Mathematical Programming (2024) 207:717–732
https://doi.org/10.1007/s10107-023-02029-0

FULL LENGTH PAPER

Series A

A polynomial time algorithm for finding a minimum 4-partition of a submodular function

Tsuyoshi Hirayama[1] · Yuhao Liu[2] · Kazuhisa Makino[3] · Ke Shi[2] · Chao Xu[2]

Received: 21 April 2023 / Accepted: 17 October 2023 / Published online: 6 December 2023

Abstract

In this paper, we study the minimum k-partition problem of submodular functions, i.e., given a finite set V and a submodular function $f : 2^V \to \mathbb{R}$, computing a k-partition $\{V_1, \ldots, V_k\}$ of V with minimum $\sum_{i=1}^{k} f(V_i)$. The problem is a natural generalization of the minimum k-cut problem in graphs and hypergraphs. It is known that the problem is NP-hard for general k, and solvable in polynomial time for fixed $k \leq 3$. In this paper, we construct the first polynomial-time algorithm for the minimum 4-partition problem.

Keywords Submodular function · Polynomial time · Combinatorial optimization

Mathematics Subject Classification 90C27

Authors are ordered alphabetically. This work was partially supported by JST ERATO Grant Number JPMJER2301 and JSPS KAKENHI Grant Numbers JP19K22841, JP20H00609, and JP20H05967. An earlier version of this paper appeared in SODA 2023 [17]. This version introduces have a more general algorithm for $(1, \ell)$-size 3-partition, and a graph example at the end.

✉ Chao Xu
the.chao.xu@gmail.com

Tsuyoshi Hirayama
tsuyoshi1.hirayama@toshiba.co.jp

Yuhao Liu
yuhaoliu126@gmail.com

Kazuhisa Makino
makino@kurims.kyoto-u.ac.jp

Ke Shi
self.ke.shi@gmail.com

1 Toshiba Digital Solutions Corporation, Kawasaki, Japan

2 University of Electronic Science and Technology of China, Chengdu, China

3 Kyoto University, Kyoto, Japan

1 Introduction

Let V be a finite set with $n = |V|$, and let $f : 2^V \to \mathbb{R}$ be a *submodular* function, i.e., $f(X) + f(Y) \geq f(X \cap Y) + f(X \cup Y)$ holds for any $X, Y \subseteq V$. For an integer $k \geq 2$, the minimum k-partition problem for a submodular function f is to compute a k-partition $\mathscr{P}_k = \{V_1, \ldots, V_k\}$ of V with the minimum value defined as $\sum_{i=1}^{k} f(V_i)$, where, for a positive integer k, $\{V_1, \ldots, V_k\}$ is called a *k-partition* if $V_i \neq \emptyset$ for all i, $\bigcup_{i=1}^{k} V_i = V$, and $V_i \cap V_j = \emptyset$ for all i and j with $i \neq j$. This is one of the most fundamental problems in combinatorial optimization, and a natural generalization of the minimum k-cut problem in graphs and hypergraphs, where both problems are polynomial time solvable for fixed k.

The minimum k-cut problem has many applications such as the traveling salesperson problem, VLSI design, evolutionary tree construction and network reliability [12, 22]. Goldschmidt-Hochbaum [13] showed that the minimum k-cut problem in graphs is NP-hard, when k is a part of input, but it can be solved in polynomial time for fixed k.

After their work, a number of algorithms for the minimum k-cut problem in graphs were proposed; See [8, 16, 18, 22, 24, 25, 29, 31], for example. The current best deterministic algorithm has $\tilde{O}(mn^{k-1})$ time for $k \leq 6$ [23–25, 33], $\tilde{O}(mn^{2k-2})$ time for $k \geq 7$ [8], $k^{O(k)} n^{(2\omega/3+\epsilon)k+O(1)}$ for any $\epsilon > 0$ and polynomially bounded weights, where ω is the matrix multiplication constant [16], and a randomized algorithm in $\tilde{O}(n^k)$ time [15].

The minimum k-cut problem in hypergraphs is NP-hard, which immediately follows from the NP-hardness of the graph problem [13]. Klimmek and Wagner [19] and Mak and Wong [21] showed that the minimum 2-cut problem in hypergraphs can be solved in $\tilde{O}(dn)$ time, where d denotes the sum of the cardinalities of all hyperedges. Chekuri and Xu showed that in the same running time, they can count and enumerate all minimum 2-cuts [9]. For $k = 3$, Xiao [32] constructed an $\tilde{O}(d(m+n)n^3)$-time algorithm. Fukunaga [11] showed that the problem can be solved in polynomial time if both k and $\max_{e \in E} |e|$ are fixed. A randomized polynomial time algorithm was found by Chandrasekaran et al. [6]. Later works speed up the randomized algorithm [10], and generalize to multicriteria objective and size constraints [3]. Recently, Chandrasekaran and Chekuri finally settled the open problem and showed minimum k-cut problem in hypergraphs is polynomial time solvable for each fixed k [5]. There is some follow-up work on counting and enumerating all minimum k-cuts [1, 2].

The minimum k-partition problem for fixed k is much less understood. A submodular function is symmetric, if $f(X) = f(V \backslash X)$. For $k = 2$, the problem is equivalent to the symmetric submodular function minimization, since $f(X)$ can be replaced with $\frac{1}{2}(f(X) + f(V - X))$. Hence it can be solved in $O(n^3\gamma)$ time by Queyranne's algorithm [28], where γ denotes the time required for function evaluation. For $k = 3$, Okumoto et al. [27] presented an $O(n^3\tau(n))$-time algorithm by extending Xiao's algorithm for the minimum 3-cut problem for hypergraphs [32], where $\tau(n)$ denotes the time required to solve the submodular function minimization, and the current best known bound for $\tau(n)$ is $\tilde{O}(n^3\gamma + n^4)$ [20]. However, it is still *open* if there exists a polynomial time algorithm for fixed $k \geq 4$ [27]. It was implied in [4] that symmetric submodular k-partition reduces to n^{2k-2} calls of submodular $(k-1)$-partition. There

are several studies on approximation algorithms for the problem [7, 27, 34, 35]. The current best approximation ratio is 1.5 for $k = 4$ [27] and 2 for $k \geq 5$ [7].

A generalization of our problem, the w-size k-partition problem, first described in [14]. Let $w = (w_1, \ldots, w_j)$, a w-size k-partition is a k-partition $V_1, \ldots, V_k$, such that $w_i \leq |V_i|$ for each $i \leq j$, and $|V_i| \leq |V_{i+1}|$. Namely, the ith smallest part of the partition has at least w_i elements. The goal is to find a minimum w-size k-partition. Since this is a strictly more general problem than submodular k-partition problem, even fewer results are known.

Our results We show the following two results.

1. An $O(n^6\tau(n))$ time algorithm for finding a minimum 4-partition of a submodular function.
2. As a corollary, an $O(n^{14}\tau(n))$ time algorithm for finding minimum 5-partition of a *symmetric* submodular function.
3. The minimum $(1, \ell)$-size 3-partition can be found in $O(n^{4\ell-3}\tau(n))$ time.

This settles the complexity status of the submodular k-partition problem for $k = 4$ [11, 22, 24, 27, 32, 35]. Our algorithm is based on the *compatibility* of 3- and 4-partitions, which is a generalization of the noncrossing property of 2- and 3-partitions proposed in [14, 27, 32]. There exist two natural and possible extensions of their noncrossing property. However, both extensions fail to produce a polynomial time solution to the minimum 4-partition problem (see the detailed discussion in Sect. 3).

The rest of the paper is organized as follows. In Sect. 3, we present the compatibility of 3- and 4-partitions, where the proof can be found in Sect. 4, and describe our algorithm. In Sect. 5, we present how to compute a minimum $(1, \ell)$-size 3-partition for fixed ℓ. Section 6 analyzes the time complexity of our algorithm. Finally in Sect. 7, we conclude with some remarks.

2 Preliminaries

We write $\binom{V}{i}$ to be the family of all size i subsets of V. We abuse the notation to generalize each function f on sets to a function on partitions. That is, for any partition $\mathscr{P}$, we define $f(\mathscr{P}) = \sum_{X \in P} f(X)$. Let $g : 2^V \to \mathbb{R}$ be a set function. For a set $U \subseteq V$, let $g_{\setminus U}$ denote a set function obtained from g by deleting U from V, and $g_{/U}$ denote a set function obtained from g by shrinking U into a new element u (i.e., $u \notin V$). Namely, $g_{\setminus U} : 2^{V \setminus U} \to \mathbb{R}$ satisfies $g_{\setminus U}(S) = g(S)$, and $g_{/U} : 2^{(V \setminus U) \cup \{u\}} \to \mathbb{R}$ satisfies $g_{/U}(S) = g(S)$ if $S \not\ni u$, and $g((S \setminus \{u\}) \cup U)$, otherwise. We note that $g_{\setminus U}$ and $g_{/U}$ are both submodular if g is submodular.

A set X is *noncrossing* with a partition $\mathscr{Y}$, if $X \subset Y$ for some $Y \in \mathscr{Y}$. A partition $\mathscr{X}$ *is noncrossing with* a partition $\mathscr{Y}$, if there exists a component $X \in \mathscr{X}$ that is noncrossing with $\mathscr{Y}$. A partition is called *h-size* for an integer h if it is (h)-size, namely all its components contain at least h elements. A partition $\mathscr{X}$ is called *trivial* if all but one component have exactly 1 element. A non-trivial k-partition is equivalent to s-size partition, where $s = (1, \ldots, 1, 2)$ is a $k-1$ size tuple. For 2-partitions over a ground set with at least 4 elements, it is trivial if and only if it is not 2-size.

Fig. 1 Compute a minimum 3-partition

```
MIN3PARTITION(f):
  V ← domain(f)
  if |V| ≤ 6
      return the optimum by brute force
  for X ∈ (V 1)
      add candidate {X} ∪ MIN2PARTITION(f_\X)
  𝒳 ← MINNONTRIVIAL2PARTITION(f)
  for X ∈ 𝒳
      add candidate MIN3PARTITION(f_/X)
  return minimum over all candidates
```

For two partitions $\mathscr{X} = \{X_1, \ldots, X_k\}$ and $\mathscr{Y} = \{Y_1, \ldots, Y_m\}$, a matrix M is a *configuration* of $\mathscr{X}$ and $\mathscr{Y}$, if $M_{i,j} = |X_i \cap Y_j|$. We will abuse the notation and write M_{ij} when it will not lead to confusion. We say configuration M is *generated* by $\mathscr{X}$ and $\mathscr{Y}$.

Two configurations M_1 and M_2 are equivalent if there exist a row permutation and a column permutation to swap M_1 into M_2. Configurations help us visualize the ways partitions intersect.

To represent a set of possible configurations M visually, we write a few numbers in the matrix to indicate what pattern the configuration matches. We use the number i to denote the values that are known to be i, i^+ if the value known to be at least i, i^- if the value known to be at most i, and ? means either we don't know or we don't care about its value.

3 Compatibility for 3- and 4-partitions

In this section, we present the compatibility of minimum 3- and 4-partitions in submodular functions, from which we derive a polynomial time algorithm for the minimum 4-partition problem. We start with the noncrossing property for 2- and 3-partitions, which was proven in [27].

The following lemma is a direct consequence of Corollary 1 from [27].

Lemma 3.1 *Let $f : 2^V \to \mathbb{R}$ be a submodular function with $n \geq 7$, and let $\mathscr{X}$ denote a minimum non-trivial 2-partition of f. If f has a minimum 3-partition of 2-size, then it contains a minimum 3-partition with which $\mathscr{X}$ is noncrossing.*

By this lemma, we can easily construct the following divide-and-conquer algorithm for the minimum 3-partition problem [27], see Fig. 1.

We first compute some candidate partitions, where one of them is a minimum non-2-size 3-partition of f (i.e., a minimum partition of type $\{\{v\}, W_1, W_2\}$ for some $v \in V$). To do this, we compute a minimum partition $\{W_1, W_2\}$ of $f_{\setminus\{v\}}$ for each $v \in V$. We next compute a minimum 2-size 2-partition $\mathscr{X} = \{X_1, X_2\}$ and recursively call the algorithm for functions $f_{/X_1}$ and $f_{/X_2}$ and obtain two candidate partitions. Finally, we take the minimum of all candidate partitions. Therefore, the noncrossing property in Lemma 3.1 produces a polynomial time algorithm for the minimum 3-partition problem.

Let us consider possible generalizations of the noncrossing property. The first one is a stronger property for 2- and 4-partitions. Let $\mathscr{X} = \{X_1, X_2\}$ denote a minimum 2-partition of f. There exists a minimum 4-partition $\mathscr{Y} = \{Y_1, Y_2, Y_3, Y_4\}$ of f such that either $\mathscr{X}$ is noncrossing with $\mathscr{Y}$ or $X_1 = Y_1 \cup Y_2$ after some renumbering of the indices. We remark that $\mathscr{X}$ is *not* necessarily h-size for $h \geq 2$, and hence the property above does not provide a polynomial divide-and-conquer algorithm for the minimum 4-partition problem. If $\mathscr{X}$ is assumed to be h-size for $h \geq 2$, then we have additional cases, one of which also blocks the construction of a polynomial time algorithm (See the case (iii) in Theorem 6 in [26]).

The second generalization is to directly use noncrossing property for 3- and 4-partitions. Assume that every minimum h-size 3-partition $\mathscr{X} = \{X_1, X_2, X_3\}$ for $h \geq 2$ is noncrossing with a minimum 4-partition. However, this does not provide a polynomial divide-and-conquer algorithm for the minimum 4-partition problem. Note that the algorithm calls itself for $f_{/X_i}, i = 1, 2, 3$, where in the worst case, two of the recursive calls have size $n-1$, this implies that a simple divide-and-conquer algorithm requires exponential time.

In this paper, we introduce the concept of compatibility to overcome this difficulty. A partition $\mathscr{X}$ is *compatible* with partition $\mathscr{Y}$, if $|\mathscr{X}| - 1$ components of $\mathscr{X}$ are noncrossing with $\mathscr{Y}$. We write $\mathscr{X} \lhd \mathscr{Y}$. Note if $\mathscr{X}$ and $\mathscr{Y}$ are 2- and 3- partitions, respectively, then the compatibility relation is identical to the noncrossing property.

Theorem 3.1 *Let f be a submodular function on at least 13 vertices. If all minimum 4-partitions are 3-size, then every minimum non-trivial 3-partition is compatible with some minimum 4-partition.*

Compatibility is a very strong property about two partitions, and it is very unlikely to hold true in general. In fact, there are examples where a minimum 4-partition is not compatible with any minimum 5-partition, as we will see in Sect. 7.

The proof of Theorem 3.1 can be found in the next section. We remark that the proof is based on case analysis. Based on Theorem 3.1, it is not difficult to see that the following simple contraction based algorithm solves the minimum 4-partition problem. We invite the readers to spot the difference between the minimum 3-partition algorithm in Fig. 1 and the minimum 4-partition algorithm in Fig. 2.

3.1 The algorithm

Either there is a minimum 4-partition that is not 3-size, so there is a part of size at most 2. We try all possible such parts, and solve the minimum 3-partition problem on the remaining part. Otherwise, there is a 3-size minimum 4-partition. We find a minimum non-trivial 3-partition, and we know it is compatible to some minimum 4-partition by Theorem 3.1. Therefore, we can contract two of the parts. Since we do not know which, we try all possibilities. Finally, we collect all candidates we computed, and find the minimum among them. The full algorithm is described in Fig. 2.

We computed $O(n^2)$ minimum 3-partitions, and computed a minimum non-trivial 3-partition. The remaining non-recursive operations take constant time per statement.

Fig. 2 Compute a minimum 4-partition

```
MIN4PARTITION(f):
  V ← domain(f)
  if |V| ≤ 12
      return the optimum by brute force
  for X ∈ (V choose 1) ∪ (V choose 2)
      add candidate {X} ∪ MIN3PARTITION(f_\X)
  𝒳 ← MINNONTRIVIAL3PARTITION(f)
  for {A,B} ∈ (𝒳 choose 2)
      add candidate MIN4PARTITION((f_/A)_/B)
  return minimum over all candidates
```

The minimum 3-partition problem is solvable in polynomial time as we noted in this section. Thus, except for the proof of Theorem 3.1, what remains to be done is to provide a polynomial time algorithm to compute a minimum *nontrivial* 3-partition, which is discussed in Sect. 5.

4 Proof of Theorem 3.1

Let V be the ground set of some submodular function f. Consider any 3-partition $\mathscr{X} = \{X_1, X_2, X_3\}$ and $\mathscr{Y} = \{Y_1, Y_2, Y_3, Y_4\}$. A submodular function f on at least 13 vertices where every minimum 4-partition is 3-size, is called a *valid* submodular function. A configuration is *valid*, if it can be generated by a minimum non-trivial 3-partition $\mathscr{X}$, and a minimum 4-partition $\mathscr{Y}$ of some valid submodular function f.

Let M be some configuration. Consider a submodular function f, such that it has a minimum non-trivial 3-partition $\mathscr{X}$ and a 3-size minimum 4-partition $\mathscr{Y}$ that generate M. Such functions are called *M-agreeable*. If there is a minimum 4-partition $\mathscr{Y}'$ of f that $\mathscr{X}$ is compatible with, then we say M is f-good. If M is f-good for all M-agreeable f, we say M is *good*. If M is good, then any configuration equivalent to M is also good.

If all valid configurations are good, then Theorem 3.1 is true. Indeed, for any valid submodular function, the minimum non-trivial 3-partition and 3-size minimum 4-partition generates a good configuration, which implies there is a minimum 4-partition that does not cross the minimum non-trivial 3-partition.

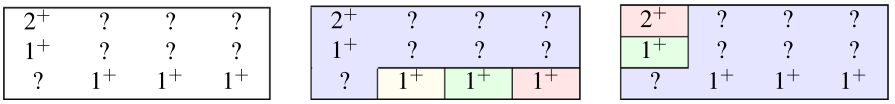
We use the following idea repeatedly, and hence we make it a proposition.

Proposition 4.1 *Let $\mathscr{X}$ be a minimum partition with property P, and $\mathscr{Y}$ be a minimum partition with property Q. Let $\mathscr{X}'$ and $\mathscr{Y}'$ be a partition with properties P and Q, respectively. If $f(\mathscr{X}) + f(\mathscr{Y}) \geq f(\mathscr{X}') + f(\mathscr{Y}')$, then $\mathscr{Y}'$ is a minimum partition with property Q.*

Proof Note that $f(\mathscr{X}') \geq f(\mathscr{X})$ and $f(\mathscr{Y}') \geq f(\mathscr{Y})$ because $\mathscr{X}'$ and $\mathscr{Y}'$ have properties P and Q, respectively. Hence, we obtain $f(\mathscr{X}') = f(\mathscr{X})$ and $f(\mathscr{Y}) = f(\mathscr{Y}')$. □

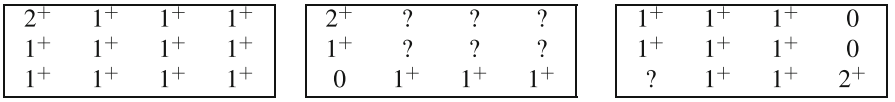
In order to simplify the proof and avoid repeating the same set up each time, the following convention is established.

When we try to prove a valid configuration M is good, we always consider a minimum non-trivial 3-partition $\mathscr{X} = \{X_1, X_2, X_3\}$, 3-size minimum 4-partition



(a) M (b) $\mathscr{Y}'$ (c) $\mathscr{X}'$

Fig. 3 The configuration M and partitions $\mathscr{Y}'$ and $\mathscr{X}'$ in the proof of Lemma 4.1. To understand the illustration of $\mathscr{Y}'$, note it is an colored overlay over the configuration matrix M. Each color represents a partition class of $\mathscr{Y}'$. Since the ith row and jth column represents $Z_{ij} = X_i \cap Y_j$ together with additional cardinally information, the illustration shows $\mathscr{Y}'$ is a 4-partition with partition classes Z_{32}, Z_{33}, Z_{34} and a set containing all the remaining elements



(a) Case 1 (b) Case 2 (c) Case 3

Fig. 4 The configurations in the proof of Theorem 4.1

$\mathscr{Y} = \{Y_1, Y_2, Y_3, Y_3\}$ and a valid submodular function f that generates the configuration M. Let $Z_{ij} = X_i \cap Y_j$ be the *cells* of $\mathscr{X}$ and $\mathscr{Y}$. Let n_i be the number of non-zero values in the ith row of M, and m_j the number of non-zero values in the jth column of M. From this point on, we omit the setup in all the proofs in this section.

There are simple patterns where the configurations has to be good. The following is a pattern adopted from [14, Lemma 2.5].

Lemma 4.1 *Let M be a configuration. If $M_{11} \geq 2$, and $M_{21}, M_{32}, M_{33}, M_{34} \geq 1$, then M is good.*

Proof Define $\mathscr{Y}' = \{X_1 \cup X_2 \cup Y_1, Z_{32}, Z_{33}, Z_{34}\}$ and $\mathscr{X}' = \{Z_{11}, Z_{21}, X_3 \cup Y_2 \cup Y_3 \cup Y_4\}$. Then $\mathscr{X}'$ is a non-trivial 3-partition and $\mathscr{Y}'$ is a 4-partition, and $\mathscr{X} \lhd \mathscr{Y}'$. See Fig. 3 for illustrations.

For any submodular function f, it holds that

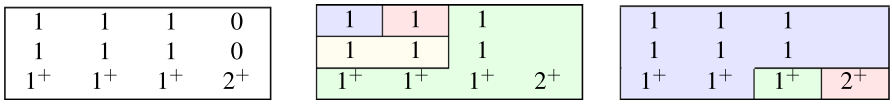
$$\begin{aligned}
f(X_1) + f(Y_1) &\geq f(X_1 \cup Y_1) + f(Z_{11}) \\
f(X_1 \cup Y_1) + f(X_2) &\geq f(X_1 \cup X_2 \cup Y_1) + f(Z_{21}) \\
f(X_3) + f(Y_2) &\geq f(X_3 \cup Y_2) + f(Z_{32}) \\
f(X_3 \cup Y_2) + f(Y_3) &\geq f(X_3 \cup Y_2 \cup Y_3) + f(Z_{33}) \\
f(X_3 \cup Y_2 \cup Y_3) + f(Y_4) &\geq f(X_3 \cup Y_2 \cup Y_3 \cup Y_4) + f(Z_{34}).
\end{aligned}$$

By summing all the inequalities above, we obtain

$$f(\mathscr{X}) + f(\mathscr{Y}) \geq f(\mathscr{X}') + f(\mathscr{Y}').$$

By Proposition 4.1, $\mathscr{Y}'$ is a minimum 4-partition, which completes the proof. □

If any configuration equivalent to M satisfies the property in Lemma 4.1, we say M contains a *cross*. If M has many non-zero elements, then M contains a cross. We formalize it below.



(a) M (b) $\mathscr{Y}'$ (c) $\mathscr{X}'$

Fig. 5 The configuration M and partitions $\mathscr{Y}'$ and $\mathscr{X}'$ in the Case 3 of the proof of Theorem 4.1

Theorem 4.1 *If M is a valid configuration where each row has at least 3 non-zero elements, then it contains a cross, and therefore M is good.*

Proof We consider the following 3 cases.

Case 1. $m_j = 3$ for all $j \in \{1, 2, 3, 4\}$, then M has at least one entry ≥ 2, and all other values are at least 1. Without loss of generality, assume $M_{11} \geq 2$. See Fig. 4a. M contains a cross.

Case 2. Assume $m_j \geq 2$ for all j and not all of them are 3. Without loss of generality, assume $M_{31} = 0$. This implies $M_{32}, M_{33}, M_{34} \geq 1$. Since $m_1 = 2$, we have that $M_{11}, M_{21} \geq 1$, and at least one is 2. Without loss of generality, let $M_{11} \geq 2$. See Fig. 4b. This gives us a cross.

Case 3. Consider $m_j = 1$ for some j. Without loss of generality, we can assume $m_4 = 1$, and $M_{14} = M_{24} = 0$. This shows $M_{34} \geq 2$. We also assume that $M_{32}, M_{33} \geq 1$, because $n_3 \geq 3$.

See Fig. 4c for the configuration.

If $M_{31} = 0$, then M_{21} or M_{11} is at least 2, and we obtain a cross. Assume that $M_{31} \geq 1$. If any M_{ij} for $i \in \{1, 2\}$ and $j \in \{1, 2, 3\}$ is 2, then there exists a cross. Hence the only remaining case is when $M_{ij} = 1$ for all $i \in \{1, 2\}$ and $j \in \{1, 2, 3\}$. Let $\mathscr{X}' = \{X_1 \cup X_2 \cup Y_1 \cup Y_2, Z_{33}, Z_{34}\}$ and $\mathscr{Y}' = \{X_3 \cup Y_3 \cup Y_4, Z_{11}, Z_{12}, Z_{21} \cup Z_{22}\}$. $\mathscr{X}'$ is a non-trivial 3-partition. Because $M_{ij} = 1$ for each $i \in \{1, 2\}$ and $j \in \{1, 2, 3\}$, $\mathscr{Y}'$ is a 4-partition. See Fig. 5 for illustration.

By submodularity, it holds that

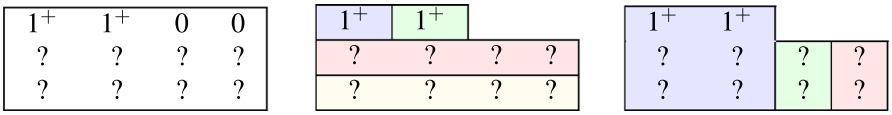
$$
\begin{aligned}
f(X_3) + f(Y_3) &\geq f(X_3 \cup Y_3) + f(Z_{33}) \\
f(X_3 \cup Y_3) + f(Y_4) &\geq f(X_3 \cup Y_3 \cup Y_4) + f(Z_{34}) \\
f(X_1) + f(Y_1) &\geq f(X_1 \cup Y_1) + f(Z_{11}) \\
f(X_1 \cup Y_1) + f(Y_2) &\geq f(X_1 \cup Y_1 \cup Y_2) + f(Z_{12}) \\
f(X_1 \cup Y_1 \cup Y_2) + f(X_2) &\geq f(X_1 \cup X_2 \cup Y_1 \cup Y_2) + f(Z_{21} \cup Z_{22}).
\end{aligned}
$$

According to the above, it holds that

$$
f(\mathscr{X}) + f(\mathscr{Y}) \geq f(\mathscr{X}') + f(\mathscr{Y}').
$$

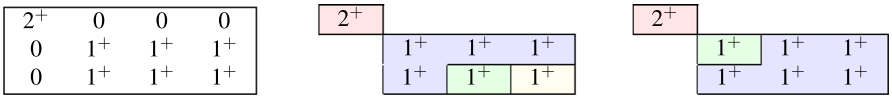
We invoke Proposition 4.1, and it shows $\mathscr{Y}'$ is a minimum 4-partition. However, $M_{11} = 1$, and hence $\mathscr{Y}'$ is not 3-size. A contradiction to M being a valid configuration. □

The following lemma is an adaptation of [14, Lemma 2.7].



(a) M (b) $\mathscr{Y}'$ (c) $\mathscr{X}'$

Fig. 6 The configuration M and partitions $\mathscr{Y}'$ and $\mathscr{X}'$ in the proof of Lemma 4.2



(a) M (b) $\mathscr{Y}'$ (c) $\mathscr{X}'$

Fig. 7 The configuration M and partitions $\mathscr{Y}'$ and $\mathscr{X}'$ in Case 2 of the proof of Theorem 4.2

Lemma 4.2 *Let M be a valid configuration with a row with exactly two non-zeros. Then M is good.*

Proof Without loss of generality, assume $M_{11}, M_{12} \geq 1$ and $M_{13} = M_{14} = 0$. Let $\mathscr{Y}' = \{Z_{11}, Z_{12}, X_2, X_3\}$ and $\mathscr{X}' = \{X_1 \cup Y_1 \cup Y_2, Y_3, Y_4\}$. Note that $\mathscr{Y}'$ is a 4-partition and $\mathscr{X} \lhd \mathscr{Y}'$. Since $\mathscr{Y}$ is 3-size, $|Y_3| \geq 2$, and therefore $\mathscr{X}'$ is a non-trivial 3-partition. See Fig. 6 for illustration.

By submodularity, it holds that

$$f(X_1) + f(Y_1) \geq f(X_1 \cup Y_1) + f(Z_{11})$$
$$f(X_1 \cup Y_1) + f(Y_2) \geq f(X_1 \cup Y_1 \cup Y_2) + f(Z_{12}).$$

According to the above, it holds that

$$f(\mathscr{X}) + f(\mathscr{Y}) \geq f(\mathscr{X}') + f(\mathscr{Y}').$$

By Proposition 4.1, $\mathscr{Y}'$ is a minimum 4-partition. □

Theorem 4.2 *If M is a valid configuration where there exists a row with at most two non-zero elements, then it is good.*

Proof If any row has exactly 2 non-zero elements, then we are done by Lemma 4.2.

Hence, consider the case where a row has exactly 1 non-zero element, and no row has exactly 2 non-zero elements. Without loss of generality, let $n_1 = 1$ and $M_{11} \geq 2$.

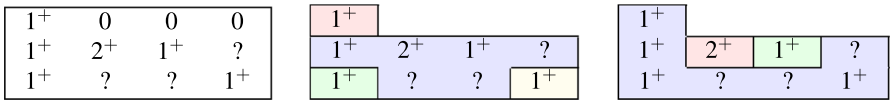
Case 1: $n_i = 1$ for some $i \in \{2, 3\}$. Then M is good because $\mathscr{X} \lhd \mathscr{Y}$.

Case 2: $n_2 = n_3 = 3, m_1 = 1$. Then we have $M_{i1} = 0$ for $i \in \{2, 3\}$ and $M_{i2}, M_{i3}, M_{i4} \geq 1$ for each $i \in \{2, 3\}$. Since $|Y_1| \geq 3$, we have $M_{11} \geq 2$.

Let $\mathscr{Y}' = \{X_2 \cup Y_2, X_1, Z_{33}, Z_{34}\}$ and $\mathscr{X}' = \{X_3 \cup Y_3 \cup Y_4, Y_1, Z_{22}\}$. Then $\mathscr{Y}'$ is a 4-partition, $\mathscr{X} \lhd \mathscr{Y}'$, and $\mathscr{X}'$ is a non-trivial 3-partition. See Fig. 7 for illustration.

By submodularity, it holds that

$$f(X_2) + f(Y_2) \geq f(X_2 \cup Y_2) + f(Z_{22})$$



(a) M (b) $\mathscr{Y}'$ (c) $\mathscr{X}'$

Fig. 8 The configuration M and partitions $\mathscr{Y}'$ and $\mathscr{X}'$ in Case 4 of the proof of Theorem 4.2

$$f(X_3) + f(Y_3) \geq f(X_3 \cup Y_3) + f(Z_{33})$$
$$f(X_3 \cup Y_3) + f(Y_4) \geq f(X_3 \cup Y_3 \cup Y_4) + f(Z_{34}).$$

According to the above, it holds that

$$f(\mathscr{X}) + f(\mathscr{Y}) \geq f(\mathscr{X}') + f(\mathscr{Y}').$$

By Proposition 4.1, $\mathscr{Y}'$ is a minimum 4-partition.

Case 3: $n_2, n_3 \geq 3, m_1 = 2$. Without loss of generality, we can assume that $M_{21} \geq 1$ and $M_{31} = 0$. Since $|Y_1| \geq 3$, we have $M_{11} \geq 2$ or $M_{21} \geq 2$, i.e.,

$$M = \begin{array}{|cccc|}\hline 2^+ & 0 & 0 & 0 \\ 1^+ & 1^+ & 1^+ & ? \\ 0 & 1^+ & 1^+ & 1^+ \\ \hline \end{array} \quad \text{or} \quad M = \begin{array}{|cccc|}\hline 1^+ & 0 & 0 & 0 \\ 2^+ & 1^+ & 1^+ & ? \\ 0 & 1^+ & 1^+ & 1^+ \\ \hline \end{array}.$$

In either case, there is a cross, and M is good.

Case 4: $n_1 = 1, n_2, n_3 \geq 3, m_1 = 3$. Without loss of generality, we can assume that $M_{23} \geq 1$. We also assume that $M_{34} \geq 1$ since $n_3 \geq 3$. Since $\mathscr{Y}$ is 3-size and $M_{12} = 0$, without loss of generality, we can assume that $M_{22} \geq 2$. Otherwise, we can exchange X_2 and X_3, and then exchange Y_3 and Y_4. Let $\mathscr{X}' = \{X_3 \cup Y_1 \cup Y_4, Z_{22}, Z_{23}\}$ and $\mathscr{Y}' = \{X_2 \cup Y_2 \cup Y_3, X_1, Z_{31}, Z_{34}\}$. $\mathscr{X}'$ is a non-trivial 3-partition, $\mathscr{Y}'$ is a 4-partition and $\mathscr{X} \lhd \mathscr{Y}'$. See Fig. 8 for illustration.

By submodularity, it holds that

$$f(X_2) + f(Y_2) \geq f(X_2 \cup Y_2) + f(Z_{22})$$
$$f(X_2 \cup Y_2) + f(Y_3) \geq f(X_2 \cup Y_2 \cup Y_3) + f(Z_{23})$$
$$f(X_3) + f(Y_1) \geq f(X_3 \cup Y_1) + f(Z_{31})$$
$$f(X_3 \cup Y_1) + f(Y_4) \geq f(X_3 \cup Y_1 \cup Y_4) + f(Z_{34}).$$

According to the above, it holds that

$$f(\mathscr{X}) + f(\mathscr{Y}) \geq f(\mathscr{X}') + f(\mathscr{Y}').$$

By Proposition 4.1, $\mathscr{Y}'$ is a minimum 4-partition. □

Theorems 4.1 and 4.2 show that all valid configurations are good, and hence prove Theorem 4.3.

Theorem 4.3 *If all minimum* 4*-partition are* 3*-size, then every minimum non-trivial* 3*-partition is compatible with some minimum* 4*-partition.*

5 $(1, \ell)$-size 3-partition

In this section, we present an algorithm for computing a minimum nontrivial 3-partition for a given submodular function by giving an algorithm for the $(1, \ell)$-size 3-partition problem. This algorithm is based on the following noncrossing property.

Lemma 5.1 *Let* $f : 2^V \to \mathbb{R}$ *be a submodular function with* $n \geq 18\ell - 17$*, and let* $\mathscr{X} = \{X_1, X_2\}$ *denote a minimum* $(3\ell - 2)$*-size* 2*-partition of* f*. If* f *has a minimum* $(1, \ell)$*-size* 3*-partition of* $(4\ell - 3)$*-size, then it contains a minimum* $(1, \ell)$*-size* 3*-partition with which* $\mathscr{X}$ *is noncrossing.*

Proof Let $\mathscr{Y} = \{Y_1, Y_2, Y_3\}$ denote a minimum $(1, \ell)$-size 3-partition of $(4\ell - 3)$-size of f. Let M be the configuration generated by $\mathscr{X}$ and $\mathscr{Y}$. Let $Z_{ij} = X_i \cap Y_j$.

Suppose any row of M has exactly one non-zero. The proof is trivial.

We say that M *contains a cross* if there exist i and j such that $M_{i'j} \geq 3\ell - 2$ for $i' \neq i$, $M'_{ij} \geq 1$ for all $j' \neq j$, and $M'_{ij} \geq \ell$ for some $j' \neq j$. We also say that *the cross is centered at* (i, j). See Fig. 9 for an example.

If M contains a cross, then there is a minimum nontrivial 3-partition which is noncrossing with $\mathscr{X}$. Indeed, without loss of generality, let $M_{11} \geq \ell$, $M_{12} \geq 1$ and $M_{23} \geq 3\ell - 2$. Define $\mathscr{Y}' = \{X_2 \cup Y_3, Z_{11}, Z_{12}\}$ and $\mathscr{X}' = \{X_1 \cup Y_1 \cup Y_2, Z_{23}\}$. Certainly $\mathscr{X} \lhd \mathscr{Y}'$. We note that $\mathscr{Y}'$ is a $(1, \ell)$-size 3-partition, and $\mathscr{X}'$ is a $(3\ell-2)$-size 2-partition. By submodularity,

$$
\begin{aligned}
f(X_1) + f(Y_1) &\geq f(X_1 \cup Y_1) + f(Z_{11}) \\
f(X_1 \cup Y_1) + f(Y_2) &\geq f(X_1 \cup Y_1 \cup Y_2) + f(Z_{12}) \\
f(X_2) + f(Y_3) &\geq f(X_2 \cup Y_3) + f(Z_{23}).
\end{aligned}
$$

Hence we obtain

$$
f(\mathscr{X}) + f(\mathscr{Y}) \geq f(\mathscr{X}') + f(\mathscr{Y}').
$$

Therefore, $\mathscr{Y}'$ is a minimum $(1, \ell)$-size 3-partition where $\mathscr{X}$ does not cross it.

Consider the case where each row of M has at least 2 non-zero elements. Because $|V| \geq 18\ell - 17$ and M has 6 entries, at least one entry in M has value at least $3\ell - 2$. We can assume $M_{23} \geq 3\ell - 2$. Without loss of generality, let $M_{11} \geq M_{12}$. Next, we show all possible configurations have a cross.

Case 1: $M_{11} \geq \ell$, $M_{12} \geq 1$. See Fig. 10a. There is a cross centered at $(1, 3)$.

Case 2: $M_{11} \geq \ell$, $M_{12} = 0$. This shows $M_{13} \geq 1$ and $M_{22} \geq 3\ell - 2$. See Fig. 10b. There is a cross centered at $(1, 2)$.

Case 3: $1 \leq M_{11} \leq \ell - 1$. This shows $M_{22} \geq 3\ell - 2$, because $M_{12} + M_{22} \geq 4\ell - 3$. Also, $M_{13} \geq 4\ell - 3 - M_{11} - M_{12} \geq \ell$. See Fig. 10c. There is a cross centered at $(1, 2)$. □

ℓ^+	1^+	?
?	?	$3\ell-2^+$

Fig. 9 A configuration M, with a cross that is centered at $(1, 3)$. Because $M_{23} \geq 3\ell - 2$, $M_{11}, M_{12} \geq 1$, and $M_{11} \geq \ell$

ℓ^+	1^+	?
?	?	$3\ell-2^+$

(a) Case 1

ℓ^+	0	1^+
?	$3\ell-2^+$	$3\ell-2^+$

(b) Case 2

$\ell-1^-$	$\ell-1^-$	ℓ^+
?	$3\ell-2^+$	$3\ell-2^+$

(c) Case 3

Fig. 10 The configurations in the proof of Lemma 5.1. Note in Case 3, the top right element is also known to be non-zero

In order to use the noncrossing property in Lemma 5.1, we need to compute a minimum $(3\ell - 2)$-size 2-partition of the submodular function f.

Vazirani-Yannakakis [30] proposed an algorithm for enumerating all 2-cuts in a given graph, in the order of nondecreasing weights. Nagamochi-Ibaraki [24] remarked that Vazirani-Yannakakis' algorithm can be extended to enumerating all 2-partitions in an arbitrary system.

For two disjoint sets $S, T \subseteq V$, a family $\{X, V\setminus X\}$ is called an (S, T)-*partition* if $S \subseteq X$ and $T \subseteq V\setminus X$.

Lemma 5.2 (Vazirani-Yannakakis [30], Nagamochi-Ibaraki [24]) *Let $f : 2^V \to \mathbb{R}$ be an arbitrary function with $n = |V|$. All the 2-partitions $\{X, V\setminus X\}$ of V can be enumerated in the order of nondecreasing weights with $O(n\tau^*(n))$-time delay between two consecutive outputs, where $\tau^*(n)$ denotes the time required for computing a minimum (S, T)-partition of f.*

It is well-known that a minimum (S, T)-partition in the submodular function f can be computed in $\tau(n)$ time, i.e., the time required for submodular function minimization. The number of 2-partitions that are not k-size is $O(n^{k-1})$. Thus after enumerating $O(n^{k-1})$ 2-partitions, we can find a minimum k-size 2-partition, which implies the following lemma.

Lemma 5.3 *Let $f : 2^V \to \mathbb{R}$ be a submodular function with $n = |V|$, and let k (≥ 2) denote a positive integer. A minimum k-size 2-partition of f can be computed in $O(n^k\tau(n))$ time.*

We are now ready to describe an algorithm for the minimum nontrivial 3-partition problem. See Fig. 11. It is similar to both MIN3PARTITION and MIN4PARTITION.

Lemma 5.4 *For a submodular function $f : 2^V \to \mathbb{R}$ with $n = |V|$, there is an algorithm that computes a minimum $(1, \ell)$-size 3-partition of f in $O(n^{4\ell-3}\tau(n))$ time.*

Proof Consider MIN$(1, \ell)$SIZE3PARTITION. Finding the minimum $3\ell - 2$-size 2-partition takes $O(n^{3\ell-2}\tau(n))$ time. Finding the minimum non-$4\ell - 3$-size 3-partitions takes $O(n^{4\ell-4}\tau(n))$ time.

Fig. 11 Compute a minimum $(1, \ell)$-size 3-partition

```
MIN(1,ℓ)SIZE3PARTITION(f,ℓ):
  V ← domain(f)
  if |V| ≤ 18ℓ − 17
      return the optimum by brute force
  for X ∈ ⋃_{j=1}^{4ℓ−4} (V choose j)
      add candidate {X} ∪ MIN2PARTITION(f_{\X})
  𝒳 ← MINSIZE2PARTITION(f, 3ℓ − 2)
  for X ∈ 𝒳
      add candidate MIN(1,ℓ)SIZE3PARTITION(f_{/X}, ℓ)
  return minimum over all candidates
```

Let $T(n)$ denote the running time of MIN$(1, \ell)$SIZE3PARTITION. The following recursive relation holds. $T(n) = O(1)$ for $n \leq 18\ell - 17$, and otherwise

$$T(n) = \max_{\substack{a+b=n \\ 1\leq a\leq b\leq n-3\ell+1}} T(a+1) + T(b+1) + O(n^{4\ell-4}\tau(n)) = O(n^{4\ell-3}\tau(n)).$$

□

By setting $\ell = 2$, we obtain the following corollary.

Corollary 5.1 *For a submodular function $f : 2^V \to \mathbb{R}$ with $n = |V|$, there is an algorithm that computes a minimum nontrivial 3-partition of f in $O(n^5\tau(n))$ time.*

6 Time complexity of Algorithm MIN4PARTITION

In this section, we analyze the time complexity of Algorithm MIN4PARTITION.

Since the minimum 3-partition problem can be solved in $O(n^3\tau(n))$ time [27], the minimum non-3-size 4-partitions can be found in $O(n^5\tau(n))$ time. A minimum nontrivial 3-partition can be computed in $O(n^5\tau(n))$ time by Corollary 5.1.

Let $T(n)$ be the running time of the 4-partition algorithm in Fig. 2 with $n = |V|$. The following recursive relation holds. $T(n) = O(1)$ for $n \leq 12$, and otherwise

$$T(n) = \max_{\substack{a+b+c=n \\ 1\leq a\leq b\leq c\leq n-3}} T(a+2) + T(b+2) + T(c+2) + O(n^5\tau(n)) = O(n^6\tau(n)).$$

As a result, we have the following main theorem.

Theorem 6.1 *For a submodular function $f : 2^V \to \mathbb{R}$ with $n = |V|$, Algorithm* MIN4PARTITION(f) *computes a minimum 4-partition of f in* $O(n^6\tau(n))$ *time.*

As a minimum k-partition in symmetric submodular functions can be found in n^{2k-2} calls to minimum $(k-1)$-partition in submodular functions [4], we observe the following corollary.

Corollary 6.1 *For a symmetric submodular function $f : 2^V \to \mathbb{R}$ with $n = |V|$, a minimum 5-partition can be found in $O(n^{14}\tau(n))$ time.*

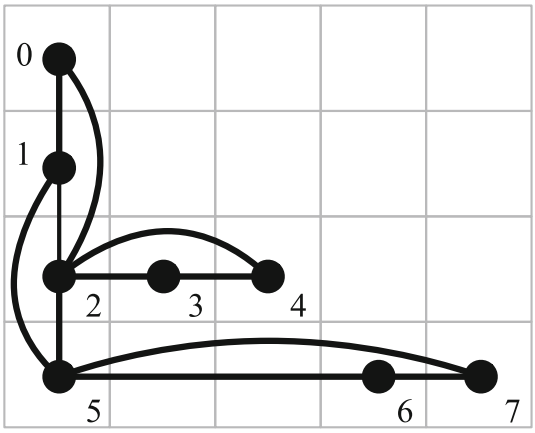


Fig. 12 The 8 vertex counterexample, the light edge is the edge of 1/2 weight

7 Concluding remark

In this paper, we have considered the minimum 4-partition problem of submodular functions. Using compatibility for 3- and 4-partitions, we have provided a polynomial-time (exact) algorithm, which settles the complexity status of the problem [11, 22, 24, 27, 32, 35].

It is still open if the minimum k-partition problem is solvable in polynomial time for each fixed k. This is left for future work. Since all our algorithms are very similar, it seems to indicate a generalization to minimum 5-partition problem.

Unfortunately, we remark the obvious generalization of our algorithm does not work for minimum 5-partition. Consider the following configuration M, if a submodular function with unique minimum 4-partition and minimum 5-partition generates it as a configuration, then the minimum 4-partition is not compatible with any minimum 5-partition.

$$M = \begin{array}{|ccccc|} \hline 1^+ & 0 & 0 & 0 & 0 \\ 1^+ & 0 & 0 & 0 & 0 \\ 1^+ & 1^+ & 1^+ & 0 & 0 \\ 1^+ & 0 & 0 & 1^+ & 1^+ \\ \hline \end{array}.$$

In fact, even in graphs, there is an infinite number of examples that generates the above configuration M. Consider a graph on 8 vertices. The vertices are 0 to 7. The edges are $\{0, 1\}$, $\{1, 2\}$, $\{0, 2\}$, $\{1, 5\}$, $\{2, 3\}$, $\{2, 4\}$, $\{2, 5\}$, $\{3, 4\}$, $\{5, 6\}$, $\{5, 7\}$ and $\{6, 7\}$. All edges have unit weight, except the edge $\{1, 2\}$, which has weight $\frac{1}{2}$. See the Fig. 12 for the example.

The graph has a unique minimum 4-partition $\{0\}$, $\{1\}$, $\{2, 3, 4\}$, $\{5, 6, 7\}$, which has value of 4.5. There is also a unique minimum 5-partition. $\{0, 1, 2, 5\}$, $\{3\}$, $\{4\}$, $\{6\}$, $\{7\}$ with value 6. One can blow up this example to arbitrarily large graph by replacing each vertex with a clique with large edge weights.

Acknowledgements Chao would like to thank Chandra Chekuri and Karthekeyan Chandrasekaran for early discussions.

Funding This work was partially supported by JST ERATO Grant Number JPMJER2301 and Japan Society for the Promotion of Science (JSPS) KAK420 ENHI Grant Numbers JP19K22841, JP20H00609, and JP20H05967.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Beideman, C., Chandrasekaran, K., Wang, W.: Counting and enumerating optimum cut sets for hypergraph k-partitioning problems for fixed k. In: Bojańczyk, M., Merelli, E., Woodruff, D.P. (eds.) 49th International Colloquium on Automata, Languages, and Programming (ICALP 2022), Leibniz International Proceedings in Informatics (LIPIcs), vol. 229, pp. 16:1–16:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2022). https://doi.org/10.4230/LIPIcs.ICALP.2022.16. https://drops.dagstuhl.de/opus/volltexte/2022/16357
2. Beideman, C., Chandrasekaran, K., Wang, W.: Deterministic enumeration of all minimum k-cut-sets in hypergraphs for fixed k. In: Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 2208–2228. Society for Industrial and Applied Mathematics, Philadelphia (2022). https://doi.org/10.1137/1.9781611977073
3. Beideman, C., Chandrasekaran, K., Xu, C.: Multicriteria cuts and size-constrained k-cuts in hypergraphs. Math. Program. (2021). https://doi.org/10.1007/s10107-021-01732-0
4. Chandrasekaran, K., Chekuri, C.: Min–max partitioning of hypergraphs and symmetric submodular functions. In: Proceedings of the Thirty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'21, pp. 1026–1038. Society for Industrial and Applied Mathematics, USA (2021)
5. Chandrasekaran, K., Chekuri, C.: Hypergraph k-cut for fixed k in deterministic polynomial time. Math. Oper. Res. (2022). https://doi.org/10.1287/moor.2021.1250
6. Chandrasekaran, K., Xu, C., Yu, X.: Hypergraph k-cut in randomized polynomial time. Math. Program. **186**(1–2), 85–113 (2021). https://doi.org/10.1007/s10107-019-01443-7
7. Chekuri, C., Ene, A.: Approximation algorithms for submodular multiway partition. In: 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science, pp. 807–816. IEEE, Palm Springs, CA, USA (2011). https://doi.org/10.1109/FOCS.2011.34
8. Chekuri, C., Quanrud, K., Xu, C.: LP Relaxation and tree packing for minimum k-cut. SIAM J. Discrete Math. **34**(2), 1334–1353 (2020). https://doi.org/10.1137/19M1299359
9. Chekuri, C., Xu, C.: Minimum cuts and sparsification in hypergraphs. SIAM J. Comput. **47**(6), 2118–2156 (2018). https://doi.org/10.1137/18M1163865
10. Fox, K., Panigrahi, D., Zhang, F.: Minimum cut and minimum k-cut in hypergraphs via branching contractions. In: Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 881–896. SIAM (2019)
11. Fukunaga, T.: Computing minimum multiway cuts in hypergraphs. Discrete Optim. **10**(4), 371–382 (2013). https://doi.org/10.1016/j.disopt.2013.10.002
12. Gasieniec, L., Jansson, J., Lingas, A., Östlin, A.: On the complexity of constructing evolutionary trees. J. Comb. Optim. **3**(2/3), 183–197 (1999). https://doi.org/10.1023/A:1009833626004
13. Goldschmidt, O., Hochbaum, D.S.: A polynomial algorithm for the k-cut problem for fixed k. Math. Oper. Res. **19**, 24–37 (1994). https://doi.org/10.1287/moor.19.1.24
14. Guinez, F., Queyranne, M.: The size-constrained submodular k-partition problem (2012). https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxmbGF2aW9ndWluZXpob21lcGFnZXxneDo0NDVlMThkMDg4ZWRlOGI1. Unpublished manuscript. See also https://smartech.gatech.edu/bitstream/handle/1853/43309/Queyranne.pdf
15. Gupta, A., Harris, D.G., Lee, E., Li, J.: Optimal bounds for the k-cut problem. J. ACM (2021). https://doi.org/10.1145/3478018

16. Gupta, A., Lee, E., Li, J.: Faster exact and approximate algorithms for k-cut. In: 2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS), pp. 113–123 (2018). https://doi.org/10.1109/FOCS.2018.00020
17. Hirayama, T., Liu, Y., Makino, K., Shi, K., Xu, C.: A polynomial time algorithm for finding a minimum 4-partition of a submodular function. In: Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1680–1691. https://doi.org/10.1137/1.9781611977554.ch64
18. Kamidoi, Y., Yoshida, N., Nagamochi, H.: A deterministic algorithm for finding all minimum k-way cuts. SIAM J. Comput. **36**(5), 1329–1341 (2006). https://doi.org/10.1137/050631616
19. Klimmek, R., Wagner, F.: A simple hypergraph min cut algorithm. Tech. Rep. B 96-02, FU Berlin Fachbereich Mathematik und Informatik (1996)
20. Lee, Y.T., Sidford, A., Wong, S.C.W.: A faster cutting plane method and its implications for combinatorial and convex optimization. In: 2015 IEEE 56th Annual Symposium on Foundations of Computer Science, pp. 1049–1065 (2015). https://doi.org/10.1109/FOCS.2015.68
21. Mak, W.K., Wong, D.F.: Fast hypergraph min-cut algorithm for circuit partitioning. Integr. VLSI J. **30**(1), 1–11 (2000). https://doi.org/10.1016/S0167-9260(00)00008-0
22. Nagamochi, H.: Algorithms for the minimum partitioning problems in graphs. Electron. Commun. Jpn. **90**(10), 63–78 (2007). https://doi.org/10.1002/ecjc.20341. (**Part III Fundamental Electronic Science**)
23. Nagamochi, H., Ibaraki, T.: Computing edge-connectivity in multigraphs and capacitated graphs. SIAM J. Discrete Math. **5**(1), 54–66 (1992)
24. Nagamochi, H., Ibaraki, T.: A fast algorithm for computing minimum 3-way and 4-way cuts. Math. Program. **88**(3), 507–520 (2000). https://doi.org/10.1007/PL00011383
25. Nagamochi, H., Katayama, S., Ibaraki, T.: A faster algorithm for computing minimum 5-way and 6-way cuts in graphs. J. Comb. Optim. **4**(2), 151–169 (2000). https://doi.org/10.1023/A:1009804919645
26. Okumoto, K., Fukunaga, T., Nagamochi, H.: Divide-and-conquer algorithms for partitioning hypergraphs and submodular systems. In: Dong, Y., Du, D.Z., Ibarra, O. (eds.) Algorithms and Computation, pp. 55–64. Springer, Berlin (2009)
27. Okumoto, K., Fukunaga, T., Nagamochi, H.: Divide-and-conquer algorithms for partitioning hypergraphs and submodular systems. Algorithmica (2010). https://doi.org/10.1007/s00453-010-9483-0
28. Queyranne, M.: Minimizing symmetric submodular functions. Math. Program. **82**(1), 3–12 (1998). https://doi.org/10.1007/BF01585863
29. Thorup, M.: Minimum k-way cuts via deterministic greedy tree packing. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, pp. 159–165 (2008). https://doi.org/10.1145/1374376.1374402
30. Vazirani, V.V., Yannakakis, M.: Suboptimal cuts: Their enumeration, weight and number. In: International Colloquium on Automata, Languages, and Programming, pp. 366–377. Springer (1992)
31. Xiao, M.: An improved divide-and-conquer algorithm for finding all minimum k-way cuts. In: Hong, S.H., Nagamochi, H., Fukunaga, T. (eds.) Algorithms and Computation, pp. 208–219. Springer, Berlin (2008)
32. Xiao, M.: Finding minimum 3-way cuts in hypergraphs. Inf. Process. Lett. **110**(14–15), 554–558 (2010). https://doi.org/10.1016/j.ipl.2010.05.003
33. Yeh, L.P., Wang, B.F., Su, H.H.: Efficient algorithms for the problems of enumerating cuts by non-decreasing weights. Algorithmica **56**(3), 297–312 (2010). https://doi.org/10.1007/s00453-009-9284-5
34. Zhao, L., Nagamochi, H., Ibaraki, T.: On generalized greedy splitting algorithms for multiway partition problems. Discrete Appl. Math. **143**(1), 130–143 (2004). https://doi.org/10.1016/j.dam.2003.10.007
35. Zhao, L., Nagamochi, H., Ibaraki, T.: Greedy splitting algorithms for approximating multiway partition problems. Math. Program. **102**(1), 167–183 (2005). https://doi.org/10.1007/s10107-004-0510-2