European Journal of Operational Research 242 (2015) 51–62

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

Incremental network design with maximum flows

Thomas Kalinowski[a,*], Dmytro Matsypura[b], Martin W. P. Savelsbergh[c]

[a] The University of Newcastle, School of Mathematical & Physical Sciences, University Drive, Callaghan NSW 2308, Australia
[b] The University of Sydney, Business School, Merewether Building, NSW 2006, Australia
[c] Georgia Institute of Technology, H. Milton Stewart School of Industrial & Systems Engineering, 755 Ferst Drive, NW Atlanta, GA 30332-0205, USA

ARTICLE INFO

Article history:
Received 24 December 2013
Accepted 1 October 2014
Available online 13 October 2014

Keywords:
Network design
Approximation algorithms
Scheduling

ABSTRACT

We study an incremental network design problem, where in each time period of the planning horizon an arc can be added to the network and a maximum flow problem is solved, and where the objective is to maximize the cumulative flow over the entire planning horizon. After presenting two mixed integer programming (MIP) formulations for this NP-complete problem, we describe several heuristics and prove performance bounds for some special cases. In a series of computational experiments, we compare the performance of the MIP formulations as well as the heuristics.

1. Introduction

In the planning process for many network infrastructures, when the network is constructed over a significant period of time, the properties of the intermediate partial networks have to be taken into account. *Incremental network design*, introduced in Baxter, Elgindy, Ernst, Kalinowski, and Savelsbergh (2014), represents a class of optimization problems capturing that feature and combining two types of decisions: which arcs should be added to a given network in order to achieve a certain goal, and when should these arcs be added?

Variants of this problem have been studied in diverse contexts, for instance, the design of transportation networks (Kim, Kim, & Song, 2008; Ukkusuri & Patil, 2009), network infrastructure restoration after disruptions due to natural disasters (Cavdaroglu, Hammel, Mitchell, Sharkey, & Wallace, 2013; Lee, Mitchell, & Wallace, 2007, 2009), and the transformation of an electrical power grid into a smart grid (Mahmood, Aamir, & Anis, 2008; Momoh, 2009). Our study is motivated by infrastructure expansion questions arising in the coal export supply chain in the Hunter Valley with coal terminals in the Port of Newcastle (see Boland & Savelsbergh, 2011 for details).

A general class of mathematical optimization problems that captures essential features of the described decision problems, and includes the problem discussed in this paper as a special case, was introduced in Nurre, Cavdaroglu, Mitchell, Sharkey, and Wallace (2012) and Nurre and Sharkey (2014), where an *integrated network design and scheduling* problem is specified by (1) a scheduling environment that describes the available resources for adding arcs to the network, (2) a performance measure, which prescribes how a given network is evaluated (for instance by the shortest s-t path or by the maximum s-t flow in the network), and (3) by the optimization goal, which is either to reach a certain level of performance as quickly as possible or to optimize the cumulative performance over the entire planning horizon.

We focus on the special case corresponding to incremental network design as introduced in Baxter et al. (2014). Our scheduling environment is such that at most one arc can be added to the network in each time period, and we optimize the cumulative performance, i.e., the sum of the performance measures of the networks in all time periods. Even in this simple setting, the problem has been shown to be NP-complete for classical network optimization problems: for the shortest s-t path problem in Baxter et al. (2014), and for the maximum s-t flow problem in Nurre and Sharkey (2014). Interestingly, the incremental variant of the minimum spanning tree problem can be solved efficiently by a greedy algorithm (Engel, Kalinowski, & Savelsbergh, 2013), while it becomes NP-complete in the more general setup of Nurre and Sharkey (2014). The performance measure considered in this paper is the value of a maximum s-t flow.

In Section 2, we introduce notation, state the problem precisely, and present two MIP formulations. In Section 3, we describe three heuristics, the first one seeks to increment the flow as quickly as possible, the second seeks to reach a maximum flow as quickly as possible, and the third one is a hybrid of the first two. In Section 4, we prove performance guarantees for the first two heuristics in special cases: for unit capacity networks they provide a 2-approximation algorithm and a 3/2-approximation algorithm, respectively, and these bounds can be strengthened when the network has a special structure. Section 5 discusses the results of a set of computational experiments using randomly generated instances. After describing the instance

* Corresponding author. Tel.: +61 2 4921 6558; fax: +61 2 4921 6898.
E-mail address: thomas.kalinowski@newcastle.edu.au (T. Kalinowski).

http://dx.doi.org/10.1016/j.ejor.2014.10.003
0377-2217/

generation, we compare the performance of the two MIP formulations, and evaluate and compare the heuristics on hard instances. We end, in Section 6, with some final remarks.

2. Problem formulation

We are given a network $D = (N, A)$ with node set N and arc set $A = A_e \cup A_p$, where A_e contains existing arcs and A_p contains potential arcs, as well as a source $s \in N$ and sink $t \in N$. For each arc, we are given an integer capacity $u_a > 0$, and for every node $v \in N$, we denote with $\delta^{\text{in}}(v)$ and $\delta^{\text{out}}(v)$ the set of arcs entering v and the set of arcs leaving v, respectively. Let $T > |A_p|$ be the length of the planning horizon. In every period, we have the option to expand the useable network, which initially consists of only the existing arcs, by "building" a single potential arc $a \in A_p$, which will be available for use from the following time period on. In every period, the value of a maximum s-t flow is recorded (using only useable arcs, i.e., existing arcs and potential arcs that have been built in previous periods). The objective is to maximize the total flow over the planning horizon. Note that the length of the planning horizon ensures that every potential arc can be built. We refer to a maximum s-t flow using only existing arcs as an *initial maximum flow*, and to a maximum flow for the complete network as an *ultimate maximum flow*. This problem is strongly NP-hard (Nurre and Sharkey, 2014) even when restricted to instances where every existing arc has capacity 1 and every potential arc has capacity 3. (A simple proof of this result can be found in Appendix A.)

The problem can be formulated as a mixed integer program. For every $a \in A$ and $k \in \{1, \dots, T\}$, we have a flow variable $x_{ak} \geqslant 0$, and for every $a \in A_p$ and $k \in \{1, \dots, T\}$, we have a binary variable y_{ak} which indicates if arc a is built before period k ($y_{ak} = 1$) or not ($y_{ak} = 0$). The incremental maximum flow problem is then

$$\max \sum_{k=1}^{T} \left(\sum_{a\in\delta^{\text{out}}(s)} x_{ak} - \sum_{a\in\delta^{\text{in}}(s)} x_{ak} \right)$$

subject to

$$\begin{aligned}
\sum_{a\in\delta^{\text{out}}(v)} x_{ak} - \sum_{a\in\delta^{\text{in}}(v)} x_{ak} = 0 \qquad & \text{for } v \in N \setminus \{s,t\},\ k \in \{1,\dots,T\},\\
x_{ak} \leqslant u_a \qquad & \text{for } a \in A_e,\ k \in \{1,\dots,T\},\\
x_{ak} \leqslant u_a y_{ak} \qquad & \text{for } a \in A_p,\ k \in \{1,\dots,T\},\\
y_{ak} \geqslant y_{a,k-1} \qquad & \text{for } a \in A_p,\ k \in \{2,\dots,T\},\\
y_{a1} = 0 \qquad & \text{for } a \in A_p,\\
\sum_{a\in A_p} (y_{ak} - y_{a,k-1}) \leqslant 1 \qquad & \text{for } k \in \{2,\dots,T-1\},\\
x_{ak} \geqslant 0 \qquad & \text{for } a \in A,\ k \in \{1,\dots,T\},\\
y_{ak} \in \{0,1\} \qquad & \text{for } a \in A_p,\ k \in \{1,\dots,T\}.
\end{aligned}$$

We denote this formulation by IMFP^1.

A potential weakness of IMFP^1 is that it may suffer from symmetry. If multiple arcs need to be build to increase the maximum s-t flow, then the order in which these arcs are build does not matter, which introduces alternative, symmetrical solutions. Next, we present an alternative MIP formulation which avoids this difficulty. Let f and F denote the initial and the ultimate maximum flow value, respectively, and let $r = F - f$. We introduce binary variables y_{ak} for $a \in A_p$ and $k = 1, 2, \dots, r$ with the interpretation

$$y_{ak} = \begin{cases} 1 & \text{if arc } a \text{ is build while the max flow value is less than } f+k,\\ 0 & \text{otherwise.} \end{cases}$$

The number of time periods with maximum flow value f is $\sum_{a\in A_p} y_{a1}$, and for $k = 1, \dots, r-1$, the number of time periods with maximum flow value $f+k$ is $\sum_{a\in A_p} (y_{a,k+1} - y_{ak})$. Consequently, the total flow is

$$\begin{aligned}
&f \sum_{a\in A_p} y_{a1} + \sum_{k=1}^{r-1} (f+k) \sum_{a\in A_p} (y_{a,k+1} - y_{ak}) + F\left(T - \sum_{a\in A_p} y_{ar}\right)\\
&\quad = TF + \sum_{a\in A_p}\sum_{k=1}^{r} y_{ak}\,[(f+k-1) - (f+k)] = TF - \sum_{a\in A_p}\sum_{k=1}^{r} y_{ak}.
\end{aligned}$$

Hence the incremental maximum flow problem can also be formulated as follows

$$\min \sum_{a\in A_p}\sum_{k=1}^{r} y_{ak}$$

subject to

$$\begin{aligned}
&\sum_{a\in\delta^{\text{out}}(v)} x_{ak} - \sum_{a\in\delta^{\text{in}}(v)} x_{ak}\\
&\quad = \begin{cases} 0 & \text{for } v \notin \{s,t\}\\ f+k & \text{for } v = s\\ -f-k & \text{for } v = t \end{cases} \qquad \text{for } v \in N,\ k \in \{1,\dots,r\},\\
&x_{ak} \leqslant u_a \qquad \text{for } a \in A_e,\ k \in \{1,\dots,r\},\\
&x_{ak} \leqslant u_a y_{ak} \qquad \text{for } a \in A_p,\ k \in \{1,\dots,r\},\\
&y_{ak} \leqslant y_{a,k+1} \qquad \text{for } a \in A_p,\ k \in \{1,\dots,r-1\},\\
&x_{ak} \geqslant 0 \qquad \text{for } a \in A,\ k \in \{1,\dots,r\},\\
&y_{ak} \in \{0,1\} \qquad \text{for } a \in A_p,\ k \in \{1,\dots,r\}.
\end{aligned}$$

We denote this formulation by IMFP^2.

Observe that the size of IMFP^1 strongly depends on the length of the planning horizon, whereas the size of IMFP^2 strongly depends on the difference between the initial and ultimate maximum flow values.

3. Heuristics

In this section, we introduce two natural strategies for trying to obtain high quality solutions: (1) getting a flow increment as quickly as possible, and (2) reaching a maximum possible flow as quickly as possible, as well as a hybrid strategy.

3.1. Quickest flow increment

A natural greedy strategy is to build the arcs such that a flow increment is always reached as quickly as possible. Suppose we have already built the arcs in $B \subseteq A_p$ to reach a maximum flow value $f+k$. A smallest set of potential arcs to be built, in addition to B, to reach a flow of value at least $f+k+1$ can be determined by solving a fixed charge network flow problem: find a flow of value $f+k+1$ where arcs in $A_e \cup B$ have zero cost, and arcs in $A_p \setminus B$ incur a cost of 1 if they carry a nonzero flow. More formally, in order to determine the smallest number of potential arcs that have to be built to increase the flow from $f+k$ to at least $f+k+1$, we solve the problem $\textsc{MinArcs}(B, k+1)$:

$$\min z = \sum_{a\in A_p\setminus B} y_a$$

subject to

$$\begin{aligned}
&\sum_{a\in\delta^{\text{out}}(v)} x_a - \sum_{a\in\delta^{\text{in}}(v)} x_a\\
&\quad = \begin{cases} 0 & \text{for } v \notin \{s,t\}\\ f+k+1 & \text{for } v = s\\ -f-k-1 & \text{for } v = t \end{cases} \qquad \text{for } v \in N,\\
&x_a \leqslant u_a \qquad \text{for } a \in A_e \cup B,\\
&x_a \leqslant u_a y_a \qquad \text{for } a \in A_p \setminus B,\\
&x_a \geqslant 0 \qquad \text{for } a \in A,\\
&y_a \in \{0,1\} \qquad \text{for } a \in A_p \setminus B.
\end{aligned}$$