Splitting-Off in Hypergraphs

Kristóf Bérczi
MTA-ELTE Matroid Optimization Research Group and HUN-REN-ELTE Egerváry Research Group, Department of Operations Research, Eötvös Loránd University, Budapest, Hungary

Karthekeyan Chandrasekaran
University of Illinois, Urbana-Champaign, IL, USA

Tamás Király
MTA-ELTE Matroid Optimization Research Group and HUN-REN-ELTE Egerváry Research Group, Department of Operations Research, Eötvös Loránd University, Budapest, Hungary

Shubhang Kulkarni
University of Illinois, Urbana-Champaign, IL, USA

Abstract

The splitting-off operation in undirected graphs is a fundamental reduction operation that detaches all edges incident to a given vertex and adds new edges between the neighbors of that vertex while preserving their degrees. Lovász [45, 47] and Mader [48] showed the existence of this operation while preserving global and local connectivities respectively in graphs under certain conditions. These results have far-reaching applications in graph algorithms literature [2, 7, 8, 12, 17, 22, 23, 24, 25, 26, 29, 30, 32, 33, 35, 38, 40, 41, 46, 48, 49, 50, 51]. In this work, we introduce a splitting-off operation in hypergraphs. We show that there exists a local connectivity preserving complete splitting-off in hypergraphs and give a strongly polynomial-time algorithm to compute it in weighted hypergraphs. We illustrate the usefulness of our splitting-off operation in hypergraphs by showing two applications: (1) we give a constructive characterization of k-hyperedge-connected hypergraphs and (2) we give an alternate proof of an approximate min-max relation for max Steiner rooted-connected orientation of graphs and hypergraphs (due to Király and Lau [38]). Our proof of the approximate min-max relation for graphs circumvents the Nash-Williams' strong orientation theorem and uses tools developed for hypergraphs.

2012 ACM Subject Classification Theory of computation → Network optimization; Theory of computation → Routing and network design problems

Keywords and phrases Hypergraphs, Hypergraph Connectivity, Splitting-off, Constructive Characterizations, Hypergraph Orientations, Submodular Functions, Combinatorial Optimization

Digital Object Identifier 10.4230/LIPIcs.ICALP.2024.23

Category Track A: Algorithms, Complexity and Games

Related Version This is an extended abstract.
Full Version: `https://arxiv.org/abs/2307.08555`

Funding Karthekeyan and Shubhang were supported in part by NSF grants CCF-1814613 and CCF-1907937. Karthekeyan was supported in part by the Distinguished Guest Scientist Fellowship of the Hungarian Academy of Sciences – grant number VK-6/1/2022. Kristóf and Tamás were supported in part by the Lendület Programme of the Hungarian Academy of Sciences – grant number LP2021-1/2021, by the Ministry of Innovation and Technology of Hungary from the National Research, Development and Innovation Fund – grant number ELTE TKP 2021-NKTA-62 funding scheme, and by the Dynasnet European Research Council Synergy project – grant number ERC-2018-SYG 810115.

Acknowledgements Part of this work was done while Karthekeyan and Shubhang were visiting Eötvös Loránd University. Karthekeyan thanks Eklavya Sharma for engaging in preliminary discussions on hypergraph splitting-off.

1 Introduction

The splitting-off operation in undirected graphs is a simple yet powerful operation in graph theory. It is a reduction operation that detaches all edges incident to a given vertex and adds new edges between the neighbors of that vertex while preserving their degrees. Equivalently, it enables a vertex to exit the network by informing its neighbors how to reconfigure the lost links among themselves in order to preserve their degrees. Lovász [45, 47] introduced the splitting-off operation and showed the existence of the operation to preserve global edge-connectivity under certain conditions. Mader [48] showed the existence of the splitting-off operation to preserve local edge-connectivity (i.e., all pairwise edge-connectivities) under certain conditions. Both Lovász's and Mader's results also admit strongly polynomial-time algorithms [24, 27, 49]. Owing to the inductive nature of the splitting-off operation, Lovász's and Mader's results have enabled fundamental results in graph theory as well as efficient algorithms and min-max relations for numerous graph optimization problems. In fact, Mader [48] illustrated the power of his local edge-connectivity preserving splitting-off result by deriving Nash-Williams' strong orientation theorem [52] (also see Frank's exposition of this derivation [26]). Subsequently, the splitting-off operation has been used to give a constructive characterization of k-edge-connected graphs [24] and to address problems in edge-connectivity augmentation [2, 22, 23, 24, 49], graph orientation [25, 38], minimum cuts enumeration [29, 32, 50], network design [12, 30, 35], tree packing [7, 41], congruency-constrained cuts [51], and approximation algorithms for TSP [8, 30]. Designing fast algorithms for global/local edge-connectivity preserving splitting-off remains an active area of research (e.g., see recent works [9, 10, 11, 42]) due to these far-reaching applications. In this work, we introduce a splitting-off operation in *hypergraphs*, show the existence of local-connectivity preserving splitting-off operation and design a strongly polynomial-time algorithm to compute it in weighted hypergraphs, and illustrate its usefulness by showing two applications.

Hypergraphs. Hypergraphs offer a richer and more accurate model than graphs for several applications. Consequently, hypergraphs have found applications in several modern areas (e.g., see [44, 54, 57, 60]) and these applications have, in turn, driven exciting recent progress in algorithms for hypergraph optimization problems [1, 4, 13, 14, 15, 18, 19, 21, 28, 31, 34, 36, 37, 39, 43, 55, 58]. A hypergraph $G = (V, E)$ consists of a finite set V of vertices and a set E of hyperedges, where every hyperedge $e \in E$ is a subset of V. We will denote a hypergraph $G = (V, E)$ with hyperedge weights $w : E \to \mathbb{Z}_+$ by the tuple (G, w) and use G_w to denote the unweighted multi-hypergraph over vertex set V containing $w(e)$ copies of every hyperedge $e \in E$. Throughout this work, we will be interested only in hypergraphs with positive integral weights and for algorithmic problems where the input/output is a hypergraph, we will require that the weights are represented in binary. If all hyperedges have size at most 2, then the hyperedges are known as edges and we call such a hypergraph as a graph. We emphasize a subtle but important distinction between hypergraphs and graphs: the number of hyperedges in a hypergraph could be exponential in the number of vertices. This is in sharp contrast to graphs where the number of edges is at most the square of the number of vertices. Consequently, in hypergraph network design problems where the goal is to construct a hypergraph with certain properties, one needs to be mindful of the number of hyperedges in the hypergraph returned by the algorithm. Recent works in hypergraph algorithms literature have focused on this issue in the context of cut/spectral sparsification of hypergraphs [4, 18, 19, 34, 36, 37, 39, 43, 55, 58].

Notation. Let $(G = (V, E), w : E \to \mathbb{Z}_+)$ be a hypergraph. For $X \subseteq V$, let $\delta_G(X) := \{e \in E : e \cap X \neq \emptyset, e \setminus X \neq \emptyset\}$. We define the cut function $d_{(G,w)} : 2^V \to \mathbb{Z}_{\geq 0}$ by $d_{(G,w)}(X) := \sum_{e \in \delta_G(X)} w(e)$ for every $X \subseteq V$. For a vertex $v \in V$, we use $d_{(G,w)}(v)$ to denote $d_{(G,w)}(\{v\})$. We define the degree of a vertex v to be the sum of the weights of hyperedges containing v – we note that the degree of a vertex is not necessarily equal to $d_{(G,w)}(v)$ since we could have $\{v\}$ itself as a hyperedge (i.e., a singleton hyperedge that contains only the vertex v). For distinct vertices $u, v \in V$, let $\lambda_{(G,w)}(u, v) := \min\{d_{(G,w)}(X) : u \in X \subseteq V \setminus \{v\}\}$ – i.e., $\lambda_{(G,w)}(u, v)$ is the value of a minimum $\{u, v\}$-cut in the hypergraph. If all hyperedge weights are unit, then we drop w from the subscript and simply use d_G and λ_G.

Hypergraph Splitting-off. We now introduce our definition of splitting-off in hypergraphs. To compare and contrast our definition of splitting-off for hypergraphs with the classical definition of splitting-off for graphs, we include both our definition and the classical definition and distinguish them by identifying them as h-splitting-off and g-splitting-off. In the definitions below, we encourage the reader to consider the input hypergraph (G, w) to be a graph while considering g-splitting-off terminology and to be a hypergraph while considering h-splitting-off terminology. We encourage the reader to also assume unit weights during first read. See Figure 1 for an example.

▶ **Definition 1.** *Let $(G = (V, E), w : E \to \mathbb{Z}_+)$ be a hypergraph and $s \in V$.*

1. *In* merge almost-disjoint hyperedges*, we pick a pair of hyperedges $e, f \in \delta_G(s)$ such that $e \cap f = \{s\}$, pick a positive integer $\alpha \in \mathbb{Z}_+$ such that $\alpha \leq \min\{w(e), w(f)\}$, reduce the weights of hyperedges e and f by α, and increase the weight of a hyperedge g by α. Here,*
   a. *if we choose $g := e \cup f$, then the associated operation will be called as* h-merge almost-disjoint hyperedges operation*.*
   b. *if we choose $g := (e \cup f) \setminus \{s\}$, then the associated operation will be called as* g-merge almost-disjoint hyperedges operation*.*

   In the above, if $\alpha = w(e)$ (resp. if $\alpha = w(f)$), then we discard the hyperedge e (resp. hyperedge f) from the hypergraph obtained after the operation; if the hyperedge $g \notin E$, then we introduce g as a new hyperedge with weight $w(g) := 0$ before performing the weight increase on the hyperedge g.
2. *In* trim hyperedge operation*, we pick a hyperedge $e \in \delta_G(s)$, pick a positive integer $\alpha \in \mathbb{Z}_+$, reduce the weight of the hyperedge e and increase the weight of the hyperedge $g := e \setminus \{s\}$. Here,*
   a. *if we choose $\alpha \leq w(e)$, reduce the weight of the hyperedge e by α, and increase the weight of the hyperedge g by α, then the associated operation will be called as* h-trim operation *(if $\alpha = w(e)$, then we discard e from the hypergraph obtained after the operation; if $g \notin E$, then we add g as a new hyperedge with weight $w(g) := 0$ before performing the weight increase on the hyperedge g).*
   b. *if we choose $\alpha \leq w(e)/2$, reduce the weight of the hyperedge e by 2α, and increase the weight of the hyperedge g by 2α, then the associated operation will be called as* g-trim operation *(if $\alpha = w(e)/2$, then we discard e from the hypergraph obtained after the operation; if $g \notin E$, then we add g as a new hyperedge with weight $w(g) := 0$ before performing the weight increase on the hyperedge g).*
3. *We say that a hypergraph $(H = (V, E_H), w_H : E_H \to \mathbb{Z}_+)$ is obtained by applying a*
   a. h-splitting-off operation at s from (G, w) *if (H, w_H) is obtained from (G, w) by either the h-merge almost-disjoint hyperedges operation or the h-trim hyperedge operation.*
   b. g-splitting-off operation at s from (G, w) *if (H, w_H) is obtained from (G, w) by either the g-merge almost-disjoint hyperedges operation or the g-trim hyperedge operation.*

Certain remarks regarding the definitions are in order. Firstly, the trim operation is valuable and unique to hypergraphs. It has been used in the hypergraph literature to obtain small-sized certificates for hypergraph connectivity [18] and for certain notions of directed hypergraph connectivity [24]. We note that the trim operation has limited value in graphs – trimming an edge leads to a singleton edge and singleton edges contribute only to the degree but not to the cut value of any set. Secondly, all operations mentioned above are degree preserving for vertices $u \in V \setminus \{s\}$: both h-trim and g-trim operations preserve degrees by definition; both h-merge and g-merge *almost-disjoint* hyperedges operations preserve degrees due to the almost-disjoint property of the chosen hyperedges. Thirdly, all operations mentioned above *do not* increase the cut values of subsets $X \subseteq V \setminus \{s\}$. Thus, the relevant goal with these operations is ensuring that the cut values do not decrease too much – i.e., preserving global/local connectivity. We will be interested in repeated application of h-splitting-off operations at a vertex from a given hypergraph to isolate that vertex while preserving global/local connectivity. We define these formally next.

▶ **Definition 2.** *Let $(G = (V, E), w : E \to \mathbb{Z}_+)$ be a hypergraph and $s \in V$.*

1. *We say that a hypergraph $(G^* = (V, E^*), w^* : E^* \to \mathbb{Z}_+)$ is a*
   a. complete h-splitting-off at s from (G, w) *if $d_{(G^*,w^*)}(s) = 0$ and (G^*, w^*) is obtained from (G, w) by repeatedly applying h-splitting-off operations at s from the current hypergraph.*
   b. complete g-splitting-off at s from (G, w) *if $d_{(G^*,w^*)}(s) = 0$ and (G^*, w^*) is obtained from (G, w) by repeatedly applying g-splitting-off operations at s from the current hypergraph.*
2. *Let (G^*, w^*) be a complete h-splitting-off/g-splitting-off at s from (G, w). We say that (G^*, w^*)*
   a. *preserves local connectivity if $\lambda_{(G^*,w^*)}(u, v) = \lambda_{(G,w)}(u, v)$ for every distinct $u, v \in V \setminus \{s\}$ and*
   b. *preserves global connectivity if*
   $\min\{\lambda_{(G^*,w^*)}(u, v) : u, v \in V \setminus \{s\}, u \neq v\} = \min\{\lambda_{(G,w)}(u, v) : u, v \in V \setminus \{s\}, u \neq v\}$.

Our first contribution in this work is the definition of h-splitting-off operations at a vertex from a hypergraph. To the best of our knowledge, this definition has not appeared in the literature before. A notion of hypergraph splitting-off motivated by hypergraph connectivity augmentation applications has been studied in the literature before [3, 6, 20]. These works have explored local connectivity preserving complete g-splitting-off at a vertex s from a hypergraph under the assumption that *all hyperedges incident to the vertex s are edges (i.e., have size at most 2)*. In contrast, our focus is on local connectivity preserving complete h-splitting-off at a vertex s from a hypergraph without any assumption on the size of the hyperedges incident to the vertex s (i.e., the vertex s could have arbitrary-sized hyperedges incident to it). We refer the reader to Figure 1 for an example of complete h-splitting-off at a vertex from a hypergraph.

We will primarily be concerned with complete *h-splitting-off* at a vertex from a *hypergraph* and complete *g-splitting-off* at a vertex from a *graph*. Complete g-splitting-off at a vertex from a *graph* is equivalent to the classical and well-studied notion of complete splitting-off at a vertex from a graph (for the definition of the classical notion in graphs, see [24, 49]). We cast the results of Lovász [45, 47] and Mader [48] in the framework of our definitions now. Let (G, w) be a *graph* and let s be a vertex in G. Lovász [45, 47] showed that if $d_{(G,w)}(\{s\})$ is even and $\min\{\lambda_{(G,w)}(u, v) : u, v \in V \setminus \{s\}\} \geq K$ for some $K \geq 2$, then there exists a *global* connectivity preserving complete g-splitting-off at the vertex s from (G, w).

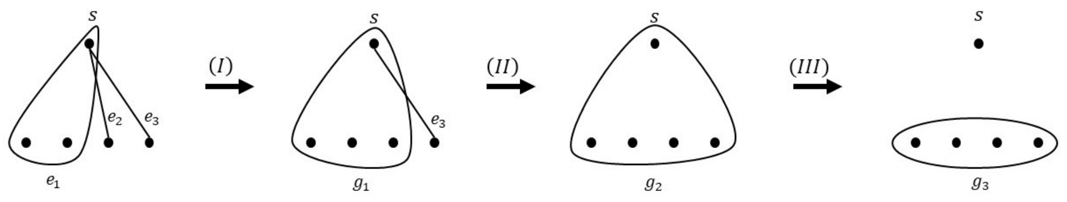

Figure 1 Example of (local connectivity preserving) complete h-splitting-off at a vertex s from a hypergraph. Consider the leftmost hypergraph where all hyperedge weights are one and the vertex s is as labeled. Operations (I) and (II) correspond to h-merge almost-disjoint hyperedges operations and Operation (III) corresponds to an h-trim hyperedge operation.

Mader [48] showed that if $d_{(G,w)}(\{s\})$ is even, there is no cut-edge[1] incident to s, and (G, w) is connected, then there exists a *local* connectivity preserving complete g-splitting-off at the vertex s from (G, w).

We compare and contrast complete h-splitting-off at a vertex from a hypergraph and complete g-splitting-off at a vertex from a graph. Complete h-splitting-off at a vertex s from a hypergraph enables the vertex s to exit the hypergraph by informing its incident hyperedges about how to merge and trim themselves in order to preserve degrees. In this sense, the definition of complete h-splitting-off at a vertex from a hypergraph serves the same role as complete g-splitting-off at a vertex from a graph. On the other hand, there are important differences between the two notions. Firstly, complete h-splitting-off at a vertex from a *graph* may not necessarily be a graph (owing to the creation of hyperedges of size at least 3) while it is an easy exercise to show that complete g-splitting-off at a vertex from a *graph* will necessarily be a graph. Secondly, local/global connectivity preserving complete g-splitting-off at a vertex from a graph may not exist – see Figure 2.

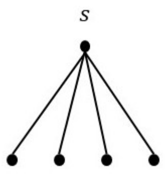

Figure 2 An example showing that global connectivity preserving complete g-splitting-off at a vertex from a graph may not exist. All edge weights are one and the vertex s is as labeled.

As our second main contribution, we show that local connectivity preserving complete h-splitting-off at a vertex from a hypergraph always exists and can be computed in strongly polynomial time (the rightmost hypergraph in Figure 1 is a local connectivity preserving complete h-splitting-off at the vertex s from the hypergraph in Figure 2).

▶ **Theorem 3.** *Given a hypergraph $(G = (V, E), w_G : E \to \mathbb{Z}_+)$ and a vertex $s \in V$, there exists a strongly polynomial-time algorithm to find a local connectivity preserving complete h-splitting-off at s from (G, w_G).*

A difference between Theorem 3 and the graph splitting-off results of Lovász and Mader is that Theorem 3 shows the existence of a local connectivity preserving complete h-splitting-off at a vertex from a hypergraph *without any assumptions on the hypergraph* whereas Lovász's

[1] Equivalently, for every edge $e \in \delta_G(s)$ with $w(e) = 1$, the removal of that edge does not disconnect the graph.

and Mader's results hold only under certain technical assumptions on the graph. In several applications of their results, additional arguments are needed to address cases where those technical assumptions do not hold. For this reason, we believe that Theorem 3 could be useful in simplifying the arguments involved in some of the applications of Lovász's and Mader's graph splitting-off results (e.g., we will later see that the edge version of Menger's theorem in undirected *graphs* follows in a straightforward fashion from Theorem 3).

A crude run-time of our algorithm that proves Theorem 3 is $O(|V|^6(|V|+|E|)^3)$. We understand that this run-time is impractical for applications. Nevertheless, we mention it here explicitly for the sake of completeness and as a potential starting point for future work: it would be interesting to design a near-linear time algorithm to find a local connectivity preserving complete h-splitting-off at a vertex from a weighted hypergraph.

▶ Remark 4. We note that existence of a local/global connectivity preserving complete h-splitting-off at a vertex from a hypergraph does not necessarily imply a polynomial-time algorithm to find it. This is because, a local/global connectivity preserving complete h-splitting-off at a vertex from a hypergraph (G, w_G) could contain exponential number of hyperedges although G contains only polynomial number of hyperedges. We give an example to illustrate this issue. Consider the graph $(G = (V, E), w_G)$ where G is the star graph on $n+1$ vertices with s being the center of the star and all edge weights are $2^{n-1} - 1$. Consider the hypergraph $(H = (V, E_H), w_H)$ such that $E_H := \{e : e \subseteq V \setminus \{s\} \text{ and } |e| \geq 2\}$ with all hyperedge weights being one. The hypergraph (H, w_H) is a local connectivity preserving complete h-splitting-off at s from (G, w_G), but (H, w_H) has exponential number of hyperedges although (G, w_G) has only n edges. In order to design a polynomial-time algorithm to find a local connectivity preserving complete h-splitting-off at a vertex from a hypergraph, a necessary step is to show the existence of a local connectivity preserving complete h-splitting-off at a vertex from a hypergraph that contains only polynomially many *additional* hyperedges. For the star graph (G, w_G) with edge weights $2^{n-1} - 1$ mentioned above, the hypergraph $(H' = (V, E_{H'}), w_{H'})$ containing only one hyperedge, namely $E_{H'} := \{V \setminus \{s\}\}$ with the weight of that hyperedge being $2^{n-1} - 1$ is also a local connectivity preserving complete h-splitting-off at s from (G, w_G). One of the features of our algorithmic proof of Theorem 3 is the existence of a local connectivity preserving complete h-splitting-off at a vertex from a hypergraph that contains only polynomially many additional hyperedges.

As our third main contribution, we present two applications of Theorem 3.

Application 1: Constructive characterization of k-hyperedge-connected hypergraphs. For the purposes of this application, graphs and hypergraphs will refer to multi-graphs and multi-hypergraphs, respectively. Let k be a positive integer. A graph $G = (V, E)$ is *k-edge-connected* if $d_G(X) \geq k$ for every non-empty proper subset $X \subsetneq V$. Constructive characterization of k-edge-connected graphs is a central problem in graph theory. It is well-known that a graph is 1-edge-connected if and only if it containss a spanning tree. Robbins' [56] showed that a graph is 2-edge-connected if and only if it admits an ear decomposition (see [24] for definition of ear decomposition). Generalizing Robbins' result, Lovász [45, 47] gave a constructive characterization of k-edge-connected graphs for *even k* using his result on global connectivity preserving complete g-splitting-off at a vertex from a graph. Mader [48] gave a constructive characterization of k-edge-connected graphs for *odd k* using his result on local connectivity preserving complete g-splitting-off at a vertex from a graph. Motivated by these results, we present a constructive characterization of k-hyperedge-connected hypergraphs using our splitting-off result in Theorem 3. A hypergraph $G = (V, E)$ is defined to be *k-hyperedge-connected* if $d_G(X) \geq k$ for every non-empty proper subset $X \subsetneq V$.

Both Lovász's and Mader's constructive characterizations of k-edge-connected graphs are based on a pinching operation in graphs. Our constructive characterization of k-hyperedge-connected hypergraphs is also based on a pinching operation, but our pinching operation is defined for hypergraphs. We define this operation now (see Figure 3 for an example).

▶ **Definition 5.** *Let $G = (V, E)$ be a hypergraph and $p, k \in \mathbb{Z}_+$ be positive integers such that $p \leq k$. In (k, p)-*pinching *hyperedges of G, we obtain a new hypergraph by performing the following sequence of operations:*

1. *pick p distinct hyperedges $e_1, \dots, e_p \in E$,*
2. *pick p positive integers $t_1, \dots, t_p \in \mathbb{Z}_+$ such that $\sum_{i=1}^{p} t_i = k$,*
3. *for each $i \in [p]$, choose a partition of the hyperedge e_i into t_i non-empty parts $e_i = \uplus_{j \in [t_i]} f_i^j$,*
4. *remove the hyperedges $e_1, \dots, e_p$ from the hypergraph G,*
5. *add a new vertex s and hyperedges $\{f_i^j \cup \{s\} : j \in [t_i], i \in [p]\}$ to the hypergraph G.*

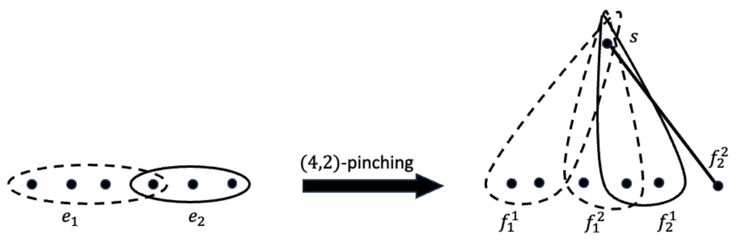

Figure 3 An example of a $(4, 2)$-pinching operation. Here, $t_1 = t_2 = 2$.

With this definition of pinching, we show the following constructive characterization of k-hyperedge-connected hypergraphs.

▶ **Theorem 6.** *Let $k \in \mathbb{Z}_+$ be a positive integer. A hypergraph $G = (V, E)$ is k-hyperedge-connected if and only if G can be obtained by starting from the single vertex hypergraph with no hyperedges and repeatedly applying one of the following two operations:*

1. *add a new hyperedge over a subset of vertices of the existing hypergraph, and*
2. *(k, p)-pinching hyperedges of the existing hypergraph for some positive integer $p \leq k$.*

Our proof of Theorem 6 is constructive: i.e., given a k-hyperedge-connected hypergraph G, our proof gives a polynomial-time algorithm to construct a sequence of hypergraphs $G_0, G_1, G_2, \dots, G_t$, where G_0 is the single vertex hypergraph with no hyperedges, $G_t = G$ and for each $i \in [t]$, the hypergraph G_i is obtained from G_{i-1} by either adding a new hyperedge over a subset of vertices in G_{i-1} or by (k, p)-pinching hyperedges in G_{i-1} for some positive integer $p \leq k$.

▶ Remark 7. Robbins' constructive characterization of 2-edge-connected graphs and Lovász's constructive characterization of $2k$-edge-connected graphs find applications in graph orientation problems – e.g., Robbins' result leads to an algorithm to find a strongly connected orientation of 2-edge-connected graphs and Lovász's result leads to an algorithm to find a strongly k-arc-connected orientation of $2k$-edge-connected graphs. In fact, the latter leads to an alternative proof of Nash-Williams' *weak orientation theorem* [52]. Along the same vein, we hope that our above characterization of k-edge-connected hypergraphs might find applications in hypergraph orientation problems.

Application 2.1: Steiner Rooted k-arc-connected Orientation of Graphs. Orienting a graph to achieve high connectivity is a fundamental area in graph theory, combinatorial optimization, and algorithms. Let $G = (V, E)$ be an undirected graph. An *orientation* $\overrightarrow{G}$ of G is a directed graph obtained by assigning a direction to each edge of G. Let $G = (V, E)$ be an undirected graph, $T \subseteq V$ be a set of terminals, $r \in T$ be a root vertex, and k be a positive integer. An orientation $\overrightarrow{G}$ of G is defined to be *Steiner rooted k-arc-connected* if there exist k arc-disjoint paths in $\overrightarrow{G}$ from t to r for every terminal $t \in T \setminus \{r\}$. In Max Steiner Rooted-Connected Orientation problem, the goal is to find the maximum integer k and an orientation $\overrightarrow{G}$ of G such that $\overrightarrow{G}$ is Steiner rooted k-arc-connected. Max Steiner Rooted-Connected Orientation generalizes two classic problems in graph theory: The case of $|T| = 2$ is the max edge-disjoint $\{r, v\}$-paths problem and is solved via Menger's theorem. The case of $T = V$ is the max edge-disjoint spanning trees problem and is solved via Tutte and Nash-Williams' theorem [53, 59]. We mention that both these problems are also generalized by the Steiner Tree Packing problem and the associated Kriesell's conjecture [33, 40, 41], but we will not focus on that generalization.

Király and Lau [38] introduced the Max Steiner Rooted-Connected Orientation, showed that it is NP-hard, and gave a 2-approximation via an approximate min-max relation. We state their approximate min-max relation now. An undirected graph G is *Steiner k-edge-connected* if $\lambda_G(u, v) \geq k$ for every pair of distinct terminals $u, v \in T$. It is clear that if the graph G has a Steiner rooted k-arc-connected orientation, then G should be Steiner k-edge-connected. However, the converse is not necessarily true. Király and Lau observed that if the graph is Steiner $2k$-edge-connected, then it has a Steiner rooted k-arc-connected orientation.

▶ **Theorem 8** (Király and Lau [38]). *Let $G = (V, E)$ be an undirected graph, $T \subseteq V$ be a subset of terminals, $r \in T$ be the root vertex, and k be a positive integer. If G is Steiner $2k$-edge-connected, then it has a Steiner rooted k-arc-connected orientation.*

Király and Lau observed that Theorem 8 follows immediately from Nash-Williams' *strong orientation* theorem. Nash-Williams' *strong orientation* theorem [52] states that every undirected graph $G = (V, E)$ admits an orientation $\overrightarrow{G}$ such that $\lambda_{\overrightarrow{G}}(u, v) \geq \lfloor \lambda_G(u, v)/2 \rfloor$ for every distinct $u, v \in V$, where $\lambda_{\overrightarrow{G}}(u, v)$ is the maximum number of arc-disjoint directed paths from u to v in $\overrightarrow{G}$. In this work, we give an alternative proof of Theorem 8 that does not rely on Nash-Williams' strong orientation theorem. Instead, we use Theorem 3. Our proof strategy is unique since it proves an orientation result for *graphs* using tools developed for *hypergraphs*.

▶ Remark 9. Nash-Williams' proof of the strong orientation theorem [52] is a sophisticated inductive argument. Giving a simple and more insightful proof of the strong orientation theorem has been a central topic of interest in graph theory and combinatorial optimization (see [26]). Mader [48] gave a different proof of the strong orientation theorem using his local connectivity preserving splitting-off theorem, but his proof also involved sophisticated technical arguments. Frank [26] condensed the ideas of both Nash-Williams and Mader to present a proof of the strong orientation theorem using Mader's local connectivity preserving splitting-off, but it is still technically complicated. The technical complication in using Mader's local connectivity preserving splitting-off result arises from the assumptions that need to be satisfied by the vertex to be split-off. In contrast, our splitting-off result for hypergraphs (namely, Theorem 3) does not need any assumptions on the vertex to be split-off. In light of these considerations, our proof of Theorem 8 using Theorem 3 provides hope that Theorem 3 (or the ideas therein) could be used to give a conceptually simpler proof of Nash-Williams' strong orientation theorem.

Application 2.2: Steiner Rooted k-hyperarc-connected Orientation of Hypergraphs. Orienting *hypergraphs* is also a fundamental area in graph theory and combinatorial optimization (see Frank's book [24]) with far reaching implications. For example, Woodall's conjecture can be reformulated as a hypergraph orientation problem (see Conjecture 9.4.15 in [24]); moreover, hypergraph orientation results have recently been used in coding theory [1]. Király and Lau [38] showed that the approximate min-max relation in Theorem 8 also holds for hypergraphs. To state their result, we need some terminology in hypergraph orientations.

Let $G = (V, E)$ be a hypergraph. An *orientation* $\overrightarrow{G} = (V, E, \texttt{head} : E \to V)$ of G is a directed hypergraph obtained by assigning a unique head vertex $\texttt{head}(e) \in e$ for each $e \in E$. A pair $(e, \texttt{head}(e))$ is denoted as a *hyperarc* with the head of the hyperarc being $\texttt{head}(e)$ and the tails of the hyperarc being $e \setminus \texttt{head}(e)$. Let $G = (V, E)$ be a hypergraph, $T \subseteq V$ be a set of terminals, $r \in T$ be a root vertex, and k be a positive integer. An orientation $\overrightarrow{G}$ of G is defined to be *Steiner rooted k-hyperarc-connected* if there exist k hyperarc-disjoint paths in $\overrightarrow{G}$ from t to r for every terminal $t \in T \setminus \{r\}$. Here, a path from t to r in a directed hypergraph is an alternating sequence of distinct vertices and hyperarcs $t = v_1, a_1, v_2, a_2, ..., a_{\ell-1}, v_\ell = r$ such that v_i is a tail of a_i and v_{i+1} is the head of a_i for every $i \in [\ell - 1]$. We say that a hypergraph G is *Steiner k-hyperedge-connected* if $\lambda_G(u, v) \geq k$ for every pair of distinct terminals $u, v \in T$. It is clear that if the hypergraph G has a Steiner rooted k-hyperarc-connected orientation, then G should be Steiner k-hyperedge-connected. However, the converse is not necessarily true. Király and Lau [38] showed that if the hypergraph is Steiner $2k$-hyperedge-connected, then it has a Steiner rooted k-hyperarc-connected orientation.

▶ **Theorem 10** (Király and Lau [38]). *Let $G = (V, E)$ be a hypergraph, $T \subseteq V$ be a subset of terminals, $r \in T$ be the root vertex, and k be a positive integer. If G is Steiner $2k$-hyperedge-connected, then it has a Steiner rooted k-hyperarc-connected orientation.*

Király and Lau's proof of Theorem 10 was based on careful uncrossing and contractions. In this work, we give an alternative proof of Theorem 10 using Theorem 3. Our proof of Theorem 10 reveals the source of the 2-factor gap in the approximate min-max relation of Király and Lau for Max Steiner Rooted-Connected Orientation Problem: it arises from the 2-factor gap between connectivity and weak-partition-connectivity of hypergraphs (see Definition 5.1 for the definition of weak-partition-connectivity and Lemma 5.2 in the full version).

▶ Remark 11. Our proof technique for Theorems 8 and 10 using Theorem 3 – i.e., via the local-connectivity preserving splitting-off operation in hypergraphs – also leads to an alternate proof of Menger's theorem in *undirected* graphs and hypergraphs (edge-disjoint version). For details, we refer the reader to Section 5 of the full version where we discuss a proof of Menger's theorem using Theorem 3 as a warm-up towards a proof of Theorems 8 and 10.

Both Theorems 8 and 10 can be extended to weighted graphs/hypergraphs (by considering parallel copies of edges/hyperedges). The weighted version of Theorems 8 and 10 can also be shown to admit strongly polynomial-time algorithms using our proof strategy as well as the proof strategy of Király and Lau. We avoid stating the weighted versions in the interests of brevity.

1.1 Proof Technique for Theorem 6

We outline the proof technique for Theorem 6. The reverse direction follows by observing that if a hypergraph is k-hyperedge-connected, then both operations in the statement of the theorem preserve k-hyperedge-connectivity. We sketch a proof of the forward direction. The proof is by induction on the number of hyperedges plus the number of vertices. First, suppose that there exists a hyperedge $e \in E$ such that $G - e$ is still k-hyperedge-connected. We note that deleting the hyperedge e is the inverse of operation (1). Consequently, the proof follows by deleting the hyperedge e, using the induction hypothesis on the resulting hypergraph H, and then noting that the hypergraph G is obtained from H by operation (1). Next, suppose that there does not exist a hyperedge $e \in E$ such that $G - e$ is k-hyperedge-connected. We call such a hypergraph to be minimally k-hyperedge-connected. In Lemma 4.1 of the full version, we show that a minimally k-hyperedge-connected hypergraph contains a vertex u with degree exactly k. By Theorem 3, there exists a global-connectivity preserving complete h-splitting-off at the vertex u from the hypergraph G. Let H be a global-connectivity preserving complete h-splitting-off at the vertex u from the hypergraph G. We note that complete h-splitting-off at u followed by deletion of the vertex u is the inverse of operation (2) at u. Consequently, the proof follows by using the induction hypothesis on the hypergraph $H - u$ and then noting that the hypergraph G is obtained from $H - u$ by operation (2).

1.2 Proof Technique for Theorems 8 and 10

We outline the proof technique for Theorem 10 and will remark after the proof about how it also implies a proof for Theorem 8. Our proof of Theorem 10 will be in three steps. Let us denote the set of non-terminals as *Steiner vertices*. Our first step is to obtain a hypergraph $H = (T, E_H)$ by applying our local connectivity preserving complete h-splitting-off at each Steiner vertex of G (sequentially, in arbitrary order of the Steiner vertices) and deleting the isolated vertices. We note that deleting the isolated vertices ensures that the vertex set of H is the set of terminals T. Moreover, our local connectivity preserving complete h-splitting-off ensures that the hypergraph H is $2k$-hyperedge-connected since the hypergraph G is Steiner $2k$-hyperedge-connected. Our second step is to show that this hypergraph $H = (T, E_H)$ admits a rooted k-hyperarc-connected orientation. A known characterization for the existence of a rooted k-hyperarc-connected orientation of a hypergraph is that the hypergraph is k-*weak-partition-connected* (see Definition 5.1 for the definition of weak-partition-connectivity and Theorem 5.1 for the characterization in the full version). We mention that the notion of weak-partition-connectivity in hypergraphs has been used recently in the context of coding theory [1, 31]. In order to use the characterization for the existence of a rooted k-hyperarc-connected orientation of a hypergraph, we relate the connectivity of a hypergraph to its *weak-partition-connectivity* and conclude that if H is $2k$-hyperedge-connected, then it is k-weak-partition-connected (see Lemma 5.2 in the full version). Consequently, the hypergraph H admits a rooted k-hyperarc-connected orientation. We note that such an orientation of H is equivalent to a Steiner rooted k-hyperarc-connected orientation of H since the vertex set of H is the set T of terminals. Our third step is to use this Steiner rooted k-hyperarc-connected orientation of H to obtain a Steiner rooted k-hyperarc-connected orientation of the hypergraph G: we will see that there is a natural way to extend the orientation of hyperedges while reversing the h-splitting-off operations to preserve Steiner rooted k-hyperarc-connected property (see Lemma 5.1 in the full version). This would complete the proof of Theorem 10.

We note that if the hypergraph G is a graph, then the same proof above obtains the required graph orientation, thus proving Theorem 8. In particular, to prove Theorem 8, we start from a graph $G = (V, E)$ that is Steiner $2k$-edge-connected, but our local connectivity preserving complete h-splitting-off operations at Steiner vertices results in a hypergraph $H = (T, E_H)$ that is $2k$-hyperedge-connected; by Lemma 5.2 in the full version, the hypergraph H is k-weak-partition-connected; now Theorem 5.1 in the full version gives a rooted k-hyperarc-connected orientation of the resulting hypergraph. Such an orientation is extended to a Steiner rooted k-hyperarc-connected orientation of the *graph* $G = (V, E)$ using Lemma 5.1 in the full version. Essentially, the proof starts from the given graph, obtains a related hypergraph, orients that hypergraph, and extends that orientation of the hypergraph back into a desired orientation of the given graph.

Our proof technique for Theorems 8 and 10 also leads to an alternate proof of Menger's theorem in *undirected* graphs and hypergraphs (edge-disjoint version) – see Section 5 in the full version.

1.3 Proof Technique for Theorem 3

We prove a more general statement that implies Theorem 3. We begin with the definitions needed for the more general statement.

▶ **Definition 12.** *Let V be a finite set, $p : 2^V \to \mathbb{Z}$ be a set function, and $(H = (V, E), w : E \to \mathbb{Z}_+)$ be a hypergraph.*

1. *The set function p*
   - **a.** *is* symmetric *if $p(X) = p(V - X)$ for every $X \subseteq V$, and*
   - **b.** *is* skew-supermodular *if for every $X, Y \subseteq V$, at least one of the following inequalities hold:*
     - **i.** $p(X) + p(Y) \le p(X \cap Y) + p(X \cup Y)$.
     - **ii.** $p(X) + p(Y) \le p(X - Y) + p(Y - X)$.
2. *The coverage function $b_{(H,w)} : 2^V \to \mathbb{Z}_{\ge 0}$ is defined by $b_{(H,w)}(X) := \sum_{e \in B_H(X)} w(e)$ for every $X \subseteq V$, where $B_H(X) := \{e \in E : e \cap X \neq \emptyset\}$ for every $X \subseteq V$.*
3. *The hypergraph (H, w)* weakly covers *the function p if $b_{(H,w)}(X) \ge p(X)$ for every $X \subseteq V$.*
4. *The hypergraph (H, w)* strongly covers *the function p if $d_{(H,w)}(X) \ge p(X)$ for every $X \subseteq V$.*

If a hypergraph (H, w) strongly covers a function $p : 2^V \to \mathbb{Z}$, then it also weakly covers the function p. However, the converse is false – i.e., a weak cover is not necessarily a strong cover[2]. Bernáth and Király [5] showed that a weak cover of a *symmetric skew-supermodular* function can be converted to a strong cover of the same function by repeated *merging of disjoint hyperedges.* We recall their definition of the merging operation, discuss their result, and its significance now.

▶ **Definition 13.** *Let $(H = (V, E), w : E \to \mathbb{Z}_+)$ be a hypergraph. We use H_w to denote the unweighted multi-hypergraph over vertex set V containing $w(e)$ copies of every hyperedge $e \in E$. By* merging two disjoint hyperedges *of H_w, we refer to the operation of replacing them by their union in H_w. We will say that a hypergraph $(G = (V, E_G), c : E_G \to \mathbb{Z}_+)$ is obtained from (H, w) by* merging hyperedges *if the multi-hypergraph G_c is obtained from the multi-hypergraph H_w by repeatedly merging two disjoint hyperedges in the current hypergraph.*

Bernáth and Király showed the following result:

▶ **Theorem 14** (Bernáth and Király [5]). *Let $(H = (V, E), w : E \to \mathbb{Z}_+)$ be a hypergraph and $p : 2^V \to \mathbb{Z}$ be a symmetric skew-supermodular function such that $b_{(H,w)}(X) \ge p(X)$ for every $X \subseteq V$. Then, there exists a hypergraph $\left(H^* = (V, E^*), w^* : E^* \to \mathbb{Z}_+\right)$ such that*

(1) *$d_{(H^*,w^*)}(X) \ge p(X)$ for every $X \subseteq V$ and*

(2) *the hypergraph (H^*, w^*) is obtained by merging hyperedges of the hypergraph (H, w).*

[2] For example, consider the function $p : 2^V \to \mathbb{Z}$ defined by $p(X) := 1$ for every non-empty proper subset $X \subsetneq V$ and $p(\emptyset) := p(V) := 0$, and the hypergraph $(H = (V, E := \{\{u\} : u \in V\}), w : E \to \{1\})$.

We observe that Theorem 14 can be used to prove the existential version of Theorem 3: namely, for every hypergraph $(G = (V, E), w_G : E \to \mathbb{Z}_+)$ and a vertex $s \in V$, *there exists* a local connectivity preserving complete h-splitting-off at s from (G, w_G). This can be shown by setting up the hypergraph (H, w) and the function p suitably based on (G, w_G) and using Theorem 14 (see the first two paragraphs of the proof of Theorem 3.1 in Section 3 of the full version). We emphasize that this conclusion regarding hypergraph splitting-off from Bernáth and Király's result was not known before in the literature and is one of our contributions.

▶ Remark 15. We were also able to prove the existential version of Theorem 3 using *element-connectivity preserving reduction operations* (see [16] for the definition of element-connectivity and the notion of element-connectivity preserving reduction operations) – we omit the details of this alternate proof in the interests of brevity. The alternate proof does not seem to be helpful for the purposes of a strongly polynomial time algorithm. In fact, it remains open to design a strongly polynomial-time algorithm to perform *complete* element-connectivity preserving reduction operations in the weighted setting [16].

We recall that existence of a local connectivity preserving complete h-splitting-off at a vertex from a hypergraph does not immediately imply a polynomial-time algorithm – see the example in Remark 4. However, the above-mentioned proof of existence of a local-connectivity preserving complete splitting-off at an arbitrary vertex from a hypergraph (i.e., existential version of Theorem 3) via Theorem 14 suggests a natural approach towards designing a strongly polynomial time algorithm to find a local-connectivity preserving complete splitting-off at a given vertex from a given hypergraph: it suffices to prove a constructive version of Theorem 14 via a strongly polynomial-time algorithm. Towards this end, the example in Remark 4 suggests a necessary structural step towards a strongly polynomial-time algorithmic version of Theorem 14: we need to show Theorem 14 with the extra conclusion that the number of additional hyperedges in H^* is polynomial in the number of hyperedges and vertices in H.

Bernáth and Király proved Theorem 14 in the context of a reduction between certain hypergraph connectivity augmentation problems. For that reduction, the existential version of Theorem 14 is sufficient. However, for the purposes of our application to hypergraph splitting-off, we need an algorithmic version of Theorem 14. Bernáth and Király's proof of Theorem 14 is in fact algorithmic, but the run-time of the associated algorithm is not necessarily polynomial. Their proof implies that the number of additional hyperedges in the hypergraph (H^*, c^*) returned by their algorithm is at most $\sum_{e\in E} w(e)$ (i.e., $|E^*| - |E| \le \sum_{e\in E} w(e)$) and the run-time of the algorithm is $O(\sum_{e\in E}(|e| + w(e)))$. In particular, their run-time is polynomial only if the input weights are given in unary. Moreover, the exponential-sized hypergraph from Remark 4 could indeed arise as a consequence of their algorithm.

We address both the structural and the algorithmic issues mentioned above by proving a stronger algorithmic version of Theorem 14. In order to phrase an algorithmic version of Theorem 14, we need suitable access to the function p. Bernáth and Király [5] suggested access to a certain function maximization oracle associated with the function p that we describe below.

▶ **Definition 16.** *Let $p : 2^V \to \mathbb{Z}$ be a set function. p-*`max-sc-Oracle`* $\left((G_0, c_0), S_0, T_0\right)$ takes as input a hypergraph $(G_0 = (V, E_0), c_0 : E_0 \to \mathbb{Z}_+)$ and disjoint sets $S_0, T_0 \subseteq V$, and returns a tuple $(Z, p(Z))$, where Z is an optimum solution to the following problem:*

$$\max\left\{p(Z) - d_{(G_0,c_0)}(Z) : S_0 \subseteq Z \subseteq V - T_0\right\}. \qquad (p\text{-}\texttt{max-sc-Oracle})$$

For the purposes of our application (namely local connectivity preserving complete h-splitting-off at a vertex from a hypergraph), the above-mentioned function maximization oracle can be implemented to run in strongly polynomial time (see Lemma 3.1 in the full version). Using the above mentioned oracle, we prove the following algorithmic version of Theorem 14.

▶ **Theorem 17.** *Let $(H = (V, E), w : E \to \mathbb{Z}_+)$ be a hypergraph and $p : 2^V \to \mathbb{Z}$ be a symmetric skew-supermodular function such that $b_{(H,w)}(X) \geq p(X)$ for every $X \subseteq V$. Then, there exists a hypergraph $\left(H^* = (V, E^*), w^* : E^* \to \mathbb{Z}_+\right)$ such that*
(1) *$d_{(H^*,w^*)}(X) \geq p(X)$ for every $X \subseteq V$,*
(2) *the hypergraph (H^*, w^*) is obtained by merging hyperedges of the hypergraph (H, w), and*
(3) *$|E^*| - |E| = O(|V|)$.*
Furthermore, given a hypergraph $(H = (V, E), w : E \to \mathbb{Z}_+)$ and access to p-`max-sc-Oracle` of a symmetric skew-supermodular function $p : 2^V \to \mathbb{Z}$ where $b_{(H,w)}(X) \geq p(X)$ for every $X \subseteq V$, there exists an algorithm that runs in $O(|V|^3(|V| + |E|)^2)$ time using $O(|V|^3(|V| + |E|))$ queries to p-`max-sc-Oracle` and returns a hypergraph $\left(H^* = (V, E^*), w^* : E^* \to \mathbb{Z}_+\right)$ satisfying the above three properties. The run-time includes the time to construct the hypergraphs used as input to the queries to p-`max-sc-Oracle`. Moreover, for each query to p-`max-sc-Oracle`, the hypergraph (G_0, c_0) used as input to the query has $O(|V|)$ vertices and $O(|V| + |E|)$ hyperedges.

Theorem 17 is a strengthening of Theorem 14 in two ways. Firstly, our theorem shows the existence of a hypergraph that not only satisfies properties (1) and (2), but also satisfies property (3) – i.e., the number of additional hyperedges in the returned hypergraph is *linear* in the size of the vertex set. Secondly, our Theorem 17 shows the existence of a strongly polynomial-time algorithm that returns a hypergraph satisfying the three properties. Our main contribution is modifying Bernáth and Király's algorithm and analyzing the modified algorithm to bound the number of additional hyperedges and the run-time. We mention that property (3) cannot be tightened to guarantee that $|E^* - E| = O(|V|)$ – we were able to construct an example where $|E^* - E| = \Omega(|V|^2)$ (see Appendix A of the full version).

Theorem 17 immediately leads to a proof of Theorem 3 (see Theorem 3.1 and its proof in Section 3 of the full version). Instead of using Theorem 17 as a black-box, if we delve into the proof of it in the context of the proof of Theorem 3, we obtain the following theorem:

▶ **Theorem 18.** *Let $(G = (V, E), w : E \to \mathbb{Z}_+)$ be a hypergraph and $s \in V$. Then, there exists a hypergraph $(G' = (V, E'), w' : E' \to \mathbb{Z}_+)$ obtained by applying a h-splitting-off operation at s from (G, w) such that $\lambda_{(G',w')}(u, v) = \lambda_{(G,w)}(u, v)$ for every distinct $u, v \in V \setminus \{s\}$.*

We omit the proof of Theorem 18 in the interests of brevity. Theorem 18 closely resembles the *existential edge splitting-off* results of Lovász [45, 47] and Mader [48] for graphs. Lovász's and Mader's existential edge splitting-off results for graphs are important since they have been used to simplify the proofs of fundamental results in graph theory – e.g., Nash-Williams' Strong Orientation Theorem. On the other hand, Theorem 18 does not immediately imply a strongly polynomial-time algorithm for finding a local connectivity preserving complete h-splitting off at a vertex from a given weighted hypergraph. So, Theorem 3 may be useful in algorithmic contexts while Theorem 18 may be useful in graph-theoretical contexts.

1.4 Proof Technique for Theorem 17

In this section, we describe our proof technique for the existential result in Theorem 17. The algorithmic results in that theorem follow from the existential result using standard algorithmic tools for submodular functions, so we focus only on outlining a proof of the existential result. Let $(H = (V, E), w : E \to \mathbb{Z}_+)$ be a hypergraph and $p : 2^V \to \mathbb{Z}_+$ be a symmetric skew-supermodular function such that (H, w) weakly covers the function p. Our goal is to show that there exists a hypergraph $(H^* = (V, E^*), w^* : E \to \mathbb{Z}_+)$ such that

(1) (H^*, w^*) strongly covers the function p,

(2) (H^*, w^*) is obtained by merging hyperedges of the hypergraph (H, w), and

(3) $|E^*| - |E| = O(|V|)$.

Preliminaries. We define a set $X \subseteq V$ to be (p, H, w)-tight if $b_{(H,w)}(X) = p(X)$. For a function p and hypergraph (H, w), let $T_{p,H,w}$ denote the family of (p, H, w)-tight sets and let $\mathcal{T}_{p,H,w}$ be the family of inclusionwise maximal sets in $T_{p,H,w}$. We will need the following two operations:

(i) For hyperedges $e, f \in E$ and a positive integer $\alpha \leq \min\{w(e), w(f)\}$, the operation Merge $((H, w), e, f, \alpha))$ returns the hypergraph obtained from (H, w) by decreasing the weight of hyperedges e and f by α and increasing the weight of the hyperedge $e \cup f$ by α. All hyperedges with zero weight are discarded.

(ii) For a hyperedge $e \in E$ and a positive integer $\alpha \leq w(e)$, the operation Reduce $((H, w), e, \alpha)$ returns the hypergraph obtained by decreasing the weight of the hyperedge e by α. All hyperedges with zero weight are discarded.

Algorithm of [5]. Our proof of the existential result builds on the techniques of Bernáth and Király [5] who proved the existence of a hypergraph $(H^* = (V, E^*), w^* : E^* \to \mathbb{Z}_+)$ satisfying properties (1) and (2), so we briefly recall their techniques. We present the algorithmic version of their proof since it will be useful for our purposes.

The proof in [5] is inductive, and consequently, the algorithm implicit in their proof is recursive. The algorithm takes as input a hypergraph $((H = (V, E), w : E \to \mathbb{Z}_+)$ and a symmetric skew-supermodular function $p : 2^V \to \mathbb{Z}$ such that the hypergraph (H, w) weakly covers the function p. If $w(E) = 0$, then the algorithm is in its base case and returns the empty hypergraph. Otherwise, $w(E) > 0$; the algorithm chooses an arbitrary hyperedge $e \in E$ and defines hypergraphs (H_0, w_0) and (H', w') and the function p' by considering two cases. First, suppose that the hyperedge e is not contained in any set of the family $\mathcal{T}_{p,H,w}$. In this case, the algorithm defines (H_0, w_0) to be the hypergraph on vertex set V consisting of a single hyperedge e with $w_0(e) = 1$, constructs the hypergraph $(H', w') := \text{Reduce}((H, w), e, 1)$, and defines the function $p' := p - d_{(H_0,w_0)}$. Second, suppose that the hyperedge e is contained in some set $X \in \mathcal{T}_{p,H,w}$. It can be shown that there exists a hyperedge $f \in E$ such that $f \subseteq V - X$. In this case, the algorithm defines (H_0, w_0) to be the empty hypergraph on vertex set V, constructs the hypergraph $(H', w') := \text{Merge}((H, w), e, f, 1)$, and defines the function $p' := p$. In both cases, the algorithm recurses on the inputs (H', w') and p' to obtain a hypergraph (H_0^*, w_0^*) and returns the hypergraph $(H^*, w^*) = (H_0^* + H_0, w_0^* + w_0)$. Here, the hyperedges of H^* are the union of the hyperedges of H_0^* and H_0 with the weight $w^*(e)$ of a hyperedge e being the sum of the weights $w_0^*(e) + w_0(e)$ if the hyperedge e is present in both H_0^* and H_0, being $w_0^*(e)$ if the hyperedge e is present only in H_0^*, and being $w_0(e)$ if the hyperedge e is present only in H_0.

We note that $w'(E') = w(E) - 1$. Furthermore, it can be shown that the function p' is symmetric skew-supermodular and the hypergraph (H', w') weakly covers the function p'. Consequently, by induction on $w(E)$, the algorithm can be shown to terminate in $w(E)$ recursive calls and returns a hypergraph satisfying properties (1) and (2). Moreover, the number of additional hyperedges in the returned hypergraph is at most the number of recursive calls where the Merge operation is performed, which is also at most $w(E)$. Thus, in order to reduce the number of additional hyperedges and to design a strongly polynomial-time algorithm, our goal is to reduce the recursion depth of the algorithm. We emphasize that the recursion depth of Bernáth and Király's algorithm could indeed be exponential (the exponential sized example mentioned in Remark 4 could arise in the execution of their algorithm), so we need to necessarily modify their algorithm.

Preprocessing for Additional Structure. Similar to Bernáth and Király's algorithm, our algorithm also takes as input a hypergraph $(H = (V, E), w : E \to \mathbb{Z}_+)$ and a symmetric skew-supermodular function $p : 2^V \to \mathbb{Z}$ such that the hypergraph (H, w) weakly covers the function p. However, unlike Bernáth and Király's algorithm, our algorithm performs a preprocessing step so that the inputs (H, w) and p satisfy the following two additional conditions:

(a) every hyperedge $e \in E$ is contained in some set of $\mathcal{T}_{p,H,w}$ and

(b) the degree of every vertex in (H, w) is non-zero.

As a consequence of these additional conditions, the family $\mathcal{T}_{p,H,w}$ will be a *disjoint family*, a property that we leverage heavily throughout our analysis. Furthermore, we modify Bernáth and Király's algorithm to ensure that these conditions hold during every recursive call.

Our Algorithm. We now describe our modification of the above-mentioned Bernáth and Király's algorithm to reduce the recursion depth. Our algorithm is also recursive and its base case is the same as that of Bernáth and Király's algorithm (i.e., $w(E) = 0$). During recursive cases (i.e., if $w(E) > 0$), instead of performing one of the two (i.e., Merge or Reduce) operations, our algorithm performs both operations in a sequential fashion. In particular, we find a pair of disjoint hyperedges $e, f \in E$ contained in distinct sets of $\mathcal{T}_{p,H,w}$ (such a pair exists by condition (a) and the arguments of Bernáth and Király mentioned above). Next, instead of performing one Merge operation (as was done by Bernáth and Király's algorithm), we perform as many Merge operations as possible using the hyperedges e and f. Formally, let

$$\alpha^M := \max\left\{\alpha \in \mathbb{Z}_+ : \text{hypergraph returned by } \textsc{Merge}((H, w), e, f, \alpha) \text{ weakly covers } p\right\}.$$

We let $(H^M, w^M) := \textsc{Merge}((H, w), e, f, \alpha^M)$ and $p^M := p$. Next, instead of recursing on $((H^M, w^M), p^M)$ (as was done by Bernáth and Király's algorithm), we perform as many Reduce operations as possible on the newly created hyperedge $e \cup f$. Formally, let

$$\alpha^R := \max\left\{\alpha \in \mathbb{Z}_{\geq 0} : \begin{array}{c}\text{Hypergraph returned by } \textsc{Reduce}((H^M, w^M), e \cup f, \alpha) \\ \text{weakly covers the function } (p^M - d_{(H_0^\alpha, w_0^\alpha)})\end{array}\right\}$$

where (H_0^α, w_0^α) denotes the hypergraph on vertex set V consisting of a single hyperedge $e \cup f$ with $w_0^\alpha(e \cup f) = \alpha$. We construct the hypergraph $(H_0, w_0) := (H_0^{\alpha^R}, w_0^{\alpha^R})$, the hypergraph $(H^R, w^R) := \textsc{Reduce}((H^M, w^M), e \cup f, \alpha^R)$ and define the function $p^R := p^M - d_{(H_0^{\alpha^R}, w_0^{\alpha^R})}$. This immediate reduce step ensures that the hypergraph (H^R, w^R) and the function p^R satisfy condition (a) – i.e., every hyperedge in (H^R, w^R) is contained in some set of $\mathcal{T}_{p^R,H^R,w^R}$. Finally, we compute sets $\mathcal{Z} := \{u \in V : b_{(H^R,w^R)}(u) = 0\}$ and $V' := V - \mathcal{Z}$, hypergraph $(H' := (V', E' := E^R), w' := w^R)$, and define the function $p' : 2^{V'} \to \mathbb{Z}$ by $p'(X) := \max\{p(X \cup R) : R \subseteq \mathcal{Z}\}$ for every $X \subseteq V'$ – this final step can be viewed as a clean up step since it gets rid of vertices that are not incident to any hyperedges (and revises the function p appropriately). This clean up step ensures that the hypergraph (H', w') satisfies condition (b) – i.e., the degree of every vertex in (H', w') is non-zero. It can be shown that the function p' is symmetric skew-supermodular and the hypergraph (H', w') weakly covers the function p'. Furthermore, the function p' and hypergraph (H', w') satisfy conditions (a) and (b). We recursively call the algorithm on input $((H', w'), p')$ to obtain a hypergraph (H_0^*, w_0^*). We obtain the hypergraph (G, c) from (H_0^*, w_0^*) by adding the vertices $\mathcal{Z}$ and return the hypergraph $(G + H_0, c + w_0)$. By induction on the total weight of hyperedges in the input hypergraph, it can be shown that our algorithm returns a hypergraph satisfying properties (1) and (2) and also terminates within a finite number of recursive calls.

Recursion Depth and Potential Functions. We now sketch our proof to show that the recursion depth of our algorithm is $|E| + O(|V|)$. We note that this also bounds the number of additional hyperedges in the hypergraph returned by the algorithm, and consequently this hypergraph also satisfies property (3). Let ℓ be the number of recursive calls made by the algorithm on the input instance $((H, w), p)$. We partition the set $[\ell]$ of recursive calls into two parts: let $P_1 \subseteq [\ell]$ be the set of recursive calls during which the merged hyperedge $e \cup f$ survives in the hypergraph (H', w') that is input to the subsequent recursive call and $P_2 \subseteq [\ell]$ be the recursive calls during which the merged hyperedge $e \cup f$ does not survive in the hypergraph (H', w') that is input to the subsequent recursive call. We note that $[\ell] = P_1 \uplus P_2$. We bound $|P_1|$ and $|P_2|$ separately using certain carefully designed potential functions.

First, we show that $|P_1| = O(|V|)$ as follows: for $i \in [\ell]$, consider the maximal tight set family $\mathcal{T}_i = \mathcal{T}_{p_i, H_i, w_i}$ where $((H_i, w_i), p_i)$ is the input to the i^{th} recursive call. Also, let $\mathcal{T}_{\leq 1} := \mathcal{T}_1$ and $\mathcal{T}_{\leq i} := \mathcal{T}_i \cup \{X \cap V_i : X \in \mathcal{T}_{\leq i-1}\}$ for integers i where $2 \leq i \leq \ell$ and V_i is the ground set of the input to the i^{th} recursive call. Thus, $\mathcal{T}_{\leq i}$ is the *projection* of all the maximal tight sets encountered in the first i recursive calls of the algorithm onto the ground set of the inputs at the i^{th} recursive call. We show that $\mathcal{T}_{\leq i}$ is laminar for every $i \in [\ell]$ (Lemma 6.11 in the full version). However, $|\mathcal{T}_{\leq i}|$ is not necessarily non-decreasing with i since projection of a set family to a subset could result in the loss of sets from the family. Consequently, $|\mathcal{T}_{\leq i}|$ is not suitable as a potential function to measure progress. Instead, we use the potential function $\phi(i) := |\mathcal{T}_{\leq i}| + 3|\mathcal{Z}_{\leq i-1}|$, where $\mathcal{Z}_{\leq i}$ is the union of the sets $\mathcal{Z}$ computed up to the i^{th} recursive call. We show that $\phi(i)$ is non-decreasing and strictly increases if $i \in P_1$ (Claim 6.4 in Lemma 6.12 of the full version). Consequently, $|P_1| = O(|V|)$.

Secondly, we bound $|P_2|$ as follows. We use a lookahead-potential function: let $\Phi_1(i)$ be the number of recursive calls between i and ℓ during which the merged hyperedge $e \cup f$ survives in the hypergraph (H', w') that is input to the subsequent recursive call and let $\Phi(i) := |E_i| + \Phi_1(i)$, where E_i is the set of hyperedges in the hypergraph (H_i, w_i) input to the i^{th} recursive call. We show that $\Phi(i)$ is non-increasing and strictly decreases if $i \in P_2$ (Claim 6.5 in Lemma 6.12 of the full version). Hence, $|P_2| \leq \Phi(1) - \Phi(\ell) \leq \Phi(1) \leq |E_1| + |P_1| = |E| + O(|V|)$), where the last equality is because of the bound on $|P_1|$ from the previous paragraph.

▶ Remark 19. Our key technical contributions are twofold. Our first key technical contribution is identifying conditions (a) and (b) under which $\mathcal{T}_{p,H,w}$ becomes a disjoint family. We ensure that conditions (a) and (b) hold in every recursive call by performing *immediate* reduction and clean-up steps in the algorithm. Our second key technical contribution is identifying appropriate potential functions to measure progress of the algorithm. The disjointness of $\mathcal{T}_{p,H,w}$ was crucial for identifying the laminar structure of the family of projected maximal tight sets across recursive calls, which was subsequently helpful in bounding the number of recursive calls corresponding to P_1. Moreover, we bound the number of recursive calls corresponding to P_2 using a lookahead-potential function that relates $|P_2|$ to $|P_1|$. As discussed above, the additive $|E|$ in the recursion depth comes from the bound on $|P_2|$.

2 Conclusion

We introduced a splitting-off operation in (weighted) hypergraphs. Our contribution on this front is conceptualizing an appropriate notion of splitting-off in hypergraphs. Next, we proved that for every hypergraph there exists a local-connectivity preserving complete h-splitting-off at an arbitrary vertex from the hypergraph. Although our proof of existence follows from previously known existential results of Bernáth and Király [5] for an abstract function cover problem, our main contribution is identifying that our newly introduced notion of hypergraph splitting-off falls within their framework. Next, we designed a strongly polynomial-time algorithm to find a local-connectivity preserving complete splitting-off at a vertex. This involved substantial technical challenges to overcome. In particular, our main contribution towards the strongly polynomial-time algorithm is a strengthening of the above-mentioned existential result: we showed that there exists a local connectivity preserving complete h-splitting-off at an arbitrary vertex from the hypergraph *in which the number of additional hyperedges is polynomial in the number of vertices*. The strongly polynomial-time algorithm follows from our techniques to achieve this stronger existential result via standard algorithmic tools in submodularity. Finally, we illustrated the usefulness of the existence of local connectivity preserving complete h-splitting-off at an arbitrary vertex from a hypergraph by presenting two applications. Our first application is to give a constructive characterization of k-hyperedge-connected hypergraphs. Our second application is to give an alternative proof of an approximate min-max relation for the max Steiner rooted-connected orientation problem in graphs and hypergraphs. Two notable features of our proof of this relation for *graphs* are that (1) it avoids the strong orientation theorem of Nash-Williams and (2) it proves a result for graphs using tools developed for hypergraphs.

Local and global connectivity preserving complete splitting-off at a vertex from a graph is a powerful operation for graphs. It finds applications in a variety of graph problems including graph orientation [25,26,38,48], connectivity augmentation [2,22,23,24,49], minimum cuts enumeration [29,32,50], network design [12,17,35,46], tree packing [7,33,40,41], congruency-constrained cuts [51], and approximation algorithms for TSP [8,30]. We believe that our local connectivity preserving complete splitting-off results for hypergraphs is likely to find future applications akin to its counterpart in graphs. We mention some of the open questions raised by our work:

1. Our work focused on local connectivity preserving complete *h-splitting-off* at a vertex from a hypergraph. We gave an example showing that local/global connectivity preserving complete *g-splitting-off* at a vertex from a hypergraph may not exist (Figure 2). Are there sufficient conditions to guarantee local/global connectivity preserving complete *g-splitting-off* at a vertex from a hypergraph? We recall that Lovász's [45,47] and Mader's [48] results give sufficient conditions to guarantee local and global connectivity preserving complete g-splitting-off at a vertex from a *graph*.
2. As one of the applications of our splitting-off result, we presented an alternative proof of an approximate min-max relation for the max Steiner *rooted*-connected orientation problem in hypergraphs. The computational complexity of a closely related hypergraph orientation problem is open: In max Steiner connected orientation problem in hypergraphs, the input is a hypergraph $G = (V, E)$ and a subset T of terminals. The goal is to find the maximum k and an orientation $\overrightarrow{G}$ of G such that $\overrightarrow{G}$ contains k hyperarc-disjoint paths from u to v for every pair of distinct terminals $u, v \in T$. Max Steiner connected orientation problem in *graphs* is solvable in polynomial time via the Nash-Williams' strong orientation theorem. Is max Steiner connected orientation problem in *hypergraphs* solvable in polynomial time?

References

1 O. Alrabiah, V. Guruswami, and R. Li. Randomly punctured reed–solomon codes achieve list-decoding capacity over linear-sized fields. In *(To appear) Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, STOC, 2024.

2 J. Bang-Jensen, A. Frank, and B. Jackson. Preserving and increasing local edge-connectivity in mixed graphs. *SIAM Journal on Discrete Mathematics*, 8(2):155–178, 1995.

3 J. Bang-Jenson and B. Jackson. Augmenting hypergraphs by edges of size two. *Mathematical Programming*, 84:467–481, 1999.

4 N. Bansal, O. Svensson, and L. Trevisan. New notions and constructions of sparsification for graphs and hypergraphs. In *IEEE 60th Annual Symposium on Foundations of Computer Science*, FOCS, pages 910–928, 2019.

5 A. Bernáth and T. Király. Covering skew-supermodular functions by hypergraphs of minimum total size. *Operations Research Letters*, 37(5):345–350, 2009.

6 A. Bernáth and T. Király. A unifying approach to splitting-off. *Combinatorica*, 32:373–401, 2012.

7 A. Bhalgat, R. Hariharan, T. Kavitha, and D. Panigrahi. Fast Edge Splitting and Edmonds' Arborescence Construction for Unweighted Graphs. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA, pages 455–464, 2008.

8 Jannis Blauth and Martin Nägele. An improved approximation guarantee for prize-collecting tsp. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023, pages 1848–1861, New York, NY, USA, 2023. Association for Computing Machinery.

9 R. Cen, W. He, J. Li, and D. Panigrahi. Steiner connectivity augmentation and splitting-off in poly-logarithmic maximum flows. In *Proceedings of the 34th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA, pages 2449–2488, 2023.

10 R. Cen, J. Li, and D. Panigrahi. Augmenting edge connectivity via isolating cuts. In *Proceedings of the 33rd Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA, pages 3237–3252, 2022.

11 R. Cen, J. Li, and D. Panigrahi. Edge connectivity augmentation in near-linear time. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC, pages 137–150, 2022.

12 Y. H. Chan, W. S. Fung, L. C. Lau, and C. K. Yung. Degree Bounded Network Design with Metric Costs. *SIAM Journal on Computing*, 40(4):953–980, 2011. Prelim. version in FOCS 2008.

13 K. Chandrasekaran and C. Chekuri. Hypergraph k-cut for fixed k in deterministic polynomial time. *Mathematics of Operations Research*, 47(4), 2022. Prelim. version in FOCS 2020.

14 K. Chandrasekaran and C. Chekuri. Min-max partitioning of hypergraphs and symmetric submodular functions. *Combinatorica*, 43:455–477, 2023. Prelim. version in SODA 2021.

15 K. Chandrasekaran, C. Xu, and X. Yu. Hypergraph k-Cut in randomized polynomial time. *Mathematical Programming*, 186:85–113, 2021. Prelim. version in SODA 2018.

16 C. Chekuri and N. Korula. A Graph Reduction Step Preserving Element-Connectivity and Packing Steiner Trees and Forests. *SIAM Journal on Discrete Mathematics*, 28(2):577–597, 2014. Prelim. version in ICALP 2009.

17 C. Chekuri and B. Shepherd. Approximate Integer Decompositions for Undirected Network Design Problems. *SIAM J. Discrete Math.*, 23:163–177, 2008.

18 C. Chekuri and C. Xu. Minimum cuts and sparsification in hypergraphs. *SIAM Journal on Computing*, 47(6):2118–2156, 2018. Prelim. version in SODA 2016.

19 Y. Chen, S. Khanna, and A. Nagda. Near-linear size hypergraph cut sparsifiers. In *IEEE 61st Annual Symposium on Foundations of Computer Science*, FOCS, pages 61–72, 2020.

20 B. Cosh. *Vertex splitting and connectivity augmentation in hypergraphs.* PhD thesis, University of London, 2000.

21 K. Fox, D. Panigrahi, and F. Zhang. Minimum cut and minimum k-cut in hypergraphs via branching contractions. *ACM Trans. Algorithms*, 19(2), 2023. Prelim. version in SODA 2019.

22 A. Frank. Augmenting graphs to meet edge-connectivity requirements. *SIAM Journal on Discrete Mathematics*, 5(1):25–53, 1992. Prelim. version in FOCS 1990.

23 A Frank. Connectivity augmentation problems in network design. *Mathematical Programming: State of the Art 1994*, pages 34–63, 1994.

24 A. Frank. *Connections in Combinatorial Optimization.* Oxford University Press, Oxford, 2011.

25 A. Frank and Z. Király. Graph orientations with edge-connection and parity constraints. *Combinatorica*, 22:47–70, 2002.

26 András Frank. Applications of submodular functions. *Surveys in Combinatorics, 1993 (Keele)*, pages 85–136, 1993.

27 H. N. Gabow. Efficient splitting off algorithms for graphs. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, STOC, pages 696–705, 1994.

28 M. Ghaffari, D. Karger, and D. Panigrahi. Random Contractions and Sampling for Hypergraph and Hedge Connectivity. In *Proceedings of the 28th annual ACM-SIAM Symposium on Discrete Algorithms*, SODA, pages 1101–1114, 2017.

29 M. X. Goemans. Approximate Edge Splitting. *SIAM Journal on Discrete Mathematics*, 14(1):138–141, 2001.

30 M.X. Goemans and D.J. Bertsimas. Survivable networks, linear programming relaxations and the parsimonious property. *Mathematical Programming*, 60:145–166, 1993.

31 Z. Guo, R. Li, C. Shangguan, I. Tamo, and M. Wootters. Improved List-Decodability and List-Recoverability of Reed-Solomon Codes via Tree Packings. In *IEEE 62nd Annual Symposium on Foundations of Computer Science*, FOCS, pages 708–719, 2022.

32 M. Henzinger and D. Williamson. On the number of small cuts in a graph. *Information Processing Letters*, 59:41–44, 1996.

33 K. Jain, M. Mahdian, and M. R. Salavatipour. Packing Steiner Trees. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA, pages 266–274, 2003.

34 A. Jambulapati, Y. P. Liu, and A. Sidford. Chaining, group leverage score overestimates, and fast spectral hypergraph sparsification. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC, pages 196–206, 2023.

35 T. Jordán. On minimally k-edge-connected graphs and shortest k-edge-connected Steiner networks. *Discrete Applied Mathematics*, 131(2):421–432, 2003.

36 M. Kapralov, R. Krauthgamer, J. Tardos, and Y. Yoshida. Towards tight bounds for spectral sparsification of hypergraphs. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC, pages 598–611, 2021.

37 M. Kapralov, R. Krauthgamer, J. Tardos, and Y. Yoshida. Spectral Hypergraph Sparsifiers of Nearly Linear Size. In *IEEE 62nd Annual Symposium on Foundations of Computer Science*, FOCS, pages 1159–1170, 2022.

38 T. Király and L. C. Lau. Approximate min–max theorems for Steiner rooted-orientations of graphs and hypergraphs. *Journal of Combinatorial Theory, Series B*, 98:1233–1252, November 2008. Prelim. version in FOCS 2006.

39 D. Kogan and R. Krauthgamer. Sketching cuts in graphs and hypergraphs. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, ITCS, pages 367–376, 2015.

40 M. Kriesell. Edge-disjoint trees containing some given vertices in a graph. *J. Comb. Theory Ser. B*, 88:53–63, 2003.

41 L. C. Lau. An Approximate Max-Steiner-Tree-Packing Min-Steiner-Cut Theorem. *Combinatorica*, 27:71–90, 2007. Prelim. version in FOCS 2004.

42 L. C. Lau and C. K. Yung. Efficient Edge Splitting-Off Algorithms Maintaining All-Pairs Edge-Connectivities. *SIAM Journal on Computing*, 42(3):1185–1200, 2013. Prelim. version in IPCO 2010.

43 J. R. Lee. Spectral hypergraph sparsification via chaining. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC, pages 207–218, 2023.

44 P. Li and O. Milenkovic. Inhomogeneous hypergraph clustering with applications. *Advances in neural information processing systems*, 30, 2017.

45 L. Lovász. Lecture. *Presented in a Conference on Graph Theory, Prague*, 1974.

46 L. Lovász. On some connectivity properties of Eulerian graphs. *Acta Math. Hungarica*, 28:129–138, 1976.

47 L. Lovász. *Combinatorial Problems and Exercises, Second Edition.* American Mathematical Soc., 1993. First Edition: North-Holland, Amsterdam, 1979.

48 W. Mader. A reduction method for edge-connectivity in graphs. In *Annals of Discrete Mathematics*, volume 3, pages 145–164. Elsevier, 1978.

49 H. Nagamochi and T. Ibaraki. *Algorithmic Aspects of Graph Connectivity.* Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2008.

50 H. Nagamochi, K. Nishimura, and T. Ibaraki. Computing all small cuts in an undirected network. *SIAM Journal on Discrete Mathematics*, 10(3):469–481, 1997.

51 M. Nägele and R. Zenklusen. A new contraction technique with applications to congruency-constrained cuts. *Mathematical Programming*, 183(2):455–481, 2020. Prelim. version in IPCO 2019.

52 C. St. J. A. Nash-Williams. On orientations, connectivity and odd vertex pairings in finite graphs. *Canad. J. Math*, 12:555–567, 1960.

53 C. St. J. A. Nash-Williams. Edge disjoint spanning trees of finite graphs. *J. London Math. Soc.*, 36:445–450, 1961.

54 S. Ornes. How big data carried graph theory into new dimensions. *Quanta Magazine*, 2021. URL: `https://www.quantamagazine.org/how-big-data-carried-graph-theory-into-new-dimensions-20210819/`.

55 K. Quanrud. Quotient sparsification for submodular functions. Manuscript available at `kentquanrud.com`, November 2022.

56 H. E. Robbins. A Theorem on Graphs with an Application to a Problem of Traffic Control. *Amer. Math. Monthly*, 46:281–283, 1939.

57 S. Schlag, T. Heuer, L. Gottesbüren, Y. Akhremtsev, C. Schulz, and P. Sanders. High-quality hypergraph partitioning. *ACM Journal of Experimental Algorithmics*, 27:1–39, 2023.

58 T. Soma and Y. Yoshida. Spectral sparsification of hypergraphs. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA, pages 2570–2581, 2019.

59 W. T. Tutte. On the problem of decomposing a graph into n connected factors. *J. London Math. Soc.*, 36:221–230, 1961.

60 N. Veldt, A. R. Benson, and J. Kleinberg. Hypergraph cuts with general splitting functions. *SIAM Review*, 64(3):650–685, 2022.