Notes on graph cuts with submodular edge weights

Stefanie Jegelka
Max Planck Institute for Biological Cybernetics
Tübingen, Germany
jegelka@tuebingen.mpg.de

Jeff Bilmes
University of Washington, Seattle
Dept. of Electrical Engineering
bilmes@uw.edu

Abstract

Generalizing the cost in the standard min-cut problem to a submodular cost function immediately makes the problem harder. Not only do we prove NP hardness even for non-negative submodular costs, but also show a lower bound of $\Omega(|V|^{1/3})$ on the approximation factor for the (s,t) cut version of the problem. On the positive side, we propose and compare three approximation algorithms with an overall approximation factor of $O(\min\{|V|, \sqrt{|E|}\log|V|\})$ that appear to do well in practice.

1 Introduction

We consider the problem of partitioning the node set V of an undirected graph $G = (V, E)$ into $(X, V \setminus X)$ with minimum cost. Let δX be the set of edges between X and $V \setminus X$, so that $\delta X = \{(i,j) \in E : i \in X, j \in V \setminus X\}$. The cost of a cut is measured by a submodular function $f : 2^E \to \mathbb{R}$ defined on the edges of G. That is, a cut δX has cost $g(X) \triangleq f(\delta X)$, and the goal is to identify an $\emptyset \subset X \subset V$ that minimizes $g(X)$. Note that g is not submodular in general. Equivalently, the problem can be viewed as minimizing f under a G-defined cut constraint, where a cut must bi-partition the graph. We name this problem edge-submodular graph cuts (ESC) to distinguish it from the standard (edge-modular cost) graph cut problem, which is the minimization of a submodular function on the *nodes* (rather than the edges) and solvable in polynomial time.

If f is a modular function (i.e., $f(A) = \sum_{e \in A} f(a)$, $\forall A \subseteq E$), then ESC reduces to the standard min-cut problem. ESC differs from submodular flows (solvable in polynomial time), where submodularity defines feasible flows but the cost of an edge set is still modular. We show in Section 2 that submodular costs make the min-cut problem much harder: it becomes NP-hard even if f is nonnegative and normalized (i.e., $f(\emptyset) = 0$). Our reduction illustrates the expressive power of submodular functions. In fact, we show that edge-submodular (s,t)-cuts are not approximable with a factor better than $\Omega(n^{1/3})$ for $n = |V|$. In Section 3, we present preliminary techniques that achieve good results in practice and an $O(\min\{|V|, \sqrt{|E|}\log|V|\})$ approximation in theory, thereby portending well for tighter approximation schemes.

An application for ESC, and our key motivation, comes from finding separators in graphical models. Given a graphical model describing a family of distributions, we often wish to find a separator, i.e., a set of variables rendering a bi-partition of the remaining variables conditionally independent. We desire either small separators (measured as number of random variables), or separators with small state-space (the joint alphabet size of the separator is small). For both measures, a standard min-cut algorithm solves the problem both exactly and efficiently. In certain cases, however, we need a separator with small entropy. The log state space is a modular upper bound on the entropy, but an arbitrarily loose one, so directly minimizing the entropy is better. A low-entropy separator can be encoded with a small number of bits and therefore can be seen as information theoretically equivalent to a small number of random variables, although more general. It is useful for graphical-model inference on distributed computers. Sharing such a separator across machines can minimize the cross-machine communication when there is no bi-partition with low mutual information. The separator also helps to find boundaries in dynamic graphical models [1]. Since entropy is submodular in the random variable set, finding a low-entropy separator is an instance of ESC.

A variety of previous work relates to our problem, such as the standard well-studied (modular) min-cut problem [2] or the aforementioned submodular flow problem (see references in [2]). In the polymatroidal maxflow problem (PF) [3], two submodular capacity functions at each node restrict the in- and outflow of that node, respectively. The inflow (resp. outflow) must lie within the independence polyhedron associated with

the corresponding inflow (resp. outflow) capacity. We use the solution to PF for a poly-time approximation to ESC. A special case of ESC are label cuts, where one pays for sets of edges [4]. In general, there has been recent interest in replacing modular functions with submodular ones in standard combinatorial problems. For example, [5] consider submodular load balancing, sparsest cut and balanced cut. Submodular vertex cover, spanning trees, shortest paths, and perfect matchings are addressed in [6]. The function approximation in [7] makes algorithms for linear costs amenable to approximate problems with submodular costs. We utilize ideas from these papers. Further recent work considers submodular minimization with set cover constraints [8] or maximization with matroid constraints [9].

2 Hardness

We start off by proving that submodular edge weights render the cut problem NP hard, and the (s,t)-cut version inapproximable at better than $\Omega(n^{1/3})$[1]. The hardness proof also holds for (s,t)-cuts with nonnegative, monotone costs. If we relax the monotonicity condition, then the lower bound also holds for ESC by adding an edge (s,t) as in the proof of NP hardness.

First, note that if f is nonnegative, normalized, and monotone, then g is subadditive on the nodes. This has no general benefit though: Let $R \subseteq V$ be an arbitrary set of nodes and $b > 1$ a large number, and define a (subadditive) function $g : 2^V \to \mathbb{R}$ as $g(X) = \mathbf{1}[X = R] + b\mathbf{1}[X \neq R \text{ and } X \neq \emptyset]$, where $\mathbf{1}[\cdot]$ is the indicator function. Without knowing R, minimizing g requires exponentially many evaluations of g. This hard function is the node-based cost $g(X) = f(\delta X)$ of ESC with edge costs $f(A) = \max_{e \in A} w(e)$ where $w(e) = \mathbf{1}[e \in \delta R] + b\mathbf{1}[e \notin \delta R]$. Knowing the graph structure (thereby breaking apart g), however, we can find the optimum in polynomial time by greedily merging node pairs that are connected by heavy edges.

2.1 Edge-submodular cuts are NP hard

It is known that the common min-cut problem with nonnegative, modular edge weights becomes hard if edge weights can be negative, or if size constraints are added on the partitions [10]. If we allow an arbitrary submodular function f for the costs, then it is immediately clear that the ESC problem becomes NP hard. As an example, correlation clustering (CC) corresponds to ESC with a modular f that takes the values ± 1 and a complete graph G. CC for a fixed number of partitions is NP hard but does have a PTAS [11]. With strictly negative modular f, ESC becomes the max-cut problem, also NP hard but with a constant-factor approximation [2]. By a graph bisection reduction, we prove that even with a nonnegative f, ESC is hard.

Definition 1 (Graph Bisection (GB))**.** *Given an undirected graph $G = (V, E)$ with weights $w : E \to \mathbb{R}_0^+$, find a partition $V_1, V_2 \subset V$ of the nodes, such that $|V_1| = |V_2| = |V|/2$ and $\sum_{e \in E \cap (V_1 \times V_2)} w(e)$ is minimal.*

GB is NP hard and does not have a PTAS [12]. Let $G_B = (V_B, E_B)$ be an instance of GB with n nodes. We create an auxiliary graph G and submodular function f whose minimum ESC corresponds to the optimal bisection of G_B. G has two additional nodes s, t and $2n + 1$ additional edges. To form G, retain G_B with the costs on E_B and connect the additional nodes s and t to every vertex in G_B with corresponding new edge sets E_s and E_t. Also connect s with t. Thus, $G = (V_B \cup \{s,t\}, E_B \cup E_s \cup E_t \cup \{(s,t)\})$. The minimum ESC will (i) separate s and t, (ii) separate the nodes in V_B into two equal-sized partitions, that is, cut $n/2$ edges each of E_s and E_t, and (iii), have minimum cost with respect to the edges E_B. We enforce the structural constraints (i) and (ii) with barrier submodular functions f_1 and f_2, respectively, and then add the cost $f_3(A) = \sum_{e \in A \cap E_B} w(e)$. The overall cost is

$$f(A) = \alpha_1 f_1(A) + \alpha_2 f_2(A) + \alpha_3 f_3(A),$$

defined on $E(G)$ with $\alpha_i > 0$ to be specified later. First, let $f_1(A) = \mathbf{1}[(s,t) \notin A]\,|A|$. This function is submodular and strongly favors the inclusion of edge (s,t).

Next, an (s,t) separation cuts n edges in $E_s \cup E_t$. A balanced cut of V_B assigns $n/2$ nodes to s, cutting their edges to t, and the other $n/2$ nodes to t, cutting their edges to s. Hence, the barrier f_2 on $E_s \cup E_t$ favors, among all sets $A \subseteq E$ separating s and t, those that fulfill two conditions: $|A \cap E_s| = n/2 = |A \cap E_t|$, and for each node $v \in V_B$, the cut A cuts off either the connection to s or to t. Cutting both connections could lead to an imbalanced partition. Let $A_s = A \cap E_s$, $A_t = A \cap E_t$, and $A_{s \cap t} = \{(v_i, s) | (v_i, s) \in A_s \text{ and } (v_i, t) \in A_t\}$. The desired function is

$$f_2(A) = (|A_s| + |A_t|)D(n) - (|A_s||A_t| - |A_{s \cap t}|)D'(n-1), \tag{1}$$

where $D(n)$ and $D'(n-1)$ are suitable constants depending on n. If $D(n)$ is the number of derangements of n elements, and $D'(n)$ is the number of "derangements" when one element can be mapped to itself, then

[1] Only after completing our proofs we became aware of the special case considered in [4] that is also NP hard but has a weaker lower bound.

f_2 is the sum of $D(n)$ rank functions. In this case, the constants are $D(n) = n! \sum_{k=0}^{n} (-1)^k/k!$ [13], and $D'(n-1) = \sum_{k=0}^{n-1} (n-2)!(n-1-k)!(-1)^k$. Both constants are computable in polynomial time.

If $|A_s| + |A_t|$ is kept constant, then $f_2(A)$ will be minimal if $A_{s\cap t} = \emptyset$. Hence, we should not cut the edges on both sides of a node in V_B. Furthermore, $|A_s||A_t|$ is maximal if $|A_s| = |A_t| = (|A_s| + |A_t|)/2$. If we choose A_s and A_t accordingly, then adding edges to A_s, i.e., cutting more than n edges in $E_s \cup E_t$, will increase the cost, thanks to the choice of $D(n)$ and $D'(n-1)$. Thus, for $|A \cap (E_s \cup E_t)| \geq n$, the value of f_2 is smallest if A contains $n/2$ edges from either side and cuts off each node either from s or t. Note that this f_2 is submodular. Lastly, we choose $\alpha_3 = 1$, $\alpha_2 = 10 \sum_{e\in E_B} w(e)$ and $\alpha_1 = 5\alpha_1 n^2 D'(n-1)$.

2.2 Lower bound for edge-submodular (ES) (s,t)-cuts

In this section, we show a lower bound on the approximation factor of ES (s,t) cuts.

Theorem 1 (Lower bound for ES (s,t) cuts with nonnegative, monotone costs)**.** *For a fixed $\epsilon > 0, \delta > 0$, any (randomized) approximation algorithm for the ES (s,t) cut problem with an approximation factor better than $n^{1/3-\epsilon}/(1+\delta)$ needs exponentially many queries.*

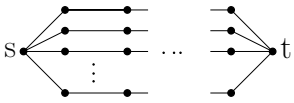

Figure 1: Ladder graph

We prove Theorem 1 with the technique of [7], also used in [5, 6, 8]. The proof shows a type of input where even for a polynomial number of evaluations, it is very unlikely that we can distinguish between two cost functions f, h that may appear as input. Their optima differ by a large factor, say α. Any solution for f that is within a factor of α of the optimum would be enough evidence to discriminate f and h. Thus, no polynomial-time algorithm can guarantee an approximation ratio better than α, since it would have to distinguish between the two functions. To achieve a low probability of discrimination, we randomly pick a cut $R \subset E$ and design f so that for a query $Q \subseteq E$, $f(Q) \neq h(Q)$ only if $|Q \cap R|$ is large, an event of exponentially small probability. By a union bound argument, the probability of having $f(Q) = h(Q)$ for a set of polynomially many queries is still too large for an approximation guarantee better than α.

The graph in Figure 1 has k columns of edges, ℓ parallel paths from s to t, and $n' < k\ell = n$ nodes. Any (s,t) cut cuts each path at least once. Thus, there are k^ℓ minimal (s,t) cuts. To sample a random cut $R \subset E$, we choose one edge from each path uniformly with probability $1/k$. Let $\beta = (1+\delta)\ell/k$ and

$$h(Q) = \min\{|Q|, \ell\}; \qquad f(Q) = \min\{|Q \cap \overline{R}| + \min\{|Q \cap R|,\ \beta\},\ \ell\}.$$

We choose $k = n^{1/3-\epsilon}$ and $\ell = n^{2/3+\epsilon}$. The ratio of optima of h and f is $\ell/\beta = n^{1/3-\epsilon}/(1+\delta)$. Let us now look at the probability $P(f(Q) \neq g(Q)) = P(f(Q) < g(Q))$ for a given $Q \subseteq E$. If $|Q| \leq \ell$, then the probability

$$P(f(Q) < h(Q)) = P(|Q \cap R| > \beta)$$

increases as Q grows. If, on the other hand, $|Q| \geq \ell$, then the probability

$$P(f(Q) < h(Q)) = P(|Q \cap \overline{R}| + \min\{|Q \cap R|,\ (1+\delta)\ell/k\} < \ell)$$

decreases as Q grows. Hence, the probability of difference is largest when $|Q| = \ell$.

So let $|Q| = \ell$. Then we can distribute Q over at most $d = \ell$ and at least $d > \beta$ paths to make $P(|Q \cap R| > \beta)$ nonzero. If Q covers $m \leq k$ edges of a path, then the probability that Q includes the cut edge in this path is m/k. Hence, $|Q \cap R|$ is the sum of d random variables, with $\mathbb{E}[\ |P\ \cap\ R|\ ] = |Q|/k = \ell/k$. Each variable takes values in $\{0,1\}$. We can bound the probability of a large intersection via Hoeffding's bound [14]:

$$P(\ |Q \cap R| \geq (1+\delta)\ell/k\ ) \ \leq\ \exp(-2\delta^2\ell^2/(dk^2)) \ \leq\ \exp(-2\delta^2\ell/k^2) \ =\ \exp(-2n^{3\epsilon}\delta^2).$$

Since the probability of $f(Q) < h(Q)$ is exponentially small, the theorem holds.

Note that this proof only relies on nonnegative monotone submodular functions, in fact, matroid rank functions [7]. For general submodular functions, the lower bound is probably worse.

3 Approximation Algorithms

The difficulty of ESC cuts lies in the non-locality of the submodularity with respect to the graph structure. If the submodularity is restricted to the sets of edges that share an adjacent node, and the function is modular on anything coarser, then the problem is exactly solvable in polynomial time [3]. Even simpler, in the common min-cut problem with a modular cost, there is no inherent edge submodularity. Our approximation algorithms

rely on a local approximation of the submodularity, that is, we split the set E into small local sets E_i (single edges or neighborhoods). The new cost function may be submodular within a set, but behaves in a modular way across sets, i.e., $\hat{f}(A) = \sum_{i=1}^{k} f_i(E_i \cap A)$. Restricted submodularity or an appropriate approximation is the basis for our bounds. In the sequel, $A^* \subseteq E$ denotes the optimal ES cut, and $n = |V|$.

3.1 Approximation of the cost function (acf)

Goemans et al. [7] present an approximation $\hat{f}$ of a submodular function f with $\hat{f}(A) \leq f(A) \leq \alpha \hat{f}(A)$. The approximation factor α is $\sqrt{|E|+1}$ if f is a matroid rank function, and $O(\sqrt{|E|}\log|E|)$ for general monotone submodular functions. For an integer-valued polymatroid rank function whose maximum cost of a single element is bounded (i.e., $\max_{e\in E} f(e) \leq c < \infty$), we can get an approximation within a factor of $\alpha = O(\sqrt{c|E|})$ by approximating the matroid expansion of the polymatroid (Section 10.3 in [15]). For general nonmonotone submodular functions, only a lower bound is known [7].

The approximation is the square root of a modular function, i.e., of the form $\hat{f}(A) = \sqrt{\sum_{e\in A} w(e)}$ where values $w(e)$ are derived from the algorithm. The minimizer of $\hat{f}$ is the same as the minimizer of $\hat{f}^2$. Thus, we set the weight of each edge to $w(e)$ and then solve a traditional min-cut (or (s,t)-cut) with edge cost function $\hat{f}^2(A) = \sum_{e\in A} w(e)$. This problem can be solved efficiently and exactly. Let $B \in \operatorname{argmin}_{B'\subset E \text{ and } B' \text{ a cut}} \hat{f}^2(B')$. The approximation quality α of $\hat{f}$ implies that

$$f(B) \leq \alpha \hat{f}(B) \leq \alpha \hat{f}(A^*) \leq \alpha f(A^*).$$

For planar graphs with $O(n)$ edges, the approximation factor becomes $O(\sqrt{n})$ or $O(\sqrt{n}\log n)$. Note that the graph in the proof of Thm. 1 is planar. Therefore, for planar graphs and matroid rank functions, the above procedure achieves a lower/upper bound gap of $\Omega(n^{1/3})$ versus $O(n^{1/2})$.

3.2 Approximation via "polymatroidal network flows" (pf)

To restrict the submodular behavior of f to local regions, we can partition E into disjoint sets E_i and then use $\hat{f}(A) = \sum_i f(A \cap E_i)$. Locality in the graph is expressed by edges adjacent to the same vertex. Let $\Pi(A) = \{A_1, \ldots, A_n\}$ be an edge-partition of the (s,t)-cut A, where A_i only contains edges adjacent to $v_i \in V(G)$. With $\mathcal{P}_A$ denoting the set of such partitions, let

$$\hat{f}(A) = \min_{\Pi(A)\in\mathcal{P}_A} \sum_i f(A_i), \tag{2}$$

that is, each edge is assigned to its adjacent node either on the s or the t side of the cut. The algorithm for polymatroidal network flows [3] solves an (s,t) cut for this $\hat{f}$ if we use the following construction: Replace each undirected edge in the graph by two opposing directed edges that are parallel with respect to the cost. In polymatroidal flows, the edge capacities are defined by a submodular function on the set of incoming and the set of outgoing edges for each node. We set this function, for both incoming and outgoing edges on each node, to f restricted to the particular set. The dual of the polymatroidal max-flow is the minimum cut with respect to a convolution of the capacities for incoming and outgoing sets [3], and the convolution is then exactly Equation 2. The polymatroidal flow problem can be solved exactly in $O(|E|^6 \log|E|)$ time with an algorithm based on augmenting paths. The cost function f can be any submodular function.

For fixed s, t, let the set A^* be the optimal directed (s,t) cut, and B the cut found by the approximation. Let $\Delta_s(A) \subset V$ be the set of nodes adjacent to A on the s side, and $\Delta_t(A) \subset V$ its analogue on the t side. The set of edges adjacent to node v is δv. Then

$$\begin{aligned}
\hat{f}(A^*) &\leq \min\Big\{ \sum_{v_i\in\Delta_s(A^*)} f(A^*\cap\delta v_i), \sum_{v_j\in\Delta_t(A^*)} f(A^*\cap\delta v_j)\Big\} \\
&\leq \min\Big\{ |\Delta_s(A^*)| \max_{v_i\in\Delta_s(A^*)} f(A^*\cap\delta v_i),\ \ |\Delta_t(A^*)| \max_{v_j\in\Delta_t(A^*)} f(A^*\cap\delta v_j)\Big\} \\
&\leq \min\{\,|\Delta_s(A^*)|,\ |\Delta_t(A^*)|\,\}\ f(A^*).
\end{aligned}$$

The last inequality holds only for monotone nondecreasing submodular functions. Furthermore, we can bound $\min\{|\Delta_s|, |\Delta_t|\} \leq |V|/2$. Let us now relate $f(A^*)$ to $f(B)$ for a a minimum $\hat{f}$-cost cut B. By submodularity, we know that $\sum_i f(A_i) \geq f(\bigcup_i A_i)$ for any collection of disjoint sets $\{A_i\}_i$. Hence, $f(B) \leq \hat{f}(B) \leq \hat{f}(A^*) \leq \min\{|\Delta_s(A)|, |\Delta_t(A)|\} f(A^*)$. If we know that $f(B)/\hat{f}(B) = |V|^{-\alpha}$, then we get a more specific ratio $f(B)/f(A^*) \leq |V|^{1-\alpha}/2$. On dense graphs where $|E|^{1/4}\log|E| > |V|$, this approximation is better than the one in the previous section.

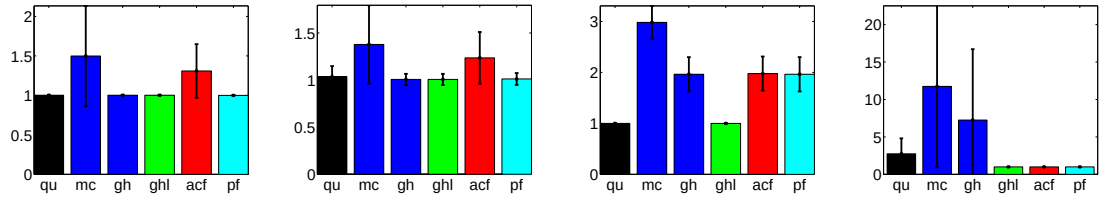

Figure 2: Average approximation factors on (i) grids, (ii) clustered graphs, (iii) grids with large $|A^*|$ and (iv) small dense worst-case graphs.

3.3 Modular minimum cocycle basis with local improvements (ghl)

In this section, we use the modular approximation $\hat{f}(A) = \sum_{e \in A} f(e)$. The minimum cut for $\hat{f}$ is simply the common modular edge-cost min-cut (our "(mc)" baseline below). To improve on this single candidate, we construct the minimum cut basis for the graph with weights $\hat{f}$. The cuts of a graph form a vector space over $\mathbb{F}_2$, and the minimum weight basis for this space can be found by a minimum cut tree [16]. This Gomory-Hu tree is computable by solving $O(n)$ min-cuts [17]. The corresponding cut basis contains a min-cut with respect to $\hat{f}$ for any pair of vertices in the graph. Any of the $n-1$ basis cuts is a candidate cut. We may evaluate f on all of them and pick the minimum (our "(gh)" baseline below).

Among the basis cuts is the minimum cut B with respect to $\hat{f}$. Define the gain of an edge e with respect to a set A to be $\rho_e(A) = f(A \cup \{e\}) - f(A)$. Furthermore, for the optimum A^*, we define $\gamma(A^*) = \min_{e \in A^*} \rho_e(E \setminus \{e\}) / \max_{e \in A^*} f(e)$. Then the approximation factor is at most

$$\frac{f(B)}{f(A^*)} \leq \frac{|A^*|}{1 + (|A^*| - 1)\gamma(A^*)}.$$

This ratio is between 1 for $\gamma(A^*) = 1$ and $|A^*|$ for $\gamma(A^*) = 0$.

To locally improve on the set of basis cuts for $\hat{f}$, we use two upper bounds on the cost of any edge set. For any $B, C \subseteq E$, we have [18]

$$f(B) \leq h_1(B) \triangleq f(C) - \sum_{e \in C \setminus B} \rho_e(E \setminus \{e\}) + \sum_{e \in B \setminus C} \rho_e(C)$$
$$f(B) \leq h_2(B) \triangleq f(C) - \sum_{e \in C \setminus B} \rho_e(C \setminus \{e\}) + \sum_{e \in B \setminus C} \rho_e(\emptyset)$$

For each basis cut C_i, we set $C = C_i$ treated as a constant and find the cut $B_{j,i}$ that minimizes $h_j(B)$. We can continue by taking the better of the two as the comparison set C, and so on, until no $B_{j,i}$ has a lower f-cost than the initial basis cut C_i.

The minimizer of h_j can again be found via a modular min-cut with modified edge weights. Set

$$w_1(e) = \begin{cases} \rho_e(E \setminus \{e\}) & \text{if } e \in C \\ \rho_e(C) & \text{otherwise;} \end{cases} \qquad w_2(e) = \begin{cases} \rho_e(C \setminus \{e\}) & \text{if } e \in C \\ \rho_e(\emptyset) & \text{otherwise.} \end{cases}$$

With these weights, the modular weight of a cut B is

$$\sum_{e \in B} w_1(e) = \sum_{e \in C \cap B} \rho_e(E \setminus \{e\}) + \sum_{e \in B \setminus C} \rho_e(C) = h_1(B) - \underbrace{f(C) + \sum_{e \in C} \rho_e(E \setminus \{e\})}_{\text{constant w.r.t. } B}, \tag{3}$$

and analogously for w_2.

This approximation works well in practice, and only relies on standard min-cut solutions. The local improvement helps most if $\rho_e(E \setminus \{e\})$ is larger than zero for most edges (this does not hold, for instance, for truncated matroid rank functions), and if the low cost of A^* with respect to f is based on interactions of small sets of edges. We know that any edge in the optimal set A^* occurs in at least one basis cut. If $|A^*| > n$, then some basis cuts must contain more than one edge of A^*. The local step works if the edges in $C_i \cap A^*$ suffice to reduce the new weight $\rho_e(C_i)$ of any $e \in A^* \setminus C_i$ enough compared to the original weight $f(e)$.

3.4 Experiments

Our experiments use randomly generated polymatroid rank functions and two types of synthetic graphs: grids and loosely interconnected cliques. The graphs have between 50 and 100 edges. The algorithms were implemented in Matlab with the help of a graph cut [19] and SFO toolbox [20]. For baselines, we compare against the following three simple heuristics: (qu) Queyranne's algorithm [21], even though $g(X) = f(\delta X)$ is only subadditive so no guarantees exists; (mc) a modular min-cut with edge weights $w(e) = f(e)$; (gh) the f-evaluated best out of *all* the cuts in a Gomory-Hu tree built with modular edge weights as in (mc). Figure 2 shows the results, normalized by an estimate of the optimal cost and then averaged. Averages are over 72 instances for the clustered graphs and 101 instances for the grid graphs. Of the three suggested algorithms, (ghl) performs best on our instances. In (i) and (ii), $|A^*|$ does not differ much from the number of edges in a modular min-cut. In (iii), $|A^*|$ is chosen to be large, and roughly bi-partitions the graph – that is where the bound for (pf) becomes worse. Overall, the factors are good. Yet the min-cut-based comparison algorithms (mc), (gh) can reach their worst-case factor $|A^*| = n^2/4$ in a clique, and (qu) performs arbitrarily poorly on some graphs. The results in (iv) are on such graphs with 7 to 10 nodes – note the y axis. In these graphs, better bounds or the local improvement come into play: (acf), (pf), and (ghl) always find the best solution. Of the three suggested algorithms, the running time of (ghl) is most uniform. The cutting plane method in (acf) converges only slowly for some functions, even on less than 100 edges.

Acknowledgments: We wish to thank Jens Vygen for the intractable subadditive function example, and Andrew Guillory for his Gomory-Hu tree code.

References

[1] J. Bilmes and C. Bartels. On triangulating dynamic graphical models. In *Uncertainty in Artificial Intelligence (UAI)*, pages 47–56, Acapulco, Mexico, 2003. Morgan Kaufmann Publishers.

[2] B. Korte and J. Vygen. *Combinatorial Optimization - Theory and Algorithms*. Springer, 2008.

[3] E. L. Lawler and C. U. Martel. Computing maximal "Polymatroidal" network flows. *Mathematics of Operations Research*, 7(3):334–347, 1982.

[4] P. Zhang, Cai J.-Y, L.-Q. Tang, and Wen-Bo Zhao. Approximation and hardness results for label cut and related problems. *J Comb Optim*, 2009. (Online First).

[5] Z. Svitkina and L. Fleischer. Submodular approximation: Sampling-based algorithms and lower bounds. In *FOCS*, 2008.

[6] G. Goel, C. Karande, P. Tripati, and L. Wang. Approximability of combinatorial problems with mult-agent submodular cost functions. In *FOCS*, 2009.

[7] M. X. Goemans, N. J. A. Harvey, A. Iwata, and V. S. Mirrokni. Approximating submodular functions everywhere. In *SODA*, 2009.

[8] S. Iwata and K. Nagano. Submodular function minimization under covering constraints. In *FOCS*, 2009.

[9] J. Vondrák. Symmetry and approximability of submodular maximization problems. In *FOCS*, 2009.

[10] D. Wagner and F. Wagner. Between min cut and graph bisection. In *FOCS*, pages 744–750, 1993.

[11] I. Giotis and V. Guruswami. Correlation clustering with a fixed number of clusters. *Theory of Computing*, 2:249–266, 2006.

[12] S. Khot. Ruling out PTAS for graph min-bisection, densest subgraph and bipartite clique. In *Proceedings of the 45th Annual IEEE Symposium on FOCS*, pages 136–145, 2004.

[13] R. P. Stanley. *Enumerative Combinatorics*, volume I of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, 1997.

[14] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

[15] H. Narayanan. *Submodular Functions and Electrical Networks*. Elsevier Science, 1997.

[16] F. Bunke, H. W. Hamacher, F. Maffioli, and A. Schwahn. Minimum cut bases in undirected networks. *Discrete Applied Mathematics*, In Press, 2009.

[17] R. E. Gomory and T. Hu. Multi-terminal network flows. *Journal of the SIAM*, 9(4), 1961.

[18] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular functions - i. *Math. Program.*, 14:265–294, 1978.

[19] Shai Bagon. Matlab wrapper for graph cut, December 2006. `http://www.wisdom.weizmann.ac.il/~bagon`.

[20] Andreas Krause. Matlab toolbox for submodular function optimization, 2009. `http://www.cs.caltech.edu/~krausea/sfo/`.

[21] M. Queyranne. Minimizing symmetric submodular functions. *Mathematical Programming*, 82:3–12, 1998.