Minimum Cost Noncrossing Flow Problem on Layered Networks

İ. Kuban Altınel*, Necati Aras, Zeynep Şuvak, Z. Caner Taşkın

Department of Industrial Engineering, Boğaziçi University, 34342, Bebek, İstanbul, Turkey

Abstract

In this work we focus on an extension of the minimum cost flow problem in layered networks. Feasible arc flows must satisfy a specific compatibility restriction in addition to flow balance and capacity restrictions. Namely, at most one of the crossing arcs is allowed to have positive flow on it. This variant of the minimum cost flow problem, which we call the minimum cost noncrossing flow problem, can frequently be encountered in real life. The determination of optimal temporal quay crane allocations to berthed vessels in container terminals, and optimal train schedules through the stations on the same railroad line are two examples. We first analyze the complexity of the problem and show that the *noncrossing flow problem* is in fact NP-complete in a layered network. Then, we introduce mixed-integer linear programming formulations and discuss a polynomially solvable special case. Next we show a sufficient condition for the existence of a crossing in an optimal solution, which can be used for preprocessing the arcs in order to reduce the problem size. Our computational experiments on a large test set show that our preprocessing algorithm can significantly reduce the number of arcs.

Keywords: Network flows, layered networks, noncrossing flow, integer programming.

1. Introduction

The ordinary minimum cost flow problem (MCFP) is well-known and has widespread applications. It is also faced as a relaxed subproblem in solving many difficult combinatorial optimization problems [1]. Due to its special structure, it can be solved efficiently, and many polynomial-time algorithms have been developed ever since Ford and Fulkerson's seminal work [8].

In this work, we focus on the minimum cost noncrossing flow problem (MCNFP), which is an extension of the ordinary MCFP in layered networks with additional compatibility constraints in conjunction with the flow balance, capacity, lower bound, and binary restrictions. Layered graphs and networks provide effective modeling tools for the solution of some difficult combinatorial optimization problems, as recently detailed and classified in [12]. They are often encountered in container terminals, especially when the temporal allocation of the quay cranes to load/unload the berthed vessels according to their technical properties [14, 21], and when the scheduling of trains through the stations on the same railroad lines is targeted. In general, a layered network provides a graphical tool to model the scheduling of flow with spatial constraints. Compatibility constraints we consider belong to a special class named as conflict, disjunctive, or exclusionary side constraints. They make MCNFP a relative of graph and network based combinatorial optimization problems, which also include conflict constraints. For example, transportation problem with conflict constraints is studied in [19, 20], and [11], assignment problem with conflict

*Corresponding author. Phone: + 90 (212) 359 6467, Fax: + 90 (212) 265 1800

Email addresses: `altinel@boun.edu.tr` (İ. Kuban Altınel*), `arasn@boun.edu.tr` (Necati Aras), `zeynep.arslan@boun.edu.tr` (Zeynep Şuvak), `caner.taskin@boun.edu.tr` (Z. Caner Taşkın)

constraints is studied in [15], minimum spanning tree problem with conflict constraints is studied in [6, 22] and [17], the shortest path problem with conflict constraints is studied in [7], and lastly maximum flow problem with conflict constraints is studied in [16].

To the best of our knowledge the first four of the above mentioned studies are the most closely related ones to ours. The problems they consider can be seen similar to ours, since their particular network structure has two layers. In the first two, Sun [19] studies the transportation problem with conflict constraints (TPC), and proposes a Tabu Search heuristic. He also develops a branch-and-bound (BB) algorithm in [20]. The third one is more theoretical and provides a complexity analysis of TPC; it is shown that even some specific cases are NP-hard [11]. In the last one, the authors assume unit flow capacities and zero lower bounds in addition to the two-layer network structure, and propose efficient heuristics for the assignment problem with conflicts [15].

This paper consists of eight sections. We explain in the next section a motivating real-life application of MCNFP, i.e. the quay crane scheduling problem. We introduce the notation and terminology in Section 3, and analyze the computational difficulty of MCNFP in Section 4. We propose mixed-integer linear programming (MILP) formulations in Section 5, which is followed by Section 6, where we discuss a polynomially solvable realistic case. Section 7 includes results that lead to a preprocessing procedure used in reducing the size of the problem. The computational results for the effect of preprocessing on the performance of the formulations are reported in Section 8. Finally, concluding remarks are provided in Section 9.

2. Flow scheduling with spatial constraints

Three important problems associated with the management of seaside operations at container terminals are the berth allocation problem (BAP), crane assignment problem (CAP), and crane scheduling problem (CSP). Excellent surveys of the related works with a classification according to some specific attributes are provided in [2, 3, 4, 5].

In general, BAP deals with the determination of optimal berthing times and positions of vessels. It is possible to visualize a solution of BAP by means of a time-berth diagram where the y-axis represents the quay discretized in berth sections which vessels can occupy, and the x-axis represents time periods. A common assumption is that each berth section is just large enough to be occupied by only one quay crane. A sample time-berth diagram is given in Figure 2.1. There are four vessels and the rectangles represent the area they cover on the time-berth diagram. CAP finds the optimal number of cranes assigned to the vessels, and thus can be seen as a special form of the optimal crane splitting problem [18]. The numbers within the parenthesis in the rectangles are the crane numbers required per time period that guarantee the determined length of stay for the vessels, and form a solution of CAP. For example, vessel 2 stays berthed at berth sections $4-8$ for nine periods, and demands three cranes during periods $1-3$ and four cranes during periods $4-9$. CSP focuses on assigning quay cranes to optimal work places in each interval, given the berth locations of the vessels along the quay and number of the cranes that should serve them (i.e. the information displayed in Figure 2.1 provides) with the objective of minimizing the total setup cost due to crane relocations on the berth over the planning horizon. An interval is the time that elapses between two sequential events. An event is a specific vessel activity capable to cause a change in the number of serving cranes; it is an arrival or departure. More than one event can occur at the same time. This is also illustrated in Figure 2.1. There are three intervals. For example, the first one starts with the arrivals of vessels 1, 2, and 3 at time 0, and ends with the departures of vessels 1 and 3, and arrival of vessel 4 at time 3, which is also the starting time of the second interval. Observe that the number of cranes in service during an interval cannot be larger than seven, which is the total number of cranes available in the terminal.

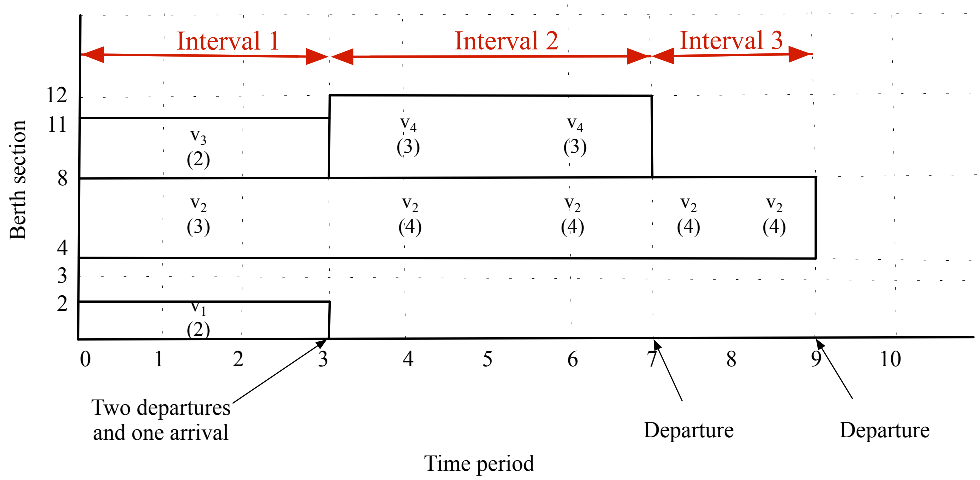


Figure 2.1: Sample berth allocations and quay crane assignments for five vessels

A layered network representation describing the sample situation explained above is given in Figure 2.2. This is a directed, layered, single source, and single sink network. The only vertex of the first layer is the super source, which represents the terminal's resources, with supply equal to the total number of available quay cranes. Similarly, the last layer consists of a single vertex. It is a super sink with demand equal to the total number of available quay cranes. The remaining vertices belong to internal layers and represent the vessels demanding cranes at each time interval. Except the first and last layers, each one of L layers represents a snapshot of the berth during a time interval. In other words, layer l exists in the network if a vessel arrives or departs changing the current snapshot; it is then followed by a new one. Notice that the number of cranes assigned to the berthed vessels can change only between consecutive layers, since only an arrival or a departure can cause such a change. These events and the intervals they represent are also depicted in Figure 2.1.

At each layer l, a berthed vessel is represented by a vertex, whose demand is equal to the number of assigned cranes. These vertices are called vessel vertices, and they are ordered according to the position of the vessels berthed along the quay from the beginning to the end. There is a second type of vertices below and above the vessel vertices. They are called parking vertices and represent the waiting area for the idle cranes. In any layer, the number of parking vertices is one more than the number of vessel vertices. To summarize, by letting n_l denote the number of berthed vessels at the quay during interval l, there are n_l vessel vertices and $n_l + 1$ parking vertices. Hence, the total number of vertices in layer l is $2n_l + 1$ and a vertex with an even index (i.e. $2, 4, \ldots, 2n_l$) correspond to a vessel vertex with a crane demand, while those with an odd index (i.e. $1, 3, \ldots, 2n_l + 1$) represent parking vertices with finite capacities.

The network of Figure 2.2 is incomplete for the MCNFP formulation. As can be easily observed, the total demand is not equal to the total supply. Besides, the odd vertices are capacitated and they have to be appropriately presented. For this purpose, except the super source s and super sink t, we replace each vertex by two vertices connected with an arc, which is a known transformation used for capacitated vertices. As a result, vessel vertices, which are originally transshipment sinks, are replaced with two pure transshipment vertices. Similarly, parking vertices, which are originally capacitated, are replaced with two uncapacitated pure transshipment vertices. The details of the transformation can be found in the work by [21]. The flows on the arcs of this network correspond to crane relocations or movements from parking areas to vessels, from vessels to vessels (this includes the case where a crane continues serving the same vessel or starts serving a new vessel without an idle period), from vessels to

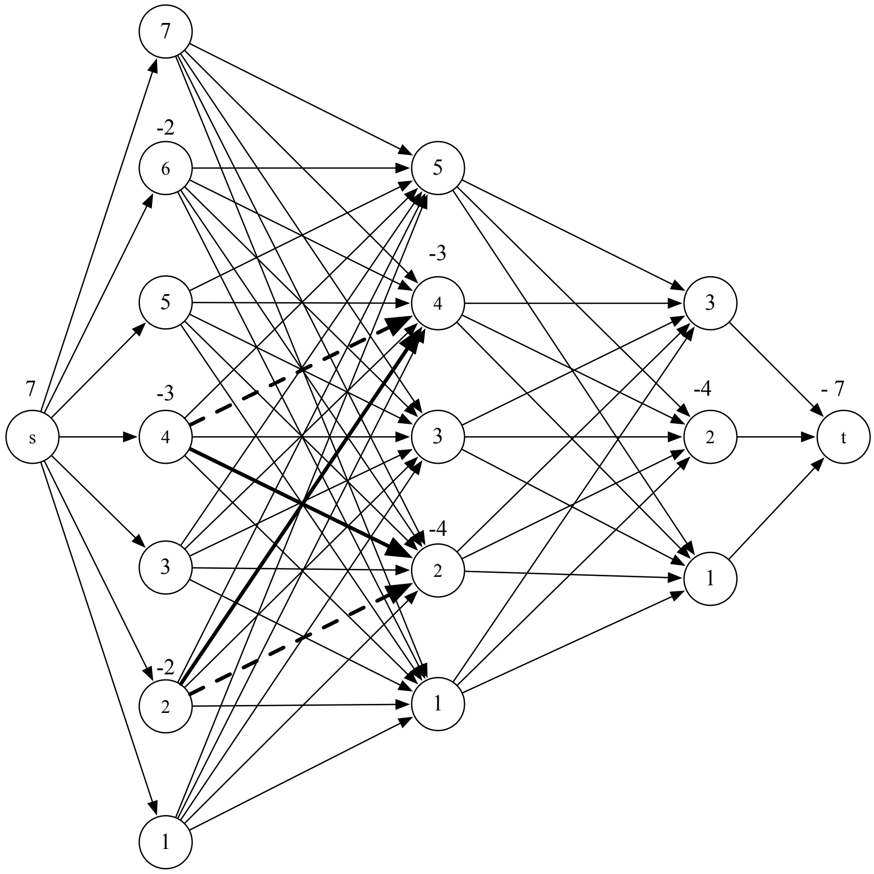


Figure 2.2: A layered network describing the situation given in Figure 2.1

parking areas in each time interval. The costs associated with these relocations are the unit flow costs.

CSP becomes an ordinary minimum cost flow problem (MCFP) on the described layered network if crane crossing is allowed. Unfortunately, this is not possible in reality; quay cranes are restricted to move on a rail and thus the relocation paths cannot cross. In other words, it is not enough to solve the MCFP on the described layered network to determine an optimal crane schedule for the example of Figure 2.1, since, for example, an optimal solution can include flows on arcs $(2, 4)$ and $(4, 2)$ between the second and third layers depicted in Figure 2.2. As a consequence, it is possible to say that CSP is in fact equivalent to a MCFP with additional spatial constraints allowing only noncrossing arcs to have positive flow values in an optimal solution, which makes it a particular subclass of MCNFP.

MCNFP is a generalization of the CSP where we consider a flow problem with suppliers and customers located on a line. The commodity flow is realized by means of vehicles, which are restricted to move along a single track lane, and hence cannot pass each other as a spatial restriction. Besides, suppliers and customers have time varying operating characteristics. At a given time, some of them can leave and/or new ones can arrive, and can change their supplies/demands. The purpose is to determine an optimal commodity flow schedule between them so that, the total distribution cost is minimized. We will consider this generalization in the rest of this work.

3. Notation and terminology

Let $N = (V(N), A(N))$ be a layered network consisting of L layers defined by the sets $V(N)$ of vertices and $A(N)$ of arcs. We define $V_l(N)$ as the set of vertices of layer l and n_l its cardinality (i.e. $n_l = |V_l(N)|$, $l = 1, 2, \ldots, L$), and assume that $V_1(N) = \{s\}$, $V_L(N) = \{t\}$, $V(N) = \cup_{l=1}^{L} V_l(N)$, $n_1 = n_L = 1$ and $n = |V(N)| = \sum_{l=1}^{L} n_l$. s has only outarcs and t has only inarcs. Any arc $(i, j) \in A(N)$ of the network is forward (i.e. tail is closer to s in the number of arcs). There are neither backward arcs, nor arcs connecting two vertices at the same layer. If we let $A_{l(l+1)}(N)$ be the set of arcs (i, j) having $i \in V_l(N)$ and $j \in V_{l+1}(N)$, then $A(N) = \cup_{l=1}^{L-1} A_{l(l+1)}(N)$. We also assume that $A_{l(l+1)}(N)$ consists of all possible arcs with tail in $V_l(N)$ and heads in $V_{l+1}(N)$; i.e. $A_{l(l+1)}(N) = \{(i, j) : i \in V_l(N), j \in V_{l+1}(N)\}$. We consider a particular embedding of the network for vertex labeling: the vertices are located on the intersection points of a grid where the vertical lines represent the layers and numbered from 1 to n_l at layer l starting from the bottom to the top. As a consequence, if two arcs $(i_1, j_1), (i_2, j_2) \in A_{l(l+1)}(N)$ *cross*, then $i_1 > i_2$ and $j_1 < j_2$ and vice versa; they form a crossing. Observe that arcs $(s, j) \in A_{12}(N)$ as well as arcs $(i, t) \in A_{(L-1)L}(N)$ are noncrossing. Any pair of paths with at least two distinct arcs that cross each other are said to be crossing. Notice that crossing paths are not necessarily arc or vertex disjoint. Any disjoint pair of paths may cross, and any two noncrossing paths may share arcs or vertices. The described layered network structure is illustrated in Figure 3.1.

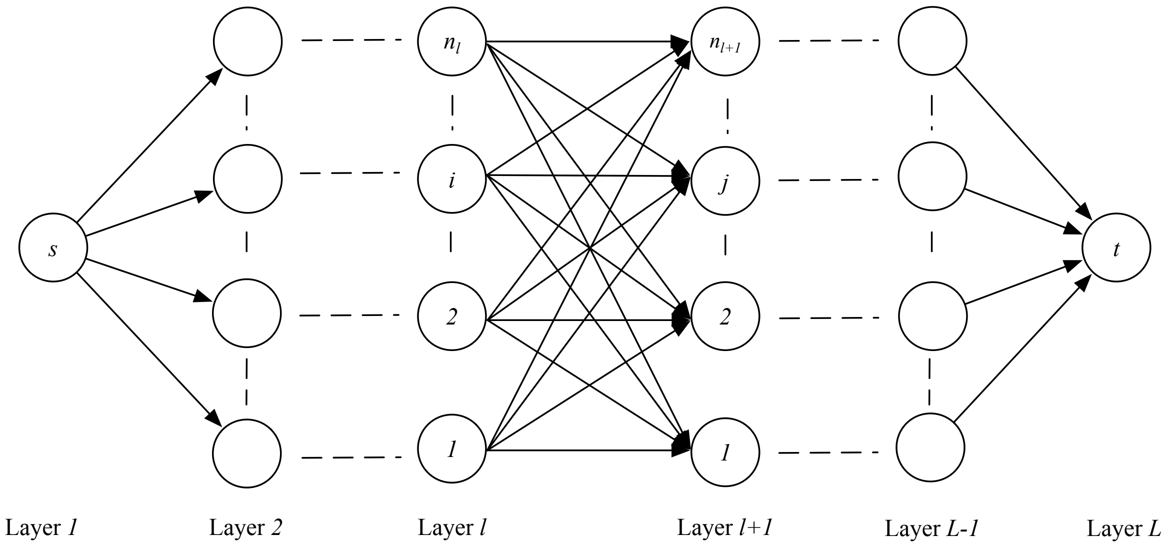
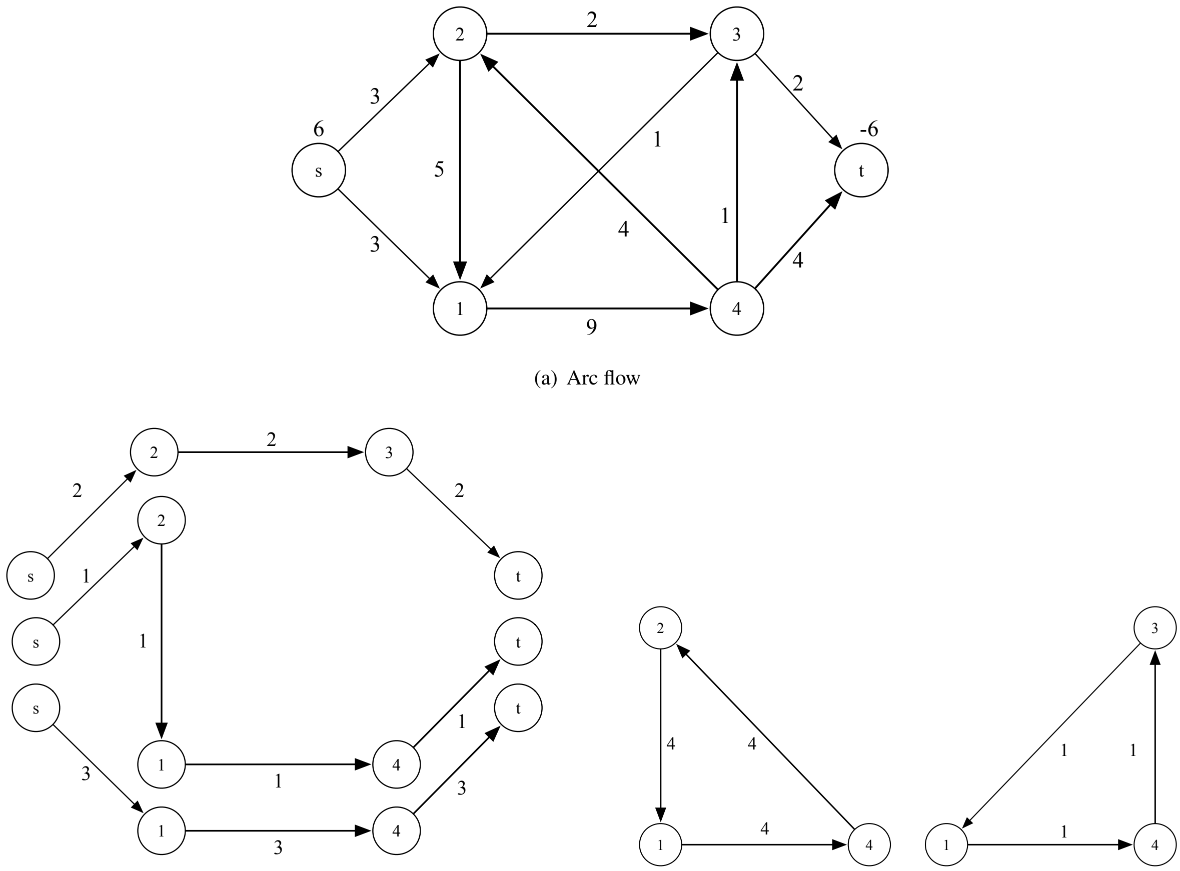


Figure 3.1: A layered network with L layers

Each arc $(i, j) \in A(N)$ has an associated unit flow cost c_{ij}. We also associate with each arc $(i, j) \in A(N)$ a capacity u_{ij} that denotes the maximum amount of flow allowed on arc (i, j) and a lower bound l_{ij} that denotes the minimum amount that must flow on arc (i, j). Each vertex $i \in V(N)$ has a number b_i representing its supply/demand. If $b_i > 0$, vertex i is a transshipment supply vertex, if $b_i < 0$ vertex i is a transshipment demand vertex with a demand of $-b_i$, and if $b_i = 0$ vertex i is a pure transshipment vertex. In other words, vertex set $V(N)$ can be expressed as $V(N) = \{s, t\} \cup V^+(N) \cup V^-(N) \cup V^{\pm}(N)$, where $V^+(N)$, $V^-(N)$ and $V^{\pm}(N)$ are respectively the subsets of transshipment supply, transshipment demand, and pure transshipment vertices at layer l. Similarly, for every level l, $V_l(N) = V_l^-(N) \cup V_l^+(N) \cup V_l^{\pm}(N)$, where $V_l^+(N)$, $V_l^-(N)$ and $V_l^{\pm}(N)$ denoting the subsets of transshipment supply, transshipment demand and pure transshipment vertices of layer l. Clearly, $V_1^-(N) = V_1^{\pm}(N) = \emptyset$, $V_1^+(N) = V_1(N) = \{s\}$, and $V_L^+(N) = V_L^{\pm}(N) = \emptyset$, $V_L^-(N) = V_L(N) = \{t\}$. We assume that $\sum_{l=1}^{L} \sum_{i \in V_l(N)} b_i = 0$, and $l_{ij} = 0 \le u_{ij}$ for all $(i, j) \in A(N)$ and they satisfy sufficient conditions for the existence of a feasible flow [1].

The function $\mathbf{f} : A(N) \to \mathbb{R}$ is the flow function and associates the variable f_{ij} with arc (i, j). In the ordinary MCFP the goal is to determine a feasible flow with the minimum total cost. Recall that a flow $\mathbf{f}$ is feasible if it satisfies flow balance equalities at the vertices, lower and upper bounds on the arcs. In the classical theory, network



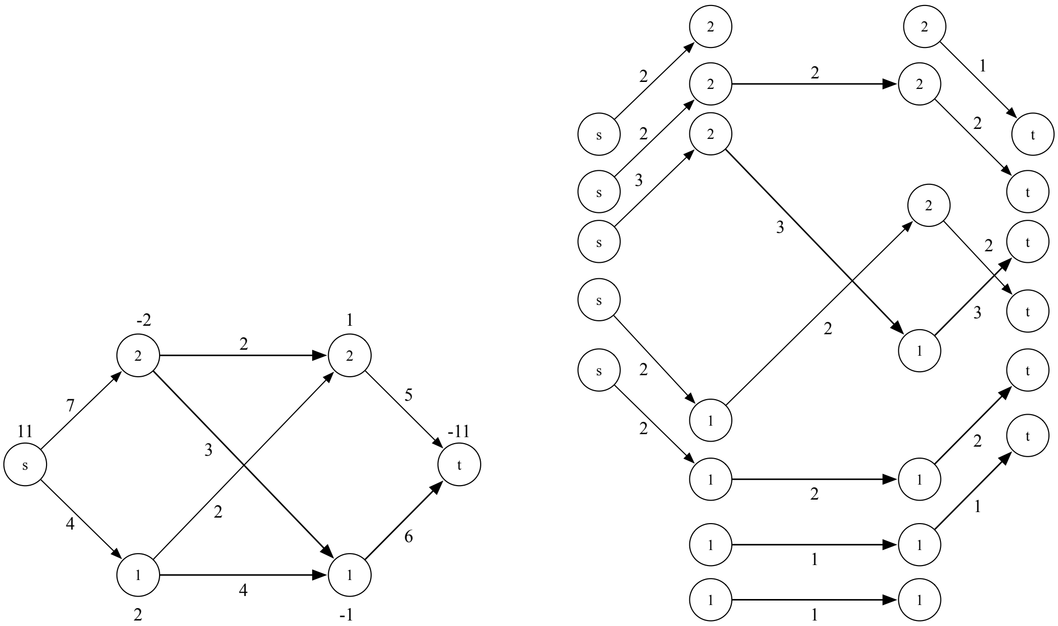
(a) Arc flow

(b) Flow paths of an equivalent path and circuit flow

(c) Flow circuits of an equivalent path and circuit flow

Figure 3.2: Two ways to express a flow in a network

flow problems can be equivalently formulated by either defining flows on arcs (i.e. arc flow) or directed paths and circuits (i.e. path and circuit flow). This is a consequence of the flow decomposition theorem [8], which eventually enables the (unique) representation of a path and circuit flow as nonnegative arc flow, and (not necessarily unique) representation of a nonnegative arc flow as a path and circuit flow. An example is provided in Figure 3.2. Notice that there is always a flow path connecting a source vertex to a sink vertex. In relation to the flow problem we consider in this work, namely MCNFP, there is no circuit involved in this decomposition because of the (directed) layered structure of the network, and thus an arc flow can be represented as a path flow, and vice-versa. Figure 3.3 provides an example for this particular situation. Then it is possible to say that an (arc) flow on $N = (V(N), A(N))$ is *noncrossing* if and only if all paths of the equivalent path flows are noncrossing, since an arc (i, j) with positive flow (i.e. $f_{ij} > 0$) appears on at least one path with positive flow on it, and if it is crossed by another arc, then there exists another path having positive flow on it with an arc crossing arc (i, j). For example, in Figure 3.3, the path $s \to 2 \to 1 \to t$ and $s \to 1 \to 2 \to t$ are crossing since they both have positive flows (i.e. 3 and 2 units of flows respectively) and arcs $(2, 1)$ and $(1, 2)$ are crossing. In short, we can refer to MCNFP as the MCFP with noncrossing flow paths, and a flow path is a directed path with positive flow on it. For the MCNFP, the directed paths from a source vertex to a demand vertex with positive flow on its arcs, namely flow paths, must be noncrossing in addition to flow balance, lower and upper bound restrictions in order to be feasible. We say such flow is feasible and also noncrossing. In other words, an optimal solution of MCNFP is a *noncrossing flow* with the minimum total cost, which is feasible with respect to the mentioned balance, lower and upper bound restrictions. Clearly, MCFP is the relaxation of MCNFP obtained by relaxing spatial *compatibility restrictions* that do not allow arc crossings on the flow paths.



(a) Arc flow

(b) An equivalent path flow

Figure 3.3: Two ways to express a flow in a layered network

4. The difficulty of the minimum cost noncrossing flow problem

We first define the decision problems associated with the MCNFP and its variant with restricted total flow costs for the flow paths (MCNFP-RC) in the following.

MCNFP Instance: A layered network $N = (V(N), A(N))$ with $L \in \mathbb{Z}_+$ layers each of which having $n_l \in \mathbb{Z}_+$ vertices $l = 1, 2, \ldots, L$ except the first and last ones: they consist of single vertices, namely a source s for layer $l = 1$ and a sink t for layer $l = L$. There is a supply/demand $b_i \in \mathbb{Z}_+$ for every vertex $i \in V(N)$ satisfying $\sum_{i \in V^+(N)} b_i = \sum_{i \in V^-(N)} b_i$. For each arc $(i, j) \in A(N)$ there is a capacity $u_{ij} \in \mathbb{Z}_+$, and unit flow cost $c_{ij} \in \mathbb{Z}_+$. There is also a given number $C \in \mathbb{Z}_+$.

Question: Is there a noncrossing arc flow with total cost less than C?

MCNFP-RC Instance: The same as the MCNFP instance.

Question: Is there a noncrossing arc flow with total cost less than C for every flow path?

Proposition 1. *MCNFP-RC is NP-complete for*

$$b_{i_l} = \begin{cases} B, & \text{if } l = 1 \text{ (i.e. } i_1 = s) \\ -B, & \text{if } l = L \text{ (i.e. } i_L = t) \\ 0, & \text{otherwise,} \end{cases}$$

with $B \in \mathbb{Z}_+$.

Proof. i. MCNFP-RC $\in$ *NP*: If a flow is given, its feasibility can be checked in polynomial time. Checking the feasibility of the flow with respect to the flow balance constraints and bounds can be done in $O(|A(N)|)$ time.

Checking whether there is a crossing requires at most $O(|V(N)|^2 L)$ time. As a consequence of the flow decomposition theorem [8] given a nonnegative arc flow it is possible to generate all flow paths in $O(|V(N)| + |A(N)|)$ time and the number of flow paths is $O(|V(N)| + |A(N)|)$ in the worst case. Thus, checking whether or not the total cost of each path is less than C takes $O((|V(N)| + |A(N)|)|A(N)|) = O(|A(N)|^2)$ time. Therefore, there is a polynomial time certificate checking algorithm and MCNFP-RC $\in NP$.

ii. MCNFP-RC is hard (Reduction from the set partitioning problem): The *Set Partitioning Problem* (SPP), which is known to be NP-complete [9], reduces polynomially to MCNFP-RC. The SPP deals with the following question: Given a set S of V elements with values $s_v \in \mathbb{Z}_+$, $v = 1, 2, \ldots, V$ and $\sum_{v \in S} s_v = D$, is there a subset $S' \subset S$ such that $\sum_{v \in S'} s_v = \sum_{v \in S \setminus S'} s_v = \frac{D}{2}$?

An instance of MCNFP-RC corresponding to an arbitrary SPP instance can be the complete layered network $N = (V(N), A(N))$ with

a. $L = V + 3$ layers,

b. $n_l = 2, l = 2, 3, \ldots, L - 1; n_1 = n_L = 1$ vertices at each layer having a supply/demand

$$b_{i_l} = \begin{cases} 2, & \text{if } l = 1 \text{ (i.e. } i_1 = s) \\ -2, & \text{if } l = L \text{ (i.e. } i_L = t) \\ 0, & \text{otherwise} \end{cases}$$

c. $C = D + (V + 1)$,

d. unit flow cost

$$c_{i_l i_{l+1}} = \begin{cases} \frac{D}{2}, & \text{if } l = 1; i_{l+1} = 1, 2 \\ 1, & \text{if } l = L - 1; i_{L-1} = 1, 2 \\ 1, & \text{if } l = 2, 3, \ldots, L - 1; i_l = 1, i_{l+1} = 2 \\ 1, & \text{if } l = 2, 3, \ldots, L - 1; i_l = 2, i_{l+1} = 1 \\ s_{l-1} + 1, & \text{if } l = 2, 3, \ldots, L - 1; i_l = i_{l+1} = 1 \\ s_{l-1} + 1, & \text{if } l = 2, 3, \ldots, L - 1; i_l = i_{l+1} = 2 \end{cases}$$

e. lower bounds $l_{i_l i_{l+1}} = 0$ and capacities

$$u_{i_l i_{l+1}} = \begin{cases} 1, & \text{if } l = 2, 3, \ldots, L - 1; i_l = 1, i_{l+1} = 2 \\ 1, & \text{if } l = 2, 3, \ldots, L - 1; i_l = 2, i_{l+1} = 1 \\ u_{i_l i_{l+1}} \in \mathbb{Z}_+, & \text{otherwise,} \end{cases}$$

Figure 4.1 illustrates the network obtained after this transformation for $S = \{s_1, s_2, s_3\}$, $V = 3$, $S' = \{s_1, s_3\}$, $L = 3 + 3 = 6$. The number on the arcs are the unit costs. For vertex numbering we use the previously mentioned convention. Two noncrossing flow paths satisfying the total flow cost restriction $C = D + 4$ are presented using dashed arcs. Observe that the paths have unit flow on them, satisfy balance equalities, lower bound and capacity restrictions, and cost restrictions. Furthermore, they are noncrossing. The first of the two paths presents subset S' (path $1 \to 2 \to 1 \to 1 \to 2 \to 1$) and the second one the subset $S \setminus S'$ (path $1 \to 1 \to 1 \to 2 \to 2 \to 1$). They

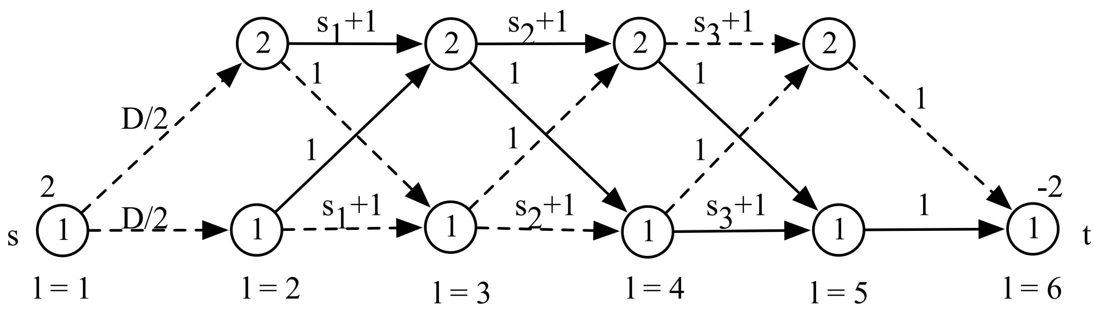


Figure 4.1: Noncrossing paths corresponding to an SPP instance

are not necessarily disjoint; arc $(2,1)$ between layers 5 and 6 is traversed by both paths. The noncrossing flow path representation of sets S' and S is not unique. They can be represented using two other paths as well. For example path $1 \to 2 \to 2 \to 1 \to 1 \to 1$ for S' and path $1 \to 1 \to 2 \to 2 \to 1 \to 1$ for $S \backslash S'$. Notice that, this time arc $(1,1)$ between layer 5 and 6 is on both paths. What must be done now is to show that S has a subset S' such that $\sum_{v \in S'} s_v = \sum_{v \in S \backslash S'} s_v = \frac{D}{2}$ if and only if there is a feasible flow with noncrossing flow paths each having at most $C = D + (V+1)$ total flow cost.

First, suppose that S has a subset S' such that $\sum_{v \in S'} s_v = \sum_{v \in S \backslash S'} s_v = \frac{D}{2}$. Then, it is possible to generalize the path structure of Figure 4.1 so that the first path includes the elements of S' and the second path the elements of $S \backslash S'$. As can be noticed, these flow paths are noncrossing, satisfy flow balance equalities, lower bounds and capacity restrictions, and each has a total flow cost $D + (V+1)$ (i.e. $\frac{D}{2} + \sum_{v \in S'} s_v + (V+1) = D + (V+1) = \frac{D}{2} + \sum_{v \in S \backslash S'} s_v + (V+1)$). Thus, if the set S has a subset S' such that $\sum_{v \in S'} s_v = \sum_{v \in S/S'} s_v = \frac{D}{2}$, it is possible to construct two noncrossing flow paths each with a total flow cost equal to $D + (V+1)$.

Conversely, suppose that we are given a flow feasible with respect to flow balance equalities, lower bounds, capacity restrictions and having only noncrossing flow paths each with a cost less than $D + (V+1)$. First of all for the described MCNFP-RC instance there can be at most two s,t-flow paths since exactly two units of flow has to be sent out of source s. Single s,t-flow path (with two units of flow on it) is not possible because the total cost of the one with the smallest total cost is $D + 2(V+1)$, which is larger than the restriction $D + (V+1)$. Hence, two distinct flow paths have to start at source s. Besides, they must satisfy the cost restrictions (i.e. each has a total cost of at most $D + (V+1)$).

Consider the arcs (i_l, i_{l+1}) such that $i_l = i_{l+1} = 1, 2$ for $l = 2, 3, \ldots, L-1$. This is the pair of arcs with costs $s_v + 1$, $v \in S$. Then, at least one of these two arcs $i_l = i_{l+1} = 1, 2$ must appear on one of these two paths for each $l = 2, 3, \ldots, L-1$. Otherwise, there is a crossing because of the network structure and unit upper bounds on arcs (i_l, i_{l+1}) with $i_l = 1$ and $i_{l+1} = 2$, and $i_l = 2$ and $i_{l+1} = 1$ for $l = 2, 3, \ldots, L-1$ (i.e. in case there is one which is missing on both paths) or one of the paths has cost larger than $D + (V+1)$ (i.e. one of them can be traversed by both paths), which is a contradiction. In short, there are two flow paths each having unit flow on it and partitioning the arcs with costs $s_v + 1$ $v \in S$ (i.e. these arcs appear exactly on one of them) and thus the sum of the total costs is equal to $2\frac{D}{2} + \sum_{v \in S} s_v + 2(V+1) = 2(D + (V+1))$. This implies that each flow path satisfies fully its total cost restriction $D + (V+1)$, since each has a total flow cost less than $D + (V+1)$ with grand total exactly equal to $2(D + (V+1))$. Finally, one of the flow paths cannot include all of them (i.e. the set S entirely) because this results in a total cost of $\frac{D}{2} + \sum_{v \in S} s_v + (V+1) = \frac{3}{2}D + (V+1) > D + (V+1)$. Let S be the set of these arcs and S' be its subset included in the first path. Then, other path would traverse the arcs in S'/S. Recall that each one of these paths has total cost $D + (V + 2)$. Therefore, $\frac{D}{2} + \sum_{v\in S'} s_v + (V + 1) = D + (V + 1)$ and $\frac{D}{2} + \sum_{v\in S\setminus S'} s_v + (V + 1) = D + (V + 1)$, which implies that $\sum_{v\in S'} s_v = \sum_{v\in S\setminus S'} s_v = D - \frac{D}{2} = \frac{D}{2}$. This transformation can be done in $O(V)$ time. □

Proposition 2. *MCNFP is NP-complete when*

$$b_{i_l} = \begin{cases} B', & \text{if } l = 1 \text{ (i.e. } i_1 = s) \\ -B', & \text{if } l = L \text{ (i.e. } i_L = t) \\ 0, & \text{otherwise,} \end{cases}$$

with $B' \in \mathbb{Z}_+$.

Proof. i. MCNFP $\in NP$: First of all any certificate of MCNFP can be checked in polynomial time similar to MCNFP-RC. Hence, MCNFP $\in NP$.

ii. MCNFP is hard (Reduction from MCNFP-RC): Consider any arbitrary instance of MCNFP-RC with unit costs $c_{ij} \in \mathbb{Z}_+$, $(i, j) \in A(N)$, supply/demand $b_i \in \mathbb{Z}_+$, $i \in V(N)$ and capacities $u_{ij} \in \mathbb{Z}_+$, $(i, j) \in A(N)$ as assumed in the assertion, and cost restrictions $D \in \mathbb{Z}_+$ for the flow paths. To generate a particular MCNFP instance we keep the same layered network structure of a MCNFP-RC instance, but modify unit costs, arc capacities, supplies and demands.

We choose

$$b'_{i_l} = \begin{cases} B', & \text{if } l = 1 \text{ (i.e. } i_1 = s) \\ -B', & \text{if } l = L \text{ (i.e. } i_L = t) \\ 0, & \text{otherwise,} \end{cases}$$

with

$$B' = B + \left\lceil \frac{1}{(L-1)} \right\rceil = B + 1.$$

We set the unit costs all equal to λ', arc capacities to $u'_{ij} = u_{ij} + 1$, and the restriction $C = \lambda'(L-1)B'$. Here $\lambda' \in \mathbb{Z}_+$ is larger enough than BD.

First suppose that MCNFP-RC has a noncrossing flow $\mathbf{f} \in \mathbb{Q}_+^{|A(N)|}$ satisfying flow balance, arc capacity constraints and cost restrictions on the flow paths $P_k = (V(P_k), A(P_k))$ $k = 1, 2, \ldots, K$. Here, K is the number of noncrossing flow paths and $K \leq |V(N)| + |A(N)|$ as a conseqeunce of the flow decomposition theorem [8]. However, for this particular case, due to the integrality of the flow and network structure $K \leq B$.

Since each flow path satisfies the cost restrictions,

$$\sum_{(i,j)\in A(P_k)} c_{ij} f_k \leq D \qquad k = 1, 2, \ldots, K,$$

and consequently

$$\sum_{k=1}^{K} \sum_{(i,j)\in A(P_k)} c_{ij} f_k \leq KD.$$

Here, f_k is the amount of positive flow sent through the kth flow path. Then,

$$\sum_{(i,j)\in A(N)} c_{ij} f_{ij} \leq KD \tag{1}$$

follows, since $f_{ij} = \sum_{k=1}^{K} \sum_{\{P_k:(i,j)\in A(P_k)\}} f_k$, where f_{ij} is the flow on arc (i, j).

At this point we have to show the following claim.

Claim 1.

$$\sum_{k=1}^{K} \sum_{(i,j)\in A(P_k)} f_k = (L-1)B$$

Proof. First of all, $\sum_{k=1}^{K} \sum_{(i,j)\in A(P_k)} f_k = \sum_{k=1}^{K} |A(P_k)| f_k$. because of the special structure of the network $N = (V(N), A(N))$ (i.e. layered network with forward arcs) every feasible arc flow can be represented as a path flow having exactly $L-1$ arcs. Besides, each flow path with positive flow on it connects a source vertex to a sink vertex [8]. Hence,

$$\sum_{k=1}^{K} \sum_{(i,j)\in A(P_k)} f_k = \sum_{k=1}^{K} |A(P_k)| f_k = (L-1) \sum_{k=1}^{K} f_k.$$

In addition, the sum of the flows over the flow paths is equal to the sum of supplies, which is equal to the negative of the sum of the demands, namely to B. In other words $\sum_{k=1}^{K} f_k = \sum_{i\in V^+(N)} b_i = -\sum_{i\in V^-(N)} b_i = B$, which completes the proof. □

Then, for unit costs $c'_{ij} = c_{ij} + \lambda_{ij}$ with $\lambda_{ij} \in \mathbb{Z}_+$, $\overline{\lambda} = \max_{(i,j)\in A(N)}\{\lambda_{ij}\} = BD$ and $\underline{\lambda} = \min_{(i,j)\in A(N)}\{\lambda_{ij}\}$ the total flow cost becomes

$$\begin{aligned}
\sum_{(i,j)\in A(N)} c'_{ij} f_{ij} &= \sum_{(i,j)\in A(N)} c_{ij} f_{ij} + \sum_{(i,j)\in A(N)} \lambda_{ij} f_{ij} \\
&= \sum_{k=1}^{K} \sum_{(i,j)\in A(P_k)} c_{ij} f_k + \sum_{k=1}^{K} \sum_{(i,j)\in A(P_k)} \lambda_{ij} f_k \\
&\leq KD + \overline{\lambda} \sum_{k=1}^{K} \sum_{(i,j)\in A(P_k)} f_k \\
&= KD + \overline{\lambda}(L-1)B \\
&\leq BD + \overline{\lambda}(L-1)B \\
&= \overline{\lambda}[(L-1)B + 1] \\
&= \overline{\lambda}(L-1)\left(B + \frac{1}{L-1}\right) \\
&\leq \overline{\lambda}(L-1)(B+1) \\
&= \overline{\lambda}(L-1)B' \\
&\leq \lambda'(L-1)B'.
\end{aligned}$$

The second term of the fourth expression follows from the second term of the third expression as consequence of claim 1. The fifth expression follows from the fourth since $B \geq K$. We also use the definition $\overline{\lambda} = BD$ and the fact that $\lambda' \geq \overline{\lambda}$. Notice that,

$$C = \lambda'(L-1)B' = \lambda' \sum_{k=1}^{K} \sum_{(i,j)\in A(P_k)} f'_k.$$

In other words this upper bound C is the total cost of an arc flow $\mathbf{f'}$, with path flow f'_k $k = 1, 2, \ldots, K$ on the flow paths P_k $k = 1, 2, \ldots, K$, for the same network structure with B' and u'_{ij} as defined previously, and unit flow costs set to λ'. Notice that it is possible to obtain $\mathbf{f'}$ by increasing the flow $\mathbf{f}$ on one of the flow paths P_k $k = 1, 2, \ldots, K$, say f_p on path P_p by one unit and keeping the remaining ones the same, i.e. by setting $f'_p = f_p + 1$, $f'_k = f_k$ for $k \neq p$. This is a feasible solution of the particular MCNFP instance we have created, i.e. a noncrossing flow feasible with respect to the flow balance equalities and capacity restrictions, with total cost equal to C.

Conversely, suppose that the particular MCNFP instance has a noncrossing flow with total cost not larger than C. Let $P'_k = (V(P'_k), A(P'_k))$, $k = 1, 2, \ldots, K'$ be the corresponding K' flow paths of a feasible flow $\mathbf{f}'$ of the particular MCNFP instance, which also satisfies total cost restriction. Let also f'_k $k = 1, 2, \ldots, K'$ be the path flow corresponding to these K'flow paths. Hence,

$$
\begin{aligned}
C = \lambda'(L-1)B' &= \lambda' \sum_{k=1}^{K'} \sum_{(i,j)\in A(P'_k)} f'_k \\
&\geq \sum_{k=1}^{K'} \sum_{(i,j)\in A(P'_k)} c'_{ij} f'_k \\
&= \sum_{(i,j)\in A(N)} c'_{ij} f'_{ij} \\
&= \sum_{(i,j)\in A(N)} c_{ij} f'_{ij} + \sum_{(i,j)\in A(N)} \lambda_{ij} f'_{ij} \\
&= \sum_{k=1}^{K'} \sum_{(i,j)\in A(P'_k)} c_{ij} f'_k + \sum_{k=1}^{K'} \sum_{(i,j)\in A(P'_k)} \lambda_{ij} f'_k \\
&\geq \sum_{k=1}^{K'} \sum_{(i,j)\in A(P'_k)} c_{ij} f_k + \sum_{k=1}^{K'} \sum_{(i,j)\in A(P'_k)} \lambda_{ij} f'_k \\
&\geq \sum_{k=1}^{K'} \sum_{(i,j)\in A(P'_k)} c_{ij} f_k + \underline{\lambda} \sum_{k=1}^{K'} \sum_{(i,j)\in A(P'_k)} f'_k \\
&= \sum_{k=1}^{K'} \sum_{(i,j)\in A(P'_k)} c_{ij} f_k + \underline{\lambda}(L-1)B'.
\end{aligned}
$$

The first inequality is a consequence of our selection of λ'. For example setting $\lambda' = \overline{c}'$ with $\overline{c}' = \max_{(i,j)\in A(N)}\{c'_{ij}\}$ is a possibility. The last equality is a consequence of claim 1, since it can be shown that $\sum_{k=1}^{K'} \sum_{(i,j)\in A(P'_k)} f'_k = (L-1)B'$ similarly. Hence, we can write

$$
\sum_{k=1}^{K'} \sum_{(i,j)\in A(P'_k)} c_{ij} f_k \leq C - \underline{\lambda}(L-1)B' = \lambda'(L-1)B' - \underline{\lambda}(L-1)B' = (L-1)B'(\lambda' - \underline{\lambda}),
$$

which becomes

$$
\sum_{k=1}^{K'} \sum_{(i,j)\in A(P'_k)} c_{ij} f_k \leq D
$$

after setting

$$
\underline{\lambda} = \lambda' - \left\lfloor \frac{D}{(L-1)B'} \right\rfloor.
$$

For example for $\lambda' = \overline{c}'$ and $\overline{\lambda} = BD$ it is possible to set

$$
\underline{\lambda} = \overline{c}' - \left\lfloor \frac{D}{(L-1)B'} \right\rfloor,
$$

provided that

$$\overline{c}' \leq BD + \left\lfloor \frac{D}{(L-1)B'} \right\rfloor$$

in order to have $\overline{\lambda} \geq \underline{\lambda}$, which makes $C - \underline{\lambda}(L-1)B' \leq D$. Also for $\overline{c}' = \lceil \alpha\overline{c} + \overline{\lambda} \rceil$ with $\overline{c} = \max_{(i,j)\in A(N)}\{c_{ij}\}$,

$$\alpha = \frac{1}{(L-1)B'},$$

and $\overline{\lambda} = BD$ we have $\overline{c} \leq D$.

In short,

$$\sum_{(i,j)\in A(P'_k)} c_{ij} f'_k \leq D \qquad k = 1, 2, \ldots, K'$$

follows, since $c_{ij} \geq 0$ and $f'_k > 0$ $(i, j) \in A(P'_k)$, implying $\sum_{(i,j)\in A(P'_k)} c_{ij} f'_k \geq 0$, $k = 1, 2, \ldots, K'$. Therefore, it is possible to obtain a feasible solution $\mathbf{f}$ of MCNFP-RC using the flow paths of the noncrossing arc flow $\mathbf{f}'$ by simply decreasing the flow on one of the flow paths P'_k $k = 1, 2, \ldots, K'$, say f'_p on path P'_p by one unit and keeping the remaining ones the same, i.e. by setting $f_p = f'_p - 1$, $f_k = f'_k$ for $k \neq p$. Finally, this transformation can be done in $\sum_{l=1}^{L-1}(n_l n_{l+1}) + \sum_{l=1}^{L} n_l = O(|V(N)|^2 + |V(N)|)$, which is polynomial and the proof is complete. □

The next two propositions follow directly form Proposition 1 and Proposition 2.

Proposition 3. *MCNFP-RC is NP-complete for general demand supply/demand, i.e.* $b_i \in \mathbb{Z}_+$ *for every vertex* $i \in V(N)$ *satisfying* $\sum_{i\in V^+(N)} b_i = \sum_{i\in V^-(N)} b_i$.

Proof. In Proposition 1 we have shown that a restriction of MCNFP-RC is NP-complete. It is obtained by setting

$$b_{i_l} = \begin{cases} B, & \text{if } l = 1 \text{ (i.e. } i_1 = s) \\ -B, & \text{if } l = L \text{ (i.e. } i_L = t) \\ 0, & \text{otherwise,} \end{cases}$$

with $B \in \mathbb{Z}_+$. □

Proposition 4. *MCNFP is NP-complete for general demand supply/demand, i.e.* $b'_i \in \mathbb{Z}_+$ *for every vertex* $i \in V(N)$ *satisfying* $\sum_{i\in V^+(N)} b'_i = \sum_{i\in V^-(N)} b'_i$.

Proof. In Proposition 2 we have shown that a restriction of MCNFP is NP-complete. It is obtained by setting

$$b'_{i_l} = \begin{cases} B', & \text{if } l = 1 \text{ (i.e. } i_1 = s) \\ -B', & \text{if } l = L \text{ (i.e. } i_L = t) \\ 0, & \text{otherwise,} \end{cases}$$

with $B' \in \mathbb{Z}_+$. □

5. Formulations

It is possible to formulate MCNFP as a mixed-integer linear programming problem (MILP) by allowing only noncrossing arcs to have positive flows. In other words, for each arc $(i, j) \in A_{l(l+1)}(N)$, if there is a positive flow on (i, j), i.e. if $f_{ij} > 0$, then $f_{pq} = 0$ for all $(p, q) \in A_{l(l+1)}(N)$ with either $1 \leq p \leq i - 1$ and $j + 1 \leq q \leq n_{l+1}$, or $i + 1 \leq p \leq n_l$ and $1 \leq q \leq j - 1$. Obviously, $f_{ij} = 0$ if $f_{pq} > 0$ for one of such $(p, q) \in A_{l(l+1)}(N)$. In addition to the flow variables f_{ij} we introduce binary design variables $x_{ij} \in A_{l(l+1)}(N)$ to model this. x_{ij} is set to 1 if $f_{ij} > 0$. Besides, if $x_{ij} = 1$ then $f_{pq} = 0$ for all (p, q) such that either $1 \leq p \leq i - 1$ and $j + 1 \leq q \leq n_{l+1}$, or $i + 1 \leq p \leq n_l$ and $1 \leq q \leq j - 1$. This allows us to define a list S_{ij} of arcs incompatible (i.e. crossing) with arc $(i, j) \in A_{l(l+1)}(N)$ as

$$S_{ij} = \{(p, q) \in A_{l(l+1)}(N) : 1 \leq p \leq i - 1, j + 1 \leq q \leq n_{l+1}; i + 1 \leq p \leq n_l, 1 \leq q \leq j - 1\} \qquad l = 1, 2, \ldots, L. \tag{2}$$

Then, we obtain the following MILP fomulation for MCNFP.

$$\begin{aligned}
\text{MCNFP:} \quad \min \quad & \sum_{(s,j)\in A_{12}(N)} c_{sj} f_{sj} + \sum_{l=2}^{L-2} \sum_{(i,j)\in A_{l(l+1)}(N)} c_{ij} f_{ij} + \sum_{(i,t)\in A_{L-1L}(N)} c_{it} f_{it} && (3)\\
\text{s.t.} \quad & \sum_{(s,j)\in A_{12}(N)} f_{s,j} = b_s && (4)\\
& \sum_{(i,j)\in A_{l(l+1)}(N)} f_{ij} - \sum_{(j,i)\in A_{(l-1)l}(N)} f_{ji} = b_i \qquad i \in V_l(N);\ l = 2, 3, \ldots, L-1 && (5)\\
& - \sum_{(i,t)\in A_{(L-1)L}(N)} f_{it} = b_t && (6)\\
& 0 \leq f_{ij} \leq u_{ij} x_{ij} \qquad (i, j) \in A_{l(l+1)}(N);\ l = 1, 2, \ldots, L-1 && (7)\\
& x_{pq} + x_{ij} \leq 1 \qquad (p, q) \in S_{ij};\ (i, j) \in A_{l(l+1)}(N);\ l = 1, 2, \ldots, L-1 && (8)\\
& x_{ij} \in \{0, 1\} \qquad (i, j) \in A_{l(l+1)}(N);\ l = 1, 2, \ldots, L-1. && (9)
\end{aligned}$$

Without constraints (8) and (9), and with u_{ij} instead of $u_{ij}x_{ij}$ in constraints (7) the formulation is the one of ordinary minimum cost flow problem on the layered network illustrated in Figure 3.1. We call constraints (8) and (9) compatibility constraints; flow can only be sent through only noncrossing arcs.

Another equivalent formulation of MCNFP is obtained by replacing inequalities (8) with

$$\sum_{(p,q)\in S_{ij}} x_{pq} + |S_{ij}| x_{ij} \leq |S_{ij}| \qquad (i, j) \in A_{l(l+1)}(N);\ l = 1, 2, \ldots, L-1. \tag{10}$$

As can be noticed, this formulation gives a weaker LP bound since inequalities (10) are obtained by aggregating inequalities (8) over the list S_{ij} for each arc (i, j).

6. A polynomially solvable special case

Let us assume that the network $N = (V(N), A(N))$ is not only layered but also complete (i.e. all arcs between the vertices of layer l and $l + 1$ exist) and the unit costs are nonnegative, symmetric and additive for $i \neq j$, and

$c_{ij} = 0$ for $i = j$. Namely,

$$\begin{aligned} c_{ii} &= 0, & (11)\\ c_{ij} &\geq 0, & (12)\\ c_{ij} &= c_{ji}, & (13)\\ c_{ij} &= \sum_{k=i}^{j-1} c_{k(k+1)}. & (14) \end{aligned}$$

Notice that (14) is valid if $i < j$. Otherwise we can interchange the limits of the summation and apply (13) as a consequence of symmetry.

Recall that for a pair of crossing arcs (i_1, j_1) and (i_2, j_2), $i_1 < i_2$ and $j_2 < j_1$. Besides, there are six possible orderings of these four vertices according to the convention we use for numbering the vertices (i.e. vertex labels denote their orders from bottom in their layers):

i.	$i_1 < i_2 \leq j_2 < j_1$	**iii.**	$i_1 \leq j_2 < j_1 \leq i_2$	**v.**	$j_2 \leq i_1 < i_2 \leq j_1$
ii.	$j_2 < j_1 \leq i_1 < i_2$	**iv.**	$j_2 \leq i_1 < j_1 \leq i_2$	**vi.**	$i_1 \leq j_2 < i_2 \leq j_1$.

These six cases are illustrated in Figure 6.1 with six snapshots from two consecutive layers of a layered network. Horizontal lines represent the inequalities of the orderings. Strict inequalities are reflected with additional nodes below or underneath of the tail/head of the crossing arcs. Solid arcs represent the crossings, whereas dashed ones represent their compatible equivalents. Observe that, if the flow conservation is satisfied and there is one unit of flow on each one of the crossing (solid) arcs before the correction, there must be one unit of flow on the new (dashed) arcs and zero unit of flow on the crossing arcs in order to correct the crossing and guarantee flow balance equations at the same time. The next lemma shows such change does not increase total flow cost.

Proposition 5. *The unit correction cost is nonincreasing under assumptions* (11) – (14) *of the unit flow costs.*

Proof. We will evaluate the cost of one unit of flow on arcs (i_1, j_1) and (i_2, j_2) (i.e. $f_{i_1 j_1} = f_{i_2 j_2} = 1$ and $f_{i_2 j_1} = f_{i_1 j_2} = 0$) with the cost of one unit of flow on arcs (i_2, j_1) and (i_1, j_2) (i.e. $f_{i_1 j_1} = f_{i_2 j_2} = 0$ and $f_{i_2 j_1} = f_{i_1 j_2} = 1$), namely $c_{i_1 j_1} + c_{i_2 j_2}$ with $c_{i_2 j_1} + c_{i_1 j_2}$ for the six possible crossings.

$$\begin{array}{lllcl}
\textbf{i.} & i_1 < i_2 \leq j_2 < j_1: & c_{i_1 j_1} + c_{i_2 j_2} & = & c_{i_1 i_2} + c_{i_2 j_2} + c_{j_2 j_1} + c_{i_2 j_2} \\
 & & & = & c_{i_2 j_1} + c_{i_1 j_2} \\
\textbf{ii.} & j_2 < j_1 \leq i_1 < i_2: & c_{i_1 j_1} + c_{i_2 j_2} & = & c_{i_2 i_1} + c_{i_1 j_1} + c_{j_1 j_2} + c_{i_1 j_1} \\
 & & & = & c_{i_2 j_1} + c_{i_1 j_2} \\
\textbf{iii.} & i_1 \leq j_2 < j_1 \leq i_2: & c_{i_1 j_1} + c_{i_2 j_2} & = & c_{i_1 j_2} + c_{j_2 j_1} + c_{i_2 j_1} + c_{j_1 j_2} \\
 & & & = & c_{i_2 j_1} + c_{i_1 j_2} + 2c_{j_1 j_2} \\
\textbf{iv.} & j_2 \leq i_1 < j_1 \leq i_2: & c_{i_1 j_1} + c_{i_2 j_2} & = & c_{i_1 j_2} + c_{j_2 j_1} + c_{i_2 j_1} + c_{j_1 j_2} \\
 & & & = & c_{i_2 j_1} + c_{i_1 j_2} + 2c_{j_1 j_2} \\
\textbf{v.} & j_2 \leq i_1 < i_2 \leq j_1: & c_{i_1 j_1} + c_{i_2 j_2} & = & c_{i_1 i_2} + c_{i_2 j_1} + c_{i_2 i_1} + c_{i_1 j_2} \\
 & & & = & c_{i_2 j_1} + c_{i_1 j_2} + 2c_{i_1 i_2} \\
\textbf{vi.} & i_1 \leq j_2 < i_2 \leq j_1: & c_{i_1 j_1} + c_{i_2 j_2} & = & c_{i_1 j_2} + c_{j_2 i_2} + c_{i_2 j_1} + c_{i_2 j_2} \\
 & & & = & c_{i_2 j_1} + c_{i_1 j_2} + 2c_{i_2 j_2}.
\end{array}$$

□

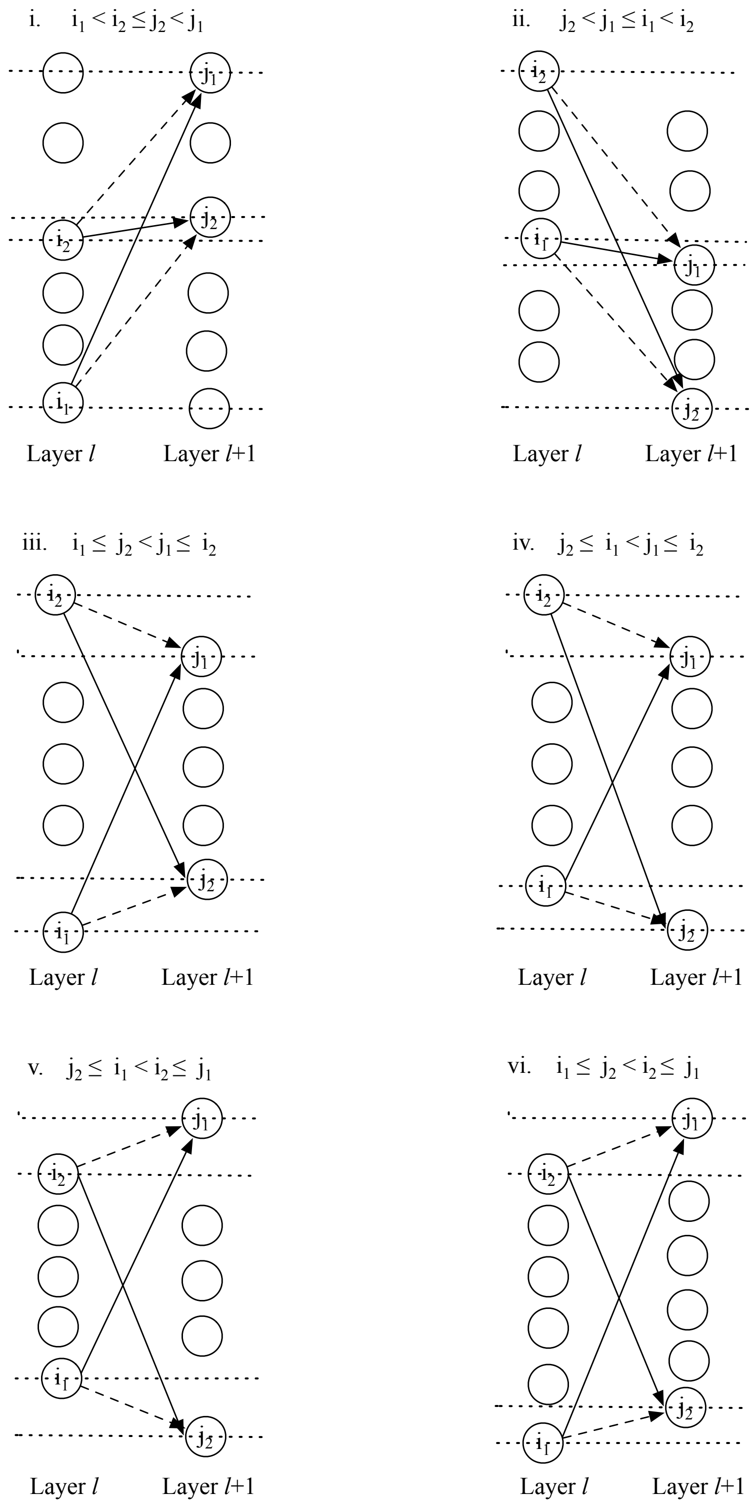


Figure 6.1: Six possible crossings

Then, as a consequence of Proposition 5, it is possible to show that correcting the crossings in an optimal alternate solution of MCFP results in an optimal noncrossing flow.

Proposition 6. *If the unit costs satisfy assumptions (11) – (14), then the MCFP has an optimal solution with no crossing arcs with positive flows.*

Proof. Consider an optimal solution $\mathbf{f}^*$ of the MCFP and crossing arcs (i_1, j_1) and (i_2, j_2), which means $f^*_{i_1,j_1} > 0$ and $f^*_{i_2,j_2} > 0$, and either $i_1 < i_2$ and $j_2 < j_1$ or $i_2 < i_1$ and $j_1 < j_2$. Without loss of generality we can assume that $i_1 < i_2$ and $j_2 < j_1$. It is possible to correct the crossing by a simple operation and adjust the flows on the corresponding arcs without harming its feasibility. If $f^*_{i_1 j_1} \geq f^*_{i_2 j_2} > 0$, then add new arcs (i_2, j_1) and (i_1, j_2) with flows $f^*_{i_2 j_2}$, adjust the flow on arc (i_1, j_2) by subtracting $f^*_{i_1 j_1}$, and finally delete arc (i_2, j_1). However, if $f^*_{i_2 j_2} > f^*_{i_1 j_1}$, then operate similarly by adding new arcs $(i_2 j_1)$ and $(i_1 j_2)$ with flows $f^*_{i_1,j_1}$, adjust the flow on arc (i_2, j_1) by subtracting $f_{i_1 j_1}$, and finally delete arc $(i_1 j_2)$.

These operations are illustrated in Figure 6.2. The crossing represented by solid arcs is corrected by replacing them with dashed arcs. Observe that flow balance is preserved at vertices i_1, i_2, j_1, j_2. Consequently, only cases (i) and (ii) or cases (iv)-(vi) respectively with $c_{j_1 j_2} = 0$, $c_{i_1 j_1} = 0$, $c_{i_1 i_2} = 0$ and $c_{i_2 j_2} = 0$ can occur in an optimal solution of the MCFP, since otherwise it is possible to create a new feasible flow with one fewer crossing and smaller total flow cost after implementing the above operations, which contradicts the optimality of flow $\mathbf{f}^*$. Therefore, the elimination of the crossings in an optimal solution of the MCFP results in an alternative optimal solution with no crossings. □

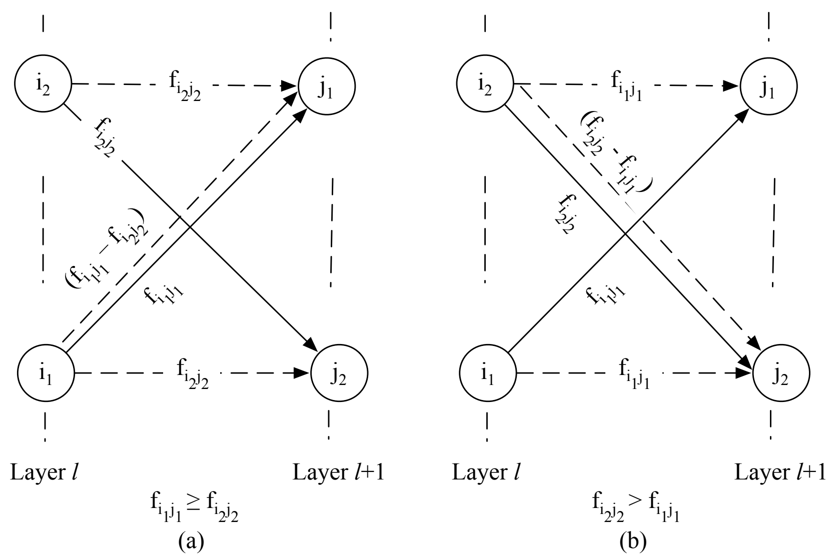


Figure 6.2: Two possible corrections

Notice that Proposition 6 has an implicit assumption as well: the two correction operations are implementable, which may not be possible if (i_1, j_1) or (i_2, j_2) are missing in the network, and/or there is not enough residual capacity on both of them. However, in case the complete layered network is uncapacitated (i.e. $u_{ij} = \infty$, $(i, j) \in A(N)$) they can be applied to correct the crossings.

Propositon 5 and Proposition 6 have also some implications when arcs have finite capacities. This is stated with the following two corollaries.

Corollary 1. *For positive (i.e. $c_{ij} > 0$, $i \neq j$, $(i, j) \in A(N)$), symmetric and additive costs, and $c_{ij} = 0$ for $i = j$, and finite upper bounds, if an optimal solution of the MCFP has crossings of one of the types (iii) - (vi), then $f^*_{i_2 j_1} = u_{i_2 j_1}$ and $f^*_{i_1 j_2} = u_{i_1 j_2}$.*

Proof. Assume that an optimal solution $\mathbf{f}^*$ of the MCFP has a crossing consisting of arcs (i_1, j_1) and (i_2, j_2). As a consequence of the positivity assumption of unit costs and unit cost comparisons given in Proposition 5,

$c_{i_1j_1} + c_{i_2j_2} > c_{i_1j_2} + c_{i_2j_1}$, and as a consequence of correction operations given in Proposition 6, the new flow is still feasible since this operation conserves flow balance at vertices i_1, i_2, j_1, j_2 and has smaller total cost, which contradicts the optimality of $\mathbf{f}^*$. Hence, this operation must have been blocked, which is possible only if $f^*_{i_2j_1} = u_{i_2j_1}$ and $f^*_{i_1j_2} = u_{i_1j_2}$. □

Corollary 2. *For positive (i.e. $c_{ij} > 0$, $i \neq j, (i, j) \in A(N)$), symmetric and additive costs, and $c_{ij} = 0$, for $i = j$, and finite upper bounds the crossing of arcs (i_1, j_1) and (i_2, j_2) can be corrected by one of the two operations given in Proposition 6 if $f^*_{i_2j_2} < \min\{u_{i_1j_2}, u_{i_2j_1}\}$ for $f^*_{i_1j_1} \geq f^*_{i_2j_2}$ or if $f^*_{i_1j_1} < \min\{u_{i_1j_2}, u_{i_2j_1}\}$ for $f^*_{i_2j_2} > f^*_{i_1j_1}$.*

Proof. Directly follows from the definition of the correction operations. □

7. Reducing the number of crossings

An optimal solution of the MCFP relaxation defined by (3)–(6), which is obtained after dropping compatibility constraints (7) and (8), and replacing $u_{ij}x_{ij}$ with u_{ij} in (9), can have crossings. The efficiency of any exact solution algorithm can be improved if some of the potential crossings can be detected and deleted in advance. The following proposition and its corollary provide a tool in this direction.

Proposition 7. *An arc $(p, q) \in A_{l(l+1)}(N)$ is crossed by an arc*

i. *$(r, s) \in A_{l(l+1)}(N)$ with $r > p$ and $s < q$ in an optimal solution $\mathbf{f}^*$ of the MCFP if*

$$-\sum_{\{j\in V_l(N):j\leq p\}} \sum_{\{(j,i)\in A_{l(l+1)}(N):i<q\}} f^*_{ji} > \sum_{\{i\in V^-_{l+1}(N):i<q\}} b_i$$

ii. *$(r, s) \in A_{l(l+1)}(N)$ with $r < p$ and $s > q$ in an optimal solution $\mathbf{f}^*$ of the MCFP if*

$$-\sum_{\{j\in V_l(N):j\geq p\}} \sum_{\{(j,i)\in A_{l(l+1)}(N):i>q\}} f^*_{ji} > \sum_{\{i\in V^-_{l+1}(N):i>q\}} b_i$$

Proof. We only show part (i), since the proof of part (ii) is similar. Consider the flow balance equation of the vertices of layer $l + 1$ with demand b_i (i.e. the set $V^-_{l+1}(N)$) and and add them side by side for vertices $i < q$ to obtain

$$\sum_{\{i\in V^-_{l+1}(N):i<q\}} b_i = \sum_{\{i\in V^-_{l+1}(N):i<q\}} \sum_{(i,j)\in A_{l+1l+2}(N)} f^*_{ij} - \sum_{i\in V^-_{l+1}(N):i<q} \sum_{(j,i)\in A_{l(l+1)}(N)} f^*_{ji}.$$

The second summation on the right hand side can be split into two for arcs $(j, i) \in A_{l(l+1)}(N)$ respectively for $j > p$ and $j \leq p$ which results in

$$\sum_{\{i\in V^-_{l+1}(N):i<q\}} b_i = \sum_{\{i\in V^-_{l+1}(N):i<q\}} \sum_{(i,j)\in A_{l+1l+2}(N)} f^*_{ij} - \sum_{\{i\in V^-_{l+1}(N):i<q\}} \sum_{\{(j,i)\in A_{l(l+1)}(N):j\leq p\}} f^*_{ji} - \sum_{\{i\in V^-_{l+1}(N):i<q\}} \sum_{\{(j,i)\in A_{l(l+1)}(N):j>p\}} f^*_{ji}.$$

Notice that the first two terms on the right hand side represent the difference between the total outflow from the demand vertices of layer $l + 1$ which are below vertex q, and the total inflow to the same vertices from the vertices of layer l which are below vertex p including p as well. Then,

$$\begin{aligned}\sum_{\{i\in V^-_{l+1}(N):i<q\}} b_i &\geq -\sum_{\{i\in V^-_{l+1}(N):i<q\}}\ \sum_{\{(j,i)\in A_{l(l+1)}(N):j\leq p\}} f^*_{ji} - \sum_{\{i\in V^-_{l+1}(N):i<q\}}\ \sum_{\{(j,i)\in A_{l(l+1)}(N):j>p\}} f^*_{ji}\\ &\geq -\sum_{\{j\in V_l(N):i<q\}}\ \sum_{\{(j,i)\in A_{l(l+1)}(N):j\leq p\}} f^*_{ji} - \sum_{\{i\in V^-_{l+1}(N):i<q\}}\ \sum_{\{(j,i)\in A_{l(l+1)}(N):j>p\}} f^*_{ji}\end{aligned}$$

follows since $f^*_{ij} \geq 0$ for all $(i, j) \in A_{l+1l+2}(N)$, and

$$\sum_{\{j\in V_l(N):i<q\}}\ \sum_{\{(j,i)\in A_{l(l+1)}(N):j\leq p\}} f^*_{ji} \geq \sum_{\{i\in V^-_{l+1}(N):i<q\}}\ \sum_{\{(j,i)\in A_{l(l+1)}(N):j\leq p\}} f^*_{ji}.$$

Therefore, if the condition of the assertion is true, then

$$0 > \sum_{\{i\in V^-_{l+1}(N):i<q\}} b_i + \sum_{\{j\in V_l(N):i<q\}}\ \sum_{\{(j,i)\in A_{l(l+1)}(N):j\leq p\}} f^*_{ji} \geq -\sum_{\{i\in V^-_{l+1}(N):i<q\}}\ \sum_{\{(j,i)\in A_{l(l+1)}(N):j>p\}} f^*_{ji}$$

and

$$0 < \sum_{\{i\in V^-_{l+1}(N):i<q\}}\ \sum_{\{(j,i)\in A_{l(l+1)}(N):j>p\}} f^*_{ji}$$

follows consequently. Hence, there must exist an arc $(r, s) \in \{(j, i) \in A_{l(l+1)}(N) : j > p, i < q\}$ with $f_{rs} > 0$. □

Corollary 3. *An arc $(p, q) \in A_{l(l+1)}(N)$ with positive flow cannot exist in an optimal solution of MCNFP if one of the following conditions holds.*

i.

$$-\sum_{\{j\in V_l(N):j\leq p\}}\ \sum_{\{(j,i)\in A_{l(l+1)}(N):i<q,i\in V^-_{l+1}(N)\}} \min\{u_{ji}, -b_i\} > \sum_{\{i\in V^-_{l+1}(N):i<q\}} b_i$$

ii.

$$-\sum_{\{j\in V_l(N):j\geq p\}}\ \sum_{\{(j,i)\in A_{l(l+1)}(N):i>q,i\in V^-_{l+1}(N)\}} \min\{u_{ji}, -b_i\} > \sum_{\{i\in V^-_{l+1}(N):i>q\}} b_i$$

Proof. Directly follows from Proposition 7 as a consequence of the fact that $0 \leq f^*_{ji} \leq \min\{u_{ji}, -b_i\}$ for $(j, i) \in A_{l(l+1)}(N)$, $i \in V^-_{l+1}(N)$. □

First of all, notice that this rule is related to the satisfaction of the total demand of a subset of vertices in $V_{l+1}(N)$. Besides, although it provides a sufficient condition for an arc to be crossed, all possible crossings cannot be prevented in an optimal solution of the MCFP relaxation by the condition described in Corollary 3. Nevertheless, it can reduce the number of crossings by deleting arcs in the network. An arc $(p, q) \in A_{l(l+1)}(N)$ is crossed by another arc $(i, j) \in A_{l(l+1)}(N)$ with $p < i \leq n_l$ and $1 \leq j < q \leq n_{l+1}$ if the total demand associated with vertices $1 \leq j < q$ cannot be satisfied by the total inflow to them from vertices $1 \leq i \leq p$ as stated in case (i) of Corollary 3. Case (ii) deals with the situation that (p, q) is crossed by an arc (i, j) with $1 \leq i < p$ and $1 \leq j < q \leq n_{l+1}$.

It is also possible to state supply versions of Proposition 7 and Corollary 3. We give their statement in the following without proof for the sake of completeness, since their proofs are very similar and can be done by rewording the arguments for the supplies instead of the demands.

Proposition 8. *An arc $(p, q) \in A_{l(l+1)}(N)$ is crossed by an arc*

i. $(r,s) \in A_{l(l+1)}(N)$ *with* $r > p$ *and* $s < q$ *in an optimal solution* $\mathbf{f}^*$ *of the MCFP if*

$$\sum_{\{i\in V_{l+1}(N):i\geq q\}} \sum_{\{(j,i)\in A_{l(l+1)}(N):j>p\}} f^*_{ji} < \sum_{\{j\in V^+_l(N):j>p\}} b_j$$

ii. $(r,s) \in A_{l(l+1)}(N)$ *with* $r < p$ *and* $s > q$ *in an optimal solution* $\mathbf{f}^*$ *of the MCFP if*

$$\sum_{\{i\in V_{l+1}(N):i\leq q\}} \sum_{\{(j,i)\in A_{l(l+1)}(N):k<p\}} f^*_{ji} < \sum_{\{j\in V^+_l(N):j<p\}} b_j$$

Corollary 4. *An arc* $(p,q) \in A_{l(l+1)}(N)$ *with positive flow cannot exist in an optimal solution of MCNFP if one of the following conditions holds.*

i.

$$\sum_{\{i\in V_{l+1}(N):i\geq q\}} \sum_{\{(j,i)\in A_{l(l+1)}(N):j>p,j\in V^+_l(N)\}} \min\{u_{ji}, b_j\} < \sum_{\{j\in V^+_l(N):j>p\}} b_j$$

ii.

$$\sum_{\{i\in V_{l+1}(N):i\leq q\}} \sum_{\{(j,i)\in A_{l(l+1)}(N):j<p,j\in V^+_l(N)\}} \min\{u_{ji}, b_j\} < \sum_{\{j\in V^+_l(N):j<p\}} b_j$$

This time, this rule is related to the satisfaction of the total supply of a subset of vertices in $V_l(N)$. An arc $(p,q) \in A_{l(l+1)}(N)$ is crossed by another arc $(i,j) \in A_{l(l+1)}(N)$ with $1 \leq p < i \leq n_l$ and $1 \leq j < q \leq n_{l+1}$ if the total outflow from vertices $1 \leq p < i \leq n_l$ to vertices $1 \leq q \leq j \leq n_{l+1}$ as stated as case (i) in Corollary 4. Case (ii) deals with the situation that (p,q) is crossed by an arc (i,j) with $1 \leq i < p \leq n_l$ and $1 \leq q < j \leq n_{l+1}$.

As a result, the efficiency of an exact solution algorithm may increase because the network size is smaller. The preprocessing process that Corollary 3 suggests can be stated formally as Algorithm 1 given below. The process Corollary 4 suggests is very similar and the corresponding algorithm is not included for the sake of brevity.

8. Computational results

We have realized a set of computational experiments on a large set of randomly generated test instances in order to assess the strength of the relaxations (i.e. LP relaxations of the formulations and MCFP relaxation) and the value of the preprocessing scheme.

A NETGEN-like [13] instance generator, which exploits the layered structure of the network, has been developed for generating test instances. After setting the number of layers L and the maximum number of vertices in a layer $n_{\max}$, the vertex number of vertices for layers $2, 3, \ldots, L-1$ are generated uniformly within $[1, n_{\max}]$. Layers 1 and L have a single vertex, namely s and t. Arcs are obtained by connecting the vertices of the adjacent layers, and the crossing ones are determined according to the vertex numbering convention we use. Then, a skeleton, which guarantees a feasible noncrossing flow, is constructed and its arcs are assigned large enough unit costs in order to prevent them from participating in an optimal solution.

33 instances are generated with 10, 11, 12,…, 20 layers; three instances for each value. $n_{\max}$ is set to 15, 16, 17, 18, 19, and 20 arbitrarily, and exactly one instance is generated for each combination. The properties of the test instances are reported in Table 1. The first column includes the instance numbers. Columns 2-5 list the basic structural properties; these are the number of layers, maximum number of vertices at each layer, number of

Algorithm 1 Preprocessing for MCNFP

Input: A layered network $N = (V(N), A(N))$, arc capacities $\mathbf{u}$ and unit flow costs $\mathbf{c}$;
Output: A preprocessed network;

```
begin
  for l = 2, 3, ..., L − 2 do
    for all arc (p, q) ∈ A_{l(l+1)}(N) such that p ≤ n_l − 1, q ≥ 2 do
      D = 0, C = 0
      for j = 1, 2, ..., q − 1 do
        if b_j < 0 then
          D ← D + b_j
          for all arc (i, j) ∈ A_{l(l+1)}(N) do
            if i ≤ p then
              C ← C + min{u_{ij}, −b_j}
            end if
          end for
        end if
      end for
    end for
    if C < −D then
      Delete (p, q)
    else
      D = 0, C = 0
      for j = q + 1, q + 2, ..., n_{l+1} do
        if b_j < 0 then
          D ← D + b_j
          for all arc (i, j) ∈ A_{l(l+1)}(N) do
            if i ≥ p then
              C ← C + min{u_{ij}, −b_j}
            end if
          end for
        end if
      end for
      if C < −D then
        Delete (p, q)
      end if
    end if
  end for
end
```

Table 1: Properties of the generated test instances

Instance Number	L	n_{max}	$\lvert V(N)\rvert$	$\lvert A(N)\rvert$	Crossing Arc Pair Number	Max. Num. Arcs	Arc Density (%)	Max. Num. Pairs	Crossing Arc Pair Density (%)
1	10	15	71	617	15,725	4,970	12.41	190,036	8.27
2	10	18	100	1,024	39,942	9,900	10.34	523,776	7.63
3	10	20	83	744	23,782	6,806	10.93	276,396	8.60
4	11	15	90	750	17,734	8,010	9.36	280,875	6.31
5	11	16	96	863	24,081	9,120	9.46	371,953	6.47
6	11	17	91	776	27,704	8,190	9.47	300,700	9.21
7	12	16	99	879	28,706	9,702	9.06	385,881	7.44
8	12	17	103	978	28,867	10,506	9.31	477,753	6.04
9	12	18	107	1,009	37,796	11,342	8.90	508,536	7.43
10	13	15	111	856	17,469	12,210	7.01	365,940	4.77
11	13	17	110	899	24,540	11,990	7.50	403,651	6.08
12	13	19	98	767	19,561	9,506	8.07	293,761	6.66
13	14	16	95	669	12,412	8,930	7.49	223,446	5.55
14	14	18	116	959	21,493	13,340	7.19	459,361	4.68
15	14	20	129	1,278	52,662	16,512	7.74	816,003	6.45
16	15	15	105	774	16,566	10,920	7.09	299,151	5.54
17	15	16	118	1,004	24,251	13,806	7.27	503,506	4.82
18	15	17	144	1,295	46,373	20,592	6.29	837,865	5.53
19	16	15	110	836	14,628	11,990	6.97	349,030	4.19
20	16	16	122	794	11,529	14,762	5.38	314,821	3.66
21	16	20	125	774	13,259	15,500	4.99	299,151	4.43
22	17	15	92	557	9,973	8,372	6.65	154,846	6.44
23	17	16	119	904	18,306	14,042	6.44	408,156	4.49
24	17	17	117	847	15,389	13,572	6.24	358,281	4.30
25	18	17	133	943	18,467	17,556	5.37	444,153	4.16
26	18	18	133	844	10,720	17,556	4.81	355,746	3.01
27	18	19	143	1,085	25,747	20,306	5.34	588,070	4.38
28	19	15	145	1,131	27,812	20,880	5.42	639,015	4.35
29	19	16	130	981	19,430	16,770	5.85	480,690	4.04
30	19	20	189	1,620	44,161	35,532	4.56	1,311,390	3.37
31	20	15	125	872	20,522	15,500	5.63	379,756	5.40
32	20	16	148	939	14,379	21,756	4.32	440,391	3.27
33	20	17	135	951	13,160	20,180	4.71	451,725	2.91

vertices and arcs. Column 6 includes the number of crossing arc pairs. The values given in columns 7 and 9 are the maximum possible number of arcs and arc pairs in the network respectively. They are equal to $|V(N)|(|V(N)| - 1)$ and $|A(N)|(|A(N)| - 1)/2$. They are used to calculate the arc and crossing arc pair densities reported in columns 8 and 10, which are obtained by dividing the elements of column 4 by the elements of column 7 and the elements of column 5 by the elements of column 9.

The computations are carried out on workstations with Intel Xeon CPU E5-2687W0 3.10 GHz processor and 64.0 GB RAM, and operating within Microsoft Windows 7 Professional environment. The programs are coded in C^{++}. The CPU times and objective values are obtained using CPLEX 12.6 with default options on.

8.1. *Formulations and relaxations*

We start our experiments with the MILP formulations and their relaxations. Based on the averages listed in the last row of Table 2, we can say that the second formulation (i.e. (3)-(7), (9), (10)) is the most efficient one in terms of relaxation, although the first formulation (i.e. (3) - (9)) gives 20.26% higher lower bound. This is probably because its LP relaxation can be solved faster, which means a faster process of the nodes of the branch-and-bound tree. The weakest lower bounds belong to the MCFP relaxation. Although their computation requires the solution of MCFP, it can be done efficiently using one of the known algorithms(e.g. [10]). As a result, a very efficient branch-and-bound algorithm can be developed by taking advantage of this.

8.2. *The effect of preprocessing*

In order to judge the effect of the preprocessing, we compare the CPU times of both formulations and relaxations with preprocessing. We prefer not to report preprocessing times since it takes less than 0.001 seconds for

Table 2: Formulations and relaxations

Instances		First Formulation			Second Formulation			MCFP Relaxation	
No.	z^*	CPU Optimum (sec.)	CPU Relaxation (sec.)	Lower Bound	CPU Optimum (sec.)	CPU Relaxation (sec.)	Lower Bound	CPU Relaxation (sec.)	Lower Bound
1	12,525	36.42	0.05	6,356.9	120.12	0.03	4,343.3	0.00	3,732.0
2	27,697	1,644.14	0.36	20,405.1	597.78	0.04	18,617.0	0.01	18,211.0
3	20,930	76.48	0.08	11,767.6	55.63	0.03	9,194.2	0.01	8,749.0
4	36,005	44.85	0.21	13,573.5	38.41	0.05	8,956.4	0.00	7,689.0
5	20,731	179.08	0.08	7,395.7	179.93	0.03	5,147.9	0.00	4,713.0
6	33,793	33.34	0.09	8,867.7	58.77	0.03	6,116.8	0.00	4,997.0
7	19,360	1,497.88	0.07	15,063.6	846.02	0.05	13,591.5	0.00	13,002.0
8	16,783	52.65	0.21	7,413.8	43.55	0.03	6,049.0	0.01	5,847.0
9	30,472	50.51	0.13	9,684.2	46.15	0.09	5,817.0	0.00	4,715.0
10	28,272	56.26	0.13	18,667.9	52.42	0.02	16,537.7	0.01	16,088.0
11	33,980	126.07	0.20	22,755.8	81.51	0.03	20,632.4	0.01	19,825.0
12	26,264	17.09	0.21	12,482.1	89.30	0.03	9,427.8	0.01	8,144.0
13	45,081	110.02	0.17	21,667.8	89.01	0.06	18,011.7	0.00	14,476.0
14	15,073	1,107.39	0.09	6,267.6	741.07	0.04	4,972.0	0.00	4,728.0
15	22,162	193.07	0.41	7,619.5	1,139.20	0.10	5,426.7	0.00	5,183.0
16	27,435	808.50	0.06	16,279.7	772.31	0.03	13,679.0	0.00	12,375.0
17	36,773	2,480.90	0.11	18,934.7	1,824.77	0.03	15,358.4	0.00	14,033.0
18	56,305	17.79	0.20	25,659.4	20.52	0.06	18,726.9	0.01	15,344.0
19	21,666	386.88	0.05	11,699.2	200.76	0.02	9,410.8	0.00	8,564.0
20	41,783	21.45	0.03	35,661.5	24.71	0.02	33,877.9	0.00	33,406.0
21	43,085	20.68	0.17	31,782.8	40.32	0.03	27,980.7	0.00	26,181.0
22	27,308	14.27	0.04	13,886.8	35.59	0.02	11,644.2	0.00	8,815.0
23	25,277	1,013.86	0.17	12,397.9	585.06	0.06	9,806.9	0.00	8,747.0
24	29,701	654.81	0.08	19,957.1	1,817.86	0.02	17,153.8	0.00	16,084.0
25	27,641	12.49	0.06	15,794.8	18.75	0.03	13,470.2	0.01	12,803.0
26	97,928	7.65	0.22	73,421.1	22.08	0.08	61,444.4	0.00	47,289.0
27	40,357	383.21	0.12	22,516.4	271.14	0.04	17,350.2	0.01	14,807.0
28	25,141	68.47	0.08	13,829.7	58.20	0.03	11,626.5	0.01	11,108.0
29	37,767	833.74	0.13	20,836.8	536.03	0.05	15,618.7	0.00	12,549.0
30	50,254	65.11	0.38	36,129.3	145.42	0.11	32,449.0	0.00	31,690.0
31	22,431	6.35	0.05	13,524.9	19.34	0.02	11,606.8	0.00	10,175.0
32	58,713	80.68	0.08	42,226.8	104.81	0.03	36,960.4	0.02	33,595.0
33	40,462	667.09	0.09	21,700.0	732.59	0.03	18,026.5	0.00	16,428.0
Average	33,308	386.94	0.14	19,279.63	345.73	0.04	16,031.29	0.00	14,366.42

Table 3: The effect of preprocessing

Instances		First Formulation			Second Formulation			MCFP Relaxation	
No.	Deleted arcs (%)	CPU Optimum (sec.)	CPU Relaxation (sec.)	Lower Bound	CPU Optimum (sec.)	CPU Relaxation (sec.)	Lower Bound	CPU Relaxation (sec.)	Lower Bound
1	17.18	150.96	0.03	10,084.1	130.07	0.01	9,004.5	0.00	7,573.0
2	22.56	807.76	0.14	23,531.1	710.11	0.04	22,343.6	0.00	21,771.0
3	32.53	58.83	0.08	17,617.6	65.47	0.01	16,370.0	0.00	15,417.0
4	34.00	46.78	0.09	26,024.5	30.83	0.02	22,215.4	0.00	18,233.0
5	33.37	199.91	0.05	13,353.6	154.49	0.02	11,405.4	0.00	10,459.0
6	39.95	29.14	0.05	17,744.5	34.02	0.02	14,853.7	0.00	11,790.0
7	17.75	813.49	0.05	15,517.5	736.54	0.02	13,927.2	0.00	13,211.0
8	27.20	54.24	0.06	11,002.1	26.60	0.02	10,051.0	0.00	9,780.0
9	41.43	46.01	0.04	19,005.1	49.86	0.02	17,008.1	0.00	13,809.0
10	26.99	47.25	0.04	23,194.2	67.05	0.01	22,016.2	0.00	21,062.0
11	27.03	84.58	0.08	26,659.7	82.01	0.02	24,549.0	0.00	22,621.0
12	31.94	36.89	0.07	17,829.4	43.48	0.03	14,982.2	0.01	12,828.0
13	31.09	73.31	0.14	30,034.7	137.09	0.02	26,735.0	0.00	21,627.0
14	22.42	1,549.86	0.07	10,500.2	902.51	0.01	9,372.4	0.00	8,876.0
15	37.72	270.41	0.07	16,377.9	246.04	0.02	14,871.8	0.00	13,818.0
16	25.97	629.98	0.05	20,596.6	801.53	0.02	18,740.5	0.00	16,511.0
17	27.29	855.85	0.06	29,049.0	1,415.23	0.02	26,260.6	0.00	23,116.0
18	40.62	11.62	0.05	40,105.0	19.06	0.02	36,289.8	0.01	32,746.0
19	23.92	120.99	0.03	16,242.7	162.94	0.00	14,291.0	0.00	13,226.0
20	20.15	15.63	0.03	39,794.9	25.82	0.02	38,818.5	0.02	37,431.0
21	39.66	25.94	0.01	40,188.5	34.07	0.01	38,722.2	0.00	35,262.0
22	33.75	16.27	0.02	21,957.7	19.31	0.01	20,464.6	0.00	15,803.0
23	25.66	834.77	0.04	19,232.2	623.70	0.02	17,545.0	0.00	15,479.0
24	26.21	641.70	0.03	24,720.9	750.19	0.02	23,241.9	0.00	22,516.0
25	32.77	10.27	0.06	23,398.6	8.97	0.02	21,655.9	0.01	18,863.0
26	26.78	15.27	0.09	81,890.1	24.32	0.03	74,967.9	0.00	61,555.0
27	24.06	242.85	0.12	29,029.3	346.73	0.04	25,195.9	0.01	21,840.0
28	34.39	58.08	0.05	21,149.7	50.28	0.02	20,268.8	0.00	18,964.0
29	21.71	660.31	0.08	27,530.7	764.88	0.02	24,636.9	0.00	21,660.0
30	30.43	89.73	0.06	45,592.5	71.62	0.02	43,524.6	0.00	42,261.0
31	33.83	7.73	0.03	19,353.6	16.16	0.02	18,220.8	0.00	14,838.0
32	30.56	64.70	0.06	51,554.2	78.23	0.02	49,395.0	0.00	43,606.0
33	32.07	599.35	0.06	27,870.6	502.49	0.02	25,607.8	0.00	23,013.0
Average	29.48	277.89	0.06	25,991.91	276.72	0.02	23,865.25	0.00	21,259.55

all test instances. According to the results summarized in Table 3, we can say that the effect of preprocessing is remarkable. First of all, the values in the second column indicate that, on the average, 29.48% of the arcs have been deleted. There is also a considerable decrease in the running times. The average CPU time decreases by 28.18% and 19.96% for the first and second formulations respectively. The second formulation is 0.4% with preprocessing compared to the first formulation. There is a higher improvement in the efficiency of the first formulation due to preprocessing, which makes its average performance comparable with that of the second formulation after preprocessing. An interesting observation is related to the lower bounds; preprocessing makes them 34.82%, 48.87%, and 48.08% higher on the average.

9. Conclusions

In this work we have considered a variant of the well-known minimum cost network flow problem, which allows positive flow values on only noncrossing arcs. This problem can be often faced in real applications. First, we show that the problem is NP-complete. Then, we introduce polynomially solvable special cases applicable to well-known practical problems such as the crane scheduling problem encountered in container terminals: the problem becomes polynomially solvable when the traveling distances are used as set-up costs, which is a common practice. We also propose mixed-integer linear programming formulations. Because of the introduced conflicting arc lists they can be easily adopted for general network topologies and conflict types different from crossings. Finally, we develop a preprocessing scheme and experimentally study its effect on the formulations. It turns out that it increases the efficiency of the formulations and relaxations at the expense of a very low computational effort.

We should point out that MCNFP is very likely to be strongly NP-hard and this might be shown using a strongly NP-hard path problem for the reduction. Yet another research challenge is to consider MCNFP for network structure more general than layered networks and conflicts more general than crossings.

Acknowledgements—We gratefully acknowledge the support of TÜBİTAK (The Scientific and Technological Research Council of Turkey) under the grant no: 213M441. We also would like to send our gratitudes to the anonymous referees for their comments and suggestions they made.

[1] Ahuja, R. K., T. L. Magnanti, J. B. Orlin. 1993. *Networks Flows.* New-York: Prentice-Hall.

[2] Bierwirth, C., F. Meisel. 2010. A survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research* **202** 615–627.

[3] Bierwirth, C., F. Meisel. 2015. A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research* **244** 675–689.

[4] Boysen, N., D. Briskorn, F. Meisel. 2017. A generalized classification scheme for crane scheduling with interference. *European Journal of Operational Research* **258** 343–357.

[5] Carlo, H.J., I.F.A. Vis, K.J. Roodbergen. 2015. Seaside operations in container terminals: literature overview, trends and research directions. *Flexible Services and Manufacturing Journal* **27** 224–262.

[6] Darmann, A., U. Pferschy, Schauer. 2009. *Determining a minimum spanning tree with disjunctive constraints.* Algorithmic Decision Theory, ADT 2009, Lecture Notes in Computer Science, vol. 5783, Springer, 414–423.

[7] Darmann, A., U. Pferschy, J. Schauer, G. J. Woeginger. 2011. Paths, trees and matchings under disjunctive constraints. *Discrete Applied Mathematics* **159** 1726–1735.

[8] Ford, L. R., Jr., D. R. Fulkerson. 1962. *Flows in Networks.* New Jersey: Princeton University Press.

[9] Garey, M. R., D. S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-completeness.* San Francisco: Freeman.

[10] Goldberg, A. V., R. E. Tarjan. 1990. Solving minimum cost flow problem by successive approximation. *Mathematics of Operations Research* **15** 430–466.

[11] Goossens, D. R., F. C. R. Spieksma. 2009. Structured modeling: survey and future research directions. *4OR* **7**(1) 51–60.

[12] Gouveia, L., Leitner M., M. Ruthmair. 2018. Layered graph approaches for combinatorial optimization problems. URL `http://www.optimization-online.org/DB_HTML/2018/04/6564.html`.

[13] Klingman, D., A. Napier, J. Stutz. 1974. Netgen: A program for generating large scale capacitated assignment, transportation, and minimum cost flow networks. *Management Science* **20** 814–820.

[14] Lee, D.-H., H. Q. Wang, L. Miao. 2008. Quay crane scheduling with non-interference constraints in port container terminals. *Transportation Research Part E* **44** 124–135.

[15] Öncan, T., R. Zhang, A. P. Punnen. 2013. The minimum cost perfect matching problem with conflict pair constraints. *Computers & Operations Research* **40** 920–930.

[16] Pferschy, U., J. Schauer. 2013. The maximum flow problem with disjunctive constraints. *Journal of Combinatorial Optimization* **26** 109–119.

[17] Samer, P., S. Urrutia. 2015. A branch and cut algorithm for minimum spanning trees under conflict constraints. *Optimization Letters* **9** 41–55.

[18] Steenken, D., S. Voß, R. Stahlbock. 2004. Container terminal operation and operations research – a classification and literature review. *OR Spectrum* **26** 3–49.

[19] Sun, M. 1998. A tabu search heuristic procedure for solving the transportation problem with exclusionary side constraints. *Journal of Heuristics* **3** 305–326.

[20] Sun, M. 2002. The transportation problem with exclusionary side constraints and two branch-and-bound algorithms. *European Journal of Operational Research* **140** 629–647.

[21] Türkoğulları, Y.B., Z. C. Taşkın, N. Aras, K. Altınel. 2016. Optimal berth allocation, time-variant quay crane assignment and scheduling with crane setups in container terminals. *European Journal of Operational Research* **254** 985–1001.

[22] Zhang, R., S. N. Kabadi, A. P. Punnen. 2011. The minimum spanning tree problem with conflict constraints and its variations. *Discrete Optimization* **8** 101–205.