A Set-covering Model for a Bidirectional Multi-shift Full Truckload Vehicle Routing Problem

Ruibin Bai, Ning Xue, Jianjun Chen
International Doctoral Innovation Centre
School of Computer Science
University of Nottingham Ningbo China, Ningbo 315100, China

Gethin Wyn Roberts
Department of Civil Engineering,
University of Nottingham Ningbo China, 315100, China.

June 24, 2015

Abstract

This paper introduces a bidirectional multi-shift full truckload transportation problem with operation dependent service times. The problem is different from the previous container transport problems and the existing approaches for container transport problems and vehicle routing pickup and delivery are either not suitable or inefficient. In this paper, a set covering model is developed for the problem based on a novel route representation and a container-flow mapping. It was demonstrated that the model can be applied to solve real-life, medium sized instances of the container transport problem at a large international port. A lower bound of the problem is also obtained by relaxing the time window constraints to the nearest shifts and transforming the problem into a service network design problem. Implications and managerial insights of the results by the lower bound results are also provided.

Keywords: Full truckload transport; container transport; vehicle routing; set covering; service network design;

1 Introduction

Freight transportation research is often classified into full truckload transport (FTL), less-than truckload transport (LTL) and express deliveries (Wieberneit, 2008). Despite numerous research studies on freight transportation, most of them have focused on consolidation based transportation (LTL and express deliveries) and research on the FTL problem is somewhat limited. Among all the FTL problems, container transport is a special case of the full truckload transport problem since containers are both shipment commodities and transport resources (Braekers et al., 2013). The container transportation industry is under fierce competition and pressure to improve its efficiency and reduce energy use and increasingly more studies have been devoted to the optimisation of operations at container terminals (see Tang et al. (2014) for a recent example). In this paper, we study a multi-shift inter-dock container forwarding problem, using data from a real-life problem faced by one of the largest ports in the world. The problem is also common for any large port with multiple docks being operated simultaneously.

Compared to other types of full truck load transportation problems, the transport distances in the cross-dock container shipment problem concerned in this paper are relatively short. At present, the daily transport demand varies from 100-300 containers. The existing schedule in practice used by the company has around 38% empty truck mileage which is too high. Due to the special features of the problem (see Section 2), there are no existing models and solution methods that can be directly applied.

2 Problem Description

The problem concerned in this paper is extracted from a real-world container inter-dock transportation problem (Chen et al., 2013). The objective is to minimise the total vehicle travel distance for transporting a large number of non-consolidatable commodities (containers) between a relatively small number of nodes (docks), satisfying various time window constraints concerning commodities and drivers. Typical time window constraints are the available time and deadline for commodities and work shifts for drivers. Thanks to internationally adopted EDI (Electronic Data Interchange) systems and GPS (Global Positioning Systems) sensors, the available and deadline times of commodities are generally known 1-2 days in advance with some tolerable estimation errors. The physical locations of different nodes are shown in Figure 1. The problem has the following unique characteristics:

- The unit size of the commodity is equivalent of that of the trucks.

Figure 1: Positions of the container terminals in the port under study.

Therefore one unit of a commodity is shipped directly to its destination without transfers or consolidations.

- Schedules are shift based. Each truck has to go back to a depot for driver changeovers after a shift due to labour law related regulations. For this particular problem, a shift is 12 hours. A schedule typically spans from 4-8 shifts in order to maximise the efficiency.

- All docks are within a short distance of each other and a unit of any commodity could be completed within a shift. The service time at each node (loading time at the the source or unloading time at the destination of a shipment) is comparable to the truck travel times between nodes. Therefore, service times cannot be ignored or simplified.

- The time window for each commodity (i.e. from time when the commodity becomes available to the time when it has to be delivered to its destination) varies considerably from 1-2 hours, up to 6 shifts.

- The total quantities of all the commodities within a planning horizon can be very large (up to 2000) but the number of distinct physical nodes is relatively small (less than 10).

Because of these distinct features, the problem cannot be solved by off-the-shelf approaches designed for vehicle routing problems with pickups and

deliveries. Similarly, the problem is different from the classic service network design problem (SNDP) which is primarily consolidation oriented. Although our problem shares similar constraints to the inland container transportation problem studied in Zhang et al. (2010), the nature of the constraints are very different. For example, the time window of a load in the problem considered in this research spans from a few hours to up to 3 days (or 6 shifts), compared to a load time window of between 1 to 4 hours in (Zhang et al., 2010). Therefore, the time-window partition heuristics will not work for our problems due to the potentially huge number of sub-loads generated, causing prohibitive computational time. When the time window of a load spans several working shifts, determination of the shift in which this load is serviced forms part of the decisions to optimise. Therefore, the problem concerned in this research requires a multi-shift model that is much larger than the single-shift model adopted in (Zhang et al. 2010).

More recently, Nossack and Pesch (2013) also proposed a nonlinear integer programming model, in which time window constraints are handled explicitly. However, due to its nonlinear property, the formulation cannot be solved exactly. In our proposed set-covering model, these time window constraints are implicitly handled offline during the feasible route generation stage. In this way, we can handle any forms (linear, nonlinear) of route related constraints, including nonlinear time window constraints and shift constraints. In fact, more and tighter constraints are advantageous to this formulation as it can reduce the size of the feasible route set.

Without losing the generality of the problem, we define the following shift-based full truckload shipment problem with operation dependent service time. A full list of the notations used in our model is given in Table 1. Denote $G = (V, A)$ a directed graph with a set of nodes V (representing origins and destinations of different commodities) and a set of arcs A between these nodes. Note that node 0 is the depot from which all vehicles depart at the beginning of the shift. Denote K be the set of all the commodities to be delivered. Each commodity $k \in K$ is defined by a tuple $(Q(k), o(k), d(k), \sigma(k), \tau(k))$, standing for the quantity, origin, destination, available time and deadline respectively. Denote S be a list of time-continuous shifts in the planning horizon and s be the s-th shift in S. All trucks have an identical capacity of 1 unit. Therefore, commodities are shipped directly to their destinations without transfers or consolidation.

Denote t_j be the service time at node j. Note that t_j is dependent on both the node a vehicle visits and the types of operation (either loading or unloading) done at this node. Denote t_j^l and t_j^u respectively be the loading

Table 1: The list of notations.

Input Parameters	
V	The set of nodes in the transportation network.
S	A list of time-continuous shifts in the planning horizon.
s	The s^{th} shift in S.
et^s	The beginning time for shift s.
lt^s	The end time for shift s.
d_{ij}	The distance between nodes $i \in V$ and $j \in V$.
μ_{ij}	The travel time between nodes $i \in V$ and $j \in V$.
t_j	The service time at node j.
t_j^l	The loading time at node j.
t_j^u	The unloading time at node j.
n	The number of trucks available for use.
R	Set of feasible truck routes within a shift. Each truck starts from depot v_0 and returns to depot before shift ends.
d_r	The distance of the route $r \in R$.
K	A set of commodities to be delivered. Each commodity is delivered by exactly one truck.
$Q(k)$	The quantity of the standard commodity k.
$o(k)$	The origin of the commodity $k \in K$.
$d(k)$	The destination of the commodity $k \in K$.
$\sigma(k)$	The available time of the commodity $k \in K$.
$\tau(k)$	The deadline or completion time by which the commodity $k \in K$ has to be serviced.
$\delta_{r^i}^k$	A binary constant indicating whether k can be serviced at the ith node in $r \in R$.
$e_{r^i}^s$	Earliest time that truck route r finishes a service (either a drop-off or a pickup) at its ith node in shift s.
$l_{r^i}^s$	The latest time that truck route r may depart from its ith node in shift s.
M	A sufficiently large positive number.
Decision Variables	
y_r^s	The number of times a given route $r \in R$ is used during shift s and $y_r^s \in \mathbf{N}^+$.
$x_{r^i}^{ks}$	Binary variable to indicate whether the ith node of r in s is used as the starting node for servicing commodity k.

and unloading time at node j, then we have

$$t_j = \begin{cases} t_j^l & \text{if loading operation only at node } j \\ t_j^u & \text{if unloading operation only at node } j, \\ t_j^l + t_j^u & \text{if both loading and unloading at node } j. \end{cases} \tag{1}$$

The problem is to find a set of vehicle routes with minimum total costs to deliver all the commodities within their time windows. Each vehicle route should depart from the depot at the start of a shift and return to the depot before the shift ends.

3 Literature Review

As stated in Section 1, the research on container transportation is somewhat limited despite numerous publications on other types of freight transport problems. To help understand the core features of this problem, it is possible to broadly classify freight transportation into `consolidated` transport and `non-consolidated` transport problems. In consolidated transportation, freights can be split and shipped via multiple paths of a service network and freights may get transferred and consolidated along some of the paths. That is, during the process, some nodes in the service network act as hubs or consolidation centres and a package may be transported by multiple vehicles before arriving at its destination. In non-consolidated freight transportation, freight is delivered to its destination directly, in its entirety, by a single vehicle.

For both the consolidated and non-consolidated transports, the shipment of the freight can be `single-directional` or `bi-directional`. In single-directional transport, each node in the transportation network is either a `supply` node or a `demand` node but not both while the nodes in a bi-directional transport can both be supplies as well as demands. Table 2 gives typical examples for each type of transportation problems. Single-directional consolidated transportation includes the classical production logistics (in which necessary raw materials, parts and sub-parts of products are consolidated and transported to the production factories) and deliveries for fresh produce and hazardous materials that require special vehicles. Single-directional, non-consolidated transport problems are studied intensively in the forms of several vehicle routing problem variants (Toth and Vigo, 2001), including capacitated vehicle routing problem (CVRP), vehicle routing with time windows (VRPTW), multi-depot vehicle routing problem (MDVRP), etc. With regards to bidirectional, consolidated transportation, service network design research (Bai et al., 2012b) for less-than-truckload transport, express delivery and postal mail delivery are typical examples. In terms of the search space and complexity, this category probably represents the most challenging freight transportation problem due to the huge size of the search space. The final category is the bidirectional, non-consolidated transportation. Typical examples include vehicle routing with pickup and delivery

Table 2: A possible classification of freight transportation problems.

	Consolidated	**Non-consolidated**
Single-directional	Production supply chain, Fresh produce delivery, Hazards transport, etc.	CVRP, VRPTW, TSP, Multi-depot VRP
Bidirectional	Service network design, Postal mail delivery, Less-than truckload transport (LTL), Express delivery	Vehicle routing with pickup and delivery, Full truckload transport (FTL), Container transport, Dial-a-ride problem, etc.

(Min, 1989), full truckload transport (Liu et al., 2010), and container transport which is a special case of the full truckload transport problem. The problem that is considered in this paper falls into this final category. The remainder of this section provides a review of existing research work for the final bidirectional, non-consolidated transportation research with a special focus on the container transportation problems.

3.1 Vehicle routing problem with pickup and delivery

The vehicle routing with pickups and deliveries (VRPPD) differs from classic VRP problems in that some of the nodes are both demand and supply nodes and the flow of the freight at these nodes is, therefore, bidirectional (both incoming and outgoing). The inherent mixed loading capacity constraints in VRPPD often leads to increased computational complexity. A comprehensive review for VRPPD can be found in (Berbeglia et al., 2007). Research by Min (1989) represents one of the first scientific studies on VRPPD. The problem was abstracted from a public library distribution system in Ohio, US and a simple three-phased procedure that resembles the well-know "cluster-first, routing-second" heuristics was developed and compared against the real-world manual solution. Pisinger and Ropke (2007) proposed a generic adaptive large neighbourhood search (ALNS) meta-heuristic for 4 variants of the VRP problems with competitive results reported for all variants. The proposed ALNS shares many common features to the simulated annealing hyper-heuristics (Bai et al., 2012a) that was shown successful for the course-work timetabling and the well-known bin packing problem. Gutierrez-Jarpa et al. (2010) studied a variant of VRPPD in which the pickups are selective while deliveries are compulsory. A branch-and-price algorithm was devel-

oped which could solve instances containing up to 50 customers optimally. Derigs et al. (2012) studied a real-life full truckload routing problem arising in timber transportation and used a multilevel neighbourhood search method to solve the problem. Liu et al. (2013) studied a vehicle scheduling problem encountered in home health care logistics. A genetic algorithm and a tabu search method were proposed for this problem. The method was tested on the benchmarks for the VRP with mixed backhauls and time windows (VRPMBTW) against existing best solutions and obtained solutions that are better than the best-known solutions in the literature. Pandelis et al. (2013) studied capacitated VRPPD in which finite and infinite-horizon single VRP with a predefined customer sequence and pickup and delivery is considered. A a special-purpose dynamic programming algorithm that determines the optimal policy was developed. Zhang et al. (2014) studied time dependent vehicle routing problems with simultaneous pickup and delivery by formulating this problem as a mixed integer programming model. A hybrid algorithm that integrates an ant colony algorithm and a tabu search method was developed and the computational results suggest that the hybrid algorithm outperforms stand-alone ant colony algorithm and tabu search. Chen et al. (2014) studied the routing problem with unpaired pickup and delivery with split loads for fashion retailer chains. However, the common time window constraints are missing. Both a simple heuristic and a variable neighbourhood search method were proposed. It can be seen that due to the NP-Hard nature of the problem, almost all studies adopted metaheuristics to solve large scale problem instances.

Another type of vehicle routing with pickup and delivery problem that has been studied specifically is the dial-a-ride (DAR) problem. Kirchler and Calvo (2013) proposed a fast algorithm for solving the static Dial-a-Ride Problem (DARP). A granular tabu Search method has been applied for the first time to solve this kind of problem. Paquette et al. (2013) developed a multicriteria heuristic embedding a tabu search process in order to solve DARPs combining cost and quality of service criteria. This is the first study that handles more than two criteria for this type of problem. Ferrucci and Bock (2014) introduced dynamic pickup and delivery problem with real-time control (DPDPRC) in order to map urgent real-world transportation services. A tabu search algorithm was proposed and computational result showed that newly arriving requests, traffic congestion, and vehicle disturbances can be efficiently handled by this approach. Braekers et al. (2014a) considered a multi-depot heterogeneous dial-a-ride Problem (MD-H-DARP) in real life. A exact branch-and-cut algorithm and a deterministic annealing meta-heuristic were developed for solving small and large problems respectively.

3.2 Bidirectional full truckload transport

In bidirectional full truckload transport, commodities are shipped to destinations in their entirety without intermediate stops or transhipment. Therefore it is different from VRPPD since some of the commodities in VRPPD go through intermediate nodes before reaching their destinations. Truck container transport is a typical example of such a problem since the size of a container is equivalent to the capacity of the truck (hence full truckload) and flow of containers (either loaded or empty) can be bidirectional. Therefore, the solution methods for VRPPD cannot directly be used for the truck container transport problem or even if some variants are applicable, their performance will not be as good since important features (e.g. no intermediate stops) are not exploited fully in the algorithms designed for VRPPD. Even for the truck container transport problems, different characteristics will lead to different problems. From the no-free-lunch theorem of Wolpert and Macready (1997), we know that it is unlikely to develop a generic algorithm, performing best for all possible instances.

Zhang et al. (2010) proposed a nonlinear model based on a preparative graph for a container transportation between shippers, receivers, depots and terminals. A solution method was designed by improving the time window partitioning scheme used in (Wang and Regan, 2002) for a multiple travelling salesman problem with time windows (m-TSPTW). The empirical results for a set of randomly generated instances indicate that improved performance can be achieved compared with a reactive tabu method in (Zhang et al., 2009). The method is effective for small instances but may suffer for large scale problems since the size of the graph can explode with increase in the number of shipments and nodes. Similar issues exist for instances with very wide time windows (e.g. time windows that spans over a few days) due to the time partitioning scheme adopted in the method.

Nossack and Pesch (2013) presented a new formulation for the truck scheduling problem based on a Full-Truckload Pickup and Delivery Problem with Time Windows and propose a 2-stage heuristic solution approach. The results of computational experiments indicate that their 2-stage heuristic outperforms the WPB method applied by Zhang et al. (2010) in terms of computational efficiency. Braekers et al. (2013) investigated a two-stage deterministic annealing algorithm for a full truckload transport problem with simultaneous pickup and delivery nodes. The problem was formulated as an asymmetric m-TSPTW. Similarly the problem was tested for a set of randomly generated instances with commodity time windows ranging between 60 to 240 min, which is much smaller than those in our problems. Better results were obtained using the algorithm than those given by the method of

Zhang et al. (2010). Most research studies assumed a constant travel time among the transportation network which is not always realistic. Therefore, Braekers et al. (2012) studied how time-dependent travel times will affect the full truckload transport planning and scheduling, in which the optimal departure times become decision variables in addition to the routing variables. In real life of drayage operation, shippers may request empty containers to be delivered while consignees may have empty containers available to be picked up. By considering this, Braekers et al. (2014b) studied vehicle routes performing all loaded and empty container transports in the service area of one or several container terminals during a single day. A bi-objective approach (minimising the number of vehicles and minimising total distance travelled) is considered and it is shown that this method obtained considerably better results than those reported in (Braekers et al., 2013). Sterzik and Kopfer (2013) proposed a single-shift general model for transporting both full and empty containers among multiple nodes (depots, terminals and customers). A tabu search heuristic is developed and tested on instances that contain up to 5 depots, 3 terminals and 75 loads with a one-day planning horizon.

It can be seen that there are a number of research studies on full truckload/container transport problems with several models and algorithms being proposed. However, none of them can be directly used to solve the problem described in Section 2. The reasons are: 1) the planning horizon of our problem is much longer than those in the previous studies. This is because the time window of shipments in our problem spans from 1 hour to up to 3 days. The time-window partitioning approach will lead to a huge graph that is prohibitively large to solve. 2) the number of shipments is significantly larger than the instances tested in the previous studies while the number of physical nodes is relatively small (10). The existing approach could not exploit these structures explicitly. 3) finally the operation in our problem is shift based (each shift is 12 hours). Although a shift can be interpreted as a time window for truck, its actual width is not necessarily bigger than the time windows of shipments, and inconveniently most VRP solution methods assumed a bigger truck time window than shipment time window. These issues lead us to consider a different approach which can fully exploit the structures of the problem, and hopefully can be more efficient than the existing approach.

In this paper, we proposed a set covering linear model for the container transport problem. By exploiting the special structure of the problem and introducing a novel formulation, we show that the model can be solved to near optimality for most of real-life instances at industrial sizes. The approach in this paper complements well with Chen et al. (2013)'s variable neighbourhood search metaheuristic method which can solve really large problem instances very quickly but may get stuck to local optima sometimes. We now describe

our model in the next section.

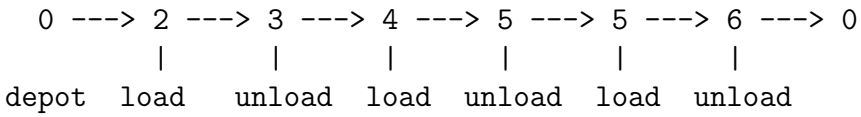
4 Model Formulation and Solving

We now describe our proposed formulation for this problem. Our formulation is similar to the classic set-covering model with additional side constraints. The underlining idea is to find a subset of truck routes (from all possible feasible routes) that sufficiently `covers` all the transportation demands with a minimum total cost (i.e. distance). Because of the fact that all shifts are of identical periods and all the trucks must depart from the depot at the beginning of every shift and return to the depot before the shift ends, the feasible route set is the same for all shifts, assuming the travel times and service times are the same at different shifts (see the next section how operation dependent service times can be transformed to conform with this assumption). Therefore, the first issue of the model development is the generation of a set of feasible truck routes. The list of notations used in the paper is given in Table 1.

4.1 Feasible route generation

For a given directed graph $G = (V, A)$ where V is the set of nodes, representing different freight forward terminals and A is the set of arcs between nodes. Let node 0 be the depot. A feasible route is defined as a sequence of nodes that a truck can cover in a shift. Since no transshipment is permitted in the operation, for any feasible route, we ensure that each node will have at least one operation (i.e. either loading or unloading) with some nodes involving both operations simultaneously. Since time taken for loading/unloading operations is substantial and is comparable to the travel time between nodes, the service time at each node in a truck route will depend on actual commodity shipments along the route. The service time for nodes involving both of the operations will be much bigger than the service time if only one operation is scheduled at this node. This creates a very challenging issue for modelling since the service time is no longer a constant and depends on the actual solution. To circumvent this problem, for any node that involves both of the operations, we insert a copy of the node immediately after it in the route, setting the distance between them to 0 but an unloading service time for the first copy and a loading time for the second. In this way, all the routes now have exactly one operation per node except the depot. Because each unit of commodity shipment involves exactly two operations (i.e. loading at the source node and then unloading at the destination node) and a truck

would never visit a node without a service, each of the feasible routes should contain an even number of nodes (including nodes copies). The following is an example of a feasible route.



In this particular route, a truck departs from the depot and picks up a commodity of unit quantity from node 2, and unloads the commodity at node 3. Then the truck picks up another commodity at node 4, drops it off at node 5. The final commodity delivered by this truck is from node 5 to node 6 before the truck returns to the depot. Therefore, in this route, in addition to truck movements to and from the depot, the truck movement from node 3 to node 4 is also empty. At node 5, the truck does both unloading and loading since it has two copies in the route. Excluding depot, odd numbered nodes are loading nodes and even numbered nodes have unloading operations only. Therefore, the service time of each node will be determined by the index of the node in the route. For an 0-indexed route, the service time of an odd-numbered node equals the loading time and even-numbered node has service time equalling its unloading time. Denote r^i be the i-th node in a feasible route r and t_{r^i} be the service time at node r^i:

$$t_{r^i} = \begin{cases} 0 & \text{if } r^i \text{ is depot,} \\ t^l_{r^i} & \text{if } r^i \text{ is an odd-numbered node in } r, \\ t^u_{r^i} & \text{if } r^i \text{ is an even-numbered node in } r. \end{cases}$$

where $t^l_{r^i}$ and $t^u_{r^i}$ are loading and unloading times at node r^i. With the route representation introduced above, we can now develop an integer model as follows. We will discuss later the algorithm to generate all feasible routes.

Denote R the set of all possible feasible routes within a shift and K the set of commodities and S the set of shifts within the planning horizon. Here each commodity $k \in K$ represents a number of containers with same properties defined by tuple $\{s(k), d(k), \sigma(k), \tau(k), Q(k)\}$, standing for its source, destination, time of arrival at port, deadline for shipment, and its quantity respectively. Note that in this application, we consolidate the quantity of commodity so that each truck carries one unit of a commodity exactly. For the real-life problem under consideration in this paper, one unit of a commodity means two small containers (20 inch) or 1 large container (40 inch).

The solution can be encoded into two decision variables $x_{r^i}^{ks}$ and y_r^s. The first variable defines whether the ith node of a route $r \in R$ is used as the loading node for commodity $k \in K$ during shift $s \in S$ while the latter defines the frequency of route r being used in a given shift s. Therefore, a given commodity k could potentially be serviced by several arcs of a route in different shifts, subject to constraints of time windows $(\sigma(k), \tau(k))$ and source-destination pairs being matched up between the arc and the commodity.

In order to speedup the processing time, one could pre-process all the possible arcs in each of the feasible routes for a given commodity. For each of the feasible route $r \in R$ and a given shift $s \in S$, a binary variable $\delta_{r^i}^{ks}$ is introduced to indicate whether the ith node in route r of shift s can be used as the starting service node for commodity k. Therefore, $\delta_{r^i}^{ks} = 1$ means that the following conditions should be satisfied, otherwise it is set to 0.

$$
\begin{aligned}
i \ \texttt{mod}\ 2 &= 1 && (2)\\
r^i &= o(k) && (3)\\
r^{i+1} &= d(k) && (4)\\
l_{r^i}^s &\geq \sigma(k) + t_{r^i} && (5)\\
e_{r^{i+1}}^s &\leq \tau(k) && (6)
\end{aligned}
$$

Condition (2) indicate that starting service node must be the node with loading action. Condition (3) and (4) define source and destination of commodity for starting service node i. In constraints (5) and (6), $l_{r^i}^s$ is the latest departure time from the i-th node of route r in shift s to ensure all the subsequent services can be delivered on time. Similarly $e_{r^{i+1}}^s$ is the earliest time that a truck can possibly arrive at the $(i+1)$th node of r during shift s. $l_{r^i}^s$ and $e_{r^i}^s$ can be pre-calculated as follows:

$$
\begin{aligned}
e_{r^0}^s &= et^s && (7)\\
e_{r^i}^s &= e_{r^{i-1}}^s + \mu_{r^{i-1}r^i} + t_{r^{i-1}} && (8)\\
l_{r^{\bar{0}}} &= lt^s && (9)\\
l_{r^i}^s &= l_{r^{i+1}}^s - \mu_{r^i r^{i+1}} - t_{r^{i+1}} && (10)
\end{aligned}
$$

where et^s and lt^s are the beginning and ending time for shift s respectively and $r^{\bar{0}}$ denotes the final node in route r (i.e. the depot). Eqs. (7) and (9) provide initial values for recursive equations (8) and (10).

4.2 Model formulation based on set covering

Once the feasible route set R is constructed, our problem can be formulated as finding a subset of R for each of the shifts such that all the tasks are `covered` (or serviced) on time and the total routing cost is minimised.

Denote $x_{r^i}^{ks}$ and y_r^s the two decision variables. y_r^s is the frequency of route r used in the s^{th} shift in a solution. Variable $x_{r^i}^{ks}$ denotes, in a given solution, whether the i^{th} node in route r is used to as the departure node for servicing task k in the s^{th} shift. The problem can be formally defined as follows:

$$\min \sum_s \sum_r d_r y_r^s \tag{11}$$

subject to

$$\begin{aligned}
\sum_r y_r^s &\leq n \quad \forall s \in S && (12)\\
\sum_s \sum_r \sum_i x_{r^i}^{ks} &= Q(k) \quad \forall k \in K && (13)\\
\sum_k x_{r^i}^{ks} &\leq y_r^s \quad \forall i \in r, \forall r \in R, \forall s \in S && (14)\\
x_{r^i}^{ks} &\leq \delta_{r^i}^k \quad \forall i \in r, \forall r \in R, \forall k \in K, \forall s \in S && (15)\\
l_{r^i}^s &\geq x_{r^i}^{ks}[\sigma(k) + t_{r^i}] \quad \forall i \in r \backslash 0, \forall r \in R, \forall k \in K, \forall s \in S && (16)\\
e_{r^{i+1}}^s &\leq x_{r^i}^{ks}\tau(k) \quad \forall i \in r \backslash 0, \forall r \in R, \forall k \in K, \forall s \in S && (17)\\
x_{r^i}^{ks} &= \{0,1\} \quad \forall i \in r, \forall r \in R, \forall k \in K, \forall s \in S && (18)\\
y_r^s &\in \mathbb{Z}^+ \quad \forall r \in R, \forall s \in S && (19)
\end{aligned}$$

The objective is to minimise the total distance of all routes used in a solution. Constraints (12) ensures the availability of trucks the company actually possesses. Constraints (13) ensures all the tasks are serviced. Constraints (15) makes sure that any positive $x_{r^i}^{ks}$ is feasible in terms of the source and destination of task k. Constraints (16) and (17) ensure that the time windows of each task are satisfied. That is: the latest departure time of a route at its i^{th} node should be no earlier than the available time of task k plus the loading time if ith node of r is used to service task k; and the earliest arrival at the $(i+1)$th node of route r should be no later than the deadline of task k, should the i^{th} node of route r is used as the departure node to service task k.

Different from other Travelling Salesmen Problem based formulations that represent each load as a node in a network (same as (Zhang et al., 2010)), the above model treats commodities as `flows` to be covered by routes. This

makes the proposed model advantageous compared to the other alternative methods. In real-life instances, it is common that large number of containers arrives with a same S/D pair and a same time window. For the above model, the complexity of solving a problem instance with $Q(k) = 10$ would be similar to the instance with $Q(k) = 100$. However, the latter instance would be multitude times harder to solve for Zhang et al. (2010)'s method.

Note that in the optimal solution of the model, variable y_r^s gives the optimal number of times that a given feasible route should be used in each shift. For clarity, we call each usage of a given route r an `instance` of r. Practically each instance of a route can be associated with a truck for implementation. Variable $x_{r^i}^{ks}$ provides the information of which route arcs are being utilised to service each commodity. Since there may exist several instances of a feasible route in the optimal solution, some of the arcs of the route may be assigned to service several commodities (although the arcs of each route instance still service a maximum of 1 commodity). However how flows of these commodities are distributed between different instances of the route arcs is unknown. In fact there exists multiple feasible distributions but all of them will lead to a same objective function. We will discuss in Section 4.4 how to obtain these distributions to recover the detailed solution. This is another advantage of this model since it is able to reduce the size of the search space by evaluating all feasible flow distributions on shared arcs only once.

4.3 Data pre-processing

The original problem contains a total of 9 docks/nodes (see Figure 1). The depot is at node BLCT2. We applied a recursive algorithm to generate all feasible routes that satisfy all the constraints in Section 4.1. Since the travelling time between some nodes are very short (e.g. 5 to 15 minutes), the algorithm generated millions of feasible routes on an 12-hour shift. This huge size of feasible routes will lead to prohibitively long solution time for our model in Section 4.2. Therefore, the original data was preprocessed before being populated into the model. Historical data show that node ZHCT always has very few shipments. In order to reduce the total number of feasible truck routes, we take out all the shipments related to node ZHCT. In addition, all the nodes that are within 15 min travel times are merged into super nodes and its servicing time is set as the mean service time of the two original nodes. In the end, our reduced problem has a total of 6 nodes, including two super nodes {BLCTZS, DXCTE} and {BLCT3, BLCTYD} and the depot. We did not merge the depot with BLCT2 as we need the depot in our route representation. Based on these 6 nodes, the feasible

route generator returns a total of 43081 feasible routes. We then exclude all the commodities within the super nodes from the reduced problem. These commodities will be inserted into the final solution during the final stage of the proposed method.

4.4 Recover the full solution for the reduced problem

As mentioned in section 4.2, the solution obtained from the model gives the number of times each feasible route is used, and what commodities are serviced at each of the route arcs. If a route is used multiple times (specified by y_r^s) in a solution, normally some of the arcs in the route are used to service multiple commodities (i.e. for a given r_i and s, $x_{r^i}^{ks}$ takes value of 1 for more than one commodity). However, for practical applications we need to specify the exact allocation of flows between different route instances (see the last paragraph in Section 4.2), not just between different routes. For example, assume there is a route $0 \to 5(k1, k2) \to 2 \to 3(k3) \to 4 \to 0$ and the frequency of the route for shift s (i.e. y_r^s) is 8. Node 5 is used as the starting service node for two commodities $k1$ and $k2$, and node 3 is used as the starting service node for commodity $k3$. So here arc $5 \to 2$ is shared for shipping two commodities while arc $3 \to 4$ is used to service commodity $k3$ alone. We defined the arcs such as $5 \to 2$ as `shared arcs` and arcs such as $3 \to 4$ `exclusive arcs`. Therefore, entire 8 unit capacity of arc $3 \to 4$ can be used to service commodity $k3$. However, the 8 unit capacity of arc $5 \to 2$ is shared between commodities $k1$ and $k2$ and the actual splits between them can be in many ways. In order to obtain all the feasible allocation of capacity on shared arcs, we can solve the following linear equations.

First we denote A be the set of all the route arcs and $\bar{A}$ the set of all the *shared arcs* like $5 \to 2$. Here the arcs from different routes but with identical source-destination pairs are considered different. Let $\bar{K}$ be the set of commodities that are serviced by at least one of the *shared arcs* and $Q^*(k)$ be the total amount of flow of commodity k that is serviced by the *exclusive arcs*. Denote variable z_a^{ks} be the amount of flow of commodity k on a given route arc $a \in A$ in shift s, then z_a^{ks} can be obtained by solving the following linear equations.

$$\sum_{s \in S} \sum_{a \in \bar{A}} z_a^{ks} + Q^*(k) \quad = \quad Q(k) \quad \forall k \in \bar{K} \tag{20}$$

$$\sum_{k \in \bar{K}} z_a^{ks} \quad <= \quad y_a^s \quad \forall a \in \bar{A}, \forall s \in S \tag{21}$$

where y_a^s is the optimal solution of y_r^s from the integer programming model

for the route containing arc a. Equations (20) ensure that the entirety of each commodity is serviced by a combination of the shared arcs and exclusive arcs. Equations (21) make sure that the total flow on each shared arc should not be more than the total capacity.

4.5 Inserting unserviced commodities

In the reduced problem, we excluded the nodes that have very few shipments and combined some nodes into super-nodes (and the shipments within the super nodes are excluded accordingly). Therefore, these commodities need to be inserted into the solution obtained from the reduced problem. For each route in the current solution, a total of 4 conditions have to be satisfied before a commodity is inserted into this route. Firstly, the route must have enough remaining time for additional commodities. Secondly, the insertion of a commodity must not affect the feasibility of the solution. Thirdly, the deadline of commodities within the super-nodes should be satisfied. Since the nodes in the super nodes are very close, most of intra-node shipments can successfully be inserted into the current solution. In a few cases where there is no feasible insertion, the procedure opens a new route. Finally, when multiple insertion points are available, the procedure favours the one that leads to the least empty load distance.

In summary, our approach is a three-stage hybrid method. 1) `Pre-processing`: the original problem is reduced to a problem with a smaller number of nodes and commodities. 2) `Solving the reduced problem`: the reduced problem is solved in Gurobi solver and the full solution (including the flow distribution among route instances) for the reduced problem is recovered by solving the linear equations described in Section 4.4. 3) `Post-processing`: finally the commodities that were temporarily excluded from the reduced problem were heuristically inserted into the existing routes whenever possible. A new truck route is opened if there are commodities that cannot be assigned to any existing routes.

5 Benchmark Instances and Experimental Tests

In order to evaluate the feasibility and performance of our model, we applied it to solve real-life instances at a large international port in China. In addition, test instances with certain features were created to fully assess the approach and to gain knowledge that may not be discovered from real-life instances. These instances can be downloaded from http://www.cs.nott.ac.uk /~rzb/research/transport/data/nbport.zip.

5.1 Benchmark instances

5.1.1 Real-life instances

A total of 15 real-life instances were extracted from the real-life problem data provided by the company. The original data contains three month demands from February to May 2012. Since the time window of most shipments ranges from 2 to 8 shifts, these instances have three planning horizons of 4, 6 and 8 shifts (the instance name, planning horizon and commodity size are given in Table 3).

Table 3: Some details of the 15 real-life instances.

instance	**no. of shifts**	**total commodity units**
NP4-1	4	465
NP4-2	4	405
NP4-3	4	526
NP4-4	4	565
NP4-5	4	765
NP6-1	6	1073
NP6-2	6	920
NP6-3	6	384
NP6-4	6	746
NP6-5	6	557
NP8-1	8	913
NP8-2	8	827
NP8-3	8	786
NP8-4	8	1008
NP8-5	8	798

5.1.2 Artificial instances

In addition to the real-life instances, we have also created a total of 17 artificial instances with controlled demand parameters in terms of commodity quantity, load (im-)balance and time windows. The number of available trucks is set to $n = 100$. The other parameters (nodes, distance matrix, time matrix, operation time, etc.) remain the same. More specifically, we distinguish emergent tasks and non-emergent tasks. A transportation task is defined as `emergent` if the difference of its available time and deadline is less than 10 hours [1]. we also measure whether the transportation demand of

[1]The 10-hour threshold is based on consultations with the port operators.

a problem is balanced or not in both space and time through the following index:

$$B = \frac{1}{|V|} \sum_{i=1}^{|V|} \sum_{s \in S} |I_i^s - O_i^s|$$

where $|V|$ is the total number of physical nodes (i.e. docks). I_i^s and O_i^s are respectively the total incoming and outgoing commodities at node i during shift s. The following 4 types of features are used in creating the test instances.

- **Tight** instance: an instance is considered having "tight" time-windows if 70%-80% of its commodities are emergent.
- **Loose** instance: an instance is defined as "loose" if up to 30% of its tasks are emergent.
- **Balanced** instances are defined by a balance index B. An instance is "balanced" if B is no more than 30.
- **Unbalanced** instance has a balance index B great than 30.

With 2 different planning horizons (4, and 8 shifts), we have a total of 8 combinations. For each combination, 2 instances are created, resulting in a total of 16 instances. In addition, we also created a very large instance with 8 shifts and 2000 commodities. Details of these instances are given in Table 4.

5.2 Computational results

Gurobi 5.6 was used to solve the reduced problem directly with the default algorithm setting. The experiments were run on a PC with Intel i7 3.40GHZ processor (8 cores) and 16GB RAM. Although the total distance is chosen as the objective for the mathematical model (see Section 4.2), the final solution is evaluated in terms of the heavy load distance rate (HLDR), which is the preferred efficiency indicator in practice. HLDR is defined as follows:

$$HLDR = \frac{\text{loaded distance}}{\text{total travel distance}} \tag{22}$$

Since containers are shipped to their destinations directly, the total amount of the loaded distance is fixed for a given commodity set K. Therefore, HLDR is equivalent to the objective function (11) as far as optimisation is concerned since minimising the total travel distance will also improve HLDR.

Table 4: The list of artificial instances

instance	**configuration**	**no. of shifts**	**total commodity units**
LB4-1	Loose,Balanced	4	484
LB4-2	Loose,Balanced	4	396
TB4-3	Tight, Balanced	4	282
TB4-4	Tight, Balanced	4	368
LU4-5	Loose,Unbalanced	4	448
LU4-6	Loose,Unbalanced	4	479
TU4-7	Tight, Unbalanced	4	217
TU4-8	Tight, Unbalanced	4	354
LB8-1	Loose,Balanced	8	592
LB8-2	Loose,Balanced	8	657
TB8-3	Tight, Balanced	8	497
TB8-4	Tight, Balanced	8	621
LU8-5	Loose,Unbalanced	8	551
LU8-6	Loose,Unbalanced	8	559
TU8-7	Tight, Unbalanced	8	607
TU8-8	Tight, Unbalanced	8	525
Large	Mixed, Unbalanced	8	2614

For brevity, we denote our three-stage method as `hybrid` method. We compare its results against a reactive shaking variable neighbourhood search (VNS) and a simulated annealing hyper-heuristic method (SAHH) (Chen et al., 2013). VNS and SAHH were run on a PC with 2.8GHz Xeon processor and 5GM memory. The computational time limit is 20 minutes per shift for VNS and 15 minutes per shift for SAHH. Therefore, for a 4-shift instance, VNS requires 80 minutes and SAHH requires 60 minutes.

5.2.1 Computational results for real-life instances

Table 5: The HLDR results of our hybrid method for 4-shift instances.

	NP4-1	NP4-2	NP4-3	NP4-4	NP4-5
total distance	13508.5	16635.5	16878.5	21886	26731
time(s)	33301	15742	11178	18537	20647
hybrid	**89.2%**	69.2%	**78.6%**	**70.0%**	79.3%
VNS	83.2%	69.2%	77.1%	68.5%	80.7%
SAHH	83.2%	**69.3%**	76.2%	69.0%	**80.8%**

Table 6: The HLDR results of our hybrid method for 6-shift instances.

	NP6-1	NP6-2	NP6-3	NP6-4	NP6-5
total distance	34054.5	33316	16191.5	26260	16880.5
time(s)	160079	138486	3978	58898	104446
hybrid	**82.7%**	**75.0%**	**66.1%**	**80.5%**	**83.2%**
VNS	79.3%	72.9%	64.2%	80.3%	77.7%
SAHH	80.5%	74.1%	65.8%	80.2%	78.8%

Table 7: The HLDR results of our hybrid method for 8-shift instances.

	NP8-1	NP8-2	NP8-3	NP8-4	NP8-5
total distance	35685	30633	28314	44224	25451.5
time(s)	148067	147241	121074	66438	131369
hybrid	72.5%	**76.8%**	**76.7%**	61.7%	**75.8%**
VNS	74.6%	74.1%	76.1%	**62.1%**	74.6%
SAHH	**74.7%**	74.7%	76.0%	**62.1%**	73.3%

The computational results for the real-life instances are given in Tables 5,6 and 7. Values in **bold** represent the best results. It can be seen from the table that for most of the instances, the hybrid 3-stage method outperformed both VNS and SAHH. Taking NP4-1 as an example, the improvement is as much as 6.0%, which translates into nearly 1000km saving in distance in 2 days. We can be fairly confident in saying that, overall, the proposed hybrid method could produce much better solutions to these real-life instances when compared to the recent multi-neighbourhood metaheuristic approaches.

For 4 instances (NP4-2, NP4-5, NP8-1, and NP8-4), the hybrid method is outperformed by either SAHH or VNS. However, the margin is relatively small. Note that our hybrid method does not guarantee the optimal solution because of the approximations made in the first and last stage of the approach. More specifically, in the first stage of the hybrid method (see Section 4.3), nodes BLCTZS and DXCTE were merged as a super node and its service time was set to the mean value of the service times of BLCTZS and DXCTE. However, because the service times for BLCTZS and DXCTE are very different (in our setting, they are 60 minutes and 5 minutes respectively), using the mean service time (33 minutes) could lead to either infeasibility (which needs to be handled in stage 3) or inferior solutions. In order to mitigate this problem, we replaced the simple mean service time for the super node to the weighted average service time, with the weights

being proportional to the number of operations at the corresponding physical nodes. Therefore, all the remaining experiments in Section 5.2.2 were conducted with this new setting.

5.2.2 Computational results for artificial instances

Table 8: The computational results by the hybrid method for artificial instances in comparison with VNS and SAHH (Chen et al., 2013).

instance	**hybrid method**			**VNS**	**SAHH**
	HLDR(%)	distance	time(s)	HLDR(%)	HLDR(%)
LB4-1	**77.6**	15763.0	13438	77.1	76.4
LB4-2	**83.4**	14319.0	3812	79.3	78.1
TB4-3	**68.9**	10866.5	1415	67.5	67.9
TB4-4	**72.4**	12507.5	186	67.1	66.7
LU4-5	**63.2**	18499.5	1590	59.3	58.8
LU4-6	65.3	20315.5	1783	66.8	**67.2**
TU4-7	**49.2**	13032.5	79	46.6	46.6
TU4-8	**54.3**	17024.5	138	51.8	51.8
LB8-1	**95.1**	18132.5	138988	94.1	93.0
LB8-2	**88.7**	22834.0	157354	88.1	87.8
TB8-3	**67.8**	21337.5	148	66.7	66.8
TB8-4	**61.6**	28167.0	561	61.3	61.1
LU8-5	**71.7**	21226	4380	61.4	61.9
LU8-6	**67.9**	23261.0	13202	65.1	64.7
TU8-7	**60.9**	31094.0	140	53.2	53.2
TU8-8	**49.9**	27406.0	66	48.5	48.5
Large	n.a.*	n.a.*	48h	56.1	**56.5**

*:The algorithm fails to solve the problem after 48 hours.

The computational results for the artificial instances by different algorithms are given in Table 8. The best results are highlighted in **bold**. It can be seen that the 3-stage hybrid method was able to obtain best overall objective results for all the instances except two, whose performance seems to be improved further by the weighted average service time. Again the better performance was achieved at the expenses of more computational time. Generally, the proposed method can solve most "tight" instances fairly quickly but struggled for some of "loose" instances. This is not surprising since "tight" time-window constraints actually help speedup the search by producing better bounds for an integer programming solver. This is in direct contrast to

the metaheuristic methods which often struggle for highly constrained instances. When the problem size, in terms of commodity size, increased to over 2000, the proposed hybrid method failed to solve the problem while both VNS and SAHH can still produce feasible solutions with significantly less computational time. In this sense, the proposed hybrid method is not a complete replacement for, but rather a nice complementation to the existing metaheuristic methods. The proposed algorithm would be the preferred solution method for small or tightly constrained instances while large and less constrained instances should be solved by the existing metaheuristics.

In terms of the transportation efficiency (i.e. HLDR), it can be observed that generally a balanced demand can contribute to a higher HLDR, which is consistent with the observation made by Chen et al. (2013). Also it can be seen that "tightness" in the time window of transportation tasks affected the HLDR negatively. The reasons are twofold: firstly, similar to the vehicle routing problem with time windows, there are less flexibilities to coordinate different transportation loads to improve HLDR when a task is highly constrained by time. Secondly, although overall transportation demand in each shift may be balanced, tight time windows of tasks will cause unbalanced demand during that particular time window, which leads to a low HLDR.

5.3 Computational time

Table 9: A comparison of HLDR results by the hybrid mehtod against metaheuristics with longer computational time (slow version). The results of the fast version of VNS and SAHH are from Section 5.2.

			Fast version		**Slow version**	
instance	**time (s)**	**hybird**	**VNS**	**SAHH**	**VNS**	**SAHH**
NP4-1	33301	**89.2**	83.2	83.2	82.9	83.2
NP6-3	3978	**66.1**	64.2	65.8	63.3	65.3
NP8-3	121074	**76.7**	76.1	76.0	76.0	76.0
LB4-1	13438	**77.6**	77.1	76.4	76.6	76.8
LB4-2	3812	**83.4**	79.3	78.1	79.4	79.4
LB8-1	138988	**95.1**	94.1	93.0	92.9	91.6
LB8-2	157354	88.7	88.1	87.8	89.9	**90.8**
LU8-5	4380	**71.7**	61.4	61.9	67.6	67.7
LU8-6	13202	**67.9**	65.1	64.7	64.2	66.0

As indicated in the previous section, the hybrid method requires more than 10 hours computation for many instances (except some of tightly time-constrained instances). However, the computational time by both VNS and

SAHH is less than 20 minutes per shift. For a fairer comparison, additional experiments were also carried out with same amount of computational time permitted for VNS and SAHH on some instances. Due to the long experiments time, a subset of 9 instances were selected for this experiment. 3 of them were selected from real-life instances for which the proposed hybrid method outperformed metaheuristics with shorter running time. The rest were chosen from artificial instances for which the hybrid algorithm took longer time in solving than the metaheuristics did. In this particular experiment, both VNS and SAHH were permitted the same amount of running time by the hybrid algorithm (see Table 9) and their results are also given in the same table, along with the previous results with a short computational time (i.e. 20 minutes per shift).

The results showed that, with the additional computational time, both VNS and SAHH are able to improve the results for some instances (e.g. LB8-2, LB8-5). However, for many other instances, they fail to make noticeable improvement. In fact, to our surprise, some of the results are even marginally worse than before (e.g. NP4-1, NP6-3, LB8-1, LU8-6). We believe that this was caused by the fact that the parameters by both VNS and SAHH were finely tuned for the previous setting only and do not perform well for the new setting. This sensitivity in parameters, again, is a common criticism for many metaheuristic approaches. It's interesting to observe that SAHH slightly outperformed VNS both for the fast version and the slow version. We believe this was probably contributed by the learning mechanism within the simulated annealing hyper-heuristic that provides better adaptation than VNS does by dynamically choosing between different neighbourhood functions for different instances and experiment conditions. A more profound analysis of neighbourhood selection and adaptation will be out of the scope of this paper. Readers are encouraged to refer to the latest hyper-heuristic research which has gradually gained more and more research attention recently.

In terms of practical applications, although the hybrid method may be too long for direct utilisation, the problem can be resolved through multi-core parallel computing facilities which have recently become available at acceptable costs either through rented clouding services or through building a moderate low-cost cluster.

6 Lower Bound

Through the computational analysis above it can be seen that the HLDR, which is the main performance indicator used by the company, can be much lower for some instances than others. For practical applications, it is im-

portant to identify the causes for such low HLDR values. Is it because of the nature of the instances or the inability of finding near optimal solutions by our approaches? Our conjecture is that these relatively low HLDR values was caused by the unbalanced demand distribution in space and time nodes. By demand imbalance, we mean the difference between the inbound and outbound demands for each node at a particular shift. In this section, we analyse to what extent that the load imbalance has contributed to this. To do this we solve a simplified problem in which the time window requirements of each shipment (i.e. arrival time and delivery deadline) are relaxed to the corresponding shift in which the time window lies. For ease of modelling, we also neglect the empty truck movements from/to the depot in computing the lower bound.

To do this, we define $\upsilon(k)$ and $\omega(k)$ respectively be the shift that commodity k becomes available and the shift that the delivery deadline of k lies in. Denote u_{ij}^{ks} be the flow of commodity k on arc (i,j) during shift s and v_{ij}^{s} be the number of vehicles covering arc (i,j) during shift s. In addition, the constraint of all trucks returning to the depot is discarded to exclude the factor of inappropriate depot location. The relaxed problem can be formulated as the following service network design problem.

$$\min \sum_{s} \sum_{(i,j)} d_{ij} v_{ij}^{s} \tag{23}$$

subject to

$$\sum_{s=\upsilon(k)}^{\omega(k)} \sum_{j} u_{ij}^{ks} - \sum_{s=\upsilon(k)}^{\omega(k)} \sum_{j} u_{ji}^{ks} = b_i^k \quad \forall k, \forall i \tag{24}$$

$$\sum_{j} v_{ij}^{s} - \sum_{j} v_{ji}^{s} = 0 \quad \forall s, \forall i \neq 0 \tag{25}$$

$$\sum_{k} u_{ij}^{ks} \leq v_{ij}^{s} \quad \forall (i,j), \forall s \tag{26}$$

where constraints (24) are the flow conservation constraints, (25) are the truck balance constraints and (26) are the capacity constraints. Note that the lower bound model (in terms of total distance) has all 9 nodes shown in Figure 1 without merging any node. The bound results (in terms of HLDR) for the 15 real-life are given in Table 10 in comparison with the results obtained through the hybrid method in section 4.2. Similarly Table 11 shows the comparison of the hybrid method with the lower bounds for the random artificial instances.

Table 10: The results of our hybrid algorithm for the real-life instances when compared with the lower bound

instance	**lower bound**	**hybrid algorithm**	**gap(%)***
NP4-1	13322.0	13508.5	1.4
NP4-2	16386.0	16635.5	1.5
NP4-3	16663.5	16878.5	1.3
NP4-4	20754.0	21886.0	5.2
NP4-5	26121.0	26731.0	2.3
NP6-1	33566.0	34054.5	1.4
NP6-2	32550.0	33316.0	2.3
NP6-3	16000.5	16191.5	1.2
NP6-4	26096.5	26260.0	0.6
NP6-5	16639.0	16880.5	1.4
NP8-1	33568.0	35685.0	5.9
NP8-2	30333.0	30633.0	1.0
NP8-3	27420.5	28314.0	3.2
NP8-4	43617.0	44224.0	1.4
NP8-5	25350.0	25451.5	0.4

*gap(%)=(ObjectiveValue-LowerBound)/ObjectiveValue * 100%

Table 11: The results of our hybrid algorithm for the random artificial instances when compared with the lower bound.

instance	**lower bound**	**hybrid algorithm**	**gap(%)**
LB4-1	15383.5	15763.0	2.4
LB4-2	13837.0	14319.0	3.4
TB4-3	8914.0	10866.5	18.0
TB4-4	10198.5	12507.5	18.5
LU4-5	15770.5	18499.5	14.8
LU4-6	17817.0	20315.5	12.3
TU4-7	9998.0	13032.5	23.3
TU4-8	14565.5	17024.5	14.4
LB8-1	17601.5	18132.5	2.9
LB8-2	20656.0	22834.0	9.5
TB8-3	16620.5	21337.5	22.1
TB8-4	18772.0	28167.0	33.4
LU8-5	20507.5	21226.0	3.4
LU8-6	21999.0	23261.0	5.4
TU8-7	26922.5	31094.0	13.4
TU8-8	24227.0	27406.0	11.6

For the real-life instances it can be seen from the Table 10 that the results of the proposed method are very close to the lower bounds. For some instances, the difference (gap%) is smaller than 2%. This is particularly true for the instances with relatively low HLDR (e.g. NP4-2, NP6-3 and NP8-4). It is indeed the demand imbalance that caused low transport efficiency. For some instances, the gaps to the lower bound are bigger (e.g. NP8-1). However, this observation cannot be repeated for the random artificial instances (see Table 11), for which the gap can be as large as 33.4%. This suggests that many artificial instances have very different characteristics to those shown by the real-life instance. Although small gap to the lower bound can prove the high performance of the algorithm, one cannot conclude that a big gap to the lower bound implies poor solutions. This is because the bound for these instances may be poor. Generally the gap is much bigger for tight instances than for loose instances. Indeed, the time window relaxation made in lower bound model leads to very different problems for the original tight instances. In addition, the lower bound model also excluded the constraint of using the depot as the sole departure node at the start of each shift, whose inclusion may have caused long-distance empty truck returning to the depot. For real-life instances, the depot is in fact very close to busier ports and in most cases trucks return to the depot fully loaded. For these cases, tighter bounds are needed. This would be out of the scope of this paper but could be an interesting research direction in the future.

It should be noted that one would only need to use the 3-stage hybrid method when the problem is too large to be handled directly. For small and moderate instances (i.e. less than 7 nodes), only the second stage is required and the exact solutions can be obtained.

7 Conclusions

This paper presents a set covering integer linear programming model for a bidirectional multi-shift full truckload shipment problem. The problem is drawn from a real-life container transshipment problem at a large container port. Because of the special structures of the problem (in particular the much longer planning horizons), existing methods are not efficient for the problem. We have shown that through some pre-processing and post-processing, the model can be applied to solve the real-life problems. In particular, compared with the node-based formulations used in some of the container drayage problems, the proposed formulation utilises commodity flows to represent containers with the same O-D pair and time constraints, which helps reduce the search space significantly. In order to investigate what has caused the

low loaded distance rate for some instances, the problem is relaxed to a network design problem which provides a lower bound. It was shown that the proposed model and method is able to find solutions that are very close to the lower bounds. In order to improve the transportation efficiency further, the problem needs to be solved at higher level where load imbalance has to be addressed.

It was found that for real-life instances, the solution obtained from the set covering model is very close to the lower bound, suggesting that the time window may not be the driving factor for the low transport efficiency but the demand imbalance between different ports is.

The model can be solved efficiently for most "tight" instances but is found to be computationally expensive for instances with "loose" time windows. In future, it will be interesting to investigate other techniques to address this problem, including multi-threading parallel computing and other novel integer optimisation techniques to speedup the solution process.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (NSFC 71471092, NSFC-RS 71311130142), Ningbo Sci&Tech Bureau (2011B81006,2012B10055) and the Internal Doctoral Innovation Centre programme.

References

R. Bai, J. Blazewicz, E. K. Burke, G. Kendall, and B. McCollum. A simulated annealing hyper-heuristic methodology for flexible decision support. *4OR - A Quarterly Journal of Operations Research*, 10:43–66, 2012a.

R. Bai, G. Kendall, R. Qu, and J. Atkin. Tabu assisted guided local search approaches for freight service network design. *Information Sciences*, 189: 266–281, 2012b.

Gerardo Berbeglia, Jean-François Cordeau, Irina Gribkovskaia, and Gilbert Laporte. Static pickup and delivery problems: a classification scheme and survey. *Top*, 15(1):1–31, 2007.

K. Braekers, A. Caris, and G.K. Janssens. Time-dependent routing of drayage operations in the service area of intermodal terminals. In *International Conference on Harbour, Maritime and Multimodal Logistics Modelling and Simulation*, pages 29–36, 2012.

K. Braekers, A. Caris, and G.K. Janssens. Integrated planning of loaded and empty container movements. *OR Spectrum*, 35(2):457–478, 2013.

Kris Braekers, An Caris, and Gerrit K. Janssens. Exact and meta-heuristic approach for a general heterogeneous dial-a-ride problem with multiple depots. *Transportation Research Part B: Methodological*, 67(0):166 – 186, 2014a. ISSN 0191-2615.

Kris Braekers, An Caris, and Gerrit K. Janssens. Bi-objective optimization of drayage operations in the service area of intermodal terminals. *Transportation Research Part E: Logistics and Transportation Review*, 65(0):50 – 69, 2014b. Special Issue on: Modelling, Optimization and Simulation of the Logistics Systems.

Jianjun Chen, Ruibin Bai, Rong Qu, and G. Kendall. A task based approach for a real-world commodity routing problem. In *Computational Intelligence In Production And Logistics Systems (CIPLS), 2013 IEEE Workshop on*, pages 1–8, April 2013.

Qingfeng Chen, Kunpeng Li, and Zhixue Liu. Model and algorithm for an unpaired pickup and delivery vehicle routing problem with split loads. *Transportation Research Part E: Logistics and Transportation Review*, 69 (0):218 – 235, 2014. ISSN 1366-5545.

U. Derigs, M. Pullmann, U. Vogel, M. Oberscheider, M. Gronalt, and P. Hirsch. Multilevel neighborhood search for solving full truckload routing problems arising in timber transportation. *Electronic Notes in Discrete Mathematics*, 39:281–288, 2012.

Francesco Ferrucci and Stefan Bock. Real-time control of express pickup and delivery processes in a dynamic environment. *Transportation Research Part B: Methodological*, 63(0):1 – 14, 2014. ISSN 0191-2615.

Gabriel Gutierrez-Jarpa, Guy Desaulniers, Gilbert Laporte, and Vladimir Marianov. A branch-and-price algorithm for the Vehicle Routing Problem with Deliveries, Selective Pickups and Time Windows. *European Journal of Operational Research*, 206(2):341–349, OCT 16 2010.

Dominik Kirchler and Roberto Wolfler Calvo. A granular tabu search algorithm for the dial-a-ride problem. *Transportation Research Part B: Methodological*, 56(0):120 – 135, 2013. ISSN 0191-2615.

R. Liu, Z. Jiang, X. Liu, and F. Chen. Task selection and routing problems in collaborative truckload transportation. *Transportation Research Part E: Logistics and Transportation Review*, 46(6):1071–1085, 2010.

Ran Liu, Xiaolan Xie, Vincent Augusto, and Carlos Rodriguez. Heuristic algorithms for a vehicle routing problem with simultaneous delivery and pickup and time windows in home health care. *European Journal of Operational Research*, 230(3):475 – 486, 2013. ISSN 0377-2217.

Hokey Min. The multiple vehicle routing problem with simultaneous delivery and pick-up points. *Transportation Research Part A: General*, 23(5):377–386, 1989.

Jenny Nossack and Erwin Pesch. A truck scheduling problem arising in intermodal container transportation. *European Journal of Operational Research*, 230(3):666 – 680, 2013. ISSN 0377-2217.

D.G. Pandelis, C.C. Karamatsoukis, and E.G. Kyriakidis. Finite and infinite-horizon single vehicle routing problems with a predefined customer sequence and pickup and delivery. *European Journal of Operational Research*, 231(3):577 – 586, 2013.

Julie Paquette, Jean-François Cordeau, Gilbert Laporte, and Marta M.B. Pascoal. Combining multicriteria analysis and tabu search for dial-a-ride problems. *Transportation Research Part B: Methodological*, 52(0):1 – 16, 2013.

David Pisinger and Stefan Ropke. A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34(8):2403–2435, 2007.

Sebastian Sterzik and Herbert Kopfer. A tabu search heuristic for the inland container transportation problem. *Computers & Operations Research*, 40 (4):953 – 962, 2013.

Lixin Tang, Jiao Zhao, and Jiyin Liu. Modeling and solution of the joint quay crane and truck scheduling problem. *European Journal of Operational Research*, 236(3):978–990, 2014.

Paolo Toth and Daniele Vigo. *The vehicle routing problem*. Siam, 2001.

Xiubin Wang and Amelia C Regan. Local truckload pickup and delivery with hard time window constraints. *Transportation Research Part B: Methodological*, 36(2):97–112, 2002.

Nicole Wieberneit. Service network design for freight transportation: a review. *OR Spectrum*, 30:77–112, 2008.

David H Wolpert and William G Macready. No free lunch theorems for optimization. *Evolutionary Computation, IEEE Transactions on*, 1(1):67–82, 1997.

Ruiyou Zhang, Won Young Yun, and Ilkyeong Moon. A reactive tabu search algorithm for the multi-depot container truck transportation problem. *Transportation Research Part E: Logistics and Transportation Review*, 45 (6):904–914, 2009.

Ruiyou Zhang, Won Young Yun, and Herbert Kopfer. Heuristic-based truck scheduling for inland container transportation. *OR spectrum*, 32(3):787–808, 2010.

T. Zhang, W.A. Chaovalitwongse, and Y. Zhang. Integrated ant colony and tabu search approach for time dependent vehicle routing problems with simultaneous pickup and delivery. *Journal of Combinatorial Optimization*, 28(1):288–309, 2014.