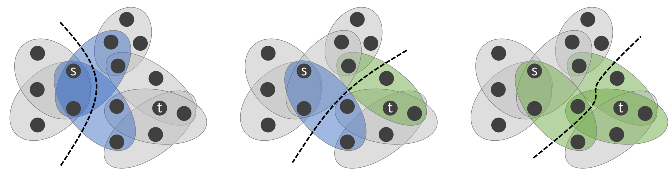
HYPERGRAPH S-T CUTS WITH GENERAL SPLITTING FUNCTIONS

Nate Veldt, Austin R. Benson, and Jon Kleinberg

SIAM Workshop on Network Science 2020
July 9–10 · Toronto

Summary

Finding a minimum s-t cut in a graph is one of the most well-studied problems in combinatorial optimization. While graphs are a useful abstraction for modeling pairwise relationships between objects in a complex system, hypergraphs provide a more faithful way to model systems characterized by multiway relationships. However, one challenge in generalizing graph cut problems to the hypergraph setting is that there are numerous ways to separate or *cut* the nodes of a hyperedge, each of which may be better or worse for downstream applications. Here we present a generalized framework for hypergraph s-t cut problems based on *splitting functions*, which map each node configuration of a hyperedge to a different splitting penalty. We consider special classes of splitting functions for which the problem can be solved in polynomial time via reduction to a graph s-t cut problem, as well as other cases where the problem is NP-hard. As an application, we incorporate our techniques into a new method for localized hypergraph clustering.



(a) $w_2 = 0.5$ solution (b) $w_2 = 1.5$ solution (c) $w_2 = 2.5$ solution

Figure 1: For 4-node hyperedges, cardinality-based functions assign a penalty of w_1 for 1-3 node splits (green), and w_2 for 2-2 node splits (blue). Solutions are shown for $w_1 = 1$. Our results show that for 4-uniform hypergraphs, the s-t cut problem is NP-hard when $w_2 < w_1$ (a), but reducible to a graph s-t cut problem when $w_2 \in [w_1, 2w_1]$ (b). When $w_2 > 2w_1$ (c), the complexity is unknown, but we show the problem is not graph reducible.

Introduction

The minimum s-t cut problem seeks a minimum weight set of edges to cut or remove from a graph in order to separate two designated terminal nodes s and t. The first hypergraph generalization of the problem was introduced by Lawler [2], motivated by applications to information storage, numerical taxonomy, and electronic circuit packaging. Lawler specifically considered how to separate s and t in a way that minimizes an *all-or-nothing* hypergraph cut function. More precisely, this function assigns no penalty to an uncut hyperedge, while a hyperedge that spans both clusters incurs a penalty equal to the hyperedge's weight, regardless of how its nodes are separated. Under this cut function, Lawler showed that the problem can be reduced to a minimum s-t cut problem in a directed graph.

A number of other hypergraph clustering objectives also rely on minimizing all-or-nothing penalties, subject to other possible constraints such as cluster size or placement of terminal nodes. However, in practice there are many reasons to cluster the nodes of a hyperedge together, and different node configurations may be better or worse depending on the application. For example, a hyperedge may represent a sets of objects in a dataset that are associated with a similar attribute or property. This may provide evidence that these objects should be clustered together. If all but a small subset of them are placed in the same cluster, this *mostly* agrees with the evidence, but the all-or-nothing function penalizes this just as severely as splitting the hyperedge perfectly in half.

While there is some limited work on generalized notions of hypergraph cuts [1, 3], the literature on this topic is somewhat fragmented. Furthermore, all previous results for the hypergraph s-t cut problem assume an all-or-nothing penalty [2]. Here we develop a rigorous framework for hypergraph s-t cut problems under much broader notions of hypergraph cuts. Our framework relies on the concept of a *splitting function*, which maps each subset of nodes in a hyperedge to a different penalty. We introduce a special class of *cardinality-based* functions, which assign penalties based only on the number of nodes on each side of split hyperedge. We prove that for cardinality-based penalties, the hypergraph s-t cut problem is reducible to a graph s-t cut problem if and only if the functions are submodular. We also demonstrate hardness results for certain cases outside the submodular regime. Figure 1 illustrates key differences between three hypergraph s-t cut problems on the same small hypergraph.

The Generalized Hypergraph s-t Cut Problem

Let $\mathcal{H} = (V, E)$ be a hypergraph, and for each hyperedge $e \in E$, let 2^e be the power set of e. A *splitting function* $\mathbf{w}_e : 2^e \to \mathbb{R}^+$ maps subsets of e to non-negative penalties. To generalize graph cut penalties, splitting functions must be symmetric and penalize only *cut* hyperedges:

$$\begin{aligned} \mathbf{w}_e(A) &= \mathbf{w}(e \backslash V) && \text{for all } A \subseteq e \\ \mathbf{w}_e(A) &= 0 && \text{if } A = e \text{ or } A = \emptyset. \end{aligned}$$

A splitting function $\mathbf{w}_e$ is *cardinality-based* if $\mathbf{w}_e(A) = \mathbf{w}_e(B)$ whenever $|A| = |B|$. This class of functions is particularly relevant for applications, since in most situations the quality of a clustering may depend on the number of nodes on different sides of a cut, but the individual identity of specific nodes is not important. A splitting function is submodular if for every $A, B \in 2^e$ it satisfies

$$\mathbf{w}_e(A) + \mathbf{w}_e(B) \geq \mathbf{w}_e(A \cap B) + \mathbf{w}_e(A \cup B)\,. \quad (1)$$

The generalized hypergraph s-t cut problem is given by

$$\min_{S \subseteq V} \sum_{e \in E} \mathbf{w}_e(S \cap e) \quad \text{s.t.} \quad s \in S, t \in V \backslash S\,. \quad (2)$$

Graph Reductions and Hardness Results

We show that a number splitting functions can be modeled by replacing a hyperedge with a small directed graph that preserves the cut penalties of the splitting function. This approach is related to the techniques Lawler applied to the all-or-nothing problem, but is significantly more general and enables us to model a broad range of splitting functions. One of the central results of our work is to completely characterize which hypergraph s-t cut problems can be reduced to a graph s-t cut problem.

Theorem 1. *The cardinality-based hypergraph s-t cut problem can be reduced to a graph s-t cut problem if and only if all splitting functions are submodular.*

Outside the submodular regime, we demonstrate cases where the problem is NP-hard, via reduction from maxcut.

Theorem 2. *In uniform hypergraphs, cardinality-based hypergraph s-t cut is NP-hard if there exists some $j > 1$ such that separating j nodes from the rest of the hyperedge has a smaller penalty than separating only one node.*

Proofs are available in an online preprint [4].

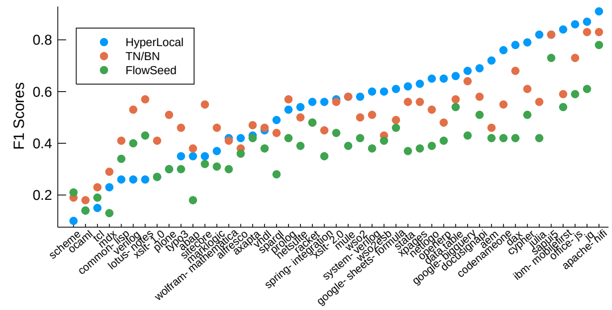


Figure 2: F1 scores for each of 45 clusters in the Stackoverflow hypergraph. FlowSeed and BN/TN are competing methods. Our method, HyperLocal, uses generalized cut functions and has the highest average F1 score.

Applications to Local Clustering

We can incorporate our techniques into a broader framework for local hypergraph clustering [5]. Let $R \subseteq V$ be a small subset of nodes in a hypergraph $\mathcal{H} = (V, E)$. One way to find clusters nearby R is to solve:

$$\underset{S \subseteq V}{\text{minimize}} \;\; \mathbf{cut}_{\mathcal{H}}(S) / (\mathbf{vol}(R \cap S) - \varepsilon \mathbf{vol}(\bar{R} \cap S)), \quad (3)$$

where $\mathbf{cut}_{\mathcal{H}}(S) = \sum_{e \in E} \mathbf{w}_e(S \cap e)$ is a generalized cut function, $\mathbf{vol}$ measures the *volume* of a node set, and $\varepsilon > 0$ controls the overlap between the input R and the output S. We can minimize objective (3) by solving sequence of localized minimum s-t cut problems. As an example, we solve this objective over a hypergraph made up of question posts (nodes) on `stackoverflow.com`, organized into sets of posts answered by the same user (hyperedges). The hypergraph has 15 million nodes, 1.1 million hyperedges, and a maximum hyperedge size of 61,315. We optimize (3) to find sets of posts with the same tag (e.g. "netsuite"), by starting with a seed set R containing a small subset of nodes with the same tag. Using a single-parameter family of generalized splitting functions, $\mathbf{w}_e(A) = \min\{|A|, |e \backslash A|, \delta\}$, we detect these clusters better than baseline methods (Figure 2).

References

[1] C. J. Alpert and A. B. Kahng. Recent directions in netlist partitioning: a survey. *Integration*, 19(1):1 – 81, 1995.

[2] E. L. Lawler. Cutsets and partitions of hypergraphs. *Networks*, 3(3):275–285, 1973.

[3] P. Li and O. Milenkovic. Inhomogeneous hypergraph clustering with applications. In *NeurIPs*, pages 2308–2318. 2017.

[4] N. Veldt, A. R. Benson, and J. Kleinberg. Hypergraph cuts with general splitting functions (arxiv.org/abs/2001.02817), 2020.

[5] N. Veldt, A. R. Benson, and J. Kleinberg. Localized flow-based clustering in hypergraphs (submitted), 2020.