20th International Congress on Modelling and Simulation, Adelaide, Australia, 1–6 December 2013
www.mssanz.org.au/modsim2013

A branch-and-bound algorithm for scheduling unit processing time arc shutdown jobs to maximize flow through a transshipment node over time

N. Boland [a], **S. Kaur** [a]

[a] *School of Mathematical and Physical Sciences, University of Newcastle, Australia*
Email: simranjit.kaur@uon.edu.au

Abstract: Many real world complex problems can be viewed as networks with arc capacities, for example, rail networks, or supply chains, in which system throughput needs to be maximized. Arcs in such a network represent important components of the corresponding system, the condition of which may degrade over time. Maintenance of these components (arcs of the network) is important to maintain their productivity. But every maintenance activity incurs some loss of productivity as the arc will be unavailable during its maintenance. To obtain maximum throughput, it is important to select the schedule that leads to minimum loss of flow. In this paper we discuss optimization models for scheduling arc maintenance so as to maximize the throughput of the network, and focus on the case in which each maintenance task requires a single period and the network has a single transshipment node. We note that even this case is strongly NP-hard.

Mathematically 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 integral capacity vector $u = (u_a)_{a \in A}$. 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:

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

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

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

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

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

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

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

where $x_{ai} \geqslant 0$ 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.

We present a branch-and-bound algorithm called the "Partial-State algorithm" to solve the problem for *single transhipment node networks* i.e. networks with $|V| = 3$. Unit processing time of each job leads to formation of symmetries in the solution space. We include powerful symmetry breaking rules in the algorithm to make it more efficient. We provide an easily-computer combinatorial expression that is proved to give the value of LP-relaxation of the problem at each node of the branch-and-bound tree. We also provide another upper bound which is even stronger than the LP value at each node of the tree, and show how this improves the run time of the algorithm.

Keywords: *Network models, maintenance scheduling, mixed integer programming, branch and bound*

1 Introduction and Literature review

For many real world complex systems such as transport system, health systems, and power systems, maximising the throughput is a key concern. Such systems can often be viewed as networks with capacitated arcs that correspond to the equipments on which the system runs. The condition of these equipments may degrade over time which can lead to reduction in productivity and even to the failure of the network. Thus regular and efficient maintenance of these equipments, arcs of the network, is a fundamental requirement to ensure optimum level of productivity, continued reliability and smooth functioning of the network. In fact it is important not only to maintain the system but to do a proper scheduling of the maintenance jobs before it breakdowns.

The maintenances scheduling literature spans a variety of applications such as aircraft maintenance, process industry, vehicle fleet maintenance, railway track maintenance, power generation, pavement maintenance, highway maintenance, and production facilities. One can find several papers on maintenance scheduling with the objective of either minimizing the cost of maintenance or maximizing the system reliability. For a detail survey of the study carried out in this field, reader may refer to Oke (2011) and Budai-Balke et al. (2006). However there is little literature on finding a maintenance schedule with an objective of maximizing the throughput.

To the best of our knowledge, Boland et al. (2011, 2012a,b) initiated study on the the problem of scheduling the maintenance jobs on a given subset of arcs of a network so that the total flow is maximized in the network over a given time horizon with general processing times. In Boland et al. (2011, 2012a), 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. And in Boland et al. (2012b), the complexity of the general problem is established, and four local search heuristics are developed and compared. Further in Boland et al. (2013), the behaviour of the complexity of the problem is analysed corresponding to changes in important characteristics of the network when each maintenance jobs can be processed in a single period of time.

Several authors such as Ford and Fulkerson (1962), Fleischer (1999), Hajek and Ogier (1984), and Hoppe and Tardos (1994) have studied variations of the dynamic maximum flow problem with or without zero transit times. However none of these variants have a scheduling component. Gandham et al. (2008) studied the problem of scheduling communication over links with the goal of minimizing the number of partitions, whereas, for network maintenance, the goal is to minimize multi-period loss of flow across partitions. Tawarmalani and Li (2011) studied multiperiod maintenance scheduling over a tree network with a limit on the number of arcs that can be shut down in any one period and the objective is based on multicommodity flows (with origin-destination demands, but without arc capacities).

In this paper we focus on the problem of scheduling maintenance jobs so as to maximize the total flow in the network over time, particularly when each job has unit processing time and the network has only one transshipment node. The problem is shown to be strongly NP-hard in Boland et al. (2013). We present a branch-and-bound algorithm, called the "Partial-State algorithm", to solve the problem for *single transhipment node networks*. In section 2, we give the description of the algorithm. In Section 3 and 4 we discuss the techniques to find upper bound and lower bound at each node of the branch-and-bound tree respectively. In Section 5 computational results are discussed.

2 Partial-State Algorithm

Since we are focussing on networks with single transshipment node, throughout the paper, v denotes the transshipment node in the network. By an incoming and outgoing arc we mean an arc in $\delta^-(v)$ and $\delta^+(v)$ respectively. Let $J_1 = J \cap \delta^-(v)$ and $J_2 = J \cap \delta^+(v)$. For a subset $S \subseteq A$, $u(S)$ denotes the total capacity of arcs in the set S.

For an instance (N, J_1, J_2, T) of the problem, a feasible schedule defines partitions of the sets J_1 and J_2 as $J_1 = J_{11} \cup J_{12} \cup \cdots \cup J_{1T}$ and $J_2 = J_{21} \cup J_{22} \cup \cdots \cup J_{2T}$ where for each $i \in [T]$, J_{1i} and J_{2i} denote the set of incoming and outgoing arcs respectively scheduled to shut in time period i. Given a feasible schedule, the flow through the network in a period $i \in [T]$ is the minimum of the total capacity of arcs in the set $\delta^-(v) \setminus J_{1i}$ coming into v and the total capacity of arcs in the set $\delta^+(v) \setminus J_{2i}$ going out of v so that the total flow through the network over all time periods is given by $\sum_{i=1}^{T} \min\{u(\delta^-(v)) - u(J_{1i}), u(\delta^+(v)) - u(J_{2i})\}$. Therefore the problem of finding the schedule that gives maximum throughput of the network is equivalent to finding partitions $(J^*_{11}, J^*_{12}, \ldots, J^*_{1T})$ and $(J^*_{21}, J^*_{22}, \ldots, J^*_{2T})$ of J_1 and J_2 respectively such that

$$\sum_{i=1}^{T} \min\{u(\delta^-(v)) - u(J^*_{1i}), u(\delta^+(v)) - u(J^*_{2i})\} \geq \sum_{i=1}^{T} \min\{u(\delta^-(v)) - u(J_{1i}), u(\delta^+(v)) - u(J_{2i})\}$$

for all other feasible partitions $(J_{11}, J_{12}, \ldots, J_{1T})$ and $(J_{21}, J_{22}, \ldots, J_{2T})$ of J_1 and J_2 respectively.

In the Partial-State algorithm we build the partition $(J^*_{11}, J^*_{12}, \ldots, J^*_{1T})$ and $(J^*_{21}, J^*_{22}, \ldots, J^*_{2T})$ of J_1 and J_2 respectively in a recursive way starting from the state when no arc outages have been scheduled. The arcs in the job set J are arranged in non increasing order of their capacities. At k^{th} level of the algorithm we maintain the set of partial states in which outage of first k arcs in J have been scheduled and decisions for the remaining are still to be taken. A partial state vector at the k^{th} level, denoted by $\mathbf{z}$, stores the total capacity of arcs, incoming as well as outgoing, scheduled for shutdown in each time period $i \in [T]$ i.e. $\mathbf{z} = ((u(J_{11}), u(J_{21})), (u(J_{21}), u(J_{22})), \ldots, (u(J_{1T}), u(J_{2T})))$ where $\sum_{i=1}^{T}(|J_{1i}| + |J_{2i}|) = k$. For each partial state we compute an upper bound on the value of the total flow due to any feasible schedule for the problem that can be derived from it. We then use a heuristic to obtain a full state, a state where outages for all arcs in J have been scheduled, from the partial state. Note that the full state so obtained corresponds to a feasible solution for the problem instance. Hence the value of the total flow from this full state provides a global lower bound for the problem instance. Now if for a partial state the computed upper bound is less than the maximum lower bound obtained so far in the algorithm then it is not explored any further. Otherwise the partial state generates a set of new partial states, one for each time period in which the outage of the $(k+1)^{th}$ arc in J, is scheduled. At each level of the algorithm, the maximum upper bound obtained is recorded. The algorithm terminates if either the maximum lower bound obtained at the current level equals maximum upper bound at the previous level or all arcs in J have been scheduled. At termination, the maximum lower bound obtained gives the optimal flow for the instance.

Since the processing time of each job is one period therefore different schedules may lead to the same partial state resulting in formation of symmetries. We include the following two powerful symmetry detecting and breaking rules in the Partial-State algorithm to make it more efficient: 1) If for a partial state the computed upper bound is greater than or equal to the maximum lower bound found so far in the algorithm, then new partial states are generated from the current partial state by scheduling the outage of the next arc only in non symmetric periods. More precisely, the new partial states are generated from the partial state say $\mathbf{z} = ((u(J_{11}), u(J_{21})), (u(J_{21}), u(J_{22})), \ldots, (u(J_{1T}), u(J_{2T})))$ by making the choice to schedule the outage of next higher capacity arc in period i for each $i = 1, \ldots, \min(T, N_{\mathbf{z}} + 1)$ where $N_{\mathbf{z}}$ denotes the number of time periods in which at least one arc outage is scheduled, that is all those period for which $J_{1i} \cup J_{2i} \neq \phi$. 2) Since the time period are permutable (due to unit processing times of jobs) we sort each partial state created in the algorithm lexicographically. This helps in detecting the symmetries efficiently. At the current level of the algorithm we maintain the set of non symmetric partial states formed so far for the next level. A new partial state created from a partial state at current level is added to the next level only if it is not equal to any of the non-symmetric partial states saved so far for that level.

3 UPPER BOUND ON TOTAL FLOW FROM A PARTIAL STATE $\mathbf{z}$

Suppose we are at a partial state $\mathbf{z} = ((u(J_{11}), u(J_{21})), (u(J_{21}), u(J_{22})), \ldots, (u(J_{1T}), u(J_{2T})))$ where $\cup_{i=1}^{T} J_{1i} \subsetneq J_1$ and/or $\cup_{i=1}^{T} J_{2i} \subsetneq J_2$. Let $J'_1 = J_1 \setminus (\cup_{1=1}^{T} J_{1i})$ and $J'_2 = J_2 \setminus (\cup_{1=1}^{T} J_{2i})$ be the sets of incoming and outgoing arcs respectively for which outages are yet to be scheduled. For $i \in [T]$, let $M_{1i} = u(\delta^-(v)) - u(J_{1i})$ and $M_{2i} = u(\delta^+(v)) - u(J_{2i})$. The total flow through the network at $\mathbf{z}$ i.e. $F(\mathbf{z})$ is $\sum_{i=1}^{T} \min\{M_{1i}, M_{2i}\}$.

Scheduling outage of an arc in $J'_1 \cup J'_2$ could possibly lead to loss of flow through the network. If we assume that there will be no loss of flow by scheduling the outage of arcs in $J'_1 \cup J'_2$ then $\sum_{i=1}^{T} \min\{M_{1i}, M_{2i}\}$ serves as an obvious upper bound on total flow achievable from any full state derived from the partial state $\mathbf{z}$. However one can obtain a tighter upper bound as follows: Let $[T_1]$ be the set of time periods where $M_{1i} \geq M_{2i}$ and $[T_2]$ be the set of time periods where $M_{1i} < M_{2i}$. For each time period $i \in [T_1]$, there is a *spare incoming flow* of $M_{1i} - M_{2i}$. In order to keep the loss of flow due to outage of arcs in J'_1 to minimum, we assume that the total spare incoming flow $\sum_{i \in [T_1]}(M_{1i} - M_{2i})$ could be taken up with some arcs in J'_1 without affecting the total flow. So the total capacity of arcs in J'_1 that may affect the total flow is $\max\{u(J'_1) - \sum_{i \in [T_1]}(M_{1i} - M_{2i}), 0\}$. Similarly the total capacity of arcs in J'_2 that may affect the total flow is $\max\{u(J'_2) - \sum_{i \in [T_2]}(M_{2i} - M_{1i}), 0\}$. Since outage of all arcs in $J'_1 \cup J'_2$ have to be scheduled, we can subtract off the maximum of the above two expressions from $F(\mathbf{z})$ to obtain a tighter upper bound. Thus we can take

$$\bar{F}(\mathbf{z}) := \sum_{i=1}^{T} \min\{M_{1i}, M_{2i}\} - \max\{u(J'_1) - \sum_{i \in [T_1]}(M_{1i} - M_{2i}), u(J'_2) - \sum_{i \in [T_2]}(M_{2i} - M_{1i}), 0\}$$

as an upper bound on the total flow that can be achieved from any full state derived from $\mathbf{z}$. The expression for $\bar{F}(\mathbf{z})$ can further be simplified to

$$\bar{F}(\mathbf{z}) = \min\{\sum_{i=1}^{T} \min\{M_{1i}, M_{2i}\}, TM_1 - u(J_1), TM_2 - u(J_2)\}. \tag{8}$$

The maximum flow that can be achieved from any full state derived from $\mathbf{z}$ can be obtained by solving the mixed integer program $P(2)$ which is the mixed integer program $P(1)$ including the following additional constraint

$$y_{ai} = 0 \qquad \forall\ a \in J_{1i} \cup J_{2i}, i \in [T], \tag{9}$$

We prove that the upper bound $\bar{F}(\mathbf{z})$ given in (8) is the value of LP relaxation of $P(2)$, which we call as $LP(2)$.

Proposition 1. $\bar{F}(\mathbf{z})$ *is the optimal value of* $LP(2)$.

Proof. To prove the proposition we first prove that $\bar{F}(\mathbf{z})$ is an upper bound for $LP(2)$ and then construct a feasible solution for $LP(2)$ that attains objective value of $\bar{F}(\mathbf{z})$.

Let (x, y) be a feasible point of $LP(2)$. For each $i \in [T]$, inequalities (2), (3), (9) and the fact that for all $a \in J$, $y_{ai} \leq 1$ imply $\sum_{a\in\delta^-(v)} x_{ai} \leq M_{1i}$ and $\sum_{a\in\delta^+(v)} x_{ai} \leq M_{2i}$. Using (5), we get $\sum_{a\in\delta^+(v)} x_{ai} \leq \min\{M_{1i}, M_{2i}\}$. Thus $\sum_{i=1}^{T}\sum_{a\in\delta^+(v)} x_{ai} \leq \sum_{i=1}^{T} \min\{M_{1i}, M_{2i}\}$. Also,

$$\sum_{i=1}^{T}\sum_{a\in\delta^+(v)} x_{ai} \leq \sum_{a\in\delta^+(v)\setminus J_2}\sum_{i=1}^{T} u_a + \sum_{a\in J_2}\sum_{i=1}^{T} u_a y_{ai} = \sum_{a\in\delta^+(v)\setminus J_2} T u_a + \sum_{a\in J_2}(T-1)u_a = TM_2 - u(J_2).$$

Similarly, $\sum_{i=1}^{T}\sum_{a\in\delta^-(v)} x_{ai} \leq TM_1 - u(J_1)$. Again using (5), we get $\sum_{i=1}^{T}\sum_{a\in\delta^+(v)} x_{ai} \leq \min\{TM_1 - u(J_1), TM_2 - u(J_2)\}$. Hence $\bar{F}(\mathbf{z})$ is an upper bound for $LP(2)$.

We now show existence of a feasible solution (x, y) of $LP(2)$ that attains $\bar{F}(\mathbf{z})$ as the objective function value. We do this in three cases: (1) when $\bar{F}(\mathbf{z}) = \sum_{i=1}^{T} \min\{M_{1i}, M_{2i}\}$, (2) when $\bar{F}(\mathbf{z}) = TM_1 - u(J_1)$, and (3) when $\bar{F}(\mathbf{z}) = TM_2 - u(J_2)$. For brevity we provide the proofs for case (1) and (2), proof for case (3) can be done on similar lines. Without loss of generality we assume that $[T_1] = \{1, \ldots, k\}$, $[T_2] = \{k+1, \ldots, T\}$, $J_1' = \{a_1, a_2, \ldots, a_n\}$ and $J_2' = \{a_1', \ldots, a_m'\}$. For convenience, let $\triangle_i = M_{1i} - M_{2i}$ for all $i \in [T]$.

Case 1: $\bar{F}(\mathbf{z}) = \sum_{i=1}^{T} \min\{M_{1i}, M_{2i}\}$

$\bar{F}(\mathbf{z}) = \sum_{i=1}^{T} \min\{M_{1i}, M_{2i}\}$ implies that $u(J_1') - \sum_{i\in[T_1]} \triangle_i \leq 0$ and $u(J_2') - \sum_{i\in[T_2]}(-\triangle_i) \leq 0$.

We first construct the variables y_{ai} *with* $0 \leq y_{ai} \leq 1$ *for each* $a \in J$ *and* $i \in [T]$ *such that constraints (4) and (9) are satisfied.*

If $a \in J_{1i} \cup J_{2i}$ for some $i \in [T]$, take $y_{ai} = 0$ and $y_{at} = 1$ for all $t \in [T] \setminus \{i\}$ so that $\sum_{i=1}^{T} y_{ai} = T - 1$. For each arc $a_j \in J_1'$, choose largest integer f_j and smallest integer l_j with $1 \leq f_j, l_j \leq k$, such that $\sum_{i=1}^{j-1} u_{a_i} \geq \sum_{i=1}^{f_j-1} \triangle_i$ and $\sum_{i=1}^{j} u_{a_i} \leq \sum_{i=1}^{l_j} \triangle_i$. The previous two inequalities imply that $u_{a_j} \leq \sum_{i=f_j}^{l_j} \triangle_i$. This means that it is possible to shut arc a_j in periods $\{f_j, \ldots, l_j\}$ without incurring any loss of flow. Note that such a choice of f_j and l_j is possible for each arc $a_j \in J_1'$ because $u(J_1') \leq \sum_{i=1}^{k} \triangle_i$. Take

$$y_{a_j i} = \begin{cases} 1, & i \in [T] \setminus \{f_j, \ldots, l_j\}; \\ \{(\sum_{i=1}^{j} u_{a_i} - \sum_{i=1}^{f_j} \triangle_i)/u_{a_j}\}^+, & i = f_j\ ; \\ (u_{a_j} - \triangle_i)/u_{a_j}, & f_j < i < l_j; \\ \{(\sum_{i=1}^{l_j-1} \triangle_i - \sum_{i=1}^{j-1} u_{a_i})/u_{a_j}\}^+, & i = l_j. \end{cases} \tag{10}$$

so that $\sum_{i=1}^{T} y_{a_j i} = T - 1$. Similarly for each arc $a_j' \in J_2'$ and $i \in [T]$ one can define variables $y_{a_j' i}$ such that $\sum_{i=1}^{T} y_{a_j' i} = T - 1$. From the above discussion it follows that (4) is satisfied. Also it is easy to verify that $0 \leq y_{ai} \leq 1$ for each $a \in J$ and $i \in [T]$.

We now construct the variables $x_{ai} \geq 0$ for each $a \in A$ and $i \in [T]$ that satisfy constraints (2), (3) and (5).

From (10) it is clear that all arcs in the set J_1' are scheduled in periods $\{1, \ldots, l_n\} \subseteq [T_1]$. Similarly all arcs in the set J_2' will be scheduled in periods $\{k+1, \ldots, l_m'\} \subseteq [T_2]$. So for a period $i \in \{1, \ldots, l_n\} \subseteq [T_1]$, if we take $x_{ai} = u_a$ for each $a \in A \backslash J$ and $x_{ai} = u_a y_{ai}$ for each $a \in J$ then $\sum_{a \in \delta^-(v)} x_{ai} = \sum_{a \in \delta^+(v)} x_{ai} = M_{2i}$.

For a period $i \in \{l_n + 1, \ldots, k\}$, take $x_{ai} = u_a$ for each $a \in \delta^-(v) \setminus J_{2i}$ and $x_{ai} = 0$ for each $a \in J_{2i}$. Then $\sum_{a \in \delta^+(v)} x_{ai} = M_{2i}$. Now $\sum_{a \in \delta^-(v) \setminus J_{1i}} u_a = M_{1i} \geq M_{2i}$. So choose a subset A' of $\delta^-(v) \setminus J_{1i}$ and an arc $a' \in \delta^-(v) \setminus (J_{1i} \cup A')$ such that $\sum_{a \in A'} u_a \leq \triangle_i$ and $u_{a'} \geq \triangle_i - \sum_{a \in A'} u_a$. Such a choice of the set A' and arc a' is possible because $\sum_{a \in \delta^-(v) \setminus J_{1i}} u_a = M_{1i} > M_{1i} - M_{2i}$. Now if we take $x_{ai} = u_a$ for each $a \in \delta^-(v) \setminus (A' \cup J_{1i} \cup \{a'\})$; $x_{ai} = 0$ for each $a \in A' \cup J_{1i}$ and $x_{a'i} = u_{a'} - (\triangle_i - \sum_{a \in A'} u_a)$ then $\sum_{a \in \delta^-(v)} x_{ai} = M_{2i} = \sum_{a \in \delta^+(v)} x_{ai}$.

Using a similar discussion as above, it can be easily proved that for period $i \in \{k+1, \ldots, T\}$, $\sum_{a \in \delta^-(v)} x_{ai} = \sum_{a \in \delta^+(v)} x_{ai} = M_{1i}$. Thus (5) is satisfied. Also is easy to see from the above construction that $0 \leq x_{ai} \leq u_a$ for each $a \in A \setminus J, i \in [T]$, and $0 \leq x_{ai} \leq u_a y_{ai}$ for each $a \in J, i \in [T]$. Hence (x, y) is feasible for $LP(2)$ with objective function value $\sum_{i \in [T_1]} M_{2i} + \sum_{i \in [T_2]} M_{1i} = \sum_{i=1}^{T} \min\{M_{1i}, M_{2i}\}$.

Case 2 $\bar{F}(\mathbf{z}) = TM_1 - u(J_1)$

$\bar{F}(\mathbf{z}) = TM_1 - u(J_1)$ implies $u(J_1') - \sum_{i \in [T_1]} \triangle_i \geq 0$ and $u(J_1') - \sum_{i \in [T_1]} \triangle_i \geq u(J_2') + \sum_{i \in [T_2]} \triangle_i$. Two cases can arise: either (i) $u(J_2') \leq \sum_{i \in [T_2]}(-\triangle_i)$, or (ii) $u(J_2') > \sum_{i \in [T_2]}(-\triangle_i)$. Here we will prove the result for case (i), proof for case (ii) can be done on similar lines.

Suppose $u(J_1') - \sum_{i \in [T_1]} \triangle_i \geq 0$ and $u(J_2') \leq \sum_{i \in [T_2]}(-\triangle_i)$. Since $u(J_1') = \sum_{j=1}^{k} u_{a_j} \geq \sum_{i \in [T_1]} \triangle_i$, we can choose j^* the highest index such that $\sum_{j=1}^{j*} u_{a_j} \leq \sum_{i=1}^{k} \triangle_i$. Now since $\sum_{j=1}^{j*} u_{a_j} \leq \sum_{i=1}^{k} \triangle_i$, for each $j = 1, \ldots, j^*$ and $i \in [T]$ we can define variables $y_{a_j i}$ as has been defined in Case 1. And for each $j = j^* + 1, \ldots, n$ take $y_{a_j i} = 1$ for all $i = [T] \setminus \{k\}$ and $y_{a_j k} = 0$. It is easy to see that $\sum_{i=1}^{T} y_{ai} = T - 1$ for each $a \in J_1'$. Also since $u(J_2') \leq \sum_{i \in [T_2]}(-\triangle_i)$, for each $a \in J_2'$ and $i \in [T]$ we can define y_{ai} as have been defined in Case 1. And for arcs in set $J_{1i} \cup J_{2i}$, take $y_{ai} = 0$ and $y_{at} = 1$ for all $t \in [T] \setminus \{i\}$.

For each $a \in A$ and $i \in [T] \setminus \{k\}$, define x_{ai} as done in Case 1. For $i = k$, take $x_{ak} = u_a y_{ak}$ for each $a \in J_1$ and $x_{ak} = u_a$ for each $a \in \delta^-(v) \setminus J_1$. Then $\sum_{a \in \delta^-(v)} x_{ak} = M_{2k} - (u(J_1') - \sum_{i=1}^{k} \triangle_i)$. Also choose a subset A' of $A_2 \setminus J_{2i}$ and an arc $a' \in A_2 \setminus (J_{2i} \cup A')$ such that $\sum_{a \in A'} u_a \leq u(J_1') - \sum_{i \in [T_1]}(M_{1i} - M_{2i})$ and $\sum_{a \in A'} u_a + u_a' \geq u(J_1') - \sum_{i \in [T_1]}(M_{1i} - M_{2i})$. Now if we take $x_{ai} = u_a$ for each $a \in \delta^+(v) \setminus (A' \cup J_{2k} \cup \{a'\})$; $x_{ai} = 0$ for each $a \in A' \cup J_{2k}$ and $x_{a'i} = u_{a'} - (u(J_1') - \sum_{i=1}^{k} \triangle_i - \sum_{a \in A'} u_a)$ then $\sum_{a \in \delta^+(v)} x_{ak} = M_{2k} - (u(J_1') - \sum_{i=1}^{k} \triangle_i))$.

The constructed point (x, y) is feasible to $LP(2)$ with the objective function value of $\sum_{i \in [T_1]} M_{2i} + \sum_{i \in [T_2]} M_{1i} - (u(J_1') - \sum_{i \in [T_1]} \triangle_i) = TM_1 - u(J_1)$. □

We now present an upper bound for a partial state $\mathbf{z}$ which is tighter than $\bar{F}(\mathbf{z})$. Observe while evaluating the expression for $\bar{F}(\mathbf{z})$ we allowed arcs in sets J_1' and J_2' to take up the *spare incoming* and *spare outgoing flow* respectively. In this process we allow the jobs to split into more than one period (in order to occupy the *spare flow* in best possible way). However this should not happen as an arc can be shut in exactly one period. Without loss of generality we can assume that $u_{a_1} \geq u_{a_2} \geq \cdots \geq u_{a_n}$ and $u_{a_1'} \geq u_{a_2'} \geq \cdots \geq u_{a_m'}$. We can also assume that the time periods in the sets $[T_1]$ and $[T_2]$ are arranged such that $\triangle_1 \geq \triangle_2 \geq \cdots \geq \triangle_k$ and $-\triangle_{k+1} \geq -\triangle_{k+2} \geq \cdots \geq -\triangle_T$ respectively. To obtain a tighter upper bound on the total flow achievable from the partial state $\mathbf{z}$, we use the fact that the arcs $\{a_j : 1 \leq j \leq k_1\}$ and $\{a_j' : 1 \leq j \leq k_2\}$ should be shut in exactly one period where k_1 and k_2 are the largest integers less than or equal to k and T respectively such that for all $i = 1, \ldots, k_1$ $u_{a_i} \geq \triangle_1$ and $i = k+1, \ldots, k_2$ $u_{a_i'} \geq -\triangle_{k+1}$ respectively. We need to decide in which periods the jobs on the arcs $\{a_i : 1 \leq i \leq k_1\}$ and $\{a_i' : k+1 \leq i \leq k_2\}$ must be scheduled to obtain a tighter upper bound. In the following proposition we give a bound better than $\bar{F}(\mathbf{z})$ and also prove that it is optimal to shut the arcs in the set $\{a_i : 1 \leq i \leq k_1\}$ in first k_1 periods in $[T_1]$ and arcs in the set $\{a_i' : k+1 \leq i \leq k_2\}$ in first k_2 periods in $[T_2]$ respectively.

Proposition 2. *An upper bound on the flow that can be achieved from any full state obtained from a partial* *state* $\mathbf{z}$ *is given by*

$$\sum_{i=1}^{T}\min\{M_{1i},M_{2i}\}-\max\{\sum_{i=1}^{k_1}(u_{a_i}-\triangle_i)+(\sum_{i=k_1+1}^{n}u_{a_i}-\sum_{i=k_1+1}^{k}\triangle_i)^+,\ \sum_{i=k+1}^{k_2}(u_{a'_i}+\triangle_i)+(\sum_{i=k_2+1}^{m}u_{a'_i}+\sum_{i=k_2+1}^{T}\triangle_i)^+\}.$$

Proof. We first claim that the loss of incoming flow over the time horizon for any full state obtained from $\mathbf{z}$ is at least $\sum_{i=1}^{k_1}(u_{a_i}-\triangle_i)+(\sum_{i=k_1+1}^{m}u_{a_i}-\sum_{i=k_1+1}^{k}\triangle_i)^+$. To prove the claim we show that for any partition of the set $[T_1]=S_1\cup S_2$ with $|S_1|=k_1$,

$$\sum_{i=1}^{k_1}(u_{a_i}-\triangle_i)+(\sum_{i=k_1+1}^{n}u_{a_i}-\sum_{i=k_1+1}^{k}\triangle_i)^+\leq\sum_{i=1}^{k_1}u_{a_i}-\sum_{i\in S_1}\triangle_i+(\sum_{i=k_1+1}^{n}u_{a_i}-\sum_{i\in S_2}\triangle_i)^+$$

The claim holds trivially when either $\sum_{i=k_1+1}^{m}u_{a_i}-\sum_{i=k_1+1}^{k}\triangle_i<0$ or $\sum_{i=k_1+1}^{m}u_{a_i}-\sum_{i=k_1+1}^{k}\triangle_i\geq 0$ and $\sum_{i=k_1+1}^{m}u_{a_i}-\sum_{i\in S_2}\triangle_i\geq 0$. Let $\sum_{i=k_1+1}^{m}u_{a_i}-\sum_{i=k_1+1}^{k}\triangle_i\geq 0$ and $\sum_{i=k_1+1}^{m}u_{a_i}-\sum_{i\in S_2}\triangle_i<0$. Then

$$\sum_{i=1}^{m}(u_{a_i}-\sum_{i=1}^{k}\triangle_i)=\sum_{i=1}^{k_1}u_{a_i}-\sum_{i\in S_1}\triangle_i+(\sum_{i=k_1+1}^{n}u_{a_i}-\sum_{i\in S_2}\triangle_i)<\sum_{i=1}^{k_1}u_{a_i}-\sum_{i\in S_1}\triangle_i.$$

Hence the claim. It can be shown similarly that the loss of outgoing flow over the time horizon for any full state obtained from $\mathbf{z}$ is at least $\sum_{i=k+1}^{k_2}(u_{a'_i}+\triangle_i)+(\sum_{i=k_2+1}^{m}u_{a'_i}+\sum_{i=k_2+1}^{T}\triangle_i)^+$. Since outage of all arcs in J'_1 and J'_2 have to be scheduled we can subtract the maximum of $\sum_{i=1}^{k_1}(u_{a_i}-\triangle_i)+(\sum_{i=k_1+1}^{m}u_{a_i}-\sum_{i=k_1+1}^{k}\triangle_i)^+$ and $\sum_{i=k+1}^{k_2}(u_{a'_i}+\triangle_i)+(\sum_{i=k_2+1}^{m}u_{a'_i}+\sum_{i=k_2+1}^{T}\triangle_i)^+$ from $\sum_{i=1}^{T}\min\{M_{1i},M_{2i}\}$ to get an upper bound on the total flow. □

4 Lower bound on total flow for a partial state z

To obtain a lower bound on the flow achievable from a partial state $\mathbf{z}$ we construct a feasible schedule for the problem from $\mathbf{z}$ using a heuristic by keeping all jobs in the partial state fixed. We schedule each remaining arc outage in a period where it fits in the 'best-way' based on the *spare incoming* or *spare outgoing flow* available. Suppose we are scheduling the outage of an arc a in J' in the the modified partial state $\mathbf{z}$ (assuming we have already scheduled outages of some of the remaining arcs) . Note that if a is an incoming arc and there are time periods in the set $[T_1]$ for which the incoming flow in the period is greater than the outgoing flow in that period by at least the capacity of arc a i.e. periods for which $\triangle_i\geq u(a)$, then we can schedule the outage of a in any of these period without incurring any loss of flow due to this action. In such a case we schedule the outage of arc a in the period which has the minimum value of $\triangle_i$ and update the partial state $\mathbf{z}$. One can also note that if a is an incoming arc and for each period in the set $[T_1]$, $\triangle_i<u(a)$ then we will get some loss of flow by scheduling the outage of a in any period and the minimum loss that we can get is $u(a)-\max\{\triangle_i : i\in[T_1]\}^+$. So we schedule the outage of arc a accordingly in the period for which $\triangle_i$ is maximum and the partial state is updated. Scheduling of a outage of an outgoing arc is done using the same idea as the incoming arcs. The result is a feasible schedule for which the total flow yields a lower bound on the optimal value.

5 Computational Results

To compare the performance of Partial-State algorithm with the CPLEX solver 400 instances were randomly generated. Each of these instances was solved using the Partial-State algorithm and also by CPLEX using the mixed integer formulation for the problem $P(1)$. Out of 400 instances 250 were solved in less than 10 seconds by both the PS algorithm and CPLEX. We analysed the run times of the remaining 150 instances. The cutoff time limit was taken to be 7200 seconds.

To compare the performance of the Partial-State algorithm on an instance I from these 150 instances of the problem, we use the performance ratio, $r(I,PS)$, which is the log_2 value of ratio of the time taken to solve I by the PS algorithm to the minimum of the time taken by the PS algorithm and CPLEX to solve I. Similarly we define the performance ratio $r(I,CPLEX)$ for CPLEX. The percentages of instances having performance ratio $r(I,PS)$ and $r(I,CPLEX)$ less than or equal to $\tau\in\mathbb{R}$ are taken as a measure to obtain an overall assessment of the PS algorithm and CPLEX respectively. The performance profile for these 150 instances is given in figure 1.

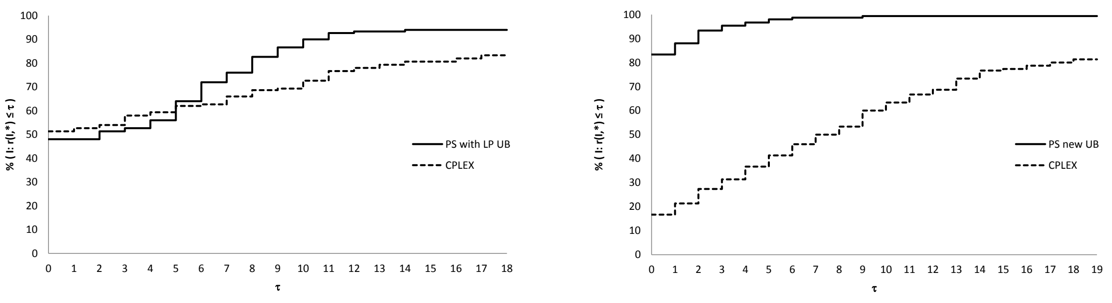

Figure 1. Comparison of PS Algorithm with CPLEX

We can observe from figure 1 that the new tighter upper bound make the Partial-State algorithm more efficient by reducing the run times significantly. With the improved upper bound 137 out of 150 instances were solved by the Partial-State algorithm in less than 10 seconds and only one was not solved within the cut off time of 7200 seconds. The reason behind the better performance of the Partial-State algorithm are the efficient ways through which the symmetries are removed from the solution space by the algorithm and the strength of the bounds used. In fact for almost 50% of the instances the lower bound evaluated at the root node is optimal and for the remaining 50% it is at least 0.8 of the optimal flow value.

REFERENCES

Boland, N., T. Kalinowski, R. Kapoor, and S. Kaur (2013). Scheduling unit processing time arc shutdown jobs to maximize network flow over time: complexity results. *arXiv preprint arXiv:1306.4917*.

Boland, N., T. Kalinowski, H. Waterer, and L. Zheng (2012a). Mixed integer programming based maintenance scheduling for the hunter valley coal chain. *Journal of Scheduling*, 1–11.

Boland, N., T. Kalinowski, H. Waterer, and L. Zheng (2012b). Scheduling arc maintenance jobs in a network to maximize total flow over time. *Discrete Applied Mathematics* (0), –.

Boland, N., T. Kalinowski, H. Waterer, L. Zheng, et al. (2011). An optimisation approach to maintenance scheduling for capacity alignment in the hunter valley coal chain— nova. the university of newcastle's digital repository. The Australasian Institute of Mining and Metallurgy.

Budai-Balke, G., R. Dekker, and R. P. Nicolai (2006). A review of planning models for maintenance and production. Econometric Institute Report EI 2006-44, Erasmus University Rotterdam, Econometric Institute.

Fleischer, L. (1999). Universally maximum flow with piecewise-constant capacities. In G. Cornujols, R. Burkard, and G. Woeginger (Eds.), *Integer Programming and Combinatorial Optimization*, Volume 1610 of *Lecture Notes in Computer Science*, pp. 151–165. Springer Berlin Heidelberg.

Ford, L. and D. Fulkerson (1962). *Flows in Networks*. Rand Corporation research study. University Press.

Gandham, S., M. Dawande, and R. Prakash (2008). Link scheduling in wireless sensor networks: Distributed edge-coloringrevisited. *Journal of Parallel and Distributed Computing 68*(8), 1122 – 1134.

Hajek, B. and R. G. Ogier (1984). Optimal dynamic routing in communication networks with continuous traffic. *Networks 14*(3), 457–487.

Hoppe, B. and E. Tardos (1994). Polynomial time algorithms for some evacuation problems. In *Proceedings of the fifth annual ACM-SIAM symposium on Discrete algorithms*, SODA '94, Philadelphia, PA, USA, pp. 433–441. Society for Industrial and Applied Mathematics.

Oke, S. (2011). Maintenance scheduling: Description, status and future directions. *The South African Journal of Industrial Engineering 15*(1).

Tawarmalani, M. and Y. Li (2011). Multi-period maintenance scheduling of tree networks with minimum flow disruption. *Naval Research Logistics (NRL) 58*(5), 507–530.