https://doi.org/10.1016/j.tre.2020.101945

Single-path Service Network Design Problem with Resource Constraints

Xiangyong Li
School of Economics & Management, Tongji University, Shanghai 200092, China, xyli@tongji.edu.cn

Yi Ding
Bank of Shanghai, Shanghai 200120, China, 740854308@qq.com

Kai Pan
Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Hong Kong, kai.pan@polyu.edu.hk

Dapei Jiang
School of Economics & Management, Tongji University, Shanghai 200092, China, 56jdp@tongji.edu.cn

Y.P. Aneja
Odette School of Business, University of Windsor, Windsor, Ontario N9B 3P4, Canada, aneja@uwindsor.ca

We study a new service network design problem, where the number of available resources at each terminal is limited, and each commodity is delivered along a single path that prevents flow partition. Such a single-path constraint is motivated by currently emerging requirements in bulk transportation, express delivery, hazardous material transportation, etc. We model this problem with two mathematical formulations, i.e., node-arc and arc-cycle formulations, both of which lead to large-scale and computationally difficult mixed-integer programs. The node-arc formulation faces a significant computation burden. To that end, we develop a two-stage mathematical integer programming based heuristic for the arc-cycle formulation to produce high-quality solutions. In the first stage, a column generation procedure is executed to generate an optimal solution for the linear relaxation of the restricted master problem, and in the second stage, four heuristic strategies are designed to efficiently generate integer feasible solutions for the original problem. We conduct extensive experiments to verify the effectiveness of our proposed approach by comparing it with a commercial solver (CPLEX). We also examine the performance differences among four heuristic strategies, in terms of the frequency of finding integer feasible solutions and the quality of solutions.

Key words: Transportation; service network design; single-path constraints; resource constraints; column generation heuristic

1. Introduction

Nowadays increasing demands of freight transportation lead to increasingly complex requirements for service network design. Carriers will be challenged in providing customers with highly efficient and reliable services. To overcome this challenge, carriers have to manage limited resources (e.g., vehicles, carrying units, handling equipment, and power units) that they own in an efficient and

cost-effective manner. Thus, recent studies have paid much attention to service network design with resource management (SNDAM), to better manage resources at the minimum total cost (Andersen et al. 2009, 2011). In particular, the resource management is achieved through the so-called design-balance constraints in the service network design models (Pedersen et al. 2009), by assuming that each service will be operated by a resource, and there should be an equal number of resources entering and leaving each node in the network. As a significant extension of the SNDAM, Crainic et al. (2016) further introduced resource-availability constraints: resources are required to return to their home terminals and there are limited resources assigned to each terminal in the network. This extension enlarges the range of resource management in the service network design. Most of the studies considering limited available resources assume that the network flow of a single commodity can use over one path from origin to destination. However, nowadays increasingly more real-world planning problems ask for single-path delivery that prevents commodity flow partition. For example, the commodity itself can be a single heavy-weight cargo, which eliminates the possibility of commodity partition and may require bulk transportation. Due to increasing e-commerce activities and express shipments, single-path requirements become more and more important in city distribution or local distribution networks in a large urban area. In express package delivery, which can be requested by customers online, each set of packages with common origin and destination will be considered as a commodity and transported along a single network path to facilitate operations and ensure customer satisfaction (Barnhart et al. 2000). Avoiding flow partition can obviously reduce the operation cost related to package loading, unloading and sorting, which accounts for a large part of the total transportation cost. For hazardous material transportation, where the risk of suffering an accident is high, it is suggested to route hazardous materials as a package along a single origin-destination path (Verter and Kara 2008). Other application of non-split flows can be found in the food distribution (Bortolini et al. 2016), and the offshore wind maintenance service logistics (Gundegjerde et al. 2015, Schrotenboer et al. 2020).

Considering resource-availability constraints and single-path requirements, in this paper, we focus on a new service network design problem, which complements the existing studies and tackles currently emerging requirements as mentioned above. In particular, we consider a scheduled service network design problem in the context of time-space networks for consolidation-type carriers, where services represent transportation operations defined in terms of origin and destination terminals, route, speed, and capacity (Andersen et al. 2011). In the corresponding time-space networks, resources for performing transportation services follow a cyclic schedule, i.e., the schedule is operated repetitively over a certain number of periods (e.g., a week, or a season). The goal is to generate a tactical plan by optimally 1) selecting services and their departure times, and 2) routing the commodities from their origins to destinations, with the aim of minimizing the total

cost. In addition, such constraints as satisfying given demands, resource availability limitations, and single-path should be respected. We refer to the introduced problem as the single-path service network design problem with resource constraints (SPSNDRC)

The inclusion of single-path requirements and resource-availability constraints poses significant methodological challenges. To the best of the authors' knowledge, it is not significantly addressed in the extant literature. Therefore, we aim to provide a complete model and efficient solution approaches for the SPSNDRC. We first present a node-arc integer programming formulation for the SPSNDRC. The arc-node formulation allows us to solve instances of relatively small size to optimality by using commercial mixed integer programming (MIP) solvers such as CPLEX or Gurobi. To solve large instances, we propose an MIP-based method in this paper. In particular, we introduce an arc-cycle formulation, which involves an exponential number of variables but enables implementable and efficient solution approaches. We develop a two-stage MIP-based heuristic. In the first stage, the column generation procedure is executed to enumerate a subset of design-cycles for given resources at each terminal, and thus optimal solution is generated for the linear relaxation of the restricted master problem, which is a relaxation of the SPSNDRC formulation. In the second stage, integer feasible solutions are generated based on the solution obtained in the first stage through four heuristic strategies. We use heuristic strategies to effectively reduce the model size, but keep the possibility of finding high-quality solutions. Finally, we evaluate our approach over forty instances, which are generated based on forty benchmark SNDAM instances in the literature, by comparing it with CPLEX. Also, for further comparisons, CPLEX is implemented to solve the node-arc model as a benchmark. The computational results demonstrate the efficiency and effectiveness of our proposed two-stage approach.

In summary, we describe the main contributions of our paper as follows.

- We propose a new problem called SPSNDRC, where a limited number of resources are available at physical terminals, and the single-path requirements for routing commodities are considered. This new setting appears in practical planning problems, enlarges the range of resource management in the service network design, and raises significant methodological challenges.
- We model the SPSNDRC problem with two mathematical formulations, i.e., node-arc and arc-cycle formulations. The MIP solver CPLEX is used to solve forty problem instances in order to evaluate the strength and weakness of the node-arc formulation, thereby analyzing the computational complexity of the SPSNDRC.
- We develop a two-stage MIP-based heuristic to produce high-quality solutions to the SPSNDRC. In particular, four heuristic strategies are introduced to efficiently search for integer feasible solutions. Extensive computational experiments are performed to evaluate the effectiveness of our proposed approach.

The remainder of this paper is organized as follows. We give the literature review in Section 2. Following the problem statement provided in Section 3, we present two mathematical formulations in Section 4. We then present the solution approach in Section 5. In Section 6, we present computational results to evaluate the performance of our proposed approach. Finally, we summarize our results and conclude this paper in Section 7.

2. Literature Review

Most studies related to the SPSNDRC can be found in the domain of network design and service network design (SND) (Magnanti and Wong 1984, Ahuja et al. 1993, Crainic 2000, Hewitt et al. 2010). For excellent survey papers on general issues of network design and service network design, please refer to Crainic (2000) and Wieberneit (2008). We provide a brief review related to SND models and resource management issues.

SND represents a large family of mathematical optimization problems and is widely used to model the decision-making issues appeared in telecommunication, transportation, logistics, manufacturing industries, and other fields; see Magnanti and Wong (1984), Crainic (2000), Pedersen et al. (2009), Li et al. (2012), Lo et al. (2013), Zhu et al. (2014), Chouman and Crainic (2015), Bortolini et al. (2016), Li et al. (2017b), and the references therein. Crainic (2000) gave an excellent survey of SND. The authors discussed tactical planning within the larger context of transportation systems and planning issues, and introduced fundamental formulations for the service network design. In the context of SND, resource management issues are explicitly studied by Pedersen et al. (2009), Vu et al. (2013), Chouman and Crainic (2015), Crainic et al. (2016), Andersen et al. (2009, 2011), Li et al. (2017b), and Crainic et al. (2018). Depending on specific applications, resources can refer to carrying units (e.g., railcars, trailers, and containers), handling equipment units, and power units (e.g., tractors) for performing transportation services, etc. By utilizing resources continuously following cyclic routes, the SNDAM model is able to improve operational efficiency.

Regarding the SNDAM, Andersen et al. (2009) studied the computational performance of arc- and cycle-based formulations, by solving limited-size instances with a commercial MIP solver. Their computational study was based on a priori enumeration of cycles for resources and paths for commodities. The computational results showed that cycle-design and path-flow variables contribute to efficient model solving. Andersen et al. (2011) further presented an effective branch-and-price approach for the SNDAM problem. This approach decomposes the SNDAM problem into (1) a master problem, handling a variant of the multicommodity network design problem with vehicle management constraints, and (2) two types of subproblems for generating cycles of resources and paths of commodities, respectively. Pedersen et al. (2009) developed a tabu search metaheuristic for the design-balanced capacitated multicommodity network design problem (DBCMND). In

this problem, the design-balanced constraints require that the number of resources entering and leaving a terminal should be equal. The neighborhood operators focus on the design vector of a feasible solution. In particular, the neighboring solutions are generated through adding or deleting arcs from the design vector. Vu et al. (2013) further developed a three-stage metaheuristic for the DBCMND, which combines exact approach and neighborhood-based heuristic. Chouman and Crainic (2015) introduced a cutting-plane matheuristic for the DBCMND. In this method, the cut-plane procedure is implemented to compute tight lower bounds, and a variable-fixing approach is used to reduce the problem size. To efficiently solve the service network design with resource constraints (SNDRC), Crainic et al. (2016) proposed a matheuristic approach in which column generation, slope scaling, and mathematical programming techniques are combined. Barnhart and Schneur (1996), Kim et al. (1999), and Lai and Lo (2004) also discussed the design-balanced constraints. Among these works, the authors all assumed the resources are homogeneous, the number of resources is unlimited, and each arc can be used once by resources. To be more realistic, Li et al. (2017b) relaxed homogeneous-asset assumption and explicitly studied the service network design with heterogeneous assets.

Among them, Pedersen et al. (2009), Vu et al. (2013), Chouman and Crainic (2015), and Crainic et al. (2016) assume that the transportation system has a single vehicle type, resources are capacitated, and each network link can be used only once by resources. The authors all assumed that resources are homogeneous. Li et al. (2017b) relaxed homogeneous-resource assumption and explicitly study the service network design with heterogeneous resources. In all the above works, except for Crainic et al. (2016), researchers implicitly dealt with where the resources should be in the network. That is, where the resources start to perform the services is the optimization output of the SND models. Crainic et al. (2016) enlarged the range of resource management by considering that resources are required to return to their home terminals and that there are limited resources assigned to each terminal in the network. Crainic et al. (2018) extended the work presented in Crainic et al. (2016) to handle multiple types of resources (heterogeneous resources). In particular, Crainic et al. (2018) studied a service network design model that encompasses both strategic and tactical planning decisions for a consolidation-based carrier. Crainic et al. (2018) further extended the matheuristic in Crainic et al. (2016) to efficiently solve this new model. The aim of this paper is to examine single-path requirements in the context of the service network design with resource constraints. Zhu et al. (2014) studied the scheduled service network design problem for freight rail transportation. Different from previous researches, Zhu et al. (2014) integrated into a single scheduled service network design formulation all the major tactical planning issues including service selection and scheduling, car classification and blocking, train makeup, and freight routing of time-dependent customer shipments. To solve the resulting complicated model, the authors proposed a

matheuristic approach integrating slope scaling, long-term memory-based perturbation strategies, and ellipsoidal search. Other researches for integer commodity flow variables include Gundegjerde et al. (2015), Bortolini et al. (2016), and Schrotenboer et al. (2020). Bortolini et al. (2016) studied the multi-modal fresh food distribution network design problem. The authors presented a tri-objective model for jointly minimizing the operating cost, the carbon footprint, and the delivery time for the fresh food distribution. This model is applied to an industrial case study where the fruits and vegetables are distributed from a set of Italian producers to multiple European retailers through a multi-modal transportation network. Gundegjerde et al. (2015) and Schrotenboer et al. (2020) examined tactical and strategic decision making in the offshore wind maintenance service logistics. Gundegjerde et al. (2015) studied the vessel fleet size and mix problem that arises for the maintenance operations at offshore wind farms, and proposed a stochastic three-stage programming model. Schrotenboer et al. (2020) studied the stochastic maintenance fleet transportation problem for offshore wind farms. In this problem setting, a maintenance provider determines an optimal, medium-term planning for maintaining multiple wind farms while controlling for uncertainty in the maintenance tasks and weather conditions.

In addition to issues of design-balance, and mandatory return to the home-terminal, the researchers also examined other aspects of the resource management. Chen and Schonfeld (2016) introduced a dispatching model, devoted to managing intermodal logistics operations while countering delay and delay propagation. The authors explicitly examined coordination decisions in freight transfer scheduling, where unsplittable flows are used to model the practical operation. Laaziz and Sbihi (2019) presented an SND model in the case of intermodal rail-road container freight transportation from the freight forwarder perspective. The authors considered the design-balance constraints for train flow conservation at each terminal and gateway, and constraints for train full loading. Further, Laaziz and Sbihi (2019) introduced penalties cost related to under loaded trains in order to take into account the trade-off between cost effectively satisfying transportation demand and cost effectively using trains and rail.

3. Problem Statement

Following the literature, we assume that in the SPSNDRC, the schedule length is given and that the services are operated in a regular and repetitive pattern, which represent the fixed schedules of real-world transportation services. The schedule length T may be one day, one week, or one month, and can be divided into t_{max} periods, i.e., $T = \{1, 2, \ldots, t_{max}\}$. The decisions include selecting services and their departure times, and routing commodity demands through the selected service network, where origin and destination are given for each demand. A route that a resource selects is a cycle composed of a series of services, waiting at terminals, and, eventually, repositioning movements.

As we discussed in Section 1, flow of a commodity must be transported along a single path in the SPSNDRC. In addition, we allow the transfer of a shipment from one resource to another. That is, while a shipment may travel on a sequence of services, each of those services can be contained in different resource routes. In addition, a single type of resource is required for each service. For ease of exposition, we use a vehicle to represent a resource for better presenting problem statement and mathematical programming models in the following part of this paper.

Let $\tilde{G} = (L, E)$ be the physical network with a terminal set L and an arc set E. To model the transportation operations of a carrier, we define a time-space network $G = (N, A)$ with a node set N and an arc set A. Different from the physical network, the nodes in the time-space network have time-dependent characteristics. We use node $l_t \in N$ to represent terminal l at time period t for any $l \in L$ and $t \in T$, i.e., $N = L \times T = \{l_t : l \in L, t \in T\}$. We divide the arcs in A into two disjoint sets. *The first one*, denoted by S, contains all the service arcs that link nodes belonging to different physical locations. In particular, each service arc models the operation of a service between two different terminals during certain time periods. Let $d_{i,j}$ be the number of time periods required for operating a service between terminals i and j. We assume that a service duration is expressed as an integer multiple of time unit (or period) and is shorter than the schedule length as well. Thus, for each service (denoted by $< i, j >$) between terminals i and $j \in L$ and for any time period $t \in T$, we create a corresponding service arc $(i_t, j_{(t+d_{i,j}) \mod t_{max}})$. Due to the cyclic nature of the studied problem, the time-space network wraps around (Crainic et al. 2016). In particular, we model a service $< i, j >$ with duration $d_{i,j}$ that departs from terminal i at period t such that $t + d_{i,j} > t_{max}$ as arriving at the destination at period $((t + d_{i,j}) \mod t_{max})$. Each service $< i, j >$ is associated with a capacity $u_{i,j}$, which restricts the maximum shipment demand that service arc (i, j) can fulfill. *The second set of holding arcs*, denoted by H, is composed of arcs connecting nodes located at the same terminal but having consecutive time realizations. In particular, for each pair of terminal l and period t, the holding arc is denoted by $(l_t, l_{(t+1) \mod t_{max}})$. Holding arcs show that commodities and resources are waiting at specific terminals. To reflect the real-world operations, we set an infinite capacity for holding arcs. It is clear that the total number of holding arcs in H is equivalent to the number of physical terminals times the schedule length because resources can wait at any terminals at any time periods. For illustration purpose, we use Figure 1 to show an example of building a time-space network from the physical network. There are three terminals in the physical network and five time periods. We can see that the time-space network includes fifteen nodes and fifteen service arcs.

Demands are defined in terms of the number of commodities that are needed to be transported through the network. Let K be the set of commodities. For each commodity $k \in K$, there are q^k such commodities needed to be transported from their origin $o(k)$ to destination $d(k)$. We define

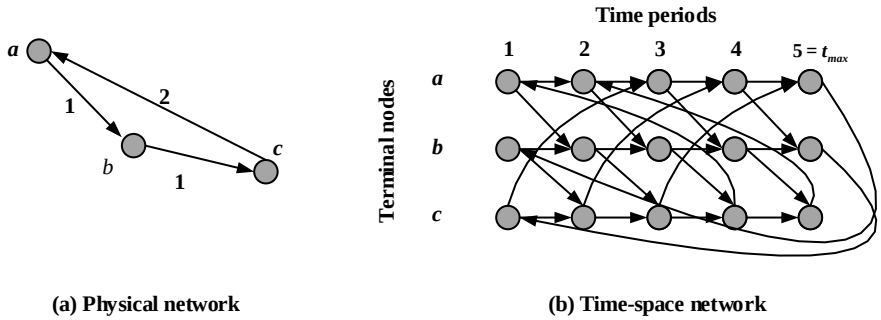


Figure 1 **Physical Network and Time-space Network**

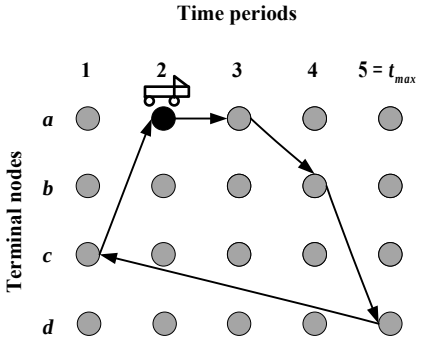


Figure 2 **Resource Cycle in the Time-space Network**

$c_{i,j}^k$ to be the cost of sending commodity k along arc (i,j). We choose to ignore the per arc costs as the costs of acquiring a resource are dominating all other costs (Andersen et al. 2009). The origin and destination of each commodity is known beforehand. Each commodity can be distributed through any itinerary, made up of service arcs and holding arcs. In the SPSNDRC, one of its unique features is that the flow of each commodity must be served along a unique path linking its origin and destination, which prevents the flow partition. This constraint is inspired by real-life instances, e.g., customers prefer to receive goods at one time, instead of in batches. Moreover, it helps to release the stress of sub-package operations and reduce the risk of transportation errors considerably.

A limited number of homogeneous resources are used to perform transportation services. Let V be the set of resources. Each terminal l contains a set of resources (denoted by $V_l \subseteq V$) and each resource v belongs to a particular terminal. Each resource $v \in V$ has a fixed cost F_v and is associated with a set of valid cycles (denoted by θ_v). In particular, due to the cyclic nature of the problem, the itinerary that a resource travels is a cycle with a sequence of service arcs and possibly, holding arcs, satisfying the design-balanced constraint. Following the full resource-utilization policy, resources must return to their home terminals after services. Specifically, for a specific terminal l, resource $v \in V_l$ must depart from one node in l_T, where $l_T = \{l_t : t \in T\}$, and then go back to the same

node. In addition, we assume that a service arc is executed by at most one resource, with no such limit applied for holding arcs though, by following existing studies in the literature (e.g., Crainic et al. (2016)). Finally, we use Figure 2 to show an example of resource operation, where one vehicle starts from terminal a at period 2 and returns to the same node after performing the service. The corresponding physical cycle is $a \to b \to d \to c \to a$ in the physical network. For each physical cycle τ', let $\chi(\tau')$ be the sum of duration of arcs included in τ'.

4. Mathematical Formulations

In the SPSNDRC, one decision is selecting resource services and their schedules. Such schedules are cyclic itineraries for the available resources. Note that an itinerary is actually a sequence of service arcs and possibly holding arcs in the time-space network. We thus can present formulations with decision variables associated with either arcs, or cycles.

In this section, an arc-based formulation (node-arc model) is first introduced. This model is straightforward and can be solved using MIP solvers (e.g., CPLEX). However, as our computational experiments will show, the computational difficulties also explode significantly as the instance size increases. We then propose a cycle-based formulation, with which we introduce an MIP-based approach to solve the SPSNDRC in the next section. This method includes particular column generation subproblems for dynamically constructing cycles, which can further contribute to decreasing the model size and thus reducing the computational efforts.

4.1. Notation

We further give the following notation.

Parameters

- T_i and T_j: the departure and arrival times of arc (i,j), respectively.
- t_k: the time period when commodity k becomes available for distribution. For the purpose of presentation, we define $t_k - 1 = t_{max}$, if $t_k = 1$.
- $r^{\tau}_{i,j}$: it takes value 1 if arc (i,j) is part of cycle τ, and 0 otherwise.

Sets

- A_t: the set of arcs across period t in the time-space network, i.e., $A_t = \{(i,j) \in A : T_i \leq t < T_j\}$.
- B_k: the set of arcs starting before period t_k and ending not later than t_k, i.e., $B_k = \{(i,j) \in A : T_i \leq t_k - 1 < T_j\}$.

Decision variables

- $x^k_{i,j}$: it indicates if arc (i,j) is used to distribute flow of commodity k (i.e., $x^k_{i,j} = 1$) or not.
- $y^v_{i,j}$: it indicates if arc (i,j) is traveled by resource v (i.e., $y^v_{i,j} = 1$) or not.
- δ_v: it indicates if resource v is used in the final schedule (i.e., $\delta_v = 1$) or not.
- z^{τ}_v: it is equal to 1 if cycle τ is chosen as the itinerary of resource v.

4.2. Node-Arc Formulation

We first present the node-arc formulation for the SPSNDRC. With this formulation, we can solve small problem instances, using MIP solvers. The integer feasible solution obtained by this formulation will be used as a benchmark for evaluating the proposed column-generation-based approach in the next section. The node-arc formulation can be described as follows:

$$\min Z = \sum_{v \in V} F_v \delta_v + \sum_{k \in K} \sum_{(i,j) \in A} q^k c^k_{i,j} x^k_{i,j} \tag{1}$$

$$\sum_{(i,j) \in A} x^k_{i,j} - \sum_{(j,i) \in A} x^k_{j,i} = \begin{cases} 1, & \text{if } i = o(k) \\ -1, & \text{if } i = d(k) \\ 0, & \text{otherwise} \end{cases} \quad \forall i \in N, \forall k \in K, \tag{2}$$

$$\sum_{k \in K} q^k x^k_{i,j} \le u_{i,j} \sum_{v \in V} y^v_{i,j}, \quad \forall (i,j) \in S, \tag{3}$$

$$\sum_{v \in V} y^v_{i,j} \le 1, \quad \forall (i,j) \in S, \tag{4}$$

$$\sum_{(i,j) \in A_t} y^v_{i,j} = \delta_v, \quad \forall t \in T, \forall v \in V, \tag{5}$$

$$\sum_{(i,j) \in A : i \in l_T} y^v_{i,j} \ge \delta_v, \quad \forall v \in V_l, \forall l \in L, \tag{6}$$

$$\sum_{(i,j) \in A} y^v_{i,j} - \sum_{(j,i) \in A} y^v_{j,i} = 0, \quad \forall i \in N, \forall v \in V, \tag{7}$$

$$\sum_{(i,j) \in B_k} x^k_{i,j} = 0, \quad \forall k \in K, \tag{8}$$

$$y^v_{i,j} \in \{0,1\}, \quad \forall (i,j) \in A, \forall v \in V, \tag{9}$$

$$\delta_v \in \{0,1\}, \quad \forall v \in V, \tag{10}$$

$$x^k_{i,j} \in \{0,1\}, \quad \forall (i,j) \in A, \forall k \in K. \tag{11}$$

The objective function (1) is to minimize the total cost including the fixed costs of resources and flow costs of commodities. Flow conservation is enforced by constraints (2). Constraints (3) state that the total commodity quantity on every service arc cannot exceed its capacity. Constraints (4) regulate that every service arc can be used by at most one resource. Then, equations (5) state that for each period, if a resource is utilized, it should be engaged in only one activity. Constraints (6) state that each resource must depart from its home terminal, if it is used. Design-balanced requirements are implied by constraints (7). Equations (8) are introduced to exclude the case that a commodity flow travels across one period more than once. We use the example in Figure 3 to illustrate constraints (8). In this example, we assume that between nodes a and b, b and c, and b and

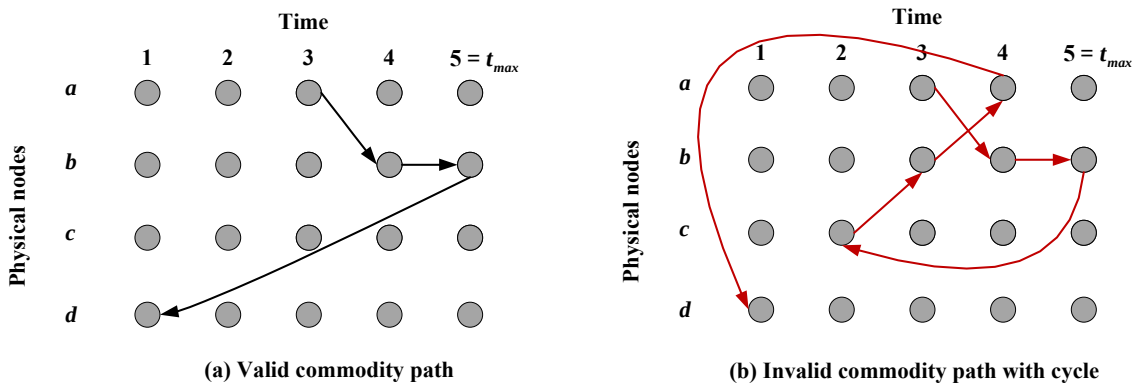


Figure 3 **Illustration of Constraints** (8)

d, the traveling time for each pair is one period, and between a and d, the duration is three periods. Remember that in the SND literature, it is generally assumed that the total transportation duration of each commodity is not greater than t_{max}. Figure 3(a) gives a valid commodity path where the total traveling duration is three periods. But in Figure 3(b), the path for the same commodity covers eight periods, greater than $t_{max} = 5$. We can avoid this scenario through constraints (8). Finally, variable restrictions are enforced by constraints (9)-(11).

4.3. Arc-Cycle Formulation

We next present the arc-cycle formulation for the SPSNDRC. With this formulation, due to its nature of incorporating service cycles, we propose a column-generation-based approach to solve it in the next section. We formulate the arc-cycle model as follows:

$$\min Z = \sum_{v\in V}\sum_{\tau\in\theta_v} F_v z_v^\tau + \sum_{k\in K}\sum_{(i,j)\in A} q^k c_{i,j}^k x_{i,j}^k \tag{12}$$

$$\sum_{(i,j)\in A} x_{i,j}^k - \sum_{(j,i)\in A} x_{j,i}^k = \begin{cases} 1, & \text{if } i = o(k) \\ -1, & \text{if } i = d(k) \\ 0, & \text{otherwise} \end{cases} \quad \forall i \in N, \forall k \in K, \tag{13}$$

$$\sum_{k\in K} q^k x_{i,j}^k \le u_{i,j} \sum_{v\in V}\sum_{\tau\in\theta_v} r_{i,j}^\tau z_v^\tau, \quad \forall (i,j) \in S, \tag{14}$$

$$\sum_{v\in V}\sum_{\tau\in\theta_v} r_{i,j}^\tau z_v^\tau \le 1, \quad \forall (i,j) \in S, \tag{15}$$

$$\sum_{\tau\in\theta_v} z_v^\tau \le 1, \quad \forall v \in V, \tag{16}$$

$$\sum_{(i,j)\in B_k} x_{i,j}^k = 0, \quad \forall k \in K, \tag{17}$$

$$z_v^\tau \in \{0,1\}, \quad \forall \tau \in \theta_v, \forall v \in V, \tag{18}$$

$$x_{i,j}^k \in \{0,1\}, \quad \forall (i,j) \in A, \forall k \in K. \tag{19}$$

The objective function (12) minimizes the summation of fixed cost for the resources and the flow costs. Flow conservation, expressed by (13), remains unchanged. Constraints (14) and (15) are identical to (3) and (4), while constraints (16) ensure each resource to be used no more than once. Commodity related constraints (17) are unchanged and constraint (18)-(19) are the variable-type constraints. Route-length request, design-balanced requirement and the demand for resources departing from their origins are needless in this new formulation since these constraints are already considered in the process of cycle-searching in advance.

The SPSNDRC formulation (12)-(19) with all variables in the constraint matrix is called the master problem (MP). We refer to a MP with only a subset of its columns as the restricted master problem (RMP). In our proposed MP-based heuristic in Section 5, the column generation procedure needs to solve the LP relaxation of MP by solving the LP relaxations of several RMPs. Hereafter, we refer to the linear relaxation of the RMP as the L-RMP. We now discuss the definition of the pricing problem, which will be solved in column generation. Let $\alpha_{i,j}$, $\beta_{i,j}$, and ϕ_v respectively, be dual variables associated with constraints (14)-(16). Then, the reduced cost π_v^τ of variable z_v^τ is calculated as

$$\pi_v^\tau = F_v - \phi_v + \sum_{(i,j)\in S} u_{i,j}\alpha_{i,j} r_{i,j}^\tau - \sum_{(i,j)\in S} \beta_{i,j} r_{i,j}^\tau = F_v - \phi_v + \sum_{(i,j)\in S} (u_{i,j}\alpha_{i,j} - \beta_{i,j}) r_{i,j}^\tau.$$

For a resource v, F_v, ϕ_v, and $u_{i,j}$ are all determined. Then, the pricing problem is equivalent to searching for a cycle τ such that $\sum_{(i,j)\in\tau} (u_{i,j}\alpha_{i,j} - \beta_{i,j})$ is minimized. In Section 5.3, we will detail how the pricing problem is solved.

5. Solution Approach

In the arc-cycle model, the number of cycles grows exponentially with increase of the schedule length, and hence the set of cycles is too large to enumerate, even for reasonably sized instances. Thus, further decreasing the number of cycle-related variables is essential for large-scale problems. In this section, we propose an MIP-based heuristic to produce high-quality solutions to the SPSNDRC. Algorithm 1 shows the outline of the proposed approach. This is a two-stage method. In the first stage, the column generation procedure is executed to solve the L-RMP. As a result, a subset of design-cycles for given resources at each terminal will be generated. In the second stage, feasible integer solutions for the SPSNDRC are created from the optimal solution of the L-RMP.

The main advantage of applying column generation is that not all the possible cycles of resources need to be enumerated. Instead, the problem can be first formulated as an RMP, which contains only a few cycles, as long as the linear relaxation of the model is feasible. Then the pricing problem

Algorithm 1. MIP-based Heuristic.

input instance data.
/* Model Initialization */
for each l with $|V_l| > 0$ **do**
for each v in V_l **do**
construct physical cycle τ' in $\tilde{G}$.
for each τ' with $\chi(\tau') \leq t_{max}$, construct cycles τ in G.
end for
end for
initialize the L-RMP using the constructed cycles.
solve the RMP.
if the L-RMP is infeasible, **then** return: no solution is found.
/* Stage 1: Column Generation Execution*/
repeatedly solve the pricing problem until no valid cycle is found (see Procedure 1).
/* Stage 2: Producing a Feasible Solution */
choose one heuristic strategy from H1, H2, H3 and H4.
build a restricted MIP model based on the optimal solution of the ultimate L-RMP.
solve the small MIP model with solver CPLEX.
if this MIP model is infeasible, **then** return: no solution is found.
else
return: the best integer solution.
end if

is solved to select a new column with a negative reduced cost to be added to the L-RMP. This process is repeated until no more columns can be added to the L-RMP. One can then conclude that the current L-RMP solution is an optimal solution of the linear relaxation of the original arc-cycle model yielding thus a lower bound of its integer version. After that, we introduce heuristic strategies to get integer solutions from the optimal solution of the L-RMP. Detailed implementations of the general column generation and branch-and-price can be found in Andersen et al. (2011), Crainic et al. (2016) and Li et al. (2017a). We now detail the proposed approach.

5.1. Restricted Master Problem and Its Linear Relaxation

In the context of the SPSNDRC, the RMP is created by including all the x-variables and some z-variables defined over a subset $\overline{\theta}$ of resource cycles. In particular, we work on a subset $\overline{\theta}_v$ for each resource v. We will detail how to generate $\overline{\theta}_v$ in the next subsection. To get the linear relaxation of the RMP, we drop integral constraints for variables and replace (18)-(19) with the following two constraints:

$$0 \leq x_{i,j}^k \leq 1, \quad \forall (i,j) \in A, \forall k \in K, \tag{20}$$

$$0 \leq z_v^\tau \leq 1, \quad \forall \tau \in \overline{\theta}_v, \forall v \in V. \tag{21}$$

The L-RMP is then presented as

$$\min Z = \sum_{v \in V} \sum_{\tau \in \overline{\theta}_v} F_v z_v^\tau + \sum_{k \in K} \sum_{(i,j) \in A} q^k c_{i,j}^k x_{i,j}^k$$

(13) and (17),

$$\sum_{k\in K} q^k x^k_{i,j} \le u_{i,j} \sum_{v\in V}\sum_{\tau\in\overline{\theta}_v} r^\tau_{i,j} z^\tau_v, \quad \forall (i,j)\in S,$$

$$\sum_{v\in V}\sum_{\tau\in\overline{\theta}_v} r^\tau_{i,j} z^\tau_v \le 1, \quad \forall (i,j)\in S,$$

$$\sum_{\tau\in\overline{\theta}_v} z^\tau_v \le 1, \quad \forall v\in V,$$

$$(20)-(21).$$

5.2. Model Initialization

The column generation algorithm solves the LP relaxation of MP by solving the LP relaxations of several RMPs. To perform the column generation and further heuristic strategies for building integer solutions, we first need to generate some feasible cycles for the resources, and thus initialize the L-RMP. Given terminal l and its associated resource set V_l, initial design-cycle variables z^τ_v are added with three steps, which we discuss in detail in the remainder of this section.

Step 1. In the physical network $\tilde{G}$, construct all cycles τ' starting from terminal l, such that the total arc duration of each cycle does not exceed the schedule length t_{max};

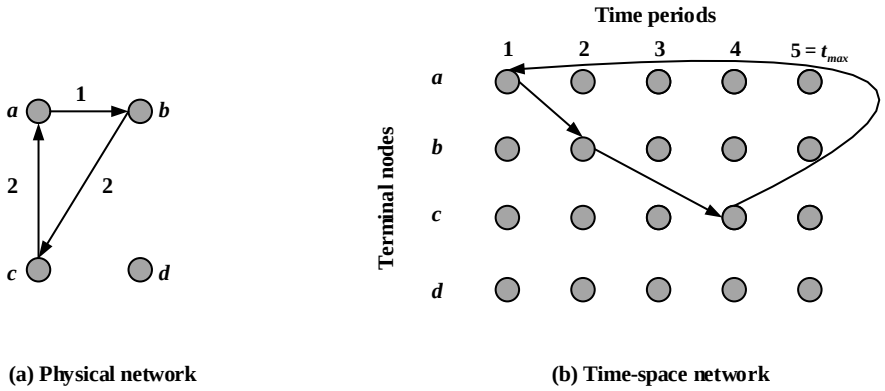
Step 2. Based on each physical cycle τ', construct cycles τ in the time-space network G;

Step 3. Add design-cycle variables.

Step 1. Construct physical cycle τ'. The first step is to find all the cycles starting from terminal l in graph $\tilde{G}$. Such cycles are referred to as physical cycles. To identify all the physical cycles, we implement the search algorithm in Ahuja et al. (1993).

The search algorithm can find all terminals in graph $\tilde{G}$ that can reach a specific terminal l along directed paths. In particular, we implement the search algorithm to find a path starting from terminal l and ending at terminal l. Note that any valid physical cycle τ' should not contain inner loop and has a total arc duration not greater than the schedule length t_{max}. To that end, We record the accumulated duration of the path, while implementing the search algorithm. The search algorithm might be terminated earlier if the accumulated duration is greater than t_{max}. Let $\Omega(l)$ be the set of all valid physical cycles at terminal l. For each τ' in $\Omega(l)$, let $\chi(\tau')$ be the sum of duration of arcs included in τ'.

Step 2. Construct cycles τ in the time-space network. The second step is to construct a set of cycles in the time-space network, based on a physical cycle τ'. The underlying idea is that we first create a cycle in the time-space network by making one physical cycle start at any involved terminal and at any period $t\in T$. If the resulting cycle has a duration being less than t_{max}, we then append some holding arcs to this cycle such that its duration is exactly equal to the schedule length. To simplify our presentation, we assume that a physical cycle τ' is associated with terminal



(a) Physical network **(b) Time-space network**

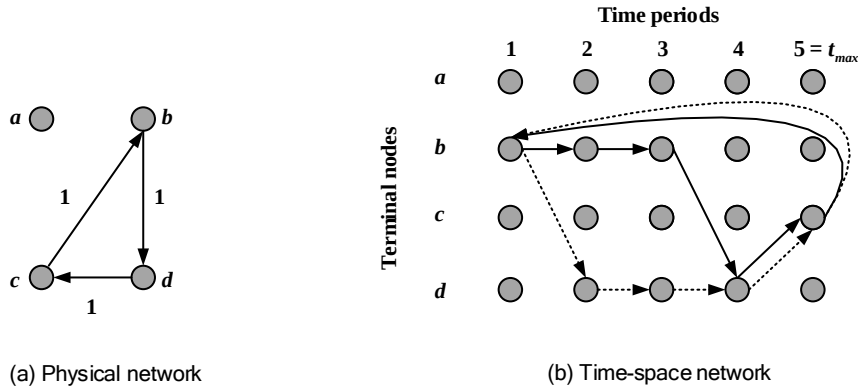
Figure 4 **Physical Cycle τ' with $\chi(\tau') = t_{max}$**

l, and is denoted as a sequence of terminals, i.e., $l \to j \to k \to \cdots \to l$. Now physical cycle τ' may fall into one of the following two cases:

Case 1. $\chi(\tau') = t_{max}$. To construct a cycle in the time-space network, the resource can start from terminal l at any period $t \in T = \{1, 2, \ldots, t_{max}\}$. Each cycle scenario defines one valid cycle starting from one period $t \in T$ in the time-space network. Let φ be the number of cycle scenarios that will be considered in the algorithm implementation. Obviously, we can consider at most $\varphi = t_{max}$ cycle scenarios in this case. As an illustration, Figure 4(b) displays one cycle starting at terminal a in period 1. It corresponds to the physical cycle τ' in Figure 4(a). Let Γ_1 be the set of cycles constructed in Case 1.

Case 2. $\chi(\tau') < t_{max}$. Recall that for a resource, the duration of its itinerary (cycle) must be equal to the schedule length t_{max}. In this case, we need to construct in the time-space network a cycle τ with duration $\chi(\tau)$ equal to t_{max}. To that end, we can add $\Delta = t_{max} - \chi(\tau')$ holding arcs to cycle τ' such that its duration can be increased to t_{max}. Note that these additional holding arcs may be inserted between any two consecutive terminals in τ'. For our considered instances in Section 6.1, we chose to insert additional holding arcs between the first and second terminals, and between the second and third terminals. Furthermore, the resource can start from terminal l at any period $t \in T = \{1, 2, \ldots, t_{max}\}$. As a result, each scenario defines one valid cycle in the time-space network. Now look at physical cycle $\tau' = l \to j \to k \to \cdots \to l$. For illustration, consider $\varphi = 2$ and starting period of 1. From τ', we are able to construct two cycles in the time-space network, which both start at period 1.

cycle 1 :
$l_1 \to l_2 \to \cdots \to l_{1+\Delta} \to j_{[(1+\Delta+d_{l,j}) \mod t_{max}]} \to k_{[(1+\Delta+d_{l,j}+d_{j,k}) \mod t_{max}]} \to \cdots \to l_1,$
cycle 2 :
$l_1 \to j_{[(1+d_{l,j}) \mod t_{max}]} \to j_{[(2+d_{l,j}) \mod t_{max}]} \to j_{[(1+\Delta+d_{l,j}) \mod t_{max}]}$
$\to k_{[(1+\Delta+d_{l,j}+d_{j,k}) \mod t_{max}]} \to \cdots \to l_1,$



(a) Physical network (b) Time-space network

Figure 5 **Physical Cycle τ' with $\chi(\tau') < t_{max}$**

where $j_{[(1+d_{l,j}) \mod t_{max}]}$ and $j_{[(1+\Delta+d_{l,j}) \mod t_{max}]}$ are terminal j at periods $[(1+d_{l,j}) \mod t_{max}]$ and $[(1+\Delta+d_{l,j}) \mod t_{max}]$, respectively. $k_{[(1+\Delta+d_{l,j}+d_{j,k}) \mod t_{max}]}$ denotes terminal k at period $[(1+\Delta+d_{l,j}+d_{j,k}) \mod t_{max}]$. Note that we can get different cycles if we change the starting period of τ'. Let Γ_2 be the set of cycles constructed in Case 2.

We now use example in Figure 5 to illustrate the above method. The physical cycle at terminal b has a duration of three period, e.g., $\tau' = b \to d \to c \to b$. Since the schedule length is 5, we need to add two holding arcs to τ'. Suppose we add these arcs between b and d, d and c. We can get two constructed cycles: $b_1 \to b_2 \to b_3 \to d_4 \to c_5 \to b_1$ and $b_1 \to d_2 \to d_3 \to d_4 \to c_5 \to b_1$, which both start at period 1 (see Figure 5(b)).

Step 3. Add design-cycle variables. Let $\Gamma = \Gamma_1 \cup \Gamma_2$. For each resource v, we add a design-cycle variable z_v^τ for each cycle τ in Γ.

Using these generated design-variables, we then initialize the L-RMP. Note that given a φ, this initialization method may not ensure that the restricted network derived from these constructed cycles can support the transportation of commodity demands. Our experimental experience shows that this method with $\varphi = 2$ can produce feasible L-RMP for all the considered instances in Section 6. If the resulting L-RMP is infeasible, we can increase the value of φ to produce more cycles until the resulting L-RMP becomes feasible.

5.3. Stage 1: Column Generation Execution

Once the L-RMP solution is obtained, the pricing problem is solved to check whether there exists some design cycle with a negative reduced cost, in which case, the corresponding decision variable would be added to the current L-RMP and might improve the optimum value of the current L-RMP.

Given a time-space network G, we first create a new graph G' by appending a copy of G after G. As an example, see Figure 6. Recall that for a resource, any valid cycle should start from and end at the same terminal in graph G. To illustrate, consider in Figure 2 the resource and its cycle:

$a_2 \to a_3 \to b_4 \to d_5 \to c_1 \to a_2$. As one may observe, this cycle corresponds to one path in graph G': $a_2 \to a_3 \to b_4 \to d_5 \to c_{1+t_{max}} \to a_{2+t_{max}}$ (see Figure 6). Following this equivalence, finding a cycle with a minimum $\sum_{(i,j)\in\tau} (u_{i,j}\alpha_{i,j} - \beta_{i,j})$ reduces to finding a shortest path in graph G', where each arc weight $w_{i,j}$ is set equal to $u_{i,j}\alpha_{i,j} - \beta_{i,j}$. Procedure 1 presents how one new column is added to the current L-RMP.

Consider terminal l with $|V_l| > 0$. For resource v at terminal l, each valid cycle can start from any time period in the schedule horizon. We thus need to solve t_{max} shortest path problems. In particular, for time period t, we solve in graph G' a shortest path problem with source l_t and sink $l_{t+t_{max}}$. Let p and $\lambda(p)$ be the returned shortest path and its length, respectively. Path p defines a cycle τ in graph G. If $F_v - \phi_v + \lambda(p) < 0$ (negative reduced cost), we add a new column z_v^τ to the current L-RMP, for resource v at terminal l.

Procedure 1. Implementation of the Pricing Algorithm.

```
input the current L-RMP.
input graph G.
input dual variables set α_{i,j}, β_{i,j} and φ_v.
create graph G' and set each arc weight w_{i,j} = u_{i,j}α_{i,j} − β_{i,j}.
for each l with |V_l| > 0 do
  for t = 1 to t_max do
    solve the shortest path problem with source l_t and sink l_{t+t_max}.
    let p be the returned shortest path in G' and the corresponding cycle τ in graph G.
    let λ(p) be the length of p.
    for each v in V_l, do
      if F_v − φ_v + λ(p) < 0, do
          add the corresponding variable z_v^τ to the current L-RMP.
      end if
    end for
  end for
end for
return the updated L-RMP.
```

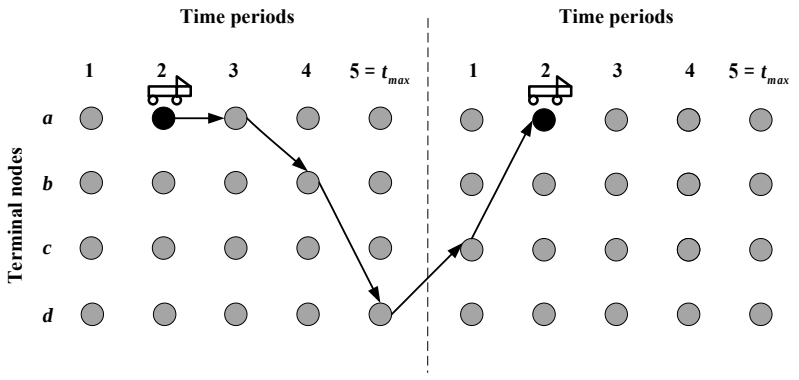


Figure 6 **Graph** G'

5.4. Stage 2: Heuristic Strategies for Building a Feasible Solution

The solution to the ultimate L-RMP provides a lower bound to the SPSNDRC, and is often not integral. In this section, we present four heuristic strategies that aim at building integer solutions starting from the solution of the last L-RMP. The fundamental idea is to first build a smaller MIP model, derived from the optimal solution $(\overline{x}, \overline{z})$ of the ultimate L-RMP, and then solve this restricted model by solver CPLEX.

5.4.1. Heuristic Strategy 1: Variables Conversion

The most intuitive idea is to apply MIP solver to the model by considering all the design-cycle variables used to optimally solve its continuous relaxation. That is, we change in the ultimate L-RMP all the design-cycle variables to binaries and solve the resulting MIP model. Note that this MIP model is defined over the original time-space network G and includes all x- and z- variables. Hereafter, we refer to this greedy strategy as H1.

5.4.2. Heuristic Strategy 2: Fixing Cycles

This strategy aims to make use of the potential information behind the optimal solution of the final L-RMP. For large-scale problem instances, the final L-RMP model contains a lot of design-cycle variables z_v^τ for each resource. On the other hand, only a small number of itineraries take positive values in the solution, while most of the design-cycle variable values are zeros. Thus, we conjecture that in the final solution, the design-cycle variables taking higher values are more promising than the others. That is, the corresponding cycles are more likely to be used to transport more goods than those cycles with design cycle variable values close to zero.

Following the above observation, the main idea of the second heuristic strategy lies in that among cycles of each resource, enumerated in the column generation procedure, ϖ cycles are chosen and fixed as potential cycles of this resource in the final schedule. These chosen cycles lead to a relatively smaller time-space network with fewer service and holding arcs. In this smaller network, we then solve an arc-cycle model for the SPSNDRC, where one decision is made for commodity flow distribution. The second heuristic strategy works as follows.

Let $\overline{\psi}_v$ be the set of cycles fixed for resource v. Consider a solution $\overline{z}$ to the ultimate L-RMP. For resource v, let ψ_v be the set of cycles enumerated in the column generation procedure. Without loss of generality, suppose $\psi_v = \{\tau_1, \tau_2, \cdots, \tau_{|\psi_v|}\}$ with $\overline{z}_v^{\tau_1} \geq \overline{z}_v^{\tau_2} \geq \cdots \geq \overline{z}_v^{\tau_{|\psi_v|}}$. Define $\tilde{V} = \{v : |\psi_v| > 0\}$. For each v in $\tilde{V}$, we choose the first $\min\{\varpi, |\psi_v|\}$ cycles in ψ_v, and add them into set $\overline{\psi}_v$. Define $\overline{\psi} = \cup_{v \in \tilde{V}} \overline{\psi}_v$. Cycles in $\overline{\psi}$ define a "smaller" time-space network $G^* = (N^*, A^*)$, where node set N^* and arc set A^* only include nodes in cycles in $\overline{\psi}$. Using $\tilde{V}$, $\overline{\psi}$, K, and G^*, we then develop a restricted arc-cycle model, which is finally solved by CPLEX. Hereafter, we refer to this heuristic strategy as H2. Its performance depends on the settings of parameter ϖ. A small ϖ may lead to an

infeasible model, whereas a large ϖ can result in many useless itineraries in the model, and thus increases computational difficulty as well. In Section 6, we will examine the effect of parameter ϖ on the performance of heuristic strategy H2.

5.4.3. Heuristic Strategy 3: Fixing Single-paths

Similar to the second strategy, this strategy aims to decrease the model scale and solution difficulty through fixing commodity flow variables. After the column generation procedure, we examine all the flow variables in the optimal solution of the ultimate L-RMP, and determine the set K_1 of commodities, of which the demands have already been transported along single-paths. In our implementation, we chose to randomly fix single-paths for a certain proportion (denoted as η) of commodities in K_1. Namely, from set K_1, $\lfloor \eta |K_1| \rfloor$ commodities are randomly chosen to follow the single-paths, which are by solution $\overline{x}$. Such chosen commodities form set K_2. Given the L-RMP solution $\overline{x}$, let $\overline{SG}$ be the set of arcs, included in single-paths for commodities in K_2. That is, $\overline{SG} = \{(i,j) \in A : \exists k \in K_2, \overline{x}^k_{i,j} = 1\}$. Each arc in $\overline{SG}$ should be traveled exactly once by one resource. Further, we re-define the capacities of arcs in $\overline{SG}$, by subtracting flows of commodities, which use these arcs. Let $\overline{\theta}_v$ be the set of cycles for resource v, after the column generation procedure is executed. Let $\overline{K} = K \backslash K_2$.

We then solve a smaller arc-cycle model to produce potential feasible solutions to the SPSNDRC.

$$\min Z' = \sum_{v \in V} \sum_{\tau \in \overline{\theta}_v} F_v z^\tau_v + \sum_{k \in \overline{K}} q^k c^k_{i,j} x^k_{i,j} \tag{22}$$

$$\sum_{(i,j) \in A} x^k_{i,j} - \sum_{(j,i) \in A} x^k_{j,i} = \begin{cases} 1, & i = o(k) \\ -1, & i = d(k) \\ 0, & \text{otherwise} \end{cases} \quad \forall i \in N, \forall k \in \overline{K}, \tag{23}$$

$$\sum_{k \in \overline{K}} q^k x^k_{i,j} \le u_{i,j} \sum_{v \in V} \sum_{\tau \in \overline{\theta}_v} r^\tau_{i,j} z^\tau_v, \quad \forall (i,j) \in S, \tag{24}$$

$$\sum_{v \in V} \sum_{\tau \in \overline{\theta}_v} r^\tau_{i,j} z^\tau_v \le 1, \quad \forall (i,j) \in S, \tag{25}$$

$$\sum_{\tau \in \overline{\theta}_v} z^\tau_v \le 1, \quad \forall v \in V, \tag{26}$$

$$\sum_{v \in V} \sum_{\tau \in \overline{\theta}_v} r^\tau_{i,j} z^\tau_v = 1, \quad \forall (i,j) \in \overline{SG}, \tag{27}$$

$$\sum_{(i,j) \in B_k} x^k_{i,j} = 0, \quad \forall k \in \overline{K}, \tag{28}$$

$$z^\tau_v \in \{0,1\}, \quad \forall \tau \in \overline{\theta}_v, \forall v \in V, \tag{29}$$

$$x^k_{i,j} \in \{0,1\}, \quad \forall k \in \overline{K}, \forall (i,j) \in A. \tag{30}$$

Equation (22) is the objective, but its value isn't the final cost because the commodity transportation costs for all the $k \in K_2$ need to be added then. Constraints (23), (24), (25), (26), and (28) are easy to understand since they are parallel to (13)-(17). Constraints (27) are newly added and they guarantee that the arcs chosen are served by resources. We refer to the third heuristic strategy as H3.

5.4.4. Heuristic Strategy 4: Simultaneously Fixing Cycles and Single-paths

Lastly, we try to further reduce the instance scale and solution complexity by simultaneously fixing resource cycles and commodity single-paths. It is expected to find potentially better solutions for some instances. On the one hand, this strategy should guarantee that service arcs in the fixed cycles can support the demand distribution of the fixed commodities. On the other hand, the fixed cycles should be able to support the distribution of those commodities, of which paths are not fixed. We refer to this strategy as H4.

6. Computational Evaluation

In this section, we evaluate the performance of the proposed approach. Our approach was coded in C and compiled on VC++ 2010. As a comparison benchmark, CPLEX was implemented to solve each instance with the node-arc formulation. We set a time limit of 10 hours for CPLEX to solve each instance, and let CPLEX focus on finding hidden feasible solutions.

Through preliminary tuning, we set some selected parameters in CPLEX as follows: (1) CPXPARAM_Emphasis_MIP =4. It lets CPLEX focus on finding hidden feasible solutions; (2) CPXPARAM_MIP_Strategy_FPHeur = 1. The feasibility pump heuristic tries to find a feasible solution without taking the objective function into account; (3) CPXPARAM_MIP_Strategy_RINSHeur = 10; (4) CPX_PARAM_PROBE = 1, a moderate probing level; and (5) CPX_PARAM_NODESEL = 1, a best-bound search strategy. Note that although the above parameter setting is used in our experiments, we still do believe that all CPLEX strategies potential are maybe, not fully explored in CPLEX's MIP Solver for the instances studied in this paper. We implemented each heuristic strategy with one hour after the standard column generation procedure. We implemented all the experiments on a PC 2.0 GHz with 8GB RAM.

6.1. Test Instances

Since the SPSNDRC is a new service network design problem, there are no benchmark instances in the literature. However, recall that the SPSNDRC is an extension of the SNDAM and that Andersen et al. (2011) evaluated their approach using 43 instances. We thus chose to generate SPSNDRC instances based on these SNDAM instances. We generated 40 problem instances, which have the same network structure and commodity demands as the first 40 instances of the data set

in Andersen et al. (2011). In particular, we increased original arc capacities and the number of resources, in order to ensure the generated instances are feasible when "single-path" constraints are introduced. In the following, we describe how instances are generated in detail.

Let γ_1 be the number of resources in an instance in Andersen et al. (2011). Let γ_2 be the number of resources which we add to the original Anderson's instance. That is, there are $(\gamma_1+\gamma_2)$ resources in each generated instance. Let γ_3 be the capacity which we add to the original capacity of each service arc. Then we generated each instance through the following five steps:

Step 1. Set $\gamma_2=\gamma_3=0$.

Step 2. We assign resources to physical terminals in turn.

Step 3. We solve the node-arc model for the SPSNDRC using CPLEX with a time limit of 10 hours. If CPLEX demonstrates it is feasible, a new instance has been generated and the procedure terminates. If within 10 hours, CPLEX cannot validate feasibility of the instance, then we decide that the instance is infeasible.

Step 4. If $\gamma_2<10$, we add μ resources to the instance (e.g., $\gamma_2=\gamma_2+\mu$) (μ is set to 2 for small and medium instances, and 5 for large instances), and go to Step 2.

Step 5. If $\gamma_3<70$, we increase each arc capacity by 10, set $\gamma_3=\gamma_3+10$, and go to Step 2.

Table 1 displays the characteristics of instances. As an indicator of the computational complexity, we display in the last two columns of Table 1, the number of x- and y-variables in each node-arc formulation. In terms of this, we refer to instances 1-15 as small instances, instances 16-25 as medium instances, and instances 26-40 as large instances.

6.2. Computational Results

6.2.1. Results of the Node-arc Model

To analyze the solution complexity of the SPSNDRC, we first solve the node-arc model using CPLEX with a time limit of 10 hours. Table 2 summarizes the results. In each case, we report in columns 2 and 3 the lower bound (LB) and the best MIP solution (UB_CPLEX) at termination respectively, which occurred after 10 hours or once the algorithm had found a provably optimal integer solution. Column 4 displays for each instance the percentage gap between the lower bound and upper bound at termination. Column 5 displays the solution time in CPU seconds for finding the provably optimal solution, or at termination. As a comparison, columns 6-8 display the results of the CG procedure, including the lower bound, the number of cycles generated, and the corresponding computation time.

The results in Table 2 demonstrate the solution difficulty of the SPSNDRC instances. As one may observe, CPLEX within 10 hours found a provably optimal solution of only three instances. Although CPLEX could obtain feasible solutions for the other 11 instances, the relative gaps

Table 1 Characteristics of Problem Instances

Instance	$\|L\|$	$\|E\|$	t_{max}	$\|N\|$	$\|S\|+\|H\|$	$\|K\|$	$\|V\|$	$\#x$	$\#y$
1	5	10	15	75	150+75	20	8	4500	1800
2	5	15	20	100	300+100	25	8	10000	3200
3	5	15	25	125	375+125	25	8	12500	4000
4	5	15	15	75	225+75	100	25	30000	7500
5	5	15	15	75	225+75	200	54	60000	16200
6	5	15	40	200	600+200	200	50	160000	40000
7	5	15	40	200	600+200	200	30	160000	24000
8	5	15	40	200	600+200	200	50	160000	40000
9	5	15	40	200	600+200	200	50	160000	40000
10	5	15	40	200	600+200	200	25	160000	20000
11	7	30	30	210	900+210	200	65	222000	72150
12	7	30	30	210	900+210	200	40	222000	44400
13	7	30	30	210	900+210	200	65	222000	72150
14	7	30	30	210	900+210	200	40	222000	44400
15	7	30	30	210	900+210	200	35	222000	38850
16	5	15	50	250	750+250	400	80	400000	60000
17	5	15	50	250	750+250	400	80	400000	60000
18	5	15	50	250	750+250	400	60	400000	60000
19	5	15	50	250	750+250	400	60	400000	60000
20	5	15	50	250	750+250	400	70	400000	60000
21	7	30	30	210	900+210	400	50	444000	55500
22	7	30	30	210	900+210	400	55	444000	61050
23	7	30	30	210	900+210	400	80	444000	77700
24	7	30	30	210	900+210	400	70	444000	66600
25	7	30	30	210	900+210	400	40	444000	44400
26	7	30	50	350	1500+350	300	50	555000	92500
27	7	30	50	350	1500+350	300	30	555000	55500
28	7	30	50	350	1500+350	300	25	555000	46250
29	7	30	50	350	1500+350	300	50	555000	92500
30	7	30	50	350	1500+350	300	50	555000	92500
31	10	40	30	300	1200+300	200	58	300000	87000
32	10	40	30	300	1200+300	200	60	300000	90000
33	10	40	30	300	1200+300	200	40	300000	60000
34	10	40	30	300	1200+300	200	60	300000	90000
35	10	40	30	300	1200+300	200	50	300000	75000
36	10	50	30	300	1500+300	100	30	180000	54000
37	10	50	30	300	1500+300	100	35	180000	63000
38	10	50	30	300	1500+300	100	30	180000	54000
39	10	50	30	300	1500+300	100	45	180000	81000
40	10	50	30	300	1500+300	100	50	180000	90000

at termination, on average, are very large. For four large instances, i.e., instances 27-29 and 32, CPLEX even cannot solve the LP relaxation at the root node in 10 hours. We think that it is because CPLEX has focused on finding hidden integer feasible solutions, did not even leave the root node, and thereby did not report on finding a lower bound.

In summary, the node-arc model can only be applied to small instances. For medium and large ones, this method is not effective, and other strategies should be introduced. Hereafter, we refer to this solution approach as the pure CPLEX strategy.

The computational results in Table 2 show that the number of cycles generated range from 167 to 24803. As instance scale increases, the CG procedure generated more cycles, which makes instances more difficult. In most of the 40 instances, the computing time of the CG procedure is less than 1700 seconds.

Table 2 Results for the Node-arc Model

Instance	CPLEX				CG procedure		
	LB	UB_CPLEX	GAP(%)	CPUs	Obj	#Cycles	CPUs
1	69153.64	69160	0.00	6750.97	62560.00	167	0.41
2	83550.68	83558	0.00	9396.86	79558.00	864	1.08
3	93968.00	93968	0.00	344.64	88118.00	1185	2.11
4	248105.77	273548	9.30	36000	262048.00	970	1.30
5	405391.97	444460	8.79	36000	434682.22	2812	5.29
6	960419.00	1030299	6.78	36000	960419.00	7880	251.30
7	250860.00	311340	19.43	36000	250780.00	5142	247.29
8	1076797.50	1167580	7.78	36000	1076227.50	7180	125.52
9	1090728.00	1160318	6.00	36000	1084790.50	5250	171.66
10	903795.00	934210	3.26	36000	895060.00	5050	137.66
11	919711.67	-	-	36000	921408.08	18078	83.15
12	192979.56	361837	46.67	36000	193212.00	12890	88.34
13	845427.67	-	-	36000	851291.00	16231	71.34
14	778116.00	-	-	36000	785976.00	12755	69.90
15	835817.00	-	-	36000	837647.00	9900	77.89
16	2451351.70	-	-	36000	2452158.33	14504	366.96
17	2724653.67	-	-	36000	2724653.67	10317	554.76
18	611043.00	777461	21.41	36000	611043.00	7405	501.90
19	1184297.00	-	-	36000	1184177.00	9440	481.70
20	2389456.89	-	-	36000	2387279.11	9860	606.45
21	407605.20	-	-	36000	409411.00	18760	360.75
22	795741.84	-	-	36000	800089.67	18117	261.40
23	1755747.22	-	-	36000	1756380.56	22401	328.85
24	1693906.78	-	-	36000	1695083.17	24803	266.14
25	809308.52	894492	9.52	36000	811328.36	13399	229.85
26	1874768.86	-	-	36000	1876387.00	20454	1684.37
27	*	-	-	36000	937443.00	12141	1662.57
28	*	-	-	36000	948619.00	11439	1285.90
29	*	-	-	36000	2076188.00	20508	1319.64
30	1968603.81	-	-	36000	1968723.00	17804	1627.64
31	433974.00	-	-	36000	440577.50	11701	83.93
32	*	-	-	36000	983704.71	15594	102.63
33	211022.20	369290	42.86	36000	212690.00	11036	99.44
34	215665.48	-	-	36000	217298.67	13356	145.81
35	207786.58	-	-	36000	210488.86	13015	93.01
36	208677.36	-	-	36000	208782.00	13224	148.97
37	218894.41	-	-	36000	219483.33	9815	168.23
38	102133.50	-	-	36000	102372.50	9684	147.51
39	412535.35	-	-	36000	413600.90	13857	169.78
40	408887.09	-	-	36000	409764.33	19685	105.19

*: CPLEX could not solve the linear relaxation within 10 hours.
-: CPLEX could not return a feasible solution within 10 hours.

6.3. Results of Heuristic Strategy H1

We next turn our attention to the performance evaluation of the first heuristic strategy. We implemented heuristic strategy H1 with a time limit of one hour after the standard column generation. The total computation time of our heuristic strategies is one hour plus the time for the CG procedure. The computational results are reported in Table 3. For heuristic strategy H1, column 3 shows the objective value of the best-found integer solution (UB). Column 4 gives the relative objective gap between the best node value (LB) and the objective value (UB) of the incumbent solution at

termination. The last column (GAP$^+$ (%)) shows the relative objective gap between the integer solutions returned by CPLEX with the node-arc model and that returned by H1.

Table 3 Results of Heuristic Strategy H1

Instance	UB_CPLEX	H1		GAP$^+$(%)
		UB	GAP(%)	
1	69160	69160	0.00	0.00
2	83558	83558	0.00	0.00
3	93968	93968	2.77	0.00
4	273548	268548	2.42	-1.83
5	444460	439460	1.09	-1.12
6	1030299	980299	2.00	-4.85
7	311340	271340	7.58	-12.85
8	1167580	1157580	7.03	-0.86
9	1160318	1135318	4.40	-2.15
10	934210	909210	1.56	-2.68
11	*	961335	4.00	-
12	361837	286837	32.64	-20.73
13	*	935041	8.96	-
14	*	863346	8.95	-
15	*	891967	5.97	-
16	*	2485945	1.36	-
17	*	2844712	4.22	-
18	777461	697461	12.39	-10.29
19	*	1318217	10.17	-
20	*	2556418	6.62	-
21	*	599131	31.67	-
22	*	954823	16.21	-
23	*	1851825	5.15	-
24	*	1814229	6.57	-
25	894492	894492	9.30	0.00
26	*	*	*	*
27	*	1013909	7.54	-
28	*	1019887	6.99	-
29	*	*	*	*
30	*	2167123	9.15	-
31	*	632640	30.36	-
32	*	1097790	10.28	-
33	369290	359290	39.86	-2.71
34	*	380682	42.92	-
35	*	380346	44.06	-
36	*	260902	18.72	-
37	*	294440	24.11	-
38	*	211710	49.58	-
39	*	476289	12.54	-
40	*	464491	11.15	-

*: CPLEX could not return a feasible solution within 10 hours.

As one may observe from Table 3, the first heuristic strategy H1 reached a remarkable improvement of the solution quality. This strategy found feasible solutions for 38 of 40 instances, whereas the pure CPLEX strategy obtained solutions only for 14 instances with ten times computing effort. Now look at those 14 instances where the pure CPLEX strategy found feasible solutions. Strategy H1 produced better solutions to 10 of 14 instances and the solutions for some remaining four instances are at least as good as that provided by the pure CPLEX strategy. The computing time of strategy H1 is less than 1.5 hours for each instance, which is significantly less than that of the pure CPLEX solution strategy (see Table 2).

6.4. Results of Heuristic Strategy H2

We next examine effectiveness of the second heuristic strategy. As previously stated, its performance depends on how many cycles are fixed for each resource. In this computational study, we examined six settings of ϖ, i.e., 5, 10, 15, 20, 30, and 40. Table 4 summarizes the statistical results of H2, in comparison with the pure CPLEX solver. Columns 2-4 display the maximum, minimum and average GAP$^+$(%) for each instance group respectively. #Fre. denotes the number of instances

Table 4 Statistical Results of Heuristic Strategy H2

Instance group	$\varpi = 5$				$\varpi = 10$				$\varpi = 15$			
	$GAP^+(\%)$			#Fre.	$GAP^+(\%)$			#Fre.	$GAP^+(\%)$			#Fre.
	Min	Max	Avg.		Min	Max	Avg.		Min	Max	Avg.	
1-15	-41.46	5.98	-8.21	13	-41.46	0.00	-7.76	15	-40.07	0.00	-7.44	15
16-25	-20.58	-7.27	-13.92	10	-20.58	-6.71	-13.64	10	-20.58	-6.71	-13.64	10
26-40[a]	-31.14	-31.14	-31.14	15	-21.66	-21.66	-21.66	14	-25.73	-25.73	-25.73	15
Instance group	$\varpi = 20$				$\varpi = 30$				$\varpi = 40$			
	$GAP^+(\%)$			#Fre.	$GAP^+(\%)$			#Fre.	$GAP^+(\%)$			#Fre.
	Min	Max	Avg.		Min	Max	Avg.		Min	Max	Avg.	
1-15	-41.46	0.00	-7.49	15	-38.69	0.00	-7.30	15	-40.07	0.00	-7.35	15
16-25	-20.58	-6.71	-13.64	10	-19.94	-3.91	-11.92	10	-19.94	0.00	-9.97	10
26-40[a]	-28.43	-28.43	-28.43	14	-28.43	-28.43	-28.43	13	-23.02	-23.02	-23.02	12

[a]: The Min, Max and Avg. are all equal because the pure CPLEX strategy solved only one instance in group 26-40.

where feasible solutions are found. We report detailed computational results in Tables 11 and 12 in the Appendix.

Results in Tables 4, 11 and 12 show that with six different settings of ϖ, strategy H2 found feasible solutions to 38, 39, 40, 39, 38, and 37 instances, respectively. This significantly extends the solution ability of the pure CPLEX strategy. Further, we can observe that for small instances 1-15, it is better to fix more cycles for each resource. As instance size increases, it benefits from fixing fewer cycles for each resource.

Now we examine the performance difference among six settings of ϖ. Let us look at instances where the pure CPLEX strategy could find feasible solutions. We can see that H2 with $\varpi = 5$, on average, reached a better performance improvement (e.g. smaller objective values). If we focus on those 35 instances (see them in Tables 11 and 12) where H2 found feasible solutions under six settings, we can see that H2 with $\varpi = 5$ obtained better upper bounds. In contrast, the worst performance was obtained for $\varpi = 40$. If we compare six settings in pairs, we can draw the same conclusion that H2 found the best average objective values when ϖ was set to 5. In addition, increasing the number of fixed cycles decreases the frequency of finding feasible solutions. The reason is described as follows. These instances are very hard to solve. When we fix more cycles, the resulting restricted arc-cycle model has more variables. This leads to a harder MIP model. When the time limit of one hour reached, the MIP solver still could not find a feasible solution. However, this does not mean that the restricted model is infeasible.

In summary, $\varpi = 5$ is a better one among six settings in strategy H2, if we aim to improve the upper bounds and frequency of finding feasible solutions.

6.5. Results of Heuristic Strategy H3

We next evaluate the performance of the third heuristic strategy. In this comparison study, we examined six different settings of parameter η: 1/10, 1/5, 1/4, 1/3, 1/2 and 100%. Table 5 summarizes some statistical results of H3, in comparison with the pure CPLEX strategy. We report detailed computational results in Tables 13 and 14 in the Appendix.

Table 5 Statistical Results of Heuristic Strategy H3

Instance group	$\eta = 1/10$				$\eta = 1/5$				$\eta = 1/4$			
	$GAP^+(\%)$			#Fre.	$GAP^+(\%)$			#Fre.	$GAP^+(\%)$			#Fre.
	Min	Max	Avg.		Min	Max	Avg.		Min	Max	Avg.	
1-15	-40.07	0.00	-6.87	13	-34.55	0.00	-6.75	14	-37.31	0.00	-6.79	14
16-25	-16.08	-16.08	-16.08	7	-5.59	-5.59	-5.59	8	-3.35	-3.35	-3.35	6
26-40[a]	-16.25	-16.25	-16.25	13	-13.54	-13.54	-13.54	12	-17.60	-17.60	-17.60	10
Instance group	$\eta = 1/3$				$\eta = 1/2$				$\eta = 100\%$			
	$GAP^+(\%)$			#Fre.	$GAP^+(\%)$			#Fre.	$GAP^+(\%)$			#Fre.
	Min	Max	Avg.		Min	Max	Avg.		Min	Max	Avg.	
1-15	-34.55	0.00	-6.34	13	-35.93	1.83	-6.97	11	-9.67	7.23	-0.48	6
16-25	-	-	-	7	-	-	-	2	-	-	-	0
26-40[a]	-24.37	-24.37	-24.37	11	-24.37	-24.37	-24.37	15	-24.37	-24.37	-24.37	14

[a]: The Min, Max and Avg. are all equal because the pure CPLEX strategy solved only one instance in group 26-40.

Results show that under six settings, strategy H3 found feasible solutions for 33, 34, 30, 31, 28 and 20 of the 40 instances, respectively. In comparison with the pure CPLEX execution, H3 significantly improved the frequency of finding feasible solutions. Parameter η affects the performance of H3. If we focus on the success frequency, $\eta = 1/10$ and $1/5$ are better settings. Under these two settings, H3 got feasible solutions for 33 and 34 out of the 40 instances, respectively. We further look at 29 instances (see them in Tables 13 and 14) for which these two values of η both found feasible solutions. On average, H3 with $\eta = 1/10$ produced solutions that were 0.23% better than what H3 with $\eta = 1/5$ could produce. Fixing all the single-paths is not a good strategy. For example, it produced feasible solutions for only 20 instances.

We next analyze the quality of solutions produced under the first four settings, which produced high successful ratio of finding feasible solutions. In small instances 1-15, the performance of H3 deteriorates when a larger η is applied. We can explained it as follows. These small instances include fewer network arcs and commodities. If we fix single-paths for more commodities, the search space shrinks and thus the probability of finding feasible solutions decreases. For small instances, the best choice is to set η to $1/5$. As one may observe, the performance of H3 varies with the number of commodities in each instance. If one instance has more commodities, it is better to not fix too much single-paths. For example, consider medium instances 16-25 where there are 400 commodities. For these instances, a higher fixing ratio may restrict the path choice for those commodities, which do not follow single-paths in the L-RMP solution. Our results also show that for most of these ten instances, the resulting models are infeasible if paths of over 50% commodities are fixed. For medium instances, it is better to fix paths for 20% commodities.

We finally examine large instances 26-40. In comparison with the last instance group, these instances include fewer commodities, but more network nodes and arcs. From the computational results, we can observe that a higher fixing ratio does not significantly affect the frequency of finding feasible solutions. Conversely, too small η will produce larger MIP models, which thus cannot be solved within one hour.

In summary, $\eta = 1/5$ on average is the best setting in strategy H3, if we want to improve the upper bounds and frequency of finding feasible solutions.

6.6. Results of Heuristic Strategy H4

We finally evaluate the performance of heuristic strategy H4. We examined its performance under nine combinations of ϖ and η. The computational results are reported in Tables 15-17 in the Appendix. Table 6 gives the frequency of finding feasible solutions. From these results, we can draw the following conclusions:

(1) Under these nine settings, heuristic strategy H4 found feasible solutions for 29, 30, 29, 20, 23, 24, 6, 12, and 11, instances respectively.

(2) This strategy also significantly shrank the search space, while decreasing the model scale. As a result, the frequency of finding feasible solutions declined, in comparison with previous three fixing strategies (H1, H2, and H3). Further, in this strategy, fixing single-paths has a greater impact on successful frequency than fixing cycles. One may observe that after fixing 50% paths, the resulting models under each setting of ϖ, become infeasible for most of the instances. But in general, $\eta = \frac{1}{10}$ is a better setting.

(3) The performance varies across instance groups. For small instances 1-15, the better frequencies of finding feasible solutions were reached for the first three parameter settings. For 11 instances, heuristic H4 with each of these three settings all got feasible solutions. In particular, heuristic H4 with $\eta = \frac{1}{10}$ and $\varpi = 20$, on average, found solutions with better objective function values. Now we focus on medium instances 16-25 that includes 400 commodities. Heuristic H4 failed in most of these 10 instances. For example, we found that in our experiments, the restricted models are infeasible for most of these 10 instances. Now look at large instances 26-40, which have fewer commodities than instances 16-25. With the first three parameter settings, heuristic H4 found feasible solutions for most instances in this group. In particular, heuristic H4 with the first parameter setting reached the best performance: higher frequency of finding feasible solutions and higher solution quality (e.g., better upper bounds).

In summary, we would recommend parameter setting $\eta = \frac{1}{10}$ and $\varpi = 20$, when heuristic strategy H4 is applied.

6.7. Comparison of Four Heuristic Strategies

We have demonstrated that our proposed four heuristic strategies significantly outperformed the pure CPLEX execution for the considered test instances. It produced better solutions within shorter computing times. In this section, we compare four heuristic strategies. In this comparison study, we chose the best parameter setting for each heuristic strategy, which is determined in the previous sections. In Table 7, we summarize the comparison results across the instance groups. Detailed

Table 6 Frequency of Finding Feasible Solutions of Heuristic Strategy H4

Parameter setting	Instance group		
	1-15	16-25	26-40
$\eta = \frac{1}{10}, \varpi = 10$	14	0	15
$\eta = \frac{1}{10}, \varpi = 20$	13	3	14
$\eta = \frac{1}{10}, \varpi = 30$	14	2	13
$\eta = \frac{1}{5}, \varpi = 10$	11	0	9
$\eta = \frac{1}{5}, \varpi = 20$	12	1	10
$\eta = \frac{1}{5}, \varpi = 30$	13	0	11
$\eta = \frac{1}{2}, \varpi = 10$	2	0	4
$\eta = \frac{1}{2}, \varpi = 20$	6	0	6
$\eta = \frac{1}{2}, \varpi = 30$	3	0	8

Table 7 Comparing Four Heuristic Strategies

Instance group	H2 ($\varpi = 5$)				H3 ($\eta = \frac{1}{5}$)				H4 ($\eta = \frac{1}{10}, \varpi = 20$)			
	Avg. UB	Avg. GAP$^+$(%)	#Fre.	#Beat	Avg. UB	Avg. GAP$^+$(%)	#Fre.	#Beat	Avg. UB	Avg. GAP$^+$(%)	#Fre.	#Beat
1-15	600191.46	-4.11	13	12	619107.64	-2.69	14	13	541898.62	-3.63	13	13
16-25	1496725.30	-9.23	10	10	1526046.75	-7.12	8	7	1242312.67	-7.73	3	3
26-40	775811.20	-17.27	15	15	521031.33	-6.51	12	12	693607.43	-14.86	14	14

results are presented in Table 18 in the Appendix. The statistical measures here are calculated based on the results of heuristic strategy H1. #Beat represents the number of instances where one heuristic strategy produced an equivalent or better solutions, in comparison with heuristic H1.

The computational results show that H2, H3, and H4 significantly outperformed H1. In particular, H2, H3 and H4 respectively produced equivalent or better solutions for 38, 29, and 29 instances. In comparison with the other three strategies, H1 only found better solutions for 1, 8, and 10 instances.

Table 8 Comparing H4 with H2 and H3

Instance group	No. of H4 beating H2	No. of H4 beating H3
1-15	2	7
16-25	1	3
26-40	2	12

We now compare the performance of H2 (fixing cycles) and H3 (fixing single-paths). The computational results show that H2, on average, outperformed H3. As one may observe, H2 reached a greater frequency of finding feasible solutions. H2 and H3 found feasible solutions for the same 32 instances. For these 32 instances, H2 produced solutions that are on average 3.78% better than what H3 could produce. Such a conclusion is stable across different instance scales.

We next examine the benefits of simultaneously fixing cycles and paths. Table 8 shows the number of instances where H4 produced better solutions. On the one hand, the computational results in Table 18 show that simultaneously fixing cycles and paths decreased the probability of finding feasible solutions in one hour, while shrinking the solution space. On the other hand, the results in Table 8 obviously demonstrate that H4 did improve the solution quality in some instances, if we

focus on those instances where three heuristic strategies could find feasible solutions. In particular, such benefits are more significant when comparing H4 and H3.

To further show the performance difference, we carried out a Wilcoxon signed ranks test (Wilcoxon 1945). The objective value is chosen as the measurement. The used data covers results for 26 instances where four heuristic strategies all found feasible solutions. The null hypothesis is that the objective value difference between two algorithms is zero. The statistical test results are given in Table 9. The asymptotical significances are all smaller than 0.05.

Following the above analysis, we recommend H2 as the best heuristic strategy.

6.8. Impact of Single-path Constraints

In the SPSNDRC, one main feature is introducing single-path constraints for commodity transportation, a main difference from previous studies. In this subsection, we analyze how this requirement would influence the computational difficulty and capacity utilization of resources and arcs.

We used the node-arc formulation to execute the computational tests. Define continuous variables $f_{i,j}^k$, which represent the flows of commodity k on arc (i,j). Changing constraints (1)-(3) and (11) to the following constraints (31)-(34), we could get a new node-arc formulation without the single-path requirements.

$$\min W = \sum_{v \in V} F_v \delta_v + \sum_{k \in K} \sum_{(i,j) \in A} c_{i,j}^k f_{i,j}^k \tag{31}$$

$$\sum_{(i,j) \in A} f_{i,j}^k - \sum_{(j,i) \in A} f_{j,i}^k = \begin{cases} q^k, & \text{if } i = o(k) \\ -q^k, & \text{if } i = d(k) \\ 0, & \text{otherwise} \end{cases} \quad \forall i \in N, \forall k \in K, \tag{32}$$

$$\sum_{k \in K} f_{i,j}^k \le u_{i,j} \sum_{v \in V} y_{i,j}^v, \quad \forall (i,j) \in S, \tag{33}$$

$$0 \le f_{i,j}^k \le q^k, \quad \forall (i,j) \in A, \forall k \in K. \tag{34}$$

We also solved this new formulation for all the 40 instances with a larger time limit of 50 hours. The computational results are reported in Table 10. Our results show that under these two settings, feasible solutions are returned for only ten out of 40 instances. That is, the small instances 1-10 are demonstrated integer feasible. Thus, we include in Table 10 results for these ten instances. Here UB and LB represent the best integer solution and the global lower bound at CPLEX's termination,

Table 9 Results of Wilcoxon Signed Ranks Test

Statistics	H2-H1	H3-H1	H4-H1	H3-H2	H4-H2	H4-H3
Z	-4.320[a]	-3.105[a]	-4.016[a]	-4.325[b]	-2.93[b]	-3.835[a]
Asymp.Sig.(2-tailed)	0.000	0.002	0.000	0.000	0.003	0.000

a. Based on positive ranks.

b. Based on negative ranks.

respectively. Columns 4 and 9 present the computing times to prove optimality, or the remaining gap after 50 hours. Here we report the computing time in hours. The capacity utilizations for services arcs are shown in columns 5 and 10, respectively. For the single-path case, the capacity utilization for arc (i, j) is calculated as $\sum_{k \in K} q^k \overline{x}^k_{i,j} / u_{i,j}$. For the other formulation, it is calculated as $\sum_{k \in K} \overline{f}^k_{i,j} / u_{i,j}$. In addition, columns 6 and 11 respectively present the number of resources used under the two settings.

Table 10 Impact of Single-path Constraints

Instance	Model with single-path constraints					Model without single-path constraints				
	UB	LB	Gap(%)(CPUs)	Arc utilization (%)	#Resource	UB	LB	Gap(%)(CPUs)	Arc utilization (%)	#Resource
1	69160	69160	0(1.88h)	71.05	7	64160	64160	0(3.71h)	85.41	6
2	83558	83558	0(2.61h)	69.05	5	83558	83558	0(18.66h)	77.84	5
3	93968	93968	0(0.10h)	66.19	4	93968	90458.22	3.88(50h)	72.44	4
4	268548	265232.59	1.25(50h)	87.66	21	263548	257195.28	2.47(50h)	90.81	20
5	444460	431556.46	2.99(50h)	89.16	29	439460	430168.36	2.16(50h)	89.65	28
6	1010299	1010299	0(49.26h)	70.28	20	1000299	978861.92	2.19(50h)	97.75	18
7	356340	320276.83	11.26(50h)	75.48	24	341340	295968.09	15.33(50h)	88.49	21
8	1072580	1036509.47	3.48(50h)	80.17	10	1062580	1012077.34	4.99(50h)	92.24	8
9	1245318	1245318	0(48.41h)	82.74	45	1215318	1215318	0(49.30h)	91.87	39
10	954210	954210	0(47.26h)	78.66	24	944210	944210	0(49.41h)	85.28	22

In regard of the computational difficulty, all the instances got larger gaps or CPU time, when we eliminated the single-path constraints. This phenomenon was explained when we explored the mechanism of CPLEX. We found that CPLEX eliminated much more invalid rows for the node-arc formulation with single-path constraints in the preprocessing process, thus largely shrinking the model scale. Although removing single-path constraints can increase feasible regions, models with more valid rows turned out to be more difficult to solve. Besides, results show that the arc utilizations are increased for all these ten instances, when we removed the single-path constraints. Consequently, enforcing single-path constraints produced larger utilizations of resources, as shown in column 4.

Note that the above observation is concluded from these specific ten instances, where CPLEX could get feasible solutions within 50 hours. We cannot generalize this conclusion for the other larger instances.

7. Conclusions and Future Work

We extended existing service network design models with resource constraints, and considered both resource-availability constraints and single-path requirements for routing commodities. In this paper, we study the single-path service network design with resource constraints in the context of the time-space network, which leads to a large-scale and computationally difficult model. We first presented a node-arc integer programming formulation for the SPSNDRC. With this formulation, only the small problem instances can be solved by using MIP solvers. In order to deal with larger instances, we next proposed an arc-cycle formulation, from which the column-generation-based

approach can be applied to solve the model efficiently as it incorporates a set of service cycles therein. In particular, due to the cycle-based nature of this model, there can be a huge number of variables for instances of even medium size. To produce good solutions, we presented a two-stage MIP-based heuristic, which is based on the second formulation. In the first stage, the column generation procedure is executed to enumerate a subset of cycles for given resources. In the second stage, feasible integer solutions are created from the optimal solution of the linear relaxation of the restricted master problem. We introduced four heuristic strategies to produce feasible solutions: variables conversion, fixing cycles, fixing single-paths, and simultaneously fixing cycles and paths. The underlying idea is to first define a small MIP model for the SPSNDRC, based on the optimal solution of the L-RMP, and then solve it using MIP solvers. We demonstrated the effectiveness of the proposed approach by comparing the quality of the solutions it produced with those produced by CPLEX with a time limit of 10 hours. The experimental results show that our four heuristic strategies could find better-quality solutions than the pure CPLEX strategy. More importantly, our approach took much less time, e.g., less than 1.5 hours for each instance. Further, we found that the other three heuristic strategies, on average, outperformed the first strategy. The strategy of fixing cycles obtained the best performance, in terms of the frequency of finding feasible solutions, and the solution quality. The results also show that the approach did benefit from simultaneously fixing cycles and paths, although this strategy declined the number of instances where we can produce feasible solutions.

In the future work, we will introduce heterogeneous resource constraints in this problem. In the aspect of solution approach, we will try to develop a branch-and-price approach and demonstrate its performance.

Acknowledgments

This research has been partially supported by the Natural Science Foundation of China (Grant no. 71672126, 71532015), and the "Shuguang Program" supported by Shanghai Education Development Foundation and Shanghai Municipal Education Commission (Grant no. 18SG23). The work of Kai Pan was supported in part by the Research Grants Council of Hong Kong (Grant PolyU 155077/18B).

References

Ahuja, R. K., T. L. Magnanti, J.B. Orlin. 1993. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Englewood Cliffs, NJ.

Andersen, J., M. Christiansen, T.G. Crainic, R. Grϕnhaug. 2011. Branch and price for service network design with asset management constraints. *Transportation Science* **54** 33–49.

Andersen, J., T.G. Crainic, M. Christiansen. 2009. Service network design with asset management: Formulations and comparative analysis. *Transportation Research Part C* **17** 197–207.

Barnhart, C., C.A. Hane, P.H. Vance. 2000. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Operations Research* **48** 318–326.

Barnhart, C., R. Schneur. 1996. Air network design for express shipment service. *Operations Research* **46** 852–863.

Bortolini, M., M. Faccio, M. Gamberi, F. Pilati. 2016. Multi-objective design of multi-modal fresh food distribution networks. *International Journal of Logistics Systems and Management* **24** 155–177.

Chen, Cheng-Chieh (Frank), Paul Schonfeld. 2016. A dispatching decision support system for countering delay propagation in intermodal logistics networks. *Transportation Planning and Technology* **39** 254–268.

Chouman, M., T.G. Crainic. 2015. Cutting-plane matheuristic for service network design with design-balance requirements. *Transportation Science* **49** 99–113.

Crainic, T.G. 2000. Service network design in freight transportation. *European Journal of Operational Research* **122** 272–288.

Crainic, T.G., M. Hewitt, M.Toulouse, D.M. Vu. 2016. Service network design with resource constraints. *Transportation Science* **50** 1380–1393.

Crainic, T.G., M. Hewitt, M. Toulouse, D.M. Vu. 2018. Scheduled service network design with resource acquisition and management. *EURO Journal on Transportation and Logistics* **7** 277–309.

Gundegjerde, C., I.B. Halvorsen, E.E. Halvorsen-Weare, L.M. Hvattum, L.M. Nonas. 2015. A stochastic fleet size and mix model for maintenance operations at offshore wind farms. *Transportation Research Part C: Emerging Technologies* **52** 74 – 92.

Hewitt, M., L.N. Nemhauser, M.W.P. Savelsbergh. 2010. Combining exact and heuristic approaches for the capacitated fixed-charge network flow problem. *INFORMS Journal on Computing* **22** 314–325.

Kim, D., C. Barnhart, G. Ware, G. Reinhardt. 1999. Multimodel express package delivery: A service network design application. *Transportation Science* **33** 391–407.

Laaziz, E.H., N. Sbihi. 2019. A service network design model for an intermodal rail-road freight forwarder. *International Journal of Logistics Systems and Management* **32** 465–482.

Lai, M.F., H.K. Lo. 2004. Ferry service network design: Optimal fleet size, routing and scheduling. *Transportation Research Patr A* **38** 305–328.

Li, X., Y.P. Aneja, J. Huo. 2012. Using branch-and-price approach to solve the directed network design problem with relays. *Omega* **40** 672 – 679.

Li, X., S. Lin, P. Tian, Y.P. Aneja. 2017a. Models and column generation approach for the resource-constrained minimum cost path problem with relays. *Omega* **66** 79 – 90.

Li, X., K. Wei, Y.P. Aneja, P. Tian. 2017b. Design-balanced capacitated multicommodity network design with heterogeneous assets. *Omega* **67** 145 – 159.

Lo, H. K., K. An, W. H. Lin. 2013. Ferry service network design under demand uncertainty. *Transportation Research Part E: Logistics and Transportation Review* **59** 48 – 70.

Magnanti, T.L., R.T. Wong. 1984. Network design and transportation planning: models and algorithms. *Transportation Science* **18** 1–55.

Pedersen, M.B., T.G. Crainic, O.B.G. Madsen. 2009. Models and tabu search metaheuristic for service network design with asset balance requirements. *Transportation Science* **43** 158–177.

Schrotenboer, Albert H., Evrim Ursavas, Iris F.A. Vis. 2020. Mixed integer programming models for planning maintenance at offshore wind farms under uncertainty. *Transportation Research Part C: Emerging Technologies* **112** 180 – 202.

Verter, V., B.Y. Kara. 2008. A path-based approach for hazmat transport network design. *Management Science* **54** 29–40.

Vu, D. M., T. G. Crainic, M. Toulouse. 2013. A three-phase matheuristic for capacitated multi-commodity fixed-cost network design with design-balance constraints. *Journal of Heuristics* **19** 757–795.

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

Wilcoxon, F. 1945. Individual comparisons by ranking methods. *Biometrics Bulletin* **1** 80–83.

Zhu, E., T.G. Crainic, M. Gendreau. 2014. Scheduled service network design for freight rail transportation. *Operations Research* **62** 383–400.

Appendix. Detailed Results

Table 11 Results of Heuristic Strategy H2

Instance	UB_CPLEX	$\varpi = 5$		$\varpi = 10$		$\varpi = 15$	
		UB	$GAP^+(\%)$	UB	$GAP^+(\%)$	UB	$GAP^+(\%)$
1	69160	+	+	69160	0.00	69160	0.00
2	83558	88558	5.98	83558	0.00	83558	0.00
3	93968	93968	0.00	93968	0.00	93968	0.00
4	273548	263548	-3.66	263548	-3.66	268548	-1.83
5	444460	439460	-1.12	439460	-1.12	439460	-1.12
6	1030299	975299	-5.34	970299	-5.82	975299	-5.34
7	311340	256340	-17.67	256340	-17.67	256340	-17.67
8	1167580	1087580	-6.85	1087580	-6.85	1087580	-6.85
9	1160318	+	+	1095318	-5.60	1100318	-5.17
10	934210	899210	-3.75	904210	-3.21	899210	-3.75
11	*	941335	-	936335	-	941335	-
12	361837	211837	-41.46	211837	-41.46	216837	-40.07
13	*	880041	-	865041	-	865041	-
14	*	808346	-	808346	-	808346	-
15	*	856967	-	861967	-	866967	-
16	*	2460945	-	2455945	-	2455945	-
17	*	2729712	-	2734712	-	2734712	-
18	777461	617461	-20.58	617461	-20.58	617461	-20.58
19	*	1188217	-	1193217	-	1193217	-
20	*	2396418	-	2401418	-	2396418	-
21	*	424131	-	424131	-	434131	-
22	*	819823	-	814823	-	819823	-
23	*	1786825	-	1771825	-	1786825	-
24	*	1714229	-	1704229	-	1704229	-
25	894492	829492	-7.27	834492	-6.71	834492	-6.71
26	*	1896005	-	1926005	-	1896005	-
27	*	953909	-	948909	-	953909	-
28	*	959887	-	964887	-	964887	-
29	*	2096664	-	+	+	2111664	-
30	*	1982123	-	1982123	-	1982123	-
31	*	482640	-	477640	-	477640	-
32	*	1017790	-	1022790	-	1017790	-
33	369290	254290	-31.14	289290	-21.66	274290	-25.73
34	*	260682	-	285682	-	255682	-
35	*	260346	-	280346	-	270346	-
36	*	230902	-	230902	-	230902	-
37	*	244440	-	244440	-	239440	-
38	*	136710	-	141710	-	136710	-
39	*	436289	-	431289	-	431289	-
40	*	424491	-	429491	-	424491	-

*: CPLEX could not return a feasible solution within 10 hours. +: No feasible solution is found by strategy H2.

Table 12 Results of Heuristic Strategy H2 (Cont'd)

Instance	UB_CPLEX	$\varpi = 20$		$\varpi = 30$		$\varpi = 40$	
		UB	$GAP^+(\%)$	UB	$GAP^+(\%)$	UB	$GAP^+(\%)$
1	69160	69160	0.00	69160	0.00	69160	0.00
2	83558	83558	0.00	83558	0.00	83558	0.00
3	93968	93968	0.00	93968	0.00	93968	0.00
4	273548	268548	-1.83	268548	-1.83	268548	-1.83
5	444460	444460	0.00	439460	-1.12	439460	-1.12
6	1030299	970299	-5.82	975299	-5.34	975299	-5.34
7	311340	256340	-17.67	256340	-17.67	256340	-17.67
8	1167580	1087580	-6.85	1087580	-6.85	1087580	-6.85
9	1160318	1095318	-5.60	1095318	-5.60	1105318	-4.74
10	934210	904210	-3.21	904210	-3.21	904210	-3.21
11	*	936335	-	946335	-	946335	-
12	361837	211837	-41.46	221837	-38.69	216837	-40.07
13	*	870041	-	880041	-	880041	-
14	*	813346	-	808346	-	818346	-
15	*	861967	-	856967	-	871967	-
16	*	2455945	-	2465945	-	2540945	-
17	*	2734712	-	2734712	-	2779712	-
18	777461	617461	-20.58	622461	-19.94	622461	-19.94
19	*	1193217	-	1188217	-	1193217	-
20	*	2396418	-	2401418	-	2431418	-
21	*	424131	-	439131	-	464131	-
22	*	814823	-	834823	-	859823	-
23	*	1771825	-	1806825	-	1811825	-
24	*	1709229	-	1729229	-	1729229	-
25	894492	834492	-6.71	859492	-3.91	894492	0.00
26	*	1906005	-	+	+	+	+
27	*	948909	-	958909	-	958909	-
28	*	964887	-	974887	-	+	+
29	*	+	+	+	+	+	+
30	*	1977123	-	1987123	-	1987123	-
31	*	477640	-	502640	-	497640	-
32	*	1032790	-	1097790	-	1097790	-
33	369290	264290	-28.43	264290	-28.43	284290	-23.02
34	*	270682	-	265682	-	290682	-
35	*	280346	-	280346	-	270346	-
36	*	230902	-	240902	-	245902	-
37	*	249440	-	254440	-	259440	-
38	*	141710	-	146710	-	146710	-
39	*	431289	-	446289	-	451289	-
40	*	429491	-	444491	-	434491	-

*: CPLEX could not return a feasible solution within 10 hours. +: No feasible solution is found by strategy H2.

Table 13 Results of Heuristic Strategy H3

Instance	UB_CPLEX	$\eta = \frac{1}{10}$		$\eta = \frac{1}{5}$		$\eta = \frac{1}{4}$	
		UB	$GAP^+(\%)$	UB	$GAP^+(\%)$	UB	$GAP^+(\%)$
1	69160	69160	0.00	69160	0.00	69160	0.00
2	83558	83558	0.00	83558	0.00	83558	0.00
3	93968	93968	0.00	93968	0.00	93968	0.00
4	273548	263548	-3.66	268548	-1.83	268548	-1.83
5	444460	434460	-2.25	+	+	+	+
6	1030299	+	+	980299	-4.85	970299	-5.82
7	311340	271340	-12.85	266340	-14.45	281340	-9.64
8	1167580	1122580	-3.85	1102580	-5.57	1097580	-6.00
9	1160318	1115318	-3.88	1100318	-5.17	1100318	-5.17
10	934210	914210	-2.14	924210	-1.07	914210	-2.14
11	*	961335	-	951335	-	951335	-
12	361837	216837	-40.07	236837	-34.55	226837	-37.31
13	*	895041	-	890041	-	885041	-
14	*	+	+	828346	-	833346	-
15	*	891967	-	871967	-	861967	-
16	*	2460945	-	2470945	-	2505945	-
17	*	+	+	2859712	-	+	+
18	777461	652461	-16.08	+	+	+	+
19	*	1268217	-	1198217	-	+	+
20	*	+	+	+	+	+	+
21	*	429131	-	469131	-	484131	-
22	*	824823	-	824823	-	829823	-
23	*	1791825	-	1816825	-	1781825	-
24	*	1764229	-	1724229	-	1709229	-
25	894492	+	+	844492	-5.59	864492	-3.35
26	*	1991005	-	+	+	+	+
27	*	998909	-	1038909	-	+	+
28	*	974887	-	1014887	-	+	+
29	*	+	+	+	+	+	+
30	*	2002123	-	+	+	+	+
31	*	512640	-	497640	-	497640	-
32	*	+	+	1097790	-	1097790	-
33	369290	309290	-16.25	319290	-13.54	304290	-17.60
34	*	290682	-	300682	-	280682	-
35	*	295346	-	290346	-	310346	-
36	*	265902	-	265902	-	270902	-
37	*	289440	-	299440	-	299440	-
38	*	236710	-	206710	-	216710	-
39	*	466289	-	456289	-	451289	-
40	*	459491	-	464491	-	464491	-

*: CPLEX could not return a feasible solution within 10 hours. +: No feasible solution is found by strategy H3.

Table 14 Results of Heuristic Strategy H3 (Cont'd)

Instance	UB_CPLEX	$\eta = \frac{1}{3}$		$\eta = \frac{1}{2}$		$\eta = 100\%$	
		UB	$GAP^+(\%)$	UB	$GAP^+(\%)$	UB	$GAP^+(\%)$
1	69160	69160	0.00	69160	0.00	74160	7.23
2	83558	83558	0.00	+	+	+	+
3	93968	93968	0.00	93968	0.00	93968	0.00
4	273548	273548	0.00	278548	1.83	+	+
5	444460	+	+	+	+	+	+
6	1030299	980299	-4.85	+	+	+	+
7	311340	281340	-9.64	291340	-6.42	+	+
8	1167580	1092580	-6.42	1102580	-5.57	+	+
9	1160318	1105318	-4.74	+	+	+	+
10	934210	904210	-3.21	909210	-2.68	939210	0.54
11	*	951335	-	921335	-	951335	-
12	361837	236837	-34.55	231837	-35.93	326837	-9.67
13	*	+	+	855041	-	+	+
14	*	818346	-	803346	-	833346	-
15	*	876967	-	861967	-	+	+
16	*	2510945	-	+	+	+	+
17	*	2849712	-	+	+	+	+
18	777461	+	+	+	+	+	+
19	*	+	+	+	+	+	+
20	*	2396418	-	+	+	+	+
21	*	539131	-	+	+	+	+
22	*	849823	-	+	+	+	+
23	*	1811825	-	1826825	-	+	+
24	*	1724229	-	1774229	-	+	+
25	894492	+	+	+	+	+	+
26	*	2021005	-	1996005	-	1906005	-
27	*	+	+	1028909	-	1018909	-
28	*	1009887	-	1019887	-	+	+
29	*	+	+	2156664	-	2186664	-
30	*	+	+	2117123	-	2052123	-
31	*	497640	-	507640	-	502640	-
32	*	+	+	1097790	-	1072790	-
33	369290	279290	-24.37	279290	-24.37	279290	-24.37
34	*	275682	-	270682	-	285682	-
35	*	280346	-	280346	-	295346	-
36	*	270902	-	270902	-	275902	-
37	*	264440	-	309440	-	279440	-
38	*	231710	-	216710	-	171710	-
39	*	471289	-	461289	-	456289	-
40	*	499491	-	464491	-	464491	-

*: CPLEX could not return a feasible solution within 10 hours. +: No feasible solution is found by strategy H3.

Table 15 Results of Heuristic Strategy H4

Instance	UB_CPLEX	$\eta=\frac{1}{10},\varpi=10$		$\eta=\frac{1}{10},\varpi=20$		$\eta=\frac{1}{10},\varpi=30$	
		UB	GAP$^+$(%)	UB	GAP$^+$(%)	UB	GAP$^+$(%)
1	69160	74160	7.23	69160	0.00	69160	0.00
2	83558	88558	5.98	83558	0.00	83558	0.00
3	93968	93968	0.00	93968	0.00	93968	0.00
4	273548	268548	-1.83	268548	-1.83	268548	-1.83
5	444460	444460	0.00	444460	0.00	444460	0.00
6	1030299	965299	-6.31	980299	-4.85	+	+
7	311340	+	+	266340	-14.45	261340	-16.06
8	1167580	1087580	-6.85	1087580	-6.85	1087580	-6.85
9	1160318	1090318	-6.03	+	+	1100318	-5.17
10	934210	904210	-3.21	904210	-3.21	909210	-2.68
11	*	941335	-	941335	-	941335	-
12	361837	221837	-38.69	211837	-41.46	226837	-37.31
13	*	875041	-	880041	-	875041	-
14	*	803346	-	813346	-	818346	-
15	*	856967	-	+	+	861967	-
16	*	+	+	+	+	+	+
17	*	+	+	+	+	+	+
18	777461	+	+	+	+	+	+
19	*	+	+	1188217	-	+	+
20	*	+	+	+	+	+	+
21	*	+	+	+	+	+	+
22	*	+	+	+	+	+	+
23	*	+	+	+	+	1796825	-
24	*	+	+	1714229	-	1714229	-
25	894492	+	+	824492	-7.83	+	+
26	*	1916005	-	1901005	-	+	+
27	*	948909	-	948909	-	958909	-
28	*	974887	-	959887	-	964887	-
29	*	2091664	-	+	+	+	+
30	*	1987123	-	1977123	-	1982123	-
31	*	497640	-	497640	-	517640	-
32	*	1022790	-	1097790	-	1097790	-
33	369290	264290	-28.43	274290	-25.73	264290	-28.43
34	*	265682	-	260682	-	260682	-
35	*	260346	-	270346	-	260346	-
36	*	230902	-	240902	-	245902	-
37	*	254440	-	259440	-	264440	-
38	*	141710	-	136710	-	146710	-
39	*	436289	-	446289	-	446289	-
40	*	439491	-	439491	-	439491	-

*: CPLEX could not return a feasible solution within 10 hours. +: No feasible solution is found by strategy H4.

Table 16 Results of Heuristic Strategy H4 (Cont'd)

Instance	UB_CPLEX	$\eta=\frac{1}{5},\varpi=10$		$\eta=\frac{1}{5},\varpi=20$		$\eta=\frac{1}{5},\varpi=30$	
		UB	GAP$^+$(%)	UB	GAP$^+$(%)	UB	GAP$^+$(%)
1	69160	74160	7.23	69160	0.00	69160	0.00
2	83558	88558	5.98	88558	5.98	88558	5.98
3	93968	93968	0.00	93968	0.00	93968	0.00
4	273548	268548	-1.83	268548	-1.83	268548	-1.83
5	444460	444460	0.00	444460	0.00	444460	0.00
6	1030299	+	+	+	+	+	+
7	311340	+	+	+	+	271340	-12.85
8	1167580	1087580	-6.85	1087580	-6.85	1087580	-6.85
9	1160318	1100318	-5.17	1095318	-5.60	1095318	-5.60
10	934210	+	+	904210	-3.21	904210	-3.21
11	*	936335	-	941335	-	946335	-
12	361837	221837	-38.69	231837	-35.93	+	+
13	*	870041	-	875041	-	880041	-
14	*	803346	-	+	+	803346	-
15	*	+	+	856967	-	856967	-
16	*	+	+	+	+	+	+
17	*	+	+	+	+	+	+
18	777461	+	+	+	+	+	+
19	*	+	+	+	+	+	+
20	*	+	+	+	+	+	+
21	*	+	+	+	+	+	+
22	*	+	+	+	+	+	+
23	*	+	+	+	+	+	+
24	*	+	+	1714229	-	+	+
25	894492	+	+	+	+	+	+
26	*	1891005	-	1916005	-	+	+
27	*	+	+	+	+	+	+
28	*	+	+	+	+	+	+
29	*	2096664	-	2106664	-	2106664	-
30	*	+	+	+	+	+	+
31	*	+	+	487640	-	492640	-
32	*	1032790	-	1042790	-	1057790	-
33	369290	+	+	+	+	264290	-28.43
34	*	+	+	260682	-	260682	-
35	*	255346	-	+	+	275346	-
36	*	230902	-	235902	-	235902	-
37	*	254440	-	264440	-	254440	-
38	*	136710	-	141710	-	146710	-
39	*	441289	-	446289	-	441289	-
40	*	439491	-	439491	-	444491	-

*: CPLEX could not return a feasible solution within 10 hours. +: No feasible solution is found by strategy H4.

Table 17 Results of Heuristic Strategy H4 (Cont'd)

Instance	UB_CPLEX	$\eta=\frac{1}{2},\varpi=10$		$\eta=\frac{1}{2},\varpi=20$		$\eta=\frac{1}{2},\varpi=30$	
		UB	GAP$^+$(%)	UB	GAP$^+$(%)	UB	GAP$^+$(%)
1	69160	+	+	69160	0.00	69160	0.00
2	83558	+	+	+	+	+	+
3	93968	93968	0.00	93968	0.00	93968	0.00
4	273548	+	+	+	+	+	+
5	444460	+	+	+	+	+	+
6	1030299	+	+	+	+	+	+
7	311340	+	+	+	+	+	+
8	1167580	+	+	1092580	-6.42	+	+
9	1160318	+	+	+	+	+	+
10	934210	+	+	909210	-2.68	+	+
11	*	+	+	946335	-	946335	-
12	361837	+	+	+	+	+	+
13	*	+	+	+	+	+	+
14	*	808346	-	803346	-	+	+
15	*	+	+	+	+	+	+
16	*	+	+	+	+	+	+
17	*	+	+	+	+	+	+
18	777461	+	+	+	+	+	+
19	*	+	+	+	+	+	+
20	*	+	+	+	+	+	+
21	*	+	+	+	+	+	+
22	*	+	+	+	+	+	+
23	*	+	+	+	+	+	+
24	*	+	+	+	+	+	+
25	894492	+	+	+	+	+	+
26	*	+	+	+	+	+	+
27	*	+	+	+	+	+	+
28	*	+	+	+	+	+	+
29	*	+	+	+	+	+	+
30	*	+	+	+	+	+	+
31	*	+	+	+	+	+	+
32	*	+	+	+	+	1052790	-
33	369290	+	+	274290	-25.73	+	+
34	*	+	+	+	+	265682	-
35	*	+	+	+	+	270346	-
36	*	230902	-	235902	-	245902	-
37	*	254440	-	259440	-	264440	-
38	*	131710	-	141710	-	141710	-
39	*	436289	-	441289	-	441289	-
40	*	+	+	434491	-	434491	-

*: CPLEX could not return a feasible solution within 10 hours. +: No feasible solution is found by strategy H4.

Table 18 Comparison Results of Four Heuristic Strategies

Instance	H1	H2 ($\varpi=5$)		H3 ($\eta=\frac{1}{5}$)		H4 ($\eta=\frac{1}{10},\varpi=20$)	
	UB	UB	GAP$^+$(%)	UB	GAP$^+$(%)	UB	GAP$^+$(%)
1	69160	+	+	69160	0.00	69160	0.00
2	83558	88558.00	5.98	83558	0.00	83558	0.00
3	93968	93968.00	0.00	93968	0.00	93968	0.00
4	268548	263548.00	-1.86	268548	0.00	268548	0.00
5	439460	439460.00	0.00	+	+	444460	1.14
6	980299	975299.00	-0.51	980299	0.00	980299	0.00
7	271340	256340.00	-5.53	266340	-1.84	266340	-1.84
8	1157580	1087580.00	-6.05	1102580	-4.75	1087580	-6.05
9	1135318	+	+	1100318	-3.08	+	+
10	909210	899210.00	-1.10	924210	1.65	904210	-0.55
11	961335	941335.00	-2.08	951335	-1.04	941335	-2.08
12	286837	211837.00	-26.15	236837	-17.43	211837	-26.15
13	935041	880041.00	-5.88	890041	-4.81	880041	-5.88
14	863346	808346.00	-6.37	828346	-4.05	813346	-5.79
15	891967	856967.00	-3.92	871967	-2.24	+	+
16	2485945	2460945.00	-1.01	2470945	-0.60	+	+
17	2844712	2729712.00	-4.04	2859712	0.53	+	+
18	697461	617461.00	-11.47	+	+	+	+
19	1318217	1188217.00	-9.86	1198217	-9.10	1188217	-9.86
20	2556418	2396418.00	-6.26	+	+	+	+
21	599131	424131.00	-29.21	469131	-21.70	+	+
22	954823	819823.00	-14.14	824823	-13.62	+	+
23	1851825	1786825.00	-3.51	1816825	-1.89	+	+
24	1814229	1714229.00	-5.51	1724229	-4.96	1714229	-5.51
25	894492	829492.00	-7.27	844492	-5.59	824492	-7.83
26	*	1896005.00	-	+	+	1901005	-
27	1013909	953909.00	-5.92	1038909	2.47	948909	-6.41
28	1019887	959887.00	-5.88	1014887	-0.49	959887	-5.88
29	*	2096664.00	-	+	+	+	+
30	2167123	1982123.00	-8.54	+	+	1977123	-8.77
31	632640	482640.00	-23.71	497640	-21.34	497640	-21.34
32	1097790	1017790.00	-7.29	1097790	0.00	1097790	0.00
33	359290	254290.00	-29.22	319290	-11.13	274290	-23.66
34	380682	260682.00	-31.52	300682	-21.01	260682	-31.52
35	380346	260346.00	-31.55	290346	-23.66	270346	-28.92
36	260902	230902.00	-11.50	265902	1.92	240902	-7.67
37	294440	244440.00	-16.98	299440	1.70	259440	-11.89
38	211710	136710.00	-35.43	206710	-2.36	136710	-35.43
39	476289	436289.00	-8.40	456289	-4.20	446289	-6.30
40	464491	424491.00	-8.61	464491	0.00	439491	-5.38

*: Heuristic strategy H1 could not return a feasible solution. +: No feasible solution is found by heuristic strategy.