Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2012 Proceedings | Proceedings

The Position-Aware-Market: Optimizing Freight Delivery for Less-Than-Truckload Transportation

Xiaoqiu Qiu
Information Systems Research, Albert-Ludwigs-Universität Freiburg, Freiburg im Breisgau, Baden Württemberg, Germany., jimworks@gmail.com

Dirk Neumann
Information Systems Research, Albert-Ludwigs-Universität Freiburg, Freiburg im Breisgau, Baden Württemberg, Germany., dirk.neumann@is.uni-freiburg.de

Markus Goetzinger
Information Systems Research, University of Freiburg, Freiburg, Germany., markus.goetzinger@gmail.com

The Position-Aware-Market: Optimizing Freight Delivery for Less-Than-Truckload Transportation

Xiaoqiu Qiu
Information Systems Research,
Albert-Ludwigs-Universität Freiburg
xiaoqiu.qiu@is.uni-freiburg.de

Markus Götzinger
Information Systems Research,
Albert-Ludwigs-Universität Freiburg
markus.goetzinger@is.uni-freiburg.de

Dirk Neumann
Information Systems Research,
Albert-Ludwigs-Universität Freiburg
dirk.neumann@is.uni-freiburg.de

ABSTRACT

The increasing competition faced by logistics carriers requires them to ship at lower cost and higher efficiency. In reality, however, many trucks are running empty or with a partial load. Bridging such residual capacity with real time transportation demand enhances the efficiency of the carriers. We therefore introduce the Position-Aware-Market (PAM), where transportation requests are traded in real time to utilize transportation capacities optimally. In this paper we mainly focus on the decision support system for the truck driver, which solves a profit- maximizing Pickup and Delivery Problem with Time Windows (PM-PDPTW). We propose a novel Recursive Branch-and-Bound algorithm that solves the problem optimally, and apply it to a Tabu-Search heuristic for larger problem instances. Simulations show that problems with up to 50 requests can be solved optimally within seconds. Larger problems with 200 requests can be solved approximately by Tabu-Search in seconds, retaining 60% of the optimal profit.

Keywords

Green Logistics, Logistics Optimization, Profit Maximization, PDP, PDPTW, Less-than-Truckload, Transportation Market

INTRODUCTION

Nowadays logistics carriers have become more globalized and deregulated. However, the competition they face is ever increasing. This requires carriers to cut their transportation costs, and to improve the utilization of the fleets' capacity. However, in reality, there exists a significant temporary imbalance in logistics supply and demand. According to (Archetti et al., 2008), 30% of the trucks travel empty in Europe, "`many more have a partial load`". This brings unnecessary cost to logistics carriers, and excess carbon footprints to the environment (Leonardi et al., 2004; Dekker et al., 2012).

The market is a classic approach to solve the imbalance in supply and demand. For people transportation, markets that connect passengers with available transportation capacity have already existed (Mitfahrzentrale). Passengers can browse through available connections and hitch a ride, while drivers can search for passengers to spread the trip cost.

Compared to people transportation, the transportation of freights has more complex constraints, e.g. Loading Sequence, Safety Regulations, etc. Considering the computation complexity of logistics problems (Parragh et al., 2008), freight deliveries are usually planned long in advanced. Therefore, the imbalance of supply and demand may exist for a long time in the market. When a new transportation request emerges, even if there is a truck with the available capacity, it might still miss the request due to the lack of information.

A simple example clarifies this: A half-full truck is traveling from City A to City C, passing City B as an intermediate stop without any freight operation. Now a new request from City B to City C emerges, while the truck has available capacity. If the driver stops to pick up the new request, he gains higher revenue with slightly increased route cost. On the other hand, the logistic sender gets a quicker delivery. This is a win-win solution.

Now assumed that the new request does not coincide with the planned route, the driver then faces a tradeoff: should he

deviate for the new request or not? Were there more requests available, what is the most profitable way to detour? To seek the answers we must solve a Profit-Maximizing Pickup and Delivery Problem with Time-Windows (PM-PDPTW), which is the focus of this paper.

Further, were there more than one truck, different drivers might bid on the same requests. To solve the request allocation problem, we need a market for winner-determination and pricing. This is to become our future studies.

Currently internet transportation markets (Teleroute, Benelog) have already implemented standardized transactions and electronic trading. They have succeeded in increasing utilization and reducing transportation cost. However, planning ahead is still required, while bidding in the auction is not yet automated.

In this paper we address the gap by proposing a Position-Aware -Market (PAM) that connects the drivers with freight senders in real time. In the market, freight senders post requests with prices on a bulletin board. Drivers access this demand information in real time. The onboard decision support system then automatically solves the profit maximizing problem of selecting requests, and suggests a feasible route. At each intermediate stop, the driver updates nearby requests, and the onboard system re-optimizes the route locally. With minor modification, our proposed algorithm can accommodate different loading constraints (e.g. First-In-First-Out, Last-In-First- Out, etc.) resulted from different vehicle types.

The remainder of this paper is structured as follows: Section 2 presents related works on similar optimization problems. The assumptions and the mathematical model are described in section 3. Section 4 presents our solutions, including a novel Recursive Branch-and- Bound (RBB) algorithm, and a Tabu-Search heuristic for approximation. The solutions are evaluated with simulations in section 5. Section 6 concludes briefly with an outlook for future research.

RELATED WORKS

Our PM-PDPTW problem originates from the classical Traveling Salesman Problem (TSP), which is NP-hard (Balas, 1999) and has received extensive studies over the years (Applegate et al., 2007). Due to its practical relevance in transportation and logistics, many variations with pickup and deliveries are developed and studied. These are usually called Vehicle Routing Problems (VRP). According to the survey by (Parragh et al., 2008), VRP falls into two classes, VRP with Backhauls (VRPB) and VRP with Pickup and Deliveries (VRPPD). In VRPB all goods are transported from or to a central depot, while in VRPPD goods are picked up and delivered between multiple locations. Our problem is a special case in VRPPD in which every transportation request is associated with a specific pickup and delivery location. A comprehensive survey on this problem class can be found in (Cordeau et al., 2008).

In VRP a common constraint is to visit all locations, similar to TSP. Relaxing this constraint and introducing profit-maximization, we get to TSP with Profits (TSPP). In this class the cost-minimizing criterion is either absent, as a bi-objective, or as a constraint on maximal cost. A recent survey in TSPP can be found in (Feillet et al., 2005a).

An overview of the related works is listed in Table 1. All these works consider vehicles with capacity constraints.

Model	Obj. [4]	#Good	#Vehicle	# Node	TW[5]	Size	Algorithm	Opt[6]	Literature
1. PDTSP [1]									
1-PDTSP	C	1	1	all	-	100	Branch- Cut	+	Hernández-Pérez et al. 2007
	C	1	1	all	-	500	Greedy + *k*-opt	-	Hernández-Pérez et al. 2004
PDTSP-L/F	C	1	1	all	-	L:35 F:27	Branch -Bound	+	Carrabs et al. 2007
2. PDP [2]									
TSPPD	C	m	1	all	-	35	Branch -Cut	+	Dumitrescu et al. 2008
PDPTW	C	m	m	all	+	500	Branch-Cut-Price	+	Ropke et al. 2009
	C	m	m	all	+	500	Set-Partitioning + Branch-Cut-Price	+	Bartolini 2009
3. PARP [3]									
PRPP	P	1	1	some	-	100 vertices 1225 edges	Cutting-Plane + Heuristics	+	Aráoz et al. 2009
PATP	P	1	1	all	-	30	Branch-Price	+	Feillet et al. 2005b
4. TSPP									
TSPP	P+C	1	1	some	-	150	ε-Consraint	+	Bérubé et al. 2007
AtTSP	P	1	1	some	+	400	Branch-Cut + Tabu	+	Erdoğan et al. 2010
PM-PDPTW	P	m	1	some	+	250	Branch-Bound + Tabu	+	**Position-Aware-Market**

Model Classes: [1)] PDTSP: Pickup and Delivery TSP; [2)]PDP: Pickup and Delivery Problem; [3)] PARP:Price-collected Arc Routing Problem. [4)] Objective: C: Cost-Minimization; P: Profit-Maximization. [5)] TW: Time-Windows. [6)] Opt: Exact Algorithm – Optimally Solved.

Table 1 List of related works

In Pickup and Delivery TSP (PDTSP), Price-collected Arc Routing Problems (PARP) and TSPP, there is only one good to transport, and it can be loaded at any pickup node and unloaded at any delivery node. Further studies introduce more complex constraints such as Last-In-First-Out (PDTSP-L) or First-In-First-Out (PDTSP-F) loading requirements (Carrabs et al., 2007).

In PARP and TSPP the objective is instead Profit Maximizing (or as a bi-objective as in Bérubé et al., 2007). This is achieved by skipping some nodes as in the Price-collecting Rural Postman Problem (PRPP, Aráoz et al., 2009) or by repeated-visit of some cycles in the Profitable Arc Tour Problem (PATP, Feillet et al. 2005b). In the Attractive TSP (AtTSP), the profit is inversely related to the distance between facility nodes and customer nodes, with a constraint on both time and route-length (Erdoğan et al., 2010).

With heterogeneous goods, the routing problem becomes more difficult as it change from scheduling locations to scheduling transportation requests. Recently the problem is solved exactly by a combination of Branch-and-Price and Branch-and-Cut algorithms to very large instances (Ropke et al. 2009, Bartolini, 2009). However, they all focus on cost-minimizing subject to delivering all requests.

Among these routing problems we see a missing gap of profit maximizing through a subset of heterogeneous transportation requests, subject to time and capacity constraints. We call it the Profit-Maximizing Pickup and Delivery Problem with Time-Windows (PM-PDPTW), and solve it both optimally and approximately in this paper. As we are proposing PAM for real-time trading of freight transportation, solving PM-PDPTW builds the foundation for future studies on bid-formation, pricing and auction allocation, which are essential to PAM.

MATHEMATICAL FORMULATION OF THE PROFIT-MAXIMIZING PICKUP AND DELIVERY PROBLEM WITH TIME WINDOWS (PM-PDPTW) FOR THE DRIVER

In this section, we first form the necessary assumptions, and then formulate the model mathematically.

Assumptions

The following assumptions are applied throughout the model.

Requests are independent of each other.

Transportation requests are independent of each other. There is no precedence constraint between the requests. Every request's revenue depends solely on its own attributes, such as capacity usage, time constraint, the valuation of the goods, etc. Attributes of the other requests, such as their locations, and the commuting costs between different requests, shall not affect the revenue of the request.

One Profit Maximizing Driver delivers a Subset of Available Requests.

In a transportation exchange, satisfying all delivery demand by one truck is not always possible. Because of capacity or time constraints, different requests can be incompatible with each other. Hence the truck driver shall choose only a subset of all available requests to bid. If a request is not bidden for a long time, it reflects an underestimation of the transportation cost by the freight sender. Under this setting the market is not always clear, and new requests can emerge continuously.

The Driver bids on Requests rather than Routes.

As compared to the "`hierarchical`" method where solutions are represented as a sequence of routing information and an overlay of pickup-delivery information, our solution is formed in a "`sequential`" way (Moura and Oliveira, 2008), where every transportation request is labeled with two unique nodes, one for pickup and one for delivery. Every solution specifies a sequence of the nodes, encoding both routing and delivery information. Therefore, different requests would have different pickup and delivery nodes, regardless of the underlying location. For instance, even if two requests start from the same city, they would still be modeled into two different pickup nodes.

Less-Than-Truckload (LTL) formation with a Capacitated Truck.

In comparison to Full-Truckload problems, Less-Than-Truckload (LTL) problems deal with the requests that only cost a fraction of the truck's capacity to deliver. A driver can therefore reduce the transportation cost by "`consolidating the`

`freight over the network`" (Powell, 1986). In other words, different requests are delivered together to maximize the utilization of the truck's capacity.

Time Windows with No Waiting Allowed, Constraint on Total Ride Time.

At each node, regardless of pickup or delivery, the driver is required to arrive between a certain time window. We assume that the driver cannot arrive before the time window (no waiting allowed), since in reality there might exist constraints such as parking space or traffic control, which limits the possibility of waiting. On the other hand, the starting time of each time window can be set arbitrary early to incorporate the situation where waiting is possible. Further, we also consider loading time at each node, and a constraint on the total ride time which is reflected by the time window of the end depot.

Formulation of PM-PDPTW

Our problem is formulated as a two-dimensional mixed-integer optimization problem, based on the Dial-A-Ride Problem in (Cordeau et al., 2008). Here, the two-index binary variable x_{ij} is equal to 1 if and only if the driver travels from node i to node j directly.

A total of n available requests are formed on a directed graph $G = (V, A)$, where the node set V is divided into pickup nodes $P = (1,2, \dots, n)$, delivery nodes $D = (n+1, n+2, \dots, 2n)$, and depots $(0, 2n+1)$. The driver always starts the route from the start depot 0 and ends at the end depot $2n+1$. Each request i is modeled by two nodes: its pickup node i and delivery node $n+i$. A request i can only be picked up at the node i and transported to the corresponding node $n+i$.

The revenue gain from request i (from node i to node $i+n$, directly or indirectly) is given by $\pi_i, (i \in P)$. Let $c_{ij}, (i, j \in V)$ be the travel cost from node i to node j, $t_{ij}, (i, j \in V)$ the travel time from node i to node j, $u_i, (i \in V)$ the exact time when the driver arrives at node i, $d_i, (i \in V)$ the time needed for loading at node i.

Let $w_i, (i \in V)$be the capacity usage of the truck after node i, $q_i, (i \in V)$ the change in capacity at node i. Finally let Q be the truck's capacity constraint, and T the total ride-time constraint. The model is as follows.

$$\max \sum_{i \in P} \pi_i \cdot (\sum_{j \in V} x_{ij}) - \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} \qquad (1)$$

subject to:

$$\sum_{i \in V} x_{0i} = \sum_{i \in V} x_{i,2n+1} = 1 \qquad (2)$$

$$\sum_{j \in V} x_{ij} - \sum_{j \in V} x_{n+i,j} = 0 \qquad (i \in P) \qquad (3)$$

$$\sum_{j \in V} x_{ij} - \sum_{j \in V} x_{ji} = 0 \qquad (i \in P \cup D) \qquad (4)$$

$$\sum_{i,j \in S} x_{ij} \leq |S| - 1 \qquad (S \subseteq P \cup D) \qquad (5)$$

$$u_j \geq (u_i + d_i + t_{ij}) x_{ij} \qquad (i, j \in V) \qquad (6)$$

$$w_j \geq (w_i + q_j) x_{ij} \qquad (i, j \in V) \qquad (7)$$

$$u_{2n+1} - u_0 \leq T \qquad (8)$$

$$e_i \leq u_i \leq l_i \qquad (i \in V) \qquad (9)$$

$$\max\{0, q_i\} \leq w_i \leq \min\{Q, Q + q_i\} \quad (i \in V) \qquad (10)$$

$$x_{ij} = 0 \text{ or } 1 \qquad (i, j \in V) \qquad (11)$$

In this formulation, (1) is the objective function on profit. The first half $\sum_{i \in P} \pi_i \cdot (\sum_{j \in V} x_{ij})$ is the revenue, if request i is picked up; the pickup node i must be visited once, therefore $\sum_{j \in V} x_{ij} = 1$, and the driver receive its revenue π_i. The second half is the route cost, if an arc (i, j) is traversed, the route cost c_{ij} is incurred.

Constraint (2) ensures that the trip starts from the start depot (node 0) and ends at the end depot (node $2n + 1$). Constraint (3) ensures the picked up tasks are delivered. Constraint (4) ensures that if a non-depot node is visited, the truck must leave the node. Constraint (5) eliminates sub-tours. Together constraints (2)-(5) guarantee the connectivity and validity of the routes.

Constraint (6) defines the service time from node i to node j, meaning that if the route (i, j) is traversed, the end time should be the starting time plus the time for loading and traveling. Constraint (7) defines the truck's load (capacity usage) from node i to node j.

Constraint (8) ensures that the total ride-time is below constraint, while constraint (9) guarantees that the service time at each node falls between the required time-windows. Finally, constraint (10) ensures that the capacity usage is feasible.

SOLUTIONS TO PM-PDPTW FOR THE DRIVER

Because of NP-hardness, it is difficult to obtain the global optimal solution for large instances in reasonable time (Hochba, 1997). Hence we devise a Recursive Branch-and-Bound (RBB) algorithm that focuses on a better branching procedure for finding the optimal solution, and a Tabu-Search heuristic to attain good solutions in controllable time.

The Recursive Branch-and-Bound Algorithm (RBB)

By denoting each request i with a pair of nodes $(i, n + i)$, the disadvantage is a huge increase in the size of the decision matrix X. With n requests, the decision matrix would have the size of $(2n + 2)^2$, with most combinations being infeasible solutions. Among the infeasibilities, many of them are invalid routes. For instance, routes that visit the delivery node before the pickup node, routes that visit only the pickup node or the delivery node but not both, routes that consist of disconnected sub-tours, etc. The RBB algorithm is designed to eliminate such infeasibilities, and to generate routes that:

1. Have no disconnected sub-tours, start from the start depot and end at the end depot.
2. If a pickup node is visited before, the delivery node must be visited later before the end depot.
3. The delivery nodes cannot be visited before corresponding pickup nodes.

We noticed that at each intermediate stop of the route (node $i \in P \cup D$), the possibilities are limited to:

1. The driver picks up a new transportation request.
2. The driver delivers an onboard transportation request.
3. The truck is empty and the driver terminates the route by heading to the end depot.

Let P be the queue of unvisited pickup nodes, D the queue of delivery nodes that the corresponding pickup nodes have been visited before, R the queue of the current route. The above possibilities can be transformed into the following branching moves:

1. Choose a node i from P, add it to the end of R, then add its corresponding delivery node $i + n$ to D, and remove i from P. (*Pickup a new request*)
2. Choose a node i from D, add it to the end of R, then remove it from D. (*Deliver an onboard request*)
3. If queue D is empty, add $i = 2n + 1$ to the end of R, search terminates, and output the route R. (T*ruck empty*)

With D initialized to be empty, P the set of possible pickup nodes, and R only the start depot $i = 0$.

Because every stop of the route shares the same search structure, the search process can be programmed into a recursive algorithm. Before each recursion call (branching), the algorithm checks the time and capacity constraints, as well as the current best solution, to reduce the number of sub-trees visited (bounding).

We compare the number of leaf nodes searched with a primitive Branch-and-Bound (BB) algorithm in Table 2. Even for small instances, the new branching rule reduces the search space significantly. (The number of leaf nodes searched by the primitive BB algorithm is calculated as $\sum_{i=0}^{n}(2i)!$, since for a route with exactly $2i$ intermediate nodes the number of possible permutations is $(2i)!$.)

No. of Requests	RBB	Primitive BB	Percentage
1	2	3	66.67%
2	9	27	33.33%

3	112	747	14.99%
4	2921	41067	7.11%
5	126966	3669867	3.46%

Table 2 Comparison of the number of leaf nodes searched by RBB and the primitive Branch-and-Bound.

The pseudo code of the Recursive Branch-and-Bound algorithm (RBB) is as follows:

```
Initialization
P ← i ∈ the possible set of pickup nodes.
R ← i = 0. D ≔ ∅

Recursion (P, R, D as input)

    if time constraint or capacity constraint violated in R then
        return

    for each i ∈ P ∪ D
        if i ∈ P then
            P → i, R ← i, D ← i + n.
            Recursion (P, R, D)
            P ← i, R → i, D → i + n.
        else
            D → i, R ← i.
            Recursion (P, R, D)
            D ← i, R → i.
        end

        if D is empty then
            R ← 2n + 1
            output R.
        end
    for loop end

Recursion Function Ends
```

For a complete search over n requests, P is initialized as a series with all integers between $[1, n]$. By initializing P differently, RBB can also be used as a local-search procedure. For example, by initializing P as a subset of requests, this algorithm would find all possible delivery plans upon that subset.

By changing D from a normal queue to a stack where only the top element can be retrieved, RBB can handle problems that require a First-In-Last-Out (FILO) loading constraint. Similarly, a First-In-First-Out (FIFO) constraint can be handled by changing D into a queue whose elements can only be added at one end and retrieved at the other end.

Finally, as RBB searches in a depth-first manner, optimal routes are updated gradually through the searching process. In other words, the convergence towards the optimal solution occurs before the completion of the search. Therefore we can use it to find approximate solutions with early termination. The performance of this variation is compared with the Tabu-Search heuristic in section 5.

Tabu-Search

The basic idea of Tabu-Search is to remember recent search moves so as to prevent repeated search of the same solution space (Glover and Laguna, 1997). Applying to our problem, Tabu-Search works as follows: At each round a subset of the requests is randomly selected. This subset would be added to a Tabu-list. Then we solve for the best route-request combination upon the subset, using the RBB algorithm proposed in 4.1. The key here is the size of the subset chosen for the

local-search, as a larger subset contains more possibilities, at the cost of a longer runtime (For instance, if the subset is limited to 6 requests, while the optimal solution contains 7 requests, then Tabu-Search can never find the optimal solution).

A small tweak of the algorithm is to introduce some greediness. Every time a new subset is created, replace some requests with high-revenue requests with probability.

For stopping criteria, we use the number of iterations or runtime. In the normal case, the search stops when there is no improvement in solution quality after certain iterations. In case runtime is critical, a timer is added to both the Tabu-Search's main procedure and the RBB local search to stop the algorithm effectively once it surpasses the required time.

EVALUATION

The Recursive Branch-and-Bound (RBB) algorithm and the Tabu-Search heuristic are programmed in C#, and tested with randomly generated instances of different sizes. The main focus is to find out the efficiency and runtime trade-off between the two solutions and their variations.

The test instances are generated with the following setup:

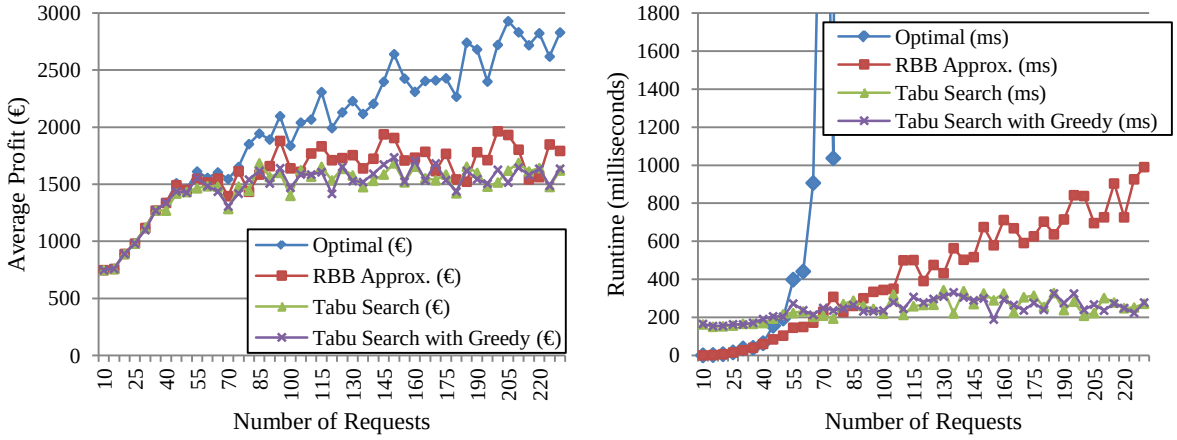
1. Route costs are uniformly distributed between 10€ and 400€, and satisfy triangular inequality.
2. To reflect the fact that travel time and route length are correlated, travel time between cities are scaled down by 20 from the corresponding route cost (a distribution between 0.5h and 20h), plus a random variation of 1h reflecting traffic difference.
3. Loading times are uniformly distributed between 0 and 10h.
4. Time Windows are uniformly distributed between 1h and 10h.
5. Total ride-time constraint is restricted to 30h.
6. Revenues are uniformly distributed between 200€ and 800€.
7. Capacity usages of the requests are uniformly distributed between 2t and 10t.
   The truck has a 20t capacity limit.

The simulations are performed with the number of iterations or runtime as stopping criteria. For each setup, 10 different random instances are tested and the averages are reported.

Optimality and Runtime Tradeoff

We first test the approximation methods, Tabu-Search and RBB with early termination (RBB Approx.), against RBB to see the optimality-runtime tradeoff. Two stopping criteria are applied respectively, limiting iterations with no improvement, and limiting runtime.

Since Tabu-Search runs RBB as a local search in every iteration, while RBB Approx. counts iterations by reaching the leaf node (route completed or constraint violated), when iterations are used as stopping criteria we limit the number to 500 for Tabu-Search and 2000 for RBB Approx. for fairness concerns.

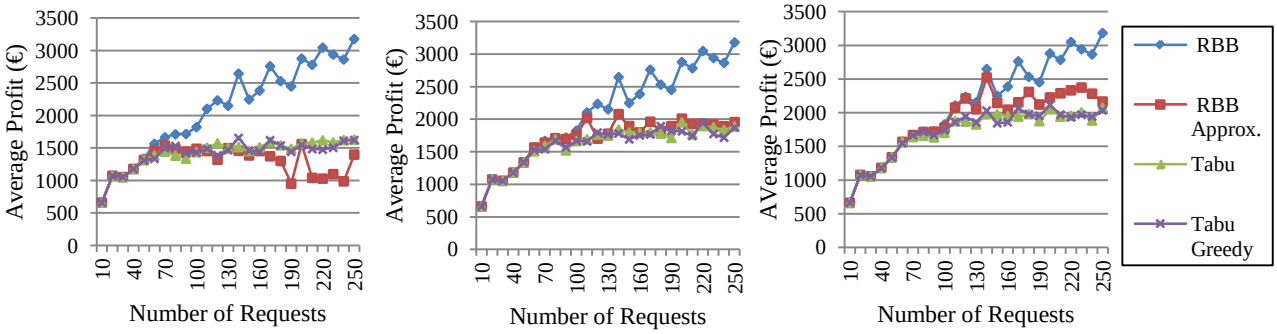


Graph 1 Profit and Runtime Comparison with Iteration Limit

Profit Comparison (€)							
No. of Requests	RBB	RBB Approx		Tabu		Tabu Greedy	
10	747.8	747.8	100.00%	747.8	100.00%	747.8	100.00%
100	1837.2	1640.5	89.29%	1399.4	76.17%	1470	80.01%
200	2722	1964.9	72.19%	1516.2	55.70%	1627.2	59.78%
Runtime Comparison (Milliseconds)							
No. of Requests	RBB	RBB Approx		Tabu		Tabu Greedy	
10	0.6	0.4	66.67%	162.7	27117%	165.1	27517%
100	3506.2	344.6	9.83%	219.3	6.25%	235.6	6.72%
200	159588.8	838.4	0.53%	208.6	0.13%	240.1	0.15%

Table 3 Profit and Runtime Comparison with Iteration Limit

In practice, we opt for heuristics when solving exactly is too time-consuming. As limiting iterations cannot guarantee an upper bound on runtime, limiting runtime can be a more direct measure. We therefore test all methods in a 0.1, 1, and 5 second time-frame, to see how it differs from the iteration criteria.



Graph 2 Profit and Runtime Comparison with Runtime Limit
0.1 second (left), 1 second (middle) and 5 seconds (right)

No. of Requests	RBB	RBB Time (ms)	0.1 second			1 second			5 second		
			RBB Approx	Tabu	Tabu Greedy	RBB Approx	Tabu	Tabu Greedy	RBB Approx	Tabu	Tabu Greedy
10	659.7	0.7	659.7	659.7	659.7	659.7	659.7	659.7	659.7	659.7	659.7
100	1823.7	6164.4	1489.3	1437.9	1430.3	1754.3	1662.8	1667.6	1774.6	1698.9	1739.1
200	2877.7	221296.9	1559.6	1573	1547.6	2007.6	1950.6	1813.3	2224.2	2049.3	2114.8
250	3179.5	1471284.5	1399.5	1630.9	1622.3	1954.3	1886.5	1870.9	2163.3	2071.5	2031.6

Table 4 Profit and Runtime Comparison with Runtime Limit

From both tests we see that under 50 requests RBB outperforms the approximation methods with a shorter runtime, making it ideal for local search or small-scale problems. For larger problems, RBB Approx. needs longer runtime to deliver better performance, while both versions of Tabu-Search perform similarly with high stability in profit and runtime.

Comparing Approximation Methods

To distinguish the performance of the approximation methods (Tabu-Search and RBB Approx.), we further test them with larger instances up to 1,500 requests.

Profit (€)				Runtime (Milliseconds)			
No. of Requests	RBB Approx.	Tabu	Tabu Greedy	No. of Requests	RBB Approx.	Tabu	Tabu Greedy
50	1315.8	1319.7 100%	1274.3 97%	50	95.1	77.4 81%	97.3 102%

500	1451.2	1476.7 102%	1526.9 105%	500	1962.3	202.5 10%	173.8 9%
1000	1674.6	1701.3 102%	1624.1 97%	1000	7342.1	403.3 5%	425.1 6%
1500	1108.4	1386.9 125%	1604 145%	1500	13066	475.5 4%	502.4 4%

Table 5 Profit and Runtime Comparison between Approximation Methods

The result shows that under 500 requests RBB Approx. attains similar profit as Tabu-Search. With larger instances RBB Approx. performs poorly in both runtime and profit. Introducing greediness into Tabu-Seach has an insignificant effect on profit but a slight increase in runtime. As in Tabu-Search the route subsets are initialized randomly, it sometimes delivers extreme results that cause the average profit to fluctuate randomly.

As a conclusion, when the problem contains less than 50 requests or runtime is not a priority concern, RBB guarantees optimality in a reasonable time, making it ideal for local-search. For larger instances or time-critical situations, Tabu-Search provides a good guess of optimal profit in a short time, which is good for generating an initial solution for PAM. With medium-size problems and long runtime, RBB Approx. can be a good alternative for approximation.

CONCLUSION AND FUTURE RESEARCH

In this paper we proposed a Position-Aware-Market (PAM) that aims to improve the utilization of Less-than-Truckload freight transportation. The key sub-problem is a Profit Maximizing Pickup-and-Delivery Problem with Time Windows (PM-PDPTW) that has not been studied before. We devise a novel Recursive Branch-and-Bound algorithm to locate the optimal solution, and a Tabu-Search heuristic for finding the approximate solutions for large instances in limited runtime. The simulations on randomly generated instances show that problems with less than 50 requests can be solved optimally within seconds. Larger problems can be solved approximately using Tabu-Search in seconds, retaining 60% of the optimal profit on average.

Another sub-problem in PAM that we have not yet addressed is the market mechanism. For future research, we planned to adopt a more realistic combinatorial-auction setup, in which the goal is to eliminate driver-side gaming during the bidding process. Finally, as logistic shipments are often planned centrally, a model for finding the general equilibrium solution under central planning is also needed to benchmark the proposed PAM approach.

REFERENCES

1. Applegate, D.L., Bixby, R.E., Chvatal, V., and Cook, W.J. (2007) The Traveling Salesman Problem: A Computational Study (Princeton Series in Applied Mathematics). Princeton University Press.
2. Aráoz, J., Fernández, E., and Meza, O. (2009) Solving the prize-collecting rural postman problem. *European Journal of Operational Research 196*, 3, 886–896.
3. Archetti, C., Feillet, D., Hertz, A., and Speranza, M.G. (2008) The capacitated team orienteering and profitable tour problems. Journal of the Operational Research Society 60, 6, 831–842.
4. Balas, E. (1999) New classes of efficiently solvable generalized traveling salesman problems. *Annals of Operations Research 86*, 529–558.
5. Bartolini, E. (2009) Algorithms for network design and routing problems. Ph.D. Thesis, Università di Bologna.
6. Benelog: http://www.benelog.com/
7. Bérubé, J.-F., Gendreau, M., and Potvin, J.-Y. (2009) An exact ϵ-constraint method for bi-objective combinatorial optimization problems: Application to the Traveling Salesman Problem with Profits. *European Journal of Operational Research 194*, 1, 39-50.
8. Carrabs, F. and Cordeau, J.-F. (2007) An Additive Branch-and-Bound Algorithm for the Pickup and Delivery Traveling Salesman Problem with LIFO or FIFO Loading. *INFOR 45*, 4, 223-238.
9. Cordeau, J.-F. J., Laporte, G., and Ropke, S. (2008) Recent Models and Algorithms for One-to-One Pickup and Delivery Problems, *The Vehicle Routing Problem: Latest Advances and New Challenges*, (Vol. 43) Springer US, pp. 327-357.
10. Dekker, R., Bloemhof, J., and Mallidis, I. (2012) Operations Research for green logistics – An overview of aspects, issues, contributions and challenges. *European Journal of Operational Research*, doi:10.1016/j.ejor.2011.11.010.

11. Dumitrescu, I., Ropke, S., Cordeau, J.-F., and Laporte, G. (2008) The traveling salesman problem with pickup and delivery: polyhedral results and a branch-and-cut algorithm. *Mathematical Programming 121*, 2, 269-305.
12. Erdoğan, G., Cordeau, J.-F., and Laporte, G. (2010) The Attractive Traveling Salesman Problem. *European Journal of Operational Research 203*, 1, 59-69.
13. Feillet, D., Dejax, P., and Gendreau, M. (2005a) Traveling Salesman Problems with Profits. *Transportation Science 39*, 2, 188-205.
14. Feillet, D., Dejax, P., and Gendreau, M. (2005b) The Profitable Arc Tour Problem: Solution with a Branch-and-Price Algorithm. *Transportation Science 39*, 4, 539-552.
15. Glover, F. and Laguna, M. (1997) Tabu Search. Kluwer Academic Publishers Group.
16. Hernández-Pérez, H. and Salazar-González, J.-J. (2007) The one-commodity pickup-and-delivery traveling salesman problem: Inequalities and algorithms. *Networks 50*, 4, 258-272.
17. Hernandez-Perez, H. and Salazar-Gonzalez, J.-J. (2004) Heuristics for the One-Commodity Pickup-and-Delivery Traveling Salesman Problem. *Transportation Science 38*, 2, 245-255.
18. Hochba, D.S. (1997) Approximation Algorithms for NP-Hard Problems. *ACM SIGACT News 28*, 2, 40-52.
19. Leonardi, J. and Baumgartner, M. (2004) CO_2 efficiency in road freight transportation: Status quo, measures and potential. *Transportation Research Part D: Transport and Environment 9*, 6, 451-464.
20. Mitfahrzentrale/Mitfahrgelegenheit.de: http://www.mitfahrgelegenheit.de/
21. Moura, A. and Oliveira, J.F. (2008) An integrated approach to the vehicle routing and container loading problems. *OR Spectrum 31*, 4, 775-800.
22. Parragh, S.N., Doerner, K.F., and Hartl, R.F. (2008) A survey on pickup and delivery problems. *Journal für Betriebswirtschaft 58*, 1, 21-51 and 2, 81-117.
23. Powell, W.B. (1986) A Local Improvement Heuristic for the Design of Less-than-Truckload Motor Carrier Networks. *Transportation Science 20*, 4, 246-257.
24. Ropke, S. and Cordeau, J.-F. (2009) Branch and cut and price for the pickup and delivery problem with time windows. *Transportation Science 43*, 3, 267–286.
25. Teleroute: http://www.teleroute.com/