5	4.00
10	3	21457	162163	206836	390456	50.6	34.43	33.0	37.3	29.6	10	4.00	64.8	28.7	46.3	25.1	4	4.00
11	1	19699	13121	20322	53142	61.3	6.07	22.7	41.7	35.6	9	4.00	62.9	23.4	41.9	34.8	7	4.00
11	2	19699	40557	103014	163270	49.3	32.10	32.4	46.0	21.6	7	4.00	63.5	25.2	44.0	30.8	5	4.00
11	3	19699	83916	261174	364789	28.9	52.95	77.6	22.4	0.0	0	4.00	63.5	24.5	41.7	33.8	6	4.00

limit). The heuristic search scheme using restricted route sets was developed for such larger instances, and its performance is analyzed in this experiment. Table 4 compares results obtained by solving these instances using the full IP and using the heuristic search scheme, where each approach is limited to 4 h of computation time. The table contains the following columns: instance number, τ (the number of crossdocks), the number of pickup-and-delivery routes R'_D generated, the number of crossdock inbound routes R'_I generated, the number of crossdock outbound routes R'_O generated, the total number of routes generated, and information pertaining to the best solution produced by the full IP and the best solution produced by the heuristic search scheme. For each of the solutions, there is a column for each of the following quantities: ΔCost, the percentage of orders shipped using LTL routes, FTL direct routes (FTL-D), and FTL routes visiting crossdocks (FTL-CD), the number of CDs used, and the solution time. The final optimality gap is also reported for the full IP approach. Recall that the maximum cost savings bound for these instances is reported in Table 1.

The results demonstrate the robustness of the heuristic search scheme. For all instances and for all considered values of τ, it finds a high-quality solution with either the same or larger cost-savings than the solution produced when solving the full IP (in the time limit of 4 h). When the total number of routes in an instance is more than 150,000, solving the full IP can no longer reliably produce high-quality within the time limit (see e.g., Instance 3 with $\tau = 3$, Instance 9 with $\tau = 2$ and $\tau = 3$, and Instance 11 with $\tau = 2$ and $\tau = 3$).

Similar to what we have seen for Instance 1, when the number of crossdocks considered for an order is increased the cost savings and the percentage of orders shipped via a crossdock almost always increases. In contrast to Instance 1, however, we see that the increased density of shipments and locations in these instances leads to solutions when $\tau = 1$ with large fractions of shipments moving via truckload routes that do not involve a crossdock (column FTL-D). The existence of good consolidation routes without crossdocks in part mitigates the cost savings that results from increasing τ.

When τ is increased, the increase in the percentage of orders shipped via a crossdock is especially noticeable for Instance 4 and Instance 8, where for $\tau = 3$ we see that 62.4 and 62.0 % of orders are shipped via a crossdock, respectively. In Instance 4 there are relatively few distribution centers and each of them receives many shipments (each receiving about 7 % of the shipments). As a result, there are many consolidation opportunities with $\tau = 3$, especially from a crossdock to a distribution center. The situation is almost completely opposite in Instance 8, because there are relatively few suppliers and all of them dispatch many shipments (all suppliers send about 2.5 % of the shipments). As a result, there are many consolidation opportunities, especially from supplier to crossdock.

It is also interesting to note that in the solution for $\tau = 3$ for each of the instances only about half of the available crossdocks are used. This demonstrates the important impact on total freight transportation costs that results from choosing the right number and location of crossdock facilities.

Table 5 Updated bounds for increased LTL costs

Instance	Max cost savings bound (%)
Instance 1	78.26
Instance 2	85.57
Instance 3	85.76
Instance 4	85.22
Instance 5	87.85
Instance 6	87.04
Instance 7	81.92
Instance 8	83.89
Instance 9	83.95
Instance 10	85.61
Instance 11	85.15

Rate differences

Since the goal of the optimization is to reduce freight transportation costs by exploiting FTL lane rates, it is clear that the difference between the LTL and FTL lane rate impacts the possible cost savings and the solution.

In our final experiment, we explore this impact by increasing the cost of LTL transportation for each shipment by 50 %, thus making FTL transportation more attractive. When increasing the LTL transportation cost, the maximum savings possible for each instance also changes; the updated bounds on cost savings are found in Table 5. The results for the experiment are reported in Table 6. For each instance, we present the instance number, τ (the number of crossdocks), columns pertaining to the solution for the original setting, columns pertaining to the solution for the new setting, and the decrease in the number of shipments dispatched on LTL routes (ΔLTL). All instances were solved using the heuristic search scheme that solves restricted IPs.

As expected, we see that when LTL transportation costs increase and FTL transportation becomes more competitive, the percentage of orders shipped on LTL routes is reduced and cost savings increase. A closer examination of shipment breakdowns reveals that in most cases the improvement comes from shifting orders from LTL routes to FTL routes that visit a crossdock. In several instances, we see that the increased consolidation occurring at crossdocks further results in a shift from direct FTL routes to FTL routes that visit a crossdock.

Conclusions and future research

We have investigated and developed integer programming-based solution approaches for a shipper pickup and delivery planning problem faced by many large retailers. The challenge in this planning problem is to exploit as much as

Table 6 Impact of increased LTL costs

Inst.	τ	Original LTL costs					Increased LTL costs					Δ LTL
		ΔCost	Shipment Breakdowns by %			Time (h)	ΔCost	Shipment breakdowns by %			Time (h)	
			LTL	FTL-D	FTL-CD			LTL	FTL-D	FTL-CD		
1	3	49.1	73.4	14.8	11.7	0.0003	59.6	40.6	13.3	46.1	0.0004	32.8
2	3	57.7	44.4	20.9	34.7	4.00	69.5	29.2	20.3	50.5	4.00	15.2
3	3	66.2	20.2	38.1	41.7	4.00	76.1	14.5	42.1	43.4	4.00	5.7
4	3	66.3	16.3	21.3	62.4	4.00	76.5	11.3	22.9	65.8	4.00	5.0
5	3	56.2	18.8	42.2	39.1	4.00	69.7	12.9	47.0	40.1	4.00	5.9
6	3	68.2	23.9	36.9	39.2	4.00	77.9	9.5	36.3	54.2	4.00	14.4
7	3	51.8	42.9	24.4	32.7	4.00	65.3	32.3	28.7	39.0	4.00	10.6
8	3	55.0	28.3	9.8	62.0	4.00	67.4	18.0	8.5	73.5	4.00	10.2
9	3	52.3	38.4	20.0	41.6	4.00	64.4	22.8	22.9	54.2	4.00	15.6
10	3	64.8	28.7	46.3	25.1	4.00	74.6	20.4	49.9	29.6	4.00	8.2
11	3	63.5	24.5	41.7	33.8	4.00	73.7	16.0	47.5	36.5	4.00	8.5

possible negotiated truckload lane rates and to judiciously make use of routes through crossdock facilities to consolidate shipments.

The solution approach generates a set of routes that are likely to be part of high-quality solutions and then uses an integer program to choose a low-cost set of routes fulfilling all orders. A search scheme in which a sequence of restricted integer programs are solved was shown to be effective in producing high-quality solutions. The search scheme was introduced to reliably produce high-quality solutions even for large integer programs. The size of the integer programs grows steadily as the number of orders, and thus the set of candidate routes, increases. For the real-life instances that motivated this research, generating a complete set of candidate routes a priori and solving the resulting integer program using the presented search scheme works well. However, if instances with substantially more orders must be solved, the approach may no longer be effective, and column generation techniques or metaheuristic search approaches that do not rely on integer programming may need to be developed. This is left for future research.

Although the primary value of our technology is to reduce transportation costs, the technology can potentially be used in support of lane rate negotiations as well. The lane rates, and the flexibility to add additional stops, have a tremendous impact on the overall costs of the shipment plan. By negotiating advantageous lane rates in strategic areas, significant cost savings can likely be achieved. Modifying the developed technology to identify lanes and lane rate structures that are most valuable in terms of reducing total costs is one of the key research challenges we plan to address in the near future.

References

Chen P, Guo Y, Lim A, Rodrigues B (2006) Multiple crossdocks with inventory and time windows. Comput Oper Res 33(1):43–63

Cortés CE, Matamala M, Contardo C (2010) The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method. Eur J Oper Res 200(3):711–724

De Franceschi R, Fischetti M, Toth P (2006) A new ILP-based refinement heuristic for vehicle routing problems. Math Program B 105:471–499

Hewitt M, Nemhauser GL, Savelsbergh MWP (2009) Combining exact and heuristic approaches for the capacitated fixed charge network flow problem. INFORMS J Comput 22:314–325

Jayaraman V, Ross A (2003) A simulated annealing methodology to distribution network design and management. Eur J Opera Res 144(3):629–645

Lee YH, Jung JW, Lee KM (2006) Vehicle routing scheduling for cross-docking in the supply chain. Comput Ind Eng 51(2):247–256

Li X, Lim A, Miao Z (2006) Reducing transportation costs in distribution networks. Adv Appl Artif Intell 4031(2006):1138–1148

Liao CJ, Lin Y, Shih SC (2010) Vehicle routing with cross-docking in the supply chain. Expert Syst Appl 37(10):6868–6873

Mitrovic-Minic S, Laporte G (2006) The pickup and delivery problem with time windows and transshipment. INFOR 44(3):217–227

Mues C, Pickl S (2005) Transshipment and time windows in vehicle routing. In: 8th International Symposium on Parallel Architetures, Algorithms and Networks, pp 113–119

Savelsbergh MWP, Sol M (1995) The general pickup and delivery problem. Transp Sci 29(1):17–29

Sung CS, Song SH (2003) Integrated service network design for a cross-docking supply chain network. J Oper Res Soc 54(12):1283–1295

Wen M, Larsen J, Clausen J, Laporte G (2009) Vehicle routing with cross-docking. J Oper Res Soc 60:1708–1718