Discrete Applied Mathematics 209 (2016) 16–26

Contents lists available at ScienceDirect

Discrete Applied Mathematics

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

The complexity of finding arc-disjoint branching flows

J. Bang-Jensen [a,*], Frédéric Havet [b], Anders Yeo [c,d]

[a] Department of Mathematics and Computer Science, University of Southern Denmark, Odense DK-5230, Denmark
[b] Project Coati, I3S (CNRS, UNSA) and INRIA, Sophia Antipolis, France
[c] Engineering Systems and Design, Singapore University of Technology and Design, 20 Dover Drive, 138682 Singapore, Singapore
[d] Department of Mathematics, University of Johannesburg, Auckland Park, 2006, South Africa

ARTICLE INFO

Article history:
Received 12 August 2014
Received in revised form 7 October 2015
Accepted 10 October 2015
Available online 10 November 2015

Keywords:
Disjoint branchings
Branching flow
Polynomial algorithm
NP-complete

ABSTRACT

The concept of arc-disjoint flows in networks was recently introduced in Bang-Jensen and Bessy (2014). This is a very general framework within which many well-known and important problems can be formulated. In particular, the existence of arc-disjoint branching flows, that is, flows which send one unit of flow from a given source s to all other vertices, generalizes the concept of arc-disjoint out-branchings (spanning out-trees) in a digraph. A pair of out-branchings $B^+_{s,1}, B^+_{s,2}$ from a root s in a digraph $D = (V, A)$ on n vertices corresponds to arc-disjoint branching flows x_1, x_2 (the arcs carrying flow in x_i are those used in $B^+_{s,i}$, $i = 1, 2$) in the network that we obtain from D by giving all arcs capacity $n - 1$. It is then a natural question to ask how much we can lower the capacities on the arcs and still have, say, two arc-disjoint branching flows from the given root s. We prove that for every fixed integer $k \geq 2$ it is

- an NP-complete problem to decide whether a network $\mathcal{N} = (V, A, u)$ where $u_{ij} = k$ for every arc ij has two arc-disjoint branching flows rooted at s.
- a polynomial problem to decide whether a network $\mathcal{N} = (V, A, u)$ on n vertices and $u_{ij} = n - k$ for every arc ij has two arc-disjoint branching flows rooted at s.

The algorithm for the later result generalizes the polynomial algorithm, due to Lovász, for deciding whether a given input digraph has two arc-disjoint out-branchings rooted at a given vertex. Finally we prove that under the so-called Exponential Time Hypothesis (ETH), for every $\epsilon > 0$ and for every $k(n)$ with $(\log(n))^{1+\epsilon} \leq k(n) \leq \frac{n}{2}$ (and for every large i we have $k(n) = i$ for some n) there is no polynomial algorithm for deciding whether a given digraph contains two arc-disjoint branching flows from the same root so that no arc carries flow larger than $n - k(n)$.

1. Introduction

Notation follows [3]. We denote the vertex set and arc set of a digraph D by $V(D)$ and $A(D)$, respectively and write $D = (V, A)$ where $V = V(D)$ and $A = A(D)$. Unless otherwise specified, the numbers n and m will always be used to denote the number of vertices, respectively arcs in the digraph in question. The digraphs may have parallel arcs but no loops. Paths and cycles are always directed unless otherwise specified. We will use the notation $[k]$ for the set of integers $\{1, 2, \ldots, k\}$.

* Corresponding author.
E-mail addresses: jbj@imada.sdu.dk (J. Bang-Jensen), frederic.havet@inria.fr (F. Havet), andersyeo@gmail.com (A. Yeo).

http://dx.doi.org/10.1016/j.dam.2015.10.012
0166-218X/© 2015 Elsevier B.V. All rights reserved.

An (s, t)-**path** in a digraph D is a directed path from the vertex s to the vertex t. The **underlying graph** of a digraph D, denoted $UG(D)$, is obtained from D by suppressing the orientation of each arc. A digraph D is **connected** if $UG(D)$ is a connected graph. When xy is an arc of D we say that x **dominates** y. For a digraph $D = (V, A)$ the **out-degree**, $d_D^+(x)$ (resp. the **in-degree**, $d_D^-(x)$) of a vertex $x \in V$ is the number of arcs of the kind xy (resp. yx) in A, where we count parallel arcs. When $X \subseteq V$ we shall also write $d_X^+(v)$ to denote the number of arcs vx with $x \in X$.

An **out-tree** rooted at s, also called an s**-out-tree**, is a tree containing the vertex s in $UG(D)$ such that every vertex v different from s has exactly one arc entering in the tree. Equivalently, s has a unique directed path to every other vertex of the tree using only arcs of the tree. An s**-out-branching** is a spanning s-out-tree. We use the notation T_s^+, B_s^+ to denote an s-out-tree, respectively an s-out-branching.

Branchings in digraphs are important from a practical point of view and appear in many applications and hence it is relevant to consider quality measures for branchings. This has been done in a number of papers, see e.g. [4,5,8,9,14]. An s-out-branching B_s^+ is k**-safe** if no matter which out-neighbour v of s in B_s^+ we consider, the s-out-tree T_s^+ that we will obtain after deleting all the vertices of the out-tree T_v^+ rooted at v in B_s^+ contains at least k vertices (s and at least $k-1$ other vertices). In applications (where an s-out-branching is used to route information or similar from the root s) it is desirable to use an out-branching which is k-safe for a high value of k, because this means that no matter which arc we cut, s can still reach k other vertices using only arcs from the remaining s-out-tree.

In terms of protection against arc-faults, branchings are not a very good way of sending information from one source to all other vertices: If we insist that the vertex sets of the routes that we use to send the information from s to all other vertices may only intersect in a prefix of each route (this is equivalent to saying that the union of the routes is an s-out-branching), then the set of routes may be very vulnerable to arc-deletions. As an example consider the digraph H consisting of vertices $s = v_1, v_2, \ldots, v_{2p+1}$ and arcs $\{sv_2, sv_3, v_2v_3\} \cup \{v_3v_i | i \geq 4\}$. This digraph contains no 3-safe s-out-branching. On the other hand, if instead we use each of the arcs sv_2, sv_3 on p of the paths from s to $V - s$, then deletion of one of the arcs sv_2, sv_3 disconnects s from only half of the other vertices. We may then ask what is the best way to route the information from s to all other vertices, while preserving a high degree of protection against arc-faults. This leads to the study of branching flows. Before we can formally define these, we need to recall a bit of flow theory.

A **network** $\mathcal{N} = (V, A, u)$ is a digraph $D = (V, A)$ equipped with a non-negative capacity function $u : A \to \mathbf{R}_0$ on its arcs. A **flow** in $\mathcal{N}$ is any non-negative function $x : A \to \mathbf{R}_0$ which satisfies that $x_{ij} \leq u_{ij}$ for every $ij \in A$, where x_{ij}, u_{ij} denote, respectively, the flow value on ij and the capacity of ij. The **balance-vector** of a flow x is the function b_x on V which to each vertex $i \in V$ associates the value $b_x(i) = \sum_{ij \in A} x_{ij} - \sum_{pi \in A} x_{pi}$.

If $\mathcal{N} = (V, A, u, b)$, that is, there is also a balance-vector specified for $\mathcal{N}$, then a flow x is **feasible** in $\mathcal{N}$ if it satisfies $b_x(v) = b(v)$ for all $v \in V$. Two flows x, y in a network $\mathcal{N}$ are **arc-disjoint** if $x_{ij} \cdot y_{ij} = 0$ for every arc ij of $\mathcal{N}$.

A **path flow** along the path P (resp. **cycle flow** along the cycle C) in a network $\mathcal{N}$ is a flow x which has $x_{ij} = k$ for every arc on P (resp. C) for some positive value k and $x_{ij} = 0$ for all arcs not on P (resp. C). An s**-branching flow** in a network $\mathcal{N}$ is a flow x in $\mathcal{N}$ with balance vector $b_x(v) = -1$ for $v \neq s$ and $b_x(s) = n - 1$, where n denotes the number of vertices in $\mathcal{N}$. Finally an $(\mathbf{s}, \mathbf{t})$**-flow** in a network $\mathcal{N}$ is a flow x with balance vector $b_x(s) = -b_x(t) = k$ and $b_x(v) = 0$ for $v \notin \{s, t\}$, where k is a non-negative real number which is also called the **value** of the (s, t)-flow. The so-called **max-flow problem** is the problem of finding the maximum value k so that a given network $\mathcal{N}$ with special vertices s, t has an (s, t)-flow. This problem is solvable in polynomial time by many different methods, see e.g. [1,3].

An important result in flow theory is the following which states that it is possible to decide in polynomial time whether or not there exists a feasible flow for a given network $\mathcal{N} = (V, A, u, b)$ (see e.g. [3, Section 4.8]).

Theorem 1.1. *For a given network $\mathcal{N} = (V, A, u, b)$ with arc-capacities given by u and vertex-balances prescribed by b, by solving one max-flow problem in an associated network, in polynomial time, one can either determine a feasible flow x in $\mathcal{N}$ or verify (by producing a certificate for the non-existence) that no such flow exists in $\mathcal{N}$.*

The connection between branching flows and max-flows is particularly simple: A network $\mathcal{N} = (V, A, u)$ on n vertices has an s-branching flow for some $s \in V$ if and only if there is an (s, t)-flow of value $n - 1$ in the network $\mathcal{N}'$ that we obtain from $\mathcal{N}$ by adding a new vertex t and an arc vt of capacity 1 for each $v \in V - s$.

The following folklore result (see e.g. [1, Section 3.5] or [3, Section 4.3.1]) is very useful when working with flows.

Theorem 1.2 (*Flow Decomposition Theorem*). *Every flow x in a network $\mathcal{N}$ on n vertices and m arcs is the arc-sum of at most $n + m$ path and cycle flows. Furthermore, the path flows can be taken along paths $P_1, \ldots, P_q$ such that P_i starts in a vertex s_i with $b_x(s_i) > 0$ and ends in a vertex t_i with $b_x(t_i) < 0$ for $i \in [q]$. In particular, if $b_x \equiv 0$ there are no path flows in the decomposition and x is the arc-sum of at most m cycle flows. Given the flow x, a decomposition as above can be found in time $O(nm)$.*

Note that when we consider branching flows below, we are only interested in the acyclic part of such a flow, that is, the collection of paths from the root to all other vertices that we obtain by flow decomposition (we leave out flow along cycles since that does not contribute to the balance of the flow). By the Flow Decomposition Theorem, every branching flow x from s contains one or more out-branchings from s as a subdigraph (x sends one unit of flow from s to all other vertices).

As in the case of branchings, we may also measure the quality of an s-branching flow in terms of how vulnerable it is towards arc-deletions. If we have no restrictions on the flow values, a branching flow x may have flow equal to $r \leq n - 1$