Presentation Notes for a Combinatorial algorithm for Submodular function minimization

Aditya Kannan

April 13, 2024

Abstract

We present a strongly polynomial time algorithm for minimizing submodular functions due to Schrijver [2].

1 Notation

We globally let V denote our ground set and $f\colon 2^V \to \mathbb{R}$ our submodular function. Let EO denote the time taken for evaluating f on an inuput.

Also $n := |V|$. For a total order $T = (V, \prec)$ on V, we denote by $S_v^{\prec}$ the section $\{w \in V : w \prec v\}$ for $v \in V$. For $s, u \in V$ with $s \prec u$, $\prec^{s,u}$ stands for the total order obtained by laying out the elements of V in a row according to $\prec$, and then displacing u to the place just before s. For a total order $\prec$ on V we let $b^{\prec}$ denote the extreme base generated by $\prec$

For $x \in \mathbb{R}^V$ and $U \subseteq V$, let $x(U)$ denote $\sum_{u \in U} x(u)$.

2 Subroutine for replacing extreme bases

Lemma 2.1. *Let $T = (V, \prec)$ be a total order on V and $s, t \in V$ with $s \prec t$. There exists a subroutine which writes $b^{\prec} + \delta(\hat{t} - \hat{s})$ (for some $\delta \geq 0$ chosen by the algorithm) as a convex combination of the extreme bases $b^{\prec^{s,u}}$ for $u \in (s, t]$ in $O(n^3 + n^2 EO)$ time. Here $\hat{t}$ is the unit vector in $\mathbb{R}^V$ in the direction of t.*

3 Subroutine for eliminating excess extreme bases

Lemma 3.1. *Suppose $x \in \mathbb{R}^n$ is written as the following convex combination:*

$$x = \sum_{i \in I} \lambda_i y_i$$

for $y_i \in \mathbb{R}^n$ and $\lambda_i \in [0,1]$ summing up to 1. There exists an algorithm running in $O(n|I|^3)$ time which outputs a $J \subseteq I$ with $|J| \leq n+1$ and $\sigma_i \in [0,1]$ summing to 1 such that

$$x = \sum_{i \in J} \sigma_i y_i$$

4 The Algorithm

First, place an arbitrary total order $<$ on V for the purpose of breaking ties

Translation by a constant preserves the submodularity of f so we are free to assume that $f(\varnothing) = 0$ henceforth.

In lieu of the min-max theorem of Edmonds, which states that

$$\min\{f(X) : X \subseteq V\} = \max\{x^-(V) : x \in B(f)\},$$

(here $x^-(v) = \min(x(v), 0)$) the algorithm intends to find a $W \subseteq V$ and an $x \in B(f)$ such that $f(W) = x^-(V)$. This W minimizes f.

We maintain at all times, an element

$$x = \lambda_1 b^{\prec_1} + \ldots \lambda_k b^{\prec_k} \tag{1}$$

of $B(f)$ as a convex combination of extreme bases $\lambda_i b^{\prec_i}$ for $1 \leq i \leq k$ with $k \leq n+1$. The algorithm is as follows:

Step 1 : Initialize x by choosing an arbitrary total order $\prec$ on V, computing the extreme base $b^{\prec}$ with respect to this total order and setting x to be that.

Step 2: Construct a directed graph $D := (V, A)$, with

$$A := \{(u, v) : u \prec_i v \text{ for some } 1 \leq i \leq k\}.$$

Define $P := \{v \in V : x(v) > 0\}$ and $N := \{v \in V : x(v) < 0\}$. If there is no directed path from P to N go to Step 3. Else jump to Step 4.

Step 3: Set W to be the set of vertices of D that can reach N via a directed path. Return W.

We claim that $f(W) = x^-(V)$. Indeed, for every $1 \leq i \leq k$, W is a lower set of $(V, \prec_i)$, i.e. if $v \in W$ and $u \prec_i v$, then $u \in W$ as well. This means that W is of the form $S_v^{\prec_i}$ for some $v \in V$. Thus $b^{\prec_i}(W) = f(W)$ by expanding out the telescoping sum. As a result of (1), $x(W) = f(W)$. But $W \cap P = \varnothing$, so $x^-(V) = x(W) = f(W)$.

Step 4: For $v \in V$ let $d(v)$ denote the distance of v from P in D. Let t be the element in N reachable from P which also maximizes $d(t)$. Break ties by picking the maximum element. Let $s \in V$ be the maximum element such that $(s, t) \in A$ and $d(s) = d(t) - 1$. Let α be the maximum size of $(s, t]_{\prec_i}$ across all $1 \leq i \leq k$. By reordering if necessary, assume that $(s, t]_{\prec_1} = \alpha$.

Invoke the subroutine in Lemma (2.1) to get a $\delta \geq 0$ and write $b^{\prec_1} + \delta(\hat{t} - \hat{s})$ as a convex combination of $b^{\prec_1^{s,u}}$ for $u \in (s,t]_{\prec_1}$. Then (1) gives us

$$y := x + \lambda_1 \delta(\hat{t} - \hat{s}) \tag{2}$$

as a convex combination of $b^{\prec_i}$ for $2 \leq i \leq k$ and $b^{\prec_1^{s,u}}$ for $u \in (s,t]_{\prec_1}$. Intersect the line segment joining x and y with the hyperplane $x(t) = 0$. Note that $t \in N$ so the intersection is either a single point or empty. If it's empty, set $x' := y$ or otherwise set x' to be the point of intersection. Note that x' is the point in the line segment closest to y such that $x'(t) \leq 0$.

We get x' in the form of a convex combination of $b^{\prec_i}$ for $1 \leq i \leq k$ and $b^{\prec_1^{s,u}}$ for $u \in (s,t]_{\prec_1}$. If $x'(t) < 0$ (so $x = y$) then there is no instance of $b^{\prec_1}$ in the convex combination.

The number of terms in the convex combination for x' might exceed n at the moment. So relabel the vectors and write

$$x' = \sum_{i \in I} \lambda_i b^{\prec_i}. \tag{3}$$

Invoke Lemma (3.1) on (3). Note that $|I| = O(n)$. Set $x \leftarrow x'$ and jump to Step 2.

References

[1] S. Iwata: A fully combinatorial algorithm for submodular function minimization, J. Combin. Theory, B84 (2002), 203–212.

[2] A. Schrijver: A combinatorial algorithm minimizing submodular functions in strongly polynomial time, J. Combin. Theory, B80 (2000), 346–355