Simple bounds and greedy algorithms for decomposing a flow into a minimal set of paths

B. Vatinlen a,*, F. Chauvet a, P. Chrétienne b, P. Mahey c

a Bouygues Telecom Research, Centre d’Affaires La Boursidiere, 92355 Le Plessis Robinson, France
b P. and M. Curie University, LIP6 Laboratory 8, rue du capitaine Scott, 75015 Paris, France
c LIMOS Laboratory, Campus de Clermont-Ferrand/Les Cezeaux, BP 10125, 63173 Aubiere Cedex, France

Received 9 September 2004; accepted 3 May 2006
Available online 1 November 2006

Abstract

Given arbitrary source and target nodes s, t and an s–t-flow defined by its flow-values on each arc of a network, we consider the problem of finding a decomposition of this flow with a minimal number of s–t-paths. This problem is issued from the engineering of telecommunications networks for which the task of implementing a routing solution consists in integrating a set of end-to-end paths. We show that this problem is NP-hard in the strong sense and give some properties of an optimal solution. We then propose upper and lower bounds for the number of paths in an optimal solution. Finally we develop two heuristics based on the properties of a special set of solutions called saturating solutions.

Keywords: Network; Flow; Path; Heuristics

1. Introduction

The problem we study in this paper concerns the integration of a routing solution in a telecommunication core network. Indeed, in networks, several demands have to be forwarded from end to end while respecting some constraints such as capacity, quality of service, . . . An optimal routing solution can be represented in two ways: the arc/node versus the arc/path formulation. The arc/node formulation gives the distribution of each demand on each arc. The arc/path formulation gives the distribution of each demand on the paths supporting that demand. Arc/path solutions which correspond to the same arc/node solution are equivalent considering the routing criterion. Our problem is motivated by two main arguments: the first one is that network managers only consider arc/path solutions, the second is that the overall cost of integrating an arc/path solution grows quite rapidly with the number of paths of that solution. Another feature of the problem is that the flow that must be integrated by the network manager is the optimal solution of an optimization problem over a subset
of feasible flows whose objective function is increasing with respect to the flow values. In practice, that property could allow us to restrict our problem to \( s-t \)-flows (called \((s \rightarrow t)\)-flows) for which all decompositions do not contain circuits. However, in this paper, we will mainly consider the general combinatorial problem.

It is well known that there are many decompositions of a given \( s-t \)-flow into a subset of \( s-t \)-paths and circuits. Each classical primal max-flow algorithm [1] provides such a decomposition of the maximal flow. Unfortunately, in the general case, that decomposition is not minimal with respect to the number of paths. A variant of the problem called the \( k \)-splittable maximum flow problem has been studied in [2] where one must find a maximum flow distributed on at most \( k \) paths. To solve it, the authors basically use iteratively a primal max-flow algorithm but cannot prove that there is no solution with less than \( k \) paths.

In this paper, we consider the special case of only one demand from a source node \( s \) to a target node \( t \) in a network \( G \). Given an \( s-t \)-flow \( r \) of \( G \) (i.e. an arc/node solution), we study the problem (called MSPFD for Minimal Set of Paths of a Flow Decomposition) of finding the minimal number of \( s-t \)-paths of a decomposition of \( r \). The special case of MSPFD restricted to \((s \rightarrow t)\)-flows, will be denoted by \( r \)-MSPFD.

Section 2 gives a formal description of MSPFD, presents some properties of its optimal solutions and introduces particular solutions called saturating solutions. Section 3 proposes simple upper and lower bounds for the minimal number of paths of an instance of \( r \)-MSPFD. The last section develops two heuristics to solve this problem.

2. The problem MSPFD

Let \( G = (V, E) \) be a directed graph with \( n \) nodes and \( m \) arcs. Let \( s, t \) be arbitrary source and target nodes of \( G \) and let \( r \in (\mathbb{R}^+)^m \) be a positive \( s-t \)-flow of \( G \). The flow value of \( r \) on arc \( e \) is denoted by \( r_e \). The incidence vector of a path \( P \) of \( G \) is denoted by \( \chi_P \). Recall that \((P, C, f, g)\), where

1. \( P = \{P_1, \ldots, P_p\} \) is a set of simple \( s-t \)-paths,
2. \( C = \{C_1, \ldots, C_c\} \) is a set of simple circuits,
3. \( f_1, \ldots, f_p \) and \( g_1, \ldots, g_c \) are strictly positive numbers,

is a decomposition of \( r \) if \( r = \sum_{i=1}^p f_i \chi_{P_i} + \sum_{j=1}^c g_j \chi_{C_j} \).

\( I = (G, s, t, r) \) is a generic instance of the MSPFD problem. The candidate solutions of \( I \) are the decompositions of \( r \) and an optimal solution is a decomposition such that \( p \) (the number of its \( s-t \)-paths) is minimal.

Since only the number of \( s-t \)-paths is taken into account in the evaluation of a solution, we will assume that the graph \( G \) has no arc with input node \( t \) and output node \( s \).

2.1. The special case \( r \)-MSPFD

\( r \)-MSPFD is the special case of MSPFD restricted to \((s \rightarrow t)\)-flows. We give below two properties of \((s \rightarrow t)\)-flows, the first is a characterization and the second explains why, in practice, the combinatorial problem that must be solved is \( r \)-MSPFD.

Recall that the support graph of \( r \), denoted by \( G(r) \), is the subgraph of \( G \) defined by the arcs \( e \) such that \( r_e > 0 \).

**Proposition 1.** \( r \) is an \((s \rightarrow t)\)-flow if and only if its support graph \( G(r) \) has no circuit.

**Proof.** The sufficient condition is obvious. Assume now that \( r \) is a circuit of \( G \) where \( r \) is an \((s \rightarrow t)\)-flow and let \( \xi > 0 \) be the minimum amount of flow of \( r \) on the arcs of \( G \). Thus \((r - \xi \mu) \) is a flow whose support graph is a partial graph of \( G \). Thus we have \( r - \xi \mu = \sum_{i=1}^p f_i \chi_{P_i} + \sum_{j=1}^c g_j \chi_{C_j} \) for a decomposition \((P, C, f, g)\) of \( r - \xi \mu \). So \( r \) is not an \((s \rightarrow t)\)-flow, what is a contradiction.

Let us now give a small example showing that, given an instance \((G, s, t, r)\) of MSPFD, it is not necessarily a good idea to choose first to extract the circuits of \( G \) and then to solve the reduced instance of \( r \)-MSPFD. Consider the example illustrated in Fig. 1. After extracting the circuit \((a, b, a)\), three paths are necessary in a min-
imal decomposition of the reduced instance while the two paths \((s,a,b,t)\) and \((s,b,a,t)\) make a decomposition of the initial instance.

### 2.2. Complexity aspects

In [2], the authors have shown that it is strongly NP-hard to approximate instances of the maximum 2-split-table \(s-t\)-flow problem with an approximation ratio strictly better than \(\frac{2}{3}\). Moreover, they used the same construction to prove that MSPFD is strongly NP-hard. In Appendix A, we use a rather simple proof to show that r.MSPFD is strongly NP-hard.

**Proposition 2.** r.MSPFD is NP-hard in the strong sense.

An instance \((G,s,t,r,\mathcal{C})\) of problem Minimum Set of Given Paths for a Flow (MSGPF) is built from an arbitrary instance \((G,s,t,r)\) of MSPFD by adding in the input a subset \(\mathcal{C}\) of simple paths from \(s\) to \(t\). The problem is now to find a decomposition \((P,C,f,g)\) such that \(p\) is minimum and \(P \subseteq \mathcal{C}\). If r.MSGPF is the special case of MSGPF restricted to \((s \rightarrow t)\)-flows, it is straightforward to show that

**Proposition 3.** r.MSGPF is NP-hard in the strong sense.

### 2.3. Properties of optimal solutions

Let \((G,s,t,r)\) be an arbitrary instance of MSPFD. It is easy to observe that an optimal solution always exists. However, as illustrated in Fig. 2, an optimal solution is not necessarily unique.

Let \((P,C,f,g)\) be an optimal solution of the instance \((G,s,t,r)\) of MSPFD. The following property is a necessary condition for optimality which will be useful for deriving heuristics.

**Proposition 4.** The incidence vectors of the paths of an optimal solution of the instance \((G,s,t,r)\) of MSPFD are linearly independent.

**Proof.** The proof may be seen as a consequence of Caratheodory’s theorem. Since \(r = \sum_{i=1}^{p} f_i x_{P_i} + \sum_{j=1}^{c} g_j x_{C_j}\), \(r\) is a conic combination of the \(p + c\) vectors \(x_{P_i}, 1 \leq i \leq p\) and \(x_{C_j}, 1 \leq i \leq c\). By Caratheodory’s theorem (see [R.T. Rockafellar, Convex Analysis, 1970, Chapter 17 and Corollary 17.1.2]), \(r\) can be expressed by a conic combination of \(p'\) vectors \(x_{P_i}\) and \(c'\) vectors \(x_{C_j}\) such that \(p' + c' \leq m\) and these \(p' + c'\) vectors are linearly independent. If the vectors \(x_{P_i}, 1 \leq i \leq p\) were dependent, the same reduction as in the classical proof of Caratheodory’s theorem would eliminate any linearly dependent generator, contradicting the optimality of the solution \((P,C,f,g)\).
2.4. Saturating solutions

We now define a special subset of solutions called saturating solutions. Although these solutions do not make a dominant subset as shown in Proposition 8, we derive some properties that are useful when searching for efficient heuristics.

**Definition 5.** Let \((P, C, f, g)\) be a solution of the instance \((G, s, t, r)\) of MSPFD. A path \(P_i \in P\) is called a saturating path if there is at least an arc \(e\) of \(P_i\) such that \(r_e = f_i\). The arcs such that \(r_e = f_i\) are said to be saturated.

**Definition 6.** Let \((P, C, f, g)\) be a solution with \(p\) paths of the instance \((G, s, t, r)\) of MSPFD. \((P, C, f, g)\) is a saturating solution if one of the two following conditions is true:

- \(p = 0\),
- \(p > 0\) and there is a saturating path \(P_i \in P\) such that if \(P' = P/P_i\) and \(G'\) is the support graph of \(r - f_i\), then \((P', C, f', g)\) is still a saturating solution of the input \((G', s, t, r - f_i)\) of MSPFD.

Note that the well-known flow decomposition proposed by Ahuja in [1] provides a decomposition of an arbitrary \(s-t\)-flow in at most \(m\) simple \(s-t\) paths and \(n\) simple circuits. The next property is a necessary condition for optimality.

**Proposition 7.** Let \((P, C, f, g)\) be a saturating solution. The matrix \(M(P)\) whose column vectors are \(\mathcal{I}_{P_1}, \ldots, \mathcal{I}_{P_p}\) has rank \(p\).

**Proof.** Assume that \((P_1, \ldots, P_p)\) is the ordered list of the paths in a saturating solution \((P, C, f, g)\). From the definition of a saturating solution, it is clear that, if \(e_j\) is a saturated arc of path \(P_j\), then \(e_p \not\in P_1, \forall i = j + 1, \ldots, p\). So, if we re-arrange the rows of \(M(P)\) such that the first \(p\) rows are associated with saturated arcs for \(P_1, \ldots, P_p\), we can see that \(M(P)\) is lower triangular and has thus rank \(p\).

The next property, whose proof is given in Appendix B, uses the input shown in Fig. 3 to prove that, contrary to common intuition, the saturating solutions are not dominant for MSPFD.

**Proposition 8.** The saturating solutions are not dominant for MSPFD.

The six paths solution \(P_0, \ldots, P_5\), with \((f_0, \ldots, f_5) = (32, 16, 8, 4, 2, 1)\), of the input of Fig. 3 shown in Fig. 4, where each arc is labelled with the indexes of the two paths of the solution that use that arc, is a unique non saturating optimal solution.

![Fig. 3. An input with a unique non saturating optimal solution.](image-url)

![Fig. 4. A non saturating optimal solution.](image-url)
The preceding example may certainly not be the smallest one to prove that saturating solutions are not dominant for MSPFD. However, using the same graph and the $r$-values shown in Fig. 5, it can be shown, following a line of argument quite close to the proof of Proposition 8, that this particular instance has a unique non-saturating fractional optimal solution. This shows indeed that integer solutions are not dominant for the special case of MSPFD where the $r$-values are integer.

3. Bounding the minimal number of paths

This section focuses on bounding the optimal number of paths for the acyclic problem r.MSPFD. Let $I = (G, s, t, r)$ be an input of r.MSPFD and let $N^*$ be the minimal number of paths in a solution of $I$. Two lower bounds and one upper bound for $N^*$ are provided.

Proposition 9

\[ N^* \leq m - n + 2. \]

Proof. Let $(P, C, f, g)$ be an optimal solution of $(G, s, t, r)$. Let $G^0$ be the graph we get by adding a backward arc $u_0$ from $t$ to $s$. Thus, each path of $P$ in $G$ becomes a simple circuit in $G^0$. Since the paths of $P$ are independent (see Proposition 4), the corresponding circuits in $G^0$ are also independent. Finally, since the maximum number of independent circuits in $G^0$ is equal to \( \left\lceil \frac{m}{n-1} \right\rceil \) [4], we get that $N^* \leq m - n + 2$. \( \square \)

Note that this bound also applies to an instance of MSPFD.

The first lower bound is issued from an easy degree argument.

Proposition 10

\[ N^* \geq \left\lceil \frac{m}{n-1} \right\rceil. \]

Proof. Let $d^+_{\text{max}}$ be the maximum outdegree of $G$. Since the outdegree of $t$ is null, we clearly have $d^+_{\text{max}} \geq \left\lceil \frac{m}{n-1} \right\rceil$. Assume now that node $x$ has output degree $d^+_{\text{max}}$. The result follows since the outgoing arcs from $x$ must be contained in distinct paths of an optimal solution. \( \square \)

From Propositions 9 and 10 we get the following Corollary, which will be useful for the evaluation of the performance of the heuristics.

Corollary 11. Let $(P, C, f, g)$ be a solution of the instance $(G, s, t, r)$ of r.MSPFD such that the paths of $P$ are independent. Then we have \( \frac{n}{N^*} \leq n - 1 - \frac{d^+_{\text{max}}}{m}. \)

Proof. Since the paths of $P$ are independent, we get, following the lines of Proposition 9, that $p \leq m - n + 2$ and from Proposition 10, we have $\frac{n}{N^*} \leq \frac{n-1}{m}$. We thus get $\frac{n}{N^*} \leq n - 1 - \frac{d^+_{\text{max}}}{m}$. \( \square \)

The second lower bound is obtained by embedding the solutions set of the input $I = (G, s, t, r)$ into the set of arc-coverings of the support graph $G(r)$ where an arc-covering of $G(r)$ is a set of simple $s$–$t$-paths such that each arc of $G(r)$ belongs to at least one path. Thus, the lower bound which is the minimal number of $s$–$t$-paths of an arc-covering of $G(r)$, may be computed in polynomial time using a minimum-cost $s$–$t$-flow algorithm on
an associated network $N(r)$. Indeed, $N(r)$ is obtained from $G(r)$ by imposing a lower bound of 1 on each arc flow and an infinite capacity on each arc and by giving the cost 1 to each arc with input node $s$ and a null cost to any other arc.

**Proposition 12.** The value of a minimum-cost flow of $N(r)$ is at most $N^*$.

**Proof.** For any solution $(P, C, f, g)$ of $I = (G, s, t, r)$, we know that $P$ is an arc-covering of $G(r)$ and that $\sum_{i=1}^p I_{P_i}$ is a feasible flow of $N(r)$ whose cost is equal to $p$. □

The previous lower bound, which does not take into account the $r$-values can be very bad for some special inputs. Consider for example the input illustrated in Fig. 6. Clearly the minimal number of $s$–$t$-paths of an arc-covering of $G(r)$ is 2 for this input. However since the flow $r$ takes $m(=2L)$ distinct values, the number of paths in any solution is at least $1 + \log_2 L$. Thus the lower bound may be arbitrary bad. Note however that this bound is tight by simply considering the same graph as in Fig. 6 and unit $r$-values.

The second lower bound is issued from the notion of directed $s$–$t$-cut. Recall that a directed $s$–$t$-cut is an $s$–$t$-cut whose all arcs are outgoing arcs. For example, $\{(s, a), (s, b)\}$ and $\{(a, t), (b, t)\}$ are two directed $s$–$t$-cuts of the graph of Fig. 1.

**Proposition 13.** The maximum cardinality of a directed $s$–$t$-cut is at most equal to $N^*$.

**Proof.** Let $(P, C, f, g)$ be an optimal solution of the instance $(G, s, t, r)$ and let $H$ be an arbitrary directed $s$–$t$-cut. Each arc $h$ of $H$ belongs to at least one path, say $P(h)$, of $P$. Moreover, the paths $P(h)$, $h \in H$ are pairwise distinct. So Card$(H) \leq N^*$. □

Note that the two preceding lower bounds are the same since it is known [5] that for a directed acyclic $s$–$t$-graph, the minimum number of paths of an arc covering is equal to the maximum cardinality of a directed $s$–$t$-cut. The weakness of the preceding lower bounds comes from the fact that they do not take the $r$-values into account. However the lower bound proposed by Proposition 13 may be extended. Let us define the capacity $\rho(P)$ of an $s$–$t$-path $P$ of an instance $(G, s, t, r)$ of r.MSPFD as the minimum $r$-value of $P$. Moreover, let $\rho(u, v)$ denote the maximal capacity of a simple path from node $u$ to node $v$. The following property provides a lower bound of $N^*$.

**Proposition 14.** Let $H$ be an arbitrary directed $s$–$t$-cut. For any $h \in H$, let $\pi(h) = \min\{\rho(s, u), \rho(v, t)\}$ where $u, v$ are the two end nodes of $h$. Then we have: $\sum_{h \in H} \frac{\pi(h)}{\rho(h)} \leq N^*$.

**Proof.** Let $(P, C, f, g)$ be an optimal solution of the instance $(G, s, t, r)$, $H$ be an arbitrary directed $s$–$t$-cut and $h$ be an arbitrary arc in $H$. If $P_i$ contains $h$, then, from the definition of $\rho(s, u)$, we have $f_i \leq \rho(P_i) \leq \rho(s, u)$ and from the definition of $\rho(v, t)$, we have $f_i \leq \rho(P_i) \leq \rho(v, t)$. We thus have $f_i \leq \min\{\rho(s, u), \rho(v, t)\} = \pi(h)$. So at least $\left\lfloor \frac{\pi(h)}{\rho(h)} \right\rfloor$ paths of $P$ must contain $h$. □

From Proposition 14 we get a performance ratio that is satisfied by any integer solution.

**Corollary 15.** Let $H$ be an arbitrary directed $s$–$t$-cut and let $(P, f, C, g)$ be an integer solution of the instance $(G, s, t, r)$ of r.MSPFD. Then we have: $p \leq \frac{p}{\rho} \leq \max_{h \in H} \pi(h)$.

**Proof.** Proposition 14 yields $\sum_{h \in H} r(h) \leq N^* \max_{h \in H} \pi(h)$. Since $r$ is an integer $s$–$t$ flow, we get: $p \leq \sum_{h \in H} r(h) \leq N^* \max_{h \in H} \pi(h)$. □

![Fig. 6. A bad input for the lower bound $\pi$.](image-url)
Note that the lower bounds given in Propositions 14 and 13 may also be polynomially computed.

A final straightforward lower bound based on the number \(D(r)\) of distinct \(r\)-values is provided by the next Proposition.

**Proposition 16**

\[
\log_2 D(r) \leq N^*.
\]

**Proof.** Let \(p\) be the number of paths in a solution \((\Gamma, f)\) of \(I\). The number of distinct values of the sums \(\sum_{x \in X \subseteq T} f_x\) is at most \(2^p\). We thus have \(p \geq \log_2 D(r) = \beta. \square\)

### 4. Heuristics for MSPFD

Many algorithms have been developped to maximize flow in networks [1]. Maybe the generic augmenting path search is the most well-known. However this algorithm, which searches for paths in the residual graph, does not necessarily provide independent paths of the original graph. Different strategies have been proposed for this algorithm: one is to select the largest augmenting path and another one is to select the shortest augmenting path. We use these two strategies to propose heuristics providing independent paths.

#### 4.1. Generic structure

Our two heuristics are two members of a class of heuristics whose two basic principles are the following:

1. path independence is preserved by choosing at each iteration a saturating path of the current residual flow \(R\).
2. at each iteration, the selected path is an optimal path with respect to a \((\max, +)\)-like cost structure on the support graph \(G(R)\).

So, the generic structure of the heuristics that obey these principles is the following where \(c\) denotes the \((\max, +)\)-like cost structure:

```
procedure HEURMSPFD(G, s, t, r, c);
(1) \(R \leftarrow r;\)
(2) while \(G(R)\) has at least an \(s\)-\(t\)-path do
(3) find a \(c\)-optimal \(s\)-\(t\)-path \(\gamma\) in \(G(R)\);
(4) \(x \leftarrow \min_{c\gamma \in \{R_e\}}\{R_e\};\)
(5) \(R \leftarrow R - x\gamma;\)
(6) endwhile.
```

Note that at the end of the algorithm, either \(R = 0\) or \(G(R)\) is a set of simple circuits. Clearly the efficiency of such an heuristic will mainly depend on the cost structure \(c\) while its complexity will essentially depend on the way the previous \(c\)-optimal path \(\gamma\) is updated at line 3 after the residual flow \(R\) has been adjusted at line 5.

#### 4.1.1. Shortest path heuristic

The shortest path heuristic (SPH for short) chooses at each step the shortest path from \(s\) to \(t\) in the support graph \(G(r)\). In this case, the associated cost structure \(c\) is the \((\min, +)\) algebra where the length of an arc equals 1. A Breadth First Search from node \(s\) may be used to compute in \(O(n + m)\) the shortest paths values from node \(s\) in the support graph of the residual flow. Since at most \(m - n + 2\) will be chosen (see Proposition 9), the complexity of \(\text{SPH}(G, s, t, r)\) is \(O(m(n + m))\).

From Corollary 15, we know that SPH has performance ratio \(n - 1 - \frac{n^2 - 3n + 2}{m}\). The instance shown in Fig. 7 shows that this worst-case bound is tight. This instance is made of \(n - 1\) groups of \(k = \frac{m}{n-1}\) parallel arcs. Assume that the parallel arcs of each group are ranked from the top to the bottom.
It is clear that the $k$ paths each made of the arcs with the same rank make an optimal solution. So $N^* \geq \frac{m}{n-1}$. Moreover, the heuristic SPH may choose:

1. at first, $(k-1)(n-1)$ paths, each made of a single arc with rank at most $k-1$ (i.e. with unit $r$-value) and of the $n-2$ arcs with rank $k$ that do not belong to its group,
2. then the path made of the $n-1$ arcs with rank $k$.

For that solution choosen by SPH, we have $\frac{p}{N^*} = n - 1 - \frac{a^2-bn^2}{m}$.

4.1.2. Maximal residual path capacity heuristic

The maximal residual path capacity heuristic (MRPCH for short) chooses at each step a path from $s$ to $t$ with a maximal residual capacity. In this case, the associated cost structure $c$ is the (max, min) algebra where the length of arc $e$ of the support graph $G(R)$ equals its current capacity $R_e$. If $\pi_e(x)$ denotes the maximal residual capacity of a path from $s$ to $x$ in $G(R)$, it is well-known that the variant of the Dijkstra algorithm for the (max, min) cost structure computes the values $\pi_e(x)$ in $O(m + n \log n)$ time. Since at most $m-n+2$ will be choosen (see Proposition 9), the complexity of MRPCH $(G,s,t,r)$ is $O(m(m + n \log n))$.

4.2. Experimental results

The experimental tests were aimed to compare the performance of SPH and MRPCH. We used the parameters $n$ (the number of nodes of the graph) and $d = \frac{m}{n}$ (the density of the graph) to draw the class of instances. Three groups of tests have been driven: keeping $d$ small while increasing values of $n$; keeping $n$ small while increasing $d$; increasing both $n$ and $d$.

The test graphs are randomly generated directed acyclic graphs. The $r$-values are randomly distributed on the arcs while preserving the conservation law at each node.

The following experimental results are shown in arrays that contain the following fields: graph is the graph name, $n$ is the number of nodes, $m$ is the number of arcs, $lb$ is the lower bound $x$, $ub$ is the upper bound $m-n+2$, NSPH is the number of paths found by SPH, NMRPCH is the number of paths found by MRPCH, and $d$ is the density.

<table>
<thead>
<tr>
<th>Graph</th>
<th>$n$</th>
<th>$m$</th>
<th>$lb$</th>
<th>$ub$</th>
<th>NSPH</th>
<th>NMRPCH</th>
<th>$d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>G01</td>
<td>6</td>
<td>9</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1.5</td>
</tr>
<tr>
<td>G02</td>
<td>7</td>
<td>13</td>
<td>6</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>1.8</td>
</tr>
<tr>
<td>G03</td>
<td>8</td>
<td>15</td>
<td>6</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>1.9</td>
</tr>
<tr>
<td>G04</td>
<td>13</td>
<td>25</td>
<td>9</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>1.9</td>
</tr>
<tr>
<td>G05</td>
<td>15</td>
<td>35</td>
<td>17</td>
<td>22</td>
<td>22</td>
<td>22</td>
<td>2.3</td>
</tr>
<tr>
<td>G06</td>
<td>27</td>
<td>53</td>
<td>15</td>
<td>28</td>
<td>28</td>
<td>28</td>
<td>1.9</td>
</tr>
<tr>
<td>G07</td>
<td>40</td>
<td>79</td>
<td>27</td>
<td>41</td>
<td>41</td>
<td>41</td>
<td>1.9</td>
</tr>
<tr>
<td>G08</td>
<td>56</td>
<td>111</td>
<td>37</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>2</td>
</tr>
<tr>
<td>G09</td>
<td>90</td>
<td>179</td>
<td>59</td>
<td>91</td>
<td>90</td>
<td>88</td>
<td>1.9</td>
</tr>
</tbody>
</table>

The first array concerns graphs with increasing sizes and whose density value is not far from 2. These are typical values for telecommunications core networks. We observe that for small graphs (G01 to G08),

![Fig. 7. A worst-case instance for SPH.](image-url)
the number of paths proposed by the two heuristics is nearly the same. However the distinction lies in the comparison to the bounds. For the smallest graph \((G_01)\), the bounds are equal. But when the size of the graph grows up, the lower bound deviates from the upper bound and from the results of the heuristics. Moreover, except for the biggest graph \((G_09)\), the two heuristics provide the upper bound value. Finally, we observe that MRPCH provides a better solution than SPH.

<table>
<thead>
<tr>
<th>Graph</th>
<th>(n)</th>
<th>(m)</th>
<th>lb</th>
<th>ub</th>
<th>NSPH</th>
<th>NMRPCH</th>
<th>(d)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(G_{10})</td>
<td>9</td>
<td>89</td>
<td>46</td>
<td>82</td>
<td>82</td>
<td>80</td>
<td>9.9</td>
</tr>
<tr>
<td>(G_{11})</td>
<td>10</td>
<td>150</td>
<td>92</td>
<td>142</td>
<td>142</td>
<td>136</td>
<td>15</td>
</tr>
<tr>
<td>(G_{12})</td>
<td>10</td>
<td>200</td>
<td>106</td>
<td>192</td>
<td>192</td>
<td>188</td>
<td>20</td>
</tr>
<tr>
<td>(G_{13})</td>
<td>10</td>
<td>250</td>
<td>135</td>
<td>242</td>
<td>242</td>
<td>235</td>
<td>25</td>
</tr>
<tr>
<td>(G_{14})</td>
<td>10</td>
<td>300</td>
<td>150</td>
<td>292</td>
<td>292</td>
<td>282</td>
<td>30</td>
</tr>
<tr>
<td>(G_{15})</td>
<td>10</td>
<td>450</td>
<td>276</td>
<td>442</td>
<td>442</td>
<td>434</td>
<td>45</td>
</tr>
<tr>
<td>(G_{16})</td>
<td>10</td>
<td>600</td>
<td>273</td>
<td>592</td>
<td>592</td>
<td>583</td>
<td>60</td>
</tr>
<tr>
<td>(G_{17})</td>
<td>20</td>
<td>500</td>
<td>195</td>
<td>482</td>
<td>482</td>
<td>460</td>
<td>25</td>
</tr>
<tr>
<td>(G_{18})</td>
<td>24</td>
<td>1078</td>
<td>455</td>
<td>1056</td>
<td>1055</td>
<td>1005</td>
<td>45</td>
</tr>
<tr>
<td>(G_{19})</td>
<td>40</td>
<td>1600</td>
<td>690</td>
<td>1562</td>
<td>1562</td>
<td>1542</td>
<td>40</td>
</tr>
<tr>
<td>(G_{20})</td>
<td>48</td>
<td>2638</td>
<td>1080</td>
<td>2592</td>
<td>2589</td>
<td>2493</td>
<td>55</td>
</tr>
</tbody>
</table>

In the second array, the graphs \(G_{10}\) to \(G_{19}\) are still small-sized graphs. We observe that when the density increases, the heuristic SPH provides solutions with as many paths as the upper bound. Contrary to the graphs of the first array, the heuristic MRPCH provides solutions with less path than the upper bound.

<table>
<thead>
<tr>
<th>Graph</th>
<th>(n)</th>
<th>(m)</th>
<th>lb</th>
<th>ub</th>
<th>NSPH</th>
<th>NMRPCH</th>
<th>(d)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(G_{65})</td>
<td>67</td>
<td>133</td>
<td>42</td>
<td>68</td>
<td>68</td>
<td>66</td>
<td>2</td>
</tr>
<tr>
<td>(G_{66})</td>
<td>102</td>
<td>304</td>
<td>112</td>
<td>204</td>
<td>204</td>
<td>197</td>
<td>3</td>
</tr>
<tr>
<td>(G_{67})</td>
<td>44</td>
<td>216</td>
<td>81</td>
<td>174</td>
<td>174</td>
<td>160</td>
<td>4.9</td>
</tr>
<tr>
<td>(G_{68})</td>
<td>52</td>
<td>517</td>
<td>230</td>
<td>467</td>
<td>467</td>
<td>444</td>
<td>9.9</td>
</tr>
<tr>
<td>(G_{69})</td>
<td>83</td>
<td>823</td>
<td>312</td>
<td>742</td>
<td>741</td>
<td>692</td>
<td>9.9</td>
</tr>
<tr>
<td>(G_{70})</td>
<td>72</td>
<td>1069</td>
<td>418</td>
<td>999</td>
<td>997</td>
<td>950</td>
<td>14.8</td>
</tr>
<tr>
<td>(G_{71})</td>
<td>75</td>
<td>1112</td>
<td>430</td>
<td>1039</td>
<td>1036</td>
<td>1000</td>
<td>14.8</td>
</tr>
<tr>
<td>(G_{72})</td>
<td>41</td>
<td>811</td>
<td>355</td>
<td>772</td>
<td>771</td>
<td>754</td>
<td>19.8</td>
</tr>
<tr>
<td>(G_{73})</td>
<td>42</td>
<td>832</td>
<td>344</td>
<td>792</td>
<td>792</td>
<td>761</td>
<td>19.8</td>
</tr>
<tr>
<td>(G_{74})</td>
<td>77</td>
<td>1913</td>
<td>780</td>
<td>1838</td>
<td>1831</td>
<td>1787</td>
<td>24.8</td>
</tr>
</tbody>
</table>

The third array concerns large graphs with increasing density. We observe that the heuristic SPH has approximatively the same behaviour as in the previous cases while MRPCH behaves better for all samples.

5. Conclusion

This paper has raised an interesting and difficult combinatorial problem, called *Minimal Set of Paths for a Flow*, which is issued from routing problems in telecommunication networks. We have first shown that this problem is NP-hard in the strong sense. Then in order to derive heuristics, we have proved some properties of an optimal solution. We have been specially interested in a subset of solutions called saturating solutions, which seems to contain good solutions but does not necessarily contain the optimal one as shown in the paper. We then proposed two lower bounds and one upper bound of the minimal number of paths. We also developed two heuristics, each of which provides a saturating solution obtained by choosing iteratively an optimal path with respect to a \((\text{max},+)-\text{like}\) cost structure in the support graph of the current residual flow. When experimented and compared on three groups of graphs, it mainly appeared that the choice of path with a maximal residual capacity give better results than the choice of a shortest path.
The results presented in this paper can only be considered as preliminary results on MSPFD. In order to solve this problem more efficiently, there is still a lot of work to do in the following directions: deriving sufficient conditions for the dominance of the saturating solutions, improving the lower and upper bounds, finding more accurate criteria to allow a new path in the current solution, finding an efficient neighborhood structure in the solutions space.

Acknowledgements

We want to express our sincere thanks to the anonymous referees for their valuable and helpful comments that improved the paper.

Appendix A. Proof of Proposition 2

Proof. Note that, in this proof, r.MSPFD will point out the decision version of the optimization problem. We prove that the 3-PARTITION problem [3] can be pseudo-polynomially reduced to r.MSPFD. We first recall that in an arbitrary instance \((A, s)\) of 3-PARTITION, \(A\) is a set with \(3q\) elements and for any \(a \in A\), \(s(a)\) is the size of \(a\) where it is assumed that \(s(a)\) is a positive integer, \(\sum_{a \in A} s(a) = qB\) and \(\frac{B}{4} < s(a) < \frac{B}{2}\). The question is: is there a partition of \(A\) into \(q\) disjoints subsets with size \(B\)?

The instance \(f(A, s)\) of r.MSPFD associated with the instance \((A, s)\) of 3-PARTITION is shown in Fig. 8 where the arc values are the \(r\)-values. Note that there are \(3q\) arcs with input node \(s\) and output node \(x\) and \(q\) arcs with input node \(x\) and output node \(t\). The question is: is there a decomposition \(P\) with at most \(3q\) paths?

(a) Assume that there exists \(q\) subsets with size \(B\) making a partition of \(A\). By associating with each of these subsets the three paths as illustrated in Fig. 9, we get a solution to \(f(A, s)\) with exactly \(3q\) paths.

(b) Conversely, assume that there exists a solution \(P\) to the instance \(f(A, s) = (G, s, t, r)\) of r.MSPFD with at most \(3q\) paths. Thus \(r = \sum_{i \in P} x_i\). We first notice that \(P\) has exactly \(3q\) paths since otherwise at least one arc with input node \(s\) and output node \(x\) would carry a null flow value in \(P\). Let \(E\) be the set of \(q\) arcs with input node \(x\) and output node \(t\) and for any \(e \in E\), denote by \(P(e)\) the subset of the paths in \(P\) that contains \(e\). If \(P\) has more than three paths, then there exists \(e'\) such that \(P(e')\) has less than three paths. We thus have that \(B(= r_{e'})\) is equal to the sum \(x_i + x_k\) where each of these two numbers is at most equal to the size of an element in \(A\), what is a contradiction. So for any \(e \in E\), \(P(e)\) has exactly three paths. It remains to show that for \(e, e' \in E\) and \(e \neq e'\), no two paths of \(P(e)\) and \(P(e')\) share the same first arc. In such a case, there again would exist at least one arc with input node \(s\) and output node \(x\) with a null flow in the solution \(P\), what is a contradiction.

So, by associating with each \(e \in E\), the subsets of \(A\) that correspond to the first arcs of the three paths of \(P(e)\), we get a solution to the instance \((A, s)\) of 3-PARTITION.

Let \(r_{\text{max}}\) denote the maximal \(r\)-value in an arbitrary instance of r.MSPFD. Since in any solution \(P\), the paths of \(P\) are simple paths from \(s\) to \(t\), for any \(i \in \{1, \ldots, p\}\) we have \(x_i \leq r_{\text{max}}\) and \(m\) is an upper bound of the minimal number of paths (Caratheodory theorem), we may conclude that r.MSPFD is in NP.

![Fig. 8. The instance \(f(A, s)\) of r.MSPFD.](image-url)
It is also easy to observe that \( nm[\log_2 r_{\text{max}}] \) is a length function for r.MSPFD. Let \( s_{\text{max}} = \max\{s(a_i) \mid a_i \in A\} \) denote the maximum size value of an arbitrary instance \((A, s)\) of 3-PARTITION. Since for both instances \((A, s)\) and \(f(A, s)\), the maximal number is \( s_{\text{max}} \) and the length is \( q[\log_2 s_{\text{max}}] \), we may conclude that 3-PARTITION pseudo-polynomially reduces to r.MSPFD, what shows that r.MSPFD is NP-complete in the strong sense. \( \square \)

Appendix B. Proof of Proposition 8

**Proof.** We use the input represented in Fig. 3 to prove the proposition. The graph of this special input has a regular structure with two types of arcs (See Fig. 10):

- 5 “upper” arcs \( e_i, i = 1, \ldots, 5 \) (with distinct \( r \)-values greater than 32),
- 30 “lower” arcs partitionned into five groups of six arcs (with distinct \( r \)-values) where each group is associated with an upper arc \( e_i \) and is composed of three “top” arcs denoted \( e_{i,1}', e_{i,2}' \) and \( e_{i,3}' \) and of three “bottom” arcs denoted by \( e_{i,1}^b, e_{i,2}^b \) and \( e_{i,3}^b \).

First note that Fig. 4 shows a non saturating solution with six paths. We now show that this solution is the unique optimal solution.

Let us first consider an arbitrary solution with at most six paths, let \( H_i \) denote the number of paths containing the arc \( e_i \) and let \( B_i \) denote the number of paths that do not contain the arc \( e_i \). Since the six lower arcs associated with arc \( e_i \) have distinct \( r \)-values, we have \( B_i \geq 3 \) for \( i = 1, \ldots, 5 \).

Let us assume that \( B_1 = 3 \) and let \( f_1 \leq f_2 \leq f_3 \) be the \( f \)-values of these three paths. Then the sorted \( r \)-values of the lower arcs of group 1 are \((f_1, f_2, f_3, f_1 + f_2, f_1 + f_3, f_2 + f_3)\). Since \( f_1 = 3 \) and \( f_2 = 5 \) and no other lower arc has \( r \)-value \( f_1 + f_2 = 8 \), we get a contradiction. Thus, in any solution with at most six paths, we have \( B_i \geq 4 \) and \( H_i \leq 2 \).

Following the same lines of reasoning, we may conclude that for any \( i = 1, \ldots, 5 \), we have \( B_i \geq 4 \) and \( H_i \leq 2 \).

Let us now find upper and lower bounds for the \( f \)-values of a solution with at most six paths. When sorted in non-increasing order, the \( f \) values of that solution will be denoted by \((f_0, \ldots, f_5)\) where \( f_0 \geq \cdots \geq f_5 \) with the convention that if the solution has less than six paths then the last \( f \)-values are null. Let \( P_0 \) be the path whose \( f \)-value is \( f_0 \), what by convention will be denoted by \( f_0 = f(P_0) \).
• \( f_0 \) satisfies \( f_0 \leq 33 \) since the \( r \)-value of \( e_5 \) and of its associated lower arcs is at most 33. So we have \( H_i = 2 \) for \( i = 1, \ldots, 4 \).

• Since \( H_1 = 2 \), then we have \( f_0 \geq 24 \), what yields \( P_0 = (e_1, \ldots, e_5) \). So, for \( i = 1, \ldots, 4 \), the arc \( e_i \) is contained by path \( P_0 \) and one other path, which we denote by \( P_i \). Moreover, since the \( r \)-values of the arcs \( e_i \), \( i = 1, \ldots, 4 \) are distinct, the only upper arc contained by path \( P_i \) is \( e_i \).

• Assume \( f_1 > 17 \). Then the corresponding path, which is distinct from \( P_0 \), contains \( e_1, e_2, e_3, e_4 \), and thus is distinct from \( P_0 \) and \( P_1 \), a contradiction. So \( f_1 \leq 17 \).

• Since \( 24 \leq f_0 \leq 33 \), we have \( 15 \leq f(P_1) \leq 24 \). Furthermore, since \( f(P_1) \leq f_1 \) and \( f_1 \leq 17 \), we have \( 15 \leq f(P_1) \leq 17 \), what yields \( 31 \leq f_0 \leq 33 \).

From the inequalities \( 31 \leq f_0 \leq 33 \), \( f(P_1) \leq f_1 \) and \( f_1 \leq 17 \), we get: \( 31 \leq f_0 = f(P_0) \leq 33 \), \( 15 \leq f(P_1) \leq 17 \), \( 7 \leq f(P_2) \leq 9 \), \( 3 \leq f(P_3) \leq 5 \), \( 1 \leq f(P_4) \leq 3 \).

Let us now assume that \( f_0 = f(P_0) = 33 \).

We then have: \( f(P_1) = 15 \), \( f(P_2) = 7 \), \( f(P_3) = 3 \) and \( f(P_4) = 1 \). Since the total flow carried from \( s \) to \( t \) equals 63, the sum \( f(P_0) + \sum_{i=1}^{4} f(P_i) = 59 \) and the solution must contain an other path (say \( Q \)) such that \( f(Q) = 4 \). Consider now the lower arcs \( e_5^{1} \) and \( e_5^{2} \). Since \( 15 \leq f(P_1) \leq 17 \) and \( 7 \leq f(P_2) \leq 9 \), the paths \( P_1 \) and \( P_2 \) both contain \( e_5^{1} \). Since \( f(P_3) = 3 \), path \( P_3 \) contains \( e_5^{2} \). Thus path \( Q \) must contain \( e_5 \), a contradiction.

Since \( f_0 \leq 33 \), the solution has an other path \( P_5 \) that contains \( e_5 \) and no other upper arc. Thus the paths of the solution are \( P_0, \ldots, P_5 \).

Let us now consider the lower arcs \( e_1^{1} \) and \( e_1^{2} \). Since \( 7 \leq f(P_2) \), path \( P_2 \) contains \( e_1^{2} \). Assume that \( P_2 \) is the only path containing \( e_1^{1} \). Then we get \( f(P_2) = 9 \), \( f_0 = 31 \), \( f_0(P_3) = 5 \), \( f_0(P_4) = 3 \) and \( f_0(P_5) = 2 \), what is a contradiction since the \( r \)-value of arc \( e_1^{1} \) equals 6. Similarly, arc \( e_1^{1} \) cannot be contained in paths \( P_2, P_4 \) and \( P_5 \) since \( f(P_3) \leq 4 \) and \( P_3 \) would be the only path containing \( e_1^{1} \). So arc \( e_1^{1} \) belongs to path \( P_2 \) and to one of the two paths \( P_4 \) and \( P_5 \). Assume that this second path is \( P_4 \). Then we get: \( f(P_2) + f(P_3) = 9 \) and \( f(P_3) + f(P_4) = 6 \). When expressed in terms of \( f_0 \), these equalities become: \( 40 - f_0 + 34 - f_0 = 9 \) and \( 36 - f_0 + 33 - f_0 = 6 \), what is a contradiction. So arc \( e_1^{1} \) must belong to paths \( P_2 \) and \( P_5 \). In that case we get: \( f(P_2) + f(P_3) = 9 \) and \( f(P_3) + f(P_4) = 6 \). When expressed in terms of \( f_0 \), these equalities become: \( 40 - f_0 + 33 - f_0 = 9 \) and \( 36 - f_0 + 34 - f_0 = 6 \), what implies first \( f_0 = 32 \) and as a consequence: \( f(P_1) = 16, f(P_2) = 8, f(P_3) = 4, f(P_4) = 2 \) and \( f(P_5) = 1 \).

We thus know that if there is a solution with at most six paths, that solution is made of the path \( P_0 \) whose \( f \)-value equals 32 and of the five paths \( P_1, \ldots, P_5 \) whose \( f \)-values are respectively 16, 8, 4, 2, and 1. We also know that \( P_0 = (e_1, \ldots, e_5) \). It is then easy to see that path \( P_1 \) is uniquely defined since exactly one arc of each pair of lower arcs has a residual \( r \)-value at least equal to 16. Following this line of reasoning, the same is true for paths \( P_0, \ldots, P_5 \). Thus, the non saturating solution \((P_0, \ldots, P_5)\), with \((f_0, \ldots, f_5) = (32, 16, 8, 4, 2, 1)\) is the unique solution with at most six paths.

References