European Journal of Operational Research 247 (2015) 414–420

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Discrete Optimization

# Integral flow decomposition with minimum longest path length

Krzysztof Pieńkosz [a], Kamil Kołtyś [a,b,*]

[a] Institute of Control and Computation Engineering, Warsaw University of Technology, Nowowiejska 15/19 Warsaw 00-665, Poland
[b] Research and Academic Computer Network, Wąwozowa 18, Warsaw 02-796, Poland

ARTICLE INFO

*Article history:*
Received 7 October 2014
Accepted 7 June 2015
Available online 12 June 2015

*Keywords:*
Networks
Flow decomposition
Integral flow
NP-hard problem
Approximation algorithms

ABSTRACT

This paper concerns the problem of decomposing a network flow into an integral path flow such that the length of the longest path is minimized. It is shown that this problem is NP-hard in the strong sense. Two approximation algorithms are proposed for the problem: the longest path elimination (LPE) algorithm and the balanced flow propagation (BFP) algorithm. We analyze the properties of both algorithms and present the results of experimental studies examining their performance and efficiency.



## 1. Introduction

Many network design problems rely on finding the best realization of a traffic demand between a pair (or pairs) of telecommunication nodes (Pióro & Medhi, 2004). These problems are usually formulated using graph theory with demand realizations represented as network flows. Depending on the modeling approach a flow may define link's bandwidth utilization (an arc flow approach) or data traffic routes (a path flow approach). Both these approaches are equivalent in such a sense that any path flow can be uniquely transformed to an arc flow with the same total flow value and, conversely, for any arc flow one can construct a path flow with the same total flow value, although there may exist many such path flows (Ahuja, Magnanti, & Orlin, 1993).

In most network design problems the arc flow approach is used because it involves much less variables and many algorithms are based on it (Du & Kabadi, 2006). However, an arc flow does not say anything about routing which is essential from a viewpoint of practical demand realization. A path flow provides this information specifying paths (routes) through which traffic is routed. For an arc flow there can be many corresponding path flows that may differ in many important aspects. For example the number of routes or the number of hops in routes of such path flows may be different which in turn may affect network management costs or data transmission latency. Thus, having an arc flow given for example as a solution of some network design problem there is a problem of decomposing it into such a path flow that will have desired properties.

So far, researchers have focused mainly on finding flow decompositions with the minimum number of paths. Vatinlen, Chauvet, Chrétienne, and Mahey (2008) show that this problem is NP-hard in the strong sense. They define a saturating solution as such a flow decomposition that can be obtained by iteratively adding one path with assigned the maximum flow it can handle. Two approximation algorithms giving saturation solutions were proposed. The first one called Shortest Path Heuristic (SPH) add in each iteration the shortest path. The second algorithm named Maximal Residual Path Capacity Heuristic (MRPCH) add in each iteration a path that can handle the largest flow. From the experimental studies (Hartman, Hassidim, Kaplan, Raz, & Segalov, 2012; Vatinlen et al., 2008) it follows that MRPCH usually gives a decomposition with the smaller number of paths.

Here we analyze the problem of decomposing an arc flow into a path flow with minimum length of the longest path. Such a minmax criteria is often considered in many network problems (Li, McCormick, & Simchi-Levi, 1990). By a path length we define the number of arcs constituting the path. In telecommunication terminology this corresponds to the number of route hops. We limit our considerations to network flows represented by directed acyclic graphs. We also assume that flows on all paths have to be integral. This is a natural requirement if the demands have to be realized in modular units such as in the case of optical networks (Pióro & Medhi, 2004). In the paper we show that this problem is NP-hard in the strong sense. However, it should be noted that without the integrality constraint the problem becomes easier to solve. It is because if fractional flows are allowed then general linear programming methods can be used to determine the maximum flow satisfying a constraint on the maximum path length (Baier, 2003).

* Corresponding author. Tel.: +48 222347125.
*E-mail addresses:* K.Pienkosz@ia.pw.edu.pl (K. Pieńkosz), K.J.Koltys@elka.pw.edu.pl (K. Kołtyś).

http://dx.doi.org/10.1016/j.ejor.2015.06.012
0377-2217/

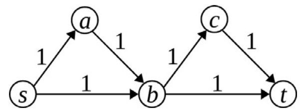


**Fig. 1.** Network flow example.

Algorithms SPH and MRPCH can be used to obtain approximate solutions of the considered problem. Hartman et al. (2012) even compare these algorithms with regard to the length of the longest path in flow decompositions showing in their experiments that better solutions are computed by SPH. Here we propose two other approximation algorithms that efficiently determine flow decompositions with the longest paths definitely shorter than the ones given by SPH.

The remainder of this paper is organized as follows. In the next section we formally define a network flow decomposition problem. Then, in Section 3 we analyze its properties. Sections 4 and 5 describe two classes of approximation algorithms which we propose to tackle the problem. Section 6 is devoted to a special type of flows called chain network flows. Some computational results are given in Section 7. Finally, the last section concludes the paper.

## 2. Problem statement

Let us consider a network flow represented as a directed graph $(V, E)$ formed by a set of vertices $V$ and a set of arcs $E \subset V \times V$ in which source $s \in V$ and sink $t \in V$ vertices are distinguished. We denote the number of vertices by $n$ and the number of arcs by $m$. Following the arc flow approach, each arc $e \in E$ has assigned a flow value $c_e$ in such a way that $\mathbf{c} = (c_e)_{e \in E}$ defines a valid $s$–$t$ flow. We assume that the graph $(V, E)$ is acyclic and the flow values $(c_e)_{e \in E}$ are positive integers. We denote the total value of $s$–$t$ flow by $F$. Fig. 1 presents an example of a network flow. The numbers next to arcs are the flow values. The total flow value for this network flow is 2.

Let $P$ be the set of all paths between $s$ and $t$. For an arc $e \in E$ and a path $p \in P$ we define the coefficient $\delta_{ep}$ equal to 1 if path $p$ uses arc $e$ and 0 otherwise. A flow decomposition $D$ of the network flow is a set of paths $P_D \subseteq P$ with a flow value $f_p$ assigned to each path $p \in P_D$ in such a way that

$$\sum_{p \in P_D} f_p = F \quad \text{and} \quad \sum_{p \in P_D} \delta_{ep} f_p = c_e \quad \forall e \in E. \tag{1}$$

We will consider only integral flow decompositions in which all paths have assigned flow values being positive integers.

For each path $p \in P$ we define its length $l_p$ as the number of arcs constituting this path, i.e. $l_p = \sum_{e \in E} \delta_{ep}$. We also introduce the notion of a flow decomposition length as the length of the longest path in this decomposition. For a flow decomposition $D$ we denote its length by $L_D = \max_{p \in P_D} l_p$.

For the network flow in Fig. 1 there exist two different flow decompositions. First one consists of two paths $s - b - t$ and $s - a - b - c - t$ with unitary flow values. The lengths of those paths are 2 and 4, respectively. Thus the length of the first flow decomposition is 4. In the second flow decomposition there are also two paths with unitary flow values: $s - a - b - t$ and $s - b - c - t$. However, the length of both these paths and thus the flow decomposition length is 3. So the second decomposition is shorter than the first one.

The problem that we are interested in this paper is to determine for a given network flow an integral flow decomposition with the minimum length. We call this problem Shortest Integral Flow Decomposition Problem (SIFDP).

## 3. Computational complexity of SIFDP

Before we analyze the computational complexity of SIFDP we give lower and upper bounds on the flow decomposition length.

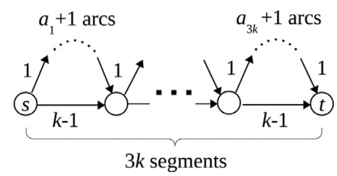


**Fig. 2.** Network flow $N^{3P}$ corresponding to the 3-PARTITION problem.

**Theorem 1.** *For any flow decomposition D*

$$\left\lceil \frac{\sum_{e \in E} c_e}{F} \right\rceil \le L_D \le \sum_{e \in E} c_e. \tag{2}$$

**Proof.** The first inequality holds because $L_D$ is an integer and

$$\sum_{e \in E} c_e = \sum_{p \in P_D} l_p f_p \le \sum_{p \in P_D} L_D f_p = L_D \sum_{p \in P_D} f_p = L_D F.$$

The second inequality follows from the fact that the length of any $s$–$t$ path in an acyclic graph is not greater than $m$. Thus $L_D \le m \le \sum_{e \in E} c_e$. □

We now show that SIFDP is NP-hard in the strong sense.

**Theorem 2.** *Problem SIFDP is NP-hard in the strong sense.*

**Proof.** We will show that the 3-PARTITION problem which is *NP*-complete in the strong sense (Garey & Johnson, 1979) can be pseudo-polynomially reduced to the decision version of SIFDP.

In an arbitrary instance of the 3-PARTITION problem we are given a positive integer $B$ and a set of $3k$ positive integers $A = \{a_1, a_2, \ldots, a_{3k}\}$ such that $\sum_{i=1}^{3k} a_i = kB$ and $B/4 < a_i < B/2$ for each $i = 1, 2, \ldots 3k$. The question is whether there exists a partition of $A$ into $k$ disjoint subsets $A_1, A_2, \ldots, A_k$ such that the sum of the elements in each subset is $B$.

The network flow $N^{3P}$ corresponding to the instance of the 3-PARTITION problem is shown in Fig. 2. The graph representing $N^{3P}$ consists of a sequence of $3k$ segments. The $i$th segment corresponds to the element $a_i$. Each segment has two parallel paths: the lower path with one arc and the upper path with $a_i + 1$ arcs. The flow values at the lower path and the upper path are $k - 1$ and 1, respectively. The total flow value for $N^{3P}$ is $k$. From the inequalities (2) it follows that the length of any flow decomposition for this network flow must be at least $B + 3k$.

Let us suppose that the 3-PARTITION problem has a solution. Then we can obtain the optimal flow decomposition of $N^{3P}$ that contains $k$ paths each one having length $B + 3k$ and unitary flow value. The $j$th path of this decomposition consists of the upper arcs of the segments that corresponds to the elements from the subset $A_j$ and the lower arcs of the remaining segments.

Conversely, suppose now that there exists a flow decomposition of $N^{3P}$ which length is $B + 3k$. Then all paths of this decomposition must have the same length. Moreover, since $B/4 < a_i < B/2$ for $i = 1, 2, \ldots 3k$ each path must contain the upper arcs from exactly 3 segments of the network flow $N^{3P}$. Because the graph representing $N^{3P}$ has $3k$ segments there must be $k$ such paths each one having unitary flow value. Moreover, the upper arcs of any segment have to be contained in exactly one decomposition path. The decomposition paths indicate then the feasible solution to 3-PARTITION with the subsets $A_j$ including the elements $a_i$ that correspond to the segments which upper arcs belong to the $j$th path. □

As SIFDP is NP-hard in the strong sense we are interested in developing approximate algorithms that will provide solutions as close to the optimal one as possible. We propose two classes of approximation algorithms: the path elimination algorithms and the flow propagation algorithms. By $A(N)$ we denote the length of the decomposition determined by an algorithm $A$ for a flow $N$. Likewise, we denote