アルゴリズム 46−11
（1995. 7. 20）

凸費用劣モジュラ流に対する容量スケーリング法

岩田 覚
京都大学 数理解析研究所

梗概: 劣モジュラ関数に対するスケーリング技法を導入する．その結果，費用が分離凸関数で与えられた劣モジュラ流問題の整数最適解を求める弱多項式時間算法が得られる．

A Capacity Scaling Algorithm for Convex Cost Submodular Flows

Satoru IWATA
Research Institute for Mathematical Sciences, Kyoto University

Abstract: This paper presents a scaling scheme for submodular functions. A small but strictly submodular function is added before scaling so that the resulting functions should be submodular. This scaling scheme leads to a weakly polynomial algorithm to solve minimum cost integral submodular flow problems with separable convex cost functions, provided that an oracle for exchange capacities are available.

1 Introduction

The submodular flow problem, introduced by Edmonds–Giles [3], is one of the most important frameworks of efficiently solvable combinatorial optimization problems. It includes the minimum cost flow, the graph orientation, the polymatroid intersection, and the directed cut covering problems as its special cases. There have been proposed a number of combinatorial algorithms, which rely on an oracle for exchange capacities (cf. Frank–Tardos [6], Fujishige [8] and references therein). In particular, Cunningham–Frank [2] developed a primal-dual algorithm, which achieves a weakly polynomial time complexity with the aid of the cost scaling technique.

The scaling technique was introduced by Edmonds–Karp [4] to solve the minimum cost flow problem, where capacities and demands were scaled rather than arc costs. One of the advantages of the capacity scaling approach is that it can easily be applied to minimum cost integral flows with separable convex cost functions (cf. Ahuja–Magnanti–Orlin [1, Chap. 14], Hochbaum–Shanthikumar [9], and Minoux [11]).

The main purpose of the present paper is to develop a polynomial algorithm for integral submodular flows with separable convex cost functions. It is quite natural to adopt a capacity scaling approach, i.e., to scale capacities as well as demands, which are given by a submodular function.

A straightforward attempt to scale a submodular function, however, destroys the submodularity. To be more specific, for a submodular function $f : 2^V \to \mathbf{Z}$, it seems natural to consider a function $\tilde{f}$ defined by

$$\tilde{f}(X) = \lfloor \frac{f(X)}{2} \rfloor \quad (X \subseteq V).$$

This function $\tilde{f}$ is, however, no longer submodular if there exist $X_1, X_2 \subseteq V$ such that $f(X_1) + f(X_2) = f(X_1 \cup X_2) + f(X_1 \cap X_2)$ holds with $f(X_1)$, $f(X_2)$ odd and $f(X_1 \cup X_2)$, $f(X_1 \cap X_2)$ even numbers.

Thus how to extend the capacity scaling approach to submodular flows is a nontrivial issue. In fact, no good algorithms for submodular flows with separable convex cost functions have been known, although an optimality criterion is given in Fujishige [8, §12].

As we have seen above, a straightforward scaling scheme may fail particularly when the submodular function has a modular pair, or in other words, when the base polyhedron has a degenerate face. In order to avoid such a phenomenon we perturb the base polyhedron by adding a small but strictly submodular function before scaling. This process makes the algorithm somewhat complicated and leads to a rather higher order polynomial complexity. Hence the capacity scaling approach may not be attractive for submodular flow problems with linear cost functions. It still has some significance, however, in the context of separable convex cost submodular flows, where no other good algorithms have been known.

2 Separable Convex Cost Submodular Flows

Let $G = (V, A)$ be a directed graph with a vertex set V of cardinality n and an arc set A of cardinality m. Functions $\overline{c} : A \to \mathbf{Z}$ and $\underline{c} : A \to \mathbf{Z}$ are upper and lower capacities, respectively. Suppose that $f : 2^V \to \mathbf{Z}$ is a submodular function with $f(V) = 0$. Let $\mathrm{B}(f)$ denote the base polyhedron, i.e.,

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

For each arc $a \in A$, let $\gamma_a : \mathbf{R} \to \mathbf{R}$ be a convex cost function. Our problem to be considered in this paper is as follows:

$$\begin{array}{lll} \text{Minimize} & \displaystyle\sum_{a \in A} \gamma_a(\varphi(a)) & (1) \\ \text{subject to} & \underline{c}(a) \leq \varphi(a) \leq \overline{c}(a) \quad (a \in A) & (2) \\ & \partial\varphi \in \mathrm{B}(f), \quad \varphi \in \mathbf{Z}^A. & (3) \end{array}$$

A feasible solution of this problem is said to be an integral submodular flow. The feasibility of this problem can be checked efficiently [5]. Once we have a feasible solution, we can transform the problem to another optimization problem in the form of (1)-(3) with $\underline{c}(a) \leq 0 \leq \overline{c}(a)$ for $a \in A$ and $f(X) \geq 0$ for $X \subseteq V$. Hence we may assume without loss of generality that $\varphi(a) = 0$ for $a \in A$ is feasible.

It should be remarked here that Edmonds–Giles [3] formulated the submodular flow problem in terms of crossing-submodular functions. On the other hand, however, it has been pointed out by Fujishige [7] that a base polyhedron of a crossing-submodular function can be expressed as a base polyhedron of a fully-submodular function. Since the main part of the algorithm to be presented in this paper is concerned with geometric properties of the base polyhedron, we deal with fully-submodular functions for the sake of simplicity.

3 A Capacity Scaling Algorithm

We present a capacity scaling algorithm, which performs a number of scaling phases for different values of a parameter Δ. Each scaling phase is a successive shortest path algorithm, where the amount of augmentation at once is exactly Δ. When there is no possibility of augmentation, the algorithm reduces the value of Δ by a factor of two.

For a given value of $\Delta = 2^{\ell}$ for $\ell = -1, 0, 1, 2, \ldots$, we consider a function $f_{\Delta} : 2^V \to \mathbf{Z}$ defined by

$$f_{\Delta}(X) = \Delta \cdot \lfloor \frac{f(X)}{\Delta} \rfloor + \lfloor \Delta \rfloor \cdot b(X),$$

where $b(X) = |X| \cdot |V - X|$. Note that f_{Δ} for $\Delta = 1/2$ coincides with f. The second term above ensures the following lemma, which is crucial for our capacity scaling scheme.

LEMMA 1. *The function f_{Δ} is submodular for $\Delta = 2^{\ell}$ $(\ell = -1, 0, 1, 2, \ldots)$.*

In this short paper, we assume that the dependence function dep_{Δ} for f_{Δ} can be computed efficiently. In fact, such a procedure is presented in [10, §4], provided that exchange capacities for f are available.

Given a flow $\varphi : A \to \mathbf{Z}$ and a base $x \in \mathrm{B}(f_{\Delta})$ for a value of Δ, we construct an auxiliary network $G_{\Delta}(\varphi, x) = (V, A_{\Delta}(\varphi, x))$ with

$$A_{\Delta}(\varphi, x) = \widehat{A}_{\Delta}(\varphi) \cup \bar{A}_{\Delta}(\varphi) \cup \widetilde{A}_{\Delta}(x),$$

where

$$\begin{aligned}
\widehat{A}_{\Delta}(\varphi) &= \{a \mid a \in A, \varphi(a) + \lceil \Delta \rceil \leq \overline{c}(a)\}, \\
\bar{A}_{\Delta}(\varphi) &= \{\bar{a} \mid a \in A, \varphi(a) - \lceil \Delta \rceil \geq \underline{c}(a)\} \quad (\bar{a}\text{: reorientation of } a), \\
\widetilde{A}_{\Delta}(x) &= \{(u, v) \mid u \in \mathrm{dep}_{\Delta}(x, v) - \{v\}\}.
\end{aligned}$$

The arc length $\xi_\Delta : A_\Delta(\varphi, x) \to \mathbf{R}$ is defined by

$$\xi_\Delta(a) = \begin{cases} \{\gamma_a(\varphi(a) + \lceil\Delta\rceil) - \gamma_a(\varphi(a))\}/\lceil\Delta\rceil & (a \in \widehat{A}_\Delta) \\ \{\gamma_a(\varphi(\bar{a}) - \lceil\Delta\rceil) - \gamma_a(\varphi(\bar{a}))\}/\lceil\Delta\rceil & (a \in \bar{A}_\Delta) \\ 0 & (a \in \widetilde{A}_\Delta). \end{cases}$$

For a node potential $\pi : V \to \mathbf{R}$, we put

$$\xi^\pi_\Delta(a) = \xi_\Delta(a) + \pi(\partial^+ a) - \pi(\partial^- a),$$

which is called the reduced arc length. Note that $x \in \mathrm{B}(f_\Delta)$ is a π-maximum base if and only if $\xi^\pi_\Delta(a) \geq 0$ for every $a \in \widetilde{A}_\Delta(x)$. We also denote

$$\begin{aligned} S_\Delta &= \{v \mid x(v) > \partial\varphi(v)\}, \\ T_\Delta &= \{v \mid x(v) < \partial\varphi(v)\}, \end{aligned}$$

where vertices in S_Δ and T_Δ are called sources and sinks, respectively.

Then we have the following theorem, which gives an optimality criterion of the problem (1)–(3) when $\Delta = 1/2$. See Theorem 12.1 of [8].

THEOREM 2. *For a given value of* $\Delta = 2^\ell$ $(\ell = -1, 0, 1, 2, \ldots)$, *consider the following problem:*

$$\begin{array}{ll} \text{Minimize} & \sum_{a \in A} \gamma_a(\varphi(a)) \\ \text{subject to} & \underline{c}(a) \leq \varphi(a) \leq \bar{c}(a) \quad (a \in A) \\ & \partial\varphi \in \mathrm{B}(f_\Delta), \\ & \varphi(a)\text{: multiple of } \lceil\Delta\rceil \quad (a \in A). \end{array}$$

Then a feasible solution φ *is an optimal solution if and only if there exists a node potential* $\pi : V \to \mathbf{R}$ *such that* $\xi^\pi_\Delta(a) \geq 0$ *for every arc* $a \in A_\Delta(\varphi, x)$ *in* $G_\Delta(\varphi, x)$ *with* $x = \partial\varphi$.

We are now ready to present an outline of the algorithm. A scaling phase with a specific value of Δ is referred to as the Δ-scaling phase. Initially, the value of Δ is set to be $2^{\lfloor \log U \rfloor}$, where

$$U = \max\{\max_{a \in A} |\underline{c}(a)|, \max_{a \in A} |\bar{c}(a)|, \max_{X \subseteq V} f(X)\}.$$

The flow φ and the node potential π are initialized by $\varphi(a) = 0$ for each arc $a \in A$ and $\pi(v) = 0$ for each vertex $v \in V$, respectively. Note that this φ is a feasible solution because of our assumption in Section 2.

In the Δ-scaling phase, the algorithm keeps a flow φ, a node potential π, and a base $x \in \mathrm{B}(f_\Delta)$ such that $\varphi(a)$ and $x(v)$ are multiples of $\lceil\Delta\rceil$ for every arc $a \in A$ and every vertex $v \in V$, respectively, and that $\xi^\pi_\Delta(a) \geq 0$ holds for every arc

$a \in A_\Delta(\varphi, x)$. In order to reduce $\sum_{v \in S_\Delta} \{x(v) - \partial\varphi(v)\}$, it repeats augmentations of the flow φ by $\lceil\Delta\rceil$ along a shortest path from S_Δ to T_Δ with respect to ξ^π_Δ as well as updates of the node potential π and the base x. When $\partial\varphi$ coincides with x, the flow φ is an optimal solution to the problem in Theorem 2. Then the algorithm cuts the value of Δ in half and starts the new scaling phase. Eventually, after $\mathrm{O}(\log U)$ scaling phases, Δ becomes $1/2$. The flow φ obtained at the end of this scaling phase turns out to be an optimal solution of the original problem. Note that, except for the last scaling phase, $\lceil\Delta\rceil = \Delta$.

Suppose, at the beginning of the Δ-scaling phase, we have a flow φ and a node potential π such that $\varphi(a)$ is a multiple of 2Δ for each arc $a \in A$ and $\xi^\pi_{2\Delta}(a) \geq 0$ for every arc $a \in A_{2\Delta}(\varphi, \partial\varphi)$. This is certainly satisfied for the first scaling phase because $\widehat{A}_{2\Delta}$ and $\bar{A}_{2\Delta}$ are empty. The algorithm starts the Δ-scaling phase with finding a π-maximum base $x \in \mathrm{B}(f_\Delta)$ that is sufficiently close to $\partial\varphi$ with $x(v)$ a multiple of Δ for each $v \in V$. How to find such an appropriate base is postponed to the end of this section because it is irrelevant to the correctness of the algorithm.

It then checks the condition $\xi^\pi_\Delta(a) \geq 0$ for each arc $a \in \widehat{A}_\Delta(\varphi) \cup \bar{A}_\Delta(\varphi)$. This condition is automatically satisfied at the last scaling phase with $\Delta = 1/2$ because ξ^π_Δ coincides with $\xi^\pi_{2\Delta}$ in this case. Hence we may assume here $\lceil\Delta\rceil = \Delta$. Note that either $a \in \widehat{A}_\Delta(\varphi)$ or its reorientation $\bar{a} \in \bar{A}_\Delta(\varphi)$ satisfies this condition because of the convexity of γ_a. If $\xi^\pi_\Delta(a) < 0$ for $a \in \widehat{A}_\Delta(\varphi)$, the algorithm augments the flow value $\varphi(a)$ by Δ. In this case, before the augmentation, we have

$$\gamma_a(\varphi(a) + \Delta) - \gamma_a(\varphi(a)) + \Delta\pi(\partial^+ a) - \Delta\pi(\partial^- a) < 0, \tag{4}$$

which implies $\widehat{\xi}^\pi_\Delta(\bar{a}) > 0$, where $\widehat{\xi}^\pi_\Delta$ denotes the reduced arc length after the augmentation. On the other hand, it follows from the supposition that

$$\gamma_a(\varphi(a) + 2\Delta) - \gamma_a(\varphi(a)) + 2\Delta\pi(\partial^+ a) - 2\Delta\pi(\partial^- a) \geq 0. \tag{5}$$

Combining (4) and (5), we have

$$\gamma_a(\varphi(a) + 2\Delta) - \gamma_a(\varphi(a) + \Delta) + \Delta\pi(\partial^+ a) - \Delta\pi(\partial^- a) \geq 0, \tag{6}$$

which implies $\widehat{\xi}^\pi_\Delta(a) \geq 0$. A similar argument justifies reducing the flow value $\varphi(a)$ by Δ if $\xi^\pi_\Delta(\bar{a}) < 0$ for $\bar{a} \in \bar{A}_\Delta(\varphi)$. Thus the algorithm obtains a flow φ, a node potential π, and a base $x \in \mathrm{B}(f_\Delta)$ with $\xi^\pi_\Delta(a) \geq 0$ for every arc $a \in A_\Delta(\varphi, x)$, being ready to start the successive shortest path procedure. Note that $\varphi(a)$ is now a multiple of Δ for each arc $a \in A$.

In the auxiliary network $G_\Delta(\varphi, x)$ with nonnegative reduced arc length ξ^π_Δ, let $d : V \to \mathbf{R}$ be a shortest path distances from S_Δ to all nodes in $G_\Delta(\varphi, x)$ and P be a shortest path from S_Δ to T_Δ with minimum number of arcs. It follows from the triangular inequality

$$d(\partial^- a) \leq d(\partial^+ a) + \xi^\pi_\Delta(a)$$

that

$$\xi_\Delta(a) + \pi(\partial^+ a) - \pi(\partial^- a) + d(\partial^+ a) - d(\partial^- a) \geq 0$$

holds for every arc $a \in A_\Delta(\varphi, x)$. Hence updating π to $\pi + d$ retains the nonnegativity of the reduced arc length. In particular, the reduced arc length becomes zero for each arc in the shortest path P. It is clear that augmenting φ along P by $\lceil\Delta\rceil$ does not violate the capacity constraints nor the nonnegativity of reduced arc length $\xi^\pi_\Delta(a)$ for $a \in \widehat{A}_\Delta(\varphi) \cup \bar{A}_\Delta(\varphi)$. The base $x \in \mathrm{B}(f_\Delta)$ is also updated along P so that $x(v) = \partial\varphi(v)$ holds for every inner vertex v of P, which is possible because of the minimality of P. See Lemma 4.5 of [8]. For the starting point $s \in S_\Delta$ of the shortest path P, either $\partial\varphi(s)$ increases or $x(s)$ decreases by $\lceil\Delta\rceil$. Hence $\sum\limits_{v \in S_\Delta} \{x(v) - \partial\varphi(v)\}$ reduces exactly by $\lceil\Delta\rceil$. Repeat this process until the sources S_Δ and consequently the sinks T_Δ become empty. Then we obtain a flow φ with $\partial\varphi \in \mathrm{B}(f_\Delta)$ and a node potential π such that $\xi^\pi_\Delta(a) \geq 0$ for every $a \in A_\Delta(\varphi, \partial\varphi)$ and that $\varphi(a)$ is a multiple of $\lceil\Delta\rceil$ for each $a \in A$. According to Theorem 2, the flow φ is now an optimal solution to the problem therein.

An algorithmic description of the capacity scaling algorithm is now given as follows:

```
algorithm capacity scaling
begin
    $\Delta := 2^{\lfloor \log U \rfloor}$;
    for $a \in A$ do $\varphi(a) := 0$;
    for $v \in V$ do $\pi(v) := 0$;
    while $\Delta \geq 1/2$ do
    begin {$\Delta$-scaling phase}
        find an appropriate $\pi$-maximum base $x \in \mathrm{B}(f_\Delta)$;
        for $a \in \widehat{A}_\Delta(\varphi)$ do if $\xi^\pi_\Delta(a) < 0$ then $\varphi(a) := \varphi(a) + \Delta$;
        for $a \in \bar{A}_\Delta(\varphi)$ do if $\xi^\pi_\Delta(a) < 0$ then $\varphi(\bar{a}) := \varphi(\bar{a}) - \Delta$;
        while $x \neq \partial\varphi$ do
        begin
            determine shortest path distances $d : V \to \mathbf{R}$ from $S_\Delta$
                to all other nodes in $G_\Delta(\varphi, x)$ with respect to $\xi^\pi_\Delta$;
            let $P$ be a shortest path from $S_\Delta$ to $T_\Delta$ with minimum number of arcs;
            for $v \in V$ do $\pi(v) := \pi(v) + d(v)$;
            for $a \in P \cap \widehat{A}_\Delta(\varphi)$ do $\varphi(a) := \varphi(a) + \lceil\Delta\rceil$;
            for $a \in P \cap \bar{A}_\Delta(\varphi)$ do $\varphi(\bar{a}) := \varphi(\bar{a}) - \lceil\Delta\rceil$;
            for $a \in P \cap \widetilde{A}_\Delta(x)$ do $x(\partial^+ a) := x(\partial^+ a) - \lceil\Delta\rceil$, $x(\partial^- a) := x(\partial^- a) + \lceil\Delta\rceil$;
        end;
        $\Delta := \Delta/2$;
    end
end.
```

We now specify how to find the "appropriate" π-maximum base $x \in \mathrm{B}(f_\Delta)$. Since one augmentation reduces $\sum_{v \in S_\Delta} \{x(v) - \partial\varphi(v)\}$ by Δ, we should choose a base sufficiently close to $\partial\varphi$ in order to obtain an efficient algorithm.

Let $\pi_1 > \pi_2 > \cdots > \pi_t$ be the distinct values of $\pi(v)$ and put

$$V_i = \{v \mid \pi(v) \geq \pi_i\} \quad (i = 1, 2, \cdots, t)$$

and $V_0 = \emptyset$. Consider a submodular function $\widehat{f}_\Delta$ defined by

$$\widehat{f}_\Delta(X) = \sum_{i=1}^{t} \{f_\Delta((X \cap V_i) \cup V_{i-1}) - f_\Delta(V_i)\} \quad (X \subseteq V).$$

Then $\mathrm{B}(\widehat{f}_\Delta)$ is the set of π-maximum bases of $\mathrm{B}(f_\Delta)$. We define $\widehat{f}_{2\Delta}$ from $f_{2\Delta}$ in the same way, i.e.,

$$\widehat{f}_{2\Delta}(X) = \sum_{i=1}^{t} \{f_{2\Delta}((X \cap V_i) \cup V_{i-1}) - f_{2\Delta}(V_i)\} \quad (X \subseteq V),$$

and then $\partial\varphi \in \mathrm{B}(\widehat{f}_{2\Delta})$ at the beginning of the Δ-scaling phase. Since $\lfloor 2\Delta \rfloor = 2\Delta = \lceil \Delta \rceil + \lfloor \Delta \rfloor$ for $\Delta = 2^\ell$ with $\ell = -1, 0, 1, 2, \ldots$, we have

$$\begin{aligned} \widehat{f}_{2\Delta}(X) - \widehat{f}_\Delta(X) \;&\leq\; \sum_{i} \{\Delta \lfloor \frac{f(V_{i-1})}{\Delta} \rfloor - 2\Delta \lfloor \frac{f(V_{i-1})}{2\Delta} \rfloor \mid X \cap (V_i \backslash V_{i-1}) \neq \emptyset\} \\ &\quad + \lceil \Delta \rceil \cdot \sum_{i=1}^{t} \{b((X \cap V_i) \cup V_{i-1}) - b(V_i)\} \\ &\leq\; \Delta \cdot |X| + \lceil \Delta \rceil \cdot b(X) \leq n \lceil \Delta \rceil \cdot |X| \end{aligned}$$

for any $X \subseteq V$.

Consider a vector $\widehat{x} \in \mathbf{Z}^V$ defined by $\widehat{x}(v) = \partial\varphi(v) - n\lceil \Delta \rceil$ for each $v \in V$. Then it holds for every $X \subseteq V$ that

$$\widehat{x}(X) = \partial\varphi(X) - n\lceil \Delta \rceil \cdot |X| \leq \widehat{f}_{2\Delta}(X) - n\lceil \Delta \rceil \cdot |X| \leq \widehat{f}_\Delta(X),$$

which means $\widehat{x} \in \mathrm{P}(\widehat{f}_\Delta)$. Hence there exists a base $x \in \mathrm{B}(\widehat{f}_\Delta)$ with $x \geq \widehat{x}$, which can be found by the greedy algorithm starting from $\widehat{x}$. See Theorem 3.19 of [8]. We adopt this x as an "appropriate" base in the algorithm.

We conclude this section by analyzing the number of shortest path computations in the algorithm. It is clear that

$$\sum_{v \in S_\Delta} \{x(v) - \partial\varphi(v)\} \leq x(V) - \widehat{x}(V) \leq n^2 \lceil \Delta \rceil$$

at the beginning of the Δ-scaling phase. Adjusting the flow φ to remove the arcs with negative reduced arc length from the auxiliary graph $G_\Delta(\varphi, x)$ changes the left-hand side at most $m\lceil \Delta \rceil$. Thus each scaling phase performs the shortest path augmentation at most $n^2 + m$ times, and the whole algorithm requires $\mathrm{O}(n^2 \log U)$ shortest path computations.

4 Conclusion

We have presented a capacity scaling algorithm for integral submodular flows with separable convex cost functions. The total amount of computation is bounded by a polynomial of the input size of the problem, provided that an oracle for exchange capacities for the original submodular function is available. It is remarked finally that the function b in the definition of f_Δ can be replaced by any other strictly submodular functions of network-type.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin: *Network Flows — Theory, Algorithms, and Applications*, Prentice Hall, 1993.

[2] W. H. Cunningham and A. Frank: A primal-dual algorithm for submodular flows, *Math. Oper. Res.* 10 (1985), pp. 251–262.

[3] J. Edmonds and R. Giles: A min-max relation for submodular functions on graphs, *Ann. Discrete Math.*, 1 (1977), pp. 185–204.

[4] J. Edmonds and R. M. Karp: Theoretical improvements in algorithmic efficiency for network flow problems, *J. ACM*, 19 (1972), pp. 248–264.

[5] A. Frank: Finding feasible vectors of Edmonds–Giles polyhedra, *J. Combin. Theory*, Ser. B, 36 (1984), pp. 221–239.

[6] A. Frank and E. Tardos: Generalized polymatroids and submodular flows, *Math. Programming*, 42 (1988), pp. 489–563.

[7] S. Fujishige: Structures of polyhedra determined by submodular functions on crossing families, *Math. Programming*, 29 (1984), pp. 125–141.

[8] S. Fujishige: *Submodular Functions and Optimization*, North-Holland, 1991.

[9] D. S. Hochbaum and J. G. Shanthikumar: Convex separable optimization is not much harder than linear optimization, *J. ACM*, 37 (1990), pp. 843–862.

[10] S. Iwata: A capacity scaling algorithm for convex cost submodular flows, RIMS-1015, Research Institute for Mathematical Sciences, Kyoto University, 1995.

[11] M. Minoux: Solving integer minimum cost flows with separable convex objective polynomially, *Math. Programming Stud.*, 26 (1986), pp. 237–239.