Non-monotone DR-Submodular Function Maximization

Submodularity and Diminishing Return Property

A function $f : 2^E \to \mathbb{R}$ is submodular if

$$f(X) + f(Y) \geq f(X \cap Y) + f(X \cup Y) \text{ for every } X, Y \subseteq E.$$

Equivalent to the diminishing return property:

$$f(X \cup \{e\}) - f(X) \geq f(Y \cup \{e\}) - f(Y) \text{ for every } X \subseteq Y \subseteq E \text{ and } e \in E \setminus Y.$$

Submodular Function Maximization (SFM)

For a submodular function $f : 2^E \to \mathbb{R}_+$,

$$\text{maximize } f(X) \quad \text{subject to} \quad X \subseteq E.$$

- Double greedy [BFNS12] achieves (tight) 1/2-approximation in polynomial time.
- Applications in many AI/ML tasks

Example: Revenue maximization

- E: Users in a social network service.
- For $X \subseteq E$, we offer for free a product to users corresponding to X.
- $f(X)$: the expected # of new users who become an advocate of a product through the word-of-mouth effect.

Extension to Integer Lattice

Motivation

- In revenue maximization, we may want to decide how much budget should be set aside for each user.
- Extend the domain from 2^E to $\mathbb{Z}_+^E$!

A function $f : \mathbb{Z}_+^E \to \mathbb{R}$ is DR-submodular (or has the diminishing return property) if

$$f(\boldsymbol{x} + \chi_e) - f(\boldsymbol{x}) \geq f(y + \chi_e) - f(\boldsymbol{y}) \text{ for every } \boldsymbol{x} \leq \boldsymbol{y} \text{ and } e \in E.$$

- Stronger than lattice-submodularity: $f(\boldsymbol{x}) + f(\boldsymbol{y}) \geq f(\boldsymbol{x} \vee \boldsymbol{y}) + f(\boldsymbol{x} \wedge \boldsymbol{y})$.
  - $\vee, \wedge$: coordinate-wise max and min.

Our Contributions

(i) Polynomial-time approximation algorithm for DR-submodular function maximization (DR-SFM):

$$\text{maximize } f(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{0} \leq \boldsymbol{x} \leq \boldsymbol{B} \text{ for } \boldsymbol{B} \in \mathbb{R}_+^E.$$

- Approximation ratio: $1/(2+\epsilon)$.
  - Cannot be better than 1/2.
  - No constant-factor approximation if we only assume lattice-submodularity.
- Time complexity: $\widetilde{O}\left(\frac{|E|}{\epsilon} \log \frac{\Delta}{\delta} \log \|\boldsymbol{B}\|_\infty \cdot (\theta + \log \|\boldsymbol{B}\|_\infty)\right)$.
  - θ: time of evaluating f.
  - δ: minimum positive marginal gain of f.
  - Δ: maximum positive value of f.

(ii) Experimentally confirm the superiority against other baseline methods on revenue maximization using real-world networks.

[EN16] independently found an algorithm with a better time complexity by reducing DR-SFM to SFM.

Pseudo-polynomial Time Algorithm (DG)

A naive extension of double greedy

$\boldsymbol{x} \leftarrow \boldsymbol{0}, \boldsymbol{y} \leftarrow \boldsymbol{B}$.
for $e \in E$ **do**
  while $\boldsymbol{x}(e) < \boldsymbol{y}(e)$ **do**
    $\alpha \leftarrow f(\chi_e \mid \boldsymbol{x})$ $(:= f(\chi_e + \boldsymbol{x}) - f(\boldsymbol{x}))$ and $\beta \leftarrow f(-\chi_e \mid \boldsymbol{y})$.
    if $\beta < 0$ **then** $\boldsymbol{x}(e) \leftarrow \boldsymbol{x}(e) + 1$.
    if $\alpha < 0$ **then** $\boldsymbol{y}(e) \leftarrow \boldsymbol{y}(e) - 1$.
    else $\boldsymbol{x}(e) \leftarrow \boldsymbol{x}(e) + 1$ w.p. $\frac{\alpha}{\alpha+\beta}$ and $\boldsymbol{y}(e) \leftarrow \boldsymbol{y}(e) - 1$ w.p. $\frac{\beta}{\alpha+\beta}$.
return $\boldsymbol{x}$.

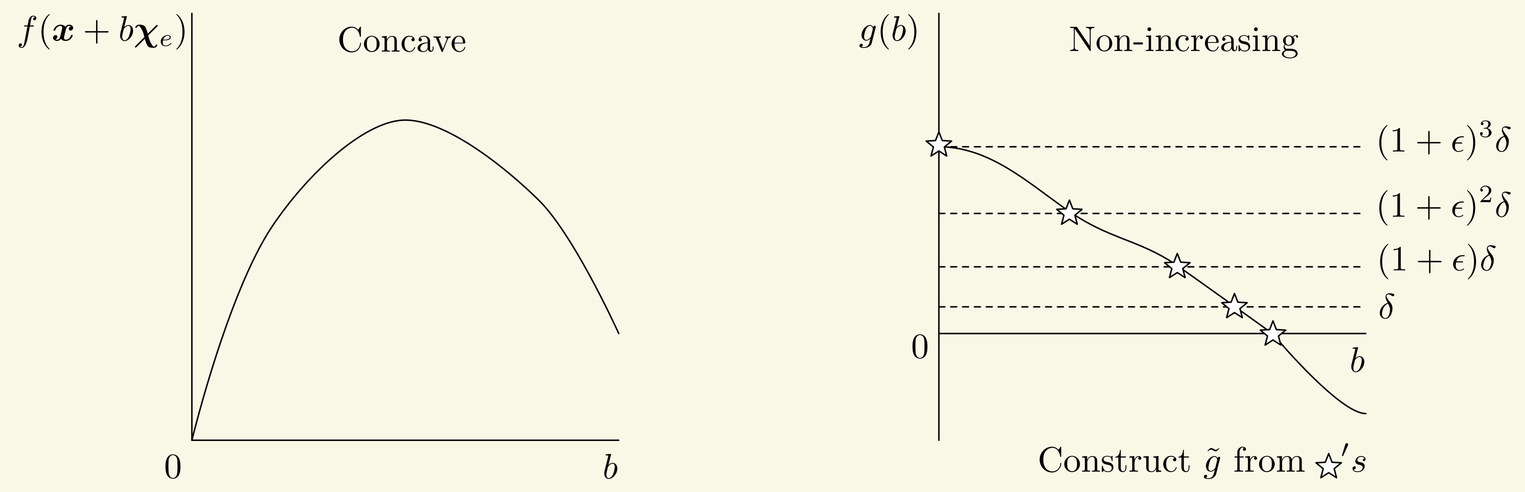
Guarantee

- 1/2-approximation.
- Time complexity: $O(\|\boldsymbol{B}\|_1 \cdot (1+\theta)) \Rightarrow$ Pseudo-polynomial.

Polynomial-time Algorithm (Fast-DG$_\epsilon$)

Idea

- Approximations to $f(\chi_e \mid \boldsymbol{x})$ and $f(-\chi_e \mid \boldsymbol{y})$ are enough to achieve an approximation ratio close to 1/2.
- Let $g(b) := f(\chi_e \mid \boldsymbol{x} + b\chi_e)$ and $h(b) := f(-\chi_e \mid y - b\chi_e)$.



Fast-DG$_\epsilon$

- Compute a compact representation $\tilde{g}$ (resp., $\tilde{h}$) from which we can approximate $(1 \pm \epsilon)g(b)$ (resp., $(1 \pm \epsilon)h(b)$).
- Then, simulate DG by using $\tilde{g}$ and $\tilde{h}$.

Guarantee

- Approximation ratio: $1/(2+\epsilon)$.
- Time complexity: $O(\frac{|E|}{\epsilon} \cdot \log(\frac{\Delta}{\delta}) \log \|\boldsymbol{B}\|_\infty \cdot \theta + \|\boldsymbol{B}\|_1 \log \|\boldsymbol{B}\|_\infty)$
  $\Rightarrow$ Still pseudo-polynomial but the number of oracle calls is polynomial.

Time complexity can be made polynomial by making large steps in the while loop.

Experiments

Revenue maximization

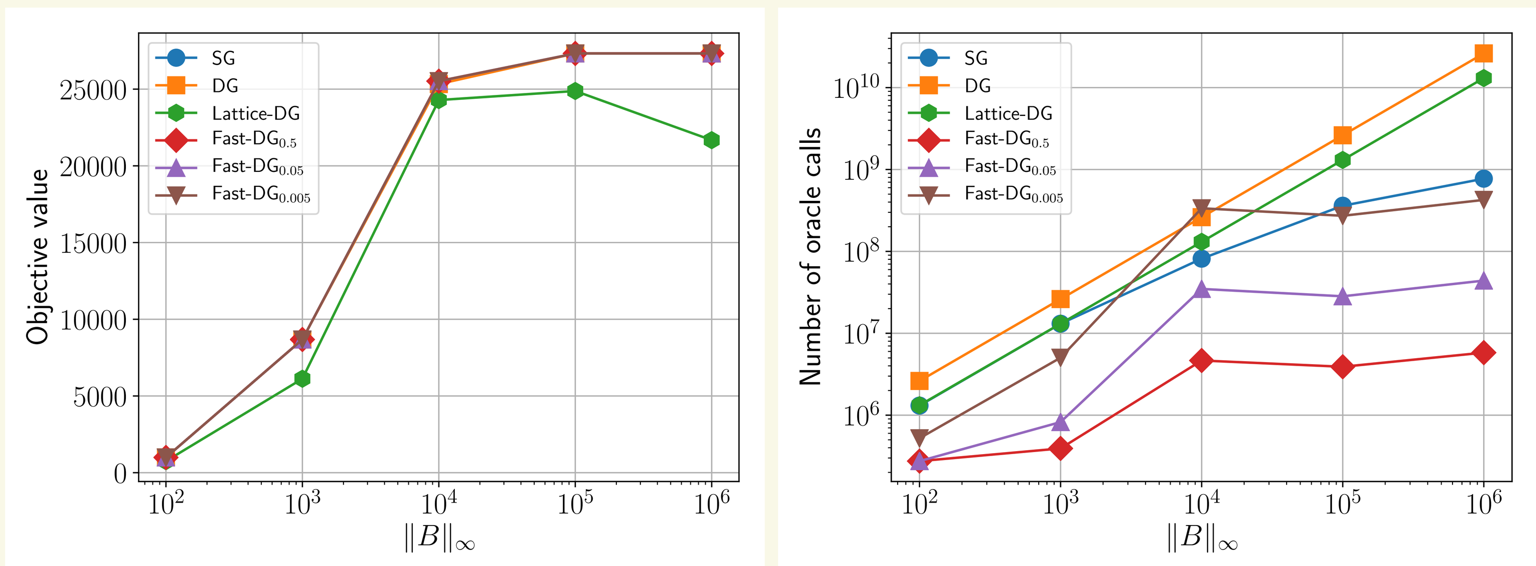
- Input: a weighted graph $G = (V, \{w_{ij}\}_{i,j \in V})$ and $p \in [0,1]$.
- If we invest $x \in \mathbb{Z}_+$ units of cost on a user, the user becomes an advocate of the product w.p. $1 - (1-p)^x$.
- The revenue is $\sum_{i \in S} \sum_{j \in V \setminus S} w_{ij}$, where S is a (random) set of advocates.
- Define DR-submodular function $f : \mathbb{Z}_+^V \to \mathbb{R}$ as the expected revenue:

$$f(\boldsymbol{x}) = \mathop{\mathbf{E}}_S\Big[\sum_{i \in S} \sum_{j \in V \setminus S} w_{ij}\Big] = \sum_{i \in S} \sum_{j \in V \setminus S} w_{ij}(1 - (1-p)^{\boldsymbol{x}(i)})(1-p)^{\boldsymbol{x}(j)}.$$

Settings

- Graph: Advogato (6,541 vertices and 61,127 edges)
- Baseline methods: Single Greedy (SG) and Lattice-DG [GP15].

Results



(a) Objective values **(b)** Number of oracle calls

Fast-DG$_{0.5}$ outperforms others:

- Achieves almost the best objective value.
- The number of oracle calls slowly grows and is two or three orders of magnitude smaller when $\|\boldsymbol{B}\|_\infty$ is large.

Future Directions

- Maximization under cardinality/polymatroid/knapsack constraint.
- Continuous analogue of a submodular function $f : [0,1]^E \to \mathbb{R}_+$ [BMBK16].
  - Can be seen as a limit of DR-submodular functions with $\|\boldsymbol{B}\|_\infty \to \infty$.

[BFNS12] N. Buchbinder, M. Feldman, J. S. Naor, and R. Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. In *FOCS*, pages 649–658, 2012.

[BMBK16] Y. Bian, B. Mirzasoleiman, J. M. Buhmann, and A. Krause. Guaranteed non-convex optimization: Submodular maximization over continuous domains. *CoRR*, abs/1606.05615, 2016.

Tasuku Soma[†] **Yuichi Yoshida**[‡]

tasuku_soma@mist.i.u-tokyo.ac.jp, yyoshida@nii.ac.jp

[†]The University of Tokyo. [‡]National Institute of Informatics *and* Preferred Infrastructure, Inc.