Generalized Max-Flows and Min-Cuts in Simplicial Complexes

William Maxwell
Oregon State University, Corvallis, OR, USA

Amir Nayyeri
Oregon State University, Corvallis, OR, USA

Abstract

We consider high dimensional variants of the maximum flow and minimum cut problems in the setting of simplicial complexes and provide both algorithmic and hardness results. By viewing flows and cuts topologically in terms of the simplicial (co)boundary operator we can state these problems as linear programs and show that they are dual to one another. Unlike graphs, complexes with integral capacity constraints may have fractional max-flows. We show that computing a maximum integral flow is NP-hard. Moreover, we give a combinatorial definition of a simplicial cut that seems more natural in the context of optimization problems and show that computing such a cut is NP-hard. However, we provide conditions on the simplicial complex for when the cut found by the linear program is a combinatorial cut. For d-dimensional simplicial complexes embedded into $\mathbb{R}^{d+1}$ we provide algorithms operating on the dual graph: computing a maximum flow is dual to computing a shortest path and computing a minimum cut is dual to computing a minimum cost circulation. Finally, we investigate the Ford-Fulkerson algorithm on simplicial complexes, prove its correctness, and provide a heuristic which guarantees it to halt.

2012 ACM Subject Classification Mathematics of computing → Algebraic topology

Keywords and phrases Max-flow min-cut, simplicial complexes, algebraic topology

Digital Object Identifier 10.4230/LIPIcs.ESA.2021.69

Related Version *Full Version*: `https://arxiv.org/abs/2106.14116`

Funding This research was supported in part by NSF grants CCF-1941086, CCF-1816442, and CCF-1617951.

1 Introduction

Computing flows and cuts are fundamental algorithmic problems in graphs, which are one dimensional simplicial complexes. In this paper, we explore generalizations of these algorithmic problems in higher dimensional simplicial complexes. In the case of graphs if an edge $e = (u, v)$ is assigned k units of flow we think of the edge as sending k units of flow from u to v. In two dimensions a flow is an assignment of real valued numbers to the triangles of the simplicial complex. A triangle assigned a value k sends k units of flow around its boundary. The difference in interpretation comes from the fact that the boundary of an edge is disconnected while the boundary of a triangle is connected.

Flows and cuts in simplicial complexes have natural algebraic definitions arising from the theory of simplicial (co)homology. A flow is an element of the kernel of the simplicial boundary operator, and a cut is an element of the image of the simplicial coboundary operator. These subspaces serve as generalizations of the cycle and cut spaces of a graph. This generalization has been studied by Duval, Klivans, and Martin in the setting of CW complexes [7]. We formulate the algorithmic problems of computing max-flows and min-cuts algebraically. By forgetting about the underlying graph structure and focusing on the (co)boundary operators, we obtain methods that naturally generalize to high dimensions.

In a graph an st-flow is an assignment of real values to the edges satisfying the conservation of flow constraints: the net flow out of any vertex other than s and t is zero, and, thus, the net flow that leaves s is equal to the net flow that enters t. Therefore, each st-flow can be viewed as a circulation in another graph with an extra edge that connects t to s. Circulations are elements of the cycle space of the graph with coefficients taken over $\mathbb{R}$. In a d-dimensional simplicial complex $\mathcal{K}$ the d-dimensional cycles are the formal sums (over $\mathbb{R}$) of d-dimensional simplices whose boundary is zero. Because there are no $(d+1)$-simplices flows are the elements of the dth homology group $H_d(\mathcal{K}, \mathbb{R})$. The maximum flow problem in a simplicial complex asks to find an optimal element of $H_d(\mathcal{K}, \mathbb{R})$ subject to capacity constraints.

The max-flow min-cut theorem states that in a graph the value of a maximum st-flow is equal to the value of a minimum st-cut. This result is a special case of linear programming duality. By rewriting the linear program in terms of the (co)boundary operator we obtain a similar result for simplicial complexes. The question of whether or not a similar max-flow min-cut theorem holds for simplicial complexes was asked, and left open, in a paper by Latorre [19]. We give a positive answer to this question, but with a caveat. When viewing flows and cuts from a topological point of view their linear programs are dual to one another. However, we also provide a combinatorial definition of a cut which feels more natural for a minimization problem. Topological and combinatorial cuts are equivalent for graphs, but they become different in dimensions $d > 1$. Flows in higher dimension, are dual to topological cuts, but not combinatorial cuts in general. From a computational complexity viewpoint the two notions of cuts are very different. We show that computing a minimum topological cut can be solved via linear programming, but that computing a minimum combinatorial cut is NP-hard.

A closely related problem is the problem of computing a max-flow in a graph which admits an embedding into some topological space. The most well-studied cases are planar graphs and the more general case when the graph embeds into a surface [2, 3, 4, 10, 12, 13, 14, 17, 18, 21, 22]. Max-flows and min-cuts are computationally easier to solve in surface embedded graphs, especially planar graphs. We consider this problem generalized to simplicial complexes. Planar graphs are 1-dimensional complexes embedded in $\mathbb{R}^2$, in Section 5 we consider the special case when a d-dimensional simplicial complex admits an embedding into $\mathbb{R}^{d+1}$. These complexes naturally admit a dual graph which we use to compute maximum flows and minimum cuts (both topological and combinatorial). We show that a maximum flow in a simplicial complex can be found by solving a shortest paths problem in its dual graph. This idea was used by Hassin to solve the maximum flow problem in planar graphs [13]. Further, we show that finding a minimum topological cut can be done by finding a minimum cost circulation in its dual graph. By setting the demand and capacity constraints equal to one in the minimum cost circulation problem we obtain an algorithm computing a minimum combinatorial cut.

Maximum flows in graphs can be computed using the Ford-Fulkerson algorithm. Moreover, the fact that the Ford-Fulkerson algorithm halts serves as a proof that there exists a maximum integral flow when the graph has integral capacity constraints. In dimensions $d > 1$ the maximum flow may be fractional, even with integral capacity constraints. The problem arises due to the existence of torsion in simplicial complexes of dimension $d > 1$. We show that despite the maximum flow being fractional the Ford-Fulkerson algorithm halts on simplicial complexes. However, in order for it to halt a special heuristic on picking the high dimensional analog of an augmenting path must be implemented. Despite the algorithm halting it could we could not prove a polynomial upper bound on the number of iterations it takes.

2 Preliminaries

Given a simplicial complex $\mathcal{K}$ we define $C_d(\mathcal{K}, G), Z_d(\mathcal{K}, G), B_d(\mathcal{K}, G)$, and $H_d(\mathcal{K}, G)$ to be the d-dimensional chain, cycle, boundary, and homology spaces with coefficients over G. In this paper we will consider coefficients over $\mathbb{R}$ and $\mathbb{Z}$, however when working over $\mathbb{R}$ we will typically drop the G in the notation; for example $C_d(\mathcal{K})$ will refer to the dth chain group over the reals. We will use ∂_d and δ_d to denote the d-dimensional boundary and coboundary operators and we will use $\mathcal{K}^d$ to denote the d-skeleton of $\mathcal{K}$. We view chains and cochains as both vectors and as functions which are equivalent viewpoints up to duality. For any chain σ by $\mathsf{supp}(\sigma)$ we denote its support which is the set of simplices given given a non-zero value on the chain. We call a $(d-1)$-chain *null-homologous* if it is the boundary of some d-chain.

The fundamental theorem of finitely generated abelian groups gives us the decomposition $H_d(\mathcal{K}, \mathbb{Z}) \cong \mathbb{Z}^k \oplus \mathbb{Z}_{t_1} \oplus \cdots \oplus \mathbb{Z}_{t_n}$ for some $k \in \mathbb{N}$. We call the subgroup $\mathbb{Z}_{t_1} \oplus \cdots \oplus \mathbb{Z}_{t_n}$ the *torsion subgroup* of $H_d(\mathcal{K}, \mathbb{Z})$ and when this subgroup is trivial we call the complex *torsion-free*. We say $\mathcal{K}$ is relatively torsion-free in dimension d if the relative homology groups $H_d(\mathcal{L}, \mathcal{L}_0, \mathbb{Z})$ is torsion-free for all subcomplexes $\mathcal{L}$ and $\mathcal{L}_0$ of dimensions d and $d-1$, respectively. There exist complexes that are torsion-free but are not relatively torsion-free; see [5] for examples.

Let A be a matrix; we say that A is *totally unimodular* if every square submatrix A' of A has $\det(A') \in \{-1, 0, 1\}$. Totally unimodular matrices are important in combinatorial optimization because linear programs with totally unimodular constraint matrices are guaranteed to have integral solutions [11]. Dey, Hirani, and Krishnamoorthy have provided topological conditions on when a simplicial complex has a totally unimodular boundary matrix [5] stated below. The dth boundary matrix ∂_d of a simplicial complex is totally unimodular if and only if the complex is relative torsion-free in dimension $d-1$.

▶ **Theorem 1** (Dey et al. [5], Theorem 5.2). *Let $\mathcal{K}$ be a d-dimensional simplicial complex. The boundary matrix $\partial_d \colon C_d(\mathcal{K}) \to C_{d-1}(\mathcal{K})$ is totally unimodular if and only if $H_{d-1}(\mathcal{L}, \mathcal{L}_0, \mathbb{Z})$ is torsion-free for all pure subcomplexes $\mathcal{L}_0, \mathcal{L}$ of $\mathcal{K}$ of dimensions $d-1$ and d where $\mathcal{L}_0 \subset \mathcal{L}$.*

Throughout this paper we utilize discrete Hodge theory and recommend the survey by Lim [20] as an introduction to the topic. In particular, we use the Hodge decomposition which can be stated as a result on real valued matrices satisfying $AB = 0$.

▶ **Theorem 2** (Hodge decomposition [20]). *Let $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{n \times p}$ be matrices satisfying $AB = 0$. We can decompose $\mathbb{R}^n$ into the orthogonal direct sum $\mathbb{R}^n = \mathrm{im}(A^T) \oplus \ker(A^T A + BB^T) \oplus \mathrm{im}(B)$.*

Setting $A = \partial_d$ and $B = \partial_{d+1}$ yields the Hodge decomposition for simplicial complexes. The middle term of the direct sum becomes $\ker(\delta_{d+1}\partial_{d+1} + \partial_d\delta_d)$. The linear operator $\delta_{d+1}\partial_{d+1} + \partial_d\delta_d$ is known as the combinatorial Laplacian of $\mathcal{K}$ which is a generalization of the graph Laplacian. Moreover, it can be shown that $\ker(\delta_{d+1}\partial_{d+1} + \partial_d\delta_d) \cong H_d(\mathcal{K}, \mathbb{R})$. We now state the Hodge decomposition on simplicial complexes as the following isomorphism $C_d(\mathcal{K}, \mathbb{R}) \cong \mathrm{im}(\delta_d) \oplus H_d(\mathcal{K}, \mathbb{R}) \oplus \mathrm{im}(\partial_{d+1})$.

3 Flows and cuts

In this section we give an overview of our generalizations of flows and cuts from graphs to simplicial complexes. Flows and cuts in higher dimensional settings have been studied previously. Duval, Klivans, and Martin have generalized cuts and flows to the setting of

CW complexes [7]. Their definitions are algebraic; defining cuts to be elements of $\mathrm{im}(\delta)$ and flows to be elements of $\ker(\partial)$. Our definitions are closely related, but are motivated by the algorithmic problems of computing max-flows and min-cuts. In Section 3.1 we give definitions of flows and cuts from from the perspective of algebraic topology, and in Section 3.2 we give a combinatorial definition of a cut in a simplicial complex. The distinction between the two types of cuts will be important when formulating the minimum cut problem on simplicial complexes.

3.1 Topological flows and cuts

First we briefly recall the definition of an st-flow in a directed graph $G = (V, E)$. An st-flow f is a function $f\colon E \to \mathbb{R}$ satisfying the conservation of flow constraint: for all $v \in V \setminus \{s, t\}$ we have $\sum_{(u,v)\in E} f(u, v) = \sum_{(v,u)\in E} f(v, u)$. That is, the amount of flow entering the vertex equals the amount of flow leaving the vertex. The value of f is equal to the amount of flow leaving s (or equivalently, entering t). Alternatively, we may view f as a 1-chain and we have $\partial f = k(t - s)$ where k is the value of f. Note that $t - s$ is a null-homologous 0-cycle. More generally, for any null-homologous $(d-1)$-cycle γ we call a d-chain f with $\partial f = k\gamma$ a γ-flow of value k. Note that under our naming convention an "st-flow" in a graph would be called a $(t - s)$-flow. However, in the case of graphs we use the traditional naming convention and call a flow from s to t an st-flow.

▶ **Definition 3** (γ-flow). *Let $\mathcal{K}$ be a d-dimensional simplicial complex and γ be a null-homologous $(d-1)$-cycle in $\mathcal{K}$. A **γ-flow** is a d-chain f with $\partial f = k\gamma$ where $k \in \mathbb{R}$. We call k the value of the flow f and denote the value of f by $\|f\|$. We say that f is feasible with respect to a capacity function $c\colon \mathcal{K}^d \to \mathbb{R}^+$ if $0 \le f(\sigma) \le c(\sigma)$ for all $\sigma \in \mathcal{K}^d$.*

Our definition of a γ-flow is very close to the algebraic definition which is element of $\ker(\partial)$. Given a simplicial complex $\mathcal{K}$ and a γ-flow f of value k we convert f into a *circulation*, where a circulation is defined to be an element of $\ker(\partial)$. To convert f into a circulation we add an additional basis element to $C_d(\mathcal{K})$, call it Σ, whose boundary is $\partial\Sigma = -\gamma$. This operation is purely algebraic; we should think of it as operating on the chain complex rather than the underlying topological space. Now we construct the circulation $f' = f + k\Sigma$. We call any circulation built from a γ-flow a γ-circulation. Clearly, $f' \in \ker(\partial)$ in the new chain complex. Moreover, there is a clear bijection between γ-flows and γ-circulations. The value of the circulation is the value of $f'(\Sigma)$, so this bijection preserves the value.

We now shift our focus to the generalization of cuts to a simplicial complex. The algebraic definition, elements of $\mathrm{im}(\delta)$, is natural. The cut space of a graph is commonly defined to be the space spanned by the coboundaries of each vertex. In a simplicial complex $\mathcal{K}$, removing the support of a d-chain in $\mathrm{im}(\delta)$ increases $\dim H_{d-1}(\mathcal{K})$. In a graph G, removing the support of any 1-chain in $\mathrm{im}(\delta)$ increases $\dim H_0(G)$ which is equivalent to increasing the number of connected components of G.

The above definition implies that a cut is a d-chain in a d-dimensional simplicial complex. However, for our purposes we will define a cut to be a $(d-1)$-cochain. To motivate our definition we recall the notion of an st-cut in a graph. An st-cut in a graph is a partition of the vertices into sets S and T such that $s \in S$ and $t \in T$. Define $p\colon V(G) \to \{0, 1\}$ such that $p(v) = 1$ if $v \in S$ and $p(v) = 0$ if $v \in T$. The support of the coboundary of p is a set of edges whose removal destroys all st-paths. That is, upon removing the support, the 0-cycle $t - s$ is no longer null-homologous. Moreover, p is a 0-cochain with $p(t - s) = -1$. The sign of $p(t - s)$ will be important when we consider directed cuts. With this in mind we define our notion of a γ-cut.

▶ **Definition 4** (γ-cut). *Let $\mathcal{K}$ be a d-dimensional simplicial complex with weight function $c\colon \mathcal{K}^d \to \mathbb{R}^+$ and γ be a null-homologous $(d-1)$-cycle in $\mathcal{K}$. A **γ-cut** is a $(d-1)$-cochain p such that $p(\gamma) = -1$. Denote the coboundary of p as the formal sum $\delta(p) = \sum \alpha_i \sigma_i$, we define the size of a γ-cut to be $\|p\| = \sum |\alpha_i c(\sigma_i)|$.*

Because of the requirement that $p(\gamma) = -1$ we call p a *unit γ-cut*. By relaxing this requirement to $p(\gamma) < 0$ the cochain p still behaves as a γ-cut, but its size can become arbitrarily small by multiplying by some small value $\epsilon > 0$. We justify our definition with the following proposition which shows that removing the support of the coboundary of a γ-cut prevents γ from being null-homologous.

▶ **Proposition 5.** *Let $\mathcal{K}$ be d-dimensional simplicial complex and p be a γ-cut. The cycle γ is not null-homologous in the subcomplex $\mathcal{K} \setminus \mathit{supp}(\delta(p))$.*

3.2 Combinatorial cuts

Alternatively, we can view a γ-cut as a discrete set of d-simplices rather than a d-chain. In the case of graphs a combinatorial st-cut is just a set of edges whose removal disconnects s from t. This distinction will become important when we consider the minimization problem of finding a minimum cost set of d-simplices whose removal prevents γ from being null-homologous.

▶ **Definition 6** (Combinatorial γ-cut). *Let $\mathcal{K}$ be a d-dimensional simplicial complex with weight function $c\colon \mathcal{K}^d \to \mathbb{R}^+$ and γ be a null-homologous $(d-1)$-cycle in $\mathcal{K}$. A **combinatorial γ-cut** is a set of d-simplices $C \subseteq \mathcal{K}^d$ such that γ is not null-homologous in $\mathcal{K} \setminus \mathit{supp}(C)$. The size of a combinatorial γ-cut is defined by the sum of the weights of the d-simplices $\|C\| = \sum_{\sigma \in C} c(\sigma)$.*

The next proposition shows a relationship between γ-cuts and combinatorial γ-cuts. Removing a combinatorial γ-cut C from $\mathcal{K}$ increases $\dim H_{d-1}(\mathcal{K})$. This is because removing C must decrease the rank of ∂_d and by duality this also decreases the rank of δ_d which increases the dimension of $H^{d-1}(\mathcal{K}) \cong H_{d-1}(\mathcal{K})$. It follows that C must contain the support of some coboundary. Given an additional minimality condition on C we show that C is equal to the support of some coboundary.

▶ **Proposition 7.** *Let C be a combinatorial γ-cut in a d-dimensional simplicial complex $\mathcal{K}$. Further, assume that C is minimal in the sense that for all $C' \subset C$ we have $\dim H_{d-1}(\mathcal{K} \setminus C') < \dim H_{d-1}(\mathcal{K} \setminus C)$. There exists a $(d-1)$-cochain p such that $\mathit{supp}(\delta(p)) = C$.*

In graphs the linear program solving the minimum st-cut problem takes as input a directed graph and returns a set of directed edges whose removal destroys all directed st-paths. This is called a directed cut. After removing the directed cut the 0-cycle $t - s$ may still be null-homologous; we can find a 1-chain with boundary $t - s$ using negative coefficients to traverse an edge in the backwards direction. In order to generalize the minimum cut linear program to simplicial complexes we will need to define a directed combinatorial γ-cut, which requires the additional assumption that the d-simplices of $\mathcal{K}$ are oriented.

▶ **Definition 8** (Directed combinatorial γ-cut). *Let $\mathcal{K}$ be an oriented d-dimensional simplicial complex with weight function $c\colon \mathcal{K}^d \to \mathbb{R}$ and γ be a null-homologous $(d-1)$-cycle in $\mathcal{K}$. A **directed combinatorial γ-cut** is a set of d-simplices $C \subset \mathcal{K}^d$ such that in $\mathcal{K} \setminus \mathit{supp}(C)$ there exists no d-chain Γ with non-negative coefficients such that $\partial\Gamma = \gamma$. The size of a directed combinatorial γ-cut is defined by the sum of the weights of the d-simplices $\|C\| = \sum_{\sigma \in C} c(\sigma)$.*

Given a directed graph consider an st-cut given by the cochain definition. That is, a 0-cochain $p\colon V(G) \to \{0,1\}$ with $p(s) = 1$ and $p(t) = 0$ partitioning V into S and T. The support of $\delta(p)$ consists of two types of edges: edges leaving S and entering T, and edges leaving T and entering S. If $e \in E$ leaves S and enters T we have $(p \circ \partial)(e) = -1$ and if e leaves T and enters S we have $(p \circ \partial)(e) = 1$. To construct a directed st-cut we simply take all of the edges mapped to -1. The following proposition shows that we can build a directed combinatorial γ-cut from a coboundary just like in the case of directed graphs.

▶ **Proposition 9.** *Let p be a γ-cut with coboundary $\delta(p) = \sum \alpha_i \sigma_i$. The set of d-simplices $C = \{\sigma_i \mid \alpha_i < 0\}$ is a directed combinatorial γ-cut.*

To conclude the section we will show that computing a minimum combinatorial γ-cut is NP-hard. As we will see in Section 4 minimum topological γ-cuts can be computed with linear programming. Our hardness result holds for both the directed and undirected cases. Our hardness result is a reduction from the well-known NP-hard hitting set problem which we will now define. Given a set S and a collection of subsets $\Sigma = (S_1, \ldots, S_n)$ where $S_i \subseteq S$ the hitting set problem asks to find the smallest subset $S' \subseteq S$ such that $S' \cap S_i \neq \emptyset$ for all S_i. We call such a subset S' a *hitting set* for (S, Σ).

▶ **Theorem 10.** *Let $\mathcal{K}$ be a d-dimensional simplicial complex and γ be a null-homologous $(d-1)$-cycle. Computing a minimum combinatorial γ-cut is NP-hard for $d \geq 2$.*

Proof. Our proof is a reduction from the hitting set problem. First we consider the case when $d = 2$ then we generalize to any $d \geq 2$. Let S be a set and $\Sigma = (S_1, \ldots, S_n)$ where each $S_i \subseteq S$. We construct a 2-dimensional simplicial complex $\mathcal{K}$ from S and Σ in the following way. For each $S_i \in \Sigma$ construct a triangulated disk $\mathcal{D}_i$ such that $\partial \mathcal{D}_i = \gamma$. That is, each $\mathcal{D}_i$ shares the common boundary γ. To accomplish this we construct each $\mathcal{D}_i$ by beginning with a single triangle t with $\partial t = \gamma$ and repeatedly adding a new vertex in the center of some triangle with edges connecting it to every vertex in that triangle. By this process we can construct a disk containing any odd number of triangles as each step increments the number of triangles in the disk by two. Moreover, at each step the boundary of the disk is always γ. We construct each disk $\mathcal{D}_i$ such that $\mathcal{D}_i$ consists of one triangle $t_{i,s}$ for each element $s \in S_i$ and potentially one extra triangle t'_i in the case that $|S_i|$ is even. Next, for each $s \in S$ and S_i with $s \in S_i$, we construct the quotient space by identifying each $t_{i,s}$ into a single triangle. A minimum combinatorial γ-cut C must contain at least one triangle from each $\mathcal{D}_i$ and without loss of generality we can assume C does not contain any t'_i. If $t'_i \in C$ then by minimality it is the only triangle in $C \cap \mathsf{supp}(\mathcal{D}_i)$ and we can swap it with any other triangle in $\mathcal{D}_i$ without increasing the size of the cut. By construction C is a hitting set for (S, Σ) since each $C \cap \mathsf{supp}(\mathcal{D}_i) \neq \emptyset$ for all $\mathcal{D}_i$. The proof generalizes to $d > 2$ by generalizing the subdivision processes. ◀

4 Linear programming

4.1 Max-flow min-cut

A *simplicial flow network* is a tuple $(\mathcal{K}, c, \gamma)$ where $\mathcal{K}$ is an oriented d-dimensional simplicial complex, c is the *capacity function* which is a non-negative function $c\colon \mathcal{K}^d \to \mathbb{R}^+$, and γ is a null-homologous $(d-1)$-cycle. In a simplicial flow network we work with real coefficients; that is, we consider the chain groups $C_k(\mathcal{K}, \mathbb{R})$. In order to utilize the Hodge decomposition (Theorem 2) in a convenient way we modify $C_d(\mathcal{K})$ by adding an additional basis element Σ such that $\partial \Sigma = -\gamma$. Moreover, we extend the capacity function such that $c(\Sigma) = \infty$. This

allows us to work with circulations instead of flows while leaving the solution unchanged. The notation n_d will refer to the number of basis elements in $C_d(\mathcal{K}, \mathbb{R})$ which is now one more than the number of d-simplices in the underlying simplicial complex.

The goal of the maximum flow problem is to find a d-chain f obeying the capacity constraints such that $\partial f = k\gamma$ where $k \in \mathbb{R}$ is maximized. Equivalently, we find a d-cycle f which maximizes $f(\Sigma)$. The linear program for the max-flow problem in a simplicial flow network is identical to the familiar linear program for graphs, but expressed in terms of the coboundary operator. In a graph, conservation of flow at a vertex v is the constraint $\delta_1(v) \cdot f = 0$; to formulate the linear program in higher dimensions we simply replace vertices with $(d-1)$-simplices. The Hodge decomposition states that cycles are orthogonal to coboundaries, so conservation of flow ensures that f is indeed a cycle. We now state the linear program for max-flow in a simplicial flow network.

$$\begin{aligned} \text{maximize} \quad & f(\Sigma) \\ \text{subject to} \quad & \delta(\tau) \cdot f = 0 \qquad \text{for each } \tau \in \mathcal{K}^{d-1} \\ & 0 \leq f(\sigma) \leq c(\sigma) \quad \text{for each } \sigma \in \mathcal{K}^{d} \end{aligned} \tag{LP1}$$

We dualize LP1 to obtain a generalization of the minimum cut problem in directed graphs. To make the dualization more explicit we will write out LP1 in matrix form: maximize $s \cdot f$ subject to $Af \leq b$ and $f \geq 0$, where we have $A = \begin{bmatrix} \partial \\ -\partial \\ I_{n_d} \end{bmatrix}$, $b = \begin{bmatrix} 0_{n_{d-1}} \\ 0_{n_{d-1}} \\ c \end{bmatrix}$, $s = \begin{bmatrix} 0_{n_d - 1} \\ 1 \end{bmatrix}$. The matrix A has dimension $(2n_{d-1} + n_d) \times n_d$. In our notation I_k is the $k \times k$ identity matrix and 0_k is the $k \times 1$ column vector consisting of all zeros. Since the value of the flow is equal to $f(\Sigma)$ the vector s is all zeros except for the final entry which is indexed by Σ and receives an entry equal to one. Further, c is the $n_d \times 1$ capacity vector indexed by the d-simplices such that the entry indexed by σ has value equal to $c(\sigma)$.

We can now state the dual program in matrix form: minimize $y \cdot b$ subject to $y^T A \geq s$ and $y \geq 0$. The vector y is a $(2n_{d-1} + n_d) \times 1$ column vector indexed by both the $(d-1)$-simplices and the d-simplices. However, only the entries indexed by d-simplices contribute to the objective function since b is zero everywhere outside of the capacity constraints. We will denote the truncated vector consisting of entries indexed by d-simplices by y_d and the entry corresponding to the d-simplex $\sigma \in \mathcal{K}^d$ will be denoted by $y_d(\sigma)$. Similarly we have two truncated vectors y^1_{d-1} and y^2_{d-1} corresponding to the entries indexed by the $(d-1)$-simplices. Moreover, the rows of $y^T A \geq s$ are in the form $(y^1_{d-1} - y^2_{d-1})^T \partial + y_d \geq s$. For simplicity we define $y_{d-1} = y^1_{d-1} - y^2_{d-1}$ and write $y_{d-1}(\tau)$ for the entry indexed by the $(d-1)$-simplex τ. Putting this together, we state the dual linear program as follows.

$$\begin{aligned} \text{minimize} \quad & \sum_{\sigma \in \mathcal{K}^d} y_d(\sigma) c(\sigma) \\ \text{subject to} \quad & y_{d-1} \cdot \partial\sigma + y_d(\sigma) \geq 0 \qquad \text{for each } \sigma \in \mathcal{K}^d \\ & y_{d-1} \cdot \partial\Sigma + y_d(\Sigma) = 1,\ y_d \geq 0 \end{aligned} \tag{LP2}$$

Note the strict equality in the second constraint does not follow from the duality. However, we can assume a strict equality since if $y_{d-1} \cdot \partial\Sigma + y_d(\Sigma) > 1$ we can multiply $[y_{d-1}, y_d]^T$ by some scalar $\epsilon < 1$ to make the inequality tight. This multiplication only decreases the value of $\sum y_d(\sigma) c(\sigma)$ so it does not change the optimal solution.

In the case of graphs LP2 has dual variables for vertices and edges. Moreover, there exists an integral solution such that each vertex is either assigned a 0 or a 1 since a graph cut is a partition of the vertices. The second inequality requires $y_0(s) = 1$ and $y_0(t) = 0$. To see this, when solving an st-cut on a graph, the basis element Σ is an edge with $\partial\Sigma = s - t$, and $y_1(\Sigma) = 0$ otherwise the solution is infinite. This naturally defines a partition of the vertices: S containing vertices assigned a 1, and T containing vertices assigned a 0. The constraints force an edge to be assigned a 1 if it leaves S and enters T, otherwise it is assigned a 0. This solution can be interpreted as a 0-cochain p with $p(st) = 1$, or in the notation of our definition of a simplicial cut: $p(t - s) = -1$. Further, $y_1(e) = 1$ for every edge e that is negative on $\delta(p)$ and a 0 otherwise, hence y_1 fits our definition of a directed st-cut in a 1-complex. We will show the same result holds in higher dimensions; that is, y_d is a directed γ-cut arising from the $(d-1)$-cochain y_{d-1}.

▶ **Lemma 11.** *Let $y = [y_{d-1}, y_d]^T$ be an optimal solution to LP2. The set $\mathsf{supp}(y_d)$ is a directed combinatorial γ-cut.*

▶ **Lemma 12.** *Let p be a γ-cut with coboundary $\delta(p) = \sum \alpha_i\sigma_i$ and let $\delta(p)^- = \sum_{\alpha_i<0} \alpha_i\sigma_i$. The vector $[p, -\delta(p)^-]^T$ is a finite feasible solution to LP2.*

Lemma 11 tells us that a solution to LP2 yields a directed combinatorial γ-cut. Recall, by Proposition 9 every γ-cut p yields a directed combinatorial γ-cut by taking the coboundary $\delta(p) = \sum \alpha_i\sigma_i$ and considering the negative components $\delta(p)^- = \{\sigma_i \mid \alpha_i < 0\}$. By Lemma 12 $\delta(p)^-$ is a feasible solution to LP2; the cost of this solution is $c \cdot \delta(p)^-$. The coefficients α_i need not always equal one; hence in general we have $\|C\| \neq c \cdot \delta(p)^-$. It follows that LP2 need not return a minimum directed combinatorial γ-cut. In Theorem 15 we will give conditions describing when LP2 returns a directed combinatorial γ-cut. To conclude the section we state our main theorem about LP2 whose proof is immediate from Lemmas 11 and 12.

▶ **Theorem 13.** *Let $y = [y_{d-1}, y_d]^T$ be an optimal solution to LP2. The set $\mathsf{supp}(y_d)$ is a directed combinatorial γ-cut such that $y_d = \delta(y_{d-1})^-$. Moreover, y_d minimizes $c \cdot \delta(p)^-$ where p ranges over all γ-cuts.*

4.2 Integral solutions

In this section we provide an example of a simplicial flow network with integral capacity constraints and fractional maximum flow. By Theorem 1 such a network must contain some relative torsion. This is achieved by the inclusion of a Möbius strip in our simplicial flow network. Our example will be used later in Theorem 14 showing that computing a maximum integral flow in a simplicial flow network is NP-hard.

We will now explicitly describe a simplicial flow network with integral capacities whose maximum flow value is fractional. Let $\mathcal{M}$ be a triangulated Möbius strip with boundary $\partial\mathcal{M} = 2\alpha + \gamma$ such that two vertices in α have been identified making α a simple cycle. This identification makes γ a figure-eight. Now let $\mathcal{D}$ be a triangulated disk oriented such that $\partial\mathcal{D} = -\alpha$. Call the resulting complex $\mathcal{M}\mathcal{D}$. See Figure 1 for an illustration. The capacity function c has $c(t) = 1$ for each triangle $t \in \mathcal{M}\mathcal{D}$. Now we solve the max-flow problem on $(\mathcal{M}\mathcal{D}, c, \gamma)$. Note that for any flow f we have $f(t_1) = f(t_2)$ for all triangles $t_1, t_2 \in \mathcal{M}$; moreover, for all $t_1, t_2 \in \mathcal{D}$ we also have $f(t_1) = f(t_2)$. The value of any flow f on $(\mathcal{M}\mathcal{D}, c, \gamma)$ is equal to its value on $\mathcal{M}$, and in order to maintain conservation of flow we must have $f(\mathcal{D}) = 2f(\mathcal{M})$. Now, the capacity constraints imply that the maximum flow f has $f(\mathcal{M}) = 1/2$ and $f(\mathcal{D}) = 1$. We have $\partial f = \frac{1}{2}\partial\mathcal{M} + \partial\mathcal{D} = \frac{1}{2}\gamma + \alpha - \alpha$. Hence, $\|f\| = 1/2$.

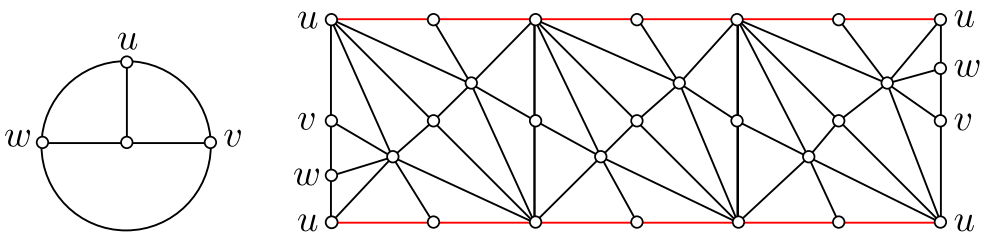

Figure 1 A triangulated disk $\mathcal{D}$ (left) and Möbius strip $\mathcal{M}$ (right). The Möbius strip has two points on its boundary identified forming the vertex u. In red we have the input cycle (a figure-eight) γ and we set $\alpha = uwvu$. We orient the complex such that $\partial\mathcal{M} = \gamma + 2\alpha$ and $\partial\mathcal{D} = -\alpha$. The capacity on each simplex in both the disk and Möbius strip is one.

Maximum integral flow. Given a simplicial flow network $(\mathcal{K}, c, \gamma)$ with integral capacities we consider the problem of finding the maximum integral flow. That is, a d-chain $f \in C_d(\mathcal{K}, \mathbb{Z})$ obeying the capacity constraints such that $\partial f = k\gamma$ where $k \in \mathbb{Z}$ is maximized. We show the problem is NP-hard by a reduction from graph 3-coloring. Our reduction is inspired by a MathOverflow post from Sergei Ivanov showing that finding a subcomplex homeomorphic to the 2-sphere is NP-hard [15]. Given a graph G we construct a 2-dimensional simplicial flow network whose maximum flow is integral if and only if G is 3-colorable.

▶ **Theorem 14.** *Let $(\mathcal{K}, c, \gamma)$ be a simplicial flow network where $\mathcal{K}$ is a 2-complex and c is integral. Computing a maximum integral flow of $(\mathcal{K}, c, \gamma)$ is NP-hard.*

Proof. Let $G = (V, E)$ be a graph. We will construct a simplicial flow network $(\mathcal{K}, c, \gamma)$ such that its maximum flow is integral if and only if G is 3-colorable.

We start our construction with a punctured sphere $\mathcal{S}$ containing $|V| + 1$ boundary components called γ and β_v for each $v \in V$. For each boundary component β_v we construct three disks R_v, B_v, G_v each with boundary $-\beta_v$. These disks represent the three colors in our coloring: red, blue, and green. We refer to these disks as *color disks* and use $\mathcal{C}_v$ to denote an arbitrary color disk associated with v and use $k \in \{r, b, g\}$ to denote an arbitrary color. On each color disk $\mathcal{C}_v$ we add a boundary component for each edge $e = (u, v)$ incident to v. By $\beta_{v,e,k}$ we denote the boundary component corresponding to the vertex v, edge e, and color k. For each edge $e = (u, v)$ and each pair of boundary components β_{u,e,k_u} and β_{v,e,k_v} with $k_u \neq k_v$ we construct a tube with boundary components $-\beta_{u,e,k_u}$ and $-\beta_{v,e,k_v}$ denoted $\mathcal{T}_{e,k_u,k_v}$. When $k_u = k_v = k$ we construct a tube $\mathcal{T}_{e,k,k}$ and puncture it with a third boundary component α and construct a negatively oriented real projective plane $\mathcal{RP}_{e,k}$ with boundary $\partial\mathcal{RP}_{e,k} = -2\alpha$. We call the resulting complex $\mathcal{K}$ and assign a capacity $c(\sigma) = 1$ for every triangle σ in $\mathcal{K}$.

We will show that a maximum integral flow f of $\mathcal{K}$ has $\|f\| = 1$ if and only if G is 3-colorable. The following four properties of a maximum integral flow f imply that G is 3-colorable.

- f must assign exactly one unit of flow to each triangle in $\mathcal{S}$ since the value of f is equal to $f(\mathcal{S})$.
- For each vertex $v \in V$ exactly one color disk $\mathcal{C}_v$ is assigned one unit of flow while the other two color disks associated with v are assigned zero units of flow. Otherwise, either conservation of flow is violated or some color disk is assigned a fractional flow value.
- For each edge $e = (u, v) \in E$ exactly one tube $\mathcal{T}_{e,k_u,k_v}$ with $k_u \neq k_v$ must be assigned one unit of flow with all other tubes associated with e assigned zero units of flow. The tube $\mathcal{T}_{e,k_u,k_v}$ assigned one unit of flow is the tube connecting the color disks $\mathcal{C}_v$ and $\mathcal{C}_u$ that are assigned one unit of flow by the previous property.

- f assigns zero flow to every $\mathcal{T}_{e,k,k}$ and $\mathcal{RP}_{e,k}$ since otherwise the triangles in $\mathcal{RP}_{e,k}$ would need to have $1/2$ units of flow assigned to them to maintain conservation of flow.

These four properties imply that the set of color disks $\{\mathcal{C}_v \mid f(\sigma) = 1, \forall \sigma \in \mathcal{C}_v\}$ corresponds to a 3-coloring of G. Conversely, given a 3-coloring of G we assign a flow value of one to each color disk corresponding to the 3-coloring. We extend this assignment to a γ-flow of value one by assigning a flow value of one to $\mathcal{S}$ and the tubes corresponding to the 3-coloring. ◀

Integral cuts. The goal of this section is to show that for simplicial complexes that are relative torsion-free in dimension $d-1$ there exists optimal solutions to LP2 whose support is a minimum combinatorial γ-cut. Note that by Theorem 1 a simplicial complex that is relative torsion-free in dimension $d-1$ has a totally unimodular d-dimensional boundary matrix. The total unimodularity is key to our proof. However, we first provide an example of a complex (with relative torsion) whose optimal solution's support does not form a minimum combinatorial γ-cut. Our construction is a slight modification of $\mathcal{MD}$ defined in Section 4.2.

Consider the simplicial complex constructed by taking $\mathcal{MD}$ and glueing a wedge sum of two disks $\mathcal{W}$ along the figure-eight γ. That is, $\partial\mathcal{W} = \gamma$. We give every triangle in the resulting complex a capacity equal to one. A maximum γ-flow has value $3/2$, so the dual program finds a γ-cut of the same value. One potential optimal solution is a $(d-1)$-cochain whose coboundary assigns a value of $-1/2$ to two triangles in $\mathcal{W}$ and a value of $-1/2$ to one triangle in $\mathcal{D}$. The support of this coboundary has weight equal to three, however a minimal combinatorial γ-cut has weight two by taking only one triangle from $\mathcal{W}$ and one from $\mathcal{D}$. See Figure 2 for an illustration.

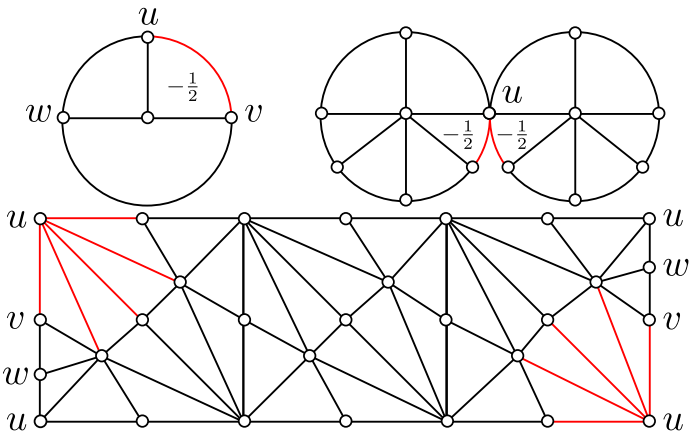

Figure 2 The simplicial complex $\mathcal{MD}$ with a wedge sum of two disks $\mathcal{W}$ identified to the figure-eight γ. In red we have a 1-cochain which assigns a value of $-1/2$ to each red edge. The coboundary of the red cochain assigns a value of $-1/2$ to one triangle in $\mathcal{D}$ and a value of $-1/2$ to two triangles in $\mathcal{W}$. The value of the red cochain coincides with the value of the maximum γ-flow. However, its support is not a minimum combinatorial γ-cut. A minimum combinatorial γ-cut picks one triangle from $\mathcal{D}$ and one triangle from $\mathcal{W}$.

Now, we show that when $\mathcal{K}$ is relative torsion-free in dimension $d-1$ LP2 has an optimal solution whose support is a minimum directed combinatorial γ-cut. Specifically, we show that a solution existing on a vertex of the polytope defined by the constraints of LP2 is a cochain y_{d-1} with negative coboundary y_d such that $y_d(\sigma) \in \{0, 1\}$ for all $\sigma \in \mathcal{K}^d$ hence $\sum y_d(\sigma)c(\sigma) = \|\mathsf{supp}(y_d)\|$. That is, the value of a vertex solution to LP2 is equal to the cost of $\mathsf{supp}(y_d)$ as a directed combinatorial γ-cut.

▶ **Theorem 15.** *Let $\mathcal{K}$ be d-dimensional simplicial complex that is relative torsion-free in dimension $d-1$ and $[y_{d-1}, y_d]^T$ be an optimal vertex solution to the dual program. The set $\mathsf{supp}(y_d)$ is a minimum directed combinatorial γ-cut.*

Proof. We can write the constraint matrix of LP2 as the $2n_d \times (n_d + n_{d-1})$ block matrix

$$A = \begin{bmatrix} \delta & I_{n_d} \\ 0_{n_d} & I_{n_d} \end{bmatrix}.$$

Since $\mathcal{K}$ is relative torsion-free in dimension $d-1$ Theorem 1 tells us that ∂_d is totally unimodular; further, we have that $\partial^T = \delta$ is also totally unimodular. Total unimodularity is preserved under the operation of adding a row or column consisting of exactly one component equal to 1 and the remaining components equal to 0, so A is totally unimodular [23, Section 19.4]. We write LP2 as the linear system $Ax \geq b$ where b is a $n_d + n_{d-1}$ dimensional vector with exactly one component equal to 1 and the remaining components equal to 0. Let $y = [y_{d-1}, y_d]^T$ be an optimal vertex solution to LP2. For every $(d-1)$-simplex $\tau \in \mathcal{K}^{d-1}$ we either have $y_{d-1}(\tau) \geq 0$ or $y_{d-1}(\tau) \leq 0$. Let $I'_{n_{d-1}}$ be the matrix whose rows correspond to these inequalities. Note that $I'_{n_{d-1}}$ is a diagonal matrix with entries in $\{-1, 1\}$. Now we consider the $(2n_d + n_{d-1}) \times (n_d + n_{d-1})$ dimensional linear system $A'x \geq b'$ where

$$A' = \begin{bmatrix} \delta & I_{n_d} \\ 0 & I_{n_d} \\ I'_{n_{d-1}} & 0 \end{bmatrix}$$

and b' is constructed by appending extra zeros to b. We construct y' from y similarly. Note that A' is totally unimodular and y' is a vertex solution of the system. There exists a vertex v of the polyhedron $P \subseteq \mathbb{R}^{n_d + n_{d-1}}$ corresponding to the linear system such that $A'y' = v \geq b'$ such that $n_{d-1} + n_d$ constraints are linearly independent and tight. Hence, there is an $(n_{d-1} + n_d) \times (n_{d-1} + n_d)$ square submatrix A'' with $A''y' = b''$ where b'' is b' restricted to the tight constraints. We will use Cramer's rule to show that the vertex solution y has components coming from the set $\{-1, 0, 1\}$. Let $A''_{i,b''}$ be the matrix obtained by replacing the i^{th} column of A'' with b''. By Cramer's rule we compute the i^{th} component of y as $y_i = \frac{\det(A''_{i,b''})}{\det(A'')}$. Since both $A''_{i,b''}$ and A'' are totally unimodular we have $v_i \in \{-1, 0, 1\}$. Further, we know that A'' is non-singular because it corresponds to linearly independent constraints.

By the above argument we know that an optimal solution y to LP2 has all of its components contained in the set $\{-1, 0, 1\}$. The constraint $y_d \geq 0$ means that for all d-simplices σ we have $y_d(\sigma) \in \{0, 1\}$ and $\sum_{\sigma \in \mathcal{K}^d} y_d(\sigma) c(\sigma) = \|\mathsf{supp}(y_d)\|$. Hence, $\mathsf{supp}(y_d)$ is a minimum directed combinatorial γ-cut. ◀

5 Embedded simplicial complexes

In this section we consider a simplicial flow network $(\mathcal{K}, c, \gamma)$ where $\mathcal{K}$ is a d-dimensional simplicial complex with an embedding into $\mathbb{R}^{d+1}$. Alexander duality implies that $\mathbb{R}^{d+1} \setminus \mathcal{K}$ consists of $\beta_d + 1$ connected components where $\beta_d = \dim H_d(\mathcal{K})$ is the dth Betti number. We call these connected components *voids*; exactly one void is unbounded and we denote the voids by V_i for $1 \leq i \leq \beta_{d+1}$. Given an embedding into $\mathbb{R}^{d+1}$, computing the voids of $\mathcal{K}$ can be done in polynomial time [6]. Further, we assume that the d-simplices are consistently oriented with respect to the voids. The embedding guarantees that every d-simplex σ appears on the boundary of at most two voids; by our assumption if σ appears on the boundary of

two voids then it most be oriented positively on one and negatively on the other. We denote the boundary of the void V_i by $\mathrm{Bd}(V_i)$. Every d-simplex contained in the support of some d-cycle is on the boundaries of exactly two voids; it follows that the boundaries of any set of β_d voids is a basis of $H_d(\mathcal{K})$.

In order to state our theorems we need one additional assumption on $\mathcal{K}$. We assume there exists some void V_i containing two unit γ-flows Γ_1, Γ_2 whose supports partition $\mathrm{Bd}(V_i)$: $\mathsf{supp}(\Gamma_1) \cap \mathsf{supp}(\Gamma_2) = \emptyset$ and $\mathsf{supp}(\Gamma_1) \cup \mathsf{supp}(\Gamma_2) = \mathrm{Bd}(V_i)$. This assumption makes our problem analogous to an st-flow network in a planar graph such that s and t appear on the same face. The existence of two unit γ-flows partitioning the boundary is analogous to the two disjoint st-paths on the boundary of the face. It will be convenient to take the negation of Γ_1 and treat it as a unit $(-\gamma)$-flow; otherwise the assumption conflicts with the assumed consistent orientation. This is equivalent as it does not change the support of the flow, so for the rest of the section we will take Γ_1 to be a unit $(-\gamma)$-flow.

From $\mathcal{K}$ we construct its directed dual graph $\mathcal{K}^*$ as follows. Each void becomes a vertex of $\mathcal{K}^*$. Each d-simplex on the boundary of two voids becomes an edge; since we assumed the d-simplices are consistently oriented we direct the dual edge from the negatively oriented void to the positively oriented void. The remaining d-simplices only appear on one void and become loops in $\mathcal{K}^*$. For a d-simplex σ on the boundary of voids u and v we denote its corresponding dual edge $\sigma^* = (u^*, v^*)$ and we weight the edges by the capacity function: $c^*(\sigma^*) = c(\sigma)$. Let v_i^* be the vertex dual to the void whose boundary is partitioned by $\mathsf{supp}(\Gamma_1)$ and $\mathsf{supp}(\Gamma_2)$. We split v_i^* into two new vertices denoted s^* and t^*. The edges incident to v_i^* whose dual d-simplices were contained in $\mathsf{supp}(\Gamma_1)$ become incident to s^*, and the edges whose dual d-simplices were contained in $\mathsf{supp}(\Gamma_2)$ become incident to t^*. We add the directed edge (t^*, s^*) and set its capacity to infinity; $c^*((t^*, s^*)) = \infty$. Returning to the analogy of a planar graph with s and t on the same face, splitting v_i^* is analogous to adding an additional edge from t to s which splits their common face into two. However, for our purposes we are only concerned with the algebraic properties of the construction and do not actually need to modify the simplicial complex.

We need to update the chain complex associated with $\mathcal{K}$ to account for the voids and the splitting of v_i^*. We add an additional basis element Σ to $C_d(\mathcal{K})$ such that $\partial\Sigma = \gamma$ and give it infinite capacity; $c(\Sigma) = \infty$. In our construction Σ is dual to the edge (t^*, s^*). In our planar graph analogy Σ plays the role of an edge from t to s drawn entirely in the outer face; to make this precise we will need to add an additional chain group $C_{d+1}(\mathcal{K})$. We add each void V_j with $j \neq i$ as a basis element of $C_{d+1}(\mathcal{K})$ and define the boundary operator as $\partial_{d+1} V_j = \sum_{\sigma \in \mathrm{Bd}(v)} (-1)^{k_\sigma} \sigma$ where $k_\sigma = 0$ if σ is oriented positively on V_j and $k_\sigma = 1$ if σ is oriented negatively on V_j. Next we add additional basis elements S and T whose boundaries are defined by $\partial_{d+1} S = \Gamma_1 + \Sigma$ and $\partial_{d+1} T = \Gamma_2 - \Sigma$. The inclusion of $C_{d+1}(\mathcal{K})$ results in a valid chain complex since by definition the image of ∂_{d+1} under each basis element is a d-cycle. Moreover, in the new complex we have $H_d(\mathcal{K}) \cong 0$ since the boundaries of the voids generate $H_d(\mathcal{K})$.

Given our new chain complex we can extend the dual graph $\mathcal{K}^*$ to a dual complex; this construction is reminiscent of the dual of a polyhedron. We define the dual complex by the isomorphism of chain groups $C_k(\mathcal{K}^*) \cong C_{d-k+1}(\mathcal{K})$. The dual boundary operator $\partial_k^* \colon C_k(\mathcal{K}^*) \to C_{k-1}(\mathcal{K}^*)$ is the coboundary operator δ_{d-k+2}, and the dual coboundary operator $\delta_k^* \colon C_{k-1}(\mathcal{K}^*) \to C_k(\mathcal{K}^*)$ is the boundary operator ∂_{d-k+2}. The primal boundary operator commutes with the dual coboundary operator, and the primal coboundary operator commutes with the dual boundary operator. We summarize the construction with the following commutative diagram.

$$\begin{array}{ccccccc} C_{d+1}(\mathcal{K}) & \underset{\delta_{d+1}}{\overset{\partial_{d+1}}{\longleftrightarrow}} & C_d(\mathcal{K}) & \underset{\delta_d}{\overset{\partial_d}{\longleftrightarrow}} & \cdots & \underset{\delta_1}{\overset{\partial_1}{\longleftrightarrow}} & C_0(\mathcal{K}) \\ \updownarrow\cong & & \updownarrow\cong & & & & \updownarrow\cong \\ C_0(\mathcal{K}^*) & \underset{\partial_1^*}{\overset{\delta_1^*}{\longleftrightarrow}} & C_1(\mathcal{K}^*) & \underset{\partial_2^*}{\overset{\delta_2^*}{\longleftrightarrow}} & \cdots & \underset{\partial_{d+1}^*}{\overset{\delta_{d+1}^*}{\longleftrightarrow}} & C_{d+1}(\mathcal{K}^*) \end{array}$$

We now have enough structure to state our duality theorems. We show that computing a max-flow for $(\mathcal{K}, c, \gamma)$ is equivalent to computing a shortest path from s^* to t^* in $\mathcal{K}^*$ and that computing a minimum cost γ-cut p is equivalent to computing a minimum cost unit s^*t^*-flow in $\mathcal{K}^*$.

Max-flow / shortest path duality. We compute a shortest path from s^* to t^* in $\mathcal{K}^*$ using a well-known shortest paths linear program. Details on the linear program can be found in [9].

$$\begin{array}{ll} \text{maximize} & \mathsf{dist}(t^*) \\ \text{subject to} & \mathsf{dist}(s^*) = 0 \\ & \mathsf{dist}(v^*) - \mathsf{dist}(u^*) \leq c^*((u^*, v^*)) \quad \forall\, (u^*, v^*) \in E \end{array} \tag{LP3}$$

The solution to LP3 is a function $\mathsf{dist}\colon V(\mathcal{K}^*) \to \mathbb{R}$ which maps a vertex to its distance from s^* under the weight function c. By duality, dist is a $(d+1)$-cochain mapping the voids to $\mathbb{R}$. In the following theorem we will show that dist is equivalent to a γ-flow with value equal to $\mathsf{dist}(t^*)$.

▶ **Theorem 16.** *Let $(\mathcal{K}, c, \gamma)$ be a simplicial flow network where $\mathcal{K}$ is a d-dimensional simplicial complex embedded into $\mathbb{R}^{d+1}$ with two unit γ-flows whose supports partition the boundary of some void $\mathrm{Bd}(V_i)$. There is a bijection between γ-flows of $(\mathcal{K}, c, \gamma)$ and s^*t^*-paths in $\mathcal{K}^*$ such that the value of a γ-flow equals the length of its corresponding s^*t^*-path.*

Min-cut / min cost flow duality. We begin this section by stating the minimum cost flow problem in graphs. The minimum cost flow problem asks to find the cheapest way to send k units of flow from s to t. An instance of the minimum cost flow problem is a tuple (G, w, c, k) where $G = (V, E)$ is a directed graph, $w, c \in C_1(G)$, and $k \in \mathbb{R}$. The 1-chains represent the weight and capacity of each edge, and k is the demand of the network. The goal of the minimum cost flow problem is to find an st-flow satisfying both capacity and demand constraints. The demand constraint can be stated as $\delta(t) \cdot f = k$ and ensures that f sends exactly k units of flow from s to t. We will compute a minimum directed γ-cut in $\mathcal{K}$ by solving the minimum cost flow problem with $k = 1$ in $\mathcal{K}^*$. We assume there is a weight function $w\colon \mathcal{K}^d \to \mathbb{R}^+$ on the d-skeleton of $\mathcal{K}$, which after dualizing becomes a weight function w^* on the edges of $\mathcal{K}^*$. In the following theorem the capacity function is not needed, so we will assume each edge in $\mathcal{K}^*$ has infinite capacity.

▶ **Theorem 17.** *Let $\mathcal{K}$ be a d-dimensional simplicial complex embedded into $\mathbb{R}^{d+1}$ with two unit γ-flows whose supports partitions the boundary of some void $\mathrm{Bd}(V_i)$. There is a bijection between γ-cuts p in $\mathcal{K}$ and unit s^*t^*-flows f in $\mathcal{K}^*$ such that $\|p\| = \sum w^*(e)f(e)$.*

▶ **Corollary 18.** *Let $\mathcal{K}$ be a d-dimensional simplicial complex embedded in $\mathbb{R}^{d+1}$ with two unit γ-flows partitioning some $\mathrm{Bd}(V_i)$. There is a polynomial time algorithm computing a minimum directed combinatorial γ-cut.*

Proof. We solve the minimum cost circulation problem in $\mathcal{K}^*$ setting the demand and every capacity constraint equal to one. The resulting flow is dual to a γ-cut p in $\mathcal{K}$. Since the minimum cost circulation is integral we have $\|\mathsf{supp}(\delta(p))\| = \|p\|$. That is, the cost of p as a γ-cut equals the cost of $\mathsf{supp}(\delta(p))$ as a combinatorial γ-cut. ◀

6 Ford-Fulkerson algorithm

In this section we show how the Ford-Fulkerson algorithm can be used to compute a max-flow of simplicial flow network $(\mathcal{K}, c, \gamma)$. In a simplicial flow network the Ford-Fulkerson algorithm picks out a *augmenting chain* at every iteration which is a high dimensional generalization of an augmenting path. As shown in Section 4.2 a max-flow of a simplicial flow network with integral capacities may not be integral, so it is not immediate that Ford-Fulkerson is guaranteed to halt. To remedy this, our implementation of Ford-Fulkerson contains a heuristic reminiscent of the network simplex algorithm. Our heuristic guarantees that at every iteration of Ford-Fulkerson the flow is a solution on a vertex of the polytope defined by the linear program. Hence, our heuristic makes our implementation of Ford-Fulkerson into a special case of the simplex algorithm. It follows that Ford-Fulkerson does halt on a simplicial flow network, but the running time may be exponential. Our heuristic for picking augmenting chains takes $O(n^{\omega+1})$ time since it requires solving $O(n)$ linear systems, each taking $O(n^{\omega})$ time using standard methods [16].

▶ **Definition 19** (Residual complex). *Let $(\mathcal{K}, c, \gamma)$ be a simplicial flow network and f be a feasible flow on the network. We define a new simplicial flow network called the **residual complex** to be the tuple $(\mathcal{K}_f, c_f, \gamma)$ constructed in the following way. The d-skeleton of $\mathcal{K}_f$ is the union $\mathcal{K}^d \cup -\mathcal{K}^d$, that is, for each d-simplex σ in $\mathcal{K}$ we add an additional d-simplex $-\sigma$ whose orientation is opposite of σ. $\mathcal{K}_f^{d'} = \mathcal{K}^{d'}$ for dimensions $d' < d$. The **residual capacity function** $c_f \colon \mathcal{K}_f^d \to \mathbb{R}$ is given by $c_f(\sigma) = \begin{cases} c(\sigma) - f(\sigma) & \sigma \in \mathcal{K}^d \\ f(\sigma) & -\sigma \in \mathcal{K}^d \end{cases}$.*

▶ **Definition 20** (Augmenting chain). *Let $\mathcal{K}_f$ be a residual complex for the simplicial flow network $(\mathcal{K}, c, \gamma)$. An **augmenting chain** is a d-chain $\Gamma \in C_d(\mathcal{K}_f)$ such that $\Gamma = \sum \alpha_i \sigma_i$ and $\partial\Gamma = \gamma$ with $\alpha_i \geq 0$.*

Note that an augmenting chain need not obey the residual capacity constraint c_f. This is because after finding an augmenting chain the amount of flow sent through the chain will be normalized by the coefficients α_i producing a new chain respecting the capacity constraints. We now state the main theorem of the section.

▶ **Theorem 21.** *Let $(\mathcal{K}, c, \gamma)$ be a simplicial flow network. A flow f is a maximum flow if and only if $\mathcal{K}_f$ contains no augmenting chains.*

Augmenting chain heuristic. In this section we provide a heuristic for the Ford-Fulkerson algorithm that is guaranteed to halt on a simplicial flow network. Our example in Section 4.2 shows that a maximum flow may have fractional value, so it's not immediately clear that Ford-Fulkerson halts on all simplicial flow networks. To remedy this our heuristic ensures that at each step the flow corresponds to a vertex of the flow polytope (defined in the next paragraph). As there are a finite number of vertices, and the value of the flow increases at every step, it follows that under this heuristic Ford-Fulkerson must halt. Under our heuristic Ford-Fulkerson becomes a special case of the simplex algorithm. Our heuristic is reminiscent of the network simplex algorithm which maintains a tree at every iteration. See the book by Ahuja, Magnanti, and Orlin for an overview of the network simplex algorithm [1].

We define the *flow polytope* of $(\mathcal{K}, c, \gamma)$ to be the polytope $P \subset \mathbb{R}^{n_d}$ defined by the constraints of the maximum flow linear program LP1. A *vertex* of the polytope P is any feasible solution to LP1 with at least n_d tight linearly independent constraints. We will ensure that at every step of Ford-Fulkerson our flow f is a vertex of P. To do this we will make sure that the d-simplices corresponding to non-tight constraints of LP1 form an acyclic complex. Some straightforward algebra implies that this condition is enough to make at least n_d constraints tight. Let $\mathcal{H}_f$ be the subcomplex of d-simplices "half-saturated" by f; that is, $\sigma \in \mathcal{H}_f$ if and only if its capacity constraint is a strict inequality: $0 < f(\sigma) < c(\sigma)$. The half-saturated simplices do not make either of their two corresponding constraints tight, while d-simplices not in $\mathcal{H}_f$ make exactly one of their corresponding constraints tight. We require that $\mathcal{H}_f$ be an acyclic complex at each step of Ford-Fulkerson. In the case of graphs, this just means that $\mathcal{H}_f$ is a forest. For a d-dimensional complex it means that $\dim H_d(\mathcal{H}_f) = 0$. Acyclic complexes have been studied by Duval, Klivans, and Martin who show that they share many properties with forests and trees in graphs [8]. The following lemma shows that if $\mathcal{H}_f$ is acyclic then f is a vertex of the flow polytope.

▶ **Lemma 22.** *Let f be a feasible flow for the d-dimensional simplicial flow network $(\mathcal{K}, c, \gamma)$. If the subcomplex of half-saturated d-simplices $\mathcal{H}_f$ is acyclic then f is a vertex of the flow polytope P.*

At each iteration of Ford-Fulkerson we want to pick an augmenting chain such that the resulting flow leaves $\mathcal{H}_f$ acyclic. It's not clear how to pick such an augmenting chain. However, no matter what augmenting chain we pick we can always repair the flow in a way that the resulting flow leaves $\mathcal{H}_f$ acyclic. To do so we compute a homology basis of $\mathcal{H}_f$ and update the flow to make $\dim H_d(\mathcal{H}_f) = 0$.

▶ **Lemma 23.** *Let f be a feasible flow for the d-dimensional simplicial flow network $(\mathcal{K}, c, \gamma)$. If the subcomplex of half-saturated d-simplices $\mathcal{H}_f$ is not acyclic then in $O(n^{\omega+1})$ time we can construct a new flow f' such that $\mathcal{H}_{f'}$ is acyclic and $\|f\| = \|f'\|$.*

To wrap up the section, we state our main theorem whose proof is immediate from Lemmas 22 and 23.

▶ **Theorem 24.** *Given a simplicial flow network $(\mathcal{K}, c, \gamma)$ we can compute a maximum flow f by using the Ford-Fulkerson algorithm with the following heuristic: at every iteration pick an augmenting chain such that the subcomplex of half-saturated d-simplices $\mathcal{H}_f$ is acyclic.*

References

1. Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., USA, 1993.
2. Glencora Borradaile and Philip Klein. An $O(n \log n)$ algorithm for maximum st-flow in a directed planar graph. *J. ACM*, 56(2), 2009. doi:10.1145/1502793.1502798.
3. Erin W. Chambers, Jeff Erickson, and Amir Nayyeri. Minimum cuts and shortest homologous cycles. In *Proc. 25th Ann. Symp. Comput. Geom.*, pages 377–385, 2009.
4. Erin W. Chambers, Jeff Erickson, and Amir Nayyeri. Homology flows, cohomology cuts. *SIAM Journal on Computing*, 41(6):1605–1634, 2012. doi:10.1137/090766863.
5. Tamal K. Dey, Anil N. Hirani, and Bala Krishnamoorthy. Optimal homologous cycles, total unimodularity, and linear programming. *SIAM J. Comput.*, 40(4):1026–1044, 2011. doi:10.1137/100800245.
6. Tamal K. Dey, Tao Hou, and Sayan Mandal. Computing minimal persistent cycles: Polynomial and hard cases. In *Proceedings of the Thirty-First Annual ACM-SIAM Symposium on Discrete*

Algorithms, SODA '20, page 2587–2606, USA, 2020. Society for Industrial and Applied Mathematics.

7 Art M. Duval, Caroline J. Klivans, and Jeremy L. Martin. Cuts and flows of cell complexes. *Journal of Algebraic Combinatorics*, 41:969–999, 2015.

8 Art M. Duval, Caroline J. Klivans, and Jeremy L. Martin. *Simplicial and cellular trees*, pages 713–752. Springer International Publishing, Cham, 2016. doi:10.1007/978-3-319-24298-9_28.

9 Jeff Erickson. *Algorithms*. http://algorithms.wtf, 2019.

10 Jeff Erickson and Amir Nayyeri. Minimum cuts and shortest non-separating cycles via homology covers. In *Proc. 22nd Ann. ACM-SIAM Symp. Discrete Algorithms*, pages 1166–1176, 2011.

11 Arthur F. Veinott, Jr. and George B. Dantzig. Integral extreme points. *SIAM Review*, 10(3):371–372, July 1968. doi:10.1137/1010063.

12 Greg N. Frederickson. Fast algorithms for shortest paths in planar graphs with applications. *SIAM J. Comput.*, 16(6):1004–1004, 1987.

13 Refael Hassin. Maximum flow in (s, t) planar networks. *Inf. Process. Lett.*, 13:107, 1981.

14 Refael Hassin and Donald B. Johnson. An $O(n \log^2 n)$ algorithm for maximum flow in undirected planar networks. *SIAM J. Comput.*, 14(3):612–624, 1985.

15 Sergei Ivanov (https://mathoverflow.net/users/4354/sergei ivanov). computational complexity. MathOverflow. URL:https://mathoverflow.net/q/118428 (version: 2013-01-09). arXiv:https://mathoverflow.net/q/118428.

16 Oscar H Ibarra, Shlomo Moran, and Roger Hui. A generalization of the fast LUP matrix decomposition algorithm and applications. *Journal of Algorithms*, 3(1):45–56, 1982. doi:10.1016/0196-6774(82)90007-4.

17 Alon Itai and Yossi Shiloach. Maximum flow in planar networks. *SIAM J. Comput.*, 8:135–150, 1979.

18 Giuseppe F. Italiano, Yahav Nussbaum, Piotr Sankowski, and Christian Wulff-Nilsen. Improved algorithms for min cut and max flow in undirected planar graphs. In *Proc. 43rd Ann. ACM Symp. Theory Comput.*, pages 313–322, 2011.

19 Fabian Latorre. The maxflow problem and a generalization to simplicial complexes, 2012. arXiv:arXiv:1212.1406.

20 Lek-Heng Lim. Hodge Laplacians on graphs. *SIAM Review*, 63(3):685–715, 2020.

21 Sarah Morell, Ina Seidel, and Stefan Weltge. Minimum-cost integer circulations in given homology classes, 2020. arXiv:1911.10912.

22 John Reif. Minimum s-t cut of a planar undirected network in $O(n \log^2 n)$ time. *SIAM J. Comput.*, 12:71–81, 1983.

23 Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, Inc., USA, 1986.