Math. Program., Ser. B (2008) 112:45–64
DOI 10.1007/s10107-006-0084-2

FULL LENGTH PAPER

Submodular function minimization

Satoru Iwata

Received: 5 February 2006 / Accepted: 2 June 2006 / Published online: 25 January 2007

Abstract Submodular functions often arise in various fields of operations research including discrete optimization, game theory, queueing theory and information theory. In this survey paper, we give overview on the fundamental properties of submodular functions and recent algorithmic devolopments of their minimization.

Keywords Submodular function · Discrete optimization · Algorithm

Mathematics Subject Classification (2000) 90C27

1 Introduction

Let V be a finite set. A set function $f : 2^V \to \mathbf{R}$ is said to be submodular if it satisfies

$$f(X) + f(Y) \geq f(X \cup Y) + f(X \cap Y), \qquad \forall X, Y \subseteq V.$$

Submodular function minimization is to compute the minimum value as well as a minimizer of a submodular function f, provided that an oracle for evaluating the function value $f(X)$ for $X \subseteq V$ is available. A set fucntion f is supermodular if $-f$ is submodular. A set function that is both submodular and supermodular is called a modular function.

Submodular functions arise in discrete optimization [41,68] and various other fields of operations research such as game theory [74], information theory [28]

S. Iwata (✉)
Research Institute for Mathematical Sciences, Kyoto University,
Kyoto 606-8502, Japan
e-mail: iwata@kurims.kyoto-u.ac.jp

and queueing theory [19,73]. Examples include the cut capacity functions of networks, the rank functions of matroids, and the entropy functions of multiple information sources. Submodular functions play important roles in statistical physics as well [1,2].

Submodular functions are discrete analogue of convex functions. This analogy was exhibited by the discrete separation theorem of Frank [25] and the Fenchel-type duality theorem of Fujishige [30]. A more direct connection was established by Lovász [52], who clarified that the submodularity of a set function can be characterized by the convexity of a continuous function obtained by extending the set function in an appropriate manner. This observation together with valuated matroids invented by Dress and Wenzel [15] motivated Murota [58–60] to develop the theory of discrete convex analysis.

The first polynomial algorithm for submodular function minimization is due to Grötschel et al. [39]. The strongly polynomial version was also developed by Grötschel et al. [40]. These algorithms employ the ellipsoid method, which was used by Khachiyan [50] to develop the first polynomial-time algorithm for linear programming. In spite of its polynomial time complexity, the ellipsoid method is not so efficient in practice.

Cunningham [11] develpoed a combinatorial strongly polynomial algorithm for solving the membership problem for matroid polyhedra, which is a special case of submodular function minimization. Then Cunningham [12] extended this method to compute the minimum value of a general submodular function in pseudopolynomial time.

Recently, combinatorial strongly polynomial algorithms have been developed by Iwata et al.(IFF) [46] and by Schrijver [71]. Both of these algorithms build on works of Cunningham [11,12]. The IFF algorithm employs a scaling scheme developed in capacity scaling algorithms for the submodular flow problem [24,43,47]. In contrast, Schrijver [71] directly achieves a strongly polynomial bound by introducing a novel subroutine in framework of lexicographic augmentation. Subsequently, Fleischer and Iwata [22,23] have described a push/relabel algorithm using Schrijver's subroutine to improve the running time bound. Combining the scaling scheme with the push/relabel technique yields a faster combinatorial algorithm [45], which currently achieves the best running time bound for general submodular function minimization.

All of these combinatorial algorithms perform multiplications and divisions, although the problem of submodular function minimization does not involve these arithmetic operations. Schrijver [71] has asked if one can minimize a submodular function in strongly polynomial time using only additions, subtractions, comparisons, and the oracle calls for function values. It turns out that the IFF strongly polynomial algorithm can be converted to such a fully combinatorial algorithm [44].

This paper provides a survey of these recent developments on submodular function minimization. Section 2 exhibits examples of submodular functions and related minimization problems. Section 3 is an introduction to the polyhedral approach to submodular functions. It describes the greedy algorithm and the connection between submodularity and convexity. In Sect. 4, we expound a

general framework that are commonly used by the combinatorial algorithms for submodular function minimization. In Sects. 5, 6, 7, 8, 9, we describe combinatorial algorithms for submodular function minimization. Then Sect. 10 is devoted to Queyranne's algorithm for symmetric submodular function minimization. We also describe an extension of the IFF scaling algorithm to bisubmodular function minimization in Sect. 11. Finally, Sect. 12 provides some open problems.

Other surveys on submodular function minimization have been given by Fleischer [21], Fujishige [33], and McCormick [53]. The readers are also referred to related chapters of Fujishige [34], Korte and Vygen [51], Murota [60], and Schrijver [72].

Throughout this paper, let $\mathbf{R}^V$ denote the set of all the real valued functions $x : V \to \mathbf{R}$, which forms a linear space of dimension $n = |V|$. We identify a vector $x \in \mathbf{R}^V$ with a modular function defined by $x(Y) = \sum_{v \in Y} x(v)$.

2 Examples of submodular functions

In this section, we describe four examples of submodular functions. The first two come from discrete mathematics, while the others are taken from queueing theory and information theory.

Matroids

The concept of matroids was introduced by Whitney [77] as a combinatorial abstraction of linear independence. Let V be a finite set and $\mathcal{I}$ be a family of subsets of V. A pair $(V, \mathcal{I})$ is a matroid if it satisfies a certain system of axioms. The rank function ρ of a matroid is defined by $\rho(X) = \max\{|J| \mid J \subseteq X, J \in \mathcal{I}\}$. Then ρ is a monotone nondecreasing submodular function that satisfies $\rho(\emptyset) = 0$ and $\rho(X) \leq |X|$ for $X \subseteq V$. Conversely, such a set function defines a matroid by $\mathcal{I} = \{J \mid J \subseteq V,\ \rho(J) = |J|\}$.

The convex hull of the characteristic vectors of the independent sets in $\mathbf{R}^V$ coincides with

$$\mathrm{MP}(\rho) = \{z \mid z \in {\mathbf{R}_+}^{\mathrm{V}},\ \forall X \subseteq V,\ z(X) \leq \rho(X)\},$$

which is called the matroid polyhedron. Testing if a given vector $z \in {\mathbf{R}_+}^{\mathrm{V}}$ is in $\mathrm{MP}(\rho)$ can be reduced to minimizing the submodular function $f(X) = \rho(X) - z(X)$. Cunningham [11] presented a combinatorial strongly polynomial algorithm for this special type of submodular function minimization.

Connected detachment

Let $G = (V, E)$ be a connected graph with vertex set V and edge E. Consider a function $b : V \to \mathbf{Z}_+$. A b-detachment of G is a new graph $G' = (W, E)$

obtained by splitting each vertex $v \in V$ into $b(v)$ vertices. Each edge $e \in E$ incident to $v \in V$ in G should be incident in G' to one of the $b(v)$ vertices that come from v. For any $X \subseteq V$, let $e(X)$ denote the number of edges incident to X. We also denote by $c(G \setminus X)$ the number of connected components in the graph obtained from G by deleting the vertices in X. Nash-Williams [64] found the following theorem on the existence of a connected b-detachment.

Theorem 1 (Nash-Williams [64]) *There exists a connected b-detachment of* $G = (V, E)$ *if and only if*

$$b(X) \leq e(X) - c(G \setminus X) + 1 \tag{1}$$

holds for any $X \subseteq V$.

Let $f(X)$ denote the right-hand side of (1). Then we have $f(\emptyset) = 0$ and $f(V) = |E| + 1$. Furthermore, it can be shown that f is a submodular function. Theorem 1 suggests that one can check the existence of a connected b-detachment by minimizing the submodular function $f(X) - b(X)$.

The original proof was based on the matroid intersection theorem. Simple alternative proofs have been given to this theorem [65–67]. The submodularity of f plays a crucial role in the one that uses orientations [66].

Detachments with higher edge-connectivity requirements have recently been investigated by Fleiner [20] and by Jordán and Szigeti [49]. See Frank [27,26] for other interesting applications of submodular functions in graph theory.

Multiclass queueing systems

Consider a queueing system which deals with various types of jobs. Each job of different classes waits in different queues and the server chooses the job to serve the next by a control policy. One of the most fundamental models of this type is the so-called preemptive M/M/1, where the arrival interval and service time of each class of jobs follow exponential distributions and the premption is allowed in its control policy.

If the average arrival rates and the average service rates of the job classes are given, the performance of the system depends only on the control policy. Let V be the set of job classes. The region of performance-measuring vectors in $\mathbf{R}^V$ achieved by all control policies is called the achievable region. The performance of a multiclass M/M/1 is often measured by the average staying time vector $s \in \mathbf{R}^V$. If the preemption is allowed, the performance region of the staying time vector is explicitly given as follows.

Theorem 2 (Coffman and Mitrani [8]) *For each job* $j \in V$, *let* λ_j *and* μ_j *be the average arrival rates and the average service rates, respectively. Suppose that the utilization* $\rho_j = \lambda_j/\mu_j$ *satisfies* $\sum_{j \in V} \rho_j < 1$. *Then the achievable region of the average staying time vector is the set of vectors* $s \in \mathbf{R}^V$ *that satisfy*

$$\sum_{j\in X}\rho_j s_j \geq \frac{\sum_{j\in X}\frac{\rho_j}{\mu_j}}{1-\sum_{j\in X}\rho_j} \tag{2}$$

for every $X \subseteq V$.

The right hand side of (2) can be written as $f(X) = y(X)h(x(X))$, where $x(j) := \rho_j$, $y(j) := \dfrac{\rho_j}{\mu_j}$ and $h(x) = \dfrac{1}{1-x}$. If we assign $z(j) := \rho_j s_j$, the problem of checking the achievability of a given vector s is reduced to minimizing a set function f defined by

$$f(X) = z(X) - y(X)h(x(X)).$$

Since h is a monotone nondecreasing convex function, one can verify that f is a submodular function.

A recent paper [42] presents an efficient algorithm for minimizing this type of submodular functions in $O(n^2)$ time. The algorithm utilizes the topological sweeping method of Edelsbrunner and Guibas [16] for line arrangements in the plane.

Apart from the multiclass preemptive M/M/1, submodular functions often arise in the analysis of achievable regions of various types of multiclass queueing systems [3,19,73].

Entropy functions

Let V be a set of discrete memoryless information sources (random variables). For each nonempty subset X of V, let $h(X)$ denote the Shannon entropy of the corresponding joint distrubution. In addition, we assign $h(\emptyset) = 0$. Then the set function h is a submodular function, which follows from the nonnegativity of conditional mutual information.

Consider the situation that we encode data generated by this set of sources. Each source has its encoder, which compresses each data and transmits the code to the central decoder, which decodes all the codes it receives. We call the rate vector $R \in \mathbf{R}^V$ achievable if there exists a coding method of rate R with arbitraily small error probability. The following theorem of Slepian and Wolf [75] suggests that one can exploit the correlation among the sources to reduce the total rate required for the transmission. See also Cover [9] and Cover and Thomas [10, Sect.14.4].

Theorem 3 (Slepian and Wolf [75]) *The rate vector R is achievable if and only if*

$$R(X) > h(V) - h(V \setminus X) \tag{3}$$

holds for any nonempty $X \subseteq V$.

Note that the right-hand side of (3) is a supermodular function. Theorem 3 implies that one can check if a specified rate vector R is achievable by minimizing a submodular function $R(X) - h(V) + h(V \setminus X)$. The rate vector R is achievable if and only if the empty set is the only minimizer. The only known method to do this is to apply an algorithm for general submodular function minimization.

Let K be a positive definite symmetric matrix whose row/column set is indexed by V. For each $X \subseteq V$, let $K[X]$ denote the principal submatrix of K indexed by X. The set function f defined by $f(\emptyset) = 0$ and $f(X) = \log\det K[X]$ for nonempty X is a submodular function. The submodularity of this function f, known as Ky Fan's inequality, is a refinement of Hadamard's inequality. It can be interpreted as the submodularity of the entropy function of a multivariate normal distribution with covariance matrix K.

3 Greedy algorithm and discrete convexity

For a submodular function f with $f(\emptyset) = 0$, we consider the submodular polyhedron $\mathrm{P}(f)$ and the base polyhedron $\mathrm{B}(f)$ defined by

$$\mathrm{P}(f) = \{x \mid x \in \mathbf{R}^V, \forall Y \subseteq V, x(Y) \leq f(Y)\},$$
$$\mathrm{B}(f) = \{x \mid x \in \mathrm{P}(f), x(V) = f(V)\}.$$

A vector in $\mathrm{B}(f)$ is called a base. In particular, an extreme point of $\mathrm{B}(f)$ is called an extreme base. The base polyhedron $\mathrm{B}(f)$ is the set of maximal vectors in $\mathrm{P}(f)$.

An extreme base can be computed by the greedy algorithm of Edmonds [17] and Shapley [74] as follows.

Let $L = (v_1, \ldots, v_n)$ be a linear ordering of V. For any $v_j \in V$, we denote $L(v_j) = \{v_1, \ldots, v_j\}$. The greedy algorithm with respect to L generates an extreme base $y \in \mathrm{B}(f)$ by

$$y(u) := f(L(u)) - f(L(u) \setminus \{u\}). \tag{4}$$

Conversely, any extreme base can be obtained in this way with an appropriate linear ordering.

Given a nonnegative vector $p \in {\mathbf{R}_+}^V$, consider a linear ordering $L = (v_1, \ldots, v_n)$ such that $p(v_1) \geq p(v_2) \geq \cdots \geq p(v_n)$. The greedy algorithm with respect to L yields an optimal solution to the problem of maximizing the inner product $\langle p, x\rangle = \sum_{v \in V} p(v)x(v)$ in $\mathrm{P}(f)$.

Let $p_1 > p_2 > \cdots > p_k$ be the distinct values of p. For $j = 1, \ldots, k$, we denote $U_j = \{v \mid p(v) \geq p_j\}$. Then p can be expressed as

$$p = \sum_{j=1}^{k} q_j \chi_{U_j},$$

with $q_j = p_j - p_{j+1}$ for $j = 1, \ldots, k-1$ and $q_k = p_k \geq 0$. We now define $\hat{f}(p)$ by

$$\hat{f}(p) = \sum_{j=1}^{k} q_j f(U_j).$$

Then the function $\hat{f}$ satisfies

$$\hat{f}(p) = \max\{\langle p, x\rangle \mid x \in \mathrm{P}(f)\}, \tag{5}$$

which follows from the validity of the greedy algorithm.

Note that the above definition of $\hat{f}$ is free from the submodularity of f. For a set function f in general, we define $\hat{f}$ in the same way. Then $\hat{f}(\chi_X) = f(X)$ holds for any $X \subseteq V$. Hence we may regard $\hat{f}$ as an extension of f.

The restriction of $\hat{f}$ to the hypercube $[0,1]^V$ can be interpreted as follows. A linear ordering L corresponds to the simplex whose extreme points are given by the characteristic vectors of $L(v)$ for $v \in V$ and the empty set. Since there are $n!$ linear orderings of V, the hypercube $[0,1]^V$ can be partitioned into $n!$ congruent simplices obtained by this way. Determine the function values of $\hat{f}$ in each simplex by the linear interpolation of the values at the extreme points. The resulting function $\hat{f}$ is a continuous function on the hypercube.

The following theorem provides a connection between submodularity and convexity.

Theorem 4 (Lovász [52]) *A set function f is submodular if and only if $\hat{f}$ is convex.*

Proof If f is a submodular function, then it follows from (5) that $\hat{f}$ is a convex function. Conversely, if $\hat{f}$ is convex, then we have

$$\hat{f}(\chi_X + \chi_Y) \leq \hat{f}(\chi_X) + \hat{f}(\chi_Y) = f(X) + f(Y).$$

On the other hand, it follows from the definition of $\hat{f}$ that

$$\hat{f}(\chi_X + \chi_Y) = \hat{f}(\chi_{X\cap Y}) + \hat{f}(\chi_{X\cup Y}) = f(X \cap Y) + f(X \cup Y)$$

holds for any $X, Y \subseteq V$. Thus f is a submodular function. □

4 Min-max theorem

For any vector $x \in \mathbf{R}^V$, we denote $x^-(v) := \min\{x(v), 0\}$. The following min-max theorem plays a central role in combinatorial algorithms for submodular function minimization.

Theorem 5 (Edmonds [17]) *For a submodular function f with $f(\emptyset) = 0$, we have*

$$\min_{X \subseteq V} f(X) = \max\{z(V) \mid z \in \mathrm{P}(f),\ z \leq 0\}$$
$$= \max\{x^-(V) \mid x \in \mathrm{B}(f)\}.$$

Moreover, if f is an integer valued function, then the maximum in the right-hand side is attained by an integer verctor z.

Proof If $z \in \mathrm{P}(f)$ and $z \leq 0$, then $z(V) \leq z(X) \leq f(X)$ holds for any $X \subseteq V$. Consider a set function $f^\circ : 2^V \to \mathbf{R}$ defined by

$$f^\circ(X) = \min_{Y \subseteq X} f(Y).$$

Then f° is submodular, and $\mathrm{P}(f^\circ) \subseteq \mathrm{P}(f)$ holds. Any base $z \in \mathrm{B}(f^\circ)$ satisfies $z \leq 0$ and $z(V) = f^\circ(V) = \min_{X \subseteq V} f(X)$, which imply the first equality. The second equality follows from the existence of a base $x \in \mathrm{B}(f)$ with $x \geq z$. Moreover, if f is integer valued, then so is f°, which implies that an extreme base $z \in \mathrm{B}(f^\circ)$ is an integer vector. □

Theorem 5 seems to provide a good characterization of the minimum value of f. In fact, if we have a pair of $W \subseteq V$ and $x \in \mathrm{B}(f)$ with $f(W) = x^-(V)$, then it follows from Theorem 5 that W attains the minimum value of f. This suggests a natural way to find the minimum by moving $x \in \mathrm{B}(f)$ so that $x^-(V)$ increases. However, it is not easy to verify that the vector x in our hand stays in $\mathrm{B}(f)$. A direct way to check this by the definition requires an exponential number of steps. On the other hand, an extreme base y of $\mathrm{B}(f)$ can be verified by a linear ordering of V generating y. According to Caratheodory's theorem, an arbitrary point in a bounded polyhedron can be expessed as a convex combination of its extreme points. Keeping $x \in \mathrm{B}(f)$ as a convex combination $x = \sum_{i \in I} \lambda_i y_i$ of extreme bases y_i, we are able to verify $x \in \mathrm{B}(f)$ efficiently, provided that I is not too large. A base $x \in \mathrm{B}(f)$ expressed by this way provides a compact certificate of $f(W)$ being the minimum value if $x^-(V) = f(W)$ holds.

This approach was introduced by Cunningham [11] in the separation problem for matroid polyhedra. Bixby et al. [4] employed this approach to develop a combinatorial algorithm for minimizing a submodular function by a finite number of steps. Furthermore, Cunningham [12] improved this algorithm to the first combinatorial pseudopolynomial algorithm for computing the minimum value of an integer valued submodular function. In general, a pseudopolynomial algorithm runs in time polynomial in the number of inputs and the maximum absolute value of the inputs. The running time bound of Cunningham's algorithm is $O(n^6 \gamma M \log nM)$, where γ is the time required for computing the function value and M is the maximum absolute value of f.

Since the dimension of a base polyhedron is at most $n - 1$, it follows from Caratheodory's theorem that any base $x \in \mathrm{B}(f)$ can be expressed as a convex combination of at most n extreme bases. When the set I becomes large, we are

able to reduce $|I|$ to at most n as follows. Consider a $V \times I$ matrix that consists of extreme bases y_i for $i \in I$. Let H be a matrix obtained by attaching a row with all components being one to this matrix. Applying the Gaussian elimination by row transformations to H, detect a linear dependence $\sum_{i\in I} \mu_i y_i = 0$, $\sum_{i\in I} \mu_i = 0$. Compute $\theta = \max\{\lambda_i/\mu_i \mid \mu_i > 0\}$ and update $\lambda_i := \lambda_i - \theta\mu_i$ for each $i \in I$. At least one index $i \in I$ will satisfy $\lambda_i = 0$, and then delete such i from I. Repeat this process until H becomes lenearly independent. This process will be referred to as $\mathsf{Reduce}(x, I)$.

5 Schrijver's algorithm

In this section, we describe the combinatorial strongly polynomial algorithm of Schrijver [71] for submodular function minimization. Following the framework of Cunningham [11,12], the algorithm keeps a base $x \in \mathrm{B}(f)$ as a convex combination $x = \sum_{i\in I} \lambda_i y_i$ of extreme bases $y_i \in \mathrm{B}(f)$ generated by linear orderings L_i. We denote $u \prec_i v$ if u precedes v in L_i. We also denote $u \preceq_i v$ to mean $u \prec_i v$ or $u = v$.

Consider an auxiliary graph $G_I = (V, A_I)$ with the vertex set V and the arc set $A_I = \{(u, v) \mid \exists i \in I, v \preceq_i u\}$. Let S and T be the vertex subsets defined by $S = \{v \mid v \in V, x(v) < 0\}$ and $T = \{v \mid v \in V, x(v) > 0\}$. If there is no directed path from S to T in G_I, let W be the set of vertices reachable from S. Then $y_i(W) = f(W)$ holds for each $i \in I$, and so does $x(W) = f(W)$. Since $S \subseteq W \subseteq V \setminus T$, we have $x^-(V) = x(W) = f(W)$, which implies that $f(W)$ attains the minimum value of f.

On the other hand, if there is a directed path from S to T in G_I, let $d(v)$ be the smallest number of arcs in a directed path from S to v. Let t be the vertex in T with maximumm $d(t)$. If two or more vertices attain the maximum, select the maximum one in a fixed total order $\leq$ on V. Let u be the maximum vertex in $\leq$ such that $(u, t) \in A_I$ and $d(u) = d(t) - 1$ hold. Furthermore, select $i \in I$ with maximum $|L_i(u) \setminus L_i(t)|$. To the triple $i \in I, u \in V, t \in T$ thus obtained, the algorithm applies the following procedure $\mathsf{Interval}(i, u, t)$.

For each vertex v in the interval $L_i(t, u) = L_i(u) \setminus L_i(t)$, let L_i^{tv} denote the linear ordering obtained from L_i by moving v to the place immediately before t. We also denote by y_i^{tv} the extreme base generated by the greedy algorithm with the linear ordering L_i^{tv}. For each $v \in L_i(t, u)$ and $w \in V$, assign $H_{wv} = y_i^{tv}(w) - y_i(w)$. Solve a linear system of equations

$$\sum_{v\in L_i(t,u)} H_{wv}\xi_v = \chi_u - \chi_t$$

to obtain ξ_v for $v \in L_i(t, u)$. It follows from the submodularity of f that $H_{vv} \geq 0$ for each $v \in L_i(t, u)$ and that $H_{wv} \leq 0$ if $t \preceq_i w \prec_i v \preceq_i u$. Thus the coefficient matrix H is in the form of

$$H = \begin{array}{c} \\ t \\ \\ \\ u \end{array} \begin{array}{c} \begin{array}{cccc} & & & u \end{array} \\ \begin{pmatrix} - & - & \cdots & - \\ + & - & \cdots & - \\ 0 & + & \ddots & \vdots \\ \vdots & \ddots & \ddots & - \\ 0 & \cdots & 0 & + \end{pmatrix} \end{array}.$$

Note that the column sum is equal to zero. Because of the triangular form of H, we obtain $\xi_v \geq 0$. Compute $\alpha = \min\{x(u), \lambda_i/\xi\}$, where $\xi = \sum_v \xi_v$, and move $x \in \mathrm{B}(f)$ to $x := x + \alpha(\chi_u - \chi_t)$. For each $v \in L_i(t,u)$, add a new index i_v to I with $L_{i_v} := L_i^{uv}$ and $\lambda_{i_v} := \alpha\xi_v$. Furthermore, update $\lambda_i := \lambda_i - \alpha\xi$. Then the new base $x \in \mathrm{B}(f)$ can be represented as the convex combination $x = \sum_{i \in I} \lambda_i y_i$ again. As a consequence, $|I|$ increases by $|L_i(t,u)|$. The algorithm applies $\mathsf{Reduce}(x, I)$ to reduce $|I|$ to at most n.

Schrijver [71] showed that the algorithm terminates after $O(n^6)$ iterations of this process. Since one execution of $\mathsf{Interval}$ takes $O(n^2\gamma + n^3)$ time, the total running time bound is $O(n^8\gamma + n^9)$. Employing the push/relabel framework introduced by Goldberg and Tarjan [38] for the maximum flow problem, Fleischer and Iwata [22,23] devised an improved variant that runs in $O(n^7\gamma + n^8)$ time. Then Vygen [76] refined the complexity analysis of Schrijver's algorithm to show that it runs in $O(n^7\gamma + n^8)$ time.

The push/relabel framework was also employed by Fujishige and Zhang [36] to devise an algorithm for the intersection problem on a pair of submodular polyhedra. Analogously to the result of Gallo et al. [37] on parametric maximum flow problems, Iwata et al. [48] extended this algorithm to the parametic intersection problem for strong map sequences without expense of running time bound. Fleischer and Iwata [23] also obtained a similar result for parametric submodular function minimization, which led to an algorithm to find the lexicographically optimal base [29] in $O(n^7\gamma + n^8)$ time.

6 A scaling algorithm

This section is devoted to the scaling algorithm of Iwata et al. [46] for minimizing an integer-valued submodular function. The scaling technique is a generic method of making pseudopolynomial algorithms run in polynomial time. It was introduced by Edmonds and Karp [18] so as to develop the first polynomial algorithm for the minimum cost flow problem.

The scaling algorithm uses a parameter $\delta > 0$. It starts with an arbitrary linear ordering L and an extreme base $x \in \mathrm{B}(f)$ generated by L. The initial value of δ is given by $\delta := \min\{|x^-(V)|, x^+(V)\}/n^2$. In each scaling phase, the algorithm cuts the value of δ in half. Finally, the algorithm terminates when $\delta < 1/n^2$. The algorithm expresses $x \in \mathrm{B}(f)$ as a convex combination $\sum_{i \in I} \lambda_i y_i$ of extreme bases y_i generated by L_i. The initial setting is $I = \{0\}$, $y_0 = x$, $L_0 = L$, $\lambda_0 = 1$.

Since the initial value of δ satisfies $\delta < M/n^2$, the algorithm performs $O(\log M)$ scaling phases.

The key idea of this scaling algorithm is to consider a flow in the complete directed graph on vertex set V. The flow is represented as a skew-symmetric function $\varphi : V \times V \to \mathbf{R}$, which satisfies $\varphi(u,v) + \varphi(v,u) = 0$ for each (u,v). The arc capacity is δ for each arc. Hence the cut capacity function is given by $\kappa_\delta(X) = \delta\,|X| \cdot |V \setminus X|$. A flow φ is said to be feasible if $-\delta \le \varphi(u,v) \le \delta$ holds for each arc (u,v). The boundary $\partial\varphi$ of φ is defined by

$$\partial\varphi(u) = \sum_{v \in V} \varphi(u,v).$$

Then we have $\partial\varphi \in \mathrm{B}(\kappa_\delta)$.

In each scaling phase, the algorithm keeps $z = x + \partial\varphi \in \mathrm{B}(f + \kappa_\delta)$ and aims at increasing $z^-(V)$ insetad of $x^-(V)$. For a feasible flow φ, construct an auxliary graph $G_\varphi = (V, E_\varphi)$ with arc set $E_\varphi = \{(u,v) \mid u \neq v,\ \varphi(u,v) \le 0\}$. A directed path from $S = \{v \mid v \in V, z(v) \le -\delta\}$ to $T = \{v \mid v \in V, z(v) \ge \delta\}$ in G_φ is called an augmenting path. If there exists an augmenting path P, the algorithm augment the flow φ through P by δ, namely $\varphi(u,v) := \varphi(u,v) + \delta$ and $\varphi(v,u) := \varphi(v,u) - \delta$ for each $(u,v) \in P$. This procedure is written as Augment(φ, P). As a result of Augment(φ, P), $z^-(V)$ increases by δ.

If there is no augmenting path, let W be the set of vertices reachable from S in G_φ. Then the algorithm finds a triple of $i \in I$, $u \in W$, $v \in V \setminus W$ such that v immediately precedes u in L_i. Such a triple is called admissible. For an admissible triple (i,u,v), the algorithm performs Double-Exchange(i,u,v) described as follows.

The procedure Double-Exchange(i,u,v) interchanges u and v in L_i. Then y_i generated by L_i moves to an adjacent extreme base in $\mathrm{B}(f)$. This move can be written as $y_i := y_i + \beta(\chi_u - \chi_v)$, where β is given by $\beta = f(L_i(u) \setminus \{v\}) - y(L_i(u) \setminus \{v\})$. If we keep the coefficients of the convex combination as before, x moves to $x := x + \lambda_i\beta(\chi_u - \chi_v)$. We want to keep z invariant by changing $\varphi(u,v)$ and $\varphi(v,u)$. However, if $\lambda_i\beta$ is too large, this would violate the feasibility of φ. Instead, we determine the step size α of x by $\alpha = \min\{\lambda_i\beta, \varphi(u,v)\}$, namely $x := x + \alpha(\chi_u - \chi_v)$, $\varphi(u,v) := \varphi(u,v) - \alpha$, and $\varphi(v,u) := \varphi(v,u) + \alpha$. If $\alpha < \lambda_i\beta$, we add a new index k to I with L_k and y_k being the previous L_i and y_i. By partitioning the coefficient λ_i into $\lambda_k := \lambda_i - \alpha/\beta$ and $\lambda_i = \alpha/\beta$, we keep the relation $x = \sum_{j \in I} \lambda_j y_j$. Since v becomes reachable from S in G_φ, the set W of reachable vertices gets enlarged. This can happen at most n times before an augmenting path is found. We keep $|I| \le 2n$ by applying Reduce(x, I) each time the algorithm performs Augment.

If there is neither an augmenting path nor an admissble triple, we have $y_i(W) = f(W)$ for each $i \in I$ and hence $x(W) = f(W)$. Since $\partial\varphi(W) \ge 0$ and $S \subseteq W \subseteq V \setminus W$, this implies $z^-(V) \ge f(W) - n\delta$ and $x^-(V) \ge f(W) - \delta/n^2$. Then the algorithm terminates the current scaling phase by cutting the values of δ and φ in half and goes to the new scaling phase. Note that this maintains the

feasibility of φ. Since $z^-(V) \leq f(X) + n^2\delta/4$ holds for any $X \subseteq V$, the number of executions of Augment as well as Reduce in each scaling phase is $O(n^2)$.

The procedure Double-Exchange(i, u, v) interchanges $u \in W$ and $v \in V \setminus W$ in L_i. Then the triple (i, u, v) will never become admissible again before an execution of Augment. Therefore, the number of executions of Double-Exchange is bounded by the number of triples, which is $O(n^3)$. Since Double-Exchange (i, u, v) can be performed in $O(\gamma)$ time, each scaling phase requires $O(n^5\gamma)$ time. Thus the total running time of the scaling algorithm is $O(n^5\gamma \log M)$.

Finally, at the end of the last scaling phase with $\delta < 1/n^2$, we have $x^-(V) \geq f(W) - n^2\delta > f(W) - 1$. Since $x^-(V) \leq f(X)$ for any $X \subseteq V$, it follows from the integrality of f that W is a minimizer of f.

7 A faster scaling algorithm

Incorporating the push/relabel framework improves the running time of the scaling algorithm. The resulting algorithms achieve the currently best bounds among combinatorial algorithms for submodular function minimization [45].

The new algorithm keeps a valid labeling d in each scaling phase. A labeling $d : V \to \mathbf{Z}$ is valid if $d(u) = 0$ for $u \in S$ and $v \preceq_i u$ implies $d(v) \leq d(u) + 1$. A valid labeling $d(v)$ serves as a lower bound on the number of arcs from S to v in the directed graph $G_I = (V, A_I)$ with the arc set $A_I = \{(u, v) \mid \exists i \in I,\ v \preceq_i u\}$.

Instead of Double-Exchange, the new algorithm performs Multiple-Exchange described below. Suppose there is no augmenting path in $G_\varphi = (V, E_\varphi)$. Let W be the set of vertices reachable from S in G_φ. Let Z be the set of vertices that attains the minimum labeling in $V \setminus W$. A triple (i, u, v) is called active if v is the first vertex of Z in L_i and u is the last vertex in L_i with $v \preceq_i u$ and $d(v) = d(u) + 1$. The procedure Multiple-Exchange(i, u, v) is applicable to an active triple (i, u, v).

For an active triple (i, u, v), the set of vertices from v to u in L_i is called an *active interval*. The active interval is divided into $Q = \{w \mid w \in W,\ v \prec_i w \preceq_i u\}$ and $R = \{w \mid w \in V \setminus W,\ v \preceq_i w \prec_i u\}$.

The procedure Multiple-Exchange(i, u, v) moves the vertices in R to the place immediately after u in L_i, without changing the ordering in Q and in R. Then it computes an extreme base y_i generated by the new L_i. This results in $y_i(q) \geq y_i^\circ(q)$ for $q \in Q$ and $y_i(r) \leq y_i^\circ(r)$ for $r \in R$, where y_i° denotes the previous y_i.

Consider a complete bipartite graph with the vertex sets Q and R. The algorithm finds a flow $\xi : Q \times R \to \mathbf{R}_+$ such that $\sum_{r \in R} \xi(q, r) = y_i(q) - y_i^\circ(q)$ for each $q \in Q$ and $\sum_{q \in Q} \xi(q, r) = y_i^\circ(r) - y_i(r)$ for each $r \in R$. Such a flow can be obtained easily by the so-called northwest corner rule. Then the procedure computes $\alpha = \min\{\lambda_i, \delta/\beta\}$ with $\beta = \max\{\xi(q, r) \mid q \in Q, r \in R\}$, and moves x to $x := x + \alpha(y_i - y_i^\circ)$. In order to keep z invariant, the procedure adjusts the flow φ by $\varphi(q, r) := \varphi(q, r) - \alpha\xi(q, r)$ and $\varphi(r, q) := \varphi(r, q) + \alpha\xi(q, r)$ for every $(q, r) \in Q \times R$. The resulting φ satisfies the capacity constraints due to the choice of α, and the vertices in W remain reachable from S in G_φ.

If $\alpha < \lambda_i$, a new index k is added to I. The associated linear ordering L_k is the previous L_i. The coefficient λ_k is determined by $\lambda_k := \lambda_i - \alpha$, and then λ_i is replaced by $\lambda_i := \alpha$. Thus the algorithm continues to keep x as a convex combination $x = \sum_{i \in I} \lambda_i y_i$.

Each scaling phase begins with setting $d(v) = 0$ for each $v \in V$. If there exists an augmenting path P in G_φ, then the algorithm performs Augment(x, P) and Reduce(x, I). Otherwise, the algorithm computes $\ell = \min\{d(v) \mid v \in V \setminus W\}$. If $\ell < n$, then the algorithm applies Multiple-Exchange(i, u, v) to an active triple (i, u, v). If there is no active triple, it applies Relabel(v), which increases $d(v)$ by one, to each $v \in Z$. Finally, if $\ell = n$, there is no directed path from S to $V \setminus W$ in G_I. Then the set X of vertices reachable from S in G_I satisfies $x(X) = f(X)$, which gives the end of the current scaling phase. The algorithm goes to the next scaling phase by cutting the value of δ in half.

The number of applications of Augment and Relabel are both $O(n^2)$ in each scaling phase. The total number of function evaluations is $O(n^2)$ in consecutive applications of Multiple-Exchange between Relabel or Augment. Thus each scaling phase takes $O(n^4\gamma + n^5)$ time. Since the algorithm performs $O(\log M)$ scaling phases, the total running time bound is $O((n^4\gamma + n^5) \log M)$.

8 A strongly polynomial scaling algorithm

This section describes a strongly polynomial algorithm for minimizing a real-valued submodular function. The algorithm keeps a directed acyclic graph $D = (U, F)$ and a subset $Z \subseteq V$. It starts with $U = V$, $F = \emptyset$, $Z = \emptyset$. The set Z represents the set of elements that turns out to be contained in any minimizer of f. The vertex set U of the directed acyclic graph D corresponds to the partition of $V \setminus Z$. For a subset $Y \subseteq U$, we denote by $\Gamma(Y)$ the union of the subsets of V represented by the vertices in U. An edge (u, v) reflects an implication that any minimizer that includes $\Gamma(\{u\})$ must include $\Gamma(\{v\})$ as well.

A submodular function $\tilde{f} : 2^U \to \mathbf{R}$ is defined by

$$\tilde{f}(Y) = \begin{cases} f(\Gamma(Y) \cup Z) - \min\{f(V), f(Z)\} & (\emptyset \neq Y \subset U) \\ 0 & (Y = \emptyset, U) \end{cases}$$

Then a minimizer of X of f can be represented as $X = \Gamma(Y) \cup Z$ by a minimizer Y of $\tilde{f}$.

For each vertex $u \in U$, we denote by $R(u)$ the set of vertices reachable from u in D. At the start of each iteration, the algorithm computes

$$\eta = \max\{\tilde{f}(R(u)) - \tilde{f}(R(u) \setminus \{u\}) \mid u \in U\} \tag{6}$$

If $\eta \leq 0$, then it turns out that either V or Z is a minimizer of f.

On the other hand, if $\eta > 0$, then the vertex $u \in U$ that attains the maximum in the right-hand side must satisfy either $\tilde{f}(R(u) \setminus \{u\}) \leq -\eta/2$ or $\tilde{f}(R(u)) > \eta/2$. In the former case, apply Fix$(\tilde{f}, \eta)$ described below to detect a vertex $w \in R(u)$

that is contained in every minimizer of $\tilde{f}$. Then the algorithm deletes w from U and adds $\Gamma(\{w\})$ to Z. In the latter case, define a submodular function $\tilde{f}_u$ by

$$\tilde{f}_u(Y) = \tilde{f}(Y \cup R(u)) - \tilde{f}(R(u)) \quad (Y \subseteq U \setminus R(u))$$

and apply $\mathsf{Fix}(\tilde{f}_u, \eta)$ to find a vertex $w \in U \setminus R(u)$ contained in every minimizer of $\tilde{f}_u$. Then the algorithm adds (u, w) to F. If the resulting graph contains a directed cycle, then the algorithm shrinks it to a new vertex.

The procedure $\mathsf{Fix}(\tilde{f}, \eta)$ is applicable to a submodular function $\tilde{f}$ such that $\tilde{f}(Y) \leq -\eta/2$ for some Y. It performs the scaling algorithm with the arc set of G_φ replaced by $E_\varphi \cup F$. If $x(w) < -m^2\eta$ holds for $m = |U|$ at the end of a scaling phase, w must be contained in every minimizer of $\tilde{f}$. The existence of Y with $\tilde{f}(Y) \leq -\eta/2$ ensures that such a vertex w must be found within $O(\log n)$ scaling phases.

Therefore, the algorithm runs in $O(n^7\gamma \log n)$ time. Replacing the scaling algorithm by the faster version, we improve this bound to $O((n^6\gamma + n^7)\log n)$, which is currently the best strongly polynomial bound among combinatorial algorithms.

9 A fully combinatorial algorithm

This section briefly explains a technique to turn the strongly polynomial scaling algorithms to a fully combinatorial one. In the design of a fully combinatorial algorithm, we are allowed to perform additions, subtractions, and comparisons. These fundamental operations, however, enable us to simulate multiplications if the multiplier is an integer bounded by a polynomial in the dimension n of the problem. We are also able to compute an integer rounding of a ratio of two numbers, provided that the answer is bounded by a polynomial in n.

Expressing a base as a convex combination of extreme bases does not seem suitable for fully combinatorial algorithms because it involves multiplications. Nevertheless, this is not a crucial obstracle if we keep the coefficients as rational numbers whose common denominators are bounded by a polynomial in n.

The first step towards a fully combinatorial implementation of the strongly polynomial scaling algorithm is to neglect Reduce. This causes growth of the number of extreme bases for convex combination. However, the number is still bounded by a polynomial in n. Since the number of extreme bases generated between augmentations is at most n, each scaling phase yields $O(n^3)$ new extreme bases. Hence the number of extreme bases through the $O(\log n)$ scaling phases is $O(n^3 \log n)$. This is in contrast to Schrijver's algorithm which may generate an exponential number of extreme bases without performing Reduce.

The next step is to choose an appropriate step length in Double-Exchange so that the coefficients should be rational numbers with a common denominator bounded by a polynomial in n. Let θ denote the value of δ in the first scaling phase. Then $\kappa = \theta/\delta$ is an integer. For each $i \in I$, we keep $\lambda_i = \mu_i/\kappa$

with an integer μ_i. In the original version, the step length α is determined by $\alpha = \min\{\lambda_i\beta, \varphi(u,v)\}$. If $\lambda_i\beta > \varphi$, the step length α can be replaced by any value that satisfies $\varphi(u,v) \leq \alpha \leq \varphi(u,v) + \delta$. Let ν be the minimum integer such that $\nu\beta > \varphi(u,v)\kappa$. Such an integer ν can be computed by binary search. Then the new coefficients λ_k and λ_i are determined by $\mu_k := \mu_i - \nu$ and $\mu_i := \nu$. Thus the coefficients are rational numbers whose common denominator is κ, which is bounded by a polynomial in n through the $O(\log n)$ scaling phases. Then it is easy to implement this algorithm using additions, subtractions, comparisons, and oracle calls for the function values. The resulting algorithm runs in $O(n^9\gamma \log^2 n)$ time.

By a similar technique, one can implement the faster scaling algorithm in a fully combinatorial manner. A straightforward implementation would result in an $O((n^8\gamma + n^9)\log^2 n)$ algorithm. McCormick [53] has suggested a more careful implementation to achieve an $O(n^8\gamma \log^2 n)$ bound.

An advantage of a fully combinatorial algorithm from theoretical point of view is not only aesthetic. Suppose we are given a vector $z \in \mathrm{P}(f)$ and a direction vector $a \in \mathbf{R}^V$. Then what is the maximum $t \in \mathbf{R}$ such that $ta + z \in \mathrm{P}(f)$? A recent paper of Nagano [63] presents the first strongly polynomial algorithm for solving this problem. The algorithm is based on the parametric search technique of Megiddo [56,57], which requires a fully combinatorial subroutine for submodular function minimization.

10 Symmetric submodular function minimization

A set function f is said to be symmetric if $f(X) = f(V \setminus X)$ holds for any $X \subseteq V$. A symmetric submodular function f satisfies $f(X) \geq f(\emptyset)$ for any $X \subseteq V$. Hence it is trivial to compute the minimum value of f. Symmetric submodular function minimization is the problem of finding the minimum value of $f(X)$ among proper nonempty subsets X.

Examples of symmetric submodular functions are the cut capacity function of an undirected network and the connectivity function of a matroid. Symmetric submodular function minimization for the cut capacity functions in undirected networks corresponds to the minimum cut problem. As a generalization of the minimum cut algorithm of Nagamochi and Ibaraki [61], Queyranne [69] presented a fully combinatorial strongly polynomial algorithm for symmetric submodular function minimization.

Queyranne's algorithm in fact deals with an arbitrary submodular function f to find a proper nonempty subset X that minimizes $f(X) + f(V \setminus X)$. It adopts a novel procedure Pendant-Pair that provides an ordering of V as follows. First, select an arbitrary element as v_1. Susequenlty, for $j = 1$ to $n-1$, given $W_j = \{v_1, \ldots, v_j\}$, the procedure selects an element $u \in V \setminus W_j$ that minimizes $f(W_j \cup \{u\}) - f(\{u\})$ as v_{j+1}. The pair (v_{n-1}, v_n) is called a pendent pair.

Theorem 6 (Queyranne [69]) *For any subset X that separates the pendent pair v_{n-1} and v_n, we have $f(X) + f(V \setminus X) \geq f(\{v_n\}) + f(V \setminus \{v_n\})$.*

Theorem 6 suggests a way to find the minimum value of $f(X)+f(V\setminus X)$. Let (u,v) be the pendant pair obtained by applying Pendant-Pair to f. Consider a submodular function f' on the ground set $V':=V\setminus\{v\}$ defined by

$$f'(X)=\begin{cases} f(X) & (u\notin X)\\ f(X\cup\{v\}) & (u\in X).\end{cases}$$

Then the mimimum value of $f(X)+f(V\setminus X)$ is equal to $f(\{v\})+f(V\setminus\{v\})$ or to the minimum value of $f'(X)+f'(V'\setminus X)$, which can be computed recursively. Thus we obtain an algorithm to find the minimum value of $f(X)+f(V\setminus X)$ by applying Pendant-Pair $O(n)$ times. Since one application of Pendant-Pair takes $O(n^2\gamma)$ time, the total running time bound is $O(n^3\gamma)$.

This algorithm is further generalized in two different directions by Nagamochi and Ibaraki [62] and by Rizzi [70].

11 Bisubmodular function minimization

A function g defined on the ordered pairs of disjoint subsets of V is said to be bisubmodular if it satisfies

$$\begin{aligned} g(X_1,Y_1)+g(X_2,Y_2)\geq{}& g(X_1\cup X_2-Y_1\cup Y_2,Y_1\cup Y_2-X_1\cup X_2)\\ &+g(X_1\cap X_2,Y_1\cap Y_2)\end{aligned}$$

for (X_1,Y_1) and (X_2,Y_2). Talking about minimization of bisubmodular function, we may assume that $g(\emptyset,\emptyset)=0$. With such a bisubmodular function g, consider a bisubmodular polyhedron $\mathrm{P}(g)$ defined by

$$\mathrm{P}(g)=\{x\mid x\in\mathbf{R}^V,\ x(X)-x(Y)\leq g(X,Y)\}.$$

For a vector $x\in\mathbf{R}^V$, we denote $\|x\|=\sum_{v\in V}x(v)$. Then the following theorem characterizes the minimum value of g.

Theorem 7 Fujishige [32] *For a bisubmodular function g with $g(\emptyset,\emptyset)=0$, we have*

$$\min\{g(X,Y)\mid X,Y\subseteq V,\ X\cap Y=\emptyset\}=\max\{-\|x\|\mid x\in\mathrm{P}(g)\}.$$

Moreover, if g is integer valued, then there is an integer vector x that attains the maximum in the right-hand side.

The scaling algorithm for submodular function minimization has been extended to bisubmodular function minimization by Fujishige and Iwata [35]. The algorithm keeps $x\in\mathrm{P}(f)$ as a convex combination $\sum_{i\in I}\lambda_i y_i$ of extreme points of $\mathrm{P}(f)$. Instead of the complete directed graph, we employ a complete bidirected graph. The resulting algorithm computes the minimum value of an integer valued bisubmodular function in $O(n^5\gamma\log M)$ time, where γ is the time

for evaluating the function value and M is the maximum absolute value of g. Subsequently, McCormick and Fujishige [54] have devised a strongly polynomial version of this algorithm.

Examples of bisubmodular functions include the rank functions of delta-matroids introduced independently by Bouchet [5] and by Chandrasekaran and Kabadi [7]. A delta-matroid is a set system $(V, \mathcal{F})$ with $\mathcal{F}$ being a nonempty family of subsets of V that satisfies the following exchange property:

$$\forall F_1, F_2 \in \mathcal{F}, \forall v \in F_1 \triangle F_2, \exists u \in F_1 \triangle F_2 : F_1 \triangle \{u, v\} \in \mathcal{F},$$

where $\triangle$ denotes the symmetric difference. A slightly restricted set system with an additional condition $\emptyset \in \mathcal{F}$ had been introduced by Dress and Havel [14]. A member of $\mathcal{F}$ is called a feasible set of the delta-matroid. Note that the base and the independent-set families of a matroid satisfy this exchange property. Thus, a delta-matroid is a generalization of a matroid.

Chandrasekaran and Kabadi [7] showed that the rank function $\varrho : 3^V \to \mathbf{Z}$ defined by

$$\varrho(X, Y) = \max\{|X \cap F| - |Y \cap F| \mid F \in \mathcal{F}\}$$

is bisubmodular. The convex hull of the characteristic vectors of the feasible sets coincides with $\mathrm{P}(\varrho)$, which is called the delta-matroid polyhedron. This fact follows from the greedy algorithm [5,7] for optimizing a linear function over the feasible sets. Given a vector $x \in \mathbf{R}^V$, one can test if x belongs to $\mathrm{P}(\varrho)$ by minimizing a bisubmodular function $g(X, Y) = \varrho(X, Y) - x(X) + x(Y)$.

The concept of delta-matroid is extended to that of jump system by Bouchet and Cunningham [6]. A jump system is a set of lattice points satisfying a certain axiom. Examples include the set of degree sequences of a graph. The lattice points contained in an integral bisubmodular polyhedron form a jump system, called a convex jump system, and conversely the convex hull of a jump system is an integral bisubmodular polyhedron. See Cunningham [13] for a survey on jump systems.

12 Conclusion

We now conclude this paper by mentioning some open problems concerning submodular function minimization.

1. An obvious one is of course to improve theoretical efficiency of submodular function minimization. In particular, the stronly polynomial bound is far from being satisfactory.
2. Fujishige [31] showed a connection between submodular funtion minimization and the minimum Euclidean norm point in the base polyhedron. A practical algorithm to solve the minimization problem based on this result is presented in Fujishige [34, Sect. 7.1 (a)]. It remains open to analyse the complexity of this algorithm.

3. What is the lower bound on the number of oracle calls for function evaluation required before determining the minimum value?

Acknowledgements The author thanks Fabian Chudak, Lisa Fleischer, Satoru Fujishige, and Tom McCormick for fruitful discussions on the topics of this paper.

References

1. Anglès d'Auriac, J.-C.: Computing the Potts free energy and submodular functions. New Optimization Algorithms in Physics. Hartmann, A.K., Rieger, H. (eds.) pp.101–117, Wiley, NewYork (2004)
2. Anglès d'Auriac, J.-C., Iglói, F., Preissmann, M., Sebő, A.: Optimal cooperation and submodularity for computing Potts' partition functions with a large number of states. J. Phys. Ser. A **35**, 6973–6983 (2002)
3. Bertsimas, D., Niño-Mora, J.: Conservation laws, extended polymatroids and multiarmed bandit problems; a polyhedral approach to indexable systems. Math. Oper. Res. **21**, 257–306 (1996)
4. Bixby, R.E., Cunningham, W.H., Topkis, D.M.: Partial order of a polymatroid extreme point. Math. Oper. Res. **10**, 367–378 (1985)
5. Bouchet, A.: Greedy algorithm and symmetric matroids. Math. Program. **38**, 147–159 (1987)
6. Bouchet, A., Cunningham, W.H.: Delta-matroids, jump systems and bisubmodular polyhedra. SIAM J. Discrete Math. **8**, 17–32 (1995)
7. Chandrasekaran, R., Kabadi, S.N.: Pseudomatroids. Discrete Math. **71**, 205–217 (1988)
8. Coffman, E.G., Jr., Mitrani, I.: A characterization of waiting time performance realizable by single-server queues. Oper. Res. **28**, 810–821 (1980)
9. Cover, T.M.: A proof of the data compression theorem of Slepian and Wolf for ergodic sources. IEEE Trans. Inform. Theory IT **21**, 226–228 (1975)
10. Cover, T.M., Thomas, J.A.: Elements of Information Theory, Wiley, Newyork (1991)
11. Cunningham, W.H.: Testing membership in matroid polyhedra. J. Combin. Theory Ser. B **36**, 161–188 (1984)
12. Cunningham, W.H.: On submodular function minimization. Combinatorica **5**, 185–192 (1985)
13. Cunningham, W.H.: Matching, matroids, and extensions. Math. Program. **91**, 515–542 (2002)
14. Dress, A.W.M., Havel, T.F.: Some combinatorial properties of discriminants in metric vector spaces. Adv. Math. **62**, 285–312 (1986)
15. Dress A.W., M., Wenzel, W.: Valuated matroids. Adv. Math. **93**, 214–250 (1992)
16. Edelsbrunner, H., Guibas, L.J.: Topologically sweeping an arrangement. J. Comput. Syst. Sci. **38**, 165–194 (1989)
17. Edmonds, J.: Submodular functions, matroids, and certain polyhedra. In: Guy, R., Hanani, H., Sauer, N., Schönheim, J. (eds.) Combinatorial Structures and Their Applications. Gordon and Breach (1970)
18. Edmonds, J., Karp, R.M.: Theoretical improvements in algorithmic efficiency for network flow problems. J. ACM **19**, 248–264 (1972)
19. Federgruen, A., Groenevelt, H.: Characterization and optimization of achievable performance in general queueing systems. Oper. Res. **36**, 733–741 (1988)
20. Fleiner, B.: Detachment of vertices of graphs preserving edge-connectivity. SIAM J. Discrete Math. **18**, 581–591 (2005)
21. Fleischer, L.: Recent progress in submodular function minimization. OPTIMA **64**, 1–11 (2000)
22. Fleischer, L., Iwata, S.: Improved algorithms for submodular function minimization and submodular flow. Proceedings of the 32nd ACM Symposium on Theory of Computing 107–116 (2000)
23. Fleischer, L., Iwata, S.: A push-relabel framework for submodular function minimization and applications to parametric optimization. Discrete Appl. Math. **131**, 311–322 (2003)
24. Fleischer L., Iwata, S., McCormick, S.T.: A faster capacity scaling algorithm for minimum cost submodular flow. Math. Programming **92**, 119–139 (2002)
25. Frank, A.: An algorithm for submodular functions on graphs. Ann. Discrete Math. **16**, 97–120 (1982)

26. Frank, A.: Submodular functions in graph theory. Discrete Math. **111**, 231–241 (1993)
27. Frank, A.:Applications of submodular functions. In: Walker, K. (ed.) Surveys in Combinatorics. pp. 85–136 Cambridge University Press, Cambridge (1993)
28. Fujishige, S.: Polymatroidal dependence structure of a set of random variables. Inform. Contr. **39**, 55–72 (1978)
29. Fujishige, S.: Lexicographically optimal base of a polymatroid with respect to a weight vector. Math. Oper. Res. **5**, 186–196 (1980)
30. Fujishige, S.: Theory of submodular programs — A Fenchel-type min-max theorem and subgradients of submodular functions. Math. Programming **29**, 142–155 (1984)
31. Fujishige, S.: Submodular systems and related topics. Math. Programming Stud. **22**, 113–131 (1984)
32. Fujishige, S.: A min-max theorem for bisubmodular polyhedra. SIAM J. Discrete Math. **10**, 294–308 (1997)
33. Fujishige, S.: Submodular function minimization and related topics. Optim. Methods Softw. **18**, 169–180 (2003)
34. Fujishige, S.: Submodular Functions and Optimization, Elsevier (2005)
35. Fujishige, S., Iwata, S.: Bisubmodular function minimization. SIAM J. Discrete Math. **19**, 1065–1073 (2006)
36. Fujishige, S., Zhang, X.: New algorithms for the intersection problem of submodular systems. Japan. J. Indust. Appl. Math. **9**, 369–382 (1992)
37. Gallo, G., Grigoriadis, M.D., Tarjan, R.E.: A fast parametric network flow algorithm and applications. SIAM J. Comput. **18**, 30–55 (1989)
38. Goldberg, A.V., Tarjan, R.E.: A new approach to the maximum flow problem. J. ACM **35**, 921–940 (1988)
39. Grötschel, M., Lovász, L., Schrijver, A.: The ellipsoid method and its consequences in combinatorial optimization. Combinatorica **1**, 169–197 (1981)
40. Grötschel, M., Lovász, L., Schrijver, A.: Geometric Algorithms and Combinatorial Optimization. Springer Heidelberg (1988)
41. Hoppe, B., Tardos, É.: The quickest transshipment problem. Math. Oper. Res. **25**, 36–62 (2000)
42. Itoko, T., Iwata, S.:Computational geometric approach to submodular function minimization for multiclass queueing systems. Technical Report METR 2005-29, University of Tokyo, October (2005)
43. Iwata, S.: A capacity scaling algorithm for convex cost submodular flows. Math. Programming **76**, 299–308 (1997)
44. Iwata, S.: A fully combinatorial algorithm for submodular function minimization. J. Combin. Theory, Ser. B **84**, 203–212 (2002)
45. Iwata, S.: A faster scaling algorithm for minimizing submodular functions. SIAM J. Comput. **32**, 833–840 (2003)
46. Iwata, S., Fleischer, L, Fujishige, S.: A combinatorial strongly polynomial algorithm for minimizing submodular functions. J. ACM **48**, 761–777 (2001)
47. Iwata, S., McCormick, S.T., Shigeno, M.: A strongly polynomial cut canceling algorithm for minimum cost submodular flow. SIAM J. Discrete Math. **19**, 304–320 (2005)
48. Iwata, S., Murota, K., Shigeno, M.: A fast parametric submodular intersection algorithm for strong map sequences. Math. Oper. Res. **22**, 803–813 (1997)
49. Jordán, T., Szigeti, Z.: Detachments preserving local edge-connectivity of graphs. SIAM J. Discrete Math. **17**, 72–87 (2003)
50. Khachiyan, L.G.: A polynomail algorithm in linear programming. Soviet Math Dokl. **20**, 191–194 (1979)
51. Korte, B., Vygen, J.: Combinatorial Optimization — Theory and Algorithms. Springer, Berlin (2000)
52. Lovász, L.: Submodular functions and convexity. Mathematical Programming — The State of the Art. Bachem A., Grötschel M., Korte B.(eds.) pp.235–257 Springer, Heidelberg (1983)
53. McCormick, S.T.: Submodular function minimization. In: Aardal, K., Nemhauser, G., Weismantel, R. (eds.) Discrete Optimization, Handbooks in Operations Research, vol. 12, Elsevier (2005)
54. McCormick, S.T., Fujishige, S.: Better algorithms for bisubmodular function minimization (2005)

55. Megiddo, N.: Optimal flows in networks with multiple sources and sinks. Math. Programming **7**, 97–107 (1974)
56. Megiddo, N.: Combinatorial optimization with rational objective functions. Math. Oper. Res. **4**, 414–424 (1979)
57. Megiddo, N.: applying parallel computation algorithms in the design of serial algorithms. J. ACM **30**, 852–865 (1983)
58. Murota, K.: Convexity and Steinitz's exchange property. Adv. Math. **124**, 272–311 (1996)
59. Murota, K.: Discrete convex analysis. Math. Programming, **83**, 313–371 (1998)
60. Murota, K.: Discrete Convex Analysis SIAM (2003)
61. Nagamochi, H., Ibaraki, T.: Computing edge-connectivity of multigraphs and capacitated graphs. SIAM J. Discrete Math. **5**, 54–66 (1992)
62. Nagamochi, H., Ibaraki, T.: A note on minimizing submodular functions. Inform. Process. Lett. **67**, 239–244 (1998)
63. Nagano, K.: A strongly polynomial algorithm for line search in submodular polyhedra. Technical Report METR 2004-33, University of Tokyo, June 2004
64. Nash-Williams, C.St.J.A.: Connected detachments of graphs and generalized Euler trails. J. London Math. Soc. **31**, 17–29 (1985)
65. Nash-Williams, C.St.J.A.: Another proof of a theorem concerning detachments of graphs. Europ. J. Combinatorics **12**, 245–247 (1991)
66. Nash-Williams, C.St.J.A.: Strongly connected mixed graphs and connected detachments of graphs. J. Combin. Math. Combin. Comput. **19**, 33–47 (1995)
67. Nash-Williams, C.St.J.A.: A direct proof of a theorem on detachments of finite graphs. J. Combin. Math. Combin. Comput. **19**, 314–318 (1995)
68. Queyranne, M.: Structure of a simple scheduling polyhedra. Math. Programming **58**, 263–285 (1993)
69. Queyranne, M.: Minimizing symmetric submodular functions. Math. Programming **82**, 3–12 (1998)
70. Rizzi, R.: On minimizing symmetric set functions. **20**, 445–450 (2000)
71. Schrijver, A.: A combinatorial algorithm minimizing submodular functions in strongly polynomial time. J. Combin. Theory Ser. B **80**, 346–355 (2000)
72. Schrijver, A.: Combinatorial Optimization — Polyhedra and Efficiency. Springer, Berlin (2003)
73. Shanthikumar, J.G., Yao, D.D.: Multiclass queueing systems: polymatroidal structure and optimal scheduling control. Oper. Res. **40**, S293–S299 (1992)
74. Shapley, L.S.: Cores of convex games. Int. J. Game Theory **1**, 11–26 (1971)
75. Slepian, D., Wolf, J.K.: Noiseless coding with of correlated information sources. IEEE Trans. Inform. Theory **IT19**, 471–480 (1973)
76. Vygen, J.: A note on Schrijver's submodular function minimization algorithm. J. Combin. Theory Ser. B **88**, 399–402 (2003)
77. Whitney, H.: On the abstract properties of linear dependence. Amer. J. Math. **57**, 509–533 (1935)