Mathematical Programming
https://doi.org/10.1007/s10107-026-02365-x

FULL LENGTH PAPER

Series A

Single-source unsplittable flows in planar and bounded-genus graphs

Vera Traub[1] · **Laura Vargas Koch**[2] · **Rico Zenklusen**[3]

Received: 19 February 2025 / Accepted: 21 April 2026
© The Author(s) 2026, modified publication 2026

Abstract

The single-source unsplittable flow (SSUF) problem asks to send flow from a common source to terminals with unrelated demands, each terminal being served through a single path. The classical SSUF objective is to minimize the violation of some given arc capacities. A seminal result of Dinitz, Garg, and Goemans showed that, whenever a fractional flow exists respecting the capacities, then there is an unsplittable one violating the capacities by at most the maximum demand. Goemans conjectured a natural cost version of the same result, where the unsplittable flow is required to be no more expensive than the fractional one. Intriguingly, there are arguably no non-trivial graph classes for which it is known to hold. We show that a slight weakening of it holds for planar graphs, by exploiting a connection to a highly structured discrepancy problem. Moreover, our techniques extend to simultaneous upper and lower bounds on the flow values. This affirmatively answers a conjecture of Morell and Skutella for planar SSUF. Finally, we show that our approach can be extended to general (non-planar) graphs with a capacity violation that depends on the genus.

Keywords Unsplittable flow · Planar graphs · Combinatorial Optimization

Mathematics Subject Classification 68W25 · 05C85 · 68W01

A preliminary version of this work appeared at the ACM-SIAM Symposium on Discrete Algorithms (SODA 2024). The first author has been supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 537750605. The third author received funding from Swiss National Science Foundation grant 200021_184622 and the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 817750).

✉ Laura Vargas Koch
vargaskoch@gdm.rwth-aachen.de

Vera Traub
vtraub@ethz.ch

Rico Zenklusen
ricoz@ethz.ch

1 Department of Computer Science, ETH Zurich, Zürich, Switzerland

2 RWTH Aachen University, Aachen, Germany

3 Department of Mathematics, ETH Zurich, Zürich, Switzerland

Published online: 22 July 2026

1 Introduction

Flow problems are among the most classical combinatorial optimization problems. To their prominent role in the field significantly contributed both their wide, and sometimes surprising, range of applications and also the existence of very fast algorithms (in practice as well as in theory) for basic flow optimization problems. (We refer the interested reader to the textbooks of Ahuja, Magnanti, and Orlin [1], and Williamson[33] for more information.) In their most canonical version, flow from a source to a sink can be split over many paths. Being able to split flows naturally allows for casting flow problems—even with multiple sources, sinks, and commodities—as linear programs. Flow problems become much harder, and are also much less understood, as soon as the flow is not allowed to be split, i.e., is unsplittable.

In this work, we consider a heavily studied variant thereof, namely the *single-source unsplittable flow problem* (SSUF), originally introduced by Kleinberg[15, 16]. Here, we are given a directed graph $G = (V, A)$, with arc capacities $u \in \mathbb{Q}^A_{\geq 0}$, a single source $s \in V$, and terminals/sinks $T \subseteq V$, where each $t \in T$ has a demand $d_t \in \mathbb{Q}_{\geq 0}$. (The notions of sink and terminal are used interchangeably in the literature.) For simplicity, and without loss of generality, we assume that the source s and sinks T are all distinct. Ideally, one would like to route the demanded flow from the source to each terminal on a single path such that all capacities are respected. Formally, this corresponds to determining one s-t path $P^t \subseteq A$ for each $t \in T$ such that $\sum_{t \in T\colon a \in P^t} d_t \leq u(a)$ for each $a \in A$. Deciding whether an *unsplittable flow*, i.e., such a family $\mathcal{P} = \{P^t\}_{t \in T}$ of paths, exists, can easily be seen to be NP-hard. Actually, even in a 2-vertex graph, one can reduce Bin Packing or Subset Sum to it. Thus, significant attention has been devoted to the setting where one is allowed to exceed the capacities and the goal is to minimize the largest capacity violation. This is also the setting we consider here.

The research in this area has been largely influenced by a seminal result of Dinitz, Garg, and Goemans [6] and a subsequent conjecture of Goemans, which we discuss next. These are statements about the existence of unsplittable flows with limited capacity violation assuming that there exists a *splittable flow* $x \in \mathbb{Q}^A$ that respects the capacities, i.e., such a vector x satisfies $x(a) \leq u(a)$ for all arcs $a \in A$ and

$$x(\delta^+(v)) - x(\delta^-(v)) = \begin{cases} \sum_{t \in T} d_t & \text{if } v = s, \\ -d_v & \text{if } v \in T, \\ 0 & \text{if } v \in V \setminus (\{s\} \cup T), \end{cases}$$

where $\delta^-(v)$ and $\delta^+(v)$ denote the sets of all arcs entering and leaving v, respectively. More precisely, these statements assume the existence of a splittable flow $x \in \mathbb{Q}^A_{\geq 0}$, and the capacity violation of an unsplittable flow is measured by the difference between the flow value of the unsplittable flow and the value of the splittable one. In short, this can be interpreted as a worst-case assumption where the capacity $u(a)$ of an arc a is equal to $x(a)$. We thus define an SSUF instance as follows.

Definition 1.1 *(Single-source unsplittable flow (SSUF) instance)* A *single-source unsplittable flow (SSUF)* instance is a tuple $(G = (V, A), s, T, d, x)$, where G is a directed graph with source $s \in V$, terminals $T \subseteq V$ with corresponding demands

$d \in \mathbb{Q}^T_{\geq 0}$, and a splittable flow $x \in \mathbb{Q}^A_{\geq 0}$. The source s and terminals T are assumed to be distinct.

The result of Dinitz, Garg, and Goemans [6], stated below, was the first to establish an additive capacity violation of $O(d_{\max})$, where $d_{\max} := \max\{d_t : t \in T\}$ is the largest demand, whenever a splittable flow exists. For ease of notation, given an unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ and an arc $a \in A$, we denote by

$$\text{flow}_{\mathcal{P}}(a) := \sum_{t \in T\,:\, a \in P^t} d_t$$

the total flow value of the unsplittable flow $\mathcal{P}$ that traverses a.

Theorem 1.2 ([6]) *Given an SSUF instance (G, s, T, d, x), one can compute in polynomial time an unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ with $\text{flow}_{\mathcal{P}}(a) \leq x(a) + d_{\max}$ for all $a \in A$.*

Shortly thereafter, Goemans conjectured that the following stronger, cost-enhanced version of the same result holds.

Conjecture 1.3 (Goemans) *Given an SSUF instance (G, s, T, d, x) and a cost vector $c \in \mathbb{Q}^A_{\geq 0}$, one can compute in polynomial time an unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ with*

(i) $\text{flow}_{\mathcal{P}}(a) \leq x(a) + d_{\max}$ *for all* $a \in A$, *and*
(ii) cost at most $c^T x$, *i.e.,* $\sum_{a \in A} c(a)\,\text{flow}_{\mathcal{P}}(a) \leq \sum_{a \in A} c(a)x(a)$.

Note that one can assume G to be acyclic in Conjecture 1.3. Indeed, if G contains directed cycles, then flow along such cycles can be reduced and zero-flow arcs can be removed. This leads to an acyclic instance that is no easier than the original one.

The following even stronger version of Goemans' conjecture, which includes both upper and lower bounds on the flow values, has been stated by Morell and Skutella [25].

Conjecture 1.4 ([25]) *Given an SSUF instance (G, s, T, d, x) on an acyclic graph G and a cost vector $c \in \mathbb{Q}^A_{\geq 0}$, one can compute in polynomial time an unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ with*

(i) $x(a) - d_{\max} \leq \text{flow}_{\mathcal{P}}(a) \leq x(a) + d_{\max}$ *for all* $a \in A$, *and*
(ii) $\sum_{a \in A} c(a)\,\text{flow}_{\mathcal{P}}(a) \leq \sum_{a \in A} c(a)x(a)$.

The fact that G is explicitly assumed to be acyclic in Conjecture 1.4 is due to the lower bounds. Without acyclicity, the nature of the problem changes. In particular, it would suddenly become important whether an unsplittable flow consists of paths or walks, because one may try to fulfill lower bounds by going along cycles. Also, one can show that statements as Conjecture 1.4 (even with a violation of $O(d_{\max})$ instead of just $d_{\max}$) cannot be obtained in general (non-acyclic) graphs (see Appendix A.1).

Morell and Skutella [25] also explicitly conjectured the following weaker version of Conjecture 1.4 without costs, which also remains open and is a natural intermediate step toward Conjecture 1.4.

Conjecture 1.5 ([25]) *Given an SSUF instance (G, s, T, d, x) on an acyclic graph G, there exists an unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ with*

$$x(a) - d_{\max} \leq \text{flow}_{\mathcal{P}}(a) \leq x(a) + d_{\max} \text{ for all } a \in A.$$

Conjecture 1.5 can be seen as the counterpart of Theorem 1.2 with additional lower bounds. One motivation for Conjecture 1.5 is that a positive resolution of it would have implications to both some open discrepancy and scheduling problems, on which we briefly expand in Section 1.2.

All three conjectures, Conjecture 1.3 to 1.5, remain open, and even a resolution of a weaker version of Goemans' conjecture with additive capacity violations in the order $O(d_{\max})$ (instead of the conjectured $d_{\max}$, which is the best one can hope for in terms of constants) would likely be considered a breakthrough.

Procedures leading to an unsplittable flow with a multiplicative violation, i.e., $\text{flow}_{\mathcal{P}}(a) = O(x(a))$ for $a \in A$, have been obtained previously [15, 16, 19, 25, 29]. Several of these approximation algorithms have also been implemented and empirically tested [7]. Moreover, Skutella [29] showed that Conjecture 1.3 holds when the demands are all multiples of each other. Furthermore, techniques presented in Lenstra, Shmoys, and Tardos [22] solve the problem for the special case when G has a source plus two layers of vertices, with arcs only going from the source to the first layer and from the first to the second layer.

Very little is known about Conjecture 1.4 and 1.5. When only dealing with lower bounds, analogous results (for acyclic graphs) can be obtained as with upper bounds only. More precisely, Morell and Skutella [25] showed that for any SSUF instance (G, s, T, d, x), there exists an unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ with $\text{flow}_{\mathcal{P}}(a) \geq x(a) - d_{\max}$ for $a \in A$. Moreover, one can also adjust the algorithm of Dinitz, Garg, and Goemans [6] to get the same lower bound guarantees. However, the combination of both lower and upper bounds seems to be much more challenging.

Prior to this work, there was arguably no non-trivial graph class for which any of Conjecture 1.3 to 1.5 was known to hold, even if additive violations in the order of $O(d_{\max})$ are allowed instead of only $d_{\max}$. The goal of this work is to address this gap, by showing that a slightly weaker version of Conjecture 1.4 (allowing a violation of $2d_{\max}$ instead of $d_{\max}$) and Conjecture 1.5 hold for planar graphs, and more generally for bounded-genus graphs. This is a further positive sign regarding Conjecture 1.3 to 1.5. We obtain this result through a connection to a very well-structured discrepancy problem, whose resolution allows us to successively transform a splittable flow into another one that uses fewer paths per terminal until, eventually, an unsplittable flow is obtained.

1.1 Our results

We start by focusing on planar graphs, before generalizing to bounded-genus graphs. This leads to a cleaner and more intuitive presentations of the main ideas of our approach, and also allows us to present improved constants for the planar case.

A planar instance of SSUF is formally defined as follows.

Definition 1.6 *(Planar single-source unsplittable flow (PSSUF) instance)* An SSUF instance (G, s, T, d, x) is a *planar single-source unsplittable flow (PSSUF)* instance if G is acyclic and planar.

Our first main result shows that, for planar graphs, unsplittable flows can be constructed with the lower and upper bound guarantees as claimed in Conjecture 1.5.

Theorem 1.7 *Given a PSSUF instance (G, s, T, d, x), there is an unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ with*

$$x(a) - d_{\max} \leq \text{flow}_{\mathcal{P}}(a) \leq x(a) + d_{\max} \qquad \forall a \in A.$$

Moreover, it can be computed in time $O(|V|^2)$.

We obtain an analogous result involving costs when allowing for a slightly larger lower and upper bound violation of $2d_{\max}$ instead of $d_{\max}$.

Theorem 1.8 *Given a PSSUF instance (G, s, T, d, x) and arc costs $c \in \mathbb{Q}^A_{\geq 0}$, there is an unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ satisfying*

(i) $x(a) - 2d_{\max} \leq \text{flow}_{\mathcal{P}}(a) \leq x(a) + 2d_{\max} \quad \forall a \in A$, *and*
(ii) $\sum_{a \in A} c(a) \text{flow}_{\mathcal{P}}(a) \leq \sum_{a \in A} c(a)x(a)$.

Moreover, it can be computed in time $O(\langle \text{input} \rangle \cdot |V|)$, where $\langle \text{input} \rangle$ denotes the input size of the instance.

Theorem 1.8 settles Conjecture 1.4 for planar graphs, when allowing for a violation of $2d_{\max}$ instead of $d_{\max}$. Moreover, the above results are the first SSUF results to simultaneously respect upper and lower bounds with only additive errors of $O(d_{\max})$ in a non-trivial graph class. We provide an overview of our techniques, which are based on a reduction to a well-structured discrepancy problem, in Section 2.

Moreover, as mentioned, our approach can be extended to general (not necessarily planar) SSUF instances, where we obtain results parameterized by the genus of the graph. We recall that the genus of a graph G is the smallest value $g \in \mathbb{Z}_{\geq 0}$, such that G can be embedded on a sphere with g handles without any intersecting edges. More precisely, we obtain the following statement, which can be thought of as a generalization of Theorem 1.7 to bounded-genus graphs, with a weaker constant for the capacity violation.

Theorem 1.9 *Let (G, s, T, d, x) be an SSUF instance with arc costs $c \in \mathbb{Q}^A_{\geq 0}$, and let g be the genus of G. Then there is an unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ satisfying*

(i) $x(a) - O(g^2 \log g)d_{\max} \leq \text{flow}_{\mathcal{P}}(a) \leq x(a) + O(g^2 \log g)d_{\max} \quad \forall a \in A$, *and*
(ii) $\sum_{a \in A} c(a) \text{flow}_{\mathcal{P}}(a) \leq \sum_{a \in A} c(a)x(a)$.

Moreover, for constant g, such an unsplittable flow $\mathcal{P}$ can be computed in polynomial time.

1.2 Connections to discrepancy problems and scheduling

SSUF is closely connected to scheduling problems. Already Kleinberg [15] observed that a natural special case of makespan minimization on unrelated machines can be cast as SSUF with upper bounds only. When allowing for upper and lower bounds, an interesting more general scheduling problem can be captured as observed by Lars Rohwedder. This is a maximum flow time minimization problem with release times, where each job can only be processed on a subset of machines. It remains open whether this setting allows for $O(1)$-approximations, and a constructive resolution of Conjecture 1.5, even with a violation of $O(d_{\max})$ instead of $d_{\max}$, would resolve this question. (See [25] for more details including a discussion of Rohwedder's reduction.) The currently best known approximation factor is $O(\log n)$ by Bansal and Kulkarni [2], while the best known lower bound for the variant with unrelated machines is $3/2$ (unless P = NP) by Lenstra, Shmoys, and Tardos [22]. Moreover, Bansal, Rohwedder, and Svensson [4] showed very recently that the integrality gap of the natural LP relaxation for this scheduling problem is $O(\sqrt{\log n})$. (A non-constructive resolution of Conjecture 1.5 would imply an $O(1)$ integrality gap for this LP relaxation.)

Furthermore, in [4] an intriguing equivalence between the existence of an LP-based $O(1)$-approximation for this scheduling problem and the resolution of an interesting open discrepancy problem was revealed. This discrepancy problem is a special case of the prefix version of Beck-Fiala, which is a well-studied discrepancy problem. A positive answer to Conjecture 1.5 would imply a positive answer to the open discrepancy problem, in an interesting special case, which would be of independent interest.

Interestingly, in our algorithm for PSSUF, we also rely on a prefix version of a discrepancy problem. However, our notion of prefixes is different from the one in the prefix Beck-Fiala problem.

1.3 Further related work

Martens, Salazar, and Skutella [26] took an approach toward Goemans' conjecture where they allow for slight demand modifications. More precisely, they provide a polynomial-time procedure that, given any splittable single-source flow, writes it as a convex combination of unsplittable flows for slightly rounded demands (by a factor of at most 2), such that the average demands used in the convex combination correspond to the original ones. If each term in the convex combination sent the original demands, then this would imply Goemans' conjecture. (Actually the existence of such a convex combination is equivalent to the existence of an unsplittable flow fulfilling the conditions of Goemans' conjecture, see, e.g., [26].)

Also, we would like to mention that there has been extensive work on unsplittable flows in a variety of settings, including with multiple sources and sinks and with different objectives. We refer the interested reader to the survey by Kolliopoulos [18] and the discussion and references in three very recent contributions, namely the work of Grandoni, Mömke, and Wiese [11] on unsplittable flows on a path, the work of Majthoub Almoghrabi, Skutella, and Warode [27] on unsplittable multiflows on series-parallel digraphs and the work of Shapley and Shmoys [31] on unsplittable

multiflows on outerplanar graphs. Moreover, intermediate notions between splittable and unsplittable flows have been considered. In particular, k-splittable flows, which have been introduced by Baier, Köhler, and Skutella [3], allow for splitting the flow across up to k paths. (See [17, 20, 30] for further results on k-splittable flows.)

1.4 Organization of the paper

In Section 2, we provide an overview of our approach for the planar setting. A key ingredient is the construction of a nice path decomposition of the fractional flow, which we discuss in Section 3. Section 4 discusses and proves properties of nice path decompositions, which we crucially exploit later. In Section 5, we present polynomial time algorithms to solve the highly structured prefix discrepancy problem that we need to solve to determine an appropriate selection of paths for the unsplittable flow problem, both for the variant with costs and without costs. This completes all the details we need for our PSSUF results. In Section 6, we show how our results can be generalized to arbitrary (not necessarily planar) graphs by parameterizing them by the genus of the graph, thus obtaining Theorem 1.9. We conclude in Section 7 and, finally, Appendix A.1 shows why statements like Theorem 1.7 or Conjecture 1.4 cannot be extended to cyclic graphs, even when allowing for a violation of $O(d_{\max})$ instead of $d_{\max}$.

2 Overview

To simplify the presentation, we assume in the following that no terminal has an outgoing arc. This is without loss of generality, because we can add for each $t \in T$ a new vertex t' together with an arc $a_t = (t, t')$ with $x(a_t) := d_t$ to obtain an equivalent PSSUF instance with terminal set $\{t' : t \in T\}$. (Note that this modification indeed preserves planarity.)

In order to prove Theorem 1.7 and Theorem 1.8, we will proceed in two steps. First, we carefully decompose the given flow x into paths starting at s, i.e., we compute paths $P_1, \dots, P_\ell$ with weights $\lambda_i > 0$ for $i \in [\ell]$ such that

$$x = \sum_{i=1}^{\ell} \lambda_i \cdot \chi^{P_i},$$

where $\chi^{P_i} \in \{0, 1\}^A$ denotes the incidence vector of P_i and $[\ell] := \{1, \dots, \ell\}$. (Here we identify a path with its arc set.) Because G is acyclic, we have $x(\delta^+(s)) = \sum_{i=1}^{\ell} d_i$ and hence every path P_i ends at some terminal. In a second step, we then select for each terminal $t \in T$ a path P^t from the set

$$\mathcal{P}^t := \{P_i : i \in [\ell] \text{ and } P_i \text{ ends at terminal } t\}.$$

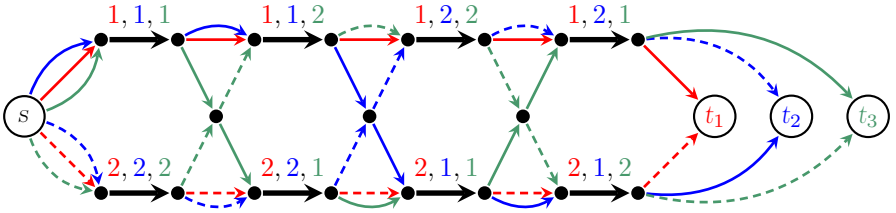


Fig. 1 The figure shows an example of a path decomposition of some flow x, which we define through the path decomposition in this example. For each terminal $t \in T = \{t_1, t_2, t_3\}$, there are two s-t paths P_t^1 and P_t^2 shown in the color of terminal t, where P_t^1 is drawn as solid arcs and P_t^2 as dashed ones. For any combination $r \in \{1, 2\}^T$ of first/second paths for each terminal, there is one black arc traversed by precisely this combination. (The corresponding combination r is highlighted above the black arcs.) Every path has weight 1 in our path decomposition of x. Thus, $x(a) = |T|$ for each black arc a, and $d_t = 2$ for all $t \in T$. Then, no matter how we choose the paths $P^t \in \mathcal{P}^t = \{P_t^1, P_t^2\}$, there is an arc a such that $\text{flow}_{\mathcal{P}}(a) = 2|T| = x(a) + \frac{1}{2}|T| \cdot d_{\max}$. Note that this construction canonically extends to more than three terminals

Figure 1 shows that it is crucial to choose the path decomposition of the flow x carefully for this approach to have a chance to lead to an unsplittable flow as claimed in Theorems 1.7 and 1.8.

Our flow decomposition will be chosen such that the paths are pairwise non-crossing in a geometric sense. We will number the paths according to the order in which they leave the source s (in a fixed planar embedding of our graph). We show that our choice of the flow decomposition implies useful structural properties of the sets $\mathcal{P}^t$ (see Lemma 2.3) and the sets $\{i \in [\ell] : a \in P_i\}$ (see Lemma 2.14).[1] These allow for reducing the selection of the paths $P^t \in \mathcal{P}^t$ to a highly structured discrepancy problem.

In the remainder of this section we provide a more detailed description of our approach. We first describe the choice of our flow decomposition and the resulting structure of the sets $\mathcal{P}^t$ in Section 2.1 and then describe the discrepancy problem and our solution for it in Section 2.2. Finally, we show how the structure of the sets $\{i \in [\ell] : a \in P_i\}$ allows for reducing the selection of the paths $P^t \in \mathcal{P}^t$ to this discrepancy problem.

2.1 Choosing the path decomposition

We now provide an overview of how we choose the path decomposition of the flow x. Further details and proofs will be provided in Section 3.

We start by computing a planar embedding of the graph G, i.e., an embedding of the vertices and arcs in the plane such that vertices are mapped to distinct points and no two arcs intersect in a point distinct from their endpoints. This can be done in linear time (see [5, 14]). Our goal will be to choose the paths $P_1, \ldots, P_\ell$ such that they do not cross in a geometric sense. To this end, it is helpful to first represent the paths $P_1, \ldots, P_\ell$ in an auxiliary graph $H = (V, F)$ obtained from $G = (V, A)$ by

[1] The paths P_i in G will be called $\phi(P_i)$ in Lemma 2.14. This is due to technical reasons explained in Section 3.

replacing each arc $a \in A$ by a well-chosen number of parallel copies. We call such graphs H *arc-split graphs* of G, and, for every $f \in F$, we denote by $\phi(f) \in A$ the arc in G corresponding to f, i.e., f is a parallel copy of $\phi(f)$ (see Definition 3.3 for a formal definition). The idea is to choose H such that $P_1, \ldots, P_\ell$ can be chosen to be arc-disjoint in H.

More precisely, given a PSSUF instance (G, s, T, d, x), we start by constructing the following:

(i) an arc-split graph $H = (V, F)$ of G (with the associated mapping ϕ);
(ii) a partition of F into paths $P_1, \ldots, P_\ell$, each of which is a path in H from s to some terminal $t \in T$, and such that these paths are (geometrically) non-crossing, and numbered counterclockwise around s;
(iii) coefficients $\lambda_1, \ldots, \lambda_\ell \in \mathbb{Q}_{\geq 0}$ such that $\sum_{i=1}^{\ell} \lambda_i \chi^{\phi(P_i)} = x$.

See Figure 2 for an example.

We denote paths $P_1, \ldots, P_\ell$, along with their corresponding coefficients $\lambda_1, \ldots, \lambda_\ell$, which satisfy conditions (ii) and (iii), as a *source-numbered nice s-path decomposition* of H. (For a more formal definition, see Section 3.) It can be thought of as a very structured flow decomposition of x.

First, we show that such a decomposition can be obtained in polynomial time.

Theorem 2.1 *Let $(G = (V, A), s, T, d, x)$ be a PSSUF instance. We can compute in $O(|V|^2)$ time an arc-split graph H of G together with a source-numbered nice s-path decomposition $(P_i, \lambda_i)_{i \in [\ell]}$ of H with $\ell = O(|V|)$.*

Recall that in the second step of our algorithm, we will choose our PSSUF solution from the paths $P_1, \ldots, P_\ell$ (or more formally, the corresponding paths $\phi(P_1), \ldots, \phi(P_\ell)$ in G). More precisely, for each terminal $t \in T$, we will pick one path among

$$\mathcal{P}^t := \{P_i : i \in [\ell] \text{ and } P_i \text{ ends at terminal } t\}.$$

One important consequence of the paths $(P_i)_{i \in [\ell]}$ being non-crossing, is that the sets $S^t := \{i \in [\ell] : P_i \in \mathcal{P}^t\}$ for $t \in T$ are *non-interleaving*, as we will prove in Section 4.2.

Definition 2.2 *(Non-interleaving)* Two disjoint sets $S_1, S_2 \subseteq \mathbb{Z}_{>0}$ are *interleaving* if there exist $a_1, b_1 \in S_1, a_2, b_2 \in S_2$ with $a_1 < a_2 < b_1 < b_2$ or $a_2 < a_1 < b_2 < b_1$. A partition $\mathcal{S}$ of $[\ell]$ is *non-interleaving* if for any $S_1, S_2 \in \mathcal{S}$ with $S_1 \neq S_2$ the sets S_1 and S_2 are non-interleaving.

Lemma 2.3 *For $t \in T$ let $S^t := \{i \in [\ell] : P_i \in \mathcal{P}^t\}$. Then the partition $\{S^t : t \in T\}$ of $[\ell]$ is non-interleaving.*

This is a key structure we will exploit in our discrepancy-based approach to select one path in each $\mathcal{P}^t$ for each $t \in T$, which we discuss next.

2.2 Interval-discrepancy on non-interleaving partitions

We now discuss the selection of paths $P^t \in \mathcal{P}^t$ for all $t \in T$, which we formulate as a discrepancy problem.

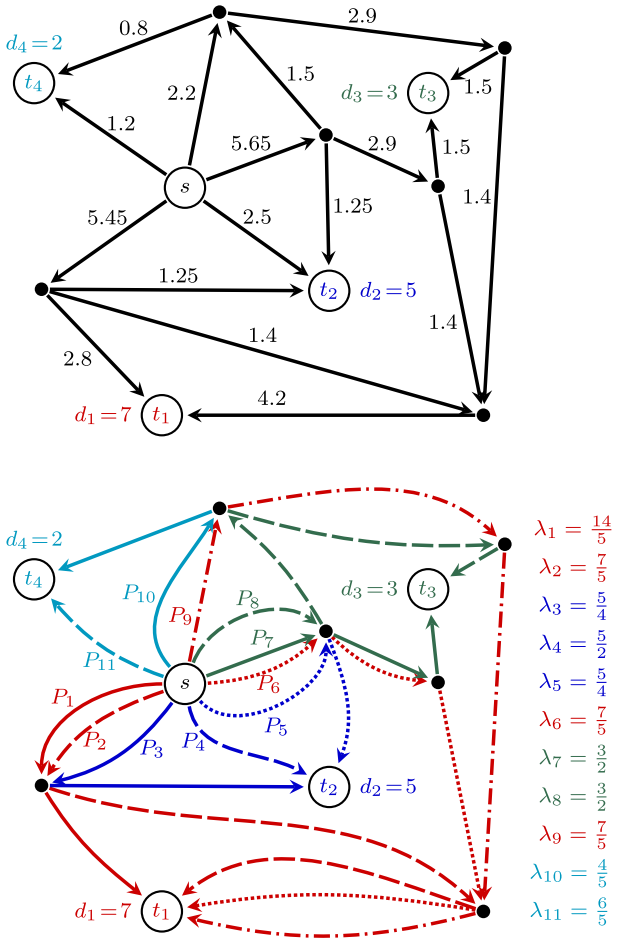


Fig. 2 The top figure shows a PSSUF instance (G, s, T, d, x) with $T = \{t_1, t_2, t_3, t_4\}$, and the x-values are shown next to the arcs. The bottom figure shows an arc-split graph H of G together with a source-numbered nice s-path decomposition $(P_i, \lambda_i)_{i\in[11]}$

Definition 2.4 *(Selections)* Let $\mathcal{S}$ be a partition of $[\ell]$. Then

- a *fractional selection* for $\mathcal{S}$ is a vector $y \in \{q \in [0,1]^\ell : q(S) = 1 \ \forall S \in \mathcal{S}\}$, and
- an *integral selection* for $\mathcal{S}$ is a vector $z \in \{q \in \{0,1\}^\ell : q(S) = 1 \ \forall S \in \mathcal{S}\}$.

We will apply this definition to the partition $\mathcal{S}$ consisting of the sets $S^t := \{i \in [\ell] : P_i \in \mathcal{P}^t\}$ with $t \in T$. The s-path decomposition $(P_i, \lambda_i)_{i\in[\ell]}$ naturally gives rise to a fractional selection y for $\mathcal{S}$ by setting $y_i := \frac{\lambda_i}{d_t}$ where $t \in T$ is the terminal where the path P_i ends. Then the vector $(\lambda_i)_{i\in[\ell]}$ will be called the *load vector*.

Definition 2.5 *(Load vector)* Given a vector $y \in [0,1]^\ell$ and a demand vector $d \in \mathbb{Q}^{\mathcal{S}}_{\geq 0}$, the *load vector* $y^d \in \mathbb{Q}^\ell_{\geq 0}$ is defined by

$$y^d_i := d_S \cdot y_i,$$

where $S \in \mathcal{S}$ is the set containing $i \in [\ell]$.

An instance of our discrepancy problem consists of the integer ℓ, the partition $\mathcal{S}$, the demand vector d, and a fractional selection y.

Definition 2.6 *(Weighted partition-constrained selection (WPCS) instance)* We call a tuple $(\ell, \mathcal{S}, d, y)$ a *weighted partition-constrained selection instance* (for short *WPCS instance*) if

- ℓ is a nonnegative integer,
- $\mathcal{S}$ is a partition of $[\ell]$,
- d is a vector in $\mathbb{Q}^{\mathcal{S}}_{\geq 0}$, and
- y is a fractional selection for $\mathcal{S}$.

We prove Theorem 1.7 by finding an integral selection z such that for each arc $a \in A$,

$$x(a) - d_{\max} \leq \sum_{i \in [\ell]:\, a \in \phi(P_i)} z_i^d \leq x(a) + d_{\max}. \tag{2.1}$$

Because $(P_i, \lambda_i)_{i \in [\ell]}$ is an s-path decomposition, we have

$$x(a) = \sum_{i \in [\ell]:\, a \in \phi(P_i)} \lambda_i = \sum_{i \in [\ell]: a \in \phi(P_i)} y_i^d,$$

and thus (2.1) is equivalent to the condition that the (y, z)-*discrepancy* of the set $\{i \in [\ell] : a \in \phi(P_i)\}$ is bounded by $d_{\max}$, where the (y, z)-discrepancy is defined as follows:

Definition 2.7 *(Discrepancy)* For a set $I \subseteq [\ell]$ and fractional selections y,z, we define the (y, z)-*discrepancy* of I as

$$D_{y,z}(I) := \left| y^d(I) - z^d(I) \right|.$$

Clearly, in general there is no integral selection z such that the (y, z)-discrepancy for every set $I \subseteq [\ell]$ is small. (For example, if y is a fractional selection with only small entries, the index set corresponding to the selected elements in any integral selection has very high discrepancy.) However, we will be able to find an integral selection z such that the (y, z)-discrepancy is at most $d_{\max}$ for all *circular intervals*.

Definition 2.8 *(Circular interval)* For $i, j \in [\ell]$, the circular interval from i to j is the set

- $\{i, i+1, \ldots, j\}$ if $i \leq j$, and
- $\{i, i+1, \ldots, \ell, 1, \ldots, j\}$ if $j < i$.

A set $I \subseteq [\ell]$ is called a *circular interval* if $I = \emptyset$ or if I is the circular interval from i to j for some $i, j \in [\ell]$.

We call the maximum (y, z)-discrepancy of any circular interval the (y, z)-*interval-discrepancy*.

Definition 2.9 *(Interval-discrepancy)* For fractional selections y and z, we define the (y, z)-interval-discrepancy as

$$D_{y,z} \coloneqq \max_{\substack{I \subseteq [\ell] \\ \text{circular interval}}} D_{y,z}(I).$$

In Section 5 we will prove the following statement.

Theorem 2.10 *Consider a WPCS instance $(\ell, \mathcal{S}, d, y)$. If $\mathcal{S}$ is non-interleaving, there exists an integral selection z with*

$$D_{y,z} \leq d_{\max},$$

where $d_{\max} \coloneqq \max_{S \in \mathcal{S}} d_S$. Moreover, it can be computed in time $O(\ell)$.

We will also prove that if we are given costs $c \in \mathbb{Q}^\ell$ and require the selection z to be no more expensive than the fractional selection y, then we can still achieve an upper bound of $2d_{\max}$ on the interval-discrepancy. This cost version of the selection problem will be used to prove Theorem 1.8.

Theorem 2.11 *Consider a WPCS instance $(\ell, \mathcal{S}, d, y)$ and let $c \in \mathbb{Q}^\ell$. If $\mathcal{S}$ is non-interleaving, there exists an integral selection z with*

$$D_{y,z} \leq 2d_{\max} \quad \text{and}$$
$$c^T z \leq c^T y,$$

where $d_{\max} \coloneqq \max_{S \in \mathcal{S}} d_S$. Moreover, it can be computed in time $O(\ell \cdot \langle \text{input} \rangle)$, where $\langle \text{input} \rangle$ refers to the length of the bit encoding of the input.

To obtain Theorem 2.10, we exploit that $\mathcal{S}$ is non-interleaving to prove that a simple greedy algorithm yields an integral selection with the desired properties. For showing Theorem 2.11, we first use a common argument in discrepancy theory (see, e.g., [23], and [4] for a recent application in a scheduling context), allowing us to reduce to the special case where each set $S \in \mathcal{S}$ has size two and the given fractional selection y fulfills $y_i = \frac{1}{2}$ for all $i \in [\ell]$; this reduction comes at the cost of losing a factor of two in our discrepancy bound. In the special case we just described, we can for each integral selection z define a complementary integral selection $\overline{z}$ that selects for each $S \in \mathcal{S}$ the element not selected by z. We observe that whenever z fulfills the desired discrepancy bounds, then so does $\overline{z}$. Moreover, one of the selections z and $\overline{z}$ is not more expensive than y. Hence, we can apply the same algorithm used to prove Theorem 2.10 to this half-integral discrepancy problem in order to find an integral selection z fulfilling the desired discrepancy bounds and then return the cheaper selection of z and $\overline{z}$. For details, see Section 5.

2.3 Bounding interval-discrepancy suffices

Recall that our goal is to find an integral selection z such that the discrepancy of the set $\{i \in [\ell] : a \in \phi(P_i)\}$ is small for every arc $a \in A$. By Theorem 2.10 and Theorem

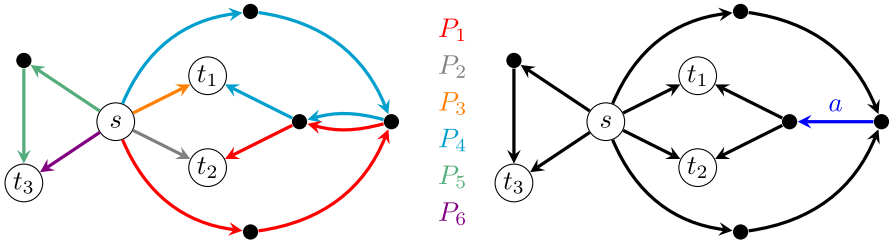


Fig. 3 The figure shows an example of an arc-split graph H (left) with a family of paths $P_1, \ldots, P_6$ from a source-numbered nice s-path decomposition. The graph G is shown on the right with an arc a highlighted in blue. In this example we have $\{i \in [\ell] \colon a \in \phi(P_i)\} = \{1, 4\}$, which is not a circular interval

2.11 we can achieve a small discrepancy for every circular interval. However, not every set $\{i \in [\ell] \colon a \in \phi(P_i)\}$ with $a \in A$ is a circular interval, as the example in Figure 3 shows.

Nevertheless, we will prove that Theorem 2.10 and Theorem 2.11 can be used to bound the discrepancy of all sets $\{i \in [\ell] \colon a \in \phi(P_i)\}$ with $a \in A$. In order to prove this, we define an equivalence relation on the set $2^{[\ell]}$ of all subsets of $[\ell]$. Recall that $S^t := \{i \in [\ell] : P_i \text{ ends at terminal } t\}$.

Definition 2.12 *(S^t-addition and S^t-removal)* Let $t \in T$. We say that a set $Y \subseteq [\ell]$ *arises from* $X \subseteq [\ell]$ *by* S^t*-addition*, if $X \cap S^t = \emptyset$ and $Y = X \cup S^t$. We say that a set $Y \subseteq [\ell]$ *arises from* $X \subseteq [\ell]$ *by* S^t*-removal*, if $S^t \subseteq X$ and $Y = X \setminus S^t$.

If a set $Y \subseteq [\ell]$ arises from $X \subseteq [\ell]$ by (potentially several) S^t-additions or S^t-removals (for some terminals $t \in T$), the (y, z)-discrepancy of X and Y is identical for every integral selection z. For this reason we will then say that X and Y are *discrepancy-equivalent*.

Definition 2.13 *(Discrepancy-equivalent)* We call two sets $X, Y \subseteq [\ell]$ *discrepancy-equivalent* if Y arises from X by a sequence of S^t-additions and S^t-removals for some terminals $t \in T$.

Note that this indeed defines an equivalence relation. We will prove in Section 4.1 the statement below, which says that the set of indices of all paths in $\mathcal{P}$ containing a copy of the same given arc $a \in A$ are discrepancy-equivalent to a circular interval. For example, the indices of the paths in Figure 3 that go over (a copy of) arc a, which are 1 and 4, are discrepancy-equivalent to the circular interval $\{4, 5, 6, 1\}$, which is obtained from $\{1, 4\}$ by S^{t_3}-addition.

Lemma 2.14 *For every arc $a \in A$, the set $\{i \in [\ell] \colon a \in \phi(P_i)\}$ is discrepancy-equivalent to a circular interval.*

The proof of this statement (in Section 4) crucially exploits that $P_1, \ldots, P_\ell$ are paths from a source-numbered nice s-path decomposition of the arc-split graph H. Using Lemma 2.3 and Lemma 2.14, we can now prove that our discrepancy statements (Theorem 2.10 and Theorem 2.11) imply our main results (Theorem 1.7 and Theorem 1.8).

Proof of Theorem 1.7 and Theorem 1.8 Let $(G = (V, A), s, T, d, x)$ be a PSSUF instance. By Theorem 3.7 we can compute in $O(|V|^2)$ time an arc-split (H, π, ϕ) of G and a nice s-path decomposition $(P_i, \lambda_i)_{i\in[\ell]}$ of H with $\ell = O(|V|)$. We define a WPCS instance $(\ell, \mathcal{S}, d, y)$ as follows. For $t \in T$ we let $S^t := \{i \in [\ell]\colon P_i \text{ is an } s-t \text{ path}\}$ and we define $\mathcal{S} := \{S^t : t \in T\}$. Because G is acyclic and $\sum_{i=1}^{\ell} \lambda_i \chi^{\phi(P_i)} = x$, every path P_i with $i \in [\ell]$ starts at s and ends at some terminal $t \in T$. Therefore, $\mathcal{S}$ is a partition of $[\ell]$. We define the demand vector $d \in \mathbb{Q}_{\geq 0}^{\mathcal{S}}$ by $d_{S^t} := d_t$. For $i \in [\ell]$ we define $y_i := \frac{\lambda_i}{d_t}$, where $t \in T$ is the terminal where the path P_i ends. Because $\sum_{i=1}^{\ell} \lambda_i \chi^{\phi(P_i)} = x$ and $x(\delta^-(t)) - x(\delta^+(t)) = d_t$ for all $t \in T$, we have $\sum_{i\in S^t} \lambda_i = d_t$ and thus $\sum_{i\in S^t} y_i = 1$ for all $t \in T$. This shows that y is a fractional selection for $\mathcal{S}$. By Lemma 2.3, the partition $\mathcal{S}$ is non-interleaving.

Now consider an integral selection z for $\mathcal{S}$. For each $t \in T$, we select the s-t path $P^t := P_i$, where i is the unique element of S^t with $z_i = 1$. Consider the unsplittable flow $\mathcal{P} = \{P^t\}_{t\in T}$ and an arc $a \in A$. By Lemma 2.14, the set $\{i \in [\ell] : a \in \phi(P_i)\}$ is discrepancy-equivalent to some circular interval I_a, implying $D_{y,z}(\{i \in [\ell] : a \in \phi(P_i)\}) = D_{y,z}(I_a)$. Using

$$x(a) = \sum_{i\in[\ell]\colon a\in\phi(P_i)} \lambda_i = \sum_{i\in[\ell]:a\in\phi(P_i)} y_i^d,$$

and

$$\text{flow}_{\mathcal{P}}(a) = \sum_{i\in[\ell]\colon a\in\phi(P_i)} z_i^d,$$

this implies

$$\begin{aligned} |x(a) - \text{flow}_{\mathcal{P}}(a)| &= \left| \sum_{i\in[\ell]\colon a\in\phi(P_i)} y_i^d - \sum_{i\in[\ell]:a\in\phi(P_i)} z_i^d \right| \\ &= D_{y,z}(\{i \in [\ell]\colon a \in \phi(P_i)\}) = D_{y,z}(I_a) \leq D_{y,z}. \end{aligned}$$

We conclude that Theorem 1.7 follows from the discrepancy statement Theorem 2.10 applied to the WPCS instance $(\ell, \mathcal{S}, d, y)$. As an overall runtime we obtain $O(|V|^2)$. Moreover, Theorem 1.8 follows from Theorem 2.11 applied to the WPCS instance $(\ell, \mathcal{S}, d, y)$ and the cost vector $c \in \mathbb{Q}_{\geq 0}^{\ell}$ defined by $c_i := \sum_{a\in P_i} c(a) \cdot d_t$, where t is the terminal of path P_i. As an overall runtime we obtain $O(|V|^2 + \langle\text{input}\rangle \cdot |V|) = O(\langle\text{input}\rangle \cdot |V|)$. □

The remainder of this paper is structured as follows. In Section 3, we show how to obtain an arc-split graph H with a nice s-path decomposition and prove Theorem 3.7. In Section 4, we prove that a nice s-path decomposition has the properties claimed in Lemma 2.3 and Lemma 2.14. Section 5 contains the proof of the two discrepancy statements Theorem 2.10 and Theorem 2.11. Section 6 extends our techniques to arbitrary (not necessarily planar) graphs, obtaining capacity violations parameterized by the genus of the graph. Section 7 contains some concluding remarks. Finally,

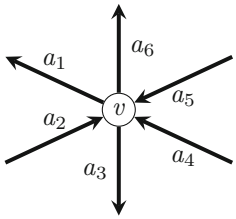


Fig. 4 A vertex v with incident arcs $a_1, \dots a_6$, geometrically embedded into the plane. Let π be the induced combinatorial embedding. In the cyclic ordering π_v, the arc a_6 is the predecessor of a_1 and the arc a_2 is the successor of a_1.
The tuple (a_5, a_2, a_3) is a π_v-progression because a_5, a_2, a_3 is a subsequence of $a_5, a_6, a_1, a_2, a_3, a_4$. The tuple (a_5, a_3, a_2) is not a π_v-progression

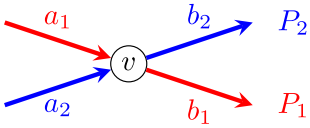


Fig. 5 P_1 and P_2 are crossing because (a_1, a_2, b_1, b_2) is a π_v-progression

Appendix A.1 discusses the necessity of the acyclicity assumption when dealing with lower bounds.

3 Computing a nice path decomposition

We now provide details on how we compute a source-numbered nice path decomposition of a flow x, which proves Theorem 2.1. Moreover, we expand on the discussion of Section 2.1 and further formalize some of the concepts introduced there.

For a vertex $v \in V$, let $\delta(v) := \delta^+(v) \cup \delta^-(v)$ be the set of all arcs incident to v. Each planar geometric embedding induces a cyclic ordering π_v of the arcs in $\delta(v)$, obtained by traversing the arcs in $\delta(v)$ in counterclockwise sense. We say that $b \in \delta(v)$ is the *successor* of $a \in \delta(v)$ in π_v if, in the geometric embedding, b is the next arc after a in counterclockwise direction. In this case, we also say that a is the *predecessor* of b in π_v.

A collection $\pi = (\pi_v)_{v \in V}$ of cyclic orderings of the arcs incident to v for all $v \in V$ is called a *combinatorial embedding*. A combinatorial embedding is *planar* if it is induced by some planar geometric embedding. See Figure 4 for an example.

In order to formally define when two paths are crossing, we introduce the notion of a π_v-progression. Figure 4 shows an example.

Definition 3.1 *(π_v-progression)* Let $v \in V$ and let $b_1, \dots, b_k \in \delta(v)$ be k distinct arcs. Let $\delta(v) = \{a_1, \dots, a_m\}$, where the numbering is chosen such that $a_1 = b_1$ and a_{i+1} is the successor of a_i in the cyclic order π_v for all $i \in [m-1]$. Then $(b_1, b_2, \dots, b_k)$ is a *π_v-progression* if $b_1, b_2, \dots, b_k$ is a subsequence of $a_1, \dots, a_m$.

We are now ready to formally define when two edge-disjoint paths are crossing. See Figure 5.

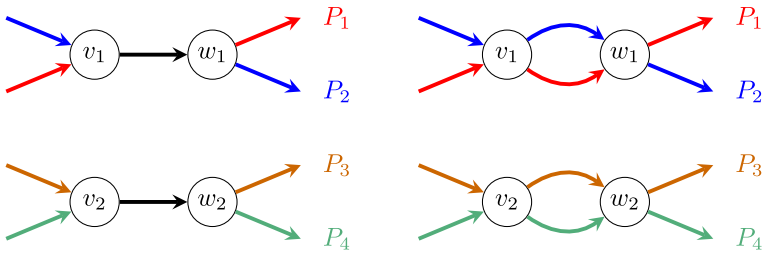


Fig. 6 The left-hand side shows paths P_1 and P_2 that use a common arc (v_1, w_1) and paths P_3 and P_4 that use a common arc (v_2, w_2). We would like to consider P_1 and P_2 as crossing, but not P_3 and P_4. However, when considering the local cyclic arc-orderings at vertices, we cannot distinguish these two cases. This is why we introduce arc-split graphs, where (v_i, w_i) is replaced by two copies, each used by one path, as depicted on the right-hand side. This leads to arc-disjoint paths P_1 and P_2 that are crossing according to Definition 3.2. (In the example, they cross at w_1, and if we swap the red and blue copy of the arc (v_1, w_1), they cross at v_1.) The paths P_3 and P_4 are not crossing

Definition 3.2 *(Crossing paths)* Let $H = (V, F)$ be a graph with planar combinatorial embedding π. Two arc-disjoint paths P_1, P_2 are *crossing* if they have a common mutually internal vertex $v \in V$ such that (a_1, a_2, b_1, b_2) or (a_2, a_1, b_2, b_1) is a π_v-progression, where $\{a_1\} = P_1 \cap \delta^-(v)$, $\{b_1\} = P_1 \cap \delta^+(v)$, $\{a_2\} = P_2 \cap \delta^-(v)$, and $\{b_2\} = P_2 \cap \delta^+(v)$.

This notion works well for arc-disjoint paths but fails to capture what would naturally be considered crossing paths if the paths have arcs in common, as illustrated in Figure 6. For this reason, we work with the arc-split graph H of G, where arcs are replaced by (a well-chosen arc-dependent number of) parallel copies. This allows for assuming that the paths $P_1, \ldots, P_\ell$ we construct later are disjoint in H.

Arc-split graphs are formally defined as follows, where for some arc $a = (v, w)$, we write $\text{head}(a) = w$ and $\text{tail}(a) = v$.

Definition 3.3 *(Arc-split graph H of G)* Let $G = (V, A)$ be a planar graph. An *arc-split* graph of G is a tuple (H, π), where $H = (V, F)$ is a planar graph with planar combinatorial embedding π, such that there exists a mapping $\phi\colon F \to A$ fulfilling for any $f \in F$:

$$\text{head}(f) = \text{head}(\phi(f)) \quad \text{and} \quad \text{tail}(f) = \text{tail}(\phi(f)),$$

and, for all $a \in A$, the set $\phi^{-1}(a) \subseteq F$ is consecutive at both $\text{head}(a)$ and $\text{tail}(a)$ in π. We call (H, π, ϕ) an *arc-split* of G.

As mentioned above, we want the paths $P_1, \ldots, P_\ell$ to be arc-disjoint in the arc-split graph H. The number of copies of an arc $a \in A$ in the arc-split graph H that we construct will be chosen such that each copy is used by precisely one path P_i. Thus, the paths $P_1, \ldots, P_\ell$ that we construct will form a *nice s-path partitioning* of H, which is defined as follows.

Definition 3.4 *(Nice s-path partitioning)* Let $H = (V, F)$ be a graph with planar combinatorial embedding π, and let $s \in V$. A collection of paths $P_1, \ldots, P_\ell \subseteq F$ in H is called an *s-path partitioning* of H if

- P_i starts at s for all $i \in [\ell]$, and
- $F = P_1 \dot{\cup} \ldots \dot{\cup} P_\ell$.

It is called *nice* if the paths $P_1, \ldots, P_\ell$ are pairwise non-crossing.

Together with the nice s-path partitioning of H we will also compute non-negative weights for the paths P_i in order to obtain a path decomposition of the flow x.

Definition 3.5 *(Nice s-path decomposition)* Let (G, s, T, d, x) be a PSSUF instance and let (H, π, ϕ) be an arc-split of $G = (V, A)$. A *nice s-path decomposition* of H is a sequence of tuples $(P_i, \lambda_i)_{i \in [\ell]}$, where $P_1, \ldots, P_\ell$ is a nice s-path partitioning of H and $\lambda_i \in \mathbb{Q}_{\geq 0}$ for $i \in [\ell]$, such that

$$\sum_{i=1}^{\ell} \lambda_i \chi^{\phi(P_i)} = x.^2$$

Recall that all paths $P_1, \ldots, P_\ell$ start at s. Thus, the cyclic ordering π_s of the arcs incident to s naturally induces a cyclic ordering on the paths $P_1, \ldots, P_\ell$. As mentioned in Section 2, if the numbering of these paths is consistent with this cyclic ordering, we say that $P_1, \ldots, P_\ell$ are *source-numbered*.

Definition 3.6 *(Source-numbered)* An s-path partitioning is called *source-numbered* if $(a_1, \ldots, a_\ell)$ is a π_s-progression, where a_i is the first arc of P_i for $i \in [\ell]$.

We now prove Theorem 2.1, which we restate here using the just introduced notation, making explicit that we compute a combinatorial embedding of H.

Theorem 3.7 *Let $(G = (V, A), s, T, d, x)$ be a PSSUF instance. We can compute in $O(|V|^2)$ time an arc-split (H, π, ϕ) of G and a source-numbered nice s-path decomposition $(P_i, \lambda_i)_{i \in [\ell]}$ of H with $\ell = O(|V|)$.*

3.1 Proof of Theorem 3.7

Note that source-numbering a set of paths starting at s simply means that the paths are numbered in counterclockwise sense around s (starting at an arbitrary path). As such a numbering can trivially be achieved for any nice s-path decomposition, we ignore the source-numbering aspect when constructing a source-numbered nice s-path decomposition fulfilling the conditions of Theorem 3.7. Hence, we focus in what follows on finding a nice s-path decomposition.

There are different ways to obtain nice s-path decompositions. One option is to start with an arbitrary flow decomposition and then iteratively reduce the number of crossings by uncrossing steps. We present another approach that avoids crossings upfront and leads to a running time that is linear in the total size $\sum_{i=1}^{\ell} |P_i|$ of the computed paths.

To obtain the desired nice s-path decomposition in an arc-split graph H of G, as claimed by Theorem 3.7, we first decide locally on rules how paths can traverse each vertex $v \in V \setminus (T \cup \{s\})$. These rules are set up such that paths that we later construct from them will be non-crossing. We formalize these rules through the notion of a

good v-wiring. A good v-wiring is a sequence of pairs, each coupling one v-incoming arc $f \in \delta^-(v)$ with one v-outgoing arc $g \in \delta^+(v)$, together with a maximum load $\mu_v((f, g))$ with which we can use this pair. The paths in our nice s-path decomposition that go over v will use one of these pairs, and the load with which each pair $(f, g) \in \delta^-(v) \times \delta^+(v)$ is used is given by $\mu_v((f, g))$.

Definition 3.8 *(Good v-wiring)* Let $(G = (V, A), s, T, d, x)$ be a PSSUF instance, and let $v \in V \setminus (T \cup \{s\})$. Moreover, let π^G be a planar combinatorial embedding of G. A *good v-wiring* is a tuple (W_v, μ_v), where $W_v = ((f_1, g_1), \ldots, (f_k, g_k))$ is a sequence of distinct arc pairs $(f_i, g_i) \in \delta^-(v) \times \delta^+(v)$ for $i \in [k]$, and $\mu_v \colon W_v \to \mathbb{Q}_{\geq 0}$, such that

$$\sum_{\substack{i \in [k]: \\ a \in \{f_i, g_i\}}} \mu_v((f_i, g_i)) = x(a) \qquad \forall a \in \delta(v),$$

and, for $i, j \in [k]$ with $i < j$, we have that neither (f_i, f_j, g_i) nor (f_i, g_j, g_i) is a π_v^G-progression.

Note that forbidding, for $i, j \in [k]$ with $i < j$, the π_v^G-progressions (f_i, f_j, g_i) and (f_i, g_j, g_i), excludes the π_v^G-progressions (f_i, f_j, g_i, g_j) and (f_j, f_i, g_j, g_i).

We start by showing that a good v-wiring can be computed fast.

Lemma 3.9 *Let $(G = (V, A), s, T, d, x)$ be a PSSUF instance with planar combinatorial embedding π^G, and let $v \in V \setminus (T \cup \{s\})$. We can find in $O(\deg(v))$ time a good v-wiring (W_v, μ_v) with $|W_v| \leq \deg(v)$.*

Proof We first describe a simple procedure to obtain the desired result, which, when implemented in a straightforward way, only runs in polynomial time instead of the desired $O(\deg(v))$ running time bound. In a second step, we expand on how to refine this procedure to obtain the claimed linear time bound.

We construct a good v-wiring by iteratively finding a pair of arcs $a \in \delta^-(v)$, $b \in \delta^+(v)$ with b being a's successor in π_v^G, adding the pair (a, b) to W_v with a μ_v-value equal to $\min\{x(a), x(b)\}$, reducing the x-values of a and b correspondingly, and deleting a and/or b if their x-value reached 0. Algorithm 1 formalizes this procedure. In its implementation, we use a circular doubly linked list L of the arcs in $\delta(v)$. This list contains all arcs in $\delta(v)$ in the order given by π_v^G. For an arc $a \in L$, we denote by $\mathrm{succ}_L(a)$ the arc in L that comes after a with respect to the circular ordering π_v^G, i.e., the one after a in L in counterclockwise order.

Note that we have $y(\delta^+(v)) = y(\delta^-(v))$ during the algorithm. At the start, this holds because $x(\delta^+(v)) = x(\delta^-(v))$, and the property is preserved because, whenever a pair gets added to W_v, we remove the same value from one incoming and one outgoing arc of v. This implies that in every iteration of the algorithm, there is indeed a pair of arcs (a, b) as desired.

Also observe that this algorithm can clearly be implemented to have polynomial running time. Indeed, at every iteration of the while loop, at least one of the arcs a or b will be removed from L, leading to at most $O(\deg(v))$ many iterations. Moreover,

Algorithm 1: Simple algorithm to create good v-wiring.

```
1  Let y = x.
2  Let L be a circular doubly linked list of δ(v).
3  W_v = ().
4  while L ≠ ∅ do
5      Find an arc a ∈ L ∩ δ⁻(v) such that b = succ_L(a) ∈ δ⁺(v).
6      Add (a, b) to W_v.
7      Set μ_v((a, b)) = min{y(a), y(b)}.
8      y(a) = y(a) − μ_v((a, b)).
9      y(b) = y(b) − μ_v((a, b)).
10     if y(a) = 0 then delete a from L. if y(b) = 0 then delete b from L.
11 return (W_v, μ_v).
```

the bottleneck operation in one iteration is finding an arc a in line 5. A trivial implementation of this step, where we simply check all arcs in L, leads to a running time of $O(\deg(v)^2)$.

Before expanding on how to obtain a faster implementation, we show that (W_v, μ_v) is a good v-wiring. Let $W_v = ((f_1, g_1), \ldots, (f_k, g_k))$. Note that we clearly have

$$\sum_{\substack{i \in [k]: \\ a \in \{f_i, g_i\}}} \mu_v((f_i, g_i)) = x(a) \qquad \forall a \in \delta(v),$$

because the procedure stops once all y-values, which are the leftover x-values, are set to zero.

Now consider $i, j \in [k]$ with $i < j$, and we observe that between f_i and g_i, in counterclockwise sense, there is neither the arc f_j nor g_j. This holds because, when Algorithm 1 adds the pair (f_i, g_i), then the arc g_i appears right after f_i in the circular doubly linked list L, i.e., $g_i = \text{succ}_L(f_i)$. As neither f_j nor g_j have been deleted at this point because $i < j$, they do not lie between f_i and g_i in counterclockwise sense. This shows that Algorithm 1 returns a good v-wiring as claimed.

It remains to show that it can be sped up to run in $O(\deg(v))$ time. To this end, we show how, after an initial $O(\deg(v))$-time preprocessing step, one can find in each iteration a pair $(a, b) \in \delta^-(v) \times \delta^+(v)$ of arcs in L with $b = \text{succ}_L(a)$ to add to the v-wiring in constant time. To this end, we maintain throughout the iterations of the while loop of Algorithm 1 the set $Z \subseteq L \cap \delta^-(v)$ of all arcs $a \in L \cap \delta^-(v)$ with $\text{succ}_L(a) \in \delta^+(v)$. The set Z thus contains the arcs a of all pairs (a, b) that could be chosen in line 5. Before the start of the while loop in Algorithm 1, we initialize Z in $O(\deg(v))$ time by exhaustively going through all arcs in $\delta(v)$. Then, at the end of each iteration of the while loop, we update Z in constant time as follows. If the arc a is deleted in line 10 of Algorithm 1, we remove it from Z and check whether its predecessor in L should be added to Z. (This case also includes the situation where a and $\text{succ}_L(a)$ are both deleted.) Otherwise, $\text{succ}_L(a)$ will get deleted in line 10, and we check whether, after deleting $\text{succ}_L(a)$, the arc a still fulfills that its successor in L is a v-outgoing arc, i.e., is in $\delta^+(v)$. If so, we keep a in Z; otherwise, we remove it from Z. One can easily verify that this procedure correctly maintains Z. Hence, in

each iteration of the while loop, we can find a pair $(a, b) \in Z$ in constant time, and the total time spent on initializing and updating Z is $O(\deg(v))$, as desired. (These time bounds are for example achieved when implementing Z as a stack.) □

Starting with good v-wirings, we now show how to compute an arc-split of G together with a nice s-path decomposition as claimed in Theorem 3.7.

Proof of Theorem 3.7 We start by obtaining a planar combinatorial embedding π^G of G, which can be done in linear time, and compute a structured s-path decomposition of x as described in Algorithm 2. Note that Algorithm 2 requires a planar combinatorial embedding π^G of G as it first computes a good v-wiring (W_v, μ_v) for each vertex $v \in V \setminus (T \cup \{s\})$. Moreover, also line 6 relies on π_s^G. To perform line 6, we interpret π_s^G as a linear order on $\delta^+(s)$ instead of a cyclic one by simply choosing an arbitrary arc $a \in \delta^+(s)$, declaring a to be the first arc, and then numbering the remaining arcs with respect to π_s^G, i.e., in counterclockwise order.

Algorithm 2 greedily performs a path-decomposition of the flow x. The algorithm builds paths step-by-step according to these v-wirings; whenever we enter a vertex v through some arc f, we look for the first pair (f, g) in the v-wiring that still has a strictly positive μ_v-value and use the arc g to leave v. To simplify notation, we drop the index v from μ_v in Algorithm 2. This is without risk of ambiguity, because each pair (f, g) of consecutive links that we consider is contained in precisely one W_v, namely the one for $v = \text{head}(f) = \text{tail}(g)$; hence, in this case $\mu((f, g)) \coloneqq \mu_v((f, g))$.

Algorithm 2: Compute structured flow decomposition of x.

```
1  Compute a good v-wiring (W_v, μ_v) for each v ∈ V \ (T ∪ {s}).
2  y = x.
3  ℓ = 0.
4  while y(δ⁺(s)) > 0 do
5      ℓ = ℓ + 1.
6      Let a ∈ δ⁺(s) be the first arc with respect to π_s^G in δ⁺(s) with y(a) > 0.
7      Q_ℓ = {a}.
8      while head(a) ∉ T do
9          Let (f, g) be the first pair in W_v with f = a and μ((f, g)) > 0, where v = head(a).;
           Q_ℓ = Q_ℓ ∪ {g}.
10         a = g.
11     Let λ_ℓ be the smallest value μ((f, g)) for any pair (f, g) of consecutive arcs in Q_ℓ.
12     For every pair (f, g) of consecutive arcs in Q_ℓ, set μ((f, g)) = μ((f, g)) − λ_ℓ.
13     y(a) = y(a) − λ_ℓ    ∀a ∈ Q_ℓ.
14 return (Q_i, λ_i)_{i∈[ℓ]}.
```

Given the family of pairs $(Q_i, \lambda_i)_{i \in [\ell]}$ obtained from Algorithm 2, we then construct the claimed arc-split (H, π, ϕ) of G together with a nice s-path decomposition as follows. Each tuple (Q_i, λ_i), for $i \in [\ell]$, will correspond to one path P_i in our s-path decomposition with coefficient being λ_i, i.e., the nice s-path decomposition we construct will be $(P_i, \lambda_i)_{i \in [\ell]}$. We start with an empty graph H on the vertices V and then add, one-by-one for i starting from 1 to ℓ, the paths P_i. When considering an index $i \in [\ell]$, we add, for each arc $a = (u, v) \in Q_i$, a fresh arc h from u to v to

the graph H and set $\phi(h) = a$. The path P_i consists of all the arcs added to H when considering the index i. Hence, P_i is a path traversing the same vertices as Q_i (in the same order), simply on a fresh set of arcs. To conclude the construction, we have to specify how to define the planar combinatorial embedding π of H. The idea is that we largely inherit the planar combinatorial embedding π^G of G. More precisely, the copy of the arc $(u, v) \in Q_i$ that we added to H will appear in π_u and π_v at the same spot as it appears in π_u^G and π_v^G, respectively. In other words, we define the planar combinatorial embedding such that, for any vertex $u \in V$, any subset of non-parallel arcs in H incident to u appear in π_u in the precise same order as the order they have in π_u^G. It remains to decide how to order different copies in H of the same arc (u, v) of G. Hence, assume that h_1, h_2 are two arcs in H, both going from $u \in V$ to $v \in V$, and assume that we added h_1 in an earlier iteration than h_2. We apply the following rule:

- Within all arcs from u to v in H, the arc h_1 comes before h_2 in π_u, i.e., the arc h_2 is on the counterclockwise side of h_1 within all arcs from u to v in H,
- Within all arcs from u to v in H, the arc h_1 comes after h_2 in π_v, i.e., the arc h_2 is on the clockwise side of h_1 within all arcs from u to v in H.

This finishes the procedure to compute the arc-split (H, π, ϕ) and the s-path decomposition $(P_i, \lambda_i)_{i \in [\ell]}$.

We start by showing the desired bound of $\ell = O(|V|)$ on the number of paths in our s-path decomposition. Note that this number of paths is the same as the number of paths Q_i, computed in Algorithm 2. Each time we compute a path Q_i in Algorithm 2, one of the values $\mu_v((f, g))$ is set to zero for some pair (f, g) in one of the sets W_v for some $v \in V \setminus (T \cup \{s\})$. We thus have

$$\ell \leq \sum_{v \in V \setminus (T \cup \{s\})} |W_v| = O\left(\sum_{v \in V \setminus (T \cup \{s\})} \deg(v) \right) = O(|A|) = O(|V|),$$

where the first equality follows from $|W_v| \leq \deg(v)$, which holds by Lemma 3.9, the second equality is due to the fact that the sum of all degrees of a graph is twice the number of its arcs, and the last inequality holds because G is planar.

We now discuss the running time of the suggested algorithm. First, note that Algorithm 2 can clearly be implemented to run in $O(|V|^2)$ time. More precisely, due to Lemma 3.9, computing the good v-wirings at the beginning of the algorithm takes $O(\sum_{v \in V \setminus (T \cup \{s\})} \deg(v)) = O(|A|) = O(|V|)$ time. Each iteration of the outer while loop constructs one path Q_i, and as we construct $\ell = O(|V|)$ many paths, this while loop has $O(|V|)$ iterations. Moreover, the inner while loop successively adds arcs to the path Q_i, each such step taking $O(1)$ time. Because $|Q_i| = O(|V|)$ for each path Q_i, this leads to a total running time of $O(|V|^2)$ for Algorithm 2. Finally, the construction of the arc-split (H, π, ϕ) and nice s-path decomposition $(P_i, \lambda_i)_{i \in [\ell]}$ takes time linear in the size of H, which is bounded by $O(\sum_{i=1}^{\ell} |Q_i|) = O(|V|^2)$ as desired, where the equality follows from $\ell = O(|V|)$ and $|Q_i| \leq |V|$ for $i \in [\ell]$.

It remains to show that the arc-split (H, π, ϕ) and the s-path decomposition $(P_i, \lambda_i)_{i \in [\ell]}$ have the desired properties. First, note that π is indeed a planar combi-

natorial embedding of H. This readily follows from the fact that a (geometric) planar embedding of H corresponding to π can be obtained by first obtaining a (geometric) planar embedding of G that corresponds to π^G and then replacing arcs of G by parallel arcs. Also, the properties of ϕ for (H, π, ϕ) to be an arc-split of G (see Definition 3.3) are clearly fulfilled. Hence, (H, π, ϕ) is an arc-split of G.

Because the arcs of H are by construction the disjoint union of the paths P_i for $i \in [\ell]$, and all these paths start at s, we have that $P_1, \ldots, P_\ell$ is an s-path partitioning of H. To show that it is a nice s-path partitioning, we have to show that they are pairwise non-crossing. Hence, let $i, j \in [\ell]$ with $i < j$ and let $v \in V$ be a common internal vertex of P_i and P_j. We name the arcs of P_i and P_j that are incident to v as follows:

$$\begin{aligned} \{a_i^H\} &= P_i \cap \delta^-(v) & a_i^G &= \phi(a_i^H) \\ \{b_i^H\} &= P_i \cap \delta^+(v) & b_i^G &= \phi(b_i^H) \\ \{a_j^H\} &= P_j \cap \delta^-(v) & a_j^G &= \phi(a_j^H) \\ \{b_j^H\} &= P_j \cap \delta^+(v) & b_j^G &= \phi(b_j^H). \end{aligned}$$

Note that we may have $a_i^G = a_j^G$, or $b_i^G = b_j^G$, or both. We need to show that neither $(a_i^H, a_j^H, b_i^H, b_j^H)$ nor $(a_j^H, a_i^H, b_j^H, b_i^H)$ is a π_v-progression. To this end, we exploit that, by construction of π_v, we have the following property:

> For any subset $U \subseteq \delta_H(v)$ of non-parallel arcs incident to v, the ordering of U
> with respect to π_v is the same as the ordering of $\phi(U)$ with respect to π_v^G. (3.1)

We first observe that $(a_i^H, a_j^H, b_i^H, b_j^H)$ is not a π_v-progression. If $a_i^G \neq a_j^G$, then (a_i^G, a_j^G, b_i^G) is not a π_v^G-progression, which follows from (W_v, μ_v) being a good v-wiring. By (3.1), this implies that (a_i^H, a_j^H, b_i^H) is not a π_v-progression, and therefore neither is $(a_i^H, a_j^H, b_i^H, b_j^H)$. Otherwise, if $a_i^G = a_j^G$, then, in the ordering π_v of the arcs $\phi^{-1}(a_i^G)$, the arc a_i^H appears after a_j^H. Again, this implies that $(a_i^H, a_j^H, b_i^H, b_j^H)$ is not a π_v-progression as $b_i^H, b_j^H \notin \phi^{-1}(a_i^G)$.

Analogously, we now show that $(a_j^H, a_i^H, b_j^H, b_i^H)$ is not a π_v-progression. If $b_i^G \neq b_j^G$, then (a_i^G, b_j^G, b_i^G) is not a π_v^G-progression, which follows from (W_v, μ_v) being a good v-wiring. By (3.1), this implies that (a_i^H, b_j^H, b_i^H) is not a π_v-progression, and therefore neither is $(a_j^H, a_i^H, b_j^H, b_i^H)$. Otherwise, if $b_i^G = b_j^G$, then, in the ordering π_v of the arcs $\phi^{-1}(b_i^G)$, the arc b_i^H appears before b_j^H. Again, this implies that $(a_j^H, a_i^H, b_j^H, b_i^H)$ is not a π_v-progression.

Thus, $P_1, \ldots, P_\ell$ are pairwise non-crossing, which shows that $P_1, \ldots, P_\ell$ is a nice s-path partitioning. To finish the proof, it remains to show that

$$\sum_{i=1}^{\ell} \lambda_i \chi^{\phi(P_i)} = x. \tag{3.2}$$

This holds because Algorithm 2 performs greedily an s-path decomposition that uses up all the μ_v-values of the good v-wirings (W_v, μ_v) for each $v \in V \setminus (T \cup \{s\})$. Indeed, this path decomposition is just one way to perform a greedy path decomposition of the original flow x. (Recall that a good v-wiring (W_v, μ_v) with $W_v = ((f_1, g_1), \ldots, (f_k, g_k))$ satisfies $\sum_{i \in [k]: a \in \{f_i, g_i\}} \mu_v((f_i, g_i)) = x(a)$ for all $a \in \delta(v)$, and thus the μ_v-values indeed represent the original flow.) Moreover, a greedy path decomposition of a flow in an acyclic graph always decomposes the full flow value on all arcs. This implies Equation 3.2. □

4 Nice path partitionings have nice properties

In this section, we prove Lemma 2.3 and Lemma 2.14, which are both statements about source-numbered nice s-path partitionings stemming from a PSSUF instance (G, s, T, d, x). As these statements do not depend on the fractional flow x or the demands d, we will only fix, throughout this section, an acyclic graph $G = (V, A)$ and an arc-split $(H = (V, F), \pi, \phi)$ of G together with a source-numbered nice s-path partitioning $P_1, \ldots, P_\ell$ of H. Moreover, we assume throughout that no end vertex of any path of the s-path partitioning has outgoing arcs, because these end vertices correspond to terminals, which do not have outgoing arcs by assumption. We thus also call such end vertices *terminals* and denote the set of all such terminals by T.

Our proofs of the statements Lemma 2.3 and Lemma 2.14 both rely on a crucial property how the paths $P_1, \ldots, P_\ell$ can interact. This property is described in terms of how two paths that share a common internal vertex divide the planar embedding into two parts. We represent these two parts as a tri-coloring, which we introduce first. Using this tri-coloring, we first prove Lemma 2.14 in Section 4.1, and then Lemma 2.3 in Section 4.2.

Tri-coloring

Let us fix a planar geometric embedding of H realizing the combinatorial embedding π. Let $\widetilde{P}$ and $\overline{P}$ be two distinct paths of the nice s-path partitioning that share a vertex $v \in V \setminus \{s\}$. Moreover, let C be the subgraph of H obtained from taking the union of the s-v subpaths of $\widetilde{P}$ and $\overline{P}$. Observe that C is a connected graph in which every vertex v has even degree $|\delta(v)|$. Thus, the planar dual C^* of the undirected graph obtained from C by ignoring the orientation of the arcs is bipartite. [3] Therefore, we can color the faces of C with two colors, red and blue. Because the geometric embedding is planar, this yields a red-blue coloring of all arcs of H except for C. Finally, we

[3] This is well known and can be seen as follows. In the undirected graph obtained from C by ignoring the orientation of the arcs, all vertex degrees are even and hence every cut contains an even number of edges. As cycles in the dual graph C^* correspond to cuts in this graph, C^* has no odd cycle and is thus bipartite.

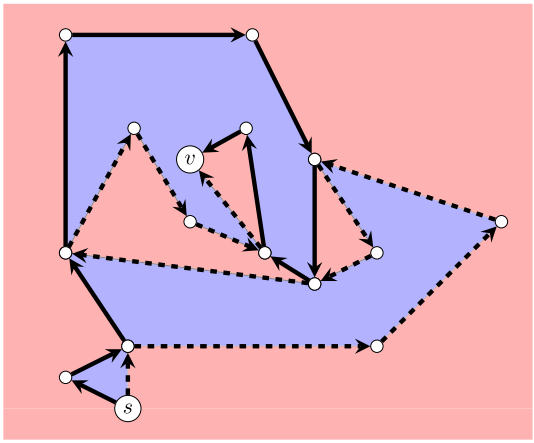


Fig. 7 Example of the coloring of the plane. The s-v subpaths of $\overline{P}$ and $\widetilde{P}$ are shown in black with solid and dashed arcs, respectively. The color assigned to the arcs in $\overline{P} \cup \widetilde{P}$ is black, arcs lying within a red face are colored red, and arcs lying within a blue face are colored blue

color the arcs in C in black. We call this coloring of all arcs of H the *tri-coloring of H induced by $\widetilde{P}$, $\overline{P}$, and v*. (It is unique up to exchanging red and blue.) See Figure 7. When we say that a path changes color, we mean that it changes the color of its arcs from either red, blue, or black, to another one of these 3 options.

Note that no path of the nice s-path partitioning except for $\widetilde{P}$ and $\overline{P}$ contains black arcs, because paths in $P_1, \ldots, P_\ell$ use distinct arcs. The paths $\widetilde{P}$ and $\overline{P}$ use black arcs until they reach v, and red or blue ones afterward.

The following lemma highlights a crucial property how different paths of a nice s-partitioning can interact, based on the above-introduced tri-coloring.

Lemma 4.1 *Let $\widetilde{P}$ and $\overline{P}$ be two distinct paths of the nice s-path partitioning that share a vertex $v \in V \setminus \{s\}$ and consider the tri-coloring of H induced by $\widetilde{P}$, $\overline{P}$, and v. Then any path P of the nice s-path partitioning can change its color at most once and, if so, only at vertex v.*

Proof As in the definition of a tri-coloring, let C be the set of all arcs colored black, i.e., let C be the union of the s-v subpaths of $\overline{P}$ and $\widetilde{P}$. First assume that $P \notin \{\overline{P}, \widetilde{P}\}$. Suppose for the sake of deriving a contradiction that the color of P changes at a vertex $u \neq v$. Note that P does not contain any black arcs since it is arc-disjoint from $\overline{P}$ and $\widetilde{P}$. Because P starts at s we have $u \neq s$. Moreover, for a color change to be possible at u, the vertex u must lie on the s-v subpath of either $\overline{P}$ or $\widetilde{P}$. Thus, we either have $|C \cap \delta(u)| = 2$, in which case u is visited by exactly one of the s-v subpaths of $\overline{P}$ and $\widetilde{P}$, or we have $|C \cap \delta(u)| = 4$, in which case u is visited both by the s-v subpath of $\overline{P}$ and the s-v subpath of $\widetilde{P}$. This holds because any path, including $\overline{P}$ and $\widetilde{P}$, can visit any vertex at most once. If $|C \cap \delta(u)| = 2$, then when changing colors at u the path P would cross the path $Q \in \{\overline{P}, \widetilde{P}\}$ whose s-v subpath visits u. More formally, let a be the single arc in $Q \cap \delta^-(u)$ and b be the single arc in $Q \cap \delta^+(v)$. The arcs counterclockwise in π_u between a and b must either all be red or blue (say red), and all arcs clockwise between a and b have the other color among red/blue (say blue), see

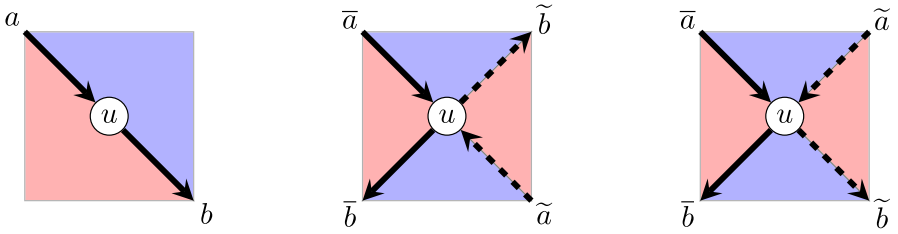


Fig. 8 Illustration of the neighborhood of a vertex u in C. The three different possible situations (up to symmetry and exchange of the colors red and blue) are illustrated in the picture. Here the solid and dashed arcs belong to the s-v subpaths of $\overline{P}$ and $\widetilde{P}$, respectively. A path changing its color from a red face to a blue face (or the other way around) at u crosses one of the paths $\overline{P}$ and $\widetilde{P}$

the left part of Figure 8. Up to symmetry this means π_u has the following structure:

$$\pi_u = (a, \underbrace{\ldots}_{red}, b, \underbrace{\ldots}_{blue}).$$

Thus, a path entering u with an arc of color red or blue and leaving u with an arc of the other of these colors implies a crossing among the paths $P_1, \ldots, P_\ell$, which violates that they are a nice s-path partitioning.

So assume $|C \cap \delta(u)| = 4$. Hence, the four primal faces adjacent to u have colors that alternate between red and blue. (More precisely, as we color the arcs, the color of a face corresponds to the color of all arcs within that face.) Let us denote by $\overline{a}$ and $\widetilde{a}$ the arc of $\overline{P}$ and $\widetilde{P}$, respectively, that enters u, and by $\overline{b}$ and $\widetilde{b}$ the arc of $\overline{P}$ and $\widetilde{P}$, respectively, that leaves u, i.e., $\{\overline{a}\} = \overline{P} \cap \delta^-(u)$, $\{\widetilde{a}\} = \widetilde{P} \cap \delta^-(u)$, $\{\overline{b}\} = \overline{P} \cap \delta^+(u)$, and $\{\widetilde{b}\} = \widetilde{P} \cap \delta^+(u)$. Because $\overline{P}$ and $\widetilde{P}$ do not cross, we can describe the structure of π_u up to symmetries as follows:

$$\pi_u = (\overline{a}, \underbrace{\ldots}_{red}, \overline{b}, \underbrace{\ldots}_{blue}, \widetilde{a}, \underbrace{\ldots}_{red}, \widetilde{b}, \underbrace{\ldots}_{blue}) \quad \text{or}$$
$$\pi_u = (\overline{a}, \underbrace{\ldots}_{red}, \overline{b}, \underbrace{\ldots}_{blue}, \widetilde{b}, \underbrace{\ldots}_{red}, \widetilde{a}, \underbrace{\ldots}_{blue}).$$

Again, a path P would have to cross one of the paths $\overline{P}$ and $\widetilde{P}$ when changing color in u. See Figure 8 for a visualization. This is not possible in a nice s-path partitioning.

We conclude that P can change its color only at the vertex v. Because P is a path, it can thus change its color at most once.

Now suppose $P \in \{\overline{P}, \widetilde{P}\}$, and let $Q \in \{\overline{P}, \widetilde{P}\}$ be the other path. By definition, P is black until v and becomes red or blue after that. A second change of color is thus only possible at a vertex that P visits after v. Because P cannot visit a vertex twice, such a vertex is not visited by the s-v subpath of P. Thus, if P changes its color at a vertex $u \neq v$, the vertex u must be an internal vertex of the s-v subpath of Q, and we have $|C \cap \delta(u)| = 2$. However, at such a vertex u, the path P cannot change color as this would imply a crossing of P and Q at u. Thus, after changing color from black to red or blue at vertex v, the path P does not change color again, as desired. □

4.1 Characterization of paths sharing an arc

We now derive Lemma 2.14 from Lemma 4.1. For convenience, we restate the lemma below and recall that, for $t \in T$, the set

$$\mathcal{P}^t := \{P_i : i \in [\ell] \text{ and } P_i \text{ ends at terminal } t\},$$

are all paths in the nice s-path partitioning that end at terminal t.

Lemma 2.14 *For every arc $a \in A$, the set $\{i \in [\ell] : a \in \phi(P_i)\}$ is discrepancy-equivalent to a circular interval.*

Proof Fix some arc $a = (v, w) \in A$ and a geometric planar embedding of H realizing the combinatorial embedding π. Let

$$I := \{i \in [\ell] : a \in \phi(P_i)\} \text{ and}$$
$$\mathcal{P} := \{P_i : i \in I\}$$

be the indices that we want to show to be discrepancy-equivalent to a circular interval, and the corresponding paths, respectively.

If $v = s$, then I is a circular interval because the paths $P_1, \ldots, P_\ell$ are source-numbered and the arcs in $\phi^{-1}(a)$ appear consecutively in π_s (see Definition 3.3, i.e., the definition of an arc-split graph). Moreover, if $|\mathcal{P}| \leq 1$ the statement of the lemma is trivial. Hence, we will now assume $s \neq v$ and $|\mathcal{P}| \geq 2$.

By definition of $\mathcal{P}$, every path in $\mathcal{P}$ uses an arc of $\phi^{-1}(a) \subseteq \delta^+(v)$. Since $v \neq s$, every path of $\mathcal{P}$ also uses an arc of $\delta^-(v)$. We denote these arcs as F_{in}, i.e., $F_{\text{in}} \subseteq F$ contains all arcs that belong to a path of $\mathcal{P}$ and enter v.

Recall that by the definition of an arc-split graph, the arcs in $\phi^{-1}(a)$ appear consecutively in π_v. Consider the first arc of F_{in} appearing after the arcs of $\phi^{-1}(a)$ in clockwise direction in the geometric embedding and let P_{clo} be the unique path in $\mathcal{P}$ using this arc. (Uniqueness follows from the fact that the different paths in $\mathcal{P}$ are arc-disjoint by definition of an s-path partitioning.) Similarly, consider the first arc of F_{in} after the arcs of $\phi^{-1}(a)$ in counterclockwise direction and let P_{cc} be the unique path using this arc. Because $|\mathcal{P}| \geq 2$, also $|F_{\text{in}}| \geq 2$ and, thus, the paths P_{cc} and P_{clo} exist and are distinct.

Both P_{clo} and P_{cc} start in s and contain the vertex v. We consider the tri-coloring of H induced by P_{clo}, P_{cc}, and v. Note that, for each terminal $t \in T$, either all arcs in $\delta^-(t)$ are red or all arcs in $\delta^-(t)$ are blue. This follows from the fact that terminals cannot lie in the interior of the paths P_{clo} or P_{cc}, because they do not have outgoing arcs. In particular, v cannot be a terminal, because it has an outgoing arc $a = (v, w)$. We call a terminal $t \in T$ *red* if all arcs in $\delta^-(t)$ are red, and we call it *blue* if all arcs in $\delta^-(t)$ are blue. Analogously, the vertex $w := \text{head}(a)$ does not lie on the s-v subpaths of either P_{clo} or P_{cc} because both paths P_{clo} and P_{cc} contain some arc from $\phi^{-1}((v, w))$, and w cannot appear twice on any path. Hence, also the arcs in $\delta(w)$ are either all red or all blue. We assume without loss of generality that the arcs in $\delta(w)$ are all colored red. See Figure 9 for an illustration.

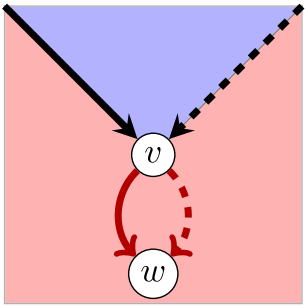


Fig. 9 Illustration of the neighborhood of v. The solid arcs are arcs of P_{clo} and the dashed arcs belong to P_{cc}

Let $\mathcal{Q}$ be the set of paths that contains P_{clo}, P_{cc}, and all paths $P \in \{P_1, \dots, P_\ell\}$ whose first arc is colored blue. By construction of the tri-coloring and since s is adjacent to only two black arcs, the index set $J := \{i \in [\ell] \colon P_i \in \mathcal{Q}\}$ is a circular interval. We claim that the set I of indices of paths in $\mathcal{P}$ is discrepancy-equivalent to the set J of indices of paths in $\mathcal{Q}$. To this end, we will show that:

(i) For every red terminal $t \in T$, we have $\mathcal{P}^t \cap \mathcal{P} = \mathcal{P}^t \cap \mathcal{Q}$.
(ii) For every blue terminal $t \in T$, we have $\mathcal{P}^t \cap \mathcal{P} = \emptyset$ and $\mathcal{P}^t \subseteq \mathcal{Q}$.

Hence, J can be obtained from I by S^t-addition for all terminals $t \in T$ that are colored blue, where $S^t := \{i \in [\ell] \colon P_i \in \mathcal{P}^t\}$.

It remains to prove (i) and (ii). By Lemma 4.1, a path $P \in \{P_1, \dots, P_\ell\}$ can only change its color at v. We show that this implies that P can only change its color from either blue or black to red, but not the other way around. Hence, once a path is red, it stays red. Because all arcs incident with w are red, including all arcs in $\phi^{-1}(a)$, this implies (ii). More concretely, given a blue terminal t, a path $P \in \mathcal{P}^t$ can never use a red arc, because it would then stay red and can therefore not end in a blue terminal. Thus, $P \in \mathcal{Q}$. Moreover, $P \notin \mathcal{P}$ since otherwise it contains w, which only has red arcs incident with it. Note that this also implies that both P_{clo} and P_{cc} do not have a blue terminal since they use an arc of $\phi^{-1}(a)$, which is red.

To show that a path can only change the color from blue or black to red but not the other way around, remember that paths can only change their color in v by Lemma 4.1. The vertex v has only two adjacent black arcs (one incoming arc each from P_{clo} and P_{cc}). Both P_{clo} and P_{cc} use an arc of $\phi^{-1}(a)$ by definition and all the arcs of $\phi^{-1}(a)$ appear consecutively in the embedding. Thus, π_v has the following form, where a_{clo} and a_{cc} are the arcs of P_{clo} and P_{cc}, respectively, that enter v, and b_{clo} and b_{cc} are the arcs of P_{clo} and P_{cc}, respectively, that leave v, i.e., $\{a_{\text{clo}}\} = P_{\text{clo}} \cap \delta^-(v)$, $\{b_{\text{clo}}\} = P_{\text{clo}} \cap \delta^+(v)$, $\{a_{\text{cc}}\} = P_{\text{cc}} \cap \delta^-(v)$, and $\{b_{\text{cc}}\} = P_{\text{cc}} \cap \delta^+(v)$:

$$\pi_v = (a_{\text{clo}}, \underbrace{\dots}_{red}, \underbrace{b_{\text{clo}}, \dots, b_{\text{cc}}}_{\subseteq \phi^{-1}(a)}, \underbrace{\dots}_{red}, a_{\text{cc}}, \underbrace{\dots}_{blue}). \tag{4.1}$$

One can even show that all arcs of $\phi^{-1}(a)$ lie counterclockwise between b_{clo} and b_{cc}, but since it is not needed for the proof, we just use the simpler fact that $b_{\text{clo}}, b_{\text{cc}} \in \phi^{-1}(a)$ and thus also all arcs which are counterclockwise between them are contained in $\phi^{-1}(a)$. See Figure 10 for an illustration of the neighborhood of v.

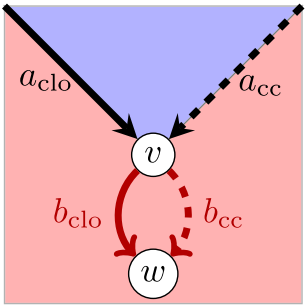


Fig. 10 Illustration of the arcs $a_{\text{clo}}, b_{\text{clo}}, a_{\text{cc}}, b_{\text{cc}}$ incident to v. The solid arcs are arcs of P_{clo} and the dashed arcs belong to P_{cc}

To change the color from red to blue, a path cannot use an arc of $\phi^{-1}(a)$, since they are all red and leave v. This means that any path changing color from red to blue needs to cross either P_{clo} or P_{cc}, which is forbidden by the definition of a nice s-path partitioning.

It remains to prove (i). To this end we show that every $P \in \{P_1, \ldots, P_\ell\}$ satisfies the following.

(a) If P enters v via a red arc, then it does not use an arc from $\phi^{-1}(a)$, i.e., $P \notin \mathcal{P}$.
(b) If P changes its color from blue to red, then it must use an arc from $\phi^{-1}(a)$, i.e., $P \in \mathcal{P}$.

Then (i) follows, because (a) implies that every path in $\mathcal{P}$, i.e., every path using an arc in $\phi^{-1}(a)$ from v to w, must change its color at v from blue to red and thus must be contained in $\mathcal{Q}$. Hence, $\mathcal{P} \subseteq \mathcal{Q}$, and thus $\mathcal{P}^t \cap \mathcal{P} \subseteq \mathcal{P}^t \cap \mathcal{Q}$ for every terminal $t \in T$. (Here we used that P_{cc} and P_{clo} are contained in $\mathcal{Q}$ by definition.) Now consider a path $P \in \mathcal{P}^t \cap \mathcal{Q}$ for some red terminal $t \in T$. Then (b) implies that P uses an arc from $\phi^{-1}(a)$, i.e., we have $P \in \mathcal{P}$. (Note that P_{clo} and P_{cc} are contained in $\mathcal{P}$ by definition.) Thus, it remains to prove (a) and (b).

First, (a) follows by definition of P_{clo} and P_{cc}. If a path enters v using a red arc, this arc lies either clockwise between b_{clo} and a_{clo} or counterclockwise between b_{cc} and a_{cc}, see Figure 9 and Equation (4.1). This is because the red arcs of $\delta(v)$ are precisely those that are counterclockwise between a_{clo} and a_{cc}; moreover, all arcs counterclockwise between b_{clo} and b_{cc} are arcs of $\phi^{-1}(a)$, which leave v. By definition of P_{clo}, there is no path entering v clockwise between b_{clo} and a_{clo} and using an arc of $\phi^{-1}(a)$ afterward. Analogously, by definition of P_{cc}, there is no path entering counterclockwise between b_{cc} and a_{cc} and then using an arc of $\phi^{-1}(a)$. This shows (a).

Finally, (b) follows because paths are non-crossing. By Lemma 4.1 a path can only change its color in v. Now, assume for the sake of deriving a contradiction that there is a path P that enters v via a blue arc and leaves v via a red one not in $\phi^{-1}(a)$. Such a path either crosses P_{clo} (when it leaves clockwise between b_{clo} and a_{clo}) or P_{cc} (when it leaves counterclockwise between b_{cc} and a_{cc})—see also Figure 9 and Equation (4.1)—which is a contradiction. □

4.2 Nice *s*-path partitionings are non-interleaving

We now show Lemma 2.3, which we restate here for convenience.

Lemma 2.3 *For $t \in T$ let $S^t := \{i \in [\ell] \colon P_i \in \mathcal{P}^t\}$. Then the partition $\{S^t : t \in T\}$ of $[\ell]$ is non-interleaving.*

Proof Assume for the sake of deriving a contradiction that there are two interleaving sets S^{t_1} and S^{t_2}, for some $t_1, t_2 \in T$, with $a_i, b_i \in S^{t_i}$ for $i \in [2]$ such that $a_1 < a_2 < b_1 < b_2$.

Consider the paths P_{a_1} and P_{b_1}, which both go to the same terminal t_1. We consider the tri-coloring of H induced by P_{a_1}, P_{b_1}, and t_1, see Figure 7 with $v = t_1$. By Lemma 4.1, a path can change the color only once and only at t_1. Because t_1 does not have any outgoing arcs, we get that no path can change its color. As paths are source-numbered, P_{a_2} and P_{b_2} start with arcs of different color. Thus, their joint terminal t_2 needs to lie on either P_{a_1} or P_{b_1}, because for all vertices w not lying on one of these two paths, the arcs in $\delta(w)$ have all the same color. However, as t_2 does not have any outgoing arcs and is disjoint from t_1, it cannot lie on either P_{a_1} or P_{b_1}, leading to a contradiction. □

5 Interval-discrepancy on non-interleaving partitions

In this section we prove our two main discrepancy statements, Theorem 2.10 and Theorem 2.11.

5.1 The case without costs (Proof of Theorem 2.10)

For convenience, we first recall Theorem 2.10.

Theorem 2.10 *Consider a WPCS instance $(\ell, \mathcal{S}, d, y)$. If $\mathcal{S}$ is non-interleaving, there exists an integral selection z with*

$$D_{y,z} \leq d_{\max},$$

where $d_{\max} := \max_{S \in \mathcal{S}} d_S$. Moreover, it can be computed in time $O(\ell)$.

In order to prove Theorem 2.10, we show the following bound on the discrepancy of the prefix intervals $[i]$ with $i \in [\ell]$.

Theorem 5.1 *Consider a WPCS instance $(\ell, \mathcal{S}, d, y)$. If $\mathcal{S}$ is non-interleaving, then there exists an integral selection z with*

$$D_{y,z}([i]) \leq \frac{d_{\max}}{2}$$

for all $i \in [\ell]$, where $d_{\max} := \max_{S \in \mathcal{S}} d_S$. Moreover, it can be computed in time $O(\ell)$.

Before proving Theorem 5.1, we first observe that it implies Theorem 2.10. For any interval $I = \{i, i+1, \ldots, j\} \subseteq [\ell]$, we have

$$D_{y,z}(I) = D_{y,z}([j] \setminus [i-1]) \leq D_{y,z}([i-1]) + D_{y,z}([j]) \leq d_{\max}.$$

If I is a circular interval, then either I is an interval or the complement $[\ell] \setminus I$ of I is an interval. In the latter case we have $D_{y,z}(I) = D_{y,z}([\ell] \setminus I) \leq d_{\max}$.

Hence, it remains to prove Theorem 5.1, which we do in the remainder of this section. We fix a WPCS instance $(\ell, \mathcal{S}, d, y)$ such that $\mathcal{S}$ is non-interleaving.

For a given integral selection $z \in \{0,1\}^\ell$, we define the values

$$D^k := y^d([k]) - z^d([k]) \quad \text{for } k \in \{0, \dots, \ell\}.$$

Then $D_{y,z}([k]) = \left|D^k\right|$. We now show that the difference of D^i and D^j for $i, j \in S \in \mathcal{S}$ depends only on the restriction of z to S and not on any other entry of the vector z. This is a key consequence of the fact that $\mathcal{S}$ is non-interleaving.

Lemma 5.2 *Let $z \in \{0,1\}^\ell$ be an integral selection. Let $S \in \mathcal{S}$ and $i, j \in S$ with $i \leq j$. Then*

$$D^j - D^{i-1} = \sum_{\substack{m \in S: \\ i \leq m \leq j}} (y_m^d - z_m^d).$$

Proof Consider a set $R \in \mathcal{S} \setminus \{S\}$. Because $\mathcal{S}$ is non-interleaving, we have either $R \cap \{i, i+1, \dots, j\} = \emptyset$ or $R \subseteq \{i, i+1, \dots, j\}$. In both cases we have $\sum_{m \in R:\, i \leq m \leq j} (y_m^d - z_m^d) = 0$, and thus

$$D^j - D^{i-1} = \sum_{m=i}^{j} (y_m^d - z_m^d) = \sum_{U \in \mathcal{S}} \sum_{\substack{m \in U: \\ i \leq m \leq j}} (y_m^d - z_m^d) = \sum_{\substack{m \in S: \\ i \leq m \leq j}} (y_m^d - z_m^d).$$

□

We now describe the algorithm which we use to compute an integral selection z with the desired properties. The algorithm fixes the entries z_j from $j = 1$ to $j = \ell$ one by one in this order, while maintaining the desired prefix discrepancy bound $D^j \in [-\frac{d_{\max}}{2}, \frac{d_{\max}}{2}]$.

Suppose the currently considered entry is z_j with $j \in S \in \mathcal{S}$. If, for some $k \in S$ with $k < j$, the entry z_k has already been fixed to 1 in an earlier iteration, the algorithm has to choose $z_j = 0$ in order to obtain an integral selection. Similarly, if j is the largest element of S, and we have chosen $z_k = 0$ for all other elements k of S, the algorithm must set $z_j = 1$ to obtain an integral selection. We will use Lemma 5.2 to prove that in both of these cases we have $\left|D^j\right| \leq \frac{d_{\max}}{2}$ (using that prior prefix discrepancies are fine, i.e., $|D^i| \leq \frac{d_{\max}}{2}$ for $i < j$). If neither of the above two cases applies, our algorithm greedily fixes the currently considered entry z_j to ensure $\left|D^j\right| \leq \frac{d_{\max}}{2}$. This is possible because $D^{j-1} \in [-\frac{d_{\max}}{2}, \frac{d_{\max}}{2}]$ and the two different values that D^j can attain, depending on the choice of $z^j \in \{0,1\}$, differ by at most $d_S \leq d_{\max}$ from each other, where one of the two values lies below D^{j-1} and the other one lies above D^{j-1}. A formal description of the algorithm is given by Algorithm 3. Note that the runtime of the algorithm is $O(\ell)$, since every iteration of the for-loop can be executed

Algorithm 3: Computing an integral selection z with small interval-discrepancy.

for $j = 1, \ldots, \ell$ **do**
Let $S \in \mathcal{S}$ such that $j \in S$.
If $z_k = 1$ for some $k \in S$ with $k < j$, set $z_j := 0$.
If j is the largest element of S and $z_k = 0$ for all $k \in S$ with $k < j$, set $z_j := 1$.
If none of the above two cases apply, set

$$z_j := \begin{cases} 1 & \text{if } D^{j-1} + y^d_j - d_S \geq -\frac{d_{\max}}{2} \\ 0 & \text{otherwise,} \end{cases}$$

where $D^{j-1} := y^d([j-1]) - z^d([j-1])$.
return z.

in constant time. (D^{j-1} does not need to be recomputed from scratch in iteration j but can be obtained from the previously computed D^{j-2} in constant time.)

The vector z returned by Algorithm 3 is an integral selection by construction. It remains to prove that it fulfills the discrepancy bound claimed by Theorem 5.1.

Lemma 5.3 *Let z be the integral selection returned by Algorithm 3. For all $j \in \{0, \ldots, \ell\}$ we have $D_{y,z}([j]) = \left|D^j\right| \leq \frac{d_{\max}}{2}$.*

Proof We prove the lemma by induction on j, starting with $j = 0$. We have $D^0 = 0$ and hence assume $j > 0$. As in Algorithm 3, we let $S \in \mathcal{S}$ such that $j \in S$. We distinguish the same three cases as the algorithm.

First, suppose $z_k = 1$ for some $k \in S$ with $k < j$. Let i be the minimal element of S. By Lemma 5.2, we have

$$D^k = D^{i-1} + y^d(\{m \in S \colon i \leq m \leq k\}) - d_S, \text{ and}$$
$$D^j = D^{i-1} + y^d(\{m \in S \colon i \leq m \leq j\}) - d_S.$$

Since y is a selection, we have $y^d(\{m \in S \colon i < m \leq j\}) \leq d_S$. This implies $D^j \leq D^{i-1}$. Moreover, because $y^d \geq 0$ and $k < j$, we get $D^k \leq D^j$. But since $\left|D^{i-1}\right| \leq \frac{d_{\max}}{2}$ and $\left|D^k\right| \leq \frac{d_{\max}}{2}$ by induction, we get $\left|D^j\right| \leq \frac{d_{\max}}{2}$.

Now suppose j is the largest element of S and $z_k = 0$ for all $k \in S$ with $k < j$. Let i be the minimal element of S. Then Lemma 5.2 implies

$$D^j - D^{i-1} = \sum_{m \in S \colon i \leq m \leq j} (y^d_m - z^d_m) = \sum_{m \in S} (y^d_m - z^d_m) = 0,$$

where the second equation follows from the fact that i is the smallest element of S and j the largest one, and the last equality holds due to $y(S) = 1$ and $z(S) = 1$ because y and z is a fractional and integral selection, respectively. By the induction hypothesis, we conclude $\left|D^j\right| = \left|D^{i-1}\right| \leq \frac{d_{\max}}{2}$.

Finally, we assume that none of the above cases applies. If we set $z_j = 1$, then we have $D^j = D^{j-1} + y^d_j - z^d_j = D^{j-1} + y^d_j - d_S \geq -\frac{d_{\max}}{2}$, where we used the choice

of z_j in Algorithm 3. Moreover, because $y_j^d = y_j d_S \leq d_S$, we have $D^{j-1} \geq D^j$, which implies $D^j \leq \frac{d_{\max}}{2}$ by the induction hypothesis. If we set $z_j = 0$, we have $D^j \geq D^{j-1}$, which implies $D^j \geq -\frac{d_{\max}}{2}$ by the induction hypothesis. Moreover, by the choice of z_j in Algorithm 3 we have $D^{j-1} + y_j^d - d_S \leq -\frac{d_{\max}}{2}$, implying $D^j = D^{j-1} + y_j^d \leq d_S - \frac{d_{\max}}{2} \leq \frac{d_{\max}}{2}$, where we used $d_S \leq d_{\max}$. □

5.2 The case with costs (Proof of Theorem 2.11)

We first restate Theorem 2.11 for convenience.

Theorem 2.11 *Consider a WPCS instance $(\ell, \mathcal{S}, d, y)$ and let $c \in \mathbb{Q}^\ell$. If $\mathcal{S}$ is non-interleaving, there exists an integral selection z with*

$$D_{y,z} \leq 2d_{\max} \quad \text{and}$$
$$c^T z \leq c^T y,$$

where $d_{\max} := \max_{S \in \mathcal{S}} d_S$. Moreover, it can be computed in time $O(\ell \cdot \langle \text{input} \rangle)$, where $\langle \text{input} \rangle$ refers to the length of the bit encoding of the input.

As in the proof of Theorem 2.10, we start by bounding the discrepancy of the prefix intervals $[i]$ with $i \in [\ell]$.

Theorem 5.4 *Consider a WPCS instance $(\ell, \mathcal{S}, d, y)$ and let $c \in \mathbb{Q}^\ell$. If $\mathcal{S}$ is non-interleaving, there exists an integral selection z with*

$$D_{y,z}([i]) \leq d_{\max} \quad \text{for all } i \in [\ell] \text{ and}$$
$$c^T z \leq c^T y,$$

where $d_{\max} := \max_{S \in \mathcal{S}} d_S$. Moreover, such a selection z can be computed in time $O(\langle \text{input} \rangle \cdot \ell)$.

Before proving Theorem 5.4, we first observe that it implies Theorem 2.11. For any interval $I = \{i, i+1, \ldots, j\} \subseteq [\ell]$, we have

$$D_{y,z}(I) = D_{y,z}([j] \setminus [i-1]) \leq D_{y,z}([i-1]) + D_{y,z}([j]) \leq 2d_{\max}.$$

If I is a circular interval, then either I is an interval or the complement $[\ell] \setminus I$ of I is an interval. In the latter case we have $D_{y,z}(I) = D_{y,z}([\ell] \setminus I) \leq 2d_{\max}$.

To prove Theorem 5.4, we first consider a special case of it where the given fractional selection y is half-integral. In this special case we will be able to achieve a stronger discrepancy bound. Then we will show that the general case can be reduced to the special case at the cost of increasing the interval discrepancy by a factor of two. The approach we employ to reduce to this special case is a common argument in discrepancy theory (see, e.g, [23], and [4] for a recent application in a scheduling context).

Lemma 5.5 *Consider a WPCS instance $(\ell, \mathcal{S}, d, y)$ and let $c \in \mathbb{Q}^\ell$. Suppose $|S| = 2$ for all $S \in \mathcal{S}$ and $y_i = \frac{1}{2}$ for all $i \in [\ell]$. If $\mathcal{S}$ is non-interleaving, there exists an integral selection z with*

$$D_{y,z}([i]) \leq \frac{d_{\max}}{2} \quad \text{for all } i \in [\ell] \text{ and}$$
$$c^T z \leq c^T y,$$

where $d_{\max} := \max_{S \in \mathcal{S}} d_S$. Moreover, such a selection z can be computed in time $O(\ell)$.

Proof We apply Theorem 5.1 to obtain an integral selection $z \in \{0,1\}^\ell$ with $D_{y,z}([i]) \leq \frac{d_{\max}}{2}$ for all $i \in [\ell]$. We show that either z or the vector $\overline{z} \in \{0,1\}^\ell$, defined by

$$\overline{z}_i := \begin{cases} 1 & \text{if } z_i = 0 \\ 0 & \text{if } z_i = 1, \end{cases}$$

has the desired properties. First observe that for any integral selection $q \in \{0,1\}^\ell$ and any $j \in S \in \mathcal{S}$, we have

$$y_j^d - q_j^d = \begin{cases} \frac{d_S}{2} & \text{if } q_j = 0 \\ -\frac{d_S}{2} & \text{if } q_j = 1, \end{cases}$$

where we used $y_i = \frac{1}{2}$ for all $i \in [\ell]$. This implies $y_j^d - \overline{z}_j^d = -(y_j^d - z_j^d)$ for all $j \in [\ell]$. Thus, we have for all $i \in [\ell]$:

$$D_{y,\overline{z}}([i]) = \left| \sum_{j \in [i]} (y_j^d - \overline{z}_j^d) \right| = \left| -\sum_{j \in [i]} (y_j^d - z_j^d) \right| = D_{y,z}([i]) \leq \frac{d_{\max}}{2}.$$

Moreover, we have $z + \overline{z} = 2y$ and thus $\min\{c^T z, c^T \overline{z}\} \leq c^T y$. We conclude that one of the integral selections z or $\overline{z}$ has the desired properties. The runtime is $O(\ell)$, since Theorem 5.1 gives an integral selection z in that time and computing $\overline{z}$ and evaluating the cost is both also linear in ℓ. □

For $\alpha \in \mathbb{Q}$, we call a vector $y \in \mathbb{Q}^\ell$ an *α-integral vector* if every entry of y is an integer multiple of α. As a consequence of Lemma 5.5 we obtain the following.

Lemma 5.6 *Consider a WPCS instance $(\ell, \mathcal{S}, d, y)$ where $\mathcal{S}$ is non-interleaving, let $c \in \mathbb{Q}^\ell$, and let $k \in \mathbb{Z}_{>0}$. Given a 2^{-k}-integral fractional selection y, we can compute in polynomial time a 2^{-k+1}-integral fractional selection $\overline{y}$ such that*

$$\left| y^d([i]) - \overline{y}^d([i]) \right| \leq 2^{-k} \cdot d_{\max} \quad \text{for all } i \in [\ell] \text{ and}$$
$$c^T \overline{y} \leq c^T y,$$

where $d_{\max} \coloneqq \max_{S \in \mathcal{S}} d_S$. *Moreover, such a selection* $\overline{y}$ *can be computed in time* $O(\ell)$.

Proof We construct a WPCS instance $(\hat{\ell}, \hat{\mathcal{S}}, \hat{d}, \hat{y})$ to which we will then apply Lemma 5.5. Let

$$F \coloneqq \{i \in [\ell] \colon y_i \text{ is not } 2^{-k+1}\text{-integral}\},$$

and we denote by $\hat{\ell} \coloneqq |F|$ the number of elements in F. Moreover, we define a renumbering $f \colon F \to [\hat{\ell}]$ such that f is the (unique) bijection preserving the order of elements, i.e., $f(i) < f(j)$ if and only if $i < j$. Because y is 2^{-k}-integral and $y(S) = 1$ for $S \in \mathcal{S}$, we have that $|F \cap S|$ is even for every set $S \in \mathcal{S}$. Hence, there is a partition $\mathcal{R}_S$ of $F \cap S$ into sets of size two, which we choose such that no two sets in $\mathcal{R}_S$ are interleaving. This can be achieved, e.g., by considering the elements of $F \cap S$ in increasing order and repeatedly putting two consecutive elements in the same set in $\mathcal{R}_S$. We define

$$\hat{\mathcal{S}} \coloneqq \{\{f(i), f(j)\} \colon \{i, j\} \in \mathcal{R}_S, S \in \mathcal{S}\}.$$

Then $\hat{\mathcal{S}}$ is a partition of $[\hat{\ell}]$. Moreover, $\hat{\mathcal{S}}$ is non-interleaving, because any two distinct sets from the same set $\mathcal{R}_S$ are non-interleaving by construction, and for $R_1 \in \mathcal{R}_{S_1}$ and $R_2 \in \mathcal{R}_{S_2}$ for distinct sets $S_1, S_2 \in \mathcal{S}$, the sets $R_1 \subseteq S_1$ and $R_2 \subseteq S_2$ are non-interleaving because S_1 and S_2 are non-interleaving. For a set $\hat{S} = \{f(i), f(j)\}$ with $\{i, j\} \subseteq S \in \mathcal{S}$, we define $\hat{d}_{\hat{S}} \coloneqq d_S$. Moreover, we set $\hat{y}_i \coloneqq \frac{1}{2}$ for all $i \in [\hat{\ell}]$ and $\hat{c}_{f(i)} \coloneqq c_i$ for all $i \in [\hat{\ell}]$.

We apply Lemma 5.5 to obtain an integral selection $\hat{z}$ for the WPCS instance $(\hat{\ell}, \hat{\mathcal{S}}, \hat{d}, \hat{y})$ with cost $\hat{c}$. Hence, $D_{\hat{y},\hat{z}}([\hat{i}]) \leq \frac{d_{\max}}{2}$ for $\hat{i} \in [\hat{\ell}]$, and $\hat{c}^T \hat{z} \leq \hat{c}^T \hat{y}$. Then we define

$$\overline{y}_i \coloneqq \begin{cases} y_i & \text{if } i \in [\ell] \setminus F \\ y_i + 2^{-k+1} \cdot \left(\hat{z}_{f(i)} - \hat{y}_{f(i)}\right) & \text{if } i \in F. \end{cases}$$

Because $\hat{z}$ is integral, the definition of F implies that $\overline{y}$ is 2^{-k+1}-integral. Moreover, $\hat{c}^T \hat{z} \leq \hat{c}^T \hat{y}$ implies $c^T \overline{y} \leq c^T y$. Finally, we consider an index $i \in [\ell]$. If $F \cap [i] = \emptyset$, we have $\overline{y}^d([i]) = y^d([i])$. Otherwise, let $i_F \coloneqq \max\{f(j) \colon j \in F \cap [i]\}$. Then

$$\left|y^d([i]) - \overline{y}^d([i])\right| = 2^{-k+1} \left|\hat{y}^d([i_F]) - \hat{z}^d([i_F])\right| = 2^{-k+1} \cdot D_{\hat{y},\hat{z}}([i_F]) \leq 2^{-k} d_{\max}.$$

By Lemma 5.5, we obtain $\hat{z}$ in $O(\hat{\ell})$, where $\hat{\ell} \leq \ell$ by definition. The subsequent definition of $\bar{y}$ can clearly be done in $O(\ell)$. □

By applying Lemma 5.6 repeatedly, we can round a 2^{-k}-integral fractional selection to an integral one. However, we might be given a fractional selection y that is not 2^{-k}-integral for any $k \in \mathbb{Z}_{>0}$. To handle this case, we will use the following simple observation.

Lemma 5.7 *Consider a WPCS instance* $(\ell, \mathcal{S}, d, y)$ *where* $\mathcal{S}$ *is non-interleaving, and let* $c \in \mathbb{Q}^\ell$. *Let* $\varepsilon > 0$ *and* $k_\varepsilon \coloneqq \lceil \log_2(\frac{\ell}{\varepsilon}) \rceil$. *Then, there exists a* 2^{-k_ε}*-integral fractional selection* $\tilde{y}$ *such that*

$$\left|y^d(U) - \tilde{y}^d(U)\right| \leq \varepsilon \cdot d_{\max} \quad \textit{for all } U \subseteq [\ell], \textit{ and}$$

$$c^T \tilde{y} \leq c^T y,$$

Moreover, such a selection $\tilde{y}$ can be computed in time $O(\ell)$.

Proof For each $S \in \mathcal{S}$, let $i_S \in S$ be such that its cost is minimum, i.e., such that $c_{i_S} \leq c_j$ for all $j \in S$. Then, for all $j \in S \setminus \{i_S\}$, let $\tilde{y}_j$ be the number obtained from rounding down y_j to the next integer multiple of 2^{-k_ε}. We define $\tilde{y}_{i_S} \coloneqq 1 - \sum_{j \in S \setminus \{i_S\}} \tilde{y}_j$. Then $\tilde{y}$ is a fractional selection and, by the choice of the element $i_S \in S$, we have $c^T \tilde{y} \leq c^T y$. Moreover, for every set $U \subseteq [\ell]$, we have $|y(U) - \tilde{y}(U)| \leq \ell \cdot 2^{-k_\varepsilon} \leq \varepsilon$ and thus $\left|y^d(U) - \tilde{y}^d(U)\right| \leq \varepsilon \cdot d_{\max}$. □

We now complete the proof of Theorem 5.4. By applying once Lemma 5.7 for some $\varepsilon > 0$ and applying Lemma 5.6 repeatedly $k_\varepsilon = \lceil \log_2(\frac{\ell}{\varepsilon}) \rceil$ times, we obtain an integral selection z such that, for all $i \in [\ell]$,

$$\left|y^d([i]) - z^d([i])\right| \leq \varepsilon \cdot d_{\max} + \sum_{i=1}^{k_\varepsilon} 2^{-i} \cdot d_{\max} < (1+\varepsilon) \cdot d_{\max}. \tag{5.1}$$

If we choose $\varepsilon > 0$ small enough, i.e., such that

$$\varepsilon \cdot d_{\max} < \left|y^d([i]) - \tilde{z}^d([i])\right| - d_{\max} \tag{5.2}$$

for all integral selections $\tilde{z}$ and all $i \in [\ell]$ for which $\left|y^d([i]) - \tilde{z}^d([i])\right| > d_{\max}$, then (5.1) implies $\left|y^d([i]) - z^d([i])\right| \leq d_{\max}$. Because the right-hand side of (5.2) can attain only finitely many values, such an $\varepsilon > 0$ does indeed exist.

To obtain a polynomial time algorithm, we exploit rationality of the entries of the vectors y and d, and write $d_i = \frac{K_i}{L_i}$ and $y_i = \frac{M_i}{N_i}$ for integers K_i, L_i, M_i, N_i for all $i \in [\ell]$. Then we set

$$\varepsilon \coloneqq \frac{1}{d_{\max}} \cdot \prod_{i \in [\ell]} \frac{1}{L_i \cdot N_i}.$$

Note that for this choice of ε, the number $k_\varepsilon = \lceil \log_2(\frac{\ell}{\varepsilon}) \rceil$ of applications of Lemma 5.6 is linearly bounded in the input size, which implies an overall runtime of $O(\langle \text{input} \rangle \cdot \ell)$. Moreover, (5.1) implies that, for every $i \in [\ell]$, we have

$$\left|y^d([i]) - z^d([i])\right| - d_{\max} < \varepsilon \cdot d_{\max} = \prod_{i \in [\ell]} \frac{1}{L_i \cdot N_i}.$$

Because both $\left|y^d([i]) - z^d([i])\right|$ and $d_{\max}$ are integer multiples of $\prod_{i \in [\ell]} \frac{1}{L_i \cdot N_i}$, this implies that we have $\left|y^d([i]) - z^d([i])\right| \leq d_{\max}$, concluding the proof of Theorem 5.4.

6 Extension to higher-genus graphs

We now provide details on how our approach can be extended to general SSUF instances $(G = (V, A), s, T, d, x)$ to obtain Theorem 1.9, where the capacity violation depends on the genus of the underlying graph G, genus(G). Recall that genus(G) is the smallest value $g \in \mathbb{Z}_{\geq 0}$ such that G can be embedded on a sphere with g handles without any intersecting edges. Hence, planar graphs are graphs with genus 0.

Our approach requires basic operations on graphs embedded on a higher-genus surface, like determining whether two simple closed curves lie in the same homotopy class. We refer the interested reader to [8, 9, 24, 32] and references therein for further details on this and related computational questions on higher-genus surfaces, including how to efficiently find an embedding of a graph of genus $g = O(1)$ on a surface of genus g.

We start in Section 6.1 with an overview of how we extend our approach for planar instances.

6.1 Overview of extension

Loosely speaking, we aim at breaking an SSUF instance $(G = (V, A), s, T, d, x)$ into smaller pieces that can be solved and analyzed analogous to the planar case, where the number of pieces into which we break the problem depends on the genus of G.

We start by noting that even though many of our notions have been introduced in the context of planar graphs, they naturally extend to graphs embedded on surfaces of higher genus. For example, the non-crossing property of paths embedded on a surface is a local property defined per vertex, and any embedding without intersecting edges of a graph on a surface (even of genus higher than zero) is locally planar. Hence, also the notion of source-numbered s-path decomposition carries over. Two main difference to keep in mind are the following:

(i) An arc-split (H, π, ϕ) of G is embedded on a surface of genus genus(G), instead of being embedded on the plane (or the sphere) as in the planar case.
(ii) When considering an embedding of a graph on a surface, like (H, π) above, we need to be able to determine homotopy classes of closed curves. This can be achieved by considering a geometric embedding, which we simply call *embedding*, instead of just a combinatorial one. For ease of notation, we denote such an embedding also by π, and, for a vertex v, we denote its combinatorial embedding at v by π_v, as before.

However, the techniques we presented to compute arc-splits and nice source-numbered s-path decompositions all carry over to graphs embedded on surfaces of higher genus.

What does not carry over is the way we performed the rounding step, which, we recall, was performed by a reduction to a Weighted Partition-Constrained Selection (WPCS) problem, which asks to minimize interval-discrepancy. In the planar case, interval-discrepancy of WPCS readily led to a bound on capacity violation because the indices of all paths in a source-numbered s-path decomposition that contain any given arc $a \in A$ are discrepancy-equivalent to a circular interval by Lemma 2.14. This property fails for higher-genus graphs. Moreover, in the planar setting, the WPCS

problem abstracted away all geometric aspects, and admitted good solutions due to properties inherited from planarity of G, which do not hold for graphs G with higher genus. More precisely, for planar graphs, the resulting WPCS problem was with respect to a non-interleaving family of sets (due to Lemma 2.3); this is another property that fails for non-planar graph.

To address these hurdles in the higher genus case, we unfold the rounding step, whose main result in the planar case is described by Theorem 2.11, which in turn is based on Lemma 5.6. We recall that Lemma 5.6 was used to successively round the fractional solution bit per bit, until an integral path selection was obtained. Instead of using Lemma 5.6 as a black-box in the context of WPCS, we inspect and extend (a slightly weaker version of) this result to graphs of higher genus. More precisely, we will show the following. (As usual, we assume that the terminals in SSUF instances are distinct and do not have any outgoing arcs.)

Lemma 6.1 *Given is an SSUF instance (G, s, T, d, x) with costs $c \in \mathbb{Q}^A_{\geq 0}$, an arc-split (H, π, ϕ) of G, and a source-numbered nice s-path decomposition $(P_i, \lambda_i)_{i\in[\ell]}$ of H with the following properties:*

- *For each terminal $t \in T$, there are precisely two s-t paths P^t_1, P^t_2 in $(P_i)_{i\in[\ell]}$.*
- *$\lambda_i = \frac{1}{2} d_t$ for $i \in [\ell]$, where t is the terminal at which P_i ends.*

Then we can efficiently determine for each $t \in T$ a path $P^t \in \{P^t_1, P^t_2\}$ such that

(i) $x(a) - O(g^2 \log g) d_{\max} \leq \sum_{t\in T:\, a\in\phi(P^t)} d_t \leq x(a) + O(g^2 \log g) d_{\max}$, *where $g = \mathrm{genus}(G)$, and*

(ii) $\sum_{t\in T} c(\phi(P^t)) \leq \sum_{a\in A} c(a) \cdot x(a)$.

Note that Lemma 6.1 implies Theorem 1.9 for the special case where we can find a source-numbered s-path decomposition with precisely two paths per terminal t, each path sending $\frac{d_t}{2}$ flow. Moreover, as we discuss in Section 6.2, Theorem 1.9 can be derived through repeated applications of Lemma 6.1, to round an initial source-numbered s-path decomposition bit per bit, analogously to how we used Lemma 5.6 to round bit per bit.

The reason why we do not abstract away these successive rounding steps in one clean problem that hides geometric aspects, like WPCS in the planar case, is that we exploit topological properties of the embedding of G to prove Lemma 6.1. By only having two paths P^t_1, P^t_2 per terminal to choose from, we can interpret these two paths as one single closed curve C^t embedded on a genus g surface. More precisely, we think of C^t as the undirected version of the closed walk/curve obtained by first following P^t_1 from s to t, and then following P^t_2 in reverse direction from t to s. Because we think of C^t as an undirected closed walk, exchanging the roles of P^t_1 and P^t_2 does not change the closed walk C^t.

(Simple) closed curves, and their homotopy classes, are a classical way to formalize and work with embeddings on higher genus surfaces, extending the planar case. Two simple closed curves on a surface are in the same homotopy class if they can be continuously deformed into each other without leaving the surface. (We refer the reader

to [28, 32] for standard topological notions in this context like homotopy classes, closed curves, orientability, etc.) Higher-genus surfaces allow for a richer set of homotopy classes than lower-genus ones. In particular, on a sphere, which allows for embedding planar graphs in a non-crossing way, all simple closed curves are in the same homotopy class, which we call the *trivial homotopy class*, or the *homotopy zero class*. Even though C^t is a closed curve for each terminal t, this curve need not be simple, i.e., it may visit a vertex twice, once for each of the two paths P_1^t, P_2^t used to construct C^t. However, because the paths P_1^t and P_2^t come from a nice s-path decomposition, they do not cross, which naturally allows for assigning to the (potentially non-simple) closed curved C^t a homotopy class, which is the one obtained by slightly pulling apart the curve wherever it touches itself. We expand on this in Section 6.3.

To prove Lemma 6.1, we first consider the case where all C^t are in the same homotopy class. In this case, we can reduce our analysis to the planar setting, yielding the following result.

Lemma 6.2 *Given is an SSUF instance (G, s, T, d, x) with costs $c \in \mathbb{Q}^A_{\geq 0}$, an arc-split (H, π, ϕ) of G, and a source-numbered nice s-path decomposition $(P_i, \lambda_i)_{i\in[\ell]}$ of H with the following properties:*

- *For each terminal $t \in T$, there are precisely two s-t paths P_1^t, P_2^t in $(P_i)_{i\in[\ell]}$.*
- *$\lambda_i = \frac{1}{2}d_t$ for $i \in [\ell]$, where t is the terminal at which P_i ends.*
- *The closed curves C^t consisting of P_1^t and P_2^t, for $t \in T$, all are in the same homotopy class.*

Then we can efficiently determine for each $t \in T$ a path $P^t \in \{P_1^t, P_2^t\}$ such that

$$(i)\ x(a) - O(d_{\max}) \leq \sum_{t\in T\,:\, a\in\phi(P^t)} d_t \leq x(a) + O(d_{\max}), \text{ and}$$

$$(ii)\ \sum_{t\in T} c(\phi(P^t)) \leq \sum_{a\in A} c(a)\cdot x(a).$$

We show Lemma 6.2 in two parts, depending on whether the common homotopy class of the curves C^t is the zero homotopy (see Section 6.5 for the proof of this case) or a non-zero homotopy (see Section 6.6).

Finally, to obtain Lemma 6.1 from Lemma 6.2, we group the closed curves C^t, obtained by combing the paths P_1^t and P_2^t in Lemma 6.1, by their homotopy class, and look at each homotopy class independently, which breaks the rounding problem into independent subproblems to which we can apply Lemma 6.2. For this approach to work out, it is crucial that the closed curves C^t for $t \in T$ only have a small number of different homotopy classes. Even though for any surface of genus larger than zero, there are infinitely many homotopy classes, the following result by Greene shows that, as long as a family of simple closed curves intersects pairwise in at most one point, in this family there can only be finitely many homotopy classes. Moreover, this finite bound depends solely on the genus of the surface.

Theorem 6.3 ([13]) *Let $\mathbb{S}_g$ be an orientable surface of genus g. Then any family of simple closed curves on $\mathbb{S}_g$ that intersect pairwise in at most one point belong to at most $O(g^2 \log g)$ homotopy classes.*

We show in Section 6.4 how Theorem 6.3 combined with Lemma 6.2 readily implies Lemma 6.1.

We summarize the structure of the rest of this section, which aims to show Theorem 1.9. We first discuss in Section 6.2 how Theorem 1.9 follows from Lemma 6.1. We then focus on proving Lemma 6.1 for which we use a canonical extension of the notion of homotopy class from simple closed curves to non-crossing closed curves, which we expand on in Section 6.3. Section 6.4 then shows that it suffices to prove Lemma 6.2 to show Lemma 6.1. Finally, we prove Lemma 6.2 by distinguishing two cases, depending on the common homotopy class of the curves C^t. For this common homotopy class being the zero homotopy, we show the statement in Section 6.5, and the remaining case of a non-zero homotopy class is discussed in Section 6.6.

6.2 Obtaining Theorem 1.9 from Lemma 6.1

Obtaining Theorem 1.9 from Lemma 6.1 works analogous to how we used Lemma 5.6 to round a 2^{-k_ε} fractional WPCS solution bit per bit into an integral one. As the approach is identical here, we focus on highlighting an overview on how things carry over.

Let (G, s, T, d, x) be an SSUF instance with costs $c \in \mathbb{Q}^A_{\geq 0}$, and let $g := \text{genus}(G)$. We start by computing an arc-split (H, π, ϕ) of G and a source-numbered nice s-path decomposition $(P_i, \lambda_i)_{i\in[\ell]}$ of H with $\ell = O(|V|)$. This involves determining the embedding π, which can be done efficiently for graphs of constant genus. We remark that this is the only place where our procedure exploits that the genus is constant to ensure that we obtain an efficient algorithm. As discussed, this can be done identically to the planar case, because the non-crossing property of such a decomposition is a local property, defined by vertex, and hence carries over to graphs embedded on surfaces of higher genus. We recall that, contrary to the planar case, (H, π) is an embedding of H on a sphere with g handles, instead of being a combinatorial embedding on the plane.

For $i \in [\ell]$, let d_i be the demand of the terminal t to which the path P_i goes. To round our path decomposition $(P_i, \lambda_i)_{i\in[\ell]}$ bit by bit, we first alter it slightly, so that the fractional path selection to which it corresponds, which is given by the vectors $y_i := \frac{\lambda_i}{d_i}$, is 2^{-k} integral for some integer $k \in \mathbb{Z}_{\geq 0}$. This can be achieved by applying Lemma 5.7 to the WPCS instance that corresponds to the path selection problem for the paths $\{P_i\}_{i\in[\ell]}$ with fractional selection vector y, by incurring only an $\varepsilon d_{\max}$-violation in the capacities for $k := \lceil \log_2(\frac{\ell}{\varepsilon}) \rceil$. Hence, we assume that we have a nice source-numbered s-path decomposition $(P_i, \lambda_i)_{i\in[\ell]}$ of H with $y_i := \frac{\lambda_i}{d_i}$ being a multiple of 2^{-k}.

Let

$$F := \{i \in [\ell] \colon y_i \text{ is not } 2^{-k+1}\text{-integral}\}$$

be all indices of paths whose y-value is not 2^{-k+1}-integral.

Analogous to the proof of Lemma 5.6, we now pair up the paths corresponding to indices in F and that go to the same terminal, to build a new PSSUF instance. Whenever a pair of paths is paired up, we think of them as going to their own terminal.

Because several paths may now go to the same terminal, we can create copies of the terminals to preserve the property that terminals are distinct. Moreover, for each $t \in T$, each terminal $\bar{t}$ that is a copy of t gets a demand of $d_t 2^{-k+1}$, and the two paths going to $\bar{t}$ each get assigned a λ-value of $d_t 2^{-k}$. Applying Lemma 6.2 to this instance will pick for each $\bar{t}$ one of the two paths that go to $\bar{t}$ as $P^{\bar{t}}$, and we interpret this as rerouting $d_t 2^{-k}$ flow from the path that did not get picked to the one that got picked. Lemma 6.1 implies that this leads to a new nice source-numbered s-path decomposition $(P_i, \lambda'_i)_{i \in [\ell]}$ of the original SSUF instance, for which (i) each $y'_i := \frac{\lambda'_i}{d_i}$ is a multiple of 2^{-k+1}, (ii) the cost of the solution is no higher than the cost of the original fractional solution, and (iii) the violation of the capacities is bounded by $O(g^2 \log g) 2^{-k+1} d_{\max}$. Applying this approach repeatedly, we end up with an unsplittable flow fulfilling the guarantees of Theorem 1.9, as desired.

6.3 Homotopy classes of non-simple closed curves

Homotopy classes are typically defined with respect to *simple* closed curves, i.e., each vertex is visited at most once by the curve. We now provide more details on how the notion of homotopy class can naturally be extended to non-crossing closed curves that are not necessarily simple, which is the setting we are interested in because we combine two paths into one closed walk. Formally, an embedded closed walk is non-crossing (or non-self-crossing), if any two edge-disjoint subpaths of it do not cross. The idea of the extension is to pull apart the curve wherever it touches itself. We call this process *unfolding*.

We typically need to unfold an arc-split of an SSUF instance that is embedded on a surface of genus equal to the genus of the underlying graph. Thus, we explain the unfolding process in the context of an undirected graph $G = (V, E)$ embedded in a non-crossing way on some surface. Moreover, let $E = C_1 \cup C_2 \cup \cdots \cup C_q$ be a partition of its edges E into closed walks/curves that are non-crossing, both pairwise and also not self-crossing. Unfolding is an operation that can be applied vertex per vertex. We thus focus on a single vertex $v \in V$. Consider all edges $\delta(v)$ incident to v. For each closed walk $C \in \{C_1, \ldots, C_q\}$, we consider the edges in $C \cap \delta(v)$. These edges appear in pairs $\{e_1, e_2\} \subseteq C \cap \delta(v)$, in the sense that, when walking along C in any of the two directions, whenever C enters v through some edge $e_1 = \{u_1, v\} \in C \cap \delta(v)$, it will leave it through another edge $e_2 = \{u_2, v\} \in C \cap \delta(v)$. For each such pair of edges e_1, e_2, we introduce a new vertex w and replace e_1 and e_2 by $\{u_1, w\}$ and $\{u_2, w\}$, respectively. We illustrate in Figure 11 how this procedure can be interpreted geometrically, which shows that it naturally leads to a graph that still embeds without edge intersections on the same surface.

The embedding as shown in Figure 11 is indeed geometrically non-intersecting. This follows from the closed walks $C_1, \ldots, C_q$ being non-crossing, which implies that edge-pairs incident to v are not interleaving, i.e., for the 4 edges of any two colors in Figure 11, the colors of these edges do not alternate along the cycle.

As mentioned, the above unfolding step allows for assigning a homotopy class to non-simple non-crossing closed curves. This homotopy class is obtained by first unfolding all vertices that are visited more than once by the closed walk with the above

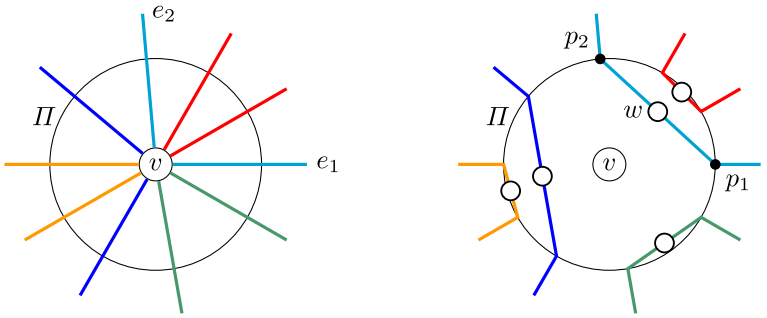


Fig. 11 One way to embed the neighborhood of an unfolded vertex v is as follows. We draw locally a circle Ξ around the vertex v. Now for any pair of edges $e_1 = \{u_1, v\}, e_2 = \{u_2, v\} \in \delta(v)$ that lie on the same closed walk $C \in \{C_1, \ldots, C_q\}$ and are traversed consecutively, we embed the vertex w and new edges $\{u_1, w\}$ and $\{u_2, w\}$ as follows. (In the above figure, we highlight such pairs of edges in the same color.) We consider the intersection point p_1 of e_1 with Ξ, and the intersection point p_2 of e_2 with Ξ. The new vertex w is placed at the midpoint of p_1 and p_2. The edge $e_1 = \{u_1, v\}$ is embedded from u_1 to p_1 like before, and then as a straight line to w. Analogously, the edge $e_2 = \{u_2, v\}$ is embedded from u_2 to p_2 like before, and then as a straight line to w

unfolding operation, and then determining the homotopy class of the resulting simple closed curve. More generally, in the context of a full SSUF instance with arc-split (H, π, ϕ) as described in Lemma 6.2, we can unfold the whole instance at once by applying the unfolding operation to all vertices except for the source and the terminals. This will simultaneously transform all closed curves $C^t = P_1^t \cup P_2^t$ into simple closed curves. We call the resulting graph the *unfolded version* of H.

6.4 Deriving Lemma 6.1 from Lemma 6.2 and Theorem 6.3

Consider the setting of Lemma 6.1. Hence, we are given an SSUF instance (G, s, T, d, x) with costs $c \in \mathbb{Q}^A_{\geq 0}$, an arc-split (H, π, ϕ) of G, and a source-numbered nice s-path decomposition $(P_i, \lambda_i)_{i \in [\ell]}$ of H with the following two properties:

- For each terminal $t \in T$, there are precisely two s-t paths P_1^t, P_2^t in $(P_i)_{i \in [\ell]}$.
- $\lambda_i = \frac{1}{2} d_t$ for $i \in [\ell]$, where t is the terminal at which P_i ends.

As previously, we denote by C^t, for $t \in T$, the closed curve obtained by combining P_1^t and P_2^t. Because the path decomposition $(P_i, \lambda_i)_{i \in [\ell]}$ is nice, the paths P_i are non-crossing. This implies that the closed curves C^t can cross each other only at the source. Hence, any pair C^{t_1}, C^{t_2} for distinct $t_1, t_2 \in T$ cross each other at most once. Thus, by Theorem 6.3, the curves C^t are in at most $O(g^2 \log g)$ different homotopy classes, where $g = \text{genus}(G)$. We now apply Lemma 6.2 separately to each homotopy class to select one path $P^t \in \{P_1^t, P_2^t\}$ for each $t \in T$ for which C^t is in that homotopy class.

More precisely, fix one homotopy class $\mathfrak{h}$ and let

$$T_{\mathfrak{h}} := \{t \in T : C^t \text{ is in homotopy class } \mathfrak{h}\}$$

be the terminals whose two paths form a closed curve in homotopy class $\mathfrak{h}$. Moreover, we define

$$I_{\mathfrak{h}} \coloneqq \{i \in [\ell] \colon P_i \text{ ends at a terminal in } T_{\mathfrak{h}}\}.$$

Restricting the given SSUF instances (G, s, T, d, x) to homotopy class $\mathfrak{h}$ leads to a new SSUF instance $(G, s, T_{\mathfrak{h}}, d_{\mathfrak{h}}, x_{\mathfrak{h}})$, where $d_{\mathfrak{h}}$ is simply the restriction of d to the terminals in $T_{\mathfrak{h}}$, and x is defined by

$$x_{\mathfrak{h}}(a) \coloneqq \sum_{i \in I_{\mathfrak{h}} \colon a \in \phi(P_i)} \lambda_i \qquad \forall a \in A.$$

In other words, x corresponds only to the load from paths in the path decomposition that go to terminals in $T_{\mathfrak{h}}$. We apply Lemma 6.2 to this new instance with the source-numbered nice s-path decomposition $(P_i, \lambda_i)_{i \in I_{\mathfrak{h}}}$. Lemma 6.2 will select paths for the terminals in $T_{\mathfrak{h}}$ that cost no more than the splittable flow given by $x_{\mathfrak{h}}$, and so that the capacity violation of any arc is at most $O(d_{\max})$. Applying the same method independently to all $O(g^2 \log g)$ homotopy classes to obtain an unsplittable flow will increase the capacity violation to at most $O(g^2 \log g) \cdot d_{\max}$, thus implying Lemma 6.1, as desired.

6.5 The homotopy zero case

In this section, we prove Lemma 6.2 in the case where the curves $\{C^t\}_{t \in T}$ are all in the zero homotopy class. The algorithm we use will be essentially the same as in the planar setting. We will establish an analogous statement to Lemma 2.14 by proving that the set of paths P_i for which $\phi(P_i)$ uses a fixed arc a of G is discrepancy-equivalent to the union of (at most) four circular intervals.

Lemma 6.4 *For every arc $a \in A$, the set $\{i \in [\ell] : a \in \phi(P_i)\}$ is discrepancy-equivalent to the union of at most four pairwise disjoint circular intervals.*

Moreover, we will prove the following non-interleaving property, which is an analogous statement to Lemma 2.3.

Lemma 6.5 *For $t \in T$ let $S^t \coloneqq \{i \in [\ell] : P_i \in \mathcal{P}^t\}$. Then the partition $\{S^t : t \in T\}$ of $[\ell]$ is non-interleaving.*

Having established Lemma 6.4 and Lemma 6.5, we proceed as in the planar setting to obtain Lemma 6.2. More precisely, we construct a WPCS instance and apply Theorem 2.11 as in the proof of Theorem 1.8. Because the discrepancies for the four disjoint circular intervals add up, we obtain

$$x(a) - 8 \cdot d_{\max} \leq \sum_{t \in T : a \in \phi(P^t)} d_t \leq x(a) + 8 \cdot d_{\max} \qquad \forall a \in A.$$

We remark that one could improve the discrepancy $8 \cdot d_{\max}$ to $4 \cdot d_{\max}$ by using the fact that, in our setting, the vector y in the WPCS instance is half-integral (in which case the proof of Theorem 2.11 can be simplified).

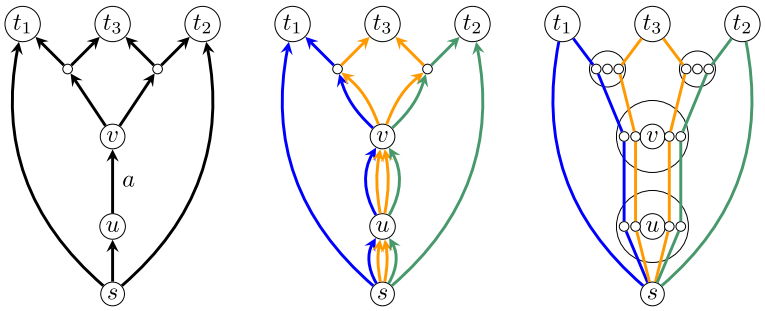


Fig. 12 An example illustrating the proof of Lemma 6.6. The left picture shows the original graph G, the middle picture the arc-split graph H, and the right picture the unfolded version of H. Here, we have $t_1, t_2 \in T^a$ and $t_3 \notin T^a$. The blue and the green curve bound the two inclusion-wise maximal discs Θ^t with $t \in T^a$

In order to complete the proof of Lemma 6.2 in the case where the curves $\{C^t\}_{t\in T}$ are all in the zero homotopy class, it now remains to prove Lemma 6.4 and Lemma 6.5. Because every curve C^t is in the zero homotopy class, it bounds a disc, i.e., when cutting the surface along C^t in the unfolded version of H, we obtain two distinct pieces with boundary C^t, one of which is a disc Θ^t. Here we denote by Θ^t an open disc with boundary C^t, i.e., the boundary is not part of Θ^t.

Lemma 6.6 *The set $\{\Theta^t : t \in T\}$ is a laminar family, i.e., for every two distinct terminals $t_1, t_2 \in T$, we have $\Theta^{t_1} \subseteq \Theta^{t_2}$, or $\Theta^{t_2} \subseteq \Theta^{t_1}$, or $\Theta^{t_1} \cap \Theta^{t_2} = \emptyset$.*

Proof This statement readily follows from the fact that the paths P_i in the nice s-path decomposition $(P_i, \lambda_i)_{i\in[\ell]}$ of H are non-crossing. Indeed, this implies that the curves C^{t_1} and C^{t_2} for two distinct terminals $t_1, t_2 \in T$ do not cross at any other vertex than possibly the source s. Because the curves C^{t_1} and C^{t_2} limit discs (namely Θ^{t_1} and Θ^{t_2}), they need to cross an even number of times. As they can only cross at most once (at the vertex s), this implies that they do not cross whatsoever, and thus $\Theta^{t_1} \subseteq \Theta^{t_2}$, or $\Theta^{t_2} \subseteq \Theta^{t_1}$, or $\Theta^{t_1} \cap \Theta^{t_2} = \emptyset$, as desired. □

We now use Lemma 6.6 to prove Lemma 6.4 and Lemma 6.5.

Lemma 6.4 *For every arc $a \in A$, the set $\{i \in [\ell] : a \in \phi(P_i)\}$ is discrepancy-equivalent to the union of at most four pairwise disjoint circular intervals.*

Proof Consider the set $T^a \subseteq T$ of terminals t for which exactly one of the two paths $\phi(P_1^t)$ and $\phi(P_2^t)$ contains the arc a. Then the set $\{i \in [\ell] : a \in \phi(P_i)\}$ is discrepancy-equivalent to the set $\{i \in [\ell] : a \in \phi(P_i),\ P_i \text{ ends in a terminal } t \in T^a\}$. We will prove that the latter set is discrepancy-equivalent to the union of (at most) four pairwise disjoint circular intervals.

Consider the (inclusion-wise) maximal discs Θ^t with $t \in T^a$. By Lemma 6.6, these maximal discs are pairwise disjoint. By the definition of T^a, there are at most two such maximal discs Θ^{t_1} and Θ^{t_2} with $t_1, t_2 \in T^a$. See Figure 12.

For each such maximal disk Θ^{t_j} with $j \in \{1, 2\}$, we consider the terminal set $T^j := \{t \in T : \Theta^t \subseteq \Theta^{t_j}\}$. Every terminal t for which exactly one of the two paths

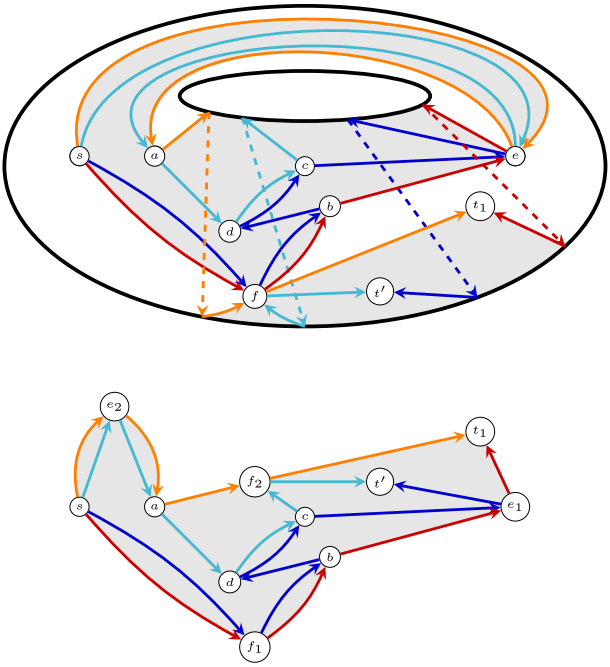


Fig. 13 Example of the cutting procedure we apply to the discs Θ^{t_j} for $j \in \{1, 2\}$, here illustrated for disc Θ^{t_1}. The top part of the figure shows four paths (red, orange, light blue, and dark blue) embedded on a torus, where edges on the back of the torus are drawn dashed. The red and orange s-t_1 path bound the disc Θ^{t_1} (in the unfolded version of H), which is indicated in gray. The bottom part of the figure shows the planar SSUF instance we obtain after cutting along the curve C^{t_1} (corresponding to orange and red edges). In the planar instance on the bottom, the two vertices e_1 and e_2 correspond to the vertex e on the top, and similarly f_1 and f_2 resulted from f by cutting along the curve C^{t_1}. We remark that the cutting step is necessary to obtain a planar SSUF instance because the restriction of H to Θ^{t_1} (including the boundary C^{t_1}) is not necessarily planar. Indeed, the graph in the top figure contains the complete bipartite graph on the two vertex sets $\{a, b, c\}$ and $\{d, e, f\}$ as a subgraph and is therefore not planar

$\phi(P_1^t)$ and $\phi(P_2^t)$ contains the arc a, is contained in exactly one of these terminal sets T^1 or T^2. It therefore suffices to prove that for $j \in \{1, 2\}$, the set $\{i \in [\ell] : a \in \phi(P_i), P_i \text{ ends in a terminal } t \in T^j\}$ is discrepancy equivalent to the union of at most two circular intervals.

To this end, we will use Lemma 2.14. Let $j \in \{1, 2\}$. We construct a planar instance of the SSUF problem as follows. We consider the unfolded version of the graph H and cut the surface along the boundary of the disc Θ^{t_j}. We obtain the disc Θ^{t_j} with the curve C^{t_j} embedded on the boundary. Note that we might have cut (potentially more than once) through the neighborhood Ξ of a vertex v considered in an unfolding operation as illustrated in Figure 11. We undo the unfolding of H, by contracting each of the pieces of Ξ that arose from cutting along C^{t_j}. See Figure 13 for an example.

The resulting graph is the arc-split graph of a planar SSUF instance with a nice source-numbered s-path partitioning inherited from the original instance. To this instance we apply Lemma 2.14. Because the curve C^{t_j} along which we cut contains the arc a only once, this SSUF instance contains an arc a' with the following

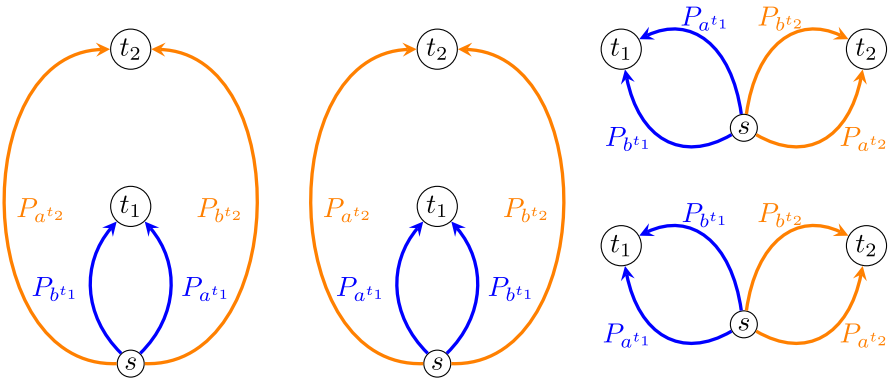


Fig. 14 Illustration of the proof of Lemma 6.5. Possible situations if $\Theta^{t_1} \subseteq \Theta^{t_2}$ or $\Theta^{t_1} \cap \Theta^{t_2} = \emptyset$. The case $\Theta^{t_2} \subseteq \Theta^{t_1}$ is not shown here (but it is similar to the case $\Theta^{t_1} \subseteq \Theta^{t_2}$)

property: a path P_i in the new planar instance contains the arc a' if and only if the corresponding path P_i contained the arc a in the original SSUF instance.

Observe that the planar instance we constructed contains only a subset of the paths from the original instance, namely the set $\{P_i : i \in I\}$ for some circular interval I. Thus, the circular interval in the planar instance that we obtain from Lemma 2.14 corresponds to up to two circular intervals of the original SSUF instance. □

Lemma 6.5 *For $t \in T$ let $S^t := \{i \in [\ell] \colon P_i \in \mathcal{P}^t\}$. Then the partition $\{S^t : t \in T\}$ of $[\ell]$ is non-interleaving.*

Proof For every $t \in T$, the set S^t has exactly two elements a^t and b^t, where we may assume $a^t < b^t$. Now consider two distinct terminals t_1 and t_2, where we may assume $a^{t_1} < a^{t_2}$. By Lemma 6.6, the corresponding discs Θ^{t_1} and Θ^{t_2} satisfy $\Theta^{t_1} \subseteq \Theta^{t_2}$, or $\Theta^{t_2} \subseteq \Theta^{t_1}$, or $\Theta^{t_1} \cap \Theta^{t_2} = \emptyset$. In each of these cases, we have $a^{t_1} \leq b^{t_1} \leq a^{t_2} \leq b^{t_2}$ or $a^{t_1} \leq a^{t_2} \leq b^{t_2} \leq b^{t_1}$ because the paths $(P_i)_{i \in [\ell]}$ are source-numbered. See Figure 14. □

This completes the proof of Lemma 6.2 in the case where the curves $\{C^t\}_{t \in T}$ are all in the zero homotopy class.

6.6 The homotopy non-zero case

The sake of this section is to show Lemma 6.2 in the case where the curves $\{C^t\}_{t \in T}$ are all in the same non-zero homotopy class.

The idea is again to use the algorithm for the planar case. For curves that are freely homotopic but not homotopic to zero, a result by Levine [21] allows for choosing two of these curves so that they bound a cylinder so that all other curves lie on that cylinder. Since a graph embedded on a cylinder is planar, we can apply our results for planar graphs to obtain the desired lemma. However, some care is needed with this approach. To reduce the setting to the cylinder we have to unfold the graph. On the cylinder we cannot just undo the unfolding, since it would destroy the structure of the

cylinder. Instead, we have to carefully split up some vertices without producing too many copies of a vertex, to be able to fold the graph back at least partly. This is similar to the proof of Lemma 6.4, but in contrast to the homotopy-zero case, we now use the cutting procedure in the algorithm and not only in the analysis.

We formalize the procedure below in Lemma 6.11, which we call the *planarization lemma*. Given any set of curves that are freely homotopic but not homotopic to zero, the planarization lemma maps these curves to a planar graph while preserving key properties of the curves and copying vertices only a bounded number of times, where the bound depends on how often vertices are visited by the same curve C^t for $t \in T$. To formalize the notion of how many times a curve can go through the same vertex, we use the notion of p-simple curves.

Definition 6.9 *(p-simple)* A curve is p-simple if it is a closed walk traversing each vertex at most p times and using each edge at most once.

Note that each curve C^t is 2-simple, as it consists of two paths P_1^t and P_2^t. (We recall that paths are by definition simple, i.e., they visit each vertex at most once.)

A k-vertex split describes the operation of splitting the vertices of the graph into up to k copies.

Definition 6.10 *(k-vertex split)* Let $G = (V, E)$ be an undirected graph embedded on some surface with embedding π. A k-vertex split $(\overline{G}, \psi, \overline{\pi})$ of (G, π) is a graph $\overline{G}$ with an embedding $\overline{\pi}$ for the same surface together with a map ψ satisfying the following:

- Each vertex $v \in V$ is replaced by up to k copies in $\overline{V}$.
- There is a one-to-one correspondence between edges in G and edges in $\overline{G}$. Moreover, an edge $e = \{v, w\}$ in G corresponds to an edge in $\overline{G}$ between a copy of v and a copy of w.
- The map $\psi : \overline{V} \to V$ maps a copy of a vertex back to its original.
- For each $\overline{v} \in \overline{V}$, in the combinatorial embedding $\overline{\pi}_{\overline{v}}$ is a $\pi_{\psi(\overline{v})}$-progression.

We also call $\overline{G}$ a *k-vertex split graph* of G.

Because of the one-to-one correspondence between edges in G and edges in a k-vertex split graph $\overline{G}$ of G, we denote both edge sets by the same letter E and identify the edges in G with the corresponding edges in $\overline{G}$. Moreover, note that a k-vertex split of a graph is clearly not unique, and G is a k-vertex split graph of itself for any $k \in \mathbb{Z}_{\geq 1}$. See Figure 15 for an example of an embedded graph and a 2-vertex split thereof. We now state the *planarization lemma*, which is exemplified in Figure 15.

Lemma 6.11 (Planarization lemma) *Given is an undirected graph $H = (V, F)$ with an embedding on a compact, orientable surface, an integer $p \in \mathbb{Z}_{\geq 1}$, and a partitioning of $F = C_1 \cup \ldots \cup C_q$ such that*

(i) C_i is a p-simple curve for all $i \in [q]$,
(ii) C_i's are non-crossing (neither mutual nor self-crossing),
(iii) all C_i's are in the same non-zero homotopy class.

Then one can efficiently determine a planar $2p$-vertex split $(\overline{H} = (\overline{V}, F), \psi, \pi)$ of H with the following properties. For each $i \in [q]$, the set C_i is a simple closed curve in $\overline{H}$. Moreover, all $\{C_i\}_{i \in [q]}$ remain non-crossing in $\overline{H}$.

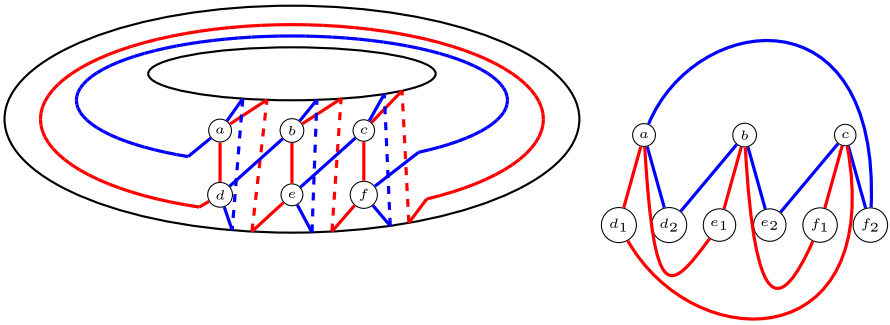


Fig. 15 The graph in the first picture is non-planar as it contains $K_{3,3}$ as a subgraph. It consists of two 1-simple curves, marked in blue and red. Thus by the planarization lemma, there is a 2-vertex split graph that is planar. Such a graph is depicted on the right. While the choice is not unique, this is the graph that would be computed by our procedure

Before we discuss the proof of the lemma, let us show that it allows for proving Lemma 6.2 in the case where the homotopy class of the curves is non-zero.

Proof of Lemma 6.2 for a non-zero homotopy class We first show that we can apply Lemma 6.11 with $p = 2$ to the curves $\{C^t\}_{t\in T}$. To do so, first observe that, by construction, all curves $\{C^t\}_{t\in T}$ in H are arc-disjoint and moreover, every arc belongs to a curve C^t. By definition, all curves are in the same non-zero homotopy class and each curve is 2-simple since it consists of two simple paths. Since the curves are constructed from non-crossing paths, the curves $\{C^t\}_{t\in T}$ are not self-crossing and can cross other curves only at s. Hence, curves can cross at most once. Since curves in the same non-zero homotopy class always cross an even number of times, see, e.g., [12], the curves $\{C^t\}_{t\in T}$ are non-crossing.

Thus, all conditions of Lemma 6.11 are fulfilled, and we can apply the lemma to the graph $H = (V, F)$ and its embedding when disregarding arc directions and to the curves $\{C^t\}_{t\in T}$ to obtain a planar 4-vertex split $(\overline{H} = (\overline{V}, F), \psi, \pi)$, where C^t for $t \in T$ remain simple closed curves that are mutually non-crossing. Even though we disregard the arc directions to apply Lemma 6.11—to be formally in an undirected setting as required by the lemma—we consider the graph $\overline{H}$ as being directed by simply keeping the same arc directions as before the $2p$-vertex split operation. In short, an arc $(u, v) \in F$ will be mapped to an arc $(\psi(u), \psi(v))$ in $\overline{H}$. This and also the following graph transformations are depicted in Figure 16.

Since $\overline{H}$ is a 4-vertex split graph of H, there are at most four copies of s in $\overline{V}$. To be able to apply our results for the planar single-source case, we consider four independent instances, one per copy of s. Let s_i be the i-th copy of s for $i \in [4]$; we create an instance for each index $i \in [4]$. (For simplicity of exposition, we assume that there are exactly 4 copies of s; the case of fewer copies is analogous.) The instance i consists of all curves C^t that go through s_i. These curves do not go through s_j for $j \neq i$ because C^t has only two edges incident to s in H and these are incident to s_i in $\overline{H}$; moreover, no other edge can be incident to any copy of s in $\overline{H}$ as each edge in $\overline{H}$ connects copies of the same vertices it connects in H. Thus, the curves are partitioned

between the four instances. We partition the terminals accordingly and denote by T_i the terminals of instance $i \in [4]$.

Fix some $i \in [4]$ and let $\overline{H}_i = (\overline{V}, F_i)$ be the graph that consists only of the arcs used by curves C^t with $t \in T_i$. This means that F is the disjoint union of F_i for $i \in [4]$.

Assume w.l.o.g. that G has no parallel arcs. Next, we define for each $i \in [4]$ a planar graph $G_i = (\overline{V}, A_i)$ obtained from $\overline{H}_i = (\overline{V}, F_i)$ by merging parallel arcs into a single one. We will argue that parallel arcs in $\overline{H}_i$ appear consecutively in the combinatorial embedding of $\overline{H}_i$. This implies that the above procedure induces a natural combinatorial embedding of G_i. Moreover, this way, $\overline{H}_i = (\overline{V}, F_i)$ together with the map ϕ_i that maps parallel arcs to their merged counterpart, is an arc-split of G_i. Note that this also makes sure that there are no parallel arcs in G_i.

We argue now that parallel arcs appear consecutively in the combinatorial embedding of $\overline{H}_i$. Since H is an arc-split of G, any set of parallel arcs appears consecutively in the combinatorial embedding of H. (Here, we use that G has no parallel arcs.) The same holds for the vertex split graph $\overline{H}$, as any set of arcs from v and w in $\overline{H}$ connects the vertices $\phi(v)$ and $\phi(w)$ in H. Moreover, because the combinatorial embedding at v is a subsequence of the combinatorial embedding at $\phi(v)$ and the same is true for w, these arcs remain in consecutive order. This order is also maintained when $\overline{H}$ is split up into $\overline{H}_i$ for $i \in [4]$.

As a next step, we define ξ_i to be the canonical mapping from the arcs A_i of $G_i = (\overline{V}, A_i)$ to A, i.e., $\xi_i : A_i \to A$ for $i \in [4]$, where an arc $a_i \in A_i$ gets mapped to a if there is an arc $f_i \in F_i$ with $\phi_i(f_i) = a_i$ and $\phi(\psi(f_i)) = a$. Now, we define a flow x_i on A_i for $i \in [4]$ such that $\sum_{i\in[4]} \sum_{\widetilde{a}\in A_i\colon \xi_i(\widetilde{a})=a} x_i(\widetilde{a}) = x(a)$ for all arcs $a \in A$. For this sake, we assume again that $G = (V, A)$ has no parallel arcs. In the following, we use the notation $d(P)$, for $P \in \mathcal{P}$, to denote the demand d_t where t is the terminal where P ends. We define $x_i(\widetilde{a}) \coloneqq \sum_{f\in F_i\colon \phi_i(f)=\widetilde{a}} \sum_{P\in\mathcal{P}^i\colon f\in P} \frac{1}{2} d(P)$ for $\widetilde{a} \in A_i$. To show that we have the desired property, observe that

$$
\begin{aligned}
x(a) &= \sum_{\substack{f\in F\colon \\ \phi(f)=a}} \sum_{\substack{P\in\mathcal{P}\colon \\ f\in P}} \frac{1}{2} d(P) \\
&= \sum_{i\in[4]} \sum_{\substack{f\in F_i\colon \\ \phi(f)=a}} \sum_{\substack{P\in\mathcal{P}^i\colon \\ f\in P}} \frac{1}{2} d(P) \\
&= \sum_{i\in[4]} \sum_{\substack{\widetilde{a}\in A_i\colon \\ \xi_i(\widetilde{a})=a}} \sum_{\substack{f\in F_i\colon \\ \phi_i(f)=\widetilde{a}}} \sum_{\substack{P\in\mathcal{P}^i\colon \\ f\in P}} \frac{1}{2} d(P) \\
&= \sum_{i\in[4]} \sum_{\substack{\widetilde{a}\in A_i\colon \\ \xi_i(\widetilde{a})=a}} x_i(\widetilde{a}).
\end{aligned}
$$

Also note that $|\{\widetilde{a} \in A_i\colon \xi_i(\widetilde{a}) = a\}| \leq 16$ for every $a \in A$. The reason is that there are no parallel arcs in A_i by construction and every arc $\widetilde{a}$ with $\xi_i(\widetilde{a}) = a$ connects copies of the start and end vertex of a. As there are at most 4 copies of a vertex, the bound follows.

Let $c(\widetilde{a}) \coloneqq c(a)$ for $a \in A$ with $\xi_i(\widetilde{a}) = a$. Now we have four PSSUF instances $(G_i, s_i, T_i, d_i, x_i)$, where d_i is just the demand vector d restricted to elements in T_i. Note that we cannot just apply Theorem 1.8 as a black box to these instances. The issue is that a black box application would not guarantee that the selected paths come from the set $\{P_1^t, P_2^t\}$. Instead, we follow along the lines of the proof of Theorem 1.8. To do so, let I_i be the index set of all indices j corresponding to paths P_j with a terminal in T_i. As $(P_j, \lambda_j)_{j \in I_i}$ is a nice s_i-path decomposition of $\overline{H}_i$ by definition of x and $\overline{H}_i$, we can construct a WPCS instance as in the proof of Theorem 1.8. Applying Theorem 2.11 yields a selection in the WPCS instance. This translates into a selection of paths per subinstance $\mathcal{P}^i_{\text{sel}} = \{P^t\}_{t \in T_i}$, such that the properties of Theorem 1.8 are fulfilled, and, additionally, the selection fulfills $P^t \in \{P_1^t, P_2^t\}$ for all $t \in T_i$.

Finally, we observe that such a selection directly induces a selection of paths $\mathcal{P}_{\text{sel}} = \cup_{i \in [4]} \mathcal{P}^i_{\text{sel}} = \{P^t\}_{t \in T}$ in the whole instance, which satisfies as well that $P^t \in \{P_1^t, P_2^t\}$ for all $t \in T$. Moreover, these paths satisfy

(i)

$$\begin{aligned} |x(a) - \text{flow}_{\mathcal{P}_{\text{sel}}}(a)| &= \left| \sum_{i \in [4]} \sum_{\substack{\widetilde{a} \in A_i : \\ \xi_i(\widetilde{a}) = a}} \left(x_i(\widetilde{a}) - \text{flow}_{\mathcal{P}^i_{\text{sel}}}(\widetilde{a}) \right) \right| \\ &\leq 4 \cdot 16 \cdot 2 d_{\max} = 128 d_{\max}, \quad \text{and} \end{aligned}$$

(ii)

$$\begin{aligned} \sum_{a \in A} c(a)(x(a) - \text{flow}_{\mathcal{P}_{\text{sel}}}(a)) &= \sum_{a \in A} c(a) \sum_{i \in [4]} \sum_{\substack{\widetilde{a} \in A_i : \\ \xi_i(\widetilde{a}) = a}} \left(x_i(\widetilde{a}) - \text{flow}_{\mathcal{P}^i_{\text{sel}}}(\widetilde{a}) \right) \\ &= \sum_{i \in [4]} \sum_{a \in A} c(a) \sum_{\substack{\widetilde{a} \in A_i : \\ \xi_i(\widetilde{a}) = a}} \left(x_i(\widetilde{a}) - \text{flow}_{\mathcal{P}^i_{\text{sel}}}(\widetilde{a}) \right) \\ &= \sum_{i \in [4]} \sum_{\widetilde{a} \in A_i} c(\widetilde{a})(x_i(\widetilde{a}) - \text{flow}_{\mathcal{P}^i_{\text{sel}}}(\widetilde{a})) \leq 0. \end{aligned}$$

This shows that the selected paths fulfill the desired properties. Lastly, we observe that the process of selecting the paths is clearly efficient. Lemma 6.11 efficiently determines a 4-vertex split. Moreover, we can construct the subinstances and the WPCS instances by considering every curve once and also Theorem 2.11 returns a selection of paths efficiently (and we need this four times). □

In the rest of this Section 6.6, we prove Lemma 6.11. As a preparation, we show the following lemma, which is a consequence of a result by Levine and will allow us to restrict our attention to curves on a well-chosen cylinder.

Lemma 6.12 (Cylinder lemma) *Let $C_1, \dots C_q$, for $q \in \mathbb{Z}_{\geq 2}$, be simple, non-intersecting closed curves in the same non-zero homotopy class embedded in some compact, orientable surface. Then there exist distinct indices $i, j \in [q]$ such that C_i*

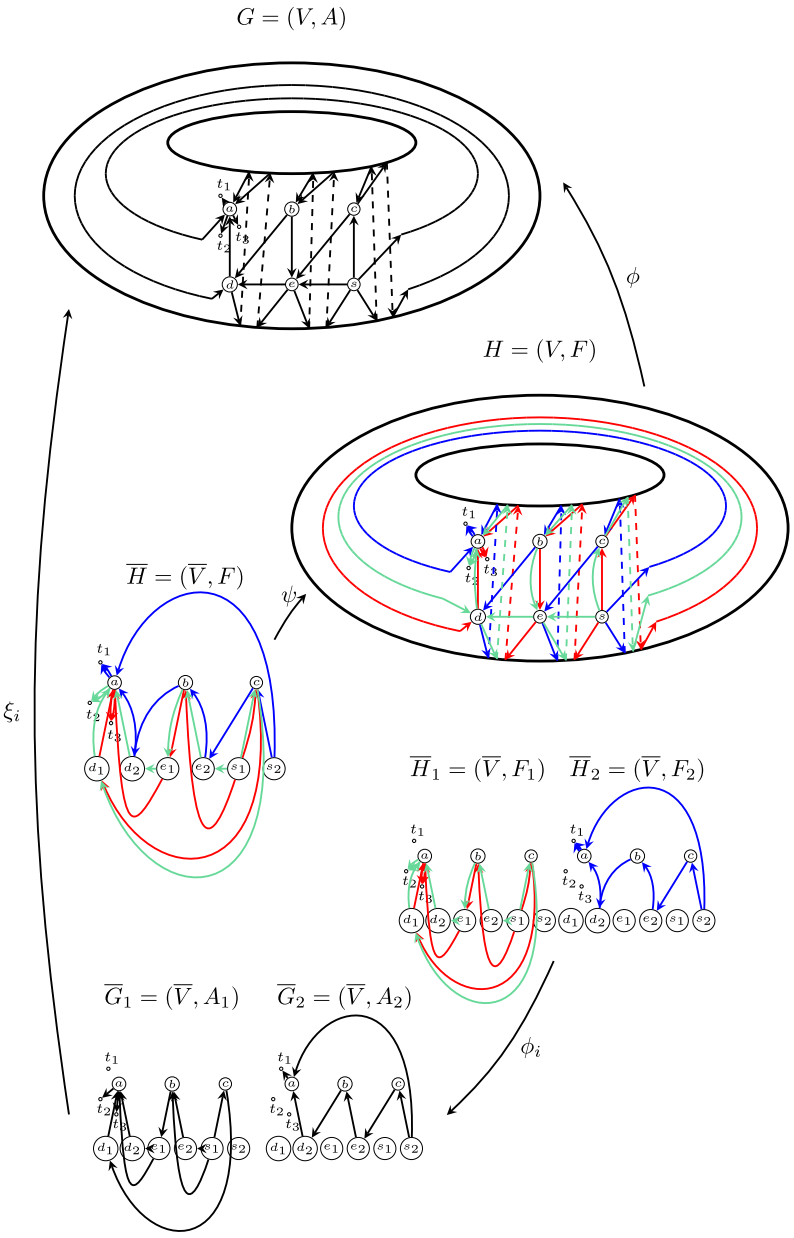


Fig. 16 In the figure we consider the different transformations applied to the graph $G = (V, A)$ throughout the proof. We start with a graph that is embedded on some bounded genus surface. In a first step we consider its edge split graph $H = (V, F)$. By application of Lemma 6.11 we obtain a 4-vertex split graph $\overline{H} = (\overline{V}, F)$ of $H = (V, F)$. Next, we consider the subinstances which are induced by the up to four copies of s. Here we have just two, $\overline{H}_i = (\overline{V}, F_i)$ for $i \in \{1, 2\}$. Lastly, we reverse the edge splitting operation in the canonical way to obtain the graphs $\overline{G}_i = (\overline{V}, A_i)$ for $i \in \{1, 2\}$. Note that while in the actual construction we assume that there are no parallel arcs in G, we do have some parallel arcs in our example for simplicity of exposition. Such parallel arcs can easily be avoided by subdividing them through the introduction of artificial nodes

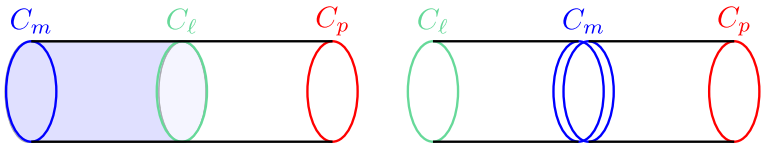


Fig. 17 In the figure we see case 2 of the proof of Lemma 6.12. On the left, C_ℓ is in the cylinder Z formed by C_m and C_p. By case distinction, we conclude that also the colored cylinder $\text{cyl}(C_\ell, C_m)$ is part of Z. On the right side on the other hand, C_ℓ is not part of Z, in this case we can glue two cylinders together at C_m to obtain a cylinder bounded by C_ℓ and C_p that satisfies all requirements of the lemma

and C_j bound a cylinder that contains $\cup_{k=1}^{q} C_k$. Moreover, i and j can be determined efficiently.

To prove the cylinder lemma, we use a result by Levine [21].

Theorem 6.13 ([21]) *Consider a compact, orientable surface W. Let A and B be two simple, closed, non-intersecting curves on W, such that A and B are in the same non-zero homotopy class. Then $A \cup B$ bounds a cylinder in W.*

Proof of Lemma 6.12 We show the statement by induction on q. In case of $q = 2$, the statement follows directly from Theorem 6.13.

For the induction step, assume the statement is true for $q \leq p - 1$. Now, we extend it to $q = p$. Let $\ell, m \in [p-1]$ be two distinct indices such that C_ℓ and C_m form a cylinder which contains $\cup_{k=1}^{p-1} C_k$. These indices exist by induction, and we call the respective cylinder $\text{cyl}(C_\ell, C_m)$. Now distinguish two cases.

Case 1: $C_p \in \text{cyl}(C_\ell, C_m)$ In this case, we are done as we can choose $i \coloneqq \ell$ and $j \coloneqq m$, and then $\text{cyl}(C_\ell, C_m)$ is a cylinder formed by C_i and C_j containing all the other curves.

Case 2: $C_p \notin \text{cyl}(C_\ell, C_m)$ By Theorem 6.13, there is also cylinder $\text{cyl}(C_m, C_p)$ formed by C_m and C_p. We now distinguish between whether C_ℓ is contained in $\text{cyl}(C_m, C_p)$ or not. (See also Figure 17 for an illustration of the two cases.)

Assume first that C_ℓ is contained in $\text{cyl}(C_m, C_p)$. If we consider C_ℓ on the cylinder $\text{cyl}(C_m, C_p)$, we are in a planar setting. So C_ℓ cuts $\text{cyl}(C_m, C_p)$ into two parts. Moreover, because C_ℓ is in the same homotopy class as C_m and C_p, one part contains C_m and the other one C_p. By the assumption of the case distinction, we know that $\text{cyl}(C_\ell, C_m)$ does not contain C_p. Thus, the cylinder $\text{cyl}(C_\ell, C_m)$ has to lie on the side of C_ℓ that faces C_m. Even though this may seem obvious from Figure 17, it is important to note that two curves in the same homotopy class may define 2 different cylinders. (Think of a torus that gets cut twice.) Hence, in other words, the cylinder $\text{cyl}(C_\ell, C_m)$ is contained in the cylinder $\text{cyl}(C_m, C_p)$. In this case we are done, since this implies that the choice $i \coloneqq m$, $j \coloneqq p$ with $\text{cyl}(C_m, C_p)$ satisfies the desired properties.

So assume $C_\ell \notin \text{cyl}(C_m, C_p)$. Then, $\text{cyl}(C_\ell, C_m) \cap \text{cyl}(C_m, C_p) = C_m$, and thus C_ℓ and C_p bound the cylinder $\text{cyl}(C_\ell, C_m) \cup \text{cyl}(C_m, C_p)$, which contains all other curves by definition, i.e, set $i \coloneqq \ell$ and $j \coloneqq p$.

Note that each step in the induction can be executed efficiently, and thus we can efficiently determine indices i and j for a given set of curves. □

We finally prove Lemma 6.11.

Proof of Lemma 6.11 We start by considering the unfolded version of H as described in Section 6.3. In Figure 11, a geometric interpretation of the unfolding procedure is given, which will be needed in the following. More explicitly, during the unfolding we define for every vertex v a small circle Ξ_v. Since the surface is locally planar this is always possible. It is needed that Ξ_v is topologically seen a circle, not necessary in a geometric sense. We will define Ξ_v small enough, such that the combinatorial embedding at v corresponds to the order in which the edges intersect Ξ_v. During the unfolding we keep all the curves outside of $\{\Xi_v\}_{v\in V}$ fixed, inside $\{\Xi_v\}_{v\in V}$ curves are modified as described in Section 6.3. This implies in particular that inside the circles, all curves are crossing-free and each node has a degree of two, where adjacent edges belong to the same curve.

This leads to a situation where all curves C_i for $i \in [q]$ are simple. Since unfolding preserves that the curves are non-crossing and in the same non-zero homotopy class, we can apply Lemma 6.12 to choose two distinct curves A and B from $\{C_i\}_{i\in[q]}$, such that all other curves lie on the cylinder formed by A and B.

This cylinder is still embedded on some surface. To isolate the cylinder as a separate object, we perform a cutting operation as follows. We direct A and B in an arbitrary direction. Now, each curve has a side facing the cylinder (inside) and a side not facing the cylinder (outside). We cut on the outside along both curves. In other words, we obtain the cylinder as an isolated object, where both curves A and B still belong to the cylinder and will form its boundaries.

Note that each time we cut through a circle Ξ_v for some $v \in V$, this means that we split the circle into two new circles (in a topological sense). Both of these circles are bounded by an arc of the old circle and the curve along which we cut (see also Figure 18 for a geometric interpretation). In other words, while cutting we create a new set of circles $\overline{\Xi}_v$ for new vertices $v \in \overline{V}$. This set satisfies two properties. First, the number of new circles that have been created from a circle Ξ_v is bounded by $2p + 1$. This follows since each of the two curves passes at most p times through the circle Ξ_v. Moreover, there is at least one new circle that is bounded by an outside side of a curve and can thus not contain any curves (all curves lie inside the cylinder). In other words, this circle is irrelevant, and we discard it. Thus, in total we split a circle Ξ_v in at most $2p$ relevant new circles.

The second important property is that each new circle $\overline{\Xi}_v$ for $v \in \overline{V}$ is, topologically seen, a circle on the cylinder. (Note that this is not true for the original circles $\{\Xi_v\}_{v\in V}$.)

We use these two properties to undo the unfolding with respect to the new circles. This means for every new circle $\{\overline{\Xi}_v\}_{v\in\overline{V}}$ we create a vertex v. All curves that pass through the circle are modified smoothly in such a way that they pass through the new vertex v without introducing any crossings. This is possible as we consider a circle. More formally, modify the circle to a geometric circle and place the vertex v in its center. Now, we modify the edges in such a way, that when an edge enters the circle it is extended straight to the center of the circle, i.e., vertex v. The artificial vertices that were introduced during the unfolding are removed. This geometric reasoning shows that the new combinatorial embedding at v corresponds to a progression of the combinatorial embedding at the original vertex before the cutting operation was

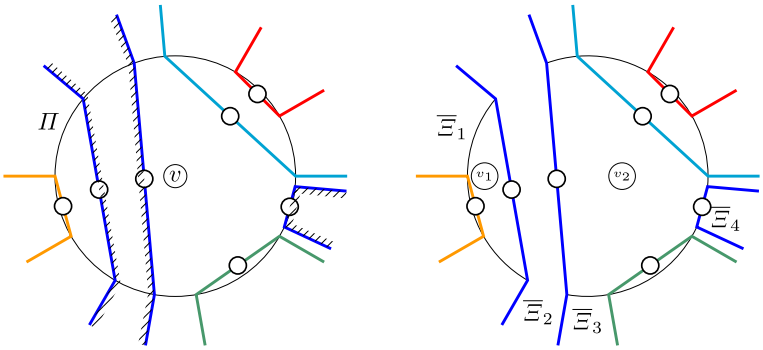


Fig. 18 A cutting operation along the blue (directed) curve. The hashed side marks the *outside* side of the curve. There are two parts that correspond to a vertex without any edges, namely $\overline{\Xi}_2$ and $\overline{\Xi}_4$. Those can be discarded and thus, there is no vertex introduced in these parts

executed. Note that with this procedure we introduce one vertex per new circle, which means we create up to $2p$ copies of each vertex.

Moreover, observe that the structure of the curves is preserved. This means, all curves remain closed as the curves are non-crossing. Since curves are non-crossing before the transformation and the cutting is performed along one of the curves, we guarantee that we never cut through a curve and destroy the property that curves are closed. The curves remain non-crossing as the embedding is not modified outside the region $\bigcup_{v \in V} \Xi_v$. Each circle Ξ_v is just split up in such a way that it contains a subset of the curves of Ξ_v which are not modified in their position and thus remain crossing free.

Like this, we obtain a graph where each vertex was copied at most $2p$ times and that lies on a cylinder and is thus planar. The mapping ψ is naturally defined by the described procedure, as during the cutting, one can keep track of the original vertex. Finally, we observed that the combinatorial embedding at v is a $\pi_{\phi(v)}$-progression by construction. Note that the cutting operation and the construction of ψ can be performed efficiently. □

7 Concluding remarks

Recall that we required all numbers in our SSUF instance to be rational, so that we have finite input length, which puts us in the traditional computational model to talk about efficient algorithms. We remark that even when allowing arbitrary real numbers (for the given flow x, the demands d, and the costs c), the existence of an unsplittable flow as in Theorem 1.7, Theorem 1.8, and Theorem 1.9 follows from our results. This can be derived from Theorem 1.7, Theorem 1.8, and Theorem 1.9 by a standard continuity argument.

Moreover, we highlight that the capacity violation $d_{\max}$ of Theorem 1.7, which shows Conjecture 1.5 in the special case of planar graphs, is tight. This is the case even if there is only a single terminal, as observed already in earlier work [6, 25]. To

see this, consider an instance with a single terminal t with demand k and a flow x that can be decomposed into k arc-disjoint paths with flow value 1 each. Then $x(a) \leq 1$ for all $a \in A$, but any unsplittable flow $\mathcal{P}$ will have value $k = d_{\max}$ on some s-t path. [25] gives a similar example showing that the lower bound on the unsplittable flow in Conjecture 1.5 is tight, even in planar graphs.

In order to prove Conjecture 1.4 for planar instances, one would need to strengthen Theorem 1.8 by improving the bound on the deviation of the unsplittable flow $\mathcal{P}$ and the given flow x, i.e., $|x(a) - \text{flow}_{\mathcal{P}}(a)|$, from $2d_{\max}$ to $d_{\max}$. We expand on two natural strategies toward this and provide examples showing why these strategies do not work.

One natural strategy would be to strengthen the interval-discrepancy bound of Theorem 2.11 from $2d_{\max}$ to $d_{\max}$. In the proof of Theorem 2.11 we showed that we can achieve a discrepancy bound of $d_{\max}$ for prefix intervals $[i]$ with $i \in [\ell]$ (see Theorem 5.4). One might hope to strengthen this to a bound of $\frac{d_{\max}}{2}$, as we did in the setting without costs (see Theorem 5.1). However, in the setting with costs such a strengthening is impossible, as the following example shows. Let $\varepsilon \in (0, 1]$ and consider the WPCS instance with $\ell = 2$, $\mathcal{S} = \{\{1, 2\}\}$, demand $d_{\{1,2\}} = 1$, and fractional selection y with $y_1 = \varepsilon$ and $y_2 = 1 - \varepsilon$. If the cost vector c fulfills $c_1 = 0$ and $c_2 = 1$, then the only integral selection z with $c^T z \leq c^T y = 1 - \varepsilon$ is the selection z with $z_1 = 1$ and $z_2 = 0$. Then $D_{y,z}([1]) = 1 - \varepsilon$. Because $d_{\max} = 1$, and ε can be chosen arbitrarily close to zero, this shows tightness of Theorem 5.4.

Another possible strategy to prove Conjecture 1.4 for planar SSUF by strengthening Theorem 2.11 would be to use the reduction to the half-integral setting described in Section 5.2, which comes at the cost of loosing a factor of two in the discrepancy bound. One might then hope to achieve a (y, z)-interval-discrepancy bound of $\frac{d_{\max}}{2}$ for the half-integral special case. However, this is in general impossible as the following example shows. Let $\ell := 6$ and and $\mathcal{S} := \{\{1, 6\}, \{2, 5\}, \{3, 4\}\}$. Note that $\mathcal{S}$ is non-interleaving. Moreover, we define $d_{\{1,6\}} := 2$, $d_{\{2,5\}} := 1$, $d_{\{3,4\}} := 2$ and $y_i := \frac{1}{2}$ for all $i \in [\ell]$. Then

$$y_1^d = 1 \quad y_2^d = 0.5 \quad y_3^d = 1 \quad y_4^d = 1 \quad y_5^d = 0.5 \quad y_6^d = 1$$

and $d_{\max} = 2$. Now consider an integral selection z. By symmetry, we may assume $z_1 = 1$ and $z_6 = 0$. If $z_3 = 1$ and $z_4 = 0$, we have

$$z_1^d = 2 \quad z_2^d \geq 0 \quad z_3^d = 2 \quad z_4^d = 0 \quad z_5^d \leq 1 \quad z_6^d = 0$$

and thus $D_{y,z}(\{1, 2, 3\}) \geq 1.5 > \frac{d_{\max}}{2}$. Otherwise, we have $z_3 = 0$ and $z_4 = 1$. Then we either have

$$z_1^d = 2 \quad z_2^d = 1 \quad z_3^d = 0 \quad z_4^d = 2 \quad z_5^d = 0 \quad z_6^d = 0$$

or

$$z_1^d = 2 \quad z_2^d = 0 \quad z_3^d = 0 \quad z_4^d = 2 \quad z_5^d = 1 \quad z_6^d = 0.$$

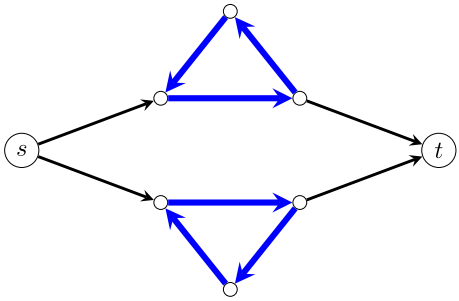


Fig. 19 The figure shows an example, where no unsplittable flow can satisfy any $O(d_{\max})$ violation of the lower bounds. Vertex s is the source and t is the terminal. The fractional flow splits half-half between the up and down option and then cycles often along the blue, thick triangles. The unsplittable flow can only choose one of the two options and thus violates the lower bounds heavily

In the first case, $D_{y,z}(\{1,2\}) = 1.5 > \frac{d_{\max}}{2}$ and in the second case $D_{y,z}(\{4,5\}) = 1.5 > \frac{d_{\max}}{2}$. We conclude that in general it is indeed impossible to achieve a (y,z)-interval discrepancy of at most $\frac{d_{\max}}{2}$ on half-integral instances.

A Appendix

A.1 Hardness of SSUF with lower bounds on cyclic graphs

We start by observing that in case of general (cyclic) graphs, Conjecture 1.4 and even Conjecture 1.5 cannot hold as stated, because the desired unsplittable flow might not exist.

Observation A.1 *Given some constant $\lambda > 0$, there is an SSUF instance $(G = (V, A), s, T, d, x)$ such that there exists no unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ with*

$$x(a) - \lambda d_{\max} \leq \text{flow}_{\mathcal{P}}(a) \quad \text{for all } a \in A.$$

Consider the graph shown in Figure 19 with one commodity of demand d. The fractional flow x on all black, thin arcs is $\frac{d}{2}$, while it is $d(\lambda+1)$ on the non-horizontal, blue, thick arcs and $d(\lambda + \frac{3}{2})$ on the horizontal, thick, blue arcs. Note that this flow is a convex combination of two walks. Still, in this case there is no path, and actually not even a walk, that can send positive flow along both triangles, which means that, for any unsplittable flow, some lower bounds are violated heavily.

Thus, when considering general graphs, the nature of the problem substantially changes. Instead of proving existence of certain unsplittable flows under the assumption that a splittable flow exists, we consider a different variant of the problem where we are given lower bounds $\{\ell(a)\}_{a \in A}$ instead.

In acyclic graphs we can use Theorem 1.7 to either decide that there exists no unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ with $\ell(a) \leq \text{flow}_{\mathcal{P}}(a)$ for all $a \in A$, or find an unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ with $\ell(a) - d_{\max} \leq \text{flow}_{\mathcal{P}}(a)$. To this end we simply

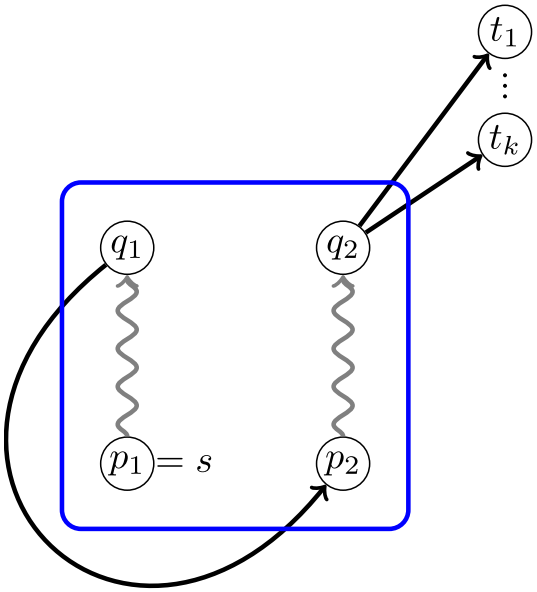


Fig. 20 Given a 2-VDP instance $I = (G, (p_1, q_1), (p_2, q_2))$, we obtain the corresponding graph $\overline{G}$ from G by adding an arc (q_1, p_2), adding terminals $t_1, \ldots, t_k$, and adding for each terminal t_i an arc (q_2, t_i). Moreover, the vertex p_1 is the source. The idea is that there are two vertex-disjoint paths (as depicted in gray) if and only if there is a path from s to a terminal using the arc (q_1, p_2)

check if there exists a splittable flow satisfying the lower bounds $\{\ell(a)\}_{a \in A}$. If no such splittable flow exists, there is also no unsplittable flow satisfying the lower bounds. Otherwise, we obtain a splittable flow x satisfying the given lower bounds, and we can apply Theorem 1.7 to x to find an unsplittable flow that violates the lower bounds by at most $d_{\max}$.

While Theorem 1.7 cannot be extended to cyclic graphs, as we observed above, one might still hope for an efficient algorithm that either decides that no unsplittable flow satisfying the lower bounds exists, or finds one that violates the lower bounds by at most $O(d_{\max})$. We next show that this is impossible for general (not necessarily planar) instances unless P = NP.

To this end, we reduce from the 2-Vertex-Disjoint Paths problem (2-VDP), which is well-known to be NP-complete [10]. In a 2-VDP instance, we are given a directed graph $G = (V, A)$ and two pairs of vertices (p_1, q_1) and (p_2, q_2). The task is to decide whether there are two vertex-disjoint paths Q_1 and Q_2 from p_1 to q_1 and from p_2 to q_2, respectively. Given a 2-VDP instance $I = (G, (p_1, q_1), (p_2, q_2))$, together with some number $k \in \mathbb{Z}_{\geq 1}$, we create an auxiliary graph $\overline{G} = (\overline{V}, \overline{A})$ together with a source $s \in V$, k terminals $T \subseteq \overline{V}$, and demands d as follows:

- $\overline{G}$ is obtained from G by adding an arc (q_1, p_2), and by adding k additional vertices $T = \{t_1, \ldots, t_k\}$, which will be the terminals, together with arcs (q_2, t_i) for all $i \in [k]$;
- all demands are unit, i.e., $d_t = 1$ for all $t \in T$;
- $s = p_1$ is the source.

See Figure 20 for an illustration of this construction.

We denote by $\overline{I} = (\overline{G}, s, T, d)$ the resulting single-source unsplittable flow instance without fractional flow vector x, as our statements here do not depend on a specific fractional flow or even its existence.

We then have the following immediate relation between the instance I and $\overline{I}$, which exhibits a gap in terms of how much flow can be sent over the arc (q_1, p_2) in $\overline{I}$, depending on whether I is feasible.

Proposition A.2 *Let $I = (G, (p_1, q_1), (p_2, q_2))$ be a 2-VDP instance, and $\overline{I} = (\overline{G}, s, T, d)$ be the corresponding SSUF instance (without fractional flow x). Then we have the following implications:*

(a) I is a yes instance $\implies$ $\exists$ unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ for $\overline{I}$ with $\text{flow}_{\mathcal{P}}((q_1, p_2)) = k$.

(b) I is a no instance $\implies$ $\nexists$ unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ for $\overline{I}$ with $\text{flow}_{\mathcal{P}}((q_1, p_2)) > 0$.

Proof If I is a yes instance, there is an s-q_2 path P in $\overline{G}$ using the arc (q_1, p_2). By setting each P^t for $t \in T$ to be the path obtained by appending the arc (q_2, t) to P, we get the desired result.

We prove the contrapositive of the second statement. Hence, if there is an unsplittable flow $\mathcal{P}$ with $\text{flow}_{\mathcal{P}}((q_1, p_2)) > 0$, then at least one path $P \in \mathcal{P}$ must use the arc (q_1, p_2). The path P contains a p_1-q_1 path and a p_2-q_2 path that are vertex-disjoint, which implies that I is a yes instance. □

There are different ways to interpret Proposition A.2. One natural way is that, in (cyclic) single-source unsplittable flow problems, it is hard to distinguish between the case where there is an unsplittable flow fulfilling all lower bounds and the case where there is no unsplittable flow even if we allow for a heavy violation of the lower bounds.

Corollary A.3 *Let $k \in \mathbb{Z}_{\geq 1}$ and let $\lambda \in \mathbb{R}_{\geq 0}$ such that $\lambda < k$ be constants. Then it is NP-hard to distinguish between the following two cases for SSUF instances on general (cyclic) graphs with k terminals:*

(i) There is an unsplittable flow satisfying all lower bounds, i.e., an unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ with $\ell(a) \leq \text{flow}_{\mathcal{P}}(a)$ for all $a \in A$.

(ii) There is no unsplittable flow satisfying all lower bounds with a violation of up to $\lambda d_{\max}$, i.e., there is no unsplittable flow $\mathcal{P} = \{P^t\}_{t \in T}$ with $\ell(a) - \lambda d_{\max} \leq \text{flow}_{\mathcal{P}}(a)$ for all $a \in A$.

Proof The hardness follows by considering SSUF instances $\overline{I} = (\overline{G}, s, T, d)$ (without fractional flow vector x) stemming from 2-VDP instances $I = (G, (p_1, q_1), (p_2, q_2))$. Moreover, we assign a lower bound of $\ell((q_1, p_2)) = k$ to the arc (q_1, p_2) and lower bounds of 0 to all other arcs. Then the two cases (i) and (ii) of Corollary A.3 correspond to (a) and (b) of Proposition A.2, respectively. Hence, they allow for distinguishing yes from no instances of 2-VDP, which is NP-hard. □

Note that the reduction and statement only hold true for general (cyclic) graphs because 2-VDP is polynomial-time solvable in acyclic graphs.

Funding Open Access funding enabled and organized by Projekt DEAL.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Ahuja, R.K., Magnanti, T., Orlin, J.B.: Network Flows: Theory, Algorithms, and Applications. Upper Saddle River, NJ, USA: Prentice Hall, (1993). ISSN:0-13-617549-X
2. Bansal, N., Kulkarni, J.: "Minimizing flow-time on unrelated machines". In: Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC). (2015), https://doi.org/10.1145/2746539.2746601
3. Baier, G., Köhler, E., Skutella, M.: The *k*-Splittalbe Flow Problem. Algorithmica **42**, 231–248 (2005). https://doi.org/10.1007/s00453-005-1167-9
4. Bansal, N., Rohwedder, L., Svensson, O.: "Flow time scheduling and prefix beck-fiala". In: Proceedings of the 54th Annual ACM Symposium on Theory of Computing (STOC). (2022), https://doi.org/10.1145/3519935.3520077
5. Chiba, N., Nishizeki, T., Abe, S., Ozawa, T.: A linear algorithm for embedding planar graphs using PQ-trees. J. Comput. Syst. Sci. **30**, 54–76 (1985). https://doi.org/10.1016/0022-0000(85)90004-2
6. Dinitz, Y., Garg, N., Goemans, M.X.: On the single-source unsplittable flow problem. Combinatorica **19**(1), 17–41 (1999). https://doi.org/10.1007/s004930050043
7. Du, J., Kolliopoulos, S.G.: Implementing approximation algorithms for the single-source unsplittable flow problem. J. Exp. Algorithmics (2005). https://doi.org/10.1145/1064546.1180614
8. Djidjev, H., Reif, J.: "An Efficient Algorithm for the Genus Problem with Explicit Construction of Forbidden Subgraphs". In: Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC). pp. 337–347 (1991)
9. Erickson, J., Whittlesey, K.: "Transforming Curves on Surfaces Redux". In: Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA). (2013), https://doi.org/10.1137/1.9781611973105.118
10. Fortune, S., Hopcroft, J., Wyllie, J.: The directed subgraph homeomorphism problem. Theoret. Comput. Sci. **10**(2), 111–121 (1980). https://doi.org/10.1016/0304-3975(80)90009-2
11. Grandoni, F., Mömke, T., Wiese, A.: "A PTAS for unsplittable flow on a path". In: Proceedings of the 54th Annual ACM Symposium on Theory of Computing (STOC). (2022), https://doi.org/10.1145/3519935.3519959
12. Guillemin, V., Pollack, A.: Differential Topology. Vol.370. American Mathematical Soc., (2010)
13. Greene, J.E.: On loops intersecting at most once. Geom. Funct. Anal. (2019). https://doi.org/10.1007/s00039-019-00517-0
14. Hopcroft, J., Tarjan, R.: Efficient planarity testing. J. ACM (1974). https://doi.org/10.1145/321850.321852
15. Kleinberg, J.M.: "Approximation Algorithms for Disjoint Paths Problems". PhD thesis. Massachusetts Institute of Technology (1996)
16. Kleinberg, J.M.: "Single-Source Unsplittable Flow". In: Proceedings of the 37th Annual Symposium on Foundations of Computer Science (FOCS). (1996), https://doi.org/10.1109/SFCS.1996.548465
17. Kolliopoulos, S.G.: Minimum-cost single-source 2-splittable flow. Inf. Process. Lett. **94**, 15–18 (2005). https://doi.org/10.1016/j.ipl.2004.12.009
18. Kolliopoulos, S.G.: "Edge-Disjoint Paths and Unsplittable Flow". In: Handbook of Approximation Algorithms and Metaheuristics. (2007). Chap.V, 57-1 to 57–16. ISSN:1-58488-550-5
19. Kolliopoulos, S.G., Stein, C.: Approximation algorithms for single-source unsplittable flow. SIAM J. Comput. **31**(3), 919–946 (2002). https://doi.org/10.1137/S0097539799355314
20. Koch, R., Skutella, M., Spenke, I.: Maximum *k*-splittable *s*-*t*-flows. Theory Comput Syst **43**, 56–66 (2008). https://doi.org/10.1007/s00224-007-9068-8

21. Levine, H.I.: “Homotopic curves on surfaces”. In: Proceedings of the American Mathematical Society 14(6), pp. 986–990 (1963)
22. Lenstra, J., Shmoys, D.B., Tardos, É.: Approximation algorithms for scheduling unrelated parallel machines. Math. Program. **46**, 259–271 (1990). https://doi.org/10.1007/BF01585745
23. Lovász, L., Spencer, J., Vesztergombi, K.: Discrepancy of set-systems and matrices. Eur. J. Comb. **7**, 151–160 (1986). https://doi.org/10.1016/S0195-6698(86)80041-5
24. Mohar, B.: A linear time algorithm for embedding graphs in an arbitrary surface. SIAM J. Discret. Math. (1999). https://doi.org/10.1137/S089548019529248X
25. Morell, S., Skutella, M.: Single source unsplittable flows with arc-wise lower and upper bounds. Math. Program. **192**, 477–496 (2022). https://doi.org/10.1007/s10107-021-01704-4
26. Martens, M., Salazar, F., Skutella, M.: “Convex Combinations of Single Source Unsplittable Flows”. In: Proceedings of the 15th Annual European Symposium on Algorithms (ESA). Vol.4698/2007. Lecture Notes in Computer Science. (2007), pp. 395–406. https://doi.org/10.1007/978-3-540-75520-3_36
27. Majthoub Almoghrabi, M., Skutella, M., Warode, P.: Integer and Unsplittable Multiflows in Series-Parallel Digraphs. (2024). https://doi.org/10.48550/arXiv.2412.05182
28. Mohar, B., Thomassen, C.: Graphs on Surfaces. Johns Hopkins University Press, https://doi.org/10.56021/9780801866890 (2001)
29. Skutella, M.: Approximating the single source unsplittable min-cost flow problem. Math. Program. **91**(3), 493–514 (2002). https://doi.org/10.1007/s101070100260
30. Salazar, F., Skutella, M.: Single-source k-splittable min-cost flows. Oper. Res. Lett. (2009). https://doi.org/10.1016/j.orl.2008.12.004
31. Shapley, R., Shmoys, D.B.: “Small Additive Error for Unsplittable Multicommodity Flow in Outerplanar Graphs”. In: Proceedings the 22nd International Workshop on Approximation and Online Algorithms (WAOA). Springer Nature. (2024), https://doi.org/10.1007/978-3-031-81396-2_12
32. de Verdière, É.C.: “Handbook of Discrete and Computational Geometry”. In: Chapman and Hall/CRC, 2017. Chap.Computational Topology of Graphs on Surfaces, pp. 605–636. ISSN:9781315119601
33. Williamson, D.P.: Network Flow Algorithms. Cambridge University Press (2019). https://doi.org/10.1017/9781316888568

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.