



**HAL**  
open science

## Enumerating $k$ -arc-connected orientations

Sarah Blind, Kolja Knauer, Petru Valicov

► **To cite this version:**

Sarah Blind, Kolja Knauer, Petru Valicov. Enumerating  $k$ -arc-connected orientations. *Algorithmica*, In press, 10.1007/s00453-020-00738-y . hal-02435175v1

**HAL Id: hal-02435175**

**<https://hal.science/hal-02435175v1>**

Submitted on 13 Jan 2020 (v1), last revised 25 Sep 2020 (v2)

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Enumerating $k$ -arc-connected orientations

Sarah Blind\*

Kolja Knauer<sup>†‡</sup>

Petru Valicov<sup>†§</sup>

November 28, 2019

## Abstract

We study the problem of enumerating the  $k$ -arc-connected orientations of a graph  $G$ , i.e., generating each exactly once. A first algorithm using submodular flow optimization is easy to state, but intricate to implement. In a second approach we present a simple algorithm with delay  $O(knm^2)$  and amortized time  $O(m^2)$ , which improves over the analysis of the submodular flow algorithm. As ingredients, we obtain enumeration algorithms for the  $\alpha$ -orientations of a graph  $G$  in delay  $O(m^2)$  and for the outdegree sequences attained by  $k$ -arc-connected orientations of  $G$  in delay  $O(knm^2)$ .

## 1 Introduction

In an enumeration problem one receives a typically small input and wants to output every element of a typically large resulting set exactly once<sup>1</sup>. Since the runtime of an enumeration algorithm depends on the output, usually one measures how much it exceeds the size of the output in terms of the size of the input. More precisely, one wants to control the *delay*, i.e., the maximum time between two consecutive outputs (including the time before the first and after the last output) in terms of the input or at least the average over these, called the *amortized time*. Clearly, the delay is an upper bound for the amortized time.

An instance that illustrates this challenge perfectly is given an undirected (not necessarily simple) graph  $G$ , enumerate all its orientations with a given property. Many types of orientations have been studied with respect to their enumeration complexity, see [7] for an overview. Here we give just some examples, before describing the set-up of this paper.

The set of all orientations of  $G = (V, E)$  can be identified with the set of vectors  $\{0, 1\}^m$  where  $m = |E|$ . Thus, enumerating all orientations of  $G$  using a Gray code can be done with constant delay, see [12]. It gets more interesting when enumerating *acyclic* orientations, i.e. those that have no directed cycles. In [38] an algorithm for enumerating all acyclic orientations with delay  $O(n^3)$  but linear amortized time  $O(n)$  was given, where  $n = |V|$ . In [3] the delay was reduced to  $O(mn)$  with an increase in amortized time to  $O(m+n)$ . Another improvement was obtained in [8], where an algorithm of delay and amortized time  $O(m)$  is given.

*Strongly connected* orientations are those such that for any two vertices  $u, v \in V$  there is a directed path from  $u$  to  $v$ . In [9] an enumeration algorithm of strongly connected orientations with delay  $O(m)$  is given. The main objective of the present paper is to enumerate a parameterized version of strong orientations, generalizing the above. Namely, we enumerate the  *$k$ -arc-connected orientations* of  $G$ , i.e., those where at least  $k$  arcs have to be removed in order to destroy the strong connectivity<sup>2</sup>. Note that an orientation is strongly connected if and only if its is 1-connected.

The concept of  $k$ -connectivity is a classic in the theory of directed graphs. Let us review some of the fundamental results of the area. A theorem of Nash-Williams [32] asserts that  $G$  admits a

---

\*Université de Lorraine, LGIPM, F-57000 Metz, France

Email: [sarah.blind@univ-lorraine.fr](mailto:sarah.blind@univ-lorraine.fr)

<sup>†</sup>Departament de Matemàtiques i Informàtica, Universitat de Barcelona (UB), Barcelona, Spain

<sup>‡</sup>Aix-Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France

Email: [{kolja.knauer,petru.valicov}@lis-lab.fr](mailto:{kolja.knauer,petru.valicov}@lis-lab.fr)

<sup>§</sup>LIRMM, CNRS, Université de Montpellier, Montpellier, France.

<sup>1</sup>We use the term *enumeration* instead of the sometimes used terms *generation* or *listing*.

<sup>2</sup>When there is no ambiguity, we will abbreviate  $k$ -arc-connected by saying  *$k$ -connected*.

$k$ -connected orientation if and only if  $G$  is  $2k$ -edge connected. A result of Lovász [30] yields an easy algorithm to find a  $k$ -connected orientation of  $G$  if there is one, that runs in  $O(n^6)$ . More involved techniques have been developed to improve this runtime, see [19, 29]. Important contributions to the theory of  $k$ -connected orientations come from Frank. In particular, he cast  $k$ -connected orientations as an instance of the theory of submodular flows, see [16]. Finding a submodular flow (and hence a  $k$ -connected orientation) can be done in polynomial time. There is a considerable amount of literature proposing different algorithmic solutions for this task on simple graphs, multigraphs, and mixed graphs, see e.g. [16, 18, 20, 26]. This is the central fact behind a first enumeration algorithm that we propose here. Another fundamental result of Frank is that any two  $k$ -connected orientations of  $G$  can be transformed into each other by reversing directed paths and directed cycles [17]. Disassembling this result lies at the core of our second algorithm.

All our algorithms are *backtrack searches*, i.e., we build a tree whose vertices are partial solutions and the leaves are full solutions. We explore the tree on the fly by a depth-first search in order to reach all the leaves. The delay of the algorithm depends on the depth of the tree and the time spent at each node.

The first algorithm can be described as follows. Using a min-cost  $k$ -connected orientation algorithm due to [26] and following ideas of [2] we design an algorithm that decides in  $O(n^3k^3 + kn^2m)$  if a given mixed graph  $G = (V, E \cup A)$  can be extended to an orientation that is  $k$ -connected. This yields an enumeration algorithm for  $k$ -connected orientations with delay  $O(m(n^3k^3 + kn^2m))$ . The idea is to simply start with  $G = (V, E)$  as the root vertex of the backtrack search tree and at each node create two children by picking an edge, fixing it to be oriented in one way or the other, and verifying if a  $k$ -connected extension exists. Since this tree has depth in  $O(m)$  the previously stated runtime follows. Note that any other algorithm finding a  $k$ -connected extension of a partial orientation could be used in this approach, see for example [16, 18, 20]. See Section 3 for a detailed description.

The main part of the paper presents an alternative enumeration algorithm. Its main advantages are its simplicity and an improvement of the runtime. Moreover, it consists of two parts enumerating objects of independent interest. The idea is based on Frank's result [17] that any two  $k$ -connected orientations can be transformed into each other by reversals of directed cycles and directed paths. Our algorithm splits this into two parts: one reversing cycles, the other paths.

By reversing directed cycles, we obtain an algorithm that enumerates all orientations of  $G$  with a prescribed outdegree sequence also known as  $\alpha$ -orientations. These orientations are of current interest with respect to computational properties (see e.g. [1]) and model many combinatorial objects such as domino and lozenge tilings of a plane region [37, 40], spanning trees of a planar graph [21], perfect matchings (and  $d$ -factors) of a bipartite graph [13, 28, 35], Schnyder woods of a planar triangulation [5], Eulerian orientations of a graph [13], and contact representations of planar graphs with homothetic triangles, rectangles, and  $d$ -gons [14, 15, 24]. The set of  $\alpha$ -orientations of a planar graph can be endowed with a natural distributive lattice structure [13] and therefore, in this case the enumeration can be done in linear amortized time [25]. It is a famous question (also open in this setting) whether the enumeration of the elements of a distributive lattice can be done in constant amortized time [36]. Our enumeration algorithm for general  $\alpha$ -orientations has a delay of  $O(m^2)$  and is explained in Section 4.

Reversing directed paths, yields an enumeration algorithm for all outdegree sequences that are attained by  $k$ -connected orientations of  $G$ . This algorithm has a delay of  $O(m^2kn)$  and is explained in Section 5.

In Section 6 we combine these algorithms, leading to an enumeration algorithm for  $k$ -connected orientations of delay  $O(m^2kn)$ . The split into two algorithms leads to a structured traversal of the solution space which allows to prove an amortized runtime in  $O(m^2)$ .

We close the paper in Section 7 with some open questions.

## 2 Preliminaries

In this section we introduce some digraph basics. All graphs we consider in this paper are multigraphs and consequently their orientations also may have multiple parallel or anti-parallel arcs. We will also consider *mixed multigraphs*. These are of the form  $G = (V, E \cup A)$ , where  $E$  is a multiset of undirected

edges and  $A$  is a multiset of directed arcs. Analogously to undirected graphs, an *orientation* of a mixed graph consists in fixing a direction for each of its undirected edges.

The digraph obtained from  $D = (V, A)$  by reversing a set of arcs  $B \subseteq A$  is denoted by  $D^B$ . If  $A = B$  we write  $D^-$  instead of  $D^B$ . Given an arc  $a = (u, v)$ , we denote by  $a^- = (v, u)$  the reversed arc. Similarly, the reversed arc set of  $B$  is denoted  $B^- = \{a^- \mid a \in B\}$ . Given a digraph  $D = (V, A)$  and a subset  $X \subseteq V$  we will denote its *outdegree*  $\delta_D^+(X) = |\{a = (u, v) \in A \mid u \in X \not\rightarrow v\}|$ . In the case where  $X = \{v\}$  consists of a single element we just write  $\delta_D^+(v)$ .

Note that by the definition of  $k$ -connectivity we have that  $D$  is  $k$ -connected if and only if  $\delta_D^+(X) \geq k$  for every  $\emptyset \neq X \subsetneq V$ . There is a directed version of Menger's Theorem that provides another characterization of  $k$ -connectivity, see [31]. For its statement, we denote by  $\lambda(u, v)$  the maximum number of arc-disjoint directed paths from  $u$  to  $v$ .

**Theorem 1** (Local Menger Theorem for digraphs). *Let  $D = (V, A)$  be a digraph and  $u, v \in V$ . We have  $\lambda(u, v) = \min\{\delta_D^+(X) \mid u \in X \not\rightarrow v\}$ .*

Thus, as a consequence of Theorem 1,  $D$  is  $k$ -connected if and only if  $\lambda(u, v) \geq k$  for all  $u, v \in V$ .

### 3 A first enumeration algorithm

We present a more detailed description of the first enumeration algorithm mentioned in the introduction. In [2] the following result is shown based on a min-cost  $k$ -connected orientation algorithm due to [26]:

**Theorem 2** ([2, 26]). *It can be decided in time  $O(k^3n^3 + kn^2m)$ , whether a mixed graph admits a  $k$ -connected orientation.*

By Theorem 2, one can easily design a backtrack search algorithm for enumerating all  $k$ -connected orientations with polynomial delay (see Algorithm 1). At each step it takes a new edge and checks for both its orientations if there exists a  $k$ -connected orientation of the graph respecting the so far fixed orientation.

---

**Algorithm 1:** enumeration of  $k$ -connected orientations via min-cost orientations

---

**Input:** A graph  $G = (V, E)$  and an integer  $k \in \mathbb{N}$

**Output:** The  $k$ -connected orientations of  $G$

1 **begin**

2     Fix any linear ordering on  $E$ ;

3     Enumerate( $G = (V, E, \emptyset)$ );

4 **end**

5 **Function** Enumerate( $G = (V, E, F)$ ):

6     **if**  $E \neq \emptyset$  **then**

7         Take the smallest  $e = \{u, v\} \in E$ ;

8         **if**  $G' := (V, E \setminus \{e\}, F \cup \{(u, v)\})$  admits a  $k$ -connected orientation **then**

9             Enumerate( $G'$ );

10         **if**  $G' := (V, E \setminus \{e\}, F \cup \{(v, u)\})$  admits a  $k$ -connected orientation **then**

11             Enumerate( $G'$ );

12         **else**

13             Output  $G$ ;

14 **end**

---

This algorithm takes as input an undirected graph  $G = (V, E)$ . Clearly, all  $k$ -connected orientations are produced and since each node built by this algorithm gives rise to disjoint branches it does not repeat the same solution twice. The depth of the binary execution tree is  $m$  and at each node we check the orientability of a mixed graph which is solvable in  $O(k^3n^3 + kn^2m)$  by Theorem 2. We conclude

**Proposition 3.** *Let  $G$  be a graph and  $k \in \mathbb{N}$ . Algorithm 1 enumerates all  $k$ -connected orientations of  $G$  with delay and amortized time in  $O(m(k^3n^3 + kn^2m))$ .*

A downside of Algorithm 1 is that Theorem 2 is based on a submodular flow algorithm and its implementation is intricate. Moreover, the literature on submodular flows and the available runtime analyses in different settings such as simple and multigraphs are hard to extract. A more detailed discussions of these approaches is discussed in [4].

In the following we will give two simple algorithms of independent interest that combined yield an alternative solution for the enumeration of  $k$ -connected orientations. The resulting algorithm improves over the delay and amortized runtime of Algorithm 1.

## 4 Orientations with prescribed outdegree sequence

Let  $G = (V, E)$  be a graph and  $\alpha : V \rightarrow \mathbb{N}$ . We say that an orientation  $D$  of  $G$  is an  $\alpha$ -orientation if  $\delta_D^+(v) = \alpha(v)$  for all  $v \in V$ . We will denote by  $\mathcal{O}_\alpha(G)$  the set of all  $\alpha$ -orientations of  $G$ . In the present section we discuss the enumeration of the set  $\mathcal{O}_\alpha(G)$ . Before presenting the algorithm let us give a folklore result about  $\alpha$ -orientations:

**Lemma 4.** *Let  $G$  be a graph and  $D$  and  $D'$  be two orientations of  $G$ . We have  $\delta_D^+ = \delta_{D'}^+$  if and only if orientation  $D'$  can be obtained from  $D$  by reversing a set of arc-disjoint directed cycles.*

*Proof.* Reversing the direction of a directed cycle does not change the outdegree function. Therefore, if  $D'$  is obtained from  $D$  by reversing a set of arc-disjoint directed cycles, then  $\delta_D^+ = \delta_{D'}^+$ .

Let conversely  $D = (V, A)$  and  $D' = (V, A')$  be orientations of  $G$  such that  $\delta_D^+ = \delta_{D'}^+$ . Consider the arcs  $A \setminus A'$  of  $D$  whose direction differ in  $D'$ . Observe that in the directed subgraph  $D \setminus A'$  formed by these arcs the indegree and the outdegree coincide at each node. Thus,  $D \setminus A'$  is Eulerian and can be decomposed into an arc-disjoint union of directed cycles. This concludes the proof.  $\square$

Our enumeration algorithm for  $\mathcal{O}_\alpha(G)$  takes a digraph  $D = (V, A) \in \mathcal{O}_\alpha(G)$  and a set of *fixed* arcs  $F \subseteq A$  as input. Initially  $D$  is an arbitrary  $\alpha$ -orientation and  $F = \emptyset$ . The algorithm recursively constructs all orientations of  $\mathcal{O}_\alpha(G)$  such that at each call all possible  $\alpha$ -orientations contain the current  $F$ . At each recursive call, an arc  $a \notin F$  will be fixed with the two possible cases to consider: The branch with  $a = (u, v)$  is known to be non-empty since  $D$  is a possible extension, thus we recurse with  $D$  and  $F \cup a$ . For the branch with  $a^- = (v, u)$  we check if there is a directed path  $P$  in  $D \setminus F$  from  $v$  to  $u$ . If this is the case, then the directed cycle  $(P, a)$  is reversed and we recurse with the new  $\alpha$ -orientation  $D^{P \cup \{a\}}$  and  $F \cup \{a^-\}$ .

The set of fixed arcs is extended along an arbitrary linear ordering of the arcs. Whenever all arcs are fixed we output the current orientation. We give a presentation of our algorithm in pseudo-code (see Algorithm 2).

---

**Algorithm 2:** Backtrack search for  $\alpha$ -orientations

---

**Input:** A graph  $G = (V, E)$  and  $\alpha : V \rightarrow \mathbb{N}$

**Output:** All elements of  $\mathcal{O}_\alpha(G)$

```

1 begin
2   if there exists  $D = (V, A) \in \mathcal{O}_\alpha(G)$  then
3     Fix an arbitrary linear order on  $A$ ;
4     EnPODS( $D, \emptyset$ );
5 end
```

---

---

```

6 Function EnOPODS( $D, F$ ):
7   if  $F \neq A$  then
8     Take the smallest  $a = (u, v) \in A \setminus F$ ;
9     EnOPODS( $D, F \cup \{a\}$ );
10    if  $D \setminus F$  has a directed path  $P$  from  $v$  to  $u$  then
11      EnOPODS( $D^{P \cup \{a\}}, F \cup \{a^-\}$ );
12    else
13      Output  $D$ ;
14 end

```

---

Algorithm 2 takes an undirected graph  $G = (V, E)$  as input, but after finding one initial  $\alpha$ -orientation  $D$ , it just consists of recursive calls of the function  $\text{EnOPODS}(D, F)$ .

**Lemma 5.** *Let  $D \in \mathcal{O}_\alpha(G)$  and  $F \subseteq A$ . The function  $\text{EnOPODS}(D, F)$  generates each orientation in  $\mathcal{O}_\alpha(G)$  that coincides with  $D$  on  $F$  exactly once and has delay in  $O(m^2)$ .*

*Proof.* We prove that each of the claimed  $\alpha$ -orientations is generated exactly once by induction on  $|A \setminus F|$ . If  $|A \setminus F| = 0$ , then  $\text{EnOPODS}(D, F) = \text{EnOPODS}(D, A) = \{D\}$  and we are done. Let now  $|A \setminus F| > 0$  and  $a = (u, v) \in A \setminus F$ . By induction hypothesis  $\text{EnOPODS}(D, F \cup \{a\})$  generates each  $\alpha$ -orientation that coincides with  $D$  on  $F \cup \{a\}$  exactly once and  $\text{EnOPODS}(D^{P \cup \{a\}}, F \cup \{a^-\})$  generates each  $\alpha$ -orientation that coincides with  $D^{P \cup \{a\}}$  on  $F \cup \{a^-\}$  exactly once. Clearly, both sets are disjoint since they differ with respect to the orientation of  $a$ . Thus, no repetitions are produced. Since  $(P, a)$  is a directed cycle by Lemma 4 we have  $D^{P \cup \{a\}} \in \mathcal{O}_\alpha(G)$ . Since  $P \cap F = \emptyset$  we have that  $D^{P \cup \{a\}}$  coincides with  $D$  on  $F$ .

To see that we do not miss any orientation, we prove that if there is no directed path  $P$  from  $u$  to  $v$  in  $D \setminus F$ , then there exists no  $\alpha$ -orientation fixing  $F$  and reversing  $a$ . By contraposition, suppose that  $D' = (V, A')$  is an  $\alpha$ -orientation that coincides with  $D$  on  $F$  but differs on  $a$ . Then by Lemma 4, there is a set of arc-disjoint directed cycles in  $D'$  whose union is  $A' \setminus A$ . Since both digraphs coincide on  $F$ , these cycles are disjoint from  $F$ . Since both digraphs differ on  $a$ , one of the directed cycles  $C$  contains  $a^-$  in  $D'$ . Thus, the path  $P = (C \setminus \{a^-\})^-$  is a directed path in  $D \setminus F$  from  $v$  to  $u$ .

For the complexity, note that in each recursion step the algorithm checks the presence of a directed path, which can be done by a single BFS from the source vertex  $u$  towards the target  $v$ . In general the complexity of a BFS algorithm is  $O(m + n)$ , however in our case the BFS tree will be constructed only on the strongly connected component of  $D$  containing  $u$  and thus the complexity is in  $O(m)$ . The depth of our recursion tree is bounded by  $m$ . Thus, the total delay is bounded by  $O(m^2)$ .  $\square$

Observe that Algorithm 2 has to use a separate method for finding a first element  $D \in \mathcal{O}_\alpha(G)$ . It is well-known that this problem can be reduced to a flow-problem, see e.g. [13]. Therefore, this preprocessing step can be done in  $O(mn)$  time, see [34]. With Lemma 5 we obtain:

**Theorem 6.** *Let  $G$  be a graph and  $\alpha : V \rightarrow \mathbb{N}$ . Algorithm 2 enumerates  $\mathcal{O}_\alpha(G)$  with delay in  $O(m^2)$ .*

## 5 Outdegree sequences

In this section we will present an algorithm to enumerate the possible outdegree sequences among the  $k$ -connected orientations of a graph  $G$ .

An easy consequence of Lemma 4 is the following that can also be found in [17].

**Lemma 7.** *If  $D, D' \in \mathcal{O}_\alpha(G)$ , then  $D$  is  $k$ -connected if and only if  $D'$  is  $k$ -connected.*

*Proof.* Let  $D, D' \in \mathcal{O}_\alpha(G)$  and  $D$  be  $k$ -connected. Thus, for all  $\emptyset \neq X \subsetneq V$  we have  $\delta_D^+(X) \geq k$ . By Lemma 4 since  $D' \in \mathcal{O}_\alpha(G)$  it can be obtained from  $D$  by reversing a set of disjoint directed cycles. But reversing a directed cycle in a digraph does not change the outdegree of the subsets of vertices of  $G$ . Therefore, after reversing the set of directed cycles to obtain  $D'$ , we have  $\delta_{D'}^+(X) \geq k$  for all  $\emptyset \neq X \subsetneq V$  and  $D'$  is  $k$ -connected.  $\square$

Given  $G = (V, E)$ , Lemma 7 allows to define a function  $\alpha : V \rightarrow \mathbb{N}$  to be  $k$ -connected if there is some  $k$ -connected  $D \in \mathcal{O}_\alpha(G)$ . In this case we call  $\alpha$  a  $k$ -connected outdegree sequence. Having in mind that we want to enumerate all  $k$ -connected orientations of  $G$  and already are able to enumerate  $\mathcal{O}_\alpha(G)$  for any given  $\alpha$ , we are left with enumerating the  $k$ -connected outdegree sequences. Here in order to change the outdegree sequence of  $D$  we will reverse a directed path  $P_{uv}$  from  $u$  to  $v$  and thus increase  $\delta_D^+(u)$  by one, decrease  $\delta_D^+(v)$  by one, and leave the remaining outdegrees unchanged. More generally we have:

**Observation 1.** *Let  $D$  be an orientation of graph  $G$  and  $X \subseteq V(G)$ . If  $D'$  is obtained from  $D$  by reversing a path from a vertex  $u$  to a vertex  $v$ , then we have*

1.  $\delta_{D'}^+(X) = \delta_D^+(X)$  if  $u, v \in X$  or  $u, v \notin X$
2.  $\delta_{D'}^+(X) = \delta_D^+(X) + 1$  if  $u \notin X$  and  $v \in X$
3.  $\delta_{D'}^+(X) = \delta_D^+(X) - 1$  if  $u \in X$  and  $v \notin X$

In order to maintain connectivity, we have to reverse paths without decreasing the number of arc-disjoint directed paths between pairs of vertices too much.

**Lemma 8.** *Let  $P_{uv}$  be a directed path from vertex  $u$  to vertex  $v$  in  $D$ . For all vertices  $u', v'$ , we have  $\lambda_{D^{P_{uv}}}(u', v') \geq \min(\lambda_D(u, v) - 1, \lambda_D(u', v'))$ . Furthermore, we have  $\lambda_{D^{P_{uv}}}(u, v) = \lambda_D(u, v) - 1$ .*

*Proof.* With Menger's Theorem (Theorem 1) and Observation 1, Lemma 8 can be easily proved:

$$\begin{aligned} \lambda_{D^{P_{uv}}}(u', v') &= \min\{\delta_{D^{P_{uv}}}^+(X) \mid X \subset V, v' \notin X \ni u'\} && \text{(Theorem 1)} \\ &= \min(\{\delta_D^+(X) - 1 \mid X \subseteq V, v, v' \notin X \ni u', u\} && \text{(Observation 1)} \\ &\quad \cup \{\delta_D^+(X) \mid X \subseteq V, v' \notin X \ni u', u, v \text{ or } v', u, v \notin X \ni u'\} \\ &\quad \cup \{\delta_D^+(X) + 1 \mid X \subseteq V, u, v' \notin X \ni u', v\}) \\ &\geq \min\{\lambda_D(u, v) - 1, \lambda_D(u', v')\} && \text{(Theorem 1)} \end{aligned}$$

Note that in the case  $u' = u$  and  $v' = v$  the second and third part of the union in the above equation are empty. We thus get:

$$\begin{aligned} \lambda_{D^{P_{uv}}}(u, v) &= \min\{\delta_{D^{P_{uv}}}^+(X) \mid X \subseteq V, u' \in X, v' \notin X\} && \text{(Theorem 1)} \\ &= \min\{\delta_D^+(X) - 1 \mid X \subseteq V, u' \in X, v' \notin X\} && \text{(Observation 1)} \\ &= \lambda_D(u, v) - 1 && \text{(Theorem 1)} \end{aligned}$$

□

In a  $k$ -connected digraph  $D$  we call a directed path  $P$  *flippable* if  $D^P$  is  $k$ -connected. Lemma 8 implies that a path from  $u$  to  $v$  is flippable, if and only if all of them are which is equivalent to  $\lambda_D(u, v) > k$ . In this case we call the pair  $(u, v)$  *flippable*.

**Lemma 9.** *Let  $D$  be  $k$ -connected, it can be decided in time  $O(km)$  if  $(u, v)$  is flippable.*

*Proof.* By Lemma 8 a simple algorithm consists in finding a directed path  $P_{uv}$  in  $D$ , reverse it and iterate in  $D^{P_{uv}}$ . We have  $\lambda_D(u, v) > k$  if and only if this procedure can be applied  $k + 1$  times. Each execution is a BFS, which yields the claimed runtime. □

The following is a slight refinement of a result of Frank [17] and constitutes the last ingredient for our algorithm:

**Lemma 10.** *Let  $G = (V, E)$  be a graph and  $D, D'$  be two  $k$ -connected orientations of  $G$ . For every  $v \in V$  with  $\delta_D^+(v) < \delta_{D'}^+(v)$ , there exists  $u \in V$  such that  $\delta_D^+(u) > \delta_{D'}^+(u)$  and  $(u, v)$  is flippable in  $D$ .*

*Proof.* Let  $v \in V$  be with  $\delta_D^+(v) < \delta_{D'}^+(v)$ . We will prove that there is a  $u \in V$  such that  $\delta_D^+(u) > \delta_{D'}^+(u)$  and  $\delta_D^+(X) > k$  whenever  $u \in X$  and  $v \notin X$ . By Menger's Theorem (Theorem 1) this implies that  $(u, v)$  is flippable. Since  $\delta_D^+(v) < \delta_{D'}^+(v)$  and because  $\sum_{w \in V} \delta_D^+(w) = |E| = \sum_{w \in V} \delta_{D'}^+(w)$ , there exists at least one vertex  $u$  such that  $\delta_D^+(u) > \delta_{D'}^+(u)$ . By contradiction suppose that every vertex  $u$  with  $\delta_D^+(u) > \delta_{D'}^+(u)$ , is contained in a set  $X$  with  $\delta_D^+(X) = k$  and  $v \notin X$ . Among all these sets for all such  $u$ , we consider those that are inclusion maximal and collect them in  $\mathcal{X}$ .

Let us first prove that the members of  $\mathcal{X}$  are pairwise disjoint. Indeed, consider two sets  $X, X'$  with  $\delta_D^+(X) = \delta_D^+(X') = k$  and  $v \notin X \cup X'$  with  $X \cap X' \neq \emptyset$ . Since  $X \cup X', X \cap X'$  are both not empty,  $X \cup X' \neq V$  and  $D$  is  $k$ -connected, we know that  $\delta_D^+(X \cup X') \geq k$  and  $\delta_D^+(X \cap X') \geq k$ . Therefore, we have

$$k + k = \delta_D^+(X) + \delta_D^+(X') \geq \delta_D^+(X \cup X') + \delta_D^+(X \cap X') \geq k + k$$

This gives  $\delta_D^+(X \cup X') = k$ , i.e.,  $X$  or  $X'$  is not maximal.

Now, let us count the number  $c$  of edges of  $G$  not contained in any subgraph of  $G$  induced by  $X \in \mathcal{X}$  and denote  $Y = V \setminus \bigcup_{X \in \mathcal{X}} X$ . Observe that since the elements of  $\mathcal{X}$  are disjoint, in any orientation of  $G$  the number  $c$  is just the sum of outdegrees of  $X \in \mathcal{X}$  plus the sum of outdegrees of vertices of  $Y$ . Thus, if we furthermore denote  $t = |\mathcal{X}|$  we can do this counting with respect to  $D$  and  $D'$  and obtain the following contradiction

$$c = \sum_{X \in \mathcal{X}} \delta_D^+(X) + \sum_{w \in Y} \delta_D^+(w) = kt + \sum_{w \in Y} \delta_D^+(w) < \sum_{X \in \mathcal{X}} \delta_{D'}^+(X) + \sum_{w \in Y} \delta_{D'}^+(w) = c$$

More precisely, since  $D'$  is  $k$ -connected we have that  $kt \leq \sum_{X \in \mathcal{X}} \delta_{D'}^+(X)$ . Recall that we supposed that if  $w$  is such that  $\delta_D^+(w) > \delta_{D'}^+(w)$ , then  $w \notin Y$ . Therefore, we have  $\delta_D^+(w) \leq \delta_{D'}^+(w)$  for all  $w \in Y$ . Adding this to the fact that  $\delta_D^+(v) < \delta_{D'}^+(v)$  and  $v \in Y$ , we obtain the strict inequality claimed above. This concludes the proof.  $\square$

Note that a complete analogue of Lemma 10 holds for the case  $\delta_D^+(v) > \delta_{D'}^+(v)$ :

**Lemma 11.** *Let  $G = (V, E)$  be a graph and  $D, D'$  be  $k$ -connected orientations of  $G$ . For every  $v \in V$  with  $\delta_D^+(v) > \delta_{D'}^+(v)$ , there exists  $u \in V$  such that  $\delta_D^+(u) < \delta_{D'}^+(u)$  and  $(v, u)$  is flippable in  $D$ .*

*Proof.* Observe that an equivalent statement of the lemma is that for every vertex  $v$  such that  $\delta_D^-(v) < \delta_{D'}^-(v)$ , there exists a vertex  $u$  such that  $\delta_D^-(u) > \delta_{D'}^-(u)$  and  $(v, u)$  is flippable in  $D$ . By fully reorienting all the arcs of  $D$  and  $D'$ , we obtain graphs  $D^-$  and  $D'^-$  and get a further equivalent statement: for every vertex  $v$  such that  $\delta_{D^-}^+(v) < \delta_{D'^-}^+(v)$ , there exists a vertex  $u$  such that  $\delta_{D^-}^+(u) > \delta_{D'^-}^+(u)$  and  $(u, v)$  is flippable in  $D^-$ . This is precisely the statement of Lemma 10, so we are done.  $\square$

Now we are ready to describe the general enumeration algorithm for  $k$ -connected outdegree se-



quences:

---

**Algorithm 3:** Enumeration of  $k$ -connected outdegree sequences

---

**Input:** A graph  $G = (V, E)$ , an integer  $k$

**Output:** All  $k$ -connected outdegree sequences of  $G$

```

1 begin
2   if there exists a  $k$ -connected orientation  $D$  of  $G$  then
3     |   Fix an arbitrary linear order on  $V$ ;
4     |   EnODS( $D, \emptyset$ );
5 end

6 Function EnODS( $D, F$ ):
7   if  $F \neq V$  then
8     |   take the smallest  $v \in V \setminus F$ ;
9     |   Reverse-( $D, F, v$ );
10    |   Reverse+( $D, F, v$ );
11    |   EnODS( $D, F \cup \{v\}$ );
12  else
13    |   Output  $\delta_D^+$ ;
14 end

```

---

```

6 Function Reverse-( $D, F, v$ ):
7   if there exists  $u \in V \setminus F$  such that  $(v, u)$  is flippable then
8     |   Take a directed path  $P_{vu}$  from  $v$  to  $u$ ;
9     |   Reverse-( $D^{P_{vu}}, F, v$ );
10    |   EnODS( $D^{P_{vu}}, F \cup \{v\}$ );

```

```

11 Function Reverse+( $D, F, v$ ):
12   if there exists  $u \in V \setminus F$  such that  $(u, v)$  is flippable then
13     |   Take a directed path  $P_{uv}$  from  $u$  to  $v$ ;
14     |   Reverse+( $D^{P_{uv}}, F, v$ );
15     |   EnODS( $D^{P_{uv}}, F \cup \{v\}$ );

```

---

**Lemma 12.** *Let  $D$  be a  $k$ -connected orientation of  $G = (V, E)$  and  $F \subseteq V$ . The function **EnODS**( $D, F$ ) enumerates the  $k$ -connected outdegree sequences coinciding with  $D$  on  $F$  with delay in  $O(knm^2)$ .*

*Proof.* We show the first part of the lemma by induction on  $|V \setminus F|$ . If  $|V \setminus F| = 0$ , then **EnODS**( $D, F$ ) outputs  $\delta_D^+$  and the claim holds. Consider now the case  $|V \setminus F| > 0$  and let  $v \in V \setminus F$  be the next vertex. By induction **EnODS**( $D, F \cup \{v\}$ ) generates every  $k$ -connected outdegree sequence coinciding with  $D$  on  $F \cup \{v\}$  exactly once. We have to show that **Reverse**<sup>+</sup>( $D, F, v$ ) (resp. **Reverse**<sup>-</sup>( $D, F, v$ )) enumerates all  $k$ -connected outdegree sequences coinciding with  $D$  on  $F$  and having outdegree of  $v$  larger (resp. smaller) than  $\delta_D^+(v)$ . Also, we have to show that each of these outdegree sequences will be generated exactly once. Note that this implies that globally each solution is produced exactly once.

Let us prove this for **Reverse**<sup>+</sup>( $D, F, v$ ). So let  $D'$  be a  $k$ -connected orientation of  $G$  such that  $\delta_{D'}^+(F) \equiv \delta_D^+(F)$  and  $\delta_{D'}^+(v) < \delta_D^+(v)$ . By Lemma 10 there exists a vertex  $u \in V \setminus F$  such that  $(u, v)$  is flippable, i.e., for any path  $P_{uv}$  the orientation  $D^{P_{uv}}$  is  $k$ -connected, its outdegree sequence coincides with  $D$  on  $F$  and  $\delta_{D'}^+(v) + 1 = \delta_{D^{P_{uv}}}^+(v)$ .

We proceed by induction on  $\delta_{D'}^+(v) - \delta_D^+(v)$  to show that  $\delta^+(D')$  is enumerated exactly once. So for the base case  $\delta_{D'}^+(v) - \delta_D^+(v) = 1$  we have  $\delta_{D'}^+(v) = \delta_{D^{P_{uv}}}^+(v)$  and by induction  $\delta_{D'}^+$  will be enumerated exactly once by the next call of **EnODS**( $D^{P_{uv}}, F \cup \{v\}$ ) and not at all by **Reverse**<sup>+</sup>( $D^{P_{uv}}, F, v$ ) since the latter outputs degree sequences with  $\alpha(v) > \delta_{D^{P_{uv}}}^+(v)$ . Suppose now that  $\delta_{D'}^+(v) - \delta_D^+(v) > 1$ . We

have  $\delta_{D'}^+(v) - \delta_{D^{P_{uv}}}^+(v) < \delta_{D'}^+(v) - \delta_D^+(v)$ , so by induction hypothesis  $\text{Reverse}^+(D^{P_{uv}}, F, v)$  enumerates the outdegree sequence of  $\delta_{D'}^+(v)$  exactly once.

The analogue proof works for  $\text{Reverse}^-$  using Lemma 11.

For the analysis of complexity note that in each call of  $\text{Reverse}^+$  or  $\text{Reverse}^-$  for at most  $n$  times it has to be checked if a pair  $(u, v)$  is flippable. The latter can be done in time  $O(km)$  by Lemma 9, finding a directed path from  $u$  to  $v$  is done in  $O(m)$ . So a call costs  $O(knm)$ .

Finally, the depth of the recursion tree is in  $O(m)$ . To see this compare the  $\delta_{D'}^+$  of a leaf orientation with the  $\delta_D^+$  of orientation  $D$  at the root. Between any two calls of  $\text{EnODS}$ , there will be a sequence of at most  $\deg(v)$  calls of  $\text{Reverse}^+$  or  $\text{Reverse}^-$ . This way  $\delta_D^+$  will be approached to  $\delta_{D'}^+$ , coordinate by coordinate, where previous coordinates are not affected by modifications on latter coordinates. Thus, there are at most  $\sum_{v \in V} \deg(v) = 2m$  calls and we get an overall delay of  $O(knm^2)$ .  $\square$

Note that, Algorithm 4 has to use a separate method for finding a first  $k$ -connected orientation  $D$  of  $G$ . This preprocessing step can be done in  $O(k^3n^3 + kn^2m)$  [26]. On the other hand, recall that in a  $k$ -connected orientation we have  $kn \leq m$ . Therefore, together with Lemma 12 we obtain:

**Theorem 13.** *Let  $G$  be a graph and  $k \in \mathbb{N}$ . Algorithm 4 enumerates all  $k$ -connected outdegree sequences of  $G$  in delay  $O(knm^2)$ .*

## 6 $k$ -connected orientations

Putting the above together we obtain an algorithm to enumerate  $k$ -connected orientations.

---

**Algorithm 4:** Simple enumeration of  $k$ -connected orientations

---

**Input:** A graph  $G = (V, E)$ , an integer  $k$

**Output:** All  $k$ -connected orientations of  $G$

```

1 begin
2   if there exists a  $k$ -connected orientation  $D$  of  $G$  then
3     Fix an arbitrary linear order on  $V$ ;
4     EnODS' ( $D, \emptyset$ );
5   end

6 Function EnODS' ( $D, F$ ):
7   if  $F \neq V$  then
8     take the smallest  $v \in V \setminus F$ ;
9     Reverse' ( $D, F, v$ );
10    Reverse' ( $D, F, v$ );
11    EnODS' ( $D, F \cup \{v\}$ );
12  else
13    EnOPODS ( $D, \emptyset$ );
14 end
```

---

We need the following easy result for analyzing the amortized complexity:

**Lemma 14.** *Let  $G = (V, E)$  be a graph and  $\alpha$   $k$ -connected, then  $|\mathcal{O}_\alpha(G)| \geq (k-1)n + 2$ .*

*Proof.* Let  $D \in \mathcal{O}_\alpha(G)$ . We will show that  $D$  contains at least  $(k-1)n + 1$  directed cycles. Since for each directed cycle  $C$ , the orientation  $D^C$  is a different element of  $\mathcal{O}_\alpha(G)$ , we obtain the result.

The *cycle space* of  $D$  is the set of vectors in  $\mathbb{R}^m$  that are linear combinations of *signed incidence vectors* of cycles of  $D$ . Given a cycle  $C$  with a direction of traversal its signed incidence vector has an entry for every arc of  $D$ , which is 1 if  $a$  is traversed forward,  $-1$  if it is traversed backward and 0 if  $a \notin C$ . It is well known, that the cycle space of a strongly connected  $D$  can be generated by linear combinations of the vectors associated to its directed cycles, see e.g. [22]. Moreover, it can be found in most books on (algebraic) graph theory that the dimension of the cycle space of a weakly connected

digraph is  $m - n + 1$ , see e.g. [23, 27]. In particular,  $D$  has at least  $m - n + 1$  directed cycles. Since  $D$  is  $k$ -connected it has at least  $kn$  edges. Therefore  $D$  contains at least  $(k - 1)n + 1$  directed cycles.  $\square$

**Theorem 15.** *Let  $G$  be a graph and  $k \in \mathbb{N}$ . Algorithm 6 enumerates all  $k$ -connected orientations of  $G$  with delay  $O(knm^2)$ . If  $k \geq 2$  the amortized time is in  $O(m^2)$ .*

*Proof.* The correctness and the delay follow directly from Theorem 13 and Theorem 6. Let us compute the amortized time complexity as an average over the delays. Let  $s$  be the number of solutions, i.e., the total number of  $k$ -connected orientations and  $t$  be the number of  $k$ -connected outdegree sequences of  $G$ . Since by Lemma 14 for every  $k$ -connected outdegree sequence  $\alpha$  there are at least  $(k - 1)n + 2$  orientations, we have that  $t \leq \frac{s}{(k-1)n+2}$ . Thus there exist constants  $c$  and  $c'$ , such that the overall runtime of our algorithm is bounded by  $cknm^2t + c'm^2s \leq cknm^2 \frac{s}{(k-1)n} + c'm^2s = O(m^2)s$ , where for the last equality we use  $k \geq 2$ . Hence the amortized complexity is in  $O(m^2)$ .  $\square$

## 7 Discussion

We have given a simple algorithm for enumerating the  $k$ -arc-connected orientations of a graph with a low amortized time complexity. The weakness of our approach is that finding the initial solution, i.e., finding a  $k$ -connected orientation of a graph, is algorithmically the most complicated part. The best implementations using splitting off techniques are rather simple and lead to computation time of roughly  $O(n^5)$  [19, 29], it would be interesting to find simpler methods.

Our original motivation was the hunt for counterexamples to a conjecture of Neumann-Lara [33] stating that the vertex set of every oriented planar graph (that is having no directed cycles of length 2) can be vertex-partitioned into two subsets each inducing an acyclic digraph. It is not hard to see that a minimal counterexample to this conjecture has to be a 2-arc-connected (planar) digraph. However, with a little more work one can actually prove that a minimal counterexample has to be 2-vertex-connected, i.e., at least two vertices have to be removed in order to destroy strong connectivity. Deciding if a graph admits a  $k$ -vertex-connected orientation can be done in polynomial time if and only if  $k \leq 2$  (unless  $P=NP$ ), see [11, 39]. Thus, this enumeration problem would be viable only for  $k = 2$ . In [2] it is posed as open problem whether it can be decided in polynomial time, if a mixed graph can be oriented such that it is 2-vertex-connected. A positive answer would allow polynomial time enumeration as in Algorithm 1. Note that concerning our initial motivation it would be interesting to answer this question even restricted to planar graphs.

A dual analogue of  $k$ -arc-connectivity could be called  $k$ -acyclicity, where at least  $k$  arcs of  $D$  have to be contracted in order to destroy its acyclicity. As mentioned in the introduction, acyclic orientations can be enumerated with polynomial delay. On the other hand, it is easy to see that a graph  $G$  admits a 2-acyclic orientation if and only if  $G$  is the cover graph of a poset. The corresponding recognition problem is NP-complete [6] and the proof can be extended to see that testing whether  $G$  admits a  $k$ -acyclic orientation is NP-complete for any  $k \geq 2$ . Using [10, Theorem 13] one can show that enumerating  $k$ -acyclic orientations (as well as  $k$ -vertex-connected orientations for  $k > 2$ ) are DelNP-hard under D-reductions and IncNP-hard under I-reduction. Since with an NP-oracle one can decide if a given partial orientation extends to one of these types, a backtrack search algorithm like Algorithm 1 yields that the above problems can be solved with delay a polynomial number of calls of an NP-oracle. Thus these problems are in the class DelNP (and a fortiori IncNP), and therefore are indeed complete in both settings.

A way of relaxing acyclicity is to consider orientations of *digirth* at least  $d$ , i.e., all directed cycles are of length at least  $d$ . We believe that in this setting polynomial delay enumeration should be possible, maybe even when combined with an arc-connectivity requirement.

### Acknowledgements:

We wish to thank Nadia Creignou and Frédéric Olive for fruitful discussions in an early stage of this paper. A preliminary version of the results obtained in this work was presented at *WEPA 2018: Second Workshop on Enumeration Problems and Applications* which was held in Pisa on November 2018. The

authors wish to thank the organizers of this workshop. The first author was supported by the French ANR project GraphEn: ANR-15-CE40-0009. The second and third authors were supported by the French ANR project DISTANCIA: ANR-17-CE40-0015. The second author moreover was supported by ANR grant GATO: ANR-16-CE40-0009-01 and by the Spanish Ministerio de Economía, Industria y Competitividad through grant RYC-2017-22701.

## References

- [1] O. AICHHOLZER, J. CARDINAL, T. HUYNH, K. KNAUER, T. MÜTZE, R. STEINER, AND B. VOGTENHUBER, *Flip distances between graph orientations*, in Graph-Theoretic Concepts in Computer Science, I. Sau and D. M. Thilikos, eds., Cham, 2019, Springer International Publishing, pp. 120–134.
- [2] J. BANG-JENSEN, J. HUANG, AND X. ZHU, *Completing orientations of partially oriented graphs*, Journal of Graph Theory, 87 (2018), pp. 285–304.
- [3] V. C. BARBOSA AND J. L. SZWARCFITER, *Generating all the acyclic orientations of an undirected graph*, Information Processing Letters, 72 (1999), pp. 71 – 74.
- [4] S. BLIND, *Output-sensitive algorithms for enumeration problems in graphs*, PhD thesis, Université de Lorraine, 2019.
- [5] E. BREHM, *3-orientations and Schnyder-3-tree-decompositions*, 2000. Diploma Thesis, FU Berlin.
- [6] G. BRIGHTWELL, *On the complexity of diagram testing*, Order, 10 (1993), pp. 297–303.
- [7] A. CONTE, *Enumeration Algorithms for Real-World Networks: Efficiency and Beyond*, PhD thesis, Università di Pisa, 2018.
- [8] A. CONTE, R. GROSSI, A. MARINO, AND R. RIZZI, *Efficient enumeration of graph orientations with sources.*, Discrete Appl. Math., 246 (2018), pp. 22–37.
- [9] A. CONTE, R. GROSSI, A. MARINO, R. RIZZI, AND L. VERSARI, *Directing Road Networks by Listing Strong Orientations*, Springer International Publishing, 2016, pp. 83–95.
- [10] N. CREIGNOU, M. KRÖLL, R. PICHLER, S. SKRITEK, AND H. VOLLMER, *A complexity theory for hard enumeration problems*, Discrete Applied Mathematics, 268 (2019), pp. 191 – 209.
- [11] O. D. DE GEVIGNEY, *On Frank’s conjecture on  $k$ -connected orientations*, Journal of Combinatorial Theory, Series B, (2019).
- [12] G. EHRLICH, *Loopless algorithms for generating permutations, combinations, and other combinatorial configurations*, J. ACM, 20 (1973), pp. 500–513.
- [13] S. FELSNER, *Lattice structures from planar graphs*, Electron. J. Combin., 11 (2004). Research Paper 15.
- [14] S. FELSNER, *Rectangle and square representations of planar graphs*, in Thirty essays on geometric graph theory, Springer, New York, 2013, pp. 213–248.
- [15] S. FELSNER, H. SCHREZENMAIER, AND R. STEINER, *Pentagon contact representations*, Electron. J. Combin., 25 (2018). Paper 3.39, 38.
- [16] A. FRANK, *An algorithm for submodular functions on graphs*, Ann. Discrete Math., 16 (1982), pp. 97–120.
- [17] A. FRANK, *A note on  $k$ -strongly connected orientations of an undirected graph*, Discrete Math., 39 (1982), pp. 103–104.
- [18] H. N. GABOW, *A framework for cost-scaling algorithms for submodular flow problems*, in 34th Annual Symposium on Foundations of Computer Science, Palo Alto, USA, November 1993, 1993, pp. 449–458.
- [19] H. N. GABOW, *Efficient splitting off algorithms for graphs*, in Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing, STOC ’94, New York, NY, USA, 1994, ACM, pp. 696–705.
- [20] H. N. GABOW, *Centroids, representations, and submodular flows*, J. Algorithms, 18 (1995), pp. 586 – 628.
- [21] P. M. GILMER AND R. A. LITHERLAND, *The duality conjecture in formal knot theory*, Osaka J. Math., 23 (1986), pp. 229–247.
- [22] P. M. GLEISS, J. LEYDOLD, AND P. F. STADLER, *Circuit bases of strongly connected digraphs.*, Discuss. Math., Graph Theory, 23 (2003), pp. 241–260.
- [23] C. GODSIL AND G. ROYLE, *Algebraic graph theory.*, vol. 207, New York, NY: Springer, 2001.

- [24] D. GONÇALVES, B. LÉVÊQUE, AND A. PINLOU, *Triangle contact representations and duality*, Discrete Comput. Geom., 48 (2012), pp. 239–254.
- [25] M. HABIB, R. MEDINA, L. NOURINE, AND G. STEINER, *Efficient algorithms on distributive lattices*, Discrete Applied Mathematics, 110 (2001), pp. 169 – 187.
- [26] S. IWATA AND Y. KOBAYASHI, *An algorithm for minimum cost arc-connectivity orientations*, Algorithmica, 56 (2010), pp. 437–447.
- [27] U. KNAUER AND K. KNAUER, *Algebraic graph theory. Morphisms, monoids and matrices (to appear). 2nd revised and extended edition.*, vol. 41, Berlin: De Gruyter, 2nd revised and extended edition ed., 2019.
- [28] P. C. B. LAM AND H. ZHANG, *A distributive lattice on the set of perfect matchings of a plane bipartite graph*, Order, 20 (2003), pp. 13–29.
- [29] L. C. LAU AND C. K. YUNG, *Efficient edge splitting-off algorithms maintaining all-pairs edge-connectivities*, SIAM J. Comput., 42 (2013), pp. 1185–1200.
- [30] L. LOVÁSZ, *Combinatorial Problems and Exercises*, North-Holland, 1979.
- [31] K. MENGER, *Zur allgemeinen Kurventheorie*, in Fundamental Mathematics, 1927, pp. 96–115.
- [32] C. S. J. A. NASH-WILLIAMS, *On orientations, connectivity and odd-vertex-pairings in finite graphs*, Canadian Journal of Mathematics, 12 (1960), p. 555–567.
- [33] V. NEUMANN-LARA, *Vertex colourings in digraphs, Some Problems*, tech. report, Waterloo, Canada, 1985.
- [34] J. B. ORLIN, *Max flows in  $O(nm)$  time, or better.*, in Proceedings of the 45th annual ACM symposium on theory of computing, STOC '13. Palo Alto, CA, USA, June 1–4, 2013, 2013, pp. 765–774.
- [35] J. PROPP, *Lattice structure for orientations of graphs*. ArXiv: math/0209005, Sept. 2002.
- [36] G. PRUESSE AND F. RUSKEY, *Gray codes from antimatroids*, Order, 10 (1993), pp. 239–252.
- [37] E. RÉMILA, *The lattice structure of the set of domino tilings of a polygon*, Theoret. Comput. Sci., 322 (2004), pp. 409–422.
- [38] M. B. SQUIRE, *Generating the acyclic orientations of a graph*, J. Algorithms, 26 (1998), pp. 275 – 290.
- [39] C. THOMASSEN, *Strongly 2-connected orientations of graphs*, Journal of Combinatorial Theory, Series B, 110 (2015), pp. 67 – 78.
- [40] W. P. THURSTON, *Conway’s tiling groups*, Amer. Math. Monthly, 97 (1990), pp. 757–773.