

Document Version

Final published version

Citation (APA)

Zhang, Y., Atasoy, B., Souravlias, D., & Negenborn, R. R. (2020). Pickup and delivery problem with transshipment for inland waterway transport. In E. Lalla-Ruiz, M. Mes, & S. Voß (Eds.), *Computational Logistics : Proceedings of the 11th International Conference, ICCL 2020* (pp. 18-35). (Lecture Notes in Computer Science ; Vol. 12433). Springer. https://doi.org/10.1007/978-3-030-59747-4_2

Important note

To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright

In case the licence states "Dutch Copyright Act (Article 25fa)", this publication was made available Green Open Access via the TU Delft Institutional Repository pursuant to Dutch Copyright Act (Article 25fa, the Taverne amendment). This provision does not affect copyright ownership.
Unless copyright is transferred by contract or statute, it remains with the copyright holder.

Sharing and reuse

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy

Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

Green Open Access added to TU Delft Institutional Repository

'You share, we take care!' - Taverne project

<https://www.openaccess.nl/en/you-share-we-take-care>

Otherwise as indicated in the copyright section: the publisher is the copyright holder of this work and the author uses the Dutch legislation to make this work public.



Pickup and Delivery Problem with Transshipment for Inland Waterway Transport

Yimeng Zhang^(✉), Bilge Atasoy, Dimitris Souravlias, and Rudy R. Negenborn

Department of Maritime and Transport Technology, Delft University of Technology,
2628 CD Delft, The Netherlands

{Yimeng.Zhang,B.Atasoy,D.Souravlias,R.R.Negenborn}@tudelft.nl

Abstract. Inland waterway transport is becoming attractive due to its minimum environmental impact in comparison with other transportation modes. Fixed timetables and routes are adopted by most barge operators, avoiding the full utilization of the available resources. Therefore a flexible model is adopted to reduce the transportation cost and environmental impacts. This paper regards the route optimization of barges as a pickup and delivery problem (PDP). A Mixed Integer Programming (MIP) model is proposed to formulate the PDP with transshipment of barges, and an Adaptive Large Neighborhood Search (ALNS) is developed to solve the problem efficiently. The approach is evaluated based on a case study in the Rhine Alpine corridor and it is shown that ALNS is able to find good solutions in reasonable computation times. The results show that the cost is lower when there is more flexibility. Moreover, the cost comparison shows that transshipment terminals can reduce the cost for barge companies.

Keywords: Inland waterway transport · Pickup and delivery problem · Transshipment · Adaptive large neighborhood search

1 Introduction

Transportation of goods and people via different inland waterways such as rivers, lakes and canals, is commonly referred as Inland Waterway Transport (IWT). Compared to road and railway transportation, IWT offers competitive advantages including lower transportation costs, reduced greenhouse gas emissions as well as noise pollution [1]. For this reason, more and more transportation stakeholders want to increase the share of water transportation in their operations. According to the Port of Rotterdam, a total of 100,000 inland barges arrived in 2019 [2]. To handle the subsequent container flows in the future, the Port of Rotterdam aims to raise the utilization of waterborne transport to have the largest modal share over the next 20 years [3]. The EU Transport White Paper (European Commission, 2011) targeted for freight transport to shift from road to rail and IWT by more than 50% by 2050 [4].

© Springer Nature Switzerland AG 2020

E. Lalla-Ruiz et al. (Eds.): ICCL 2020, LNCS 12433, pp. 18–35, 2020.

https://doi.org/10.1007/978-3-030-59747-4_2

Barge operators are the key carriers when it comes to IWT and they usually work with fixed timetables [5]. However, it might be beneficial to adapt the routes during operations, e.g., serve a new transportation request halfway, in order to reduce costs and/or serve additional requests. In this situation, the optimization problem for the barge carrier/operator can be regarded as a pickup and delivery problem (PDP) [6], especially for big transportation companies, that operate over multiple terminals in a river. A representative example of such a company is Contargo [7], which operates barges in the Rhine-Alpine corridor.

In transport operations, it is shown that the shipment of goods to an intermediate destination before they reach their intended destination (i.e., transshipment) can reduce costs and emissions [8, 9]. In case of IWT, one barge drives the request to a predetermined terminal, called transshipment terminal. The second barge picks up the request at the transshipment terminal and transports it to the delivery terminal. In this way, several requests can share a barge as long as capacity is not exceeded, the number of used barges can be minimized and the capacity of barges is utilized better. A real-world example is shown in Sect. 3. To the best of our knowledge, no study focused on the PDP with transshipment for intermodal freight transportation involving IWT. Moreover, most barge operators adopt the fixed timetable and routes for barges, which may lead to loss of flexibility. In this paper, the flexible barges, i.e., barges without predefined timetable and routes, are considered.

In this paper, we study the PDP with transshipment over an intermodal transportation network including inland waterways. To formulate this optimization problem, we propose a Mixed Integer Programming (MIP) model. The objective is to determine the routes of barges that minimize the overall cost, under several practical constraints that realize the transshipment between barges. Given that for real size networks, the studied problem cannot be solved to optimality in reasonable times, we additionally propose an adaptive large neighborhood search algorithm. Based on real-world data including barge timetables and technical specifications (speed profiles, capacities), a series of instances with different sizes and diverse characteristics are generated. An extensive experimental evaluation is conducted that demonstrates the efficiency of the proposed approach under various configurations and across different problem instances.

The remainder of this paper is structured as follows: Sect. 2 presents a brief literature review. The proposed MIP model is described in Sect. 3, while the solution methodology is given in Sect. 4. The experimental settings and results are reported in Sect. 5. The paper concludes and provides directions for future research in Sect. 6.

2 Literature Review

In this section, we briefly review the literature related to the role of transshipment in transportation. Also, we outline several research studies that investigate the use of inland barges in waterway transport operations, including transport routes optimization and service network optimization.

2.1 Transshipment in Transportation Domain

Transshipment is a common practice in various transportation operations and considered in various modeling approaches, including vehicle routing problem with trailers and transshipment (VRPTT), vehicle routing problem with cross-docking (VRPCD), and pickup and delivery problem with transshipment (PDPT).

The VRPTT considers a set of non-autonomous vehicles (trailers), which can only move when pulled by a lorry [10]. In the VRPTT, the trailers can be parked at transshipment locations, where the load is transferred from lorries to trailers. The VRPCD is concerned with defining a set of routes that satisfy transportation requests among different pickup and delivery points. In the VRPCD, the vehicles bring goods from pickup locations to a cross-docking platform, where the items may be consolidated for efficient delivery [11]. The PDPT is a variant of the PDP, where requests can change vehicle at transshipment points during their trip [12]. In the PDP, the pickup and delivery points of a request should be serviced by the same vehicle. By relaxing related constraints, a request can be transported by more than one vehicle. There is a common characteristic in the above problems due to transshipment: synchronization among different vehicles. These problems are Vehicle Routing Problems (VRPs) which exhibit additional synchronization requirements with regard to spatial, temporal, and load aspects [9]. The model proposed in this paper belongs to PDPT.

Some studies proposed heuristics to reduce computation time. To determine the benefits of transshipment in a daily route planning problem at a regional air carrier, a greedy randomized adaptive search procedure (GRASP) was developed to find optimal routes efficiently [14]. A branch-and-price algorithm was proposed for the VRPTT, using problem specific enhancements in the pricing scheme and alternative lower bound computations [15]. In this study, an adaptive large neighborhood search (ALNS) algorithm was additionally used to obtain good initial columns. To solve PDPT efficiently, a heuristic capable of efficiently inserting requests through transfer points was proposed and it was embedded into an ALNS algorithm [12].

Besides freight transportation, some scholars solve similar problems in passenger transportation. For example, a pickup and delivery problem with transfers is formulated and the proposed model is applied to passenger transport [13]. There are three differences between the approach in [13] and this research: a) From the mathematical model perspective, the transfer node in [13] is regarded as two separate nodes (start node and finish node) to model the transshipment. In this research, vehicle flow and request flow are modeled to achieve container transshipment between barges. b) From the solution methodology perspective, a branch-and-cut solution method is used in [13] and there are up to 6 requests, 2 vehicles, and 2 transshipment terminals. In this research, ALNS is used to improve the scalability so that large instances can be handled. c) From the application perspective, the model in [13] and the model proposed in this research are applied to passenger transport and IWT separately, and different transport characteristics are considered. For example, this research takes upstream/downstream speed into account.

2.2 Optimization Challenges for Inland Barges

Barge route optimization is mainly studied by focusing on the waterway transport itself. The optimal values of parameters, which influence the efficient utilization of barges, were investigated in [16]. Several years later, the barge routing problem was studied with the objective of maximizing the profit for a shipping company [17]. In the same study, the upstream and downstream calling sequence as well as the number of loaded and empty containers were also determined. To generate rotation plans for inland barges, an approach was developed on the integration of mixed-integer programming (MIP) and constraint programming (CP) [18]. To address long waiting times and congestion of inland barges in the port, a two-phase approach for planning inter-terminal transport of inland barges was proposed in the presence of several practical constraints [3].

The service network design for barges can be related to intermodal transport. The relations between barge network design, transport market, and the performance of intermodal barge transport is typically the central issue [19]. A general framework that describes design variables of barge networks and identifies their connection to the performance indicators of intermodal barge transport is presented in [20]. There are additional studies on the service network of barges that consider regional characteristics. A hub-and-spoke network was designed for a shipping company that is consistent with the characteristics of the Yangtze River [21]. The subset of ports needs to be called and the amount of containers need to be shipped are determined in [22]. In order to save possible leasing or storage costs of empty containers at the respective ports, the repositioning of empty containers is explored in [22].

Although the VRP with transshipment and waterway transport optimization are well studied, only a few research studies have been devoted to the VRP with barges, and the benefits of transshipment have been rarely exploited. Table 1 provides a detailed comparison among different studies of the relevant literature considering several aspects.

Table 1. Comparison on models in literature.

Literature	Method	Objective	Transshipment	Time & capacity constraints	Upstream/downstream	Waiting time	Heuristic
[19]	QA	–	×	–	×	×	–
[16]	NF	profit	×	✓	✓	×	×
[17]	NF	profit	×	✓	✓	×	✓
[20]	NF	profit	×	✓	×	×	×
[21]	NF	cost	✓	✓	✓	×	×
[18]	NF	time	×	✓	×	✓	✓
[3]	NF	time	×	✓	×	✓	✓
[22]	NF	profit	×	✓	✓	✓	×
Our paper	PDPT	cost	✓	✓	✓	✓	✓

QA: Quantitative Analysis; NF: Network Flow Model.

– means the relevant item is not applicable to the paper.

3 Optimization Model for Inland Barges

We formulate the optimization problem for the barge carrier as a PDPT. Compared with traditional modelling approaches for barges, such as network flow optimization, the proposed model can reduce the cost due to its flexibility and transshipment benefits, as in Fig. 1. In Fig. 1a, a barge named Nova picks up two requests (0 and 1) at Antwerp, then delivers request 1 and picks up request 2 at Rotterdam, finally deliveries requests 0 and 2 at Neuss terminal. If the barge Nova has a fixed shuttle from Antwerp to Neuss, requests 1 and 2 can only be served by other barges, a choice that induces unnecessary costs. In Fig. 1b, request 0 needs to be transported from Frankfurt-Ost terminal to Rotterdam, and request 1 needs to be transported from Mannheim terminal to Rotterdam. As they share the same route segment, request 1 is transferred from barge Michigan to barge So Long at transshipment terminal Koblenz, which avoids extra travel for Michigan and makes full use of So Long’s capacity.

The proposed model allows containers to be transferred from one barge to another at transshipment terminals, as shown in Fig. 2. Therefore, different from traditional PDP, the routes of requests and routes of barges need to be considered separately.

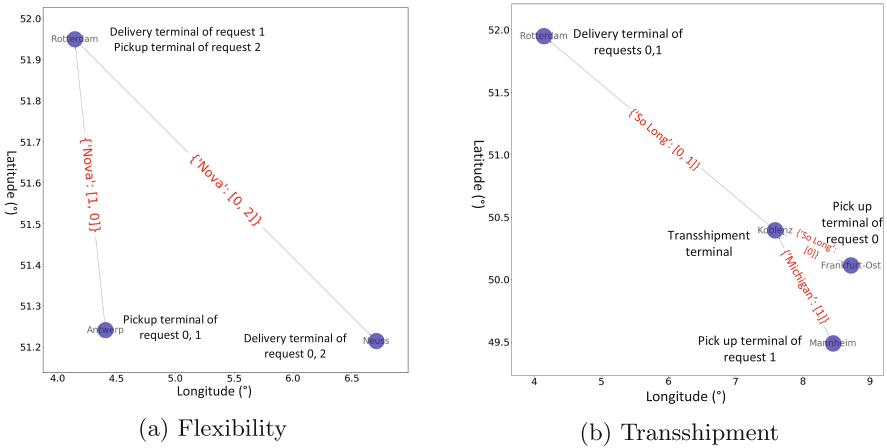


Fig. 1. Illustration of the flexibility and transshipment of the model.

The loading/unloading time is called service time for pickup/delivery, and it is called transshipment time in case of transshipment. Figure 2 also shows how the time is added in the model. The time from the arrival time till the service start time is the waiting time, and departure happens after service time

or transshipment time is completed. There are three situations that lead to waiting time. The first one is when the barge arrives before the pickup/delivery time window, and so needs to wait for containers. The other two situations take place in the transshipment terminal. Assuming barge l will pickup containers and barge k will deliver containers at transshipment terminal. The second situation is that barge l arrives at the transshipment terminal before barge k completes the unloading, and l needs to wait for k . The last situation is that barge k arrives before barge l , and k needs to wait for l . For the first and second situations, the waiting time is necessary. For the last situation, as the reviewer suggested, barge k can unload container directly and doesn't need to wait for barge l . However, this paper doesn't consider the storage in the terminals, and the waiting time is used. Figure 2 shows the first and second situations.

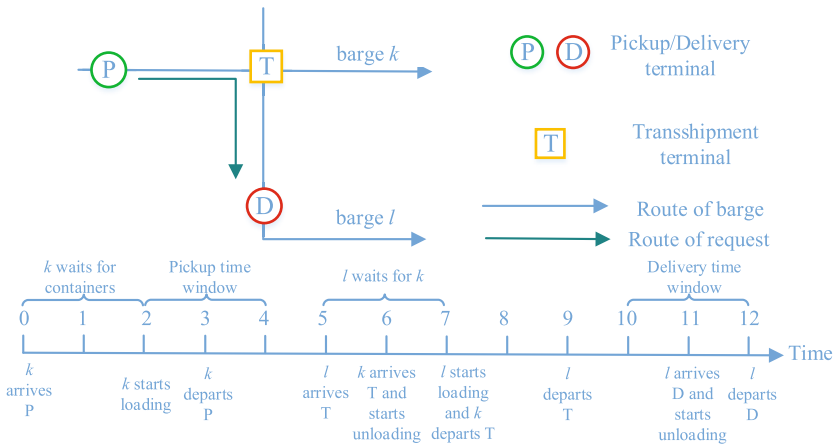


Fig. 2. Transshipment and time in the model.

The notation used in the mathematical model is provided in Table 2. A request consists of pickup and delivery time windows, and number of containers (TEU). The barge set includes barges' name, capacity, and speed. Terminals and arcs form the transport network of the carrier or barge operator. In the terminal set, there are some special terminals, including transshipment terminals, initial/final depots of barges and pickup/delivery terminals of requests. Each barge may start from different depot.

Table 2. Notation used in the model.

Sets	
R	Set of requests
K	Set of barges
N	Set of terminals
A	Set of arcs. For $i, j \in N$, the arc from i to j is denoted by $(i, j) \in A$
$T \subseteq N$	Set of transshipment terminals
$o(k)/o'(k) \subseteq N$	Initial/final depot of barge $k \in K$
$p(r)/d(r) \subseteq N$	Pickup/delivery terminal of request $r \in R$
Variables	
x_{ij}^k	Binary variable; 1 if barge $k \in K$ uses the arc $(i, j) \in A$, 0 otherwise
y_{ij}^{kr}	Binary variable; 1 if request $r \in R$ transported by barge k uses arc $(i, j) \in A$, 0 otherwise
z_{ij}^k	Binary variable; 1 if terminal $i \in N$ precedes (not necessarily immediately) terminal $j \in N$ in the route of the barge $k \in K$, 0 otherwise
s_{ir}^{kl}	Binary variable; 1 if request $r \in R$ is transferred from barge $k \in K$ to barge $l \neq k$ at node $i \in N$, 0 otherwise
t_i^k / \bar{t}_i^k	The arrival/departure time of barge $k \in K$ at terminal $i \in N$
t_i^k	The service start time of barge $k \in K$ at terminal $i \in N$
Parameters	
u_k	Capacity (TEU) of barge $k \in K$
q_r	Quantity (TEU) of request $r \in R$
τ_{ij}^k	The transportation time (in hours) on arc $(i, j) \in A$ for barge $k \in K$
$[a_{p(r)}, b_{p(r)}]$	The pickup time window for request $r \in R$
$[a_{d(r)}, b_{d(r)}]$	The delivery time window for request $r \in R$
t_i^k / \bar{t}_i^k	The transshipment/service time (in hours) for barge $k \in K$ at terminal $i \in N$
v_k	Speed (km/h) of barge $k \in K$. The upstream speed and downstream speed for a same barge are different
d_{ij}	Distance (km) between terminal $i \in N$ and $j \in N$
c_k^{1-5}	c_k^1/c_k^2 are unit (one container) cost (euro) of transportation per km/hour using barge $k \in K$. c_k^3 is the fuel cost per km of barges. c_k^4 is the cost per hour of waiting time, transshipment time and service time at a terminal. c_k^5 is the transshipment cost per container

The objective of the model is to minimize cost, which consists of transportation cost of containers, fuel cost, transshipment cost, and cost associated with waiting, service, and transshipment time, defined as follows:

$$\begin{aligned}
 \text{Minimize } F = & \sum_{k \in K} \sum_{(i,j) \in A} \sum_{r \in R} (c_k^1 d_{ij} + c_k^2 \tau_{ij}) y_{ij}^{kr} q_r \\
 & + \sum_{k \in K} \sum_{(i,j) \in A} (c_k^3 d_{ij} x_{ij}^k + c_k^4 (t_i^k - \bar{t}_i^k)) + \sum_{k,l \in K, k \neq l} \sum_{r \in R} \sum_{i \in T} c_k^5 q_r s_{ir}^{kl}
 \end{aligned} \tag{1}$$

Subject to:

$$\sum_{j \in N} x_{ij}^k \leq 1 \quad \forall k \in K, \forall i = o(k) \quad (2)$$

$$\sum_{j \in N} x_{ij}^k = \sum_{j \in N} x_{jl}^k \quad \forall k \in K, \forall i = o(k), \forall l = o'(k) \quad (3)$$

$$\sum_{k \in K} \sum_{j \in N} y_{ij}^{kr} = 1 \quad \forall r \in R, \forall i = p(r) \quad (4)$$

$$\sum_{k \in K} \sum_{j \in N} y_{ji}^{kr} = 1 \quad \forall r \in R, \forall i = d(r) \quad (5)$$

$$\sum_{j \in N} x_{ij}^k - \sum_{j \in N} x_{ji}^k = 0 \quad \forall k \in K, \forall i \in N \setminus o(k), o'(k) \quad (6)$$

$$\sum_{k \in K} \sum_{j \in N} y_{ij}^{kr} - \sum_{k \in K} \sum_{j \in N} y_{ji}^{kr} = 0 \quad \forall r \in R, \forall i \in T \setminus p(r), d(r) \quad (7)$$

$$\sum_{j \in N} y_{ij}^{kr} - \sum_{j \in N} y_{ji}^{kr} = 0 \quad \forall k \in K, \forall r \in R, \forall i \in N \setminus T, p(r), d(r) \quad (8)$$

$$y_{ij}^{kr} \leq x_{ij}^k \quad \forall (i, j) \in A, \forall k \in K, \forall r \in R \quad (9)$$

$$\sum_{j \in N} y_{ji}^{kr} + \sum_{j \in N} y_{ij}^{lr} \leq s_{ir}^{kl} + 1 \quad \forall r \in R, \forall i \in T, \forall k, l \in K, k \neq l \quad (10)$$

$$s_{ir}^{kl} \leq \sum_{j \in N} y_{ji}^{kr} \quad \forall r \in R, \forall i \in T, \forall k, l \in K, k \neq l \quad (11)$$

$$s_{ir}^{kl} \leq \sum_{j \in N} y_{ij}^{lr} \quad \forall r \in R, \forall i \in T, \forall k, l \in K, k \neq l \quad (12)$$

$$t_i^k \leq t'_i{}^k \quad \forall i \in N, \forall k \in K \quad (13)$$

$$t'_i{}^k \leq \bar{t}_i^k \quad \forall i \in N, \forall k \in K \quad (14)$$

$$\bar{t}_i^k + \tau_{ij}^k - t_j^k \leq M(1 - x_{ij}^k) \quad \forall (i, j) \in A, \forall k \in K \quad (15)$$

$$a_{p(r)} \leq t'_{p(r)}{}^k, \bar{t}_{p(r)}^k \leq b_{p(r)} \quad \forall r \in R, \forall k \in K \quad (16)$$

$$a_{d(r)} \leq t'_{d(r)}{}^k, \bar{t}_{d(r)}^k \leq b_{d(r)} \quad \forall r \in R, \forall k \in K \quad (17)$$

$$t'_{p(r)/d(r)}{}^k + t''_{p(r)/d(r)}{}^k = \bar{t}_{p(r)/d(r)}^k \quad \forall r \in R, \forall k \in K \quad (18)$$

$$\bar{t}_i^k - t'_i{}^k \leq M(1 - s_{ir}^{kl}) \quad \forall r \in R, \forall i \in T, \forall k, l \in K, k \neq l \quad (19)$$

$$\bar{t}_i^k - t'_i{}^k = t''_i{}^k s_{ir}^{kl} \quad \forall r \in R, \forall i \in T, \forall k, l \in K, k \neq l \quad (20)$$

$$\bar{t}_i^l - t'_i{}^l = t''_i{}^l s_{ir}^{kl} \quad \forall r \in R, \forall i \in T, \forall k, l \in K, k \neq l \quad (21)$$

$$x_{ij}^k \leq z_{ij}^k \quad \forall i, j \in N, \forall k \in K \quad (22)$$

$$z_{ij}^k + z_{ji}^k = 1 \quad \forall i, j \in N, \forall k \in K \quad (23)$$

$$z_{ij}^k + z_{jl}^k + z_{li}^k \leq 2 \quad \forall i, j, l \in N, \forall k \in K \quad (24)$$

$$\sum_{r \in R} q_r y_{ij}^{kr} \leq u_k x_{ij}^k \quad \forall (i, j) \in A, \forall k \in K \quad (25)$$

$$x_{ij}^k \in \{0, 1\} \quad \forall (i, j) \in A, \forall k \in K \quad (26)$$

$$y_{ij}^{kr} \in \{0, 1\} \quad \forall (i, j) \in A, \forall k \in K, \forall r \in R \quad (27)$$

Constraints (2) and (3) ensure that a barge begins and ends at its begin and end depot, respectively. Constraints (4) and (5) ensure that containers for each

request must be picked and delivered at its pick up and delivery terminal, respectively. Constraints (6) represent flow conservation for vehicle flow and (7)–(8) represent flow conservation for request flow. Constraints (7) are for transshipment terminals, and constraints (8) are for normal terminals. Constraints (9) link y and x variables in order to guarantee that for a request to be transported by a barge, that barge needs to be traversing the associated arc. Constraints (10)–(12) facilitate transshipment. (10) ensures that the transshipment occurs only once in the transshipment terminal. Furthermore, (11) and (12) let the transshipment only when the request is transported by both barges k and l . Constraints (13)–(15) are the time related constraints. Constraints (13) guarantee that the arrival time of barge is earlier than service start time. Constraints (14) maintain that the departure happens only after the service is completed. Constraints (15) ensure that the time on route is consistent with the distance travelled and speed, and M is a large enough positive number. Constraints (16) and (17) take care of the time windows. These constraints give possibility of waiting at terminals when barge arrives earlier. Constraints (18) add service time of pickup and delivery. Constraints (19)–(21) include time constraints for transshipment. If there is a transshipment from barge k to barge l , but barge l arrives before barge k departs, (19) allows barge l to wait until barge k completes its unloading. Constraints (20) and (21) add transshipment time at the transshipment terminal. Constraints (22)–(24) are the subtour elimination constraints, which provide tight bounds among several polynomial-size versions of subtour elimination constraints [23]. Constraints (25) are the capacity constraints. Constraints (26) and (27) set variables x and y as binary variables.

A MIP model for the PDPT in IWT is thus given by the objective function (1) and constraints (2)–(27) described above.

4 Solution Methodology

For the studied PDPT problem, we consider two main solution methodologies: (i) an exact solution by a solver, Gurobi [24], to the MIP, (ii) an ALNS. Section 5 presents comparative results for both approaches.

The ALNS was proposed in 2006 based on an extension of the large neighborhood search (LNS) heuristic [25], and ALNS adopted an adaptive mechanism to make it robust in different scenarios. ALNS has already been used for VRP problems successfully and it performs well on large-scale instances. There are other different approaches. But this adaptive nature of ALNS, i.e., choosing operators according to their past performances, is a significant advantage over other approaches which do not use it. The pseudocode of the ALNS in the paper is shown in Algorithm 1. X means the solution. For instance, $X_{initial}$ means initial solution, which is generated by Greedy Insertion proposed in Sect. 4.1. R_{pool} is a set of active requests, and it includes requests need to be inserted to routes.

The ALNS is composed of a number of competing sub-heuristics, i.e., insertion and removal operators. An insertion operator is concerned with inserting requests to the routes of barges. In contrast, a removal operator is used for

Algorithm 1: ALNS algorithm

Input: K, R ; **Output:** X_{best} ;
 set $X_{initial}$ as empty routes of K ; $R_{pool} = R$;
 $[X_{initial}, R_{pool}] = GreedyInsertion(X_{initial}, R_{pool})$;
while R_{pool} is not empty **do**
 | $[X_{initial}, R_{pool}] = RandomRemoval(X_{initial})$;
 | $[X_{initial}, R_{pool}] = GreedyInsertion(X_{initial}, R_{pool})$;
end
 $X_{last} \leftarrow X_{initial}$; $X_{best} \leftarrow X_{last}$;
repeat
 | refresh weights and choose operators at the beginning of each segment;
 | $X_{current} \leftarrow X_{last}$;
 | **while** R_{pool} is not empty **do**
 | | $[X_{current}, R_{pool}] = RemovalOperator(X_{current})$;
 | | $[X_{current}, R_{pool}] = InsertionOperator(X_{current}, R_{pool})$;
 | **end**
 | **if** $F(X_{current}) < F(X_{last})$ **then**
 | | $X_{last} \leftarrow X_{current}$;
 | **else**
 | | $X_{last} \leftarrow X_{current}$ with probability p ;
 | **end**
 | **if** $F(X_{last}) < F(X_{best})$ **then**
 | | $X_{best} \leftarrow X_{last}$;
 | **end**
until stop-criterion met;

removing requests from a route. The combination of insertion and removal operators is called *operations*. The process of using an operation until all requests are served is called an iteration. After each iteration, the algorithm will assign to the used insertion operator and removal operator the same score based on the operation's performance. The score criteria are reported in [25].

The entire search is divided into disjoint parts, henceforth called *segments*. A segment assumes a fixed number of iterations, and the number of iterations is denoted as s . The weights of the insertion operator and removal operator are updated after every s iterations, i.e., in every segment, according to their past performance, as follows:

$$w_{i,j+1} = w_{i,j}(1 - \mu) + \mu \frac{\pi_i}{\theta_i}, \quad (28)$$

where $w_{i,j}$ is the weight of operator i used in segment j , π_i denotes the score of operator i obtained during the last segment, and θ_i stands for the number of times operator i is used during the last segment. The reaction factor μ controls how quickly the weight reacts to changes in the performance of the operators.

A roulette wheel selection mechanism is employed to specify which operator will be applied next. This mechanism assumes a probability to select operator j , defined as follows:

$$\frac{w_j}{\sum_{i=1}^n w_i}, \quad (29)$$

where w_j is the weight of operator j , and n is the number of operators.

By using an operation, a feasible solution is detected at the end of an iteration. If the current solution is worse than the last solution, it will be accepted with a probability p in order to avoid local optima easier. Simulated annealing idea is used and probability p gradually declines in order to avoid local optima [25], as the following equation shows:

$$p = e^{-\frac{(F(X_{current}) - F(X_{last}))}{T}} \quad (30)$$

where $T > 0$ is the temperature which starts from an initial temperature and gradually decreases in every iteration using the expression $T = T \cdot c$. c is the cooling rate and $0 < c < 1$.

After a number of iterations, the search tends to converge and finally a (sub)optimal solution is found.

4.1 Insertion Operators

Three insertion operators are used, namely greedy insertion, random insertion, and transshipment insertion.

Greedy Insertion: This insertion finds the best position in all routes for a request. The algorithm of greedy insertion without transshipment has been widely discussed in literature [14]. In algorithm with transshipment, the requests are segmented by transshipment terminals firstly, then every request is divided into two sub-requests for one transshipment terminal. For each sub-request, the algorithm without transshipment is used to find its best position. If both sub-requests satisfy constraints, they will be added to the candidate list and then the best will be chosen. In greedy insertion, the operator without transshipment is tried first. If it doesn't work, the transshipment will be considered.

The advantage of this operator is that it can find the best position for one request. However, it needs a longer time than other insertion operators as it may get stuck in a local optimum when the best position for one request is not the best position for the overall objective.

Random Insertion: To make up for the disadvantage of greedy insertion, the random insertion is designed such that the insertion position is chosen randomly, rather than trying all positions for both with and without transshipment case. Random insertion can expand the search space and avoid local optimum, and therefore needs less time.

Transshipment Insertion: The greedy insertion evaluates the insertion of each request sequentially and may miss good opportunities to use transshipment terminals. Some requests may benefit from the transshipment, as illustrated in [12],

hence applying insertion with transshipment may detect the optimal solution quicker. Different from greedy insertion, this operator only uses the algorithm with transshipment.

4.2 Removal Operators

Five removal operators are considered to remove requests from routes and transfer them to the request pool: worst removal, random removal, delete node, clear route, and remove all.

Worst Removal: This operator removes the requests with the highest cost in each route. The cost of the request is calculated as follows:

$$cost_r = F(X) - F(X_{-r}), \quad (31)$$

where X_{-r} is the solution without request r . A request, that is served by more than one barge, will be removed completely from all routes.

Random Removal: Similar to the idea of random insertion, the random removal operator removes requests randomly, offering more unexplored spaces for insertion operators.

Delete Node: In most times, a barge carries multiple requests, therefore removing part of the requests may not change the routes of barges. However, most of the cost-savings are obtained from minimizing distance, i.e., changing routes of barges. To get better solutions quicker, Delete Node operator is designed, which deletes visited terminals in the routes. If one pickup terminal is deleted, the delivery terminal and relevant requests will be deleted too.

Clear Route: Insertion operators may not be able to find feasible solutions based on a small number of removals in a short time. In this case, the route needs to be cleared, which means all requests in a route are removed to the request pool. Another idea behind this operator is to guide the search to the direction of minimizing the number of used barges and making full use of capacity.

Remove All: This operator deletes all requests in routes and fills the request pool. This operator may change the search direction from the beginning and thus provide a larger neighborhood for insertion operators.

The synchronization for relevant barges is considered. Due to the transshipment, other barges may be influenced when one request is inserted to/removed from the route of a barge. These affected barges can cooperate with changes by extending or shortening the waiting time.

4.3 Performance Improvement

The application of an insertion operator typically involves the evaluation of the same move repeatedly, thereby resulting in a high number of duplicate computations during the optimization. Avoiding these repetitive computations can

significantly reduce computation time, especially for large instances. Inspired by the idea proposed in [14], a cache structure that uses a hash table is implemented. Specifically, the hash table holds the best insertion positions and infeasible insertion positions for a given request and route. In total, eight hash tables are established. From these, four hash tables are devoted to best insertion while the rest four hash tables are built for failed insertion. There are two hash tables for insertion without transshipment, which include all possible positions and the best position during the search separately. The other two hash tables are for insertion with transshipment. Similarly, the other four hash tables for failed insertion include the same keys but they don't have values because the solution is infeasible.

Moreover, the complexity of transshipment is reduced by limiting the solution space. According to specific requests and barges, the transshipment terminals that are far away from them are not considered.

5 Case Study

In order to evaluate the proposed methodology, we carry out a set of experiments. All experiments are implemented in Python and run on a laptop with 8 GB of memory and an Intel Core i7 CPU with two 1.90 GHz and 2.11 GHz cores. First, we validate the ALNS algorithm with respect to the exact solution of the MILP for relatively small instances as a benchmark comparison. Then we perform numerical experiments on larger instances to show the performance of the ALNS algorithm for realistic size problems. For the experiments, we chose a case in the Rhine-Alpine water corridor that covers a wide area in Europe from Rotterdam/Antwerp to Basel. Improving transport across this corridor can benefit the transport operators in the Rhine river as well as transport stakeholders in the Trans-European Transport Network (TEN-T) [26].

5.1 Data

The Contargo company is one of the largest intermodal transportation providers in the Rhine-Alpine corridor, which is used as our case. In Rhine river, Contargo transports containers among 21 terminals, including deep-sea terminals in Rotterdam and Antwerp ports and inland terminals along the Rhine. According to figures [27] and timetables [28] in Contargo's website, it has nine services, which all have both import and export. The cost data used in the paper are reported in [29]. The speed data is obtained from an online ship monitoring system [30]. Affected by water flow, the upstream speed is lower than downstream speed.

5.2 Benchmark Comparison

Table 3 compares the results of exact approach by Gurobi and the best solution by ALNS in terms of average cost and computation time for small instances (1–8 requests, 2 barges and 1 transshipment terminal). For each instance, 20

independent experiments are conducted to get the average value. When Gurobi finds the optimal solution in the limited time (12 h), ALNS is able to provide the same solution in significantly less computational time. For the instance with 7 requests, Gurobi cannot reach optimality within the time limit and ALNS outperforms within a significantly lower computational time. As of 8 requests, Gurobi cannot provide a feasible solution within the time limit.

Table 3. The comparison between benchmark and ALNS.

Number of requests	Cost (euro)		Avg. running time (s)		Cost gap (%)
	Gurobi	ALNS	Gurobi	ALNS	
1	18048.3	18048.3	2.0	0.05	0
2	30033.1	30033.1	37.3	0.1	0
3	43660.7	43660.7	192.0	0.7	0
4	58431.2	58431.2	269.0	1.0	0
5	58518.4	58518.4	2482.6	2.6	0
6	58757.5	58757.5	10780.3	103.5	0
7	64160.5	59089.4	43200.0 ^a	141.7	-7.9
8	-	59250.4	43200.0 ^a	334.7	-

^aTime limit reached (12 h).

-Gurobi can't find a feasible solution within the limited time.

5.3 Experiments with Large Instances

The possibility of transshipment in our model is one of the key complexities towards computational burden. Nevertheless, transshipment plays a vital role in supply chain today, allowing cargo to reach different parts of the world. It offers logistics players a high level of flexibility that can bring significant cost benefits. To obtain the saving cost of transshipment, the following experiments with large instances will compare the results of ALNS with/without transshipment.

Before proceeding with the results, we discuss the parameters of ALNS that affect the performance. Among those we analyzed the size of segments, s , and reaction factor μ based on an instance with 10 requests, 4 barges, and 1 transshipment terminal. These requests use Rhine-Main and Rhine-Upper services of Contargo company. To make sure the tuned parameters perform well under PDPT, these requests are designed to benefit from transshipment. There are 200 iterations in each experiment, and all experiments are repeated 5 times to get the average result. We concluded that the size of the segment needs to be sufficiently large compared to the number of operators in order to be able to update the weights accordingly in the early iterations. Otherwise it might lead to misleading weights. For example, for the case of 10 requests we need at least 8 iterations in a segment. When it comes to the reaction factor, our experiments showed that one should not ignore the history completely, i.e., μ should be smaller than 1. We chose 0.8 for the reactive factor μ . Note that, these parameters need to be carefully analyzed for ensuring the performance of ALNS for different problems.

Based on the above-mentioned tuning of parameters, we run larger instances as the results are shown in Table 4. Three large instances are designed with a naming convention such that “20r4k1T” means 20 requests, 4 barges and 1 transshipment terminal. The first instance contains 20 requests using same services with the small instance used in parameter tuning. Other two large instances include 30 requests, which cover all services of Contargo company. The only difference between these two instances is the loads of requests that are randomly drawn. The cost of the instances with 30 requests is lower than the instance with 20 requests because loads of requests and capacity of barges are different. Generally, the running time with large instances is less than 10 min.

Table 4. Results for large instances.

Instances	# Flexible barges	Transshipment		No transshipment		Cost gap (%)
		Cost	Running time	Cost	Running time	
20r4k1T	0	59438.0	220.6	59438.0	313.3	0
	1	59438.0	246.0	59438.0	175.4	0
	2	55423.8	17.7	59441.5	148.4	7.3
	3	54964.8	233.7	59439.3	229.7	8.1
	4	54986.3	397.2	58954.2	120.1	7.2
30r5k2T-I	0	31620.8	1.4	31620.8	1.6	0
	1	29070.7	219.0	29070.7	105.4	0
	2	26460.6	946.7	26584.1	489.1	0.4
	3	26575.0	233.3	28363.4	309.0	6.7
	4	21355.8	257.6	21355.8	100.5	0
	5	19861.2	658.7	19861.2	314.3	0
30r5k2T-II	0	25502.2	578.0	29284.8	2.2	14.9
	1	23714.4	332.6	23714.4	2.2	4.1
	2	23714.4	243.3	23714.4	231.3	0
	3	23277.5	250.0	23277.5	233.4	0
	4	18424.9	88.4	18424.9	16.0	0
	5	16930.2	7.1	16930.2	7.0	0

In Table 4, “# Flexible Barges” column represents the number of barges without fixed timetables. Except for flexible barges, other barges use real fixed timetable. When the number of flexible barges increases, the cost is reduced due to the flexibility of the proposed PDPT model.

The results show that the savings from transshipment vary a lot from one instance to another. The three factors that are playing an important role in an interdependent way are: (1) the cost of transshipment (based on the load) (2) the number of flexible barges (3) the capacity of the barges with respect to the loads. When the load is higher the transshipment costs are higher and it may not be preferable to use the transshipment option. When the number of flexible barges increases, more benefits from transshipment can be exploited, as shown in results of instances 20r4k1T (number of flexible barges from 0 to 2) and 30r5k1T-I (number of flexible barges from 0 to 3). Meanwhile, when there

are enough flexible barges, the attractiveness of transshipment may decrease as the same phenomenon can be represented by the flexibility in routes, as shown in results of instances 30r5k1T-I (number of flexible barges from 3 to 5) and 30r5k1T-II (number of flexible barges from 0 to 5).

From the results, we can conclude that the following cases could benefit from transshipment: a) Requests that on the overlap (or similar) route of barge A and B can be transferred from one barge to another, and the cost-savings result from avoiding an unnecessary barge trip. b) Barge A's transport cost is lower than barge B, then part of requests of barge B can be transferred to barge A to make full use of its capacity.

6 Conclusions and Future Work

Inland waterway transport has been widely recognized as reliable, cost-efficient, and sustainable. To make full use of resources in waterway transport, a MIP model for Pick and Delivery Problem with Transshipment of inland barges is proposed. Because the solution of the studied problem is computationally expensive, an ALNS heuristic is proposed by developing operators specific to this problem. The results show that the ALNS approach reduces computation time significantly, and the best solution of large instances (up to 30 requests) can be obtained within 10 min. Due to the flexibility of PDPT, the cost decreases gradually when the number of barges without fixed timetable increases in the transport network. Additionally, the introduction of transshipment terminals brings reduction on costs (up to 14%), but it differs greatly from one instance to another. Future research will focus on the following aspects:

1. Multiple barge operators may want to cooperate to optimize their transport networks. Moreover, barge operators need to communicate with terminal operators about the arrival time and berth allocation. Therefore collaborative planning among different players will be studied.
2. New requests from a spot market may be released and new barges may be added into the transport network, which will lead to dynamic optimization. If the new request (new barge) is more suitable for a barge (a request), the original plan can be replaced with a penalty.
3. Boundaries of the work can be extended with the expansion of the network to include other modalities and the optimization for barges within the inter-modal transport will be studied.
4. The storage in the terminal will be considered in the future, and the storage cost and terminal capacity will be added in the model.

Acknowledgments. This research is financially supported by the China Scholarship Council under Grant 201906950085 and the project “Complexity Methods for Predictive Synchronicity” (project 439.16.120) of the Netherlands Organisation for Scientific Research (NWO). In the meantime, we would like to express our sincere thanks to the reviewers for the constructive and positive comments.

References

1. Caris, A., Limbourg, S., Macharis, C., van Lier, T., Cools, M.: Integration of inland waterway transport in the intermodal supply chain: a taxonomy of research challenges. *J. Transp. Geogr.* **41**, 126–136 (2014)
2. Port of Rotterdam: Facts and figures about the port. <https://www.portofrotterdam.com/en/our-port/facts-figures-about-the-port>. Accessed 2 Apr 2020
3. Li, S., Negenborn, R.R., Lodewijks, G.: Planning inland vessel operations in large seaports using a two-phase approach. *Comput. Ind. Eng.* **106**, 41–57 (2017)
4. Ambra, T., Caris, A., Macharis, C.: Towards freight transport system unification: reviewing and combining the advancements in the physical internet and synchro-modal transport research. *Int. J. Prod. Res.* **57**(6), 1606–1623 (2019)
5. Li, L., Negenborn, R.R., De Schutter, B.: Distributed model predictive control for cooperative synchro-modal freight transport. *Transp. Res. Part E Logist. Transp. Rev.* **105**, 240–260 (2017)
6. Savelsbergh, M.W., Sol, M.: The general pickup and delivery problem. *Transp. Sci.* **29**(1), 17–29 (1995)
7. Contargo. <https://www.contargo.net/>. Accessed 2 Apr 2020
8. Rais, A., Alvelos, F., Carvalho, M.S.: New mixed integer-programming model for the pickup-and-delivery problem with transshipment. *Eur. J. Oper. Res.* **235**(3), 530–539 (2014)
9. Drexl, M.: Applications of the vehicle routing problem with trailers and transshipments. *Eur. J. Oper. Res.* **227**(2), 275–283 (2013)
10. Drexl, M.: Branch-and-cut algorithms for the vehicle routing problem with trailers and transshipments. *Networks* **63**(1), 119–133 (2014)
11. Grangier, P., Gendreau, M., Lehuédé, F., Rousseau, L.: A matheuristic based on large neighborhood search for the vehicle routing problem with cross-docking. *Comput. Oper. Res.* **84**, 116–126 (2017)
12. Masson, R., Lehuédé, F., Péton, O.: An adaptive large neighborhood search for the pickup and delivery problem with transfers. *Transp. Sci.* **83**(3), 344–355 (2013)
13. Cortés, C.E., Matamala, M., Contardo, C.: The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method. *Eur. J. Oper. Res.* **200**(3), 711–724 (2010)
14. Qu, Y., Bard, J.F.: A GRASP with adaptive large neighborhood search for pickup and delivery problems with transshipment. *Comput. Oper. Res.* **39**(10), 2439–2456 (2012)
15. Parragh, S.N., Cordeau, J.: Branch-and-price and adaptive large neighborhood search for the truck and trailer routing problem with time windows. *Comput. Oper. Res.* **83**, 28–44 (2017)
16. Maraš, V.: Determining optimal transport routes of inland waterway container ships. *Transp. Res. Rec.* **2062**(1), 50–58 (2008)
17. Maraš, V., Lazić, J., Davidović, T., Mladenović, N.: Routing of barge container ships by mixed-integer programming heuristics. *Appl. Soft Comput.* **13**(8), 3515–3528 (2013)
18. Li, S., Negenborn, R.R., Lodewijks, G.: Approach integrating mixed-integer programming and constraint programming for planning rotations of inland vessels in a large seaport. *Transp. Res. Rec.* **2549**(1), 1–8 (2016)
19. Konings, R.: Network design for intermodal barge transport. *Transp. Res. Rec.* **1820**(1), 17–25 (2003)

20. Braekers, K., Caris, A., Janssens, G.K.: Optimal shipping routes and vessel size for intermodal barge transport with empty container repositioning. *Comput. Ind.* **64**(2), 155–164 (2013)
21. Zheng, J., Yang, D.: Hub-and-spoke network design for container shipping along the Yangtze River. *J. Transp. Geogr.* **55**, 51–57 (2016)
22. Alfandari, L., Davidović, T., Furini, F., Ljubić, I., Maroš, V., Martin, S.: Tighter MIP models for barge container ship routing. *Omega* **82**, 38–54 (2019)
23. Öncan, T., Altinel, İ.K., Laporte, G.: A comparative analysis of several asymmetric traveling salesman problem formulations. *Comput. Oper. Res.* **36**(3), 637–654 (2009)
24. Gurobi. <https://www.gurobi.com/>. Accessed 2 Apr 2020
25. Stefan, R., David, P.: An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transp. Sci.* **40**(4), 455–472 (2006)
26. Trans-European Transport Network (TEN-T). <https://ec.europa.eu/transport/infr-aststructure/tentec/tentec-portal/site/index.en.htm>. Accessed 2 Apr 2020
27. Contargo: Facts and figures. <https://www.contargo.net/en/transport/inlandbarge/facts>. Accessed 2 Apr 2020
28. Contargo: Timetables. <https://www.contargo.net/en/transport/schedules/>. Accessed 2 Apr 2020
29. Van Riessen, B., Negenborn, R.R., Lodewijks, G., Dekker, R.: Impact and relevance of transit disturbances on planning in intermodal container networks using disturbance cost analysis. *Marit. Econ. Logist.* **17**(4), 440–463 (2015). <https://doi.org/10.1057/mel.2014.27>
30. Marine Traffic. <https://www.marinetraffic.com/>. Accessed 2 Apr 2020