CS599: Approximation Algorithms April 22nd, 2010

Pipage Rounding for Submodular Maximization

Lecturer: David Kempe *Scribe: Abhimanyu Das and Majed Alresaini*

1 Introduction

We continue with the problem of maximizing a submodular function $f(S)$ over the matroid $\mathcal{M} = (X, \mathcal{I})$. After having obtained a fractional solution $\vec{y} \in \mathcal{P}(\mathcal{M})$ using the continuous greedy process over the matroid polytope $\mathcal{P}(\mathcal{M})$, the next step is to round $\vec{y}$ to obtain an integral solution. For this we use the Pipage Rounding technique, that aims to gradually convert components of the $\vec{y}$ vector to integers, by taking fractional pairs y_i and y_j and increasing one while decreasing the other, until one of the elements becomes integral. However, one problem is that the vector might leave the polytope before any one of the i, j components become integral. To address this issue, we define the notion of a tight set.

Definition 1 *A set $A \subseteq X$ is* tight *with respect to $y \in \mathcal{P}(\mathcal{M})$ iff $\text{rank}_{\mathcal{M}}(A) = y(A)$.*

Here, we define $y(A) = \sum_{i \in A} y_i$. Recall that the definition of the rank function of a matroid is $\text{rank}_{\mathcal{M}}(A) := \max\{|S| : S \in \mathcal{I}, S \subseteq A\}$.

We have the following lemma, which shows that there is the notion of a minimal tight set.

Lemma 2 *If A, B are two tight sets, then $A \cup B$ and $A \cap B$ are also tight.*

Proof. Since A, B are tight, $y(A) = \text{rank}_{\mathcal{M}}(A)$ and $y(B) = \text{rank}_{\mathcal{M}}(B)$. Now, since the rank function is submodular, we have

$$\begin{aligned}
y(A) + y(B) &= \text{rank}_{\mathcal{M}}(A) + \text{rank}_{\mathcal{M}}(B) \quad \text{(by tightness of } A \text{ and } B\text{)} \\
&\geq \text{rank}_{\mathcal{M}}(A \cup B) + \text{rank}_{\mathcal{M}}(A \cap B) \quad \text{(by submodularity of rank function)} \\
&\geq y(A \cup B) + y(A \cap B) \quad \text{(since } \vec{y} \text{ is feasible)} \\
&\geq y(A) + y(B)
\end{aligned}$$

Hence, all the inequalities above must be equalities, and we get $\text{rank}_{\mathcal{M}}(A \cup B) = y(A \cup B)$ and $\text{rank}_{\mathcal{M}}(A \cap B) = y(A \cap B)$. ■

2 Pipage Rounding

The overall pipage rounding algorithm to round all components of $\vec{y}$ to $\{0, 1\}$ is as follows:

- While there still exist components of $\vec{y}$ with fractional values, do the following steps:
  1. Let $y_i, y_j \in (0, 1)$ and $T := X$ (i.e., the entire ground set). Note that existence of $y_i, y_j \in (0, 1)$ is guaranteed, since $\sum_k y_k = \text{rank}_{\mathcal{M}}(X)$, which is an integer. Hence, as long as $\vec{y}$ has fractional components, it must have at least two of them.
  2. Let δ^+ and δ^- be the maximum values such that $(y_i + \delta^+, y_j - \delta^+)$ and $(y_i - \delta^-, y_j + \delta^-)$ are both feasible. (It is possible that δ^+ or $\delta^- = 0$). Here we use the notation $(y_i + \delta^+, y_j - \delta^+)$ to refer to the vector obtained by replacing y_i with $y_i + \delta^+$, replacing y_j with $y_j - \delta^+$, and keeping the other components of $\vec{y}$ unchanged.
  3. As shown in the previous class, at least one of these above two vectors does not decrease the value of F. Without loss of generality, let this be $(y_i + \delta^+, y_j - \delta^+)$.

4. If one of $y_i + \delta^+$ or $y_j - \delta^+$ is integer, then we go back to step 1 (we have made progress by making at least one more component of $\vec{y}$ integer).

5. Otherwise, there must be a tight set A preventing us from using a larger δ^+. Because A prevents us from raising y_i further, $i \in A$, $j \notin A$.

6. Let $T' := T \cap A$. By lemma 2, T' is tight. Also $i \in T'$, and y_i is fractional. Hence, T' must contain another fractional component y_k (since $y(T') = \text{rank}_{\mathcal{M}}(T') \in \mathbb{N}$).

7. Go back to step 2 with $T := T'$ and $j := k$. (Note that the algorithm has made progress since $T' \subset T$ and hence we have a smaller tight set. Eventually we will reach the minimal tight set T_0, such that $T_0 \subseteq A$ for all tight sets A. At this point, we could then take two fractional components $y_i, y_j \in T_0$, and increase one and decrease the other until one of them becomes an integer, without being constrained by any smaller tight set).

To complete the algorithm, we need to specify how to find δ^+, δ^- and A in steps 2 and 5. Given the tight set $A \ni i$, one can easily find the corresponding δ^+ which when added to y_i will cause A to become tight: this is exactly $\text{rank}_{\mathcal{M}}(A) - y(A)$. Thus $\delta^+ = \text{rank}_{\mathcal{M}}(A) - y(A)$. The goal then is to find A, such that $i \in A, j \notin A$, minimizing $\text{rank}_{\mathcal{M}}(A) - y(A)$.

Since $\text{rank}_{\mathcal{M}}(A)$ is a submodular function and $y(A)$ is linear, $\text{rank}_{\mathcal{M}}(A) - y(A)$ is also a submodular function. Then, finding A involves minimizing a submodular function, which, by the result of Fleischer, Iwata and Fujishige, can be performed in polynomial time.

Remark 3 *Instead of picking one of the two directions for moving $\vec{y}$, one can also "round" randomly to either of $(y_i + \delta^+, y_j - \delta^+)$ or $(y_i - \delta^-, y_j + \delta^-)$; by careful choice of distribution, it can be shown that the expected value of the final solution is at least $F(\vec{y})$. (The advantage of this approach is that we do not need to evaluate F to decide which direction to move)*

3 Wrapping up loose ends

1. Evaluating F: At various places in the algorithm (e.g., in the continuous greedy process), we need to evaluate $F(\vec{y})$, for $\vec{y} \in [0,1]^n$. If $\vec{y}$ is integral (i.e., $\vec{y} \in \{0,1\}^n$), then $F(\vec{y}) = f(S)$ [with $\vec{y} = \vec{y_S}$], which we can evaluate easily, since we have a value oracle for f.

   However, for fractional $\vec{y}$, $F(\vec{y}) = \sum_S f(S) \prod_{i\in S} y_i \prod_{i \notin S}(1 - y_i)$. To evaluate this using the value oracle for f would take exponential oracle access. Instead, F is evaluated approximately by randomly sampling $\hat{y} \sim \vec{y}$ repeatedly, and taking the average. Then, one can show using Chernoff bounds that a $\text{poly}(n) \cdot \text{poly} - \log(1/\epsilon)$ number of samples are enough for a $(1 \pm \epsilon)$ approximation. One can then show, similar to the homework assignment, that the effect of approximating F on the quality of the final outcome is small.

2. Finding $\hat{v}(\vec{y}(t))$: In the continuous greedy process, we want to find a vector $\vec{v} \in \mathcal{P}(\mathcal{M})$ maximizing $\vec{v} \cdot \nabla F(\vec{y}(t))$. We showed earlier that $\vec{v}$ is a convex combination of bases, i.e., $\vec{v} = \sum_{\text{bases } S} \alpha_S \vec{y_S}$.

   Thus, $\vec{v} \cdot \nabla F(\vec{y}) = \sum_S \alpha_S(\vec{y_S} \cdot \nabla F(\vec{y}))$. This can be considered as a weighted average over $(\vec{y_S} \cdot \nabla F(\vec{y}))$, so for the best set S, $\vec{y_S} \cdot \nabla F(\vec{y}) \geq \vec{v} \cdot \nabla F(\vec{y})$. Thus the maximum value of $\vec{v} \cdot \nabla F(\vec{y})$ is achieved for some $\vec{y_S}$. Hence, it is enough to search over $\vec{v} = \vec{y_S}$ for some basis S.

   Now $\vec{y_S} \cdot \nabla F(\vec{y}) = \sum_{i \in S} \frac{\partial F}{\partial y_i}$. By considering $\frac{\partial F}{\partial y_i}$ as weights w_i, this can be seen to be a case of maximizing a linear function over bases of a matroid, which can be solved by Kruskal's greedy algorithm.

3. Discrete Time: We need to convert the continuous greedy method to a discretized time scale. Thus, we want to move from $\vec{y}_t$ to $\vec{y}_{t+1} = \vec{y}_t + \delta \cdot \hat{v}(\vec{y}_t)$ with $\delta > 0$. It can be shown that $\delta = \Theta(\frac{1}{n^2})$ is small enough to bound the discretization error ϵ, so that the approximation guarantee is $1 - \frac{1}{e} - \epsilon$. One can make $\epsilon \to 0$ by finer discretization and more sampling.