Approximation Algorithms Workshop June 13-17, 2011, Princeton

Lecture :Submodular Optimization — June 16, 2011

Pushkar Tripathi *Scribe: Pushkar Tripathi*

1 Overview

Applications in complex systems such as the Internet have spawned recent interest in studying situations involving multiple agents with their individual cost or utility functions. In this lecture, we introduce an algorithmic framework for studying combinatorial problems in the presence of multiple agents with submodular cost functions. We study several fundamental covering problems (Vertex Cover, Shortest Path, Perfect Matching, and Spanning Tree) in this setting and establish tight upper and lower bounds for the approximability of these problems.

As an example consider a setting where we have three agents each having a set function representing her costs for subsets of edges. The agents wish to collectively build a spanning tree for a given graph having the minimum cost. Note that this is quite easy to do if the agent's cost functions are linear i.e. the cost of any subset of edges is simply the cost of each of its elements. In this case this problem reduces to the classical minium spanning tree problem since for each edge we only need to consider the agent with the least cost.

However we wish to model economic incentives such as decreasing marginal cost and economy of scale and towards this end we will use submodular cost functions to model these properties. In this lecture we will discuss various issues involved in designing efficient approximation algorithms for combinatorial optimization over submodular cost functions.

1.1 General Framework

Throughout this talk we will use $G(V, E)$ to denote the given graph and use i o index the agents. The i^{th} agent's cost function will be denoted by f_i. We will make the following minimal assumptions over these functions.

1. **Normalized:** $f_i(\phi) = 0$
2. **Monotone:** $f_i(T) \geq f_i(S)$ for every $T \subseteq S$.
3. **Submodularity:** $f_i(T + e) - f_i(T) \geq f_i(S + e) - f_i(S)$ for every $T \subseteq S$ and $e \notin S$.

Intuitively the last property states that the incremental cost of adding an element e to a set S is smaller than the incremental cost of adding the same element to any of the subsets of S. This property captures the economic observation of decreasing marginal cost.

Let us define the following class of *combinatorial problems with multi-agent submodular cost functions* (MSCP) - We are given a set of elements X and a collection $C \subseteq 2^X$. this collection may be exponentially large and may be specified by a combinatorial propoerty. We are also given k agents,

where each agent i specifies a normalized monotone submodular cost function $f_i : 2^X \to R^+$. The goal is to find a set $S \in C$ and a partition $S_1, ..., S_k$ of S such that $\sum_i f_i(S_i)$ is minimized.

Remark: Since submodular functions are defined over an exponentially large domain we will assume value oracle access to them, i.e. we will assume that each of the functions is given by a value oracle that can be queried to get the value of any set.

1.2 Our Results

Table 1: Results

	Single-agent		Multi-agent	
	Lower bound	Upper bound	Lower bound	Upper bound
Vertex Cover	$2 - \epsilon$	2	$\Omega(\log n)$	$2 \log n$
Shortest Path	$\Omega(n^{2/3})$	$O(n^{2/3})$	$\Omega(n^{2/3})$	$O(n^{2/3})$
Perfect Matching	$\Omega(n)$	n	$\Omega(n)$	n
Spanning Tree	$\Omega(n)$	n	$\Omega(n)$	n

We studied the following four fundamental combinatorial optimization problems in the frameowrk described above. for each of the these problems we give optimal algorithms and matching lower bounds. Our lower bounds are information theoretic and our algorithms are based on techniques such as rounding of configurational LPs, approximating sumdodular functions and greedy.For each of the above problems, we study both the single agent and the multi-agent setting.

- **Submodular Vertex Cover (MS-VC)**: We are given an undirected graph $G(V, E)$. Element set X is the same as the set of vertices V and the collection C consists of all the vertex covers of the graph. Recall that a set $S \subseteq V$ is a vertex cover if every $e \in E$ is incident on a vertex in S.
- **Submodular Shortest Path (MS-SP)**: We are given a connected undirected graph $G(V, E)$, and a pair of vertices $s, t \in V$. Element set X is the same as the set of edges E and the collection C consists of all the paths from s to t.
- **Submodular Minimum Perfect Matchings (MS-MPM)**: In this setting, we have a undirected graph $G = (V, E)$ with cost functions over E. G contains at least one perfect matching. Element set X is the set of all edges, and the collection C is defined as the set of all perfect matchings of G. Recall that a set $M \subseteq E$ is a perfect matching of G if exactly one edge in M is incident on every vertex.
- **Submodular Minimum Spanning Tree (MS-MST)**: We are given a connected undirected graph $G = (V, E)$ with cost functions over E. Element set X is the set of all edges, and the collection C is the set of spanning trees of G. Recall that a spanning tree is a minimal connected subgraph of G.

1.3 Related Work

Submodular functions have been of great interest in Optimization in the have received considerable attention over the last past three decades. The most fundamental of them is, perhaps, the non-monotone submodular function minimization problem. A sequence of works in this direction [Sch00, IFF01, Iwa03, Orl07, Iwa08, IO09] has resulted in fast strongly polynomial time combinatorial algorithms. Another related work is that of non-monotone submodular function maximization [FMV07]. Both these algorithms are sometimes used as a subroutine in solving the configuration LPs corresponding to some other submodular combinatorial optimization problem.

Another body of work in optimization over submodular functions deals with welfare maximization [CCPV07, Von08, FV06, KLMM08]. Calinescu et al. [CCPV07] studied submodular function maximization subject to matroid constraints. They showed that their problem contains as a subcase many other allocation problems, thus giving a unified framework for studying such problems. Matching information theoretic lower bounds were established in [MSV08].

Very recently, Svitkina and Fleischer [SF08] studied submodular objective function for problems like Sparsest cut, load balancing, and knapsack. They gave $O(\sqrt{\frac{n}{\log n}})$ upper and lower bounds for all these problems, showing that all these problems become much more hard in the submodular setting. Some other related work in optimization that uses submodular functions include [SSW07, HST05, ST06, Svi04, Wol82].

2 Submodular Shortest Path

In this section we will discuss the submodular shortest path problem(MS-SP) defined above.

2.1 Algorithm for Submodular Shortest Path Problem

We will now briefly discuss an $O(n^{2/3})$ factor approximation algorithm for the Submodular shortest path problem for a single agent. The algorithm may be viewed as a combination of two different subroutines that work for different settings. Througout this section we will use f to denote the cost function for the agent and use OPT to denote the cost of the optimal s-t path.

Linear Approximation: As the first attempt let us try to approximate f by a linear cost function. We will use w_e to denote the cost of a edge e. The idea is that simply guessing the costliest edge in the optimal path gives a good enough approximation if the optimal path isnt too long. So suppose e^* is the costliest edge in the optimal path. Let us remove all edges that are costlier than e* from the graph to generate residual graph G'. Let P_1 be the s-t path with the fewest edges in G'. The following lemma bounds the cost of $P1$.

Lemma 1. *Cost*$(P_1) \leq$ *Diameter*$(G').OPT$

Proof. Observe that the path returned by our algorithm can be as long as the diameter of the residual graph. Since each edge in the residual graph is cheaper than w_e, which in turn is a lower bound on OPT, the total cost of P_1 is at most the w_e times the diameter of G'. This gives the desired bound. □

Note that in the worse case the diameter can be as large as $O(n)$ in which was this method is not a good approximation.

Ellipsoidal Approximation: We will now present another attempt towards approximating the cost of the optimal s-t path using an ellipsoidal approximation using John's theorem given below.

Theorem 2. *For every polytope P, there exists an ellipsoid contained in it that can be scaled by a factor of $O(\sqrt{n})$ to contain P.*

John's theorem is an inherently non-constructive statement about the structure of polytopes. Goemans et.al [GHIM09] studied a contructive version of this problem and showed that if the polytope is polymatroid then this ellipsoid can be found in polynomial time. In the context of submodular functions their result can be stated as follows.

Theorem 3. *For any monotone submodular function f over a ground set X there exists a polynomial time algorithm to compute an assignment $\chi : X \to R^+$ such that the function $g : 2^X \to R^+$ defined as $g(S) = \sqrt{\sum_{e \in S} \chi(e)}$ satisfies $g(S) \leq f(S) \leq \sqrt{|X|} g(S)$ for every $S \subseteq X$.*

In our setting X is the set of all edges in the given graph. Consider the following algorithm for finding the shortest s-t path. Begin by approximating the given function f by its ellipsoidal approximation g. Find the shortest s-t path with respect to g, say P_2, and return it as the solution. Note that this is easy to do due to the special structure of the ellipsoidal approximation. The following lemma bounds the cost of P_2.

Lemma 4. $f(P_2) \leq \sqrt{E} OPT$

Proof. The proof follows trivially by the properties of the ellipsoidal approximation and by the choice of g. □

Note that none of the above two techniques would work for a dense graph wih large diameter. Now we will present an algorithm that combines the insight obtained from the preceeding discussion to give a factor $O(n^{2/3})$ approximation algorithm.

$O(n^{2/3})$ **Algorithm:** The algorithm runs in 5 steps described below.

1. **Pruning** As before guess the costliest edge in the optimal path and prune away all edges costier than it.
2. **Contraction** Iteratively search for vertices with degree larger than $n^{1/3}$ and contract theor neighborhood.
3. **Ellipsoid Approximation** Calculate ellipsoidal approximation (d, g) for the residual graph.
4. **Search** Find shortest s-t path according to g in the residual graph.
5. **Reconstruction** Replace the path through each contracted vertex with the path having the fewest edges.

Analysis: Let P be the path returned by our algorithm and let R be the set of edges in the residual graph. Let us split the edges in the path returned by our algorithm in to two sets - $P_1 = R \cap P$ and $P_2 = (E/R) \cap P$. In lemma 5 and 6 we will bound the cost of P_1 and P_2 with respect to OPT.

Lemma 5. $f(P_1) \leq n^{2/3} OPT$

Proof. The first thing to note is that since the degree of each vertex in R is at most $n^{1/3}$ the total number of edges in R is $n^{4/3}$. So we have the following chain of inequalities that prove the claim.

$$\begin{aligned} f(P_1) &\leq \sqrt{E(R)}.g(P_1) && (1)\\ &\leq \sqrt{E(R)}.OPT && (2)\\ &\leq \sqrt{E(R)}.f(OPT) && (3)\\ &\leq n^{2/3} f(OPT) && (4) \end{aligned}$$

□

Lemma 6. $f(P_2) \leq n^{2/3} OPT$

Proof. To bound the cost of P2 we note that every vertex in the dense region has degree larger than $n^{1/3}$ thus the diameter of any of the dense regions is bounded by $Diameter(G_i) \leq |G_i|/n^{1/3}$. Since we pick the path with the fewest edges through each of the dense regions, in the worst case we will use as many edges as the diameter while bridging across a dense region. Summing over all dense regions we see that here too the cost of segment P_2 is at most $O(n^{2/3})$ times the optimal solution.

$$\begin{aligned} f(P_2) &\leq \sum_i diam(G_i) w_e^* && (5)\\ &\leq \sum_i |G_i|/n^{1/3} w_e^* && (6)\\ &\leq (n/n^{1/3}) w_e^* && (7)\\ &\leq n^{2/3} OPT && (8) \end{aligned}$$

□

3 Information Theoretic Lower Bounds

In this section we will discuss the information theoretic lower bound for the submodular shortest path problem. We begin by informally defining information theoretic lower bounds. Suppose we are have an algorithm that makes polynomially many queries to the value oracle for f. Also suppose the algorithm is allowed unbounded amount of time to process the results of these queries. If suppose we can show that even if given infinite amount of time that any algorithm can not get the optimal solution using only polynomially many queries then the problem is said to have an

information theoretic lower bound. Intuitively this means that it is not computation but the lack of information that is preventing us from getting the optimal solution. Note that such a bound is not contingent on P vs NP.

3.1 General Technique

Let us assume that we have two functions f and g which satisfy these two properties.

1. The optimal value of g is much larger than the optimal value of f.
2. f and g agree on most sets S i.e. For any randomized algorithm $\mathcal{A}$, f(Q) = g(Q) with high probabilty for every query Q made by $\mathcal{A}$. Here probability is calculated over random bits in the algorithm.

Also note that this must be true for every algorithm, not just a particular algorithm. We can use Yaos lemma to convert a statement over all randmized algorithms over deterministic input, to one over deterministic algorithms over distributions over inputs. Applying Yao's lemma to the second statement we find that it is equivalent to finding f and a distribution $\mathcal{D}$ from which we choose g, such that for an arbitrary query Q, $f(Q) = g(Q)$ with high probability. Before we jump in to the analysis for the lower bound let us do a warm up by deriving a lower bound while ignoring the combinatorial structure.

3.2 Non-combinatorial Setting

In this setting we prove the fololwing theorem.

Theorem 7. *It is information-theoretically hard to learn a submodular function to a factor better than $n^{1/2}/\log n$ in polynomial value queries.*

Proof. From the preceeding discussion in order to show a lower bound we need to define a function f and a distribution from which we choose function g. We define $f(S) = \min\{|S|, \alpha\}$. We choose g from the following distribution - Pick a random subset R of size α from X, and define the function $g_R(S) = \min\left\{|S \cap \bar{R}| + \min(S \cap R, \beta)\right\}$.

The constants α and β will be determined later. The next step is to show that these two functions are hard to distinguish. To do so we need to derive properties of the optimal query that has the largest probability of distinguishing f and g in particular we can show that the optimal query has to be of size α.

Finally we bound the probability of distinguishing f and g through a query of size α in the following lemma.

Lemma 8.

$\Pr[g_R(Q) < f(Q)] = \Pr[|Q \cap R| > \beta]$ □

Proof.

$$\begin{aligned}\Pr[g_{_R}(Q) < f(Q)] &= \Pr[g_{_R}(Q) < \alpha] \\ &= \Pr[|Q \cap \bar{R}| + \min\{|Q \cap R|, \beta\} < \alpha] \\ &= \Pr[|Q \cap R| > \beta]\end{aligned}$$

Setting $\alpha = \sqrt{n}\log n$ and $\beta = 1.001\log^2 n$ ensures that $|Q \cap R|$ is super-logarithmic and a simple application of chernoff bounds proves that f and g are hard to distinguish. Next note that $f(R) = \min\{|R|, \alpha\} = |R| = \alpha = n^{1/2}\log n$ and $g_R(R) = min|R \cap \bar{R}| + min(R \cap R, \beta) = \beta = \log^2 n$ Thus, even though f and g are similar on most queries they have vastly differing values on R. Thus it is hard to learn a submodular function to a factor better than $\alpha/\beta = \sqrt{n}/\log n$ which proves the theorem. □

3.3 Combinatorial Setting

Note that in the previous section we used a randomly chosen hidden set as a proxy for the optimal solution. If we try to extend these ideas to the combinatorial setting for obtaining lower bounds for the shortest path problem we immediately hit a road-block - randomly chosen sets of the given domain rarely form a feasible solution for most combinatorial problems. For example a randomly chosen set of edges rarely yields an s-t path in a graph. We circumvent these problem by the following two tricks.

1. Do not choose R randomly from the entire domain X, instead sample R uniformly at random froma subdomain of X.
2. Use a subset of R as a proxy for the solution.

Now we will put these ideas into action to prove the following theorem.

Theorem 9. *Submodular Shortest Path problem is hard to approximate to a factor better than* $O(n^{2/3})$

Proof. We consider the level graph with $n^{2/3}$ levels for our purposes. Level 1 has just s and the last level has just vertex t. Each of the other levels has $n^{1/3}$ vertices and there is a complete graph between successive levels. Let us partition the edges between alternate levels in to two disjoint sets Y and B. Define $f(S) = \min(|S \cap B|, \alpha)$ and define $g_R(S) = \min\{|S \cap R \cap B| + \min(S \cap R \cap B, \beta)\}$ where $R \subset B$ is chosen uniformly at random. We will use the second trick mentioned above as a guiding principle to determine the size of R. We want a subset of R to be a proxy for the optimal solution under function g. Thus if we set R to be $n^{2/3}\log^2 n$ then by standard coupon collector arguments we have an edge from R landing in each of the even levels. These edges together with the *free* edges in set Y can be used to build a s-t path having small cost β. Setting $\alpha = n^{2/3}\log^2 n$ and $\beta = \log^2 n$, we can show that f and g are indeed indistinguishable. The optimal solution with respect to f has cost $n^{2/3}$ where as the optimal solution with respect to g_R has small optimal cost $\beta = \log^2 n$. This proves the theorem. □

4 Discussion

Even though the results here are tight, the lower bounds are polynomially large and raise the question about the right model to study economies of scale. Note that the above model yielded information theoretic lower bounds which seems to suggest that these models are much to general and motivates us to consider succinctly representable submodular functions. In [GTW10] we considered the *discounted price model* that is a special case of the above model.

We define a function $d : \mathrm{R}^+ \to \mathrm{R}^+$ to be a discounted price function if it satisfies the following properties: (1) $d(0) = 0$; (2) d is increasing; (3) $d(x) \leq x$ for all x; (4) d is concave. We study combinatorial problems in the following general setting. We are given a set of elements E, and a collection Ω of its subsets. We are also given a set $\mathcal{A}$ of k agents where each agent $a \in \mathcal{A}$ specifies a cost function $c_a : E \to \mathrm{R}^+$ where $c_a(e)$ indicates her cost for the element e. Each agent also declares a discounted price function d_a. If an agent a is assigned a set of elements T, then her total price is specified by $d_a(\sum_{e \in T} c_a(e))$. This is called her *discounted price*. For ease of notation we will use $d_a(T)$ to denote $d_a(\sum_{e \in T} c_a(e))$. The objective is to select a subset S from Ω and a partition $S_1, S_2, ..., S_k$ of S, such that $\sum_{a \in \mathcal{A}} d_a(S_a)$ is minimized.

Note that the functions defined above are indeed submodular and can be represented in polynomially many bits. We considered various combinatorial problems in this setting and gave vastly improved bounds for them. Here our lower bounds were based on P vs NP, instead of being information theoretic. Finding newer models which lie between linear and submodular cost functions and establishing sharp approximability thresholds is an interesting avenue in this line of research.

References

[CCPV07] Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint (extended abstract). In *IPCO*, pages 182–196, 2007.

[FMV07] Uriel Feige, Vahab S. Mirrokni, and Jan Vondrák. Maximizing non-monotone submodular functions. In *FOCS*, pages 461–471, 2007.

[FV06] Uriel Feige and Jan Vondrák. Approximation algorithms for allocation problems: Improving the factor of 1 - 1/e. In *FOCS*, pages 667–676, 2006.

[GHIM09] Michel X. Goemans, Nicholas J. A. Harvey, Satoru Iwata, and Vahab S. Mirrokni. Approximating submodular functions everywhere. In *SODA*, pages 535–544, 2009.

[GTW10] Gagan Goel, Pushkar Tripathi, and Lei Wang. Optimal approximation algorithms for multi-agent combinatorial problems with discounted price functions. In *FSTTCS*, 2010.

[HST05] Ara Hayrapetyan, Chaitanya Swamy, and Éva Tardos. Network design for information networks. In *SODA*, pages 933–942, 2005.

[IFF01] Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *J. ACM*, 48(4):761–777, 2001.

[IO09] Satoru Iwata and James B. Orlin. A simple combinatorial algorithm for submodular function minimization. In *SODA '09: Proceedings of the Nineteenth Annual ACM -SIAM Symposium on Discrete Algorithms*, pages 1230–1237, 2009.

[Iwa03] Satoru Iwata. A faster scaling algorithm for minimizing submodular functions. *SIAM J. Comput.*, 32(4):833–840, 2003.

[Iwa08] Satoru Iwata. Submodular function minimization. *Math. Program.*, 112(1):45–64, 2008.

[KLMM08] Subhash Khot, Richard J. Lipton, Evangelos Markakis, and Aranyak Mehta. Inapproximability results for combinatorial auctions with submodular utility functions. *Algorithmica*, 52(1):3–18, 2008.

[MSV08] Vahab S. Mirrokni, Michael Schapira, and Jan Vondrák. Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. In *ACM Conference on Electronic Commerce*, pages 70–77, 2008.

[Orl07] James B. Orlin. A faster strongly polynomial time algorithm for submodular function minimization. In *IPCO*, pages 240–251, 2007.

[Sch00] A. Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *J. Comb. Theory, Ser. B*, 80(2):346–355, 2000.

[SF08] Zoya Svitkina and Lisa Fleischer. Submodular approximation: Sampling-based algorithms and lower bounds. In *FOCS*, pages 697–706, 2008.

[SSW07] Yogeshwer Sharma, Chaitanya Swamy, and David P. Williamson. Approximation algorithms for prize collecting forest problems with submodular penalty functions. In *SODA*, pages 1275–1284, 2007.

[ST06] Zoya Svitkina and Éva Tardos. Facility location with hierarchical facility costs. In *SODA*, pages 153–161, 2006.

[Svi04] Maxim Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Oper. Res. Lett.*, 32(1):41–43, 2004.

[Von08] Jan Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *STOC*, pages 67–74, 2008.

[Wol82] L. A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. In *Combinatorica*, pages 385–393, 1982.