RESEARCH   **Open Access**

Hypergraph cuts with edge-dependent vertex weights

Yu Zhu[*] and Santiago Segarra

*Correspondence:
yz126@rice.edu

Department of Electrical
and Computer Engineering, Rice
University, 6100 Main Street,
Houston 77005, USA

Abstract

We develop a framework for incorporating edge-dependent vertex weights (EDVWs) into the hypergraph minimum *s-t* cut problem. These weights are able to reflect different importance of vertices within a hyperedge, thus leading to better characterized cut properties. More precisely, we introduce a new class of hyperedge splitting functions that we call EDVWs-based, where the penalty of splitting a hyperedge depends only on the sum of EDVWs associated with the vertices on each side of the split. Moreover, we provide a way to construct submodular EDVWs-based splitting functions and prove that a hypergraph equipped with such splitting functions can be reduced to a graph sharing the same cut properties. In this case, the hypergraph minimum *s-t* cut problem can be solved using well-developed solutions to the graph minimum *s-t* cut problem. In addition, we show that an existing sparsification technique can be easily extended to our case and makes the reduced graph smaller and sparser, thus further accelerating the algorithms applied to the reduced graph. Numerical experiments using real-world data demonstrate the effectiveness of our proposed EDVWs-based splitting functions in comparison with the all-or-nothing splitting function and cardinality-based splitting functions commonly adopted in existing work.

Keywords: Hypergraphs, Minimum *s-t* cut, Edge-dependent vertex weights, Hyperedge expansion, Sparsification

Introduction

The *graph* minimum *s-t* cut problem, or equivalently the maximum *s-t* flow problem, is a fundamental problem in network science. A cut is a bipartition of the graph vertices and its weight is computed by summing the weights of the edges crossing the cut. The problem aims to find the minimum weight cut that disconnects the source vertex *s* from the sink vertex *t*. It has various applications such as bipartite matching (Cherkassky et al. 1998; Lovász and Plummer 2009), network reliability (Colbourn 1991; Ramanathan and Colbourn 1987), distributed computing (Bokhari 1987), image segmentation (Boykov and Funka-Lea 2006; Boykov and Kolmogorov 2004) and very large scale integration (VLSI) circuit design (Leighton and Rao 1999; Li et al. 1995). Classical solutions to this problem include "augmenting paths"-based algorithms (Ford and Fulkerson 1956) and "push-relabel" style algorithms (Goldberg and Tarjan 1988), to name a few.

A natural extension is the *hypergraph* minimum *s-t* cut problem. Graphs are limited to modeling pairwise relations, while hypergraphs generalize the notion of an edge to a hyperedge, which can represent higher-order interactions connecting more than two vertices. Many practical problems can be better modeled by hypergraphs (Papa and Markov 2007; Schaub et al. 2021). For instance, in the columnwise decomposition of a sparse matrix for parallel sparse-matrix vector multiplication, hypergraphs provide a more accurate representation for the communication volume requirement than graphs, where vertices and hyperedges are respectively used to model columns of the sparse matrix and the non-zero pattern of each row (Catalyurek and Aykanat 1999). In image segmentation, hypergraphs are leveraged to describe higher-order relations among superpixels (Ding and Yilmaz 2008; Kim et al. 2011). In VLSI circuit design, vertices and hyperedges respectively represent gates and signal nets (Karypis et al. 1999).

The weight of a hypergraph cut is defined as the sum of splitting penalties associated with every hyperedge. Different from the graph case, there may exist multiple ways to split a hyperedge. Consequently, for each hyperedge e, we consider a splitting function $w_e : 2^e \to \mathbb{R}_{\geq 0}$ that assigns a penalty to every possible cut of e where 2^e denotes the power set of e. For any $\mathcal{S} \subseteq e$, $w_e(\mathcal{S})$ indicates the penalty of partitioning e into $\mathcal{S}$ and $e \setminus \mathcal{S}$ (Li and Milenkovic 2017; Veldt et al. 2020a). Existing works mainly adopt two kinds of splitting functions. One is the so-called all-or-nothing splitting function in which an identical penalty is charged if the hyperedge is cut no matter how it is cut (Hein et al. 2013). It is a straightforward extension of the graph case since an edge in a graph is associated with only one non-zero splitting penalty. Another slightly more general type is the class of cardinality-based splitting functions where the splitting penalty $w_e(\mathcal{S})$ depends only on the number of vertices placed into $\mathcal{S}$ (Veldt et al. 2020a; Zhou et al. 2006).

There are two major approaches for solving the minimum *s-t* cut problem in hypergraphs. One is to adopt submodular splitting functions for all hyperedges (cf. (5) for the definition of submodular functions), then the hypergraph minimum *s-t* cut problem can be solved using submodular function minimizers (Li and Milenkovic 2018b; Veldt et al. 2020a). Another more efficient approach is to reduce the hypergraph to a graph that has the same cut properties and then leverage existing solutions to the graph minimum *s-t* cut problem (Ihler et al. 1993; Lawler 1973; Li and Milenkovic 2017; Veldt et al. 2020a). The reduction is generally implemented by expanding every hyperedge into a small graph possibly with additional auxiliary vertices and then concatenating these small graphs to form the final graph. It has been proved in Veldt et al. (2020a) that, for cardinality-based splitting functions, the hypergraph cut problem is reducible to a graph cut problem if and only if the splitting functions are submodular. Moreover, Veldt et al. (2020a) proposes a graph reduction method for an arbitrary submodular cardinality-based splitting function where a hyperedge e is expanded into a graph that has up to $O(|e|)$ auxiliary vertices and $O(|e|^2)$ edges in the worst case. This may result in large and dense graphs thus affecting the efficiency of algorithms applied to the reduced graph. To tackle this problem, sparsification techniques have been developed which try to approximate the hypergraph cut using a sparse graph with fewer auxiliary vertices (Bansal et al. 2019; Benczúr and Karger 1996; Benson et al. 2020; Chekuri and Xu 2018; Kogan and Krauthgamer 2015). A follow-up paper (Benson et al. 2020) proposes a sparsification method for approximating hypergraph cuts defined by submodular cardinality-based splitting

functions. The proposed method reduces the number of auxiliary vertices and the number of edges needed to expand a hyperedge e to $O(\epsilon^{-1} \log |e|)$ and $O(\epsilon^{-1}|e| \log |e|)$ respectively, where ϵ is the approximation tolerance parameter.

A disadvantage of the all-or-nothing splitting function as well as cardinality-based ones is that they treat all the vertices in a hyperedge equally while in practice these vertices might contribute differently to the hyperedge. Such information can be captured by edge-dependent vertex weights (EDVWs): Every vertex v is associated with a weight $\gamma_e(v)$ for each incident hyperedge e that reflects the contribution of v to e (Chitra and Raphael 2019). The hypergraph model with EDVWs is very relevant in practice. For example, an e-commerce system can be modeled as a hypergraph with EDVWs where vertices and hyperedges respectively correspond to users and products, and EDVWs represent the quantity of a product bought by a user (Li et al. 2018). EDVWs can also be used to model the author positions in a co-authored manuscript (Chitra and Raphael 2019), the probability of a pixel belonging to a segment in image segmentation (Ding and Yilmaz 2010), and the relevance of a word to a document in text mining (Hayashi et al. 2020; Zhu et al. 2021), to name a few.

Contributions In this paper, we propose a new class of splitting functions that we call EDVWs-based. In an EDVWs-based splitting function, the splitting penalty $w_e(\mathcal{S})$ depends only on the sum of EDVWs in $\mathcal{S}$, namely $\sum_{v \in \mathcal{S}} \gamma_e(v)$. Hence, we can write $w_e(\mathcal{S}) = g_e(\sum_{v \in \mathcal{S}} \gamma_e(v))$ for some continuous function g_e. We prove that w_e is submodular if g_e is concave. The submodularity is necessary for graph reducibility. We study the EDVWs-based counterparts of four cardinality-based splitting functions in existing work and show that they are graph reducible. Moreover, we prove that any EDVWs-based splitting function with a concave g_e is graph reducible and provide a way for such a reduction. We also show that the sparsification technique proposed in Benson et al. (2020) can be easily adapted to the EDVWs-based case. The size and the density of the reduced graph depend on both the shape of g_e and the EDVWs' values. In a nutshell, our paper provides a framework to study hypergraph cut problems incorporating EDVWs and generalizes the results presented in Benson et al. (2020); Veldt et al. (2020a) from cardinality-based splitting functions to EDVWs-based ones.

Paper outline The rest of this paper is structured as follows. Preliminary concepts and related work about graph and hypergraph cut problems are reviewed in Sect. 2. The main theoretical results are presented in Sect. 3, where the hypergraph model with EDVWs is introduced in Sect. 3.1, the proposed EDVWs-based splitting functions are studied in Sect. 3.2, the graph reducibility results are stated in Sect. 3.3, and the sparsification technique is discussed in Sect. 3.4. The numerical results shown in Sect. 4 validate the effectiveness of introducing EDVWs into hypergraph cuts. Closing remarks are included in Sect. 5.

Preliminaries and related work
Graph cuts
Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ denote a weighted and possibly directed graph where $\mathcal{V}$ is the vertex set, $\mathcal{E}$ is the edge set, and $\mathbf{W}$ is the weighted adjacency matrix whose entry W_{uv} denotes the weight of the edge from u to v. A cut is a partition of the vertex set $\mathcal{V}$ into

two disjoint, non-empty subsets denoted by $\mathcal{S}$ and its complement $\mathcal{V} \setminus \mathcal{S}$. The weight of the cut is defined as

$$\text{cut}_{\mathcal{G}}(\mathcal{S}) = \sum_{u \in \mathcal{S}, v \in \mathcal{V} \setminus \mathcal{S}} W_{uv}. \tag{1}$$

Given two vertices s, t in the graph, the minimum s-t cut problem aims to find the minimum weight cut that separates s and t. Formally, the problem can be written as

$$\min_{\emptyset \subset \mathcal{S} \subset \mathcal{V}} \text{cut}_{\mathcal{G}}(\mathcal{S}) \quad \text{s.t. } s \in \mathcal{S}, t \in \mathcal{V} \setminus \mathcal{S}. \tag{2}$$

The minimum s-t cut problem is the dual of the maximum s-t flow problem. There exist a number of algorithms for the min-cut/max-flow problem and a summary can be found in Goldberg (1998). Moreover, it is established in Orlin (2013) that the min-cut/max-flow problem is solvable in $O(|\mathcal{V}||\mathcal{E}|)$ time. We also notice that a recent paper (Chen et al. 2022) provides an algorithm that solves this problem in almost-linear time.

Hypergraph cuts

Let $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ be a hypergraph where $\mathcal{V}$ and $\mathcal{E}$ respectively denote the vertex set and the hyperedge set. Unlike the graph case, a hyperedge can connect more than two vertices thus there may exist multiple ways to split a hyperedge. For each hyperedge $e \in \mathcal{E}$, we introduce a splitting function $w_e : 2^e \to \mathbb{R}_{\geq 0}$ that assigns a non-negative penalty to every possible cut of e (Li and Milenkovic 2017; Veldt et al. 2020a). The splitting function satisfies $w_e(\emptyset) = w_e(e) = 0$, in other words, a penalty of zero is assigned when the hyperedge is not cut. Moreover, the splitting function is symmetric if it satisfies $w_e(\mathcal{S}) = w_e(e \setminus \mathcal{S})$ for any $\mathcal{S} \subseteq e$. The weight of the hypergraph cut induced by $\mathcal{S} \subseteq \mathcal{V}$ is defined as the sum of splitting penalties associated with every hyperedge, i.e.,

$$\text{cut}_{\mathcal{H}}(\mathcal{S}) = \sum_{e \in \mathcal{E}} w_e(\mathcal{S} \cap e). \tag{3}$$

There are mainly two types of splitting functions in existing work: (i) An *all-or-nothing* splitting function assigns the same penalty to every possible cut of the hyperedge regardless of how its vertices are separated (Hein et al. 2013). More precisely, $w_e(\mathcal{S})$ is equal to some positive constant, e.g., the hyperedge weight, for all non-empty $\mathcal{S} \subset e$ and $w_e(\mathcal{S}) = 0$ if $\mathcal{S} \in \{\emptyset, e\}$. (ii) A splitting function is *cardinality-based* if $w_e(\mathcal{S}_1) = w_e(\mathcal{S}_2)$ for all $\mathcal{S}_1, \mathcal{S}_2 \subseteq e$ whenever $|\mathcal{S}_1| = |\mathcal{S}_2|$ (Veldt et al. 2020a). In other words, the value of $w_e(\mathcal{S})$ depends only on the cardinality of $\mathcal{S}$. Several examples are given in Table 1.

Table 1 Examples of cardinality-based and EDVWs-based splitting functions and their corresponding gadgets where $\mathcal{S}$ is a subset of the hyperedge e and a, b are positive constants

Cardinality-based	EDVWs-based	Corresponding gadget
$w_e(\mathcal{S}) = \lvert\mathcal{S}\rvert \cdot \lvert e \setminus \mathcal{S}\rvert$	$w_e(\mathcal{S}) = \gamma_e(\mathcal{S}) \cdot \gamma_e(e \setminus \mathcal{S})$	Clique gadget
$w_e(\mathcal{S}) = \min\{\lvert\mathcal{S}\rvert, \lvert e \setminus \mathcal{S}\rvert\}$	$w_e(\mathcal{S}) = \min\{\gamma_e(\mathcal{S}), \gamma_e(e \setminus \mathcal{S})\}$	Star gadget
$w_e(\mathcal{S}) = \min\{\lvert\mathcal{S}\rvert, \lvert e \setminus \mathcal{S}\rvert, b\}$	$w_e(\mathcal{S}) = \min\{\gamma_e(\mathcal{S}), \gamma_e(e \setminus \mathcal{S}), b\}$	Sym. cardinality/EDVWs-based gadget
$w_e(\mathcal{S}) = \min\{a\lvert\mathcal{S}\rvert, b\lvert e \setminus \mathcal{S}\rvert\}$	$w_e(\mathcal{S}) = \min\{a\gamma_e(\mathcal{S}), b\gamma_e(e \setminus \mathcal{S})\}$	Asym. cardinality/EDVWs-based gadget

Similar to the graph case, the hypergraph minimum *s-t* cut problem is formulated as

$$\min_{\emptyset \subset \mathcal{S} \subset \mathcal{V}} \text{cut}_{\mathcal{H}}(\mathcal{S}) \quad \text{s.t. } s \in \mathcal{S}, t \in \mathcal{V} \setminus \mathcal{S}. \tag{4}$$

For a finite set $\mathcal{S}$, a set function $F : 2^{\mathcal{S}} \to \mathbb{R}$ is called *submodular* if

$$F(\mathcal{S}_1 \cup \{v\}) - F(\mathcal{S}_1) \geq F(\mathcal{S}_2 \cup \{v\}) - F(\mathcal{S}_2) \tag{5}$$

for every $\mathcal{S}_1 \subseteq \mathcal{S}_2 \subset \mathcal{S}$ and every $v \in \mathcal{S} \setminus \mathcal{S}_2$ (Bach 2013). If the splitting function w_e associated with every hyperedge $e \in \mathcal{E}$ is submodular, the resulting hypergraph cut in the form of a sum of submodular functions is also submodular. In this case, the hypergraph minimum *s-t* cut problem can be solved using general submodular function minimizers (Grötschel et al. 1981; Iwata 2003; Iwata et al. 2001; Iwata and Orlin 2009; Orlin 2009; Schrijver 2000) or minimizers for decomposable submodular functions (Ene et al. 2017; Kolmogorov 2012; Li and Milenkovic 2018a). The all-or-nothing splitting function and the cardinality-based ones listed in Table 1 are all submodular.

Graph reducibility

Since algorithms for the graph minimum *s-t* cut problem are more efficient than algorithms for general submodular function minimization, another way of solving hypergraph cut problems is to reduce the hypergraph to a graph that shares the same or similar cut properties (Ihler et al. 1993; Lawler 1973; Li and Milenkovic 2017). The reduction is generally accomplished via hyperedge expansions. Recently, a generalized hyperedge expansion has been formulated in Veldt et al. (2020a), which projects a hyperedge onto a graph allowing directed edges and additional vertices. The formal definition is given below.

Definition 1   (*Gadget splitting function* (Veldt et al. 2020a)) A gadget associated with a hyperedge e is a weighted and possibly directed graph $\mathcal{G}_e = (\mathcal{V}', \mathcal{E}')$ with vertex set $\mathcal{V}' = e \cup \hat{\mathcal{V}}$ where $\hat{\mathcal{V}}$ is a set of auxiliary vertices. The corresponding gadget splitting function $\hat{w}_e : 2^e \to \mathbb{R}_{\geq 0}$ is defined as

$$\hat{w}_e(\mathcal{S}) = \min_{\mathcal{T} \subseteq \mathcal{V}', \mathcal{T} \cap e = \mathcal{S}} \text{cut}_{\mathcal{G}_e}(\mathcal{T}). \tag{6}$$

A hyperedge splitting function is *graph reducible* if it is identical to some gadget splitting function. A hypergraph cut function defined as (3) or the hypergraph minimum *s-t* cut problem is graph reducible if all its hyperedge splitting functions are graph reducible. It has been proved in Veldt et al. (2020a) that every gadget splitting function $\hat{w}_e$ defined as (6) is submodular. Hence, if a hyperedge splitting function is graph reducible, it must be submodular.

In the following, we give several examples of splitting functions that have been shown to be graph reducible in existing works. The all-or-nothing splitting function is graph reducible and can be constructed from the Lawler gadget described as follows.

Lawler gadget (Lawler 1973). The Lawler gadget replaces a hyperedge e with a digraph defined on the vertex set $\mathcal{V}' = e \cup \{e', e''\}$ where e', e'' are two auxiliary vertices. For each $v \in e$, add a directed edge of weight infinity from v to e' and a directed edge of weight

infinity from e'' to v. Finally, add a directed edge of weight equal to the hyperedge weight from e' to e''.

The cardinality-based splitting functions listed in Table 1 are all graph reducible and correspond to the following gadgets, respectively.

Clique gadget (Agarwal et al. 2006). This gadget is an (undirected) clique graph with vertex set $\mathcal{V}' = e$. For every $u, v \in e$, add an edge of weight 1 between u and v.

Star gadget (Agarwal et al. 2006). This gadget is an (undirected) star graph with vertex set $\mathcal{V}' = e \cup \{v_e\}$ where v_e is an auxiliary vertex. For each $v \in e$, add an edge of weight 1 between v and v_e.

Symmetric cardinality-based gadget (Veldt et al. 2020a). Similar to the Lawler gadget, this gadget is a digraph with vertex set $\mathcal{V}' = e \cup \{e', e''\}$. For each $v \in e$, add a directed edge of weight 1 from v to e' and a directed edge of weight 1 from e'' to v. Moreover, add a directed edge of weight $b \in \mathbb{N}$ from e' to e''.

Asymmetric cardinality-based gadget (Veldt et al. 2020a). This gadget is a digraph defined on $\mathcal{V}' = e \cup \{v_e\}$. For each $v \in e$, add a directed edge of weight a from v to v_e and a directed edge of weight b from v_e to v.

Hypergraph cuts with EDVWs
The hypergraph model with EDVWs
In this paper, we consider the hypergraph model with EDVWs as defined next.

Definition 2 (*Hypergraph with EDVWs* (Chitra and Raphael 2019)) Let $\mathcal{H} = (\mathcal{V}, \mathcal{E}, \kappa, \{\gamma_e\})$ be a hypergraph with EDVWs where $\mathcal{V}$ and $\mathcal{E}$ respectively denote the vertex set and the hyperedge set. The function $\kappa : \mathcal{E} \to \mathbb{R}_+$ assigns positive weights to hyperedges and those weights reflect the strength of connection. Each hyperedge $e \in \mathcal{E}$ is associated with a function $\gamma_e : e \to \mathbb{R}_+$ to assign EDVWs. For convenience, we define $\gamma_e(\mathcal{S}) = \sum_{v \in \mathcal{S}} \gamma_e(v)$ for any $\mathcal{S} \subseteq e$.

The introduction of EDVWs enables the hypergraph to model the cases when the vertices in the same hyperedge contribute differently to this hyperedge. For example, in a coauthorship network, every author (vertex) in general has a different degree of contribution to a paper (hyperedge), usually reflected by the order of the authors. This information is lost in traditional hypergraph models but it can be easily encoded through EDVWs. In the following, we study how to incorporate EDVWs into hypergraph cut problems.

EDVWs-based splitting functions
A natural extension from cardinality-based splitting functions to EDVWs-based ones is to make the splitting penalty $w_e(\mathcal{S})$ dependent only on the sum of EDVWs in $\mathcal{S}$.

Definition 3 (*EDVWs-based splitting function*) We refer to splitting functions defined in the following way as EDVWs-based splitting functions:

$$w_e(\mathcal{S}) = g_e(\gamma_e(\mathcal{S})), \quad \forall \mathcal{S} \subseteq e, \tag{7}$$

where $g_e : [0, \gamma_e(e)] \to \mathbb{R}_{\geq 0}$ satisfies $g_e(0) = g_e(\gamma_e(e)) = 0$.

For trivial EDVWs, namely $\gamma_e(v) = 1$ for all $v \in e$, we have $\gamma_e(\mathcal{S}) = |\mathcal{S}|$ and the EDVWs-based splitting function reduces to a cardinality-based one. Actually, g_e can be viewed as a continuous extension of the splitting function w_e. In practice, we can also incorporate the hyperedge weight $\kappa(e)$ into the splitting function such as setting $w_e(\mathcal{S}) = \kappa(e) \cdot g_e(\gamma_e(\mathcal{S}))$. This does not influence the results presented in this paper.

We are interested in submodular splitting functions which make it possible to leverage existing solvers for submodular function minimization. Moreover, as mentioned in Sect. 2.3, submodularity is a necessary condition for a splitting function to be graph reducible (Veldt et al. 2020a). In the following Theorem 3.2, we show that the EDVWs-based splitting function defined as (7) is submodular if g_e is concave. We first present several properties of concave functions which will be used later.

Lemma 3.1   (Properties of concave functions) *For a concave function g, we have*

(i)   *If $b_1 \leq b_2$, $a > 0$, the inequality $g(b_1 + a) - g(b_1) \geq g(b_2 + a) - g(b_2)$ holds.*
(ii)   *If $b_1 < b_2 < b_3$, the inequality $g(b_2) \geq \frac{b_3 - b_2}{b_3 - b_1}g(b_1) + \frac{b_2 - b_1}{b_3 - b_1}g(b_3)$ holds.*
(iii)   *If $b_1 < b_2 \leq a$ and g is symmetric with respect to a, the inequality $g(b_1) \leq g(b_2)$ holds.*

Proof

According to the definition of concave functions, the following inequality holds for any x and y in the domain of a concave function g,

$$g(tx + (1 - t)y) \geq tg(x) + (1 - t)g(y), \quad \forall t \in [0, 1].$$

To prove Property (i), we set $x = b_1$ and $y = b_2 + a$. For $t = \frac{b_2 - b_1}{b_2 - b_1 + a}$ and $t = \frac{a}{b_2 - b_1 + a}$, the above inequality respectively becomes

$$g(b_1 + a) \geq \frac{b_2 - b_1}{b_2 - b_1 + a}g(b_1) + \frac{a}{b_2 - b_1 + a}g(b_2 + a),$$
$$g(b_2) \geq \frac{a}{b_2 - b_1 + a}g(b_1) + \frac{b_2 - b_1}{b_2 - b_1 + a}g(b_2 + a).$$

Property (i) can be obtained by respectively adding both sides of these two inequalities together. Property (ii) can be proved by setting $x = b_1$, $y = b_3$ and $t = \frac{b_3 - b_2}{b_3 - b_1}$. Property (iii) can be proved by setting $x = b_1$, $y = 2a - b_1$ and $t = \frac{2a - b_1 - b_2}{2a - 2b_1}$. $\square$

Theorem 3.2   *For EDVWs-based splitting functions defined as (7), if g_e is a concave function, then w_e is submodular. If g_e is concave as well as symmetric with respect to $\gamma_e(e)/2$, then w_e is submodular and symmetric.*

Proof

For every $\mathcal{S}_1 \subseteq \mathcal{S}_2 \subset e$ and every $v \in e \setminus \mathcal{S}_2$, set $b_1 = \gamma_e(\mathcal{S}_1)$, $b_2 = \gamma_e(\mathcal{S}_2)$ and $a = \gamma_e(v)$. When g_e is concave, it follows from (i) in Lemma 3.1 that

$$g_e(\gamma_e(\mathcal{S}_1) + \gamma_e(v)) - g_e(\gamma_e(\mathcal{S}_1)) \geq g_e(\gamma_e(\mathcal{S}_2) + \gamma_e(v)) - g_e(\gamma_e(\mathcal{S}_2)).$$

It immediately follows from (7) that

$$w_e(\mathcal{S}_1 \cup \{v\}) - w_e(\mathcal{S}_1) \geq w_e(\mathcal{S}_2 \cup \{v\}) - w_e(\mathcal{S}_2).$$

Hence, w_e is submodular according to the definition of submodular functions. Moreover, it is straightforward to show that w_e satisfies $w_e(\mathcal{S}) = w_e(e \setminus \mathcal{S})$ for all $\mathcal{S} \subseteq e$ if g_e is symmetric with respect to $\gamma_e(e)/2$. $\square$

Graph reducibility of EDVWs-based splitting functions

We first consider the following concave functions for g_e.

$$g_e(x) = x \cdot (\gamma_e(e) - x), \tag{8}$$

$$g_e(x) = \min\{x, \gamma_e(e) - x\}, \tag{9}$$

$$g_e(x) = \min\{x, \gamma_e(e) - x, b\}, \ \ b > 0, \tag{10}$$

$$g_e(x) = \min\{ax, b(\gamma_e(e) - x)\}, \ \ a, b > 0. \tag{11}$$

By substituting these concave functions into (7), we obtain the EDVWs-based splitting functions listed in Table 1, which are submodular according to Theorem 3.2. The first three are also symmetric while the last one is generally asymmetric unless $a = b$. For the third one, if the parameter b is set to some value no greater than $\min_{v \in e} \gamma_e(v)$, it reduces to the all-or-nothing splitting function; while if $b \geq \max_{\mathcal{S} \subseteq e} \min\{\gamma_e(\mathcal{S}), \gamma_e(e) - \gamma_e(\mathcal{S})\}$, it reduces to the second one. We are going to show that the EDVWs-based splitting functions listed in Table 1 are also graph reducible by constructing gadgets that generalize the gadgets introduced in Sect. 2.3. An illustration of these generalized gadgets is given in Fig. 1.

Theorem 3.3 *The EDVWs-based splitting functions listed in Table 1 are all graph reducible and respectively correspond to the gadgets described in Table 2.*

Table 2 Examples of gadgets $\mathcal{G}_e = (e \cup \hat{\mathcal{V}}, \mathcal{E}')$ incorporating EDVWs

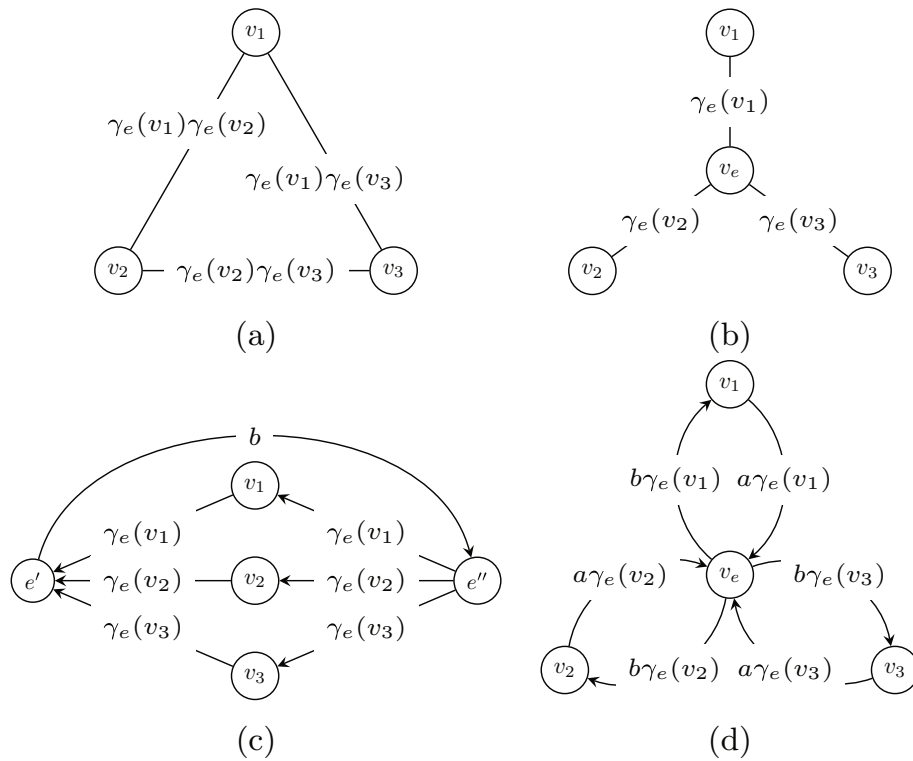| Gadget name	Type	$\hat{\mathcal{V}}$	$|\mathcal{E}'|$	Description
(EDVWs-based) Clique gadget	Undirected	$\emptyset$	$|e|(|e| - 1)/2$	For every $u, v \in e$, add an edge (u, v) of weight $\gamma_e(u)\gamma_e(v)$.
(EDVWs-based) Star gadget	Undirected	$\{v_e\}$	$|e|$	For every $v \in e$, add an edge (v, v_e) of weight $\gamma_e(v)$.
Sym. EDVWs-based gadget	Directed	$\{e', e''\}$	$2|e| + 1$	For every $v \in e$, add a directed edge of weight $\gamma_e(v)$ from v to e' and a directed edge of weight $\gamma_e(v)$ from e'' to v. Moreover, add a directed edge of weight b from e' to e''.
Asym. EDVWs-based gadget	Directed	$\{v_e\}$	$2|e|$	For every $v \in e$, add a directed edge of weight $a\gamma_e(v)$ from v to v_e and a directed edge of weight $b\gamma_e(v)$ from v_e to v.

Fig. 1 The illustration of (**a**) a clique gadget, (**b**) a star gadget, (**c**) a symmetric EDVWs-based gadget, and (**d**) an asymmetric EDVWs-based gadget, for a hyperedge $e = \{v_1, v_2, v_3\}$

Proof

In the following, we prove these four cases one by one.

(i) It follows from (6) that the splitting function constructed from the clique gadget is

$$\hat{w}_e(\mathcal{S}) = \mathrm{cut}_{\mathcal{G}_e}(\mathcal{S}) = \sum_{u \in \mathcal{S}, v \in e \setminus \mathcal{S}} \gamma_e(u)\gamma_e(v) = \sum_{u \in \mathcal{S}} \gamma_e(u) \cdot \sum_{v \in e \setminus \mathcal{S}} \gamma_e(v) = \gamma_e(\mathcal{S}) \cdot \gamma_e(e \setminus \mathcal{S}).$$

(ii) According to (6), there are two ways to split $\mathcal{S}$ from e in the star gadget: set $\mathcal{T} = \mathcal{S}$ or $\mathcal{T} = \mathcal{S} \cup \{v_e\}$. The corresponding graph cut weights $\mathrm{cut}_{\mathcal{G}_e}(\mathcal{T})$ are respectively equal to $\gamma_e(\mathcal{S})$ and $\gamma_e(e \setminus \mathcal{S})$. Take the minimum one and the result follows.

(iii) For the symmetric EDVWs-based gadget, there are four ways to split $\mathcal{S}$ from e: set $\mathcal{T} = \mathcal{S}$, $\mathcal{T} = \mathcal{S} \cup \{e', e''\}$, $\mathcal{T} = \mathcal{S} \cup \{e'\}$ or $\mathcal{T} = \mathcal{S} \cup \{e''\}$. They respectively result in $\mathrm{cut}_{\mathcal{G}_e}(\mathcal{T})$ equal to $\gamma_e(\mathcal{S})$, $\gamma_e(e \setminus \mathcal{S})$, b and $\gamma_e(e)$. Notice that the last one is always no less than the first two. The result follows by taking the minimum one.

(iv) To split $\mathcal{S}$ from e in the asymmetric EDVWs-based gadget, we can set $\mathcal{T} = \mathcal{S}$ or $\mathcal{T} = \mathcal{S} \cup \{v_e\}$ which respectively lead to $\mathrm{cut}_{\mathcal{G}_e}(\mathcal{T})$ equal to $a\gamma_e(\mathcal{S})$ and $b\gamma_e(e \setminus \mathcal{S})$. The proof is completed by taking the minimum one.   □

It has been proved in Veldt et al. (2020a) that all submodular cardinality-based splitting functions are graph reducible. More precisely, any submodular cardinality-based splitting

function can be constructed from a combination of up to $|e| - 1$ different asymmetric cardinality-based gadgets. For a symmetric submodular cardinality-based splitting function, it can also be constructed from a combination of up to $\lfloor |e|/2 \rfloor$ symmetric cardinality-based gadgets. In Theorem 3.4 below, we show graph reducibility for the proposed submodular EDVWs-based splitting functions. To this end, we first define two sets as follows.

$$\mathcal{Q}_s = \{\gamma_e(\mathcal{S}) | \mathcal{S} \subseteq e, 0 < \gamma_e(\mathcal{S}) \leq \gamma_e(e)/2\}, \tag{12}$$

$$\mathcal{Q}_a = \{\gamma_e(\mathcal{S}) | \emptyset \subset \mathcal{S} \subset e\}. \tag{13}$$

Denote by $\mathcal{S}_i$ the subset of e corresponding to the ith smallest element in $\mathcal{Q}_a$. The set $\mathcal{Q}_s$ is a subset of $\mathcal{Q}_a$ and contains the smallest $|\mathcal{Q}_s|$ elements in $\mathcal{Q}_a$.

Theorem 3.4   *All EDVWs-based splitting functions defined as* (7) *with concave g_e are graph reducible. A hyperedge paired with such a splitting function can be reduced to a graph which is a combination of at most $|\mathcal{Q}_a|$ asymmetric EDVWs-based gadgets. If, in addition, g_e is symmetric, the hyperedge can also be reduced to a graph combining at most $|\mathcal{Q}_s|$ symmetric EDVWs-based gadgets.*

Proof
We first consider the case when g_e is concave and symmetric, then study the more general case when g_e is concave and possibly asymmetric.

(i) Consider a hyperedge e and an EDVWs-based splitting function w_e with concave, symmetric g_e. We are going to show that w_e corresponds to some gadget splitting function $\hat{w}_e$ and one way for constructing such a gadget is to combine $r = |\mathcal{Q}_s|$ symmetric EDVWs-based gadgets. For the ith symmetric EDVWs-based gadget, denote the two auxiliary vertices by e_i' and e_i'', set the weight of the edge from e_i' to e_i'' to $b_i = \gamma_e(\mathcal{S}_i)$, then scale all edge weights by a factor $a_i \geq 0$. The combined gadget contains $|e| + 2|\mathcal{Q}_s|$ vertices and $(2|e| + 1)|\mathcal{Q}_s|$ edges. Its corresponding splitting function can be written as

$$\hat{w}_e(\mathcal{S}) = \sum_{i=1}^r a_i \cdot \min\{\gamma_e(\mathcal{S}), \gamma_e(e \setminus \mathcal{S}), b_i\}. \tag{14}$$

By substituting $\mathcal{S}_i$ into (14) we get

$$\hat{w}_e(\mathcal{S}_i) = \sum_{j=1}^r a_j \cdot \min\{b_i, \gamma_e(e) - b_i, b_j\}$$
$$= \sum_{j=1}^i a_j b_j + \sum_{j=i+1}^r a_j b_i, \quad \forall i = 1, \cdots, r,$$

which can be condensed in the following matrix form

$$\underbrace{\begin{bmatrix} b_1 & b_1 & b_1 & b_1 & \cdots & b_1 \\ b_1 & b_2 & b_2 & b_2 & \cdots & b_2 \\ b_1 & b_2 & b_3 & b_3 & \cdots & b_3 \\ b_1 & b_2 & b_3 & b_4 & \cdots & b_4 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ b_1 & b_2 & b_3 & b_4 & \cdots & b_r \end{bmatrix}}_{\mathbf{B}_1} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ \vdots \\ a_r \end{bmatrix} = \begin{bmatrix} \hat{w}_e(\mathcal{S}_1) \\ \hat{w}_e(\mathcal{S}_2) \\ \hat{w}_e(\mathcal{S}_3) \\ \hat{w}_e(\mathcal{S}_4) \\ \vdots \\ \hat{w}_e(\mathcal{S}_r) \end{bmatrix}. \tag{15}$$

The question left is whether there exist non-negative $a_1, \cdots, a_r$ such that $\hat{w}_e(\mathcal{S}_i) = w_e(\mathcal{S}_i)$ for all $i \in [r]$. To identify such a_i, we replace $\hat{w}_e(\mathcal{S}_i)$ with $w_e(\mathcal{S}_i)$ and invert the system (15) as follows

$$
\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{r-1} \\ a_r \end{bmatrix} = \mathbf{B}_1^{-1} \begin{bmatrix} w_e(\mathcal{S}_1) \\ w_e(\mathcal{S}_2) \\ \vdots \\ w_e(\mathcal{S}_{r-1}) \\ w_e(\mathcal{S}_r) \end{bmatrix},
$$

where

$$
\mathbf{B}_1^{-1} = \begin{bmatrix}
\frac{b_2}{(b_2-b_1)b_1} & -\frac{1}{b_2-b_1} & 0 & \cdots & 0 & 0 & 0 \\
-\frac{1}{b_2-b_1} & \frac{b_3-b_1}{(b_3-b_2)(b_2-b_1)} & -\frac{1}{b_3-b_2} & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & -\frac{1}{b_{r-1}-b_{r-2}} & \frac{b_r-b_{r-2}}{(b_r-b_{r-1})(b_{r-1}-b_{r-2})} & -\frac{1}{b_r-b_{r-1}} \\
0 & 0 & 0 & \cdots & 0 & -\frac{1}{b_r-b_{r-1}} & \frac{1}{b_r-b_{r-1}}
\end{bmatrix}.
$$

It follows that

$$
\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{r-1} \\ a_r \end{bmatrix} = \begin{bmatrix}
\frac{b_2}{(b_2-b_1)b_1}w_e(\mathcal{S}_1) - \frac{1}{b_2-b_1}w_e(\mathcal{S}_2) \\
\frac{b_3-b_1}{(b_3-b_2)(b_2-b_1)}w_e(\mathcal{S}_2) - \frac{1}{b_2-b_1}w_e(\mathcal{S}_1) - \frac{1}{b_3-b_2}w_e(\mathcal{S}_3) \\
\vdots \\
\frac{b_r-b_{r-2}}{(b_r-b_{r-1})(b_{r-1}-b_{r-2})}w_e(\mathcal{S}_{r-1}) - \frac{1}{b_{r-1}-b_{r-2}}w_e(\mathcal{S}_{r-2}) - \frac{1}{b_r-b_{r-1}}w_e(\mathcal{S}_r) \\
\frac{1}{b_r-b_{r-1}}w_e(\mathcal{S}_r) - \frac{1}{b_r-b_{r-1}}w_e(\mathcal{S}_{r-1})
\end{bmatrix}.
$$

Since $a_1, \cdots, a_r$ need to be non-negative, we are left to prove the following inequalities:

$$
w_e(\mathcal{S}_1) \geq \frac{b_1}{b_2}w_e(\mathcal{S}_2), \tag{16}
$$

$$
w_e(\mathcal{S}_i) \geq \frac{b_{i+1}-b_i}{b_{i+1}-b_{i-1}}w_e(\mathcal{S}_{i-1}) + \frac{b_i-b_{i-1}}{b_{i+1}-b_{i-1}}w_e(\mathcal{S}_{i+1}), \quad \forall i = 2, \cdots, r-1, \tag{17}
$$

$$
w_e(\mathcal{S}_r) \geq w_e(\mathcal{S}_{r-1}). \tag{18}
$$

Moreover, it can be observed from (15) that $\hat{w}_e(\mathcal{S}_1) \leq \hat{w}_e(\mathcal{S}_2) \leq \cdots \leq \hat{w}_e(\mathcal{S}_r)$ due to the structure of $\mathbf{B}_1$ and the non-negativity of coefficients a_i. Hence, we also need to show that

$$
w_e(\mathcal{S}_1) \leq w_e(\mathcal{S}_2) \leq \cdots \leq w_e(\mathcal{S}_r). \tag{19}
$$

By introducing $b_0 = 0$, the inequalities (16), (17) can be rewritten as

$$
g_e(b_i) \geq \frac{b_{i+1}-b_i}{b_{i+1}-b_{i-1}}g_e(b_{i-1}) + \frac{b_i-b_{i-1}}{b_{i+1}-b_{i-1}}g_e(b_{i+1}), \quad \forall i = 1, \cdots, r-1. \tag{20}
$$

It follows immediately from (ii) in Lemma 3.1. The inequalities (19) (including (18)) can be rewritten as

$$
g_e(b_1) \leq g_e(b_2) \leq \cdots \leq g_e(b_r), \tag{21}
$$

which can be proved according to (iii) in Lemma 3.1. Notice that, when there exists any equality in (16)–(18), the corresponding a_i equals 0, which implies that the number of symmetric EDVWs-based gadgets needed to construct the combined gadget can be further reduced. The equality in (17) (or see (20)) means that the points at b_{i-1}, b_i and b_{i+1} are colinear.

(ii) Next we consider the case when g_e is concave and possibly asymmetric. In this case, w_e can be shown to be identical to some gadget splitting function $\hat{w}_e$ and such a gadget can be constructed by combing $r = |\mathcal{Q}_a|$ asymmetric EDVWs-based gadgets. The ith one is paired with parameters $a_i(\gamma_e(e) - b_i)$ and $a_i b_i$ where $a_i \geq 0$ is a scaling parameter. The combined gadget consists of $|e| + |\mathcal{Q}_a|$ vertices and $2|e| \cdot |\mathcal{Q}_a|$ edges. Its corresponding splitting function can be written as

$$\hat{w}_e(\mathcal{S}) = \sum_{i=1}^{r} a_i \cdot \min\{(\gamma_e(e) - b_i)\gamma_e(\mathcal{S}), b_i\gamma_e(e \setminus \mathcal{S})\}. \tag{22}$$

We set $b_i = \gamma_e(\mathcal{S}_i)$ for all $i \in [r]$. Then it holds that $b_i + b_{r+1-i} = \gamma_e(e)$. By substituting $\mathcal{S}_i$ into (22) we get

$$\hat{w}_e(\mathcal{S}_i) = \sum_{j=1}^{r} a_j \cdot \min\{(\gamma_e(e) - b_j)b_i, b_j(\gamma_e(e) - b_i)\},$$
$$= \sum_{j=1}^{i} a_j b_j b_{r+1-i} + \sum_{j=i+1}^{r} a_j b_{r+1-j} b_i, \quad \forall i = 1, \cdots, r,$$

which can also be written in the following matrix from

$$\underbrace{\begin{bmatrix}
b_1 b_r & b_1 b_{r-1} & b_1 b_{r-2} & b_1 b_{r-3} & \cdots & b_1 b_2 & b_1 b_1 \\
b_1 b_{r-1} & b_2 b_{r-1} & b_2 b_{r-2} & b_2 b_{r-3} & \cdots & b_2 b_2 & b_2 b_1 \\
b_1 b_{r-2} & b_2 b_{r-2} & b_3 b_{r-2} & b_3 b_{r-3} & \cdots & b_3 b_2 & b_3 b_1 \\
b_1 b_{r-3} & b_2 b_{r-3} & b_3 b_{r-3} & b_4 b_{r-3} & \cdots & b_4 b_2 & b_4 b_1 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
b_1 b_2 & b_2 b_2 & b_3 b_2 & b_4 b_2 & \cdots & b_{r-1} b_2 & b_{r-1} b_1 \\
b_1 b_1 & b_2 b_1 & b_3 b_1 & b_4 b_1 & \cdots & b_{r-1} b_1 & b_r b_1
\end{bmatrix}}_{\mathbf{B}_2}
\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ \vdots \\ a_{r-1} \\ a_r \end{bmatrix}
=
\begin{bmatrix} \hat{w}_e(\mathcal{S}_1) \\ \hat{w}_e(\mathcal{S}_2) \\ \hat{w}_e(\mathcal{S}_3) \\ \hat{w}_e(\mathcal{S}_4) \\ \vdots \\ \hat{w}_e(\mathcal{S}_{r-1}) \\ \hat{w}_e(\mathcal{S}_r) \end{bmatrix}. \tag{23}$$

Replace $\hat{w}(\mathcal{S}_i)$ with $w(\mathcal{S}_i)$ and invert the system (23) to find the valid coefficients a_i. For convenience, we introduce $b_{r+1} = \gamma_e(e)$. The inverse of $\mathbf{B}_2$ can be written as

$$\frac{1}{b_{r+1}} \cdot \begin{bmatrix}
\frac{b_2}{(b_2 - b_1)b_1} & -\frac{1}{b_2 - b_1} & 0 & \cdots & 0 & 0 & 0 \\
-\frac{1}{b_2 - b_1} & \frac{b_3 - b_1}{(b_3 - b_2)(b_2 - b_1)} & -\frac{1}{b_3 - b_2} & \cdots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & -\frac{1}{b_{r-1} - b_{r-2}} & \frac{b_r - b_{r-2}}{(b_r - b_{r-1})(b_{r-1} - b_{r-2})} & -\frac{1}{b_r - b_{r-1}} \\
0 & 0 & 0 & \cdots & 0 & -\frac{1}{b_r - b_{r-1}} & \frac{b_{r+1} - b_{r-1}}{(b_{r+1} - b_r)(b_r - b_{r-1})}
\end{bmatrix}.$$

Notice that $\mathbf{B}_2^{-1}$ has the same structure as $\mathbf{B}_1^{-1}$ except for the last element as well as the scaling coefficient $1/b_{r+1}$. Since $a_1, \cdots, a_r$ need to be non-negative, we are left to prove the following inequalities:

$$w_e(\mathcal{S}_1) \geq \frac{b_1}{b_2} w_e(\mathcal{S}_2), \tag{24}$$

$$w_e(\mathcal{S}_i) \geq \frac{b_{i+1}-b_i}{b_{i+1}-b_{i-1}} w_e(\mathcal{S}_{i-1}) + \frac{b_i-b_{i-1}}{b_{i+1}-b_{i-1}} w_e(\mathcal{S}_{i+1}), \quad \forall i = 2, \cdots, r-1, \tag{25}$$

$$w_e(\mathcal{S}_r) \geq \frac{b_{r+1}-b_r}{b_{r+1}-b_{r-1}} w_e(\mathcal{S}_{r-1}). \tag{26}$$

By introducing $b_0 = 0$, the above inequalities can be rewritten and summarized as

$$g_e(b_i) \geq \frac{b_{i+1}-b_i}{b_{i+1}-b_{i-1}} g_e(b_{i-1}) + \frac{b_i-b_{i-1}}{b_{i+1}-b_{i-1}} g_e(b_{i+1}), \quad \forall i = 1, \cdots, r, \tag{27}$$

which immediately follows (ii) in Lemma 3.1. $\qquad\square$

For the two cases discussed in Theorem 3.4, the required number of building gadgets $|\mathcal{Q}_s|$ and $|\mathcal{Q}_a|$ are respectively upper bounded by $2^{|e|-1} - 1$ and $2^{|e|} - 2$. For trivial EDVWs, the theorem coincides with the results for cardinality-based splitting functions in (Veldt et al. 2020a) with $|\mathcal{Q}_s|$ and $|\mathcal{Q}_a|$ respectively reducing to $\lfloor|e|/2\rfloor$ and $|e| - 1$. Hence, EDVWs-based splitting functions generally lead to a denser graph than cardinality-based splitting functions in the worst case. In addition, when g_e is concave as well as symmetric, Theorem 3.4 provides two ways for the reduction while the one leveraging symmetric EDVWs-based gadgets requires fewer edges.

Sparsifying hypergraph-to-graph reductions
Graph min-cut/max-flow algorithms have a complexity that depends on the number of vertices and the number of edges in the graph (Goldberg 1998). The reduction procedures discussed above may result in large and dense graphs, thus affecting the efficiency of algorithms applied to the reduced graph. A workaround is to find a smaller and sparser graph whose cut approximates, rather than exactly recovers, the hypergraph cut (Bansal et al. 2019; Benczúr and Karger 1996; Benson et al. 2020; Chekuri and Xu 2018; Kogan and Krauthgamer 2015). In Benson et al. (2020), a sparsification technique is proposed for hypergraphs with submodular cardinality-based splitting functions. It is based on approximating concave functions using piecewise linear curves and can be generalized to our EDVWs-based case due to the formulation (7). In the following, we discuss this in detail.

As shown in (3), the hypergraph cut function is defined as the sum of splitting functions associated with every hyperedge. Hence, the problem can be decomposed into approximately modeling each hyperedge using a sparse gadget with a smaller set of auxiliary vertices. More formally, we would like to find such a gadget whose corresponding splitting function $\hat{w}_e$ as defined in (6) approximates the splitting penalties associated with a hyperedge, i.e.,

$$w_e(\mathcal{S}) \leq \hat{w}_e(\mathcal{S}) \leq (1 + \epsilon) w_e(\mathcal{S}), \quad \forall \mathcal{S} \subseteq e, \tag{28}$$

where $\epsilon \geq 0$ is an approximation tolerance parameter. This is equivalent to finding some gadget splitting function $\tilde{w}_e$ satisfying $\frac{1}{\delta} w_e(\mathcal{S}) \leq \tilde{w}_e(\mathcal{S}) \leq \delta w_e(\mathcal{S})$ with the correspondence $\hat{w}_e(\mathcal{S}) = \delta \tilde{w}_e(\mathcal{S})$ and $\epsilon = \delta^2 - 1$.

We first consider EDVWs-based splitting functions with concave and symmetric g_e. We have shown in Theorem 3.4 that these splitting functions can be exactly constructed from a combination of $|\mathcal{Q}_s|$ symmetric EDVWs-based gadgets. Following the idea

in Benson et al. (2020), we next show how to approximate these splitting functions using a smaller set of symmetric EDVWs-based gadgets. Recall from the proof of Theorem 3.4 that the combination of r symmetric EDVWs-based gadgets respectively with positive parameters $b_1 < \cdots < b_r$ and combination coefficients $a_1, \cdots, a_r$ has the splitting function in the form of (14). Its continuous extension can be written as follows. Since it is symmetric, we can just consider the first half of the function

$$\hat{g}_e(x) = \sum_{i=1}^r a_i \cdot \min\{x, b_i\}, \quad \text{where } x \in [0, \gamma_e(e)/2]. \tag{29}$$

When $x \leq b_1$, (29) can be rewritten as $\hat{g}_e(x) = (\sum_{i=1}^r a_i) \cdot x$; when $x > b_r$, we have $\hat{g}_e(x) = \sum_{i=1}^r a_i b_i$; when $b_{i-1} < x \leq b_i$ for any $i = 2, \cdots, r$, we have $\hat{g}_e(x) = \sum_{j=1}^{i-1} a_j b_j + (\sum_{j=i}^r a_j) \cdot x$. Hence, $\hat{g}_e(x)$ can also be characterized as the lower envelope of a set of $r + 1$ linear functions having non-negative decreasing slopes and non-negative increasing intercepts (Benson et al. 2020), i.e.,

$$\begin{aligned}
&\hat{g}_e(x) = \min\{f_1(x), f_2(x), \cdots, f_{r+1}(x)\}, \\
&\text{where} \quad f_i(x) = m_i x + d_i, \\
&m_i = \sum_{j=i}^r a_j \text{ for } 1 \leq i \leq r \text{ and } m_{r+1} = 0, \\
&d_1 = 0 \text{ and } d_i = \sum_{j=1}^{i-1} a_j b_j \text{ for } 2 \leq i \leq r+1.
\end{aligned} \tag{30}$$

Equivalently, the relations between the coefficients a_i, b_i and m_i, d_i can also be described as $a_i = m_i - m_{i+1}$ and $b_i = (d_{i+1} - d_i)/a_i$.

The sparsification problem can be described as: Find the piecewise linear function $\hat{g}_e$ with the minimum number of pieces that approximates g_e at the points in $\mathcal{Q}_s$. It can be formulated as follows.

$$\begin{aligned}
&\min \ r \\
&\text{s.t.} \quad g_e(x) \leq \hat{g}_e(x) \leq (1+\epsilon)g_e(x), \forall x \in \mathcal{Q}_s, \\
&\qquad \hat{g}_e(x) \text{ is defined as (30)}, \\
&\qquad \text{For each } i \in [r+1], f_i(x) = g_e(x) \text{ for some } x \in \{0\} \cup \mathcal{Q}_s.
\end{aligned} \tag{31}$$

The first constraint is from (28). The third constraint is added without loss of generality since an improved approximation could be found if there is some f_i strictly greater than g_e at all points in $\{0\} \cup \mathcal{Q}_s$. The main difference with the corresponding formulation in (Benson et al. 2020) is that, for the cardinality-based case there considered, the set $\mathcal{Q}_s$ consists of only integers from 1 to $\lfloor |e|/2 \rfloor$; for the EDVWs-based case, the elements in $\mathcal{Q}_s$ depend on the values of EDVWs thus they are not necessarily integers or evenly spaced.

We can modify the algorithm proposed in Benson et al. (2020) to find an optimal solution to (31). Here we briefly introduce the procedure and please refer to Benson et al. (2020) for a detailed explanation. For convenience, we denote the elements in $\mathcal{Q}_s$ by $q_1 < q_2 < \cdots < q_n$ where $n = |\mathcal{Q}_s|$ and define a series of functions with the ith one joining $(q_i, g_e(q_i))$ and $(q_{i+1}, g_e(q_{i+1}))$, i.e., $h_i(x) = \frac{g_e(q_{i+1}) - g_e(q_i)}{q_{i+1} - q_i}(x - q_i) + g_e(q_i)$. The last linear piece f_{r+1} is first determined. According to (30) and (31), it has a zero slope and passes through $(q_n, g_e(q_n))$, thus we have $f_{r+1}(x) = g_e(q_n)$. For the first linear piece f_1, it goes through the origin according to (30) and its slope m_1 is set to $\frac{g_e(q_1)}{q_1}$ so that it can provide a qualified approximation for as many points in $\mathcal{Q}_s$ as possible. Then we identify the first point q_ℓ in $\mathcal{Q}_s$ that does not have a $(1+\epsilon)$-approximation. In other words,

$f_1(q_i) \leq (1 + \epsilon)g_e(q_i)$ holds for $i = 1, \cdots, \ell - 1$ and becomes invalid since $i = \ell$. We stop searching more linear pieces if (i) $\ell \geq n$, or (ii) $g_e(q_n) \leq (1 + \epsilon)g_e(q_\ell)$ where (ii) implies that $q_\ell, \cdots, q_n$ have a $(1 + \epsilon)$-approximation provided by the last linear piece. Otherwise, we continue to find the next linear piece f_2 in order to cover the rest of the points. We identify i^* such that $h_{i^*-1}(q_\ell) \leq (1 + \epsilon)g_e(q_\ell)$ and $h_{i^*}(q_\ell) > (1 + \epsilon)g_e(q_\ell)$. In other words, the linear function h_{i^*-1} provides a qualified approximation at q_ℓ (i.e., the first point has not been covered yet), but the following functions h_i for $i \geq i^*$ do not. Hence, f_2 should pass through the point $(q_{i^*}, g_e(q_{i^*}))$ and lie between h_{i^*-1} and h_{i^*} (its slope should be greater than h_{i^*}'s slope and no greater than h_{i^*-1}'s slope). In the meantime, we expect f_2 to provide a qualified approximation at q_ℓ and have a slope as small as possible so that more points after q_{i^*} can be covered. Therefore, we set f_2 to the line joining $(q_\ell, (1 + \epsilon)g_e(q_\ell))$ and $(q_{i^*}, g_e(q_{i^*}))$. We refer the reader to Figure 4 in (Benson et al. 2020) for an illustration. Then we update q_ℓ to the new start point in $\mathcal{Q}_s$ that does not have a $(1 + \epsilon)$-approximation yet, and check the stopping criterion. We repeat the process of picking f_2 to add more linear pieces until we meet the stopping criterion.

We can also ignore the particular positions of elements in $\mathcal{Q}_s$ and try to provide a $(1 + \epsilon)$-approximation for $g_e(x)$ everywhere in the range $[0, \gamma_e(e)/2]$. This approach has two benefits: (i) It avoids building the set $\mathcal{Q}_s$. (ii) If multiple hyperedges share the same continuous extension g_e, we can find their sparsified reductions all at once. The procedure of finding the set of linear pieces can be modified as follows. The last linear piece should be $f_{r+1}(x) = g_e(\gamma_e(e)/2)$. The first linear piece f_1 is tangent to g_e at the origin. We identify the value $z \in [0, \gamma_e(e)/2]$ that satisfies $f_1(z) = (1 + \epsilon)g_e(z)$. If no such a z exits or $g_e(\gamma_e(e)/2) \leq (1 + \epsilon)g_e(z)$, we stop. Otherwise, we add the next linear piece f_2 which is selected to pass through $(z, (1 + \epsilon)g_e(z))$ and be tangent to g_e as some point greater than z. We repeat the process of choosing f_2 to add more linear pieces until the whole range $[0, \gamma_e(e)/2]$ has been covered. An illustrative example is given in Fig. 2.

For EDVWs-based splitting functions with concave and possibly asymmetric g_e, they can be approximated using a smaller set of asymmetric EDVWs-based gadgets. It has been proved in (Benson et al. 2020) that the continuous extension of the splitting function in the form of (22) is also piecewise linear, thus we can adopt a similar reduction procedure as described above.

Moreover, a direct extension of Theorem 4.1 in (Benson et al. 2020) is that the number of symmetric/asymmetric EDVWs-based gadgets needed to approximate an EDVWs-based splitting function with concave, symmetric/asymmetric g_e is upper bounded by $O(\log_{1+\epsilon} \gamma_e(e))$ which behaves as $\epsilon^{-1} \log \gamma_e(e)$ as ϵ approaches zero. This bound could be further improved if a specific concave function g_e is chosen.

Experiments

We show the effects of introducing EDVWs into hypergraph cuts via numerical experiments. We consider the binary classification of hypergraph vertices: Given the labels of a subset of vertices $\mathcal{V}^L = \mathcal{V}_1^L \cup \mathcal{V}_2^L$ where $\mathcal{V}_1^L$ and $\mathcal{V}_2^L$ respectively consist of all labeled vertices in two classes, the task is to estimate the labels of the rest of the vertices. The problem can be formulated as a generalized hypergraph minimum s-t cut problem with multiple source and sink vertices, i.e.,
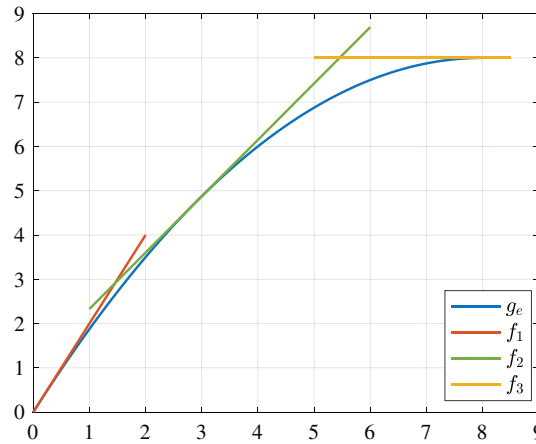
Fig. 2 An example where $g_e(x) = -0.125x^2 + 2x$. We want to find a $(1 + \epsilon)$-approximation for $g_e(x)$ everywhere in the range [0, 8] where we set $\epsilon = 0.1$. The last linear piece f_3 has a zero slope and passes through (8, 8), thus $f_3(x) = 8$. The first linear piece f_1 is tangent to g_e at the origin, hence $f_1(x) = 2x$. At the point $x = 1.4545$, $f_1(x) = (1 + \epsilon)g_e(x)$, meaning that f_1 provides a qualified approximation for $g_e(x)$ in the range [0, 1.4545]. The second linear piece f_2 passes through (1.4545, 2.909) and is tangent to g_e at (2.909, 4.7602), namely $f_2(x) = 1.2727x + 1.0579$. At $x = 5.2894$, $f_2(x) = (1 + \epsilon)g_e(x)$, hence f_2 provides a qualified approximation for $g_e(x)$ in the range [1.4545, 5.2894]. The rest of the points in the range [5.2894, 8] have been covered by f_3

$$\min_{\emptyset \subset \mathcal{S} \subset \mathcal{V}} \mathrm{cut}_{\mathcal{H}}(\mathcal{S}) \quad \text{s.t.} \ \mathcal{V}_1^L \subseteq \mathcal{S}, \mathcal{V}_2^L \subseteq \mathcal{V} \setminus \mathcal{S}. \tag{32}$$

Then the vertices in $\mathcal{S}$ and $\mathcal{V} \setminus \mathcal{S}$ are classified into two categories, respectively. We consider the first three EDVWs-based splitting functions listed in Table 1, namely $w_e(\mathcal{S}) = \gamma_e(\mathcal{S}) \cdot \gamma_e(e \setminus \mathcal{S})$, $w_e(\mathcal{S}) = \min\{\gamma_e(\mathcal{S}), \gamma_e(e \setminus \mathcal{S})\}$ and $w_e(\mathcal{S}) = \min\{\gamma_e(\mathcal{S}), \gamma_e(e \setminus \mathcal{S}), b\}$, which are symmetric and graph reducible. For any of them, we can reduce the hypergraph to a graph sharing the same cut properties. Following the idea in (Blum and Chawla 2001), we introduce another two vertices – a super-source s and a super-sink t – into the reduced graph. For every $v \in \mathcal{V}_1^L$, add an edge of weight infinity from s to v; for every $v \in \mathcal{V}_2^L$, add an edge of weight infinity from v to t. Then problem (32) can be converted into a common minimum s-t cut problem defined on the new graph in the form of (2). We solve the graph minimum s-t cut problem using the highest-label preflow-push algorithm implemented in the NetworkX package (Hagberg et al. 2008).

We adopt the 20 Newsgroups dataset and consider the task of document classification. The dataset contains documents in different categories and we consider the documents in categories "rec.motorcycles" and "sci.space". We extract the 200 most frequent words in the corpus after removing stop words and words appearing in $> 3\%$ and $< 0.2\%$ of the documents. We then remove a small fraction of documents that do not contain the selected words and finally get 1852 documents with 932 and 920 documents in two classes, respectively. To model the text dataset using hypergraphs with EDVWs, we consider documents as vertices and words as hyperedges. A document (vertex) belongs to a word (hyperedge) if the word appears in the document. The EDVWs are taken as the corresponding tf-idf (term frequency-inverse document frequency) values (Leskovec et al. 2020) to the power of α, where α is a tunable parameter. More precisely, we set

$$\gamma_e(v) = \text{tf-idf}(e, v)^\alpha, \quad \text{where tf-idf}(e, v) = \text{tf}(e, v) \cdot \text{idf}(e). \tag{33}$$

The term frequency $\text{tf}(e, v)$ is the relative frequency of word e in document v. The inverse document frequency $\text{idf}(e)$ measures the informativeness of word e, i.e., if it is common or rare across all documents. Hence, the tf-idf values are able to reflect the importance of a word to a document in a corpus and thus an ideal choice for EDVWs. We adopt the TfidfTransformer function in the scikit-learn package with default parameters to compute the tf-idf values. The parameter α is introduced for extra flexibility. When $\alpha = 0$, we get the trivial EDVWs and the splitting functions reduce to cardinality-based ones. For the splitting function $w_e(\mathcal{S}) = \min\{\gamma_e(\mathcal{S}), \gamma_e(e \setminus \mathcal{S}), b\}$, we set $b = \beta\gamma_e(e)$ where β is also adjustable. If a small enough β is selected, the splitting function reduces to the all-or-nothing case.

Figure 3 shows the effects of the parameters α and β on the classification performance. We plot the average classification accuracy and the standard deviation over 10 realizations which adopt different sets of labeled vertices. We respectively set the fraction of labeled vertices to 0.3 in (a-d) and 0.5 in (e-h). The three considered EDVWs-based splitting functions respectively correspond to (a) (e), (b) (f), and (c-d) (g-h). For the third splitting function, we fix $\beta = 0.15$ to observe the influence of α in (c) (g) and fix $\alpha = 1$ to test β in (d) (h). It can be observed that, for all of them, the best performance is achieved for intermediate values of α or β rather than the extreme cases when the EDVWs-based splitting function reduces to a cardinality-based splitting function or the all-or-nothing splitting function.

Figure 4 provides a more direct comparison between the proposed EDVWs-based splitting functions and existing ones. For EDVWs-based splitting functions, we adopt a 5-fold cross-validation to find the optimal α and β. In (a), we search the optimal α over the set $\{0 : 0.2 : 3\}$. In (b-c), we search α over $\{0 : 0.2 : 5\}$. In (d), we search β over
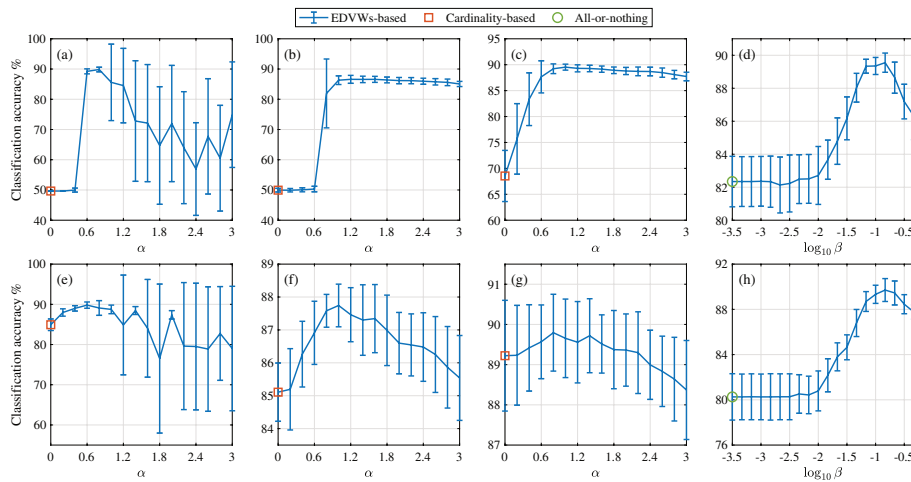

Fig. 3 Classification performance as a function of the parameters α and β. For the two rows, the fraction of labeled vertices is respectively set to 0.3 and 0.5. (**a**) and (**e**) correspond to the splitting function $w_e(\mathcal{S}) = \gamma_e(\mathcal{S}) \cdot \gamma_e(e \setminus \mathcal{S})$; (**b**) and (**f**) correspond to the splitting function $w_e(\mathcal{S}) = \min\{\gamma_e(\mathcal{S}), \gamma_e(e \setminus \mathcal{S})\}$; (**c**), (**d**) and (**g**), (**h**) correspond to the splitting function $w_e(\mathcal{S}) = \min\{\gamma_e(\mathcal{S}), \gamma_e(e \setminus \mathcal{S}), \beta\gamma_e(e)\}$ where we fix $\beta = 0.15$ in (**c**), (**g**) and we fix $\alpha = 1$ in (**d**), (**h**)
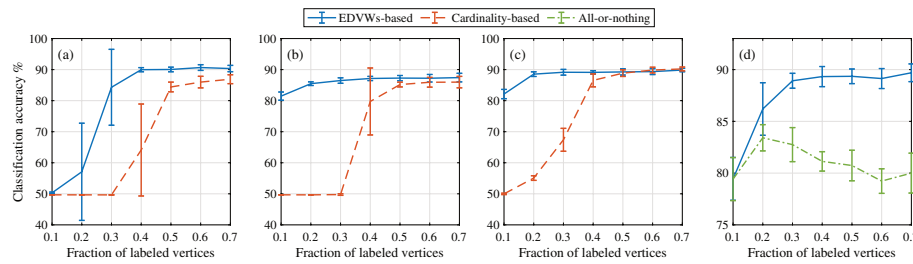
Fig. 4 Performance comparison between the proposed splitting functions and existing ones. (**a**), (**b**) and (**c-d**) respectively correspond to the splitting functions $w_e(\mathcal{S}) = \gamma_e(\mathcal{S}) \cdot \gamma_e(e \setminus \mathcal{S})$, $w_e(\mathcal{S}) = \min\{\gamma_e(\mathcal{S}), \gamma_e(e \setminus \mathcal{S})\}$ and $w_e(\mathcal{S}) = \min\{\gamma_e(\mathcal{S}), \gamma_e(e \setminus \mathcal{S}), \beta\gamma_e(e)\}$. We fix $\beta = 0.15$ in (**c**) and fix $\alpha = 1$ in (**d**). The red curves (cardinality-based) and the green curve (all-or-nothing) respectively correspond to the cases when $\alpha = 0$ and when β is small enough ($\beta = 10^{-3.5}$ here). For the blue curves (EDVWs-based), a 5-fold cross-validation is adopted in (**a–c**) to search the optimal α and in (**d**) to search the optimal β

20 equally spaced values between $10^{-3.5}$ and $10^{-1/3}$ in the log scale. We can see that adopting EDVWs-based splitting functions improves the classification performance over a wide range of train-test split ratios.

Conclusion

We developed a framework for incorporating EDVWs into hypergraph cut problems and generalized reduction as well as sparsification techniques recently proposed for cardinality-based splitting functions. Through a real-world text mining application, we showcased the value of the introduction of EDVWs. There are numerous directions for future work: (i) As mentioned in the introduction, hypergraph minimum cuts can be used to solve various real-world applications (Catalyurek and Aykanat 1999; Ding and Yilmaz 2008; Karypis et al. 1999; Kim et al. 2011) or as subroutines in many machine learning algorithms (Liu et al. 2021; Veldt et al. 2020b). Hence, it would be desirable to apply the proposed framework to these applications and algorithms and evaluate the performance. (ii) Another direction is to further extend the proposed framework to multiway cuts (Chekuri and Ene 2011; Okumoto et al. 2012; Veldt et al. 2020a; Zhao et al. 2005) or other types of cuts such as normalized cuts (Li and Milenkovic 2017, 2018b; Fountoulakis et al. 2021). (iii) An open problem is whether all submodular splitting functions are graph reducible (Veldt et al. 2020a).

Declarations

Ethics approval and consent to participate
Not applicable

Consent for publication
Not applicable

Competing interests
The authors declare that they have no competing interests.

References

Agarwal S, Branson K, Belongie S (2006) Higher order learning with graphs. In: International conference on machine learning. https://doi.org/10.1145/1143844.1143847

Bach F (2013) Learning with submodular functions: a convex optimization perspective. Foundations and Trends® in machine learning 6(2-3):145–373

Bansal N, Svensson O, Trevisan L (2019) New notions and constructions of sparsification for graphs and hypergraphs. In: IEEE symposium on foundations of computer science, pp 910–928, https://doi.org/10.1109/focs.2019.00059

Benczúr AA, Karger DR (1996) Approximating s-t minimum cuts in $\tilde{O}(n^2)$ time. In: Symposium on theory of computing, pp 47–55. https://doi.org/10.1145/237814.237827

Benson AR, Kleinberg J, Veldt N (2020) Augmented sparsifiers for generalized hypergraph cuts with applications to decomposable submodular function minimization. arXiv preprint arXiv:2007.08075 https://arxiv.org/abs/2007.08075

Blum A, Chawla S (2001) Learning from labeled and unlabeled data using graph mincuts. In: International conference on machine learning, pp 19–26. https://doi.org/10.1184/R1/6606860.V1

Bokhari SH (1987) Assignment problems in parallel and distributed computing, vol 32. Springer, Boston. https://doi.org/10.1007/978-1-4613-2003-6

Boykov Y, Funka-Lea G (2006) Graph cuts and efficient N-D image segmentation. Int J Comput Vis 70(2):109–131. https://doi.org/10.1007/s11263-006-7934-5

Boykov Y, Kolmogorov V (2004) An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans Pattern Anal Mach Intell 26(9):1124–1137. https://doi.org/10.1109/TPAMI.2004.60

Catalyurek UV, Aykanat C (1999) Hypergraph-partitioning-based decomposition for parallel sparse-matrix vector multiplication. IEEE Trans Parallel Distrib Syst 10(7):673–693. https://doi.org/10.1109/71.780863

Chekuri C, Ene A (2011) Approximation algorithms for submodular multiway partition. In: IEEE symposium on foundations of computer science, pp 807–816. https://doi.org/10.1109/FOCS.2011.34

Chekuri C, Xu C (2018) Minimum cuts and sparsification in hypergraphs. SIAM J Comput 47(6):2118–2156. https://doi.org/10.1137/18M1163865

Chen L, Kyng R, Liu YP, Peng R, Gutenberg MP, Sachdeva S (2022) Maximum flow and minimum-cost flow in almost-linear time. arXiv preprint arXiv:2203.00671 https://arxiv.org/abs/2203.00671

Cherkassky BV, Goldberg AV, Martin P, Setubal JC, Stolfi J (1998) Augment or push: a computational study of bipartite matching and unit-capacity flow algorithms. ACM J Exp Algorithm 3:8. https://doi.org/10.1145/297096.297140

Chitra U, Raphael B (2019) Random walks on hypergraphs with edge-dependent vertex weights. In: International conference on machine learning, pp 1172–1181. http://proceedings.mlr.press/v97/chitra19a.html

Colbourn CJ (1991) Combinatorial aspects of network reliability. Ann Oper Res 33(1):1–15. https://doi.org/10.1007/BF02061656

Ding L, Yilmaz A (2008) Image segmentation as learning on hypergraphs. In: IEEE international conference on machine learning and applications, pp 247–252. https://doi.org/10.1109/ICMLA.2008.17

Ding L, Yilmaz A (2010) Interactive image segmentation using probabilistic hypergraphs. Pattern Recognit 43(5):1863–1873. https://doi.org/10.1016/j.patcog.2009.11.025

Ene A, Nguyên HL, Végh LA (2017) Decomposable submodular function minimization discrete and continuous. In: Advances in neural information processing systems, pp 2874–2884. https://dl.acm.org/doi/10.5555/3294996.3295047

Ford LR, Fulkerson DR (1956) Maximal flow through a network. Can J Math 8:399–404. https://doi.org/10.4153/CJM-1956-045-5

Fountoulakis K, Li P, Yang S (2021) Local hyper-flow diffusion. In: Advances in neural information processing systems, vol 34, arXiv:2102.07945

Goldberg AV (1998) Recent developments in maximum flow algorithms. In: Scandinavian workshop on algorithm theory, pp 1–10, https://doi.org/10.1007/BFb0054350

Goldberg AV, Tarjan RE (1988) A new approach to the maximum-flow problem. J ACM 35(4):921–940. https://doi.org/10.1145/48014.61051

Grötschel M, Lovász L, Schrijver A (1981) The ellipsoid method and its consequences in combinatorial optimization. Combinatorica 1(2):169–197. https://doi.org/10.1007/BF02579273

Hagberg A, Swart P, S Chult D (2008) Exploring network structure, dynamics, and function using networkx. Tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States), https://www.osti.gov/biblio/960616

Hayashi K, Aksoy SG, Park CH, Park H (2020) Hypergraph random walks, laplacians, and clustering. In: Conference on information and knowledge management, pp 495–504. https://doi.org/10.1145/3340531.3412034

Hein M, Setzer S, Jost L, Rangapuram SS (2013) The total variation on hypergraphs-learning on hypergraphs revisited. In: Advances in neural information processing systems, vol 26, pp 2427–2435. https://doi.org/10.5555/2999792.2999883

Ihler E, Wagner D, Wagner F (1993) Modeling hypergraphs by graphs with the same mincut properties. Inf Process Lett 45(4):171–175. https://doi.org/10.1016/0020-0190(93)90115-P

Iwata S (2003) A faster scaling algorithm for minimizing submodular functions. SIAM J Comput 32(4):833–840. https://doi.org/10.1137/S0097539701397813

Iwata S, Orlin JB (2009) A simple combinatorial algorithm for submodular function minimization. In: ACM SIAM symposium on discrete algorithms, pp 1230–1237, https://doi.org/10.1137/1.9781611973068.133

Iwata S, Fleischer L, Fujishige S (2001) A combinatorial strongly polynomial algorithm for minimizing submodular functions. J ACM 48(4):761–777. https://doi.org/10.1145/502090.502096

Karypis G, Aggarwal R, Kumar V, Shekhar S (1999) Multilevel hypergraph partitioning: Applications in VLSI domain. IEEE Trans Very Large Scale Integr (VLSI) Syst 7(1):69–79, https://doi.org/10.1109/92.748202

Kim S, Nowozin S, Kohli P, Yoo C (2011) Higher-order correlation clustering for image segmentation. In: Advances in neural information processing systems, vol 24, pp 1530–1538, https://doi.org/10.5555/2986459.2986630

Kogan D, Krauthgamer R (2015) Sketching cuts in graphs and hypergraphs. In: Innovations in theoretical computer science, pp 367–376. https://doi.org/10.1145/2688073.2688093

Kolmogorov V (2012) Minimizing a sum of submodular functions. Discrete Appl Math 160(15):2246–2258. https://doi.org/10.1016/j.dam.2012.05.025

Lawler EL (1973) Cutsets and partitions of hypergraphs. Networks 3(3):275–285. https://doi.org/10.1002/net.3230030306

Leighton T, Rao S (1999) Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. J ACM 46(6):787–832. https://doi.org/10.1145/331524.331526

Leskovec J, Rajaraman A, Ullman JD (2020) Mining of massive data sets. Cambridge university press, New York, https://doi.org/10.1017/CBO9781139924801

Li P, Milenkovic O (2017) Inhomogeneous hypergraph clustering with applications. In: Advances in neural information processing systems, vol 30, pp 2308–2318. https://doi.org/10.5555/3294771.3294991

Li P, Milenkovic O (2018a) Revisiting decomposable submodular function minimization with incidence relations. In: Advances in neural information processing systems, pp 2242–2252. https://doi.org/10.5555/3327144.3327151

Li P, Milenkovic O (2018b) Submodular hypergraphs: p-laplacians, cheeger inequalities and spectral clustering. In: International conference on machine learning, pp 3014–3023. http://proceedings.mlr.press/v80/li18e.html

Liu M, Veldt N, Song H, Li P, Gleich DF (2021) Strongly local hypergraph diffusions for clustering and semi-supervised learning. In: International world wide web conference, pp 2092–2103. https://doi.org/10.1145/3442381.3449887

Li J, Lillis J, Cheng CK (1995) Linear decomposition algorithm for VLSI design applications. In: IEEE international conference on computer aided design, pp 223–228, https://doi.org/10.1109/ICCAD.1995.480016

Li J, He J, Zhu Y (2018) E-tail product return prediction via hypergraph-based local graph cut. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp 519–527, https://doi.org/10.1145/3219819.3219829

Lovász L, Plummer MD (2009) Matching theory, vol 367. American Mathematical Soc., Providence. https://www.ams.org/books/chel/367/chel367-endmatter.pdf

Okumoto K, Fukunaga T, Nagamochi H (2012) Divide-and-conquer algorithms for partitioning hypergraphs and submodular systems. Algorithmica 62(3):787–806. https://doi.org/10.1007/978-3-642-10631-6_8

Orlin JB (2013) Max flows in o (nm) time, or better. In: Proceedings of the forty-fifth annual ACM symposium on theory of computing, pp 765–774. https://doi.org/10.1145/2488608.2488705

Orlin JB (2009) A faster strongly polynomial time algorithm for submodular function minimization. Math Programm 118(2):237–251. https://doi.org/10.1007/s10107-007-0189-2

Papa DA, Markov IL (2007) Hypergraph partitioning and clustering. Handbook of Approximation Algorithms and Metaheuristics 20073547:61–1. https://web.eecs.umich.edu/~imarkov/pubs/book/part_survey.pdf

Ramanathan A, Colbourn CJ (1987) Counting almost minimum cutsets with reliability applications. Math Program 39(3):253–261. https://doi.org/10.1007/BF02592076

Schaub MT, Zhu Y, Seby JB, Roddenberry TM, Segarra S (2021) Signal processing on higher-order networks: Livin'on the edge... and beyond. Signal Process 187:108–149. https://doi.org/10.1016/j.sigpro.2021.108149

Schrijver A (2000) A combinatorial algorithm minimizing submodular functions in strongly polynomial time. J Combin Theory Ser B 80(2):346–355. https://doi.org/10.1006/jctb.2000.1989

Veldt N, Benson AR, Kleinberg J (2020a) Hypergraph cuts with general splitting functions. arXiv preprint arXiv:2001.02817https://arxiv.org/abs/2001.02817

Veldt N, Benson AR, Kleinberg J (2020b) Minimizing localized ratio cut objectives in hypergraphs. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp 1708–1718, https://doi.org/10.1145/3394486.3403222

Zhao L, Nagamochi H, Ibaraki T (2005) Greedy splitting algorithms for approximating multiway partition problems. Math Program 102(1):167–183. https://doi.org/10.1007/s10107-004-0510-2

Zhou D, Huang J, Schölkopf B (2006) Learning with hypergraphs: Clustering, classification, and embedding. In: Advances in neural information processing systems, vol 19, pp 1601–1608. https://doi.org/10.5555/2976456.2976657

Zhu Y, Li B, Segarra S (2021) Co-clustering vertices and hyperedges via spectral hypergraph partitioning. In: European signal processing conference, pp 1416–1420, https://doi.org/10.23919/EUSIPCO54536.2021.9616223

Publisher's Note