Scheduling unit processing time arc shutdown jobs to maximize network flow over time: complexity results*

Natashia Boland, Thomas Kalinowski, Reena Kapoor, Simranjit Kaur
School of Mathematical & Physical Sciences, University of Newcastle

Keywords : network models, complexity theory, maintenance scheduling, mixed integer programming
AMS Classification: 90C10, 90B10, 68Q25

Motivation and Problem Formulation

Many real life systems can be viewed as a network with arc capacities, supporting the flow of a commodity. For example, transportation networks, or supply chains, may on occasion be viewed this way. We were motivated by a particular coal export supply chain [4], in which – as in many resources supply chains today – maximizing throughput is a key concern. Whilst this suggests a maximum flow model would be appropriate, in fact, the real network is not static: capacities change over time, and in particular, some arcs are shut down for maintenance at certain times. Often there is some flexibility in the time when maintenance jobs can be scheduled. Every maintenance schedule will incur some loss in the total throughput of the network. To obtain maximum throughput, it is important to select the schedule that leads to minimum loss of flow. This leads to a model in which arc maintenance jobs need to be scheduled so as to maximize the total flow in the network over time [1, 2, 3].

In this paper we consider the case of this problem in which all maintenance jobs have unit processing time. The problem is defined over a network $N = (V, A, s, t, u)$ with node set V, arc set A, source $s \in V$, sink $t \in V$ and nonnegative capacity vector $u = (u_a)_{a\in A}$. Note that we permit parallel arcs, i.e. there may exist more than one arc in A having the same start and end node. By $\delta^-(v)$ and $\delta^+(v)$ we denote the set of incoming and outgoing arcs of node v, respectively. We consider this network over a set of T time periods indexed by the set $[T] := \{1, 2, \dots, T\}$, and our objective is to maximize the total flow from s to t. In addition, we are given a subset $J \subseteq A$ of arcs that have to be shut down for exactly one time period in the time horizon. In other words, there is a set of maintenance jobs, one for each arc in J, each with unit processing time. Our optimization problem is to choose these outage time periods in such a way that the total flow from s to t is maximized. More formally, this can be written as a mixed binary program as follows:

$$\max\ z = \sum_{i=1}^{T} \sum_{a\in\delta^+(s)} x_{ai} \tag{1}$$

$$\text{s.t.}\quad x_{ai} \leqslant u_a \qquad a \in A \setminus J,\ i \in [T], \tag{2}$$

$$x_{ai} \leqslant u_a y_{ai} \qquad a \in J, i \in [T], \tag{3}$$

$$\sum_{i=1}^{T} y_{ai} = T - 1 \qquad a \in J, \tag{4}$$

$$\sum_{a\in\delta^-(v)} x_{ai} = \sum_{a\in\delta^+(v)} x_{ai} \qquad v \in N \setminus \{s, t\},\ i \in [T], \tag{5}$$

$$x_{ai} \geqslant 0 \qquad a \in A,\ i \in [T], \tag{6}$$

$$y_{ai} \in \{0, 1\} \qquad a \in J,\ i \in [T], \tag{7}$$

where $x_{ai} \geq 0$ for $a \in A$ and $i \in [T]$ denotes the flow on arc a in time period i, and $y_{ai} \in \{0, 1\}$ for $a \in J$ and $i \in [T]$ indicates when the arc a is *not* shut down for maintenance in time period i, i.e. $y_{ai} = 0$ in the period i in which the outage for arc a is scheduled.

To the best of our knowledge, Boland et.al. [1, 2, 3] initiated study on the problem with general processing times. In [1, 3], the coal supply chain application, which has a number of additional side constraints, is modelled and solved using a rolling time horizon mixed integer programming approach. In [2], the complexity of the general problem is established, and four local search heuristics are developed and compared. We are not aware of any other studies on this, or on closely related problems. Several authors have studied dynamic network flows. For instance [5] studied the problem of finding the maximum flow that can be sent from a source to a sink in T time units, in a network with transit times on the arcs. Variations of the dynamic maximum flow problem with zero transit times are discussed in [6], [7], and [8]. None of these have a scheduling component. Machine scheduling problems have received a great deal of attention in the literature [9], but in the problem we study here, there is no underlying machine, and the association of jobs with network arcs and a maximum flow objective give it quite a different character.

*This research was supported by the ARC Linkage Grant no. LP0990739.

Our key contribution in this paper is an analysis of how the complexity of the problem depends on important characteristics. In particular, we consider (i) the case that all arcs have jobs, i.e. $J = A$, (ii) *balanced* networks, in which the capacity into and out of each (non-terminal, i.e. transhipment) node is equal, and (iii) networks with a single transhipment node. We show for case (i) that it is optimal to schedule all jobs in the same time period, and that this is also true if the network is both balanced and has a single transhipment node. However if the network is balanced (but with more than one transhipment node), then the problem is strongly NP-hard.

Tractability Results

The problem in general is NP-hard [2]. In this proof, the reduction gave rise to a network with a single transhipment node, which was not balanced, and in which not all arcs needed to be shut down. This left open the complexity of the cases that all arcs have an associated outage, or the network is balanced. The former case is relatively easy to resolve.

Proposition 1. *If all arcs in a network must have an outage, i.e. if $A = J$ then it is optimal to shut them down at the same time.*

Proof. Let C be a minimum cut in the network separating s and t, and let $M = \sum_{a\in C} u_a$ be its capacity. Clearly, the objective value for the solution with all jobs scheduled in the same time period is $M(T-1)$, since (i) in any period with no outage scheduled, the maximum flow is simply the capacity of the minimum cut, M, (ii) if all outages are scheduled in the same period, there must be $T-1$ such periods, and (iii) in the remaining period all arcs are shut down so the flow is zero. But $M(T-1)$ is also an upper bound on the value of the optimal solution, which we show as follows. First, the flow in each time period $i \in [T]$ satisfies $\sum_{a\in\delta^+(s)} x_{ai} \le \sum_{a\in C} x_{ai}$, since C is a cut separating s and t. Thus the total flow is given by

$$\sum_{i=1}^{T}\sum_{a\in\delta^+(s)} x_{ai} \le \sum_{i=1}^{T}\sum_{a\in C} x_{ai} \le \sum_{i=1}^{T}\sum_{a\in C} u_a y_{ai}$$

by (3) and since $J = A$ so $C \subseteq J$. But

$$\sum_{i=1}^{T}\sum_{a\in C} u_a y_{ai} \le \sum_{a\in C} u_a \sum_{i=1}^{T} y_{ai} = \sum_{a\in C}(T-1)u_a = M(T-1),$$

by (4), and again since $C \subseteq J$. The result follows. □

Proposition 2. *If the network has a single transhipment node, and is balanced, then it is optimal to schedule all jobs at the same time.*

Proof. Consider a balanced network having only one transhipment node say v. Let M be the flow in the network when no arc is on job. Then $\sum_{a\in\delta^-(v)} u_a = \sum_{a\in\delta^+(v)} u_a = M$. Let $J = J_1 \cup J_2$ where $J_1 \subseteq \delta^-(v)$ and $J_2 \subseteq \delta^+(v)$. Let $N_1 = \sum_{a\in J_1} u_a$ and $N_2 = \sum_{a\in J_2} u_a$. When all arcs in J are scheduled at the same time, the total throughput of the network is given by

$$M(T-1) + \min\{M - N_1, M - N_2\} = MT - \max\{N_1, N_2\}.$$

Also for any schedule the total flow in the network over time T will be $\min\{\sum_{i=1}^{T}\sum_{a\in\delta^-(v)} x_{ai}, \sum_{i=1}^{T}\sum_{a\in\delta^+(v)} x_{ai}\}$. But for any schedule

$$\sum_{i=1}^{T}\sum_{a\in\delta^-(v)} x_{ai} \le \sum_{i=1}^{T}\sum_{a\in\delta^-(v)} u_a y_{ai} = \sum_{i=1}^{T}\sum_{a\in J_1} u_a y_{ai} + \sum_{i=1}^{T}\sum_{a\in\delta^-(v)\setminus J_1} u_a y_{ai} \le N_1(T-1) + (M - N_1)T = MT - N_1.$$

Similarly

$$\sum_{i=1}^{T}\sum_{a\in\delta^+(v)} x_{ai} \le MT - N_2.$$

Therefore

$$\min\{\sum_{i=1}^{T}\sum_{a\in\delta^-(v)} x_{ai}, \sum_{i=1}^{T}\sum_{a\in\delta^+(v)} x_{ai}\} \le MT - \max\{N_1, N_2\}.$$

Thus $MT - \max\{N_1, N_2\}$ is an upper bound on the total throughput under any schedule of jobs and is attained when all jobs scheduled together. Hence the result. □

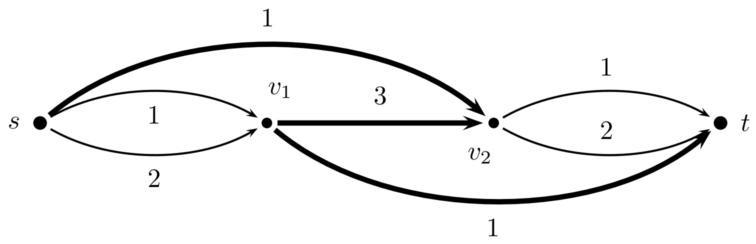

Figure 1: Balanced network but not optimal all together

If the network is balanced, but has more than one transhipment node, then scheduling all jobs at the same time may not be optimal, as the following example shows.

Example 1. *Consider the balanced network with as shown in Figure 1, where arc labels indicates capacities and the bold arcs don't have jobs associated with them. Let T=2. Shutting down all arcs on job together gives a total flow of 4 units. Whereas if we schedule the arcs with unit capacity at first time period and the remaining arcs on job at second time period then the total flow would be 5.*

In fact, as we shall show next, in general, if the network is balanced the problem is strongly NP-hard.

Hardness Result

Proposition 3. *The problem remains strongly NP-hard for balanced networks.*

Proof. We proceed by reduction from 3-Partition. Let a 3-Partition instance be given by B and $u = (u_1, \dots, u_{3m})$. Consider a network shown in Figure 2, where the arc labels indicate capacities, and the bold arcs don't have jobs associated with them. The total throughput is bounded by $2m(m-1)B$. To achieve the bound of $2m(m-1)B$, the

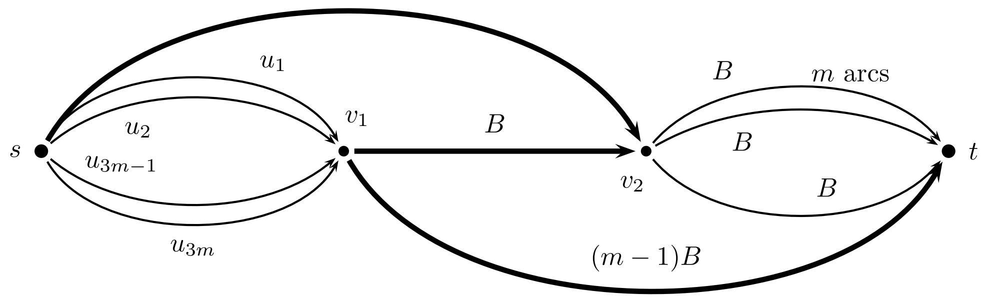

Figure 2: The network for the reduction from 3-Partition.

arcs (s, v_2) and (v_1, t) have to be at capacity in each time period. If the arc (s, v_2) is at capacity in every time period, then the arc (v_1, v_2) does not carry any flow at all. Thus the bound $2m(m-1)B$ can be achieved if and only if there is a solution for the 3-Partition instance. □

Future Work

The above results show either the problem is strongly NP-hard, or it is optimal to schedule all jobs at the same time. Other network features that may distinguish cases that are not strongly NP-hard, but in which it is also not optimal to have all jobs at the same time, are of interest. The complexity of the special case in which arc capacities are also identical is yet to be resolved. We will also consider practical algorithms and approximation algorithms for the general problem.

References

[1] N. Boland, T. Kalinowski, H. Waterer and L. Zheng, "An optimisation approach to maintenance scheduling for capacity alignment in the Hunter Valley coal chain", in: *Proceedings of the 35th APCOM Symposium: Applications of Computers and Operations Research in the Minerals Industry*, E.Y. Baafi, R.J. Kininmonth and I. Porter (eds), 24-30 Sept. 2011, Wollongong, Australia, The Australasian Institute of Mining and Metallurgy Publication Series No 11/2011, pp. 887-897.

[2] N. Boland, T. Kalinowski, H. Waterer, and L. Zheng, "Scheduling arc maintenance jobs in a network to maximize total flow over time", *Discrete Applied Mathematics: Special Issue on Matheuristics*, under review.

[3] N. Boland, T. Kalinowski, H. Waterer, and L. Zheng, "Mixed integer programming based maintenance scheduling for the Hunter Valley coal chain", *Journal of Scheduling: Special Issue on Maintenance Scheduling: Theory and Applications*, under review.

[4] N. Boland and M. Savelsbergh, "Optimizing the Hunter Valley coal chain", in: *Supply Chain Disruptions: Theory and Practice of Managing Risk*, H. Gurnani, A. Mehrotra and S. Ray (eds), Springer London, 2012, pp. 275-302.

[5] L.R. Ford and D.R. Fulkerson, *Flows in Networks.* Princeton University Press, Princeton, NJ, 1962.

[6] L. Fleischer, "Universally maximum flow with piecewise-constant capacities", *Networks*, 38:115-125, 2001.

[7] B. Hajek and R.G. Ogier, "Optimal dynamic routing in communication networks with continuous traffic", *Networks*, (1984), 14:457-487, 1984.
[8] B.Hoppe and E. Tardos, "Polynomial time algorithms for some evacuation problems", *Proceedings of the 5th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1994, pp. 433-441.
[9] M.Pinedo, *Scheduling: Theory, Algorithms, and Systems*, Springer, 3rd edition, 2008.