Chapter 22
An Integer Programming Model for the Ferry Scheduling Problem

Daniel Karapetyan and Abraham P. Punnen

1 Introduction

Routing and scheduling of public transport vehicles and other commercial vehicles is one of the most extensively studied areas in the operations research literature. Such problems include airline scheduling [5], scheduling of passenger and freight trains, transit bus routing and scheduling, and a variety of other vehicle routing problems [3, 4, 15, 16]. The routing and scheduling of passenger ferries is yet another problem in this class that received relatively little attention.

In a ferry scheduling problem, we are given a set of ports, a set of ferries, and a planning horizon (examples of a ferry and a terminal are given in Figs. 22.2 and 22.3). Then we need to find a routing and scheduling scheme for the ferries so that the travel demands emanating at the ports at different periods of the planning horizon are satisfied while the operating cost and passenger dissatisfaction are kept at a minimum. Note that the transition of travel demand from a lower time state to a higher time state is permitted but the reverse transition is not permitted. As reported in [7], periodic changes in ferry schedules are required for a variety of reasons. These include changing demographics within the service area, seasonal changes, changes in fleet size and characteristics, changes in port configuration, altered service level restrictions, responses to customer feedback, and/or changes in government regulations. Developing an "optimal" schedule is crucial in managing operating costs efficiently while maintaining sufficient level of satisfaction of the ferry users.

The online version of this chapter (doi: 10.1007/978-1-4939-1007-6_22) contains supplementary material, which is available to authorized users.

D. Karapetyan (✉)
ASAP Research Group, School of Computer Science, University of Nottingham, Nottingham NG8 1BB, UK
e-mail: daniel.karapetyan@gmail.com

A. P. Punnen
Department of Mathematics, Simon Fraser University, Surrey, BC V3T 0A3, Canada
e-mail: apunnen@sfu.ca

K. G. Murty (ed.), *Case Studies in Operations Research,*
International Series in Operations Research & Management Science 212,
DOI 10.1007/978-1-4939-1007-6_22, © Springer Science+Business Media New York 2015

Even for scheduling problems that appear to be of smaller size (e.g., four ferries, seven ports with a 20-h planning horizon), the ferry scheduling problem is complex and requires systematic scientific approaches to achieve high-quality schedules and identifying operational bottlenecks. Fig. 22.1 provides a sample schedule taken from our case study using real data [7].

Lai and Lo [10] studied the ferry scheduling problem and provided an integer programming formulation with the assumption that all ferries are of the same speed. That assumption, however, can easily be relaxed with appropriate modifications of the model. They also provided heuristic algorithms for the problem with excellent computational results. Wang et al. [17] considered a slightly more general problem with demand estimation included in the model, and Mitrovic-Minic and Punnen [13] studied the ferry scheduling problem of reconfigurable ferries. Recently, Karapetyan and Punnen [7] proposed yet another integer programming model for the ferry scheduling problem and reported experimental results using real data. For other related works on ferry scheduling we refer to [8, 18, 19].

Modern general-purpose integer programming solvers embed powerful heuristic algorithms and sophisticated valid inequality generators to augment branch-and-cut or branch-and-bound type enumerative schemes. This together with advances in hardware technologies increased our problem solving capabilities multifold simply by using integer programming as a stand-alone solver or as a major component of developing powerful metaheuristics [2, 6, 9, 12]. These observations provided additional motivation to study integer programming formulations for the ferry scheduling problem.

In this chapter we discuss two integer programming models for the ferry scheduling problem and present a data set to test new and old algorithms. The data set is created based on insights gained from real-world applications. Some details of experimental analysis with one of the models are presented. Students using the contents of this chapter and data sets can experiment with the models as they are or with appropriate modifications to implement additional constraints and/or objectives. Most of the notations and results we use in this chapter are taken from our paper [7]. The data set provided here and related computational results are new.

2 Notations and Definitions

We first give a formal mathematical definition of the ferry scheduling problem (FSP). Let $P = \{1, 2, \dots, n\}$ be a set of ports and $F = \{1, 2, \dots, m\}$ be a set of ferries. For each ferry $f \in F$, a home port $h^f \in P$ is assigned. Ferry f starts and ends its service at its home port h^f. Let $[\ell, L]$ be the planning horizon. For example, if $\ell = $ 5:00 a.m. and $L = $ 12:00 noon, then the planning horizon is the time interval between 5:00 a.m. and 12:00 noon. Travel demands originate at a port with a prescribed destination port at discrete times within the planning horizon. Starting from the home port at time ℓ, a ferry visits a fixed number of ports, possibly multiple times (including the home ports) and returns to the home port no later than time L. We call this a *ferry traversal*. An arrangement of all the arrival times and departure times of ferry f at each of the ports it visits in a traversal is called a schedule of the ferry f (see Fig. 22.1).

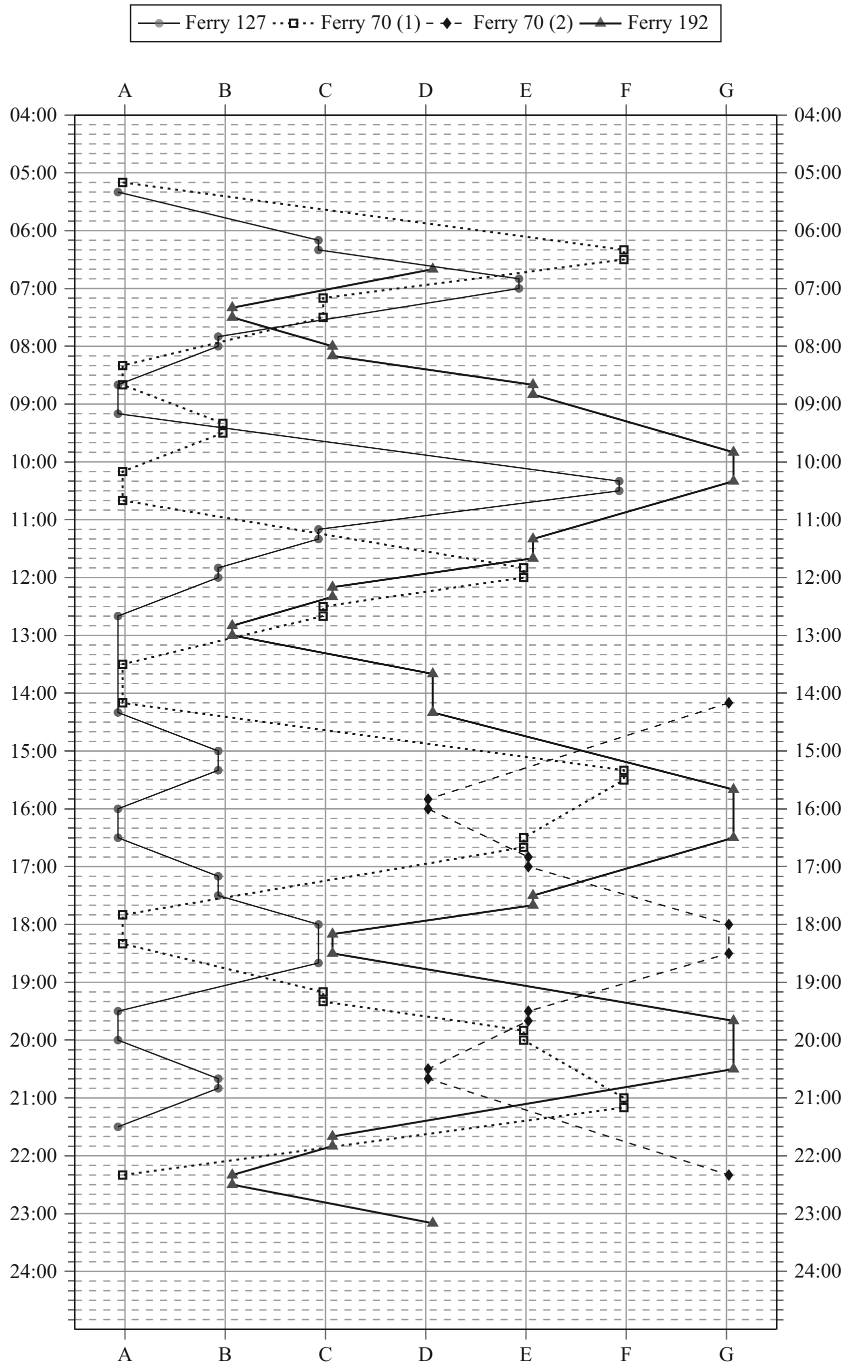

Fig. 22.1 A schedule we constructed for a peak Friday with four ferries and seven ports using real data for a ferry company

Fig. 22.2 A ferry owned by BC Ferries serving between Tsawwassen and Swartz Bay, BC, Canada

A ferry carries passengers as well as vehicles of various types such as trucks, SUVs, buses, cars, other commercial vehicles, etc. To simplify the demand types, we consider a measure called *automobile equivalent* (AEQ) which is calculated from the itemized demand using a conversion formula. Thus, hereafter we assume that demand is given in terms of AEQ. Each travel data (demand) can be represented by a 4-tuple (o, σ, t, λ) where, o is the origin port, σ is the destination port, t is the departure time, and λ is the demand volume in AEQ. The demand volume is deterministic and is part of the input data. Then the ferry scheduling problem is to develop a schedule for all the ferries to move the demand volume from the respective

Fig. 22.3 An areal view of the BC Ferries terminal at Tsawwassen, BC, Canada

Table 22.1 Main symbols and their meanings used in the chapter

m	Number of ferries
n	Number of ports
P	Set of ports
h^f	Home port of ferry f
C^f	Capacity of ferry f
β_k	Number of berths at that port k
k_i	Time state i of port k
V_k	$\{k_1, k_2, \dots, k_q\}$
$[\ell, L]$	Planning horizon
$T^f(s,k)$	Direct travel time for ferry f from port s to port k
W_p^f	Load/unload time for ferry f at port p
b	The demand volume in AEQ
δ	Time increment factor to discretize planning horizon
ψ	The length of the planning horizon
q	$1 + \frac{\psi}{\delta}$, number of time states
$G^f = (V, E^f)$	The ferry flow network corresponding to ferry f
$F(u,v)$	$\{f : (u,v) \in E^f\}$
$\tau(k_i)$	The exact time represented by the node k_i in G^f
y_{uv}^f	Decision variable representing ferry flow along (u,v)
$\Omega^k = (U, A^k)$	Passenger flow network with destination port k
x_{uv}^k	Decision variable representing the number of passengers (measured in AEQ) traveling along arc (u,v) in Ω^k
$d_{k_i}^p$	The travel demand (in terms of AEQ) originating at node k_i
T_k	The transfer time required at port k
c_{uv}^k	The cost of arc (u,v) in Ω^k
g_e^f	Operating cost of ferry f along arc e

origin ports to destination ports so that the weighted sum of total operating cost and total travel time is minimized.

Let $T^f(s,k)$ be the direct travel time for ferry f from port s to port k without intermediate stops, $k \neq s$. If travel between ports s and k are not permitted, $T^f(s,k)$ is set to a large number. We call s the origin port, k the destination, and (s,k) an origin–destination pair (OD pair). The load/unload times mostly depend on the number of vehicles uploading to and downloading from the ferry. For busy ports, load/unload times are higher, and it is reasonably short for all other ports. Further, large ferries are likely to carry more load, and hence load/unload time also depends on ferries. We denote W_p^f the load/unload time for ferry f at port p.

A summary of all symbols used in this chapter is given in Table 22.1.

3 Integer Programming Formulations

Let us now consider our two integer programming formulations for the FSP. Two models are discussed. The first one called the compact model is taken from our paper [7]. We then provide a larger model that has an increased problem size but is more flexible. This can be viewed as an enhancement to the models discussed in [7, 11]. We also present a comparison of the relative merits of the two models.

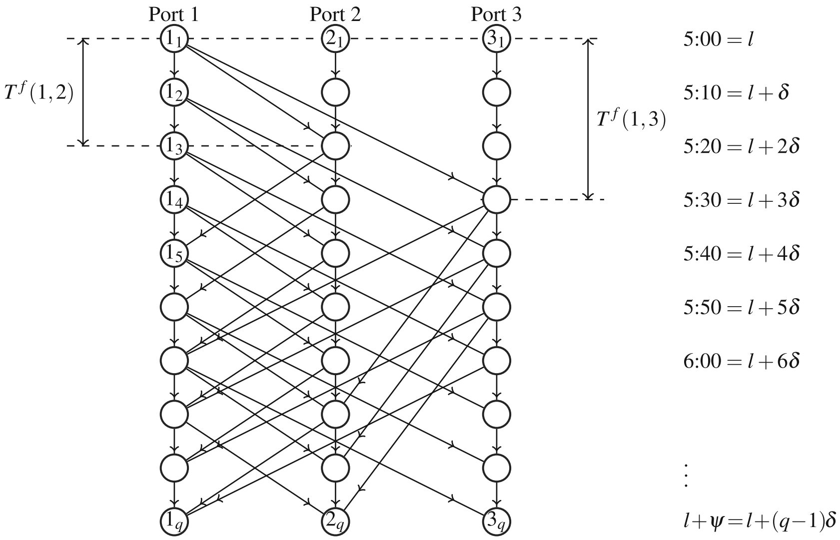

Fig. 22.4 A sample ferry flow network having three ports with *Port 1* as home port. The travel time for the ferry between *Port 1* and *Port 2* is 20 min, *Port 1* and *Port 3* is 30 min, and *Port 2* and *Port 3* is 40 min. The discretization parameter δ is set to 10 min

The backbone of our models is two networks, the *ferry flow network* and the *passenger flow network*. This is a standard approach used in many passenger vehicle models. A general overview of the model is as follows: The ferries are routed through ferry flow networks and passengers are routed through passenger flow networks. These networks are joined by imposing the constraints that a passenger cannot travel if a ferry is not available. This provides the general framework of the models. There are additional constraints that need to be handled efficiently. These include berth conflicts, transfer time constraints, load/unload times, crew exchange restrictions, among others.

Let us first construct our *ferry flow networks*. Note that the planning horizon is $[\ell, L]$ and let ψ be the length of the planning horizon. Let δ be a given *time increment factor* which is used to discretize the planning horizon. For example, ℓ represents 5:00 a.m., $\psi = 1200$ min and $\delta = 10$ min. We assume that ψ is a multiple of δ and let $q = 1 + \frac{\psi}{\delta}$.

Let $G^f = (V, E^f)$ be the ferry flow network corresponding to ferry f. For each port $k \in P$, we denote $V_k = \{k_1, k_2, \ldots, k_q\}$ to be the set of nodes in G^f representing "different time states" of port k. For example, if $\ell = $ 5:00 a.m. and $\delta = 10$ min, then k_1 represents 5:00 a.m., k_2 represents 5:10 a.m., and so on. The node set V of G^f is given by $V = \bigcup_{k \in P} V_k$. The nodes in V can be considered as a rectangular arrangement as illustrated in Fig. 22.4. We denote the exact time represented by the node k_i as $\tau(k_i)$. Thus, $\tau(k_i) = \ell + \delta(i-1)$ for all $k_i \in V$.

Introduce an arc (k_i, h_j) in E^f for $i = 1, 2, \ldots, q$ and $k, h \in P$, $k \neq h$, if a direct service from k to h is allowed and $j \leq q$ is the smallest index such that $\tau(h_j) \geq \tau(k_i) + T^f(k, h)$. Such arcs connecting time state nodes of two different ports are called *service arcs*. Also, for each $k \in P$ and $i = 1, 2, \ldots, q-1$, we introduce an arc (k_i, k_{i+1}). These arcs connect two consecutive time state nodes of the same port and are called *waiting arcs*.

An example of a ferry flow network $G^f = (V, E^f)$ with three ports is shown in Fig. 22.4. Each service arc $(k_i, h_j) \in E^f$ represents a potential service of ferry f from port k to port h with departure time $\tau(k_i)$. Each waiting arc (k_i, k_{i+1}) represents a portion (or full) in-port time of ferry f at port k between times $\tau(k_i)$ and $\tau(k_{i+1})$.

For any $v \in V$, let $I^f(v) = \{v' : (v', v) \in E^f\}$ and $O^f(v) = \{v' : (v, v') \in E^f\}$. For each $(u, v) \in E^f$ consider the 0–1 variable y^f_{uv} defined by

$$y^f_{uv} = \begin{cases} 1 & \text{if ferry } f \text{ traverses arc } (u,v) \\ 0 & \text{otherwise.} \end{cases}$$

Define

$$b^f_v = \begin{cases} -1 & \text{if } v = h^f_1 \\ 1 & \text{if } v = h^f_q \\ 0 & \text{otherwise.} \end{cases}$$

The flow of ferry f along the ferry flow network from the beginning of the planning horizon to the end of the planning horizon is governed by the *ferry flow balancing constraints* which are given by:

$$\sum_{v' \in I^f(v)} y^f_{v'v} - \sum_{v' \in O^f(v)} y^f_{vv'} = b^f_v \text{ for every } v \in V \text{ and } f \in F. \tag{22.1}$$

Constraints of the type (22.1) are widely used in the network flow literature and generally known as flow balancing constraints. For an excellent treatment of network flow problems we refer to the books [1, 14]. The ferry flow balancing constraints allow ferry f to pass through the arcs of the ferry flow network and eventually reach back to the home port at the end of the planning horizon.

To facilitate load and unload operations at a node k_i, $i \neq q$ of G^f, we want to make sure that a ferry that enters k_i through a service arc stays at k_i for at least W^f_k min before departing from k_i. This can be achieved with the following inequalities forcing that the ferry traverse at least $w^f_k = \left\lceil \frac{W^f_k}{\delta} \right\rceil$ consecutive waiting arcs:

$$|\Delta^f_{k_i}| \sum_{v' \in I^f_{\text{serv}}(k_i)} y^f_{v'k_i} \leq \sum_{(u,v) \in \Delta^f_{k_i}} y^f_{uv} \text{ for every } k_i \in V \text{ and } f \in F, \tag{22.2}$$

where $\Delta^f_{k_i} = \{(k_i, k_{i+1}), (k_{i+1}, k_{i+2}), \ldots (k_{r-1}, k_r)\}$, $r = \min\{i + w^f_k, q\}$, and $I^f_{\text{serv}}(k_i) = \{v' : (v', k_i) \in E^f, v' \notin V_k\}$. We call (22.2) the *load/unload constraints*.

Our next task is to make sure there are no berth conflicts arise at any of the ports. Note that at any time, the number of ferries staying at port k should not exceed the number of berths β_k at that port. We guarantee this by introducing the *berth constraints* given below:

$$\sum_{f \in F} y^f_{k_i k_{i+1}} \leq \beta_k \text{ for every waiting arc } (k_i, k_{i+1}), k_i \in V_k, k \in P, i \neq q. \quad (22.3)$$

The ferry flow network is the same for both integer programming models that we consider. However, the passenger flow networks in these two models are different in size and structure.

3.1 Passenger Flow Networks for the Compact Model

Let $G = (V, E)$ be the minimal supergraph of all $G^f, f \in F$. Thus, $(i, j) \in E$ implies $(i, j) \in E^f$ for at least one f. Note that all ferry flow networks have the same vertex set V which is also the vertex set of G. The network topology of G^f will be the same as G for all f if all ferries take the same time to travel between two specified ports. The only difference in this case would be the home port designation, which does not affect the network topology. However, we permit ferries to operate at different speed levels, and hence in our case, G does not need to be identical to G^f. Create new nodes $\zeta_1, \zeta_2, \ldots, \zeta_n$ and arcs (k_q, ζ_i), called *infeasibility arcs*, for $i \in P \setminus \{k\}$, $k \in P$. The infeasibility arcs are introduced to detect infeasibility easily and to measure the "magnitude" of infeasibility. In an optimal solution, if there is a positive passenger flow along an infeasibility arc, then the ferry scheduling problem is infeasible. Obviously, there are other ways to detect infeasibility of the system. We also introduce the arcs (k_i, ζ_k) called destination arcs for port k, for $i = 1, 2, \ldots, q$. The resulting graph $\Omega^k = (U, A^k)$ is called the *passenger flow network for destination port k* (see Fig. 22.5). Note that $U = V \cup \{\zeta_1, \zeta_2, \ldots \zeta_n\}$, and A^k consists of E together with all destination arcs for port k and the infeasibility arcs.

Let $d^p_{k_i}$ be the travel demand (in terms of AEQ) originating at node k_i, $k \in P$, $i = 1, 2, \ldots, q$ with destination port p. If there is no travel demand originating at node k_i with destination port p, we choose $d^p_{k_i} = 0$. The travel demand with destination port p at the node ζ_t of Ω^p is given by

$$d^p_{\zeta_t} = \begin{cases} -\sum_{k \in P} \sum_{i=1}^{q} d^p_{k_i} & \text{if } t = p \\ 0 & \text{otherwise.} \end{cases}$$

Let x^k_{uv} be the number of passengers (measured in AEQ) traveling along arc (u, v) in Ω^k. Thus the destination port of passengers x^k_{uv}, $(u, v) \in \Omega^k$ is port k. To guarantee that all passengers reach their destination port, we have *passenger flow balancing constraints* for each passenger flow network, i.e.,

$$\sum_{v' \in I_k(v)} x^k_{v',v} - \sum_{v' \in O_k(v)} x^k_{v,v'} = -d^k_v \text{ for every } v \in U \text{ and for every } k \in P, \quad (22.4)$$

where $I_k(v) = \{v' : (v', v) \in A^k\}$ and $O_k(v) = \{v' : (v, v') \in A^k\}$. Note that service arcs are the same in Ω^k for all k. The primary difference between different passenger

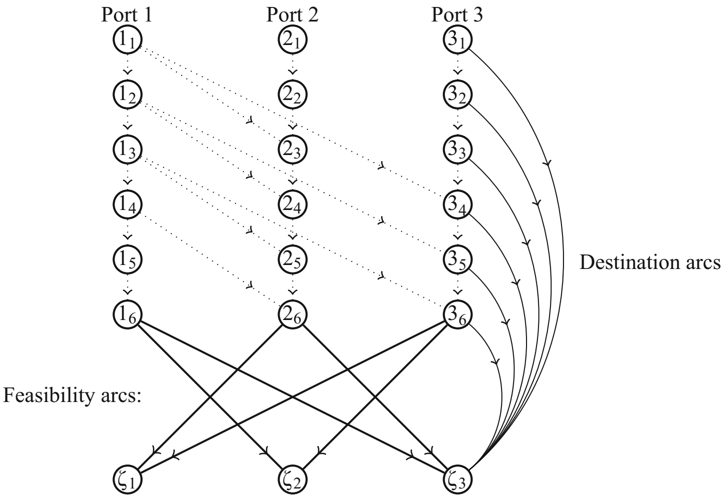

Fig. 22.5 A passenger flow network Ω^3 for aggregated decision variables with $q = 6$ and destination *Port 3*

flow networks is the destination and infeasibility arcs. Let Ω be the minimal supergraph of Ω^k for all k, i.e., each arc in Ω is an arc in at least one Ω^k for some k. For each service arc (u, v) of Ω, let $F(u, v) = \{f : (u, v) \in E^f\}$. The *capacity* of a service arc (u, v) in Ω is given by $\sum_{f \in F(u,v)} C^f y^f_{uv}$, where C^f is the capacity of ferry f. A passenger can travel along a service arc only if a ferry traverses that arc. To insure this, we require that the sum of passenger flows to all destinations along a service arc should not exceed the capacity of that arc, i.e.,

$$\sum_{k \in P} x^k_{uv} \le \sum_{f \in F(u,v)} C^f y^f_{uv} \text{ for every service arc } (u, v) \in \Omega \text{ and } k \in P. \tag{22.5}$$

The inequalities (22.5) are referred to as *capacity constraints*.

It is possible that some passengers might not travel to their destinations directly in the ferry and might need to transfer from one ferry to another one or more times. Although multiple transfers are not desirable, the model does not impose any penalty for such travel. However, to make such transfers feasible, we need to allocate sufficient time for passengers to unload from their ferry and load onto another ferry.

For simplicity, we assume that the transfer time at a given port is independent of the type of ferries and time state nodes. Let T_k be the transfer time required at port k, $\hat{T}_k = \lceil \frac{T_k}{\delta} \rceil$ and $\Delta_{k_i} = \{k_i, k_{i+1}, \ldots, k_{r-1}\}$, where $r = \min\{i + \hat{T}_k, q\}$. Let $I_{\text{serv}}(k_i) = \{v' : (v', k_i) \in E, v' \notin V_k\}$. Then the *transfer constraints* can be stated as

$$\sum_{j=i}^{t} \sum_{v' \in I_{\text{serv}}(k_j)} x^p_{v'k_j} \le x^p_{k_t k_{t+1}} \text{ for } t = i \text{ to } r - 1, k_i \in V, i \neq q \text{ and } p \in P \setminus \{k\}. \tag{22.6}$$

Note that (22.6) affects not only the transferring passengers but also the passengers remaining on a ferry. Thus, we require $T_k \leq \min_{f \in F} W_k^f$.

The transfer constraints force that the flow with destination port p along each of the arcs $(k_j, k_j + 1)$, $k_j \in \Delta_{k_i}$ is at least the total flow with destination port p arriving at node k_i through service arcs. Note that we have $|\Delta_{k_i}|$ transfer inequalities for each $k_i \in V$, $k \in P$. If port k has only one berth or $\hat{T}_k = 1$, we can replace the $|\Delta_{k_i}|$ transfer inequalities at node k_i by a single inequality

$$\sum_{v' \in I_{\text{serv}}(k_i)} x_{v'k_i}^p \leq x_{k_i k_{i+1}}^p \text{ for every } k_i \in V, i \neq q \text{ and } p \in P. \tag{22.7}$$

The next ingredient in the development of our model is the objective function. This depends on our "objective," as the name suggests. One can consider different factors in defining the objective which is a combination of operating cost, passenger dissatisfaction, number of transfers, etc. We restrict our goal to "minimize" the operating cost and "maximize" the level of service. Since these two objectives are contradictory in nature, we define the objective function as a compromise between these competing goals. Let μ^f be the cost of operating ferry f for 1 h. This cost includes fuel costs and crew salary. For each edge $e \in E^f$ define its ferry operating cost g_e^f as

$$g_e^f = \mu^f \cdot (\tau(v) - \tau(u)),$$

where $e = (u, v)$.

Let c_{uv}^k be the cost of arc (u, v) in the passenger flow network Ω^k. Then c_{uv}^k is given by

$$c_{uv}^k = \begin{cases} M & \text{if } (u,v) \text{ is an infeasibility arc} \\ 0 & \text{if } (u,v) \text{ is a destination arc} \\ \tau(h_j) - \tau(k_i) & \text{otherwise, where } (u,v) = (k_i, h_j), \end{cases}$$

where M is a very large number. Note that the destination arcs have zero cost and any passenger that does not reach the correct destination port will travel through the infeasibility arcs incurring a very large cost. This is to make sure the passengers reach their destination if there is a feasible solution. Using these cost elements, we define our objective function as

$$\phi(x, y) = \lambda \sum_{f \in F} \sum_{e \in E^f} g_e^f y_e^f + \nu \sum_{k \in P} \sum_{e \in A^k} c_e^k x_e^k, \tag{22.8}$$

where $\lambda \geq 0$ and $\nu > 0$ are coefficients controlling the cost/level of service balance and we want to minimize $\phi(x, y)$.

The compact integer programming model for the ferry scheduling problem discussed above is summarized as follows:

$$\text{Minimize} \quad \lambda \sum_{f \in F} \sum_{e \in E^f} g_e^f y_e^f + \nu \sum_{k \in P} \sum_{e \in A^k} c_e^k x_e^k$$

subject to:

$$\sum_{v' \in I^f(v)} y^f_{v'v} - \sum_{v' \in O^f(v)} y^f_{vv'} = b^f_v \text{ for every } v \in V \text{ and } f \in F,$$

$$|\Delta^f_{k_i}| \sum_{v' \in I^f_{\text{serv}}(k_i)} y^f_{v'k_i} \leq \sum_{(u,v) \in \Delta^f_{k_i}} y^f_{uv} \text{ for every } k_i \in V \text{ and } f \in F,$$

$$\sum_{f \in F} y^f_{k_i k_{i+1}} \leq \beta_k \text{ for every waiting arc } (k_i, k_{i+1}), k_i \in V_k, k \in P, i \neq q,$$

$$\sum_{v' \in I_k(v)} x^k_{v',v} - \sum_{v' \in O_k(v)} x^k_{v,v'} = -d^k_v \text{ for every} v \in U \text{ and for every } k \in P,$$

$$\sum_{k \in P} x^k_{uv} \leq \sum_{f \in F(u,v)} C^f y^f_{uv} \text{ for every service arc} (u,v) \in \Omega,$$

$$\sum_{j=i}^{t} \sum_{v' \in I_{\text{serv}}(k_j)} x^p_{v'k_j} \leq x^p_{k_t k_{t+1}} \text{ for } t = i \text{ to } r-1,$$

$$k_i \in V, i \neq q \text{ and } p \in P \setminus \{k\},$$

$$y^f_{uv} \in \{0,1\} \text{ for } (u,v) \in E^f, f \in F,$$

$$x^k_{uv} \text{ is a nonnegative integer } for (u,v) \in A^k, k \in P.$$

The model discussed above has $O(qmn^2)$ constraints and $O(qmn^3)$ variables. This gives a reasonably compact model that can be solved using general-purpose integer programming solvers for moderate size problems. The compactness, however, comes with a disadvantage that the model is less flexible. We cannot distinguish passengers traveling to the same destination in terms of their origin ports. However, in our case it is not necessary because the model allows us to calculate the total traveling time of all the passengers.

3.2 Enlarged Integer Programming Model

Let us now consider a more elaborate model by increasing the number of variables to gain additional flexibility. If we use demand types as OD pair, as considered in [11, 13, 17], we can get an integer programming formulation for the ferry scheduling problem with $O(qmn^3)$ constraints and $O(qmn^4)$ variables. This increases the model size by a factor of $O(n)$ compared to the compact model discussed above.

We now consider a further elaborate model where passengers are completely distinguishable in terms of origin port, destination port, and time of arrival at the origin port. This obviously increases the number of decision variables, but allows considerable flexibility. For example, if we know that most of the passengers arriving at a given port, say port k, at 7:30 a.m. and going to port t are students who need to reach their destination by 8:30 a.m., we will be able to impose this by a hard constraint or modifying the cost of the corresponding variables in the objective function.

For each pair (k_i, t), where k_i represents port k at time state i (essentially a node k_i of the ferry flow network) and t represents a destination port, we define a passenger flow network $\Omega^{k_i,t} = (U^{k_i,t}, A^{k_i,t})$. The network $\Omega^{k_i,t}$ is the same as Ω^t (i.e. $U^{k_i,t} = U^t$ and $A^{k_i,t} = A^t$) except that the demand $d_{r_j}^{k_i,t}$ at node r_j, for $r \in P$ and $j = 1, 2, \ldots q$ is given by

$$d_{r_j}^{k_i,t} = \begin{cases} d_{k_i}^t & \text{if} r_j = k_i \\ -d_{k_i}^t & \text{if} r_j = \zeta_t \\ 0 & \text{otherwise.} \end{cases}$$

Let $x_{uv}^{k_i,t}$ be the number of passengers with origin node k_i (i.e., origin port k and time of arrival at port k corresponds to the time state i in the network), destination node t, and traveling along arc (u, v) of the passenger flow network $\Omega^{k_i,t}$. Then the passenger flow balancing constraints become:

$$\sum_{v' \in I_v^{k_i,t}} x_{v',v}^{k_i,t} - \sum_{v' \in O_v^{k_i,t}} x_{v,v'}^{k_i,t} = -d_v^{k_i,t} \text{for every } v \in U^{k_i,t} \text{ and for every } (k_i, t) \in (V \times P), \tag{22.9}$$

where, $I_v^{k_i,t} = \{v' : (v', v) \in A^{k_i,t}\}$ and $O_v^{k_i,t} = \{v' : (v, v') \in A^{k_i,t}\}$.

As in the case of our compact model, the service arcs are the same in $\Omega^{k_i,t}$ for all $(k_i, t) \in V \times P$. It can be verified that be the minimal supergraph of $\Omega^{k_i,t}$ for all $(k_i, t) \in V \times P$ is the same the graph Ω defined in the previous section,. i.e., each arc in Ω is an arc in at least one $\Omega^{k_i,t}$ for some (k_i, t). Recall that for each service arc (u, v) of Ω, we define $F(u, v) = \{f : (u, v) \in E^f\}$ and the *capacity* of a service arc (u, v) in Ω by $\sum_{f \in F(u,v)} C^f y_{uv}^f$, where C^f is the capacity of ferry f. A passenger can travel along a service arc only if a ferry traverses that arc. Thus, the sum of passenger flows to all destinations along a service arc should not exceed the capacity of that arc, i.e.,

$$\sum_{(k_i,t) \in v \times P} x_{uv}^{k_i,t} \le \sum_{f \in F(u,v)} C^f y_{uv}^f \text{ for every service arc } (u, v) \in \Omega\,. \tag{22.10}$$

Again, as in our previous model, we call constraints (22.10) the *capacity constraints*. We now discuss the transfer constraints for our model. Recall that T_k is the transfer time required at port k, $\hat{T}_k = \left\lceil \frac{T_k}{\delta} \right\rceil$ and $\Delta_{k_i} = \{k_i, k_{i+1}, \ldots, k_{r-1}\}$, where $r = \min\{i + \hat{T}_k, q\}$. Let $I_{\text{serv}}(k_i) = \{v' : (v', k_i) \in E, v' \notin V_k\}$. Then the *transfer constraints* can be stated as

$$\sum_{j=i}^{t} \sum_{v' \in I_{\text{serv}}(k_j)} x_{v'k_j}^p \le x_{k_t k_{t+1}}^p \text{ for } t = i \text{ to } r - 1, k_i \in V, i \ne q \text{ and } p \in P \setminus \{k\}. \tag{22.11}$$

Note that (22.11) affects not only the transferring passengers but also the passengers remaining on a ferry. Thus, we require $T_k \le \min_{f \in F} W_k^f$.

To complete this model, we need to define the objective function and transfer constraints. We leave it for the reader to complete this, using insights gained from our compact model.

3.3 Model Refinements and Implementation Details

The formulations discussed in the previous sections are reasonably general representations of the ferry scheduling problem. Additional refinements may be necessary to enhance the model, depending on specific application and availability of information. We illustrate this with a case study using five ports and three ferries. The data used in this study is simulated based on our experience in a real case study consisting of seven ports and four ferries [7]. The problem size is slightly reduced in the present case, yet kept reasonably large, so that the readers can experiment with the data and make reasonable conclusions. These refinements are taken directly from our paper [7] keeping the same notations and terminology, although the data and experimental results reported are different.

We restrict ourselves to the compact model that is used in our experiments. The readers can make appropriate changes to the distributed model and run experiments with our data. This is indicated as an exercise at the end of this chapter.

Note that to facilitate load/unload operations, we need to make sure that a ferry f entering port k through node k_i must stay at port k for at least $w_k^f \delta$ time. The load/unload constraints (22.2) assures this in our model. However, there are alternative ways to handle this. In our experiments, constraints (22.2) are replaced by

$$\sum_{v' \in O^f_{\text{serv}}(k_i)} y_{k_i,v'} \leq y_{k_j,k_{j+1}} \text{ for every } k_i \in V \text{ and } f \in F\,, \tag{22.12}$$

where $O^f_{\text{serv}}(v) = \{v' \in O^f(v) : (v,v') \text{ is a service arc}\}$ and j is the largest index such that $\tau(k_i) - \tau(k_j) \geq W_k^f$. If no such j exists, we do not have a constraint for the corresponding node k_i. This constraint is invalid if $W_k^f \geq \min_{h\in P} T^f(k,h) + T^f(h,k)$ and our applications are consistent with this requirement.

We also replaced constraints (22.6) by constraints (22.7) in our case study. It reduces the number of constraints and simplifies the model without affecting solution quality. Again, it is a restriction on the general model but the data used in our case study permits this simplification.

The cost of a waiting arc $e = (k_i, k_{i+1})$ is set to $g_e^f = \delta\mu_f$ in the general model. If all ferries are in operation exactly at the beginning of the planning horizon and continue service till the end of the planning horizon, this cost assignment is perfectly valid. However, this can lead to over estimation of cost (and hence affect optimality) in some applications. It is not necessary that all ferries need to be in operation exactly at 5:00 a.m. A ferry could start service late, say at 10:00 a.m. and terminate service at 4:00 p.m. It can also skip all morning or serve only in the morning. The general

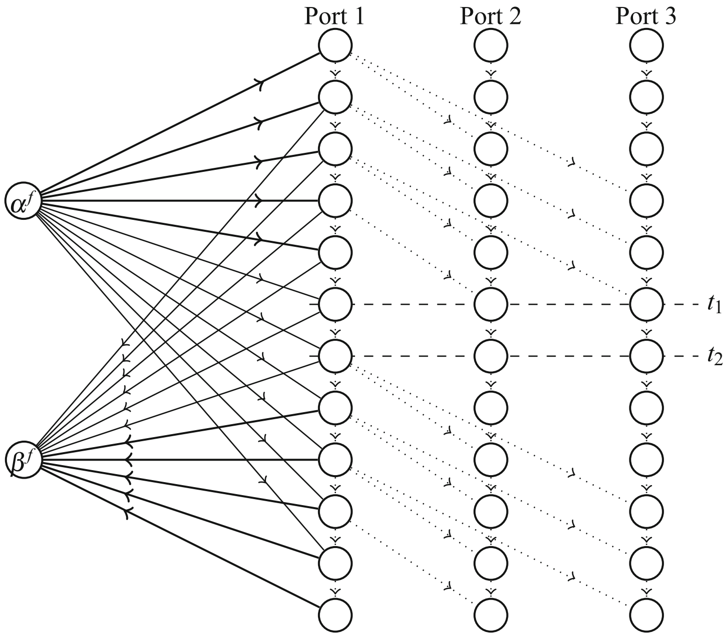

Fig. 22.6 Ferry flow network with nodes α^f and β^f that allow efficient calculation of the crew salaries and loading/unloading stays

discussions given in the last section does not take care of this, leading to suboptimal solutions. With a minor change in the model, we can address this difficulty.

Introduce two new nodes α^f and β^f to the ferry flow network G^f, $f \in F$ and connect α^f to all nodes in $V_{h^f} \setminus \{h_q^f\}$ using arcs (α^f, h_i^f), $i = 1, 2, \ldots, q-1$, referred to as *in-port arcs*. Similarly, connect each node of $V_{h^f} \setminus \{h_1^f\}$ to β^f and the resulting arcs (h_i^f, β^f), $i = 2, 3, \ldots, q$ are called *out-port arcs* (see Fig. 22.6).

We set $g_e^f = 0$ whenever e is an in-port or out-port arc. It is assumed that ferry f is stationed at node α in G^f and returns to node β in G^f at the end of service. Consequently, we change the definition of b_v^f for this modified G^f as

$$b_v^f = \begin{cases} -1 & \text{if } v = \alpha^f \\ 1 & \text{if } v = \beta^f \\ 0 & \text{otherwise.} \end{cases}$$

Normally, the ferry crew operates in two shifts per day. Dividing the planning horizon into two is not a good approach as some passengers may start their travel in the morning and finish it in the afternoon. Let $[t_1, t_2]$ be a given time interval during which the ferry must come back to its home port for crew exchange, regardless of the number of hours the ferry was in service. Let σ^f be the total crew salary for ferry f per shift.

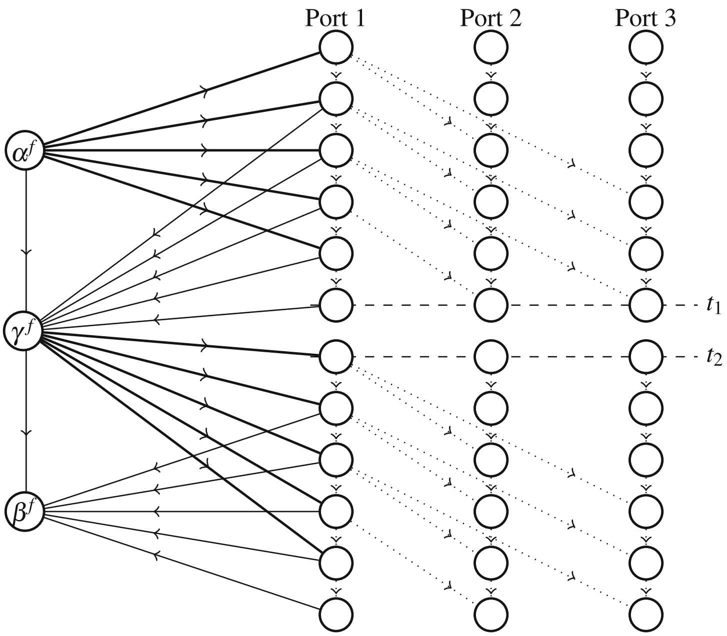

Fig. 22.7 Ferry flow network with nodes α^f, β^f, and γ^f that allow better calculation of the crew salaries and correct calculation of the fuel charges

In G^f, introduce three nodes α^f, β^f, and γ^f. Join α^f to $h_i^f \in V_{h^f}$ by an arc with cost σ^f whenever $\tau(h_i^f) < t_1$. Likewise, join h_i^f to γ^f by an arc (h_i^f, γ^f) of zero weight for every $i > 1$ such that $\tau(h_i^f) \leq t_1$. Set the cost g_e^f of the waiting arcs e involving time state nodes v with $t_1 \leq \tau(v) \leq t_2$ to a very large number M.[1] Then connect γ^f to h_i^f by an arc (γ^f, h_i^f) of weight σ^f for every $i < q$ such that $\tau(h_i^f) \geq t_2$. Also, connect each node h_i^f to β^f by an arc (h_i^f, β^f) of zero weight for every i such that $\tau(h_i^f) > t_2$. Finally connect α^f to γ^f and γ^f to β by arcs of zero weights, see Fig. 22.7.

We leave it to the reader to verify that the modifications discussed here achieve the desired objective. We used this modified model in the experimental analysis discussed in the next section.

4 Experimental Analysis

The integer programming model with aggregated variables, incorporating modifications discussed in the previous section was tested on simulated data with three ferries and five ports. The data were generated using insights gained from an earlier

[1] We assume that such arcs exist.

Table 22.2 Travel time for the ferries. 999 indicates that the corresponding travel is not permitted

	Small ferry					Large ferry				
Ports	Port 1	Port 2	Port 3	Port 4	Port 5	Port 1	Port 2	Port 3	Port 4	Port 5
Port 1	0	40	50	40	999	0	40	50	40	999
Port 2	40	0	50	30	999	40	0	50	30	70
Port 3	50	50	0	40	999	50	50	0	40	75
Port 4	40	30	40	0	999	40	30	40	0	65
Port 5	999	999	999	999	0	999	70	75	65	0

case study involving four ferries and seven ports. We have two sets of data: peak season and off-peak season. We only consider vehicles to define the demand value. However, if no vehicles were transferred between two ports but there were some foot passengers, we assume the demand volume to be 1 AEQ to ensure that these passengers will be taken into account by the model. The terminology passenger (in terms of AEQ) refers to both foot passengers as well as vehicles. The total demand in the considered problem was from 1000 to 2000 AEQ per day.

The integer programming model was solved using CPLEX 12.5 and the problem input was prepared using a C# code. The experiments were run on a Windows 7 PC with an Intel i7-3820 3.6 GHz CPU and 16 GB of memory. The parameter values for the model were set as follows: $\delta = 10$ min, $\ell = 5{:}00$, $\psi = 19$ h. In our test data, we have three ferries, two are of capacity 100 AEQ called small ferries and one having capacity of 200 AEQ, which is called large ferry. For small ferries, fuel cost is £ 400/h enroute and £ 160/h in port. Similarly for large ferry, fuel cost is £ 600/h enroute and £ 240/h in port. The crossing times for these ferries are given in Table 22.2.

Table 22.3 represents travel demand in terms of AEQ.

Figure 22.8 gives the schedule generated by CPLEX running 4 h and it reached within 0.43 % of optimality using the value 10 for cost coefficient. Figure 22.9 gives the schedule generated by CPLEX running 4 h and it reached within 0.44 % of optimality using the value 1 for cost coefficients.

5 Conclusions

In the current chapter, we presented a compact integer programming model for the ferry scheduling problem. This model has reduced size compared to the standard models by a factor of approximately $O(n)$. As shown, this is achieved at the cost of considerably lower flexibility. However, it efficiently handles all the operational constraints including load/unload times and passenger transfers. The model was able to produce high-quality solutions in 4 h using CPLEX 12.5. We also provided a more flexible model with increased number of variables and constraints.

Table 22.3 Demand data used in our experiments. *Origin* origin port, *Destination* destination port, *Desired dep. time* desired departure time, *Desired arr. time* desired arrival time, *AEQ* demand in terms of AEQ

Origin	Destination	Desired dep. time	Desired arr. time	AEQ
Port1	Port4	06:30	07:30	21
Port3	Port5	06:30	07:30	32
Port1	Port2	07:00	08:00	4
Port1	Port3	07:00	08:00	6
Port2	Port1	08:00	08:30	91
Port3	Port1	08:00	08:30	80
Port4	Port1	08:00	08:30	40
Port5	Port2	08:00	09:00	31
Port5	Port3	08:00	09:00	76
Port5	Port4	08:00	10:00	7
Port4	Port2	09:00	10:00	8
Port1	Port2	10:00	11:00	45
Port1	Port3	10:00	11:00	65
Port2	Port1	11:00	12:00	49
Port2	Port5	11:00	12:00	34
Port3	Port1	11:00	12:00	77
Port3	Port5	11:00	12:00	48
Port4	Port1	11:00	12:00	33
Port4	Port5	11:00	12:00	20
Port1	Port3	12:00	13:00	15
Port2	Port5	14:00	15:00	40
Port3	Port5	14:00	15:00	79
Port2	Port4	15:00	16:00	8
Port3	Port1	15:00	16:00	23
Port1	Port2	16:00	16:30	32
Port4	Port5	16:00	17:00	32
Port1	Port3	16:30	17:00	40
Port1	Port4	16:30	17:00	54
Port1	Port2	18:00	19:00	40
Port1	Port3	18:00	19:00	32
Port4	Port1	18:00	19:00	20
Port2	Port1	19:00	20:00	21
Port3	Port1	19:00	20:00	8
Port5	Port2	19:00	20:00	31
Port5	Port3	19:00	20:00	93
Port1	Port2	20:00	21:00	42
Port1	Port3	20:00	21:00	30
Port1	Port4	20:00	21:00	19
Port3	Port5	20:00	21:00	52
Port5	Port2	20:00	21:00	11
Port5	Port3	20:00	21:00	39
Port5	Port4	20:00	21:00	14

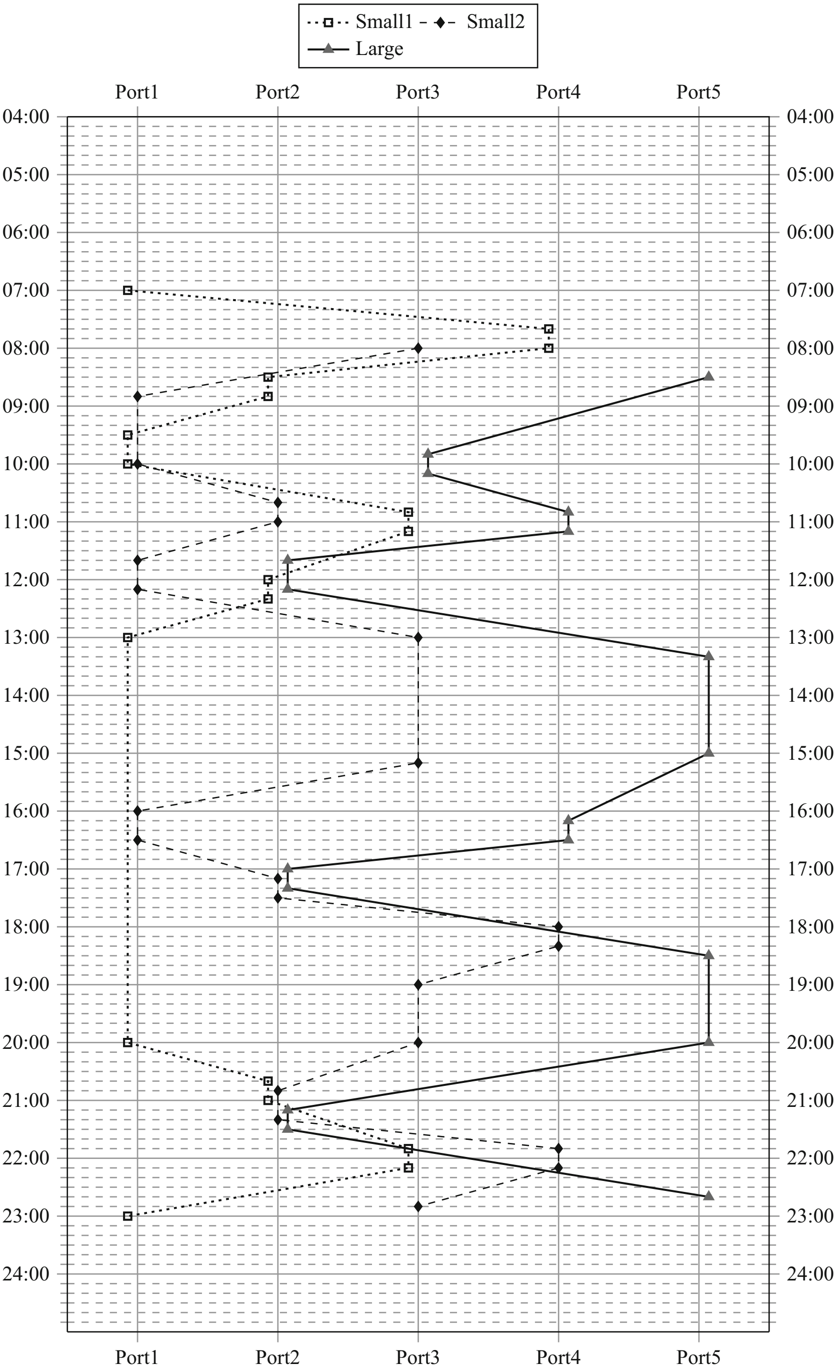

Fig. 22.8 Schedule generated by CPLEX for the cost coefficient 10 after running 4 h. The optimality gap reached is 0.43 %

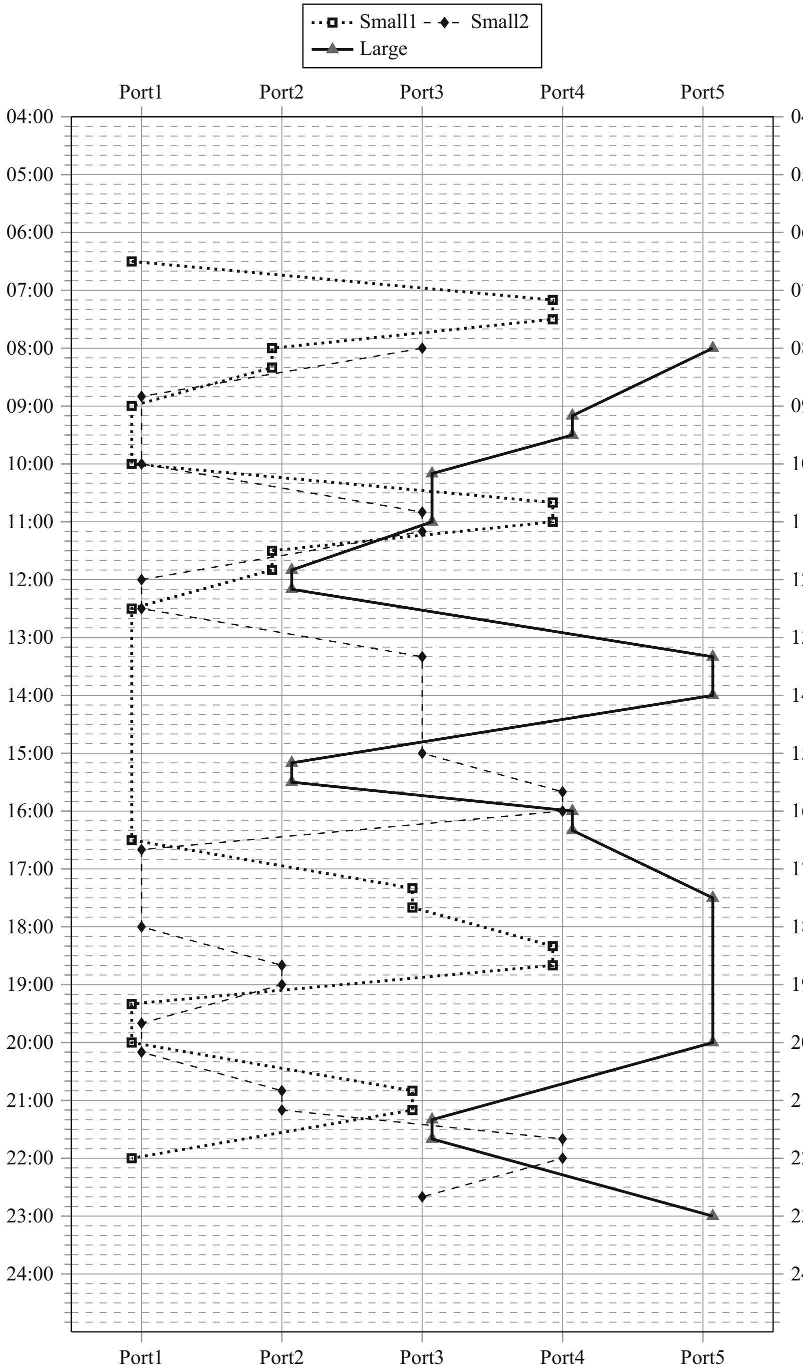

Fig. 22.9 Schedule generated by CPLEX for the cost coefficient 1 after running 4 h. The optimality gap reached is 0.44 %

6 Exercises

1. Develop the objective function and transfer constraints for the second integer programming model. If a desired arrival time for passengers is known, how do you integrate this in this model?
2. Solve the model above using CPLEX and the data set provided.
3. Compare and contrast the advantages and disadvantages of the two models given.
4. Introduce an additional group of passengers and explain how to handle this in an integer programming model where these groups are distinguishable.
5. Identify new meaningful constraints that can be added to the models and solve the resulting problems.
6. Develop metaheuristic algorithms to solve the FSP and report results of experimental analysis with the data provided.

Acknowledgment We are thankful to Steve Anderson and Peter Simpson at BC Ferries for their help and support in this work. Comments of Katta G. Murty, editor of this book, improved the presentation of this chapter. This work was partially supported by BC Ferries and MITACS. The chapter is primarily based on our research paper [7], but it also includes additional new material and experimental results.

References

1. Ahuja, R. K., Orlin, J. B., & Magnanti, T. L. (1993). *Network flows: Theory, algorithms, and applications*. Upper Saddle River: Prentice Hall.
2. Bixby, R. E. (2010). Mixed-integer programming: it works better than you may think. *FERC Conference*, Washington: DC.
3. Ceder, A. (2003). Designing public transport network and routes. In W. Lam and M. Bell (eds.), *Advanced modeling for transit operations and service planning*. Amsterdam: Elsevier.
4. Goetschalckx, M. (2011). *International series in operations research & management science. Supply chain engineering*. Berlin: Springer.
5. Gopalan, R., & Talluri, K. T. (1998). Mathematical models in airline schedule planning: A survey. *Annals of Operations Research, 76,* 155–185.
6. Hvattum, L. M., Lokketangen, A., & Glover, F. (2012). Comparisons of commercial MIP solvers and an adaptive memory (tabu search) procedure for a class of 0-1 integer programming problems. *Algorithmic Operations Research, 7,* 13–20.
7. Karapetyan, D., & Punnen, A. P. (2013). A reduced integer programming model for the ferry scheduling problem. *Public Transport, 4,* 151–163.
8. Khairy Adly Wan Zaimi, W. M., Abu, M. S., Junoh, A. K., Ariffin, W. N. M. (2011). Ferry scheduling model using linear programming technique. In *2011 3rd International Conference on Computer Research and Development (ICCRD)* (pp. 337–341), vol. 4, 11–13 March 2011
9. Koch, T., Achterberg, T., Anderson, E., Bastert, O., Berthold, T. Bixby, R. E., Danna, E., Gamrath, G., Gleixner, A. M., Heinz, S., Lodi, A., Mittelmann, H., Ralphs, T., Salvagnin, D., Steffy, D. E., & Wolter, K. (2011). MILP 2010: Mixed integer programming library version 5. *Mathematical Programming Computation, 3,* 103–163.
10. Lai, M. F., Lo, H. K. (2004). Ferry service network design: Optimal fleet size, routing, and scheduling. *Transportation Research Part A: Policy and Practice, 38,* 305–328.

11. Linderoth, J. T., & Lodi, A. (2011). MILP software. In J. J. Cochrane (Ed.), *Wiley Encyclopedia of Operations Research and Management Science*. Hoboken: Wiley.
12. Lodi, A. (2012). *MIP computation and beyond*. Technical report ARRIVAL-TR-0229.
13. Mitrovic-Minic, S., & Punnen, A. P.. (2011). Routing and scheduling of a heterogeneous fleet of re-configurable ferries: A model, a heuristic, and a case study. In *OR 2011 - International Conference on Operations Research*, Zurich, Switzerland, 2011.
14. Murty, K. G.. (1992). *Network programming*. Upper Saddle River: Prentice Hall.
15. Toth, P., & Vigo, D. (2002). *The vehicle routing problem*. Philadelphia: SIAM.
16. Voß, S., & Daduna, J. R. (Eds.) (2001). *Springer lecture notes in economics and mathematical systems, vol. 505. Computer-aided scheduling of public transport*. Berlin: Springer.
17. Wang, Z. W., Lo, H., & Lai, M. F. (2008). Mixed-fleet ferry routing and scheduling. In M. Hickman, P. Mirchandani, and S. Voss (Eds.), *Lecture notes in economics and mathematical systems 600. Computer-aided systems in public transport* (pp. 181–194). Berlin: Springer.
18. Wang, D. Z. W., & Lo, H. K. (2008). Multi-fleet ferry service network design with passenger preferences for differential services. *Transportation Research Part B, 42,* 798–822.
19. Yan, S., Chen, C.-H., Chen, H.-Y., & Lou, T.-C. (2007). Optimal scheduling models for ferry companies under alliances. *Journal of Marine Science and Technology, 15,* 53–66.