Bounds on Double-Sided Myopic Algorithms for Unconstrained Non-monotone Submodular Maximization

Norman Huang and Allan Borodin

University of Toronto

Abstract. Unconstrained submodular maximization captures many NP-hard combinatorial optimization problems, including MAX-CUT, MAX-DICUT, and variants of facility location problems. Recently, Buchbinder *et al.* [4] presented a surprisingly simple linear time randomized *greedy-like* online algorithm that achieves a constant approximation ratio of $\frac{1}{2}$, matching optimally the hardness result of Feige *et al.* [10]. Motivated by the algorithm of Buchbinder *et al.*, we introduce a precise algorithmic model called *double-sided myopic algorithms*. We show that while the algorithm of Buchbinder *et al.* can be realized as a randomized online double-sided myopic algorithm, no such deterministic algorithm, even with adaptive ordering, can achieve the same approximation ratio. With respect to the MAX-DICUT problem, we relate the Buchbinder *et al.* algorithm and our myopic framework to the online algorithm and inapproximation of Bar-Noy and Lampis [2].

1 Introduction

Submodularity emerges in natural settings such as economics, algorithmic game theory, and operations research; many combinatorial optimization problems can be abstracted as the maximization/minimization of submodular functions. In particular, submodular maximization generalizes NP-hard problems such as MAX-(DI)CUT [1], [14, 15], MAX-COVERAGE [8], expected influence in a social network [17] and facility location problems [6, 7]. As such, a number of approximation heuristics have been studied in the literature. For monotone submodular functions, maximization under a cardinality constraint can be achieved greedily with an approximation ratio of $(1-\frac{1}{e})$ [20], which is in fact optimal in the *value oracle model* [19]. The same approximation ratio is obtainable for the more general matroid constraints [5], [12], as well as knapsack constraints [24], [18]. We limit our discussion to unconstrained non-monotone submodular maximization (USM) — typical examples of which include MAX-CUT and MAX-DICUT. We refer the reader to our paper version [16] for a more comprehensive discussion on related works.

Recently, linear time (linear in counting one step per oracle call) *double-sided greedy* algorithms were developed by Buchbinder *et al.* [4] for the general USM. The deterministic version of their algorithm achieves an approximation ratio of

$\frac{1}{3}$, while the randomized version achieves $\frac{1}{2}$ in expectation - improving upon the $\frac{2}{5}$ randomized local-search approach in [10], and the 0.42 simulated-annealing technique in [13], [11], in terms of approximation ratio, time complexity and arguably, algorithmic simplicity. While the hardness result of Feige *et al.* [10] implies optimality of the randomized algorithm, the gap between the deterministic and randomized variants remains an open problem. More specifically, is there any de-randomization that would preserve both the greedy aspect of the algorithm as well as the approximation? To address this question, we adapt the priority algorithms framework of Borodin *et al.* [3]. Specifically, we define a *double-sided myopic algorithms* framework that captures the Buchbinder *et al.* algorithms; and show that no deterministic algorithm in this framework can de-randomize the $\frac{1}{2}$-ratio double-sided greedy algorithm.

In addition to [4], Bar-Noy and Lampis [2] give a $\frac{1}{3}$ deterministic online greedy algorithm for MAX-DICUT matching the deterministic approximation obtained by Buchbinder *et al.* for USM. They also give an improved $\frac{2}{3\sqrt{3}}$ approximation for MAX-DICUT on DAGs, and a precise online model with respect to which this approximation is optimal. Feige and Jozeph [9] introduce *oblivious online algorithms* for MAX-DICUT and give a 0.483 approximation and a 0.4899 inapproximation for randomized oblivious algorithms. Independent of our work, Paul, Poloczek and Williamson [22] have very recently derived a number of deterministic algorithms, and deterministic and randomized inapproximations for MAX-DICUT with respect to the priority algorithm framework.

Algorithm 1 DeterministicUSM($f, \mathcal{N}$)

```
S_0 ← ∅, T_0 ← N
for i = 1 to n do
    a_i ← f(S_{i-1} ∪ {u_i}) − f(S_{i-1})
    b_i ← f(T_{i-1} \ {u_i}) − f(T_{i-1})
    if a_i ≥ b_i then
        S_i ← S_{i-1} ∪ {u_i}, T_i ← T_{i-1}
    else
        T_i ← T_{i-1} \ {u_i}, S_i ← S_{i-1}
    end if
end for
return S_n
```

1.1 Basic Definitions

A set function $f : 2^{\mathcal{N}} \to \mathbb{R}$ is *submodular* if for any $S, T \subseteq \mathcal{N}$, $f(S \cup T) + f(S \cap T) \leq f(S) + f(T)$. We say f is *monotone* if $f(S) \leq f(T)$ for all $S \subseteq T \subseteq \mathcal{N}$, and *non-monotone* otherwise. In USM, the goal is to find a subset $S \subseteq \mathcal{N} = \{u_1, ..., u_n\}$ maximizing $f(S)$ for a specified submodular function f. In general, since the specification of f requires knowing its value on all possible subsets, f

is accessed via a value oracle which given $X \subseteq \mathcal{N}$, returns $f(X)$. We state the $\frac{1}{3}$ deterministic double greedy algorithm by Buchbinder *et al.* in Algorithm 1.

1.2 Our Contribution

We introduce a formalization of *double-sided myopic algorithms* - an adaptation of the priority framework under a *restricted* value oracle model. To make this precise, we will introduce three types of *relevant oracle queries*. In Proposition 1 and 2 we show that the double-sided greedy algorithms of Buchbinder *et al.* can be realized as online double-sided myopic algorithms. Moreover, our framework also captures the online model of Bar-Noy and Lampis for Max-Dicut, upon which Theorem 1 follows essentially by definition.

Algorithm 2 OnlineMyopic($f, \bar{f}$)

$X_0 \leftarrow \emptyset, Y_0 \leftarrow \emptyset$
for $i = 1$ to n **do**
  $a_i \leftarrow f(X_{i-1} \cup \{u_i\}) - f(X_{i-1})$
  $b_i \leftarrow \bar{f}(Y_{i-1} \cup \{u_i\}) - \bar{f}(Y_{i-1})$
  if $a_i \geq b_i$ **then**
    $X_i \leftarrow X_{i-1} \cup \{u_i\}, Y_i \leftarrow Y_{i-1}$
  else
    $Y_i \leftarrow Y_{i-1} \cup \{u_i\}, X_i \leftarrow X_{i-1}$
  end if
end for
return X_n

Proposition 1. *The deterministic double greedy algorithm in [4] can be modeled by Algorithm 2, a Q-Type 1 online double-sided myopic algorithm.*

Proposition 2. *The randomized double greedy algorithm in [4] can be modeled by a random-choice Q-Type 1 online double-sided myopic algorithm.*

Theorem 1. *No deterministic online double-sided myopic algorithm (with respect to our strongest oracle model Q-Type 3) can achieve a competitive ratio of* $\frac{2}{3\sqrt{3}} + \epsilon \approx 0.385 + \epsilon$ *for any* $\epsilon > 0$ *for USM.*

As our main contribution, we extend to the class of (deterministic) fixed and adaptive priority algorithms in Theorem 2 and 3. We construct via linear programming submodular functions and corresponding adversarial strategies that would force an inapproximability ratio strictly less than $\frac{1}{2}$. In terms of oracle restrictiveness, our inapproximation holds for the *all subsets query* model (Q-Type 3) for the fixed case and the *already attained partial solution query* model (Q-Type 2), which is more powerful than what is sufficient to achieve the $\frac{1}{2}$ ratio by the online randomized greedy algorithm (Q-Type 1). Due to space constraint we refer to [16] for missing proofs as well as further discussion.

Theorem 2. *There exists a problem instance such that no fixed priority double-sided myopic algorithm with respect to oracle model Q-Type 3 can achieve an approximation ratio better than 0.450.*

Theorem 3. *There exists a problem instance such that no adaptive priority double-sided myopic algorithm with respect to oracle model Q-Type 2 can achieve an approximation ratio better than 0.432.*

2 The Double-Sided Myopic Algorithms Framework

By integrating a *restricted* value oracle model in a priority framework, we propose a general class of *double-sided myopic algorithms* that captures the Buchbinder *et al.* double-sided greedy algorithm. Using a pair of complementary objective functions, we rephrase the double-sided procedure (*i.e.* simultaneously evolving a bottom-up and a top-down solution) as a single sided sweep common to most greedy algorithms; and this facilitates an adaptation of a priority-like framework. We must also specify how input items are represented and accessed. The generality of USM raises some representational issues, which we address by employing a *marginal value* representation that is compatible with both the value oracle model and the priority framework.

While a precise description of a data item is necessary in determining the input ordering and in quantifying the availability of information in the decision step, the value oracle model measures complexity in terms of information access. An apparent incompatibility arises when an exact description of a data item can trivialize query complexity - as in the case of the *marginal value* representation, where exponentially many queries are needed to fully describe an item when f is an arbitrary submodular function. For this reason, we propose a hierarchy of oracle restrictions that categorizes the concept of myopic *short-sightedness*, while preserving the fundamental characteristics of the priority framework.

2.1 Value Oracle and the Marginal Value Representation

Since f may not have a succinct encoding, to avoid having an exponential sized input we employ a *value oracle*. That is, given a query $S \subseteq \mathcal{N}$, the value oracle returns the value of $f(S)$. Abusing notation, we will interchangeably refer to f as the objective function (*i.e.* when referring to $f(S)$ as a real number) and as the value oracle (*i.e.* when an algorithm submits a query to f).

We also introduce for notational convenience a complementary oracle $\bar{f}$, such that $\bar{f}(X) = f(\mathcal{N} \setminus X)$. This allows us to express the double-sided myopic algorithm similar to the priority setting in [3], where the solution set X is constructed using only locally available information. In other words, the introduction of $\bar{f}$ allows access to $f(\mathcal{N} \setminus X)$ using X (which is composed of items that have already been considered) as query argument, instead of $\mathcal{N} \setminus X$. The **double-sidedness** of the framework follows in the sense that f and $\bar{f}$ can be simultaneously accessed.

We wish to model *greedy-like* algorithms that process the problem instance item by item. But what is an item when considering an arbitrary submodular function? While the natural choice is that an item is an element of $\mathcal{N}$ (to include or not include in the solution S), the input of USM is the objective function itself, drawn from the family $\mathcal{F} = \{f | f : 2^{\mathcal{N}} \to \mathbb{R}\}$ of all submodular set functions over a fixed $\mathcal{N}$. To address this issue, we propose the *marginal value representation*, in which f is instilled into the elements of $\mathcal{N}$. Specifically, we describe a data item as an element $u \in \mathcal{N}$, plus a list of marginal differences $\rho(u|S) = f(S \cup \{u\}) - f(S)$, and $\bar{\rho}(u|S) = \bar{f}(S \cup \{u\}) - \bar{f}(S)$ for every subset $S \subseteq \mathcal{N}$. A complete representation in this form is not only exponential in $|\mathcal{N}|$, it leads to a trivial optimal greedy algorithm. Therefore we assume that an oracle query must be made by the algorithm in order to access each marginal value. However, if certain natural constraints are imposed on allowable oracle queries during the computation, then the model allows us to justify when input items are indistinguishable. In fact, our inapproximation results will not be based on bounding of the number of oracle queries but rather by the restricted myopic nature of the algorithm.

2.2 Classes of Relevant Oracle Queries

The **myopic** condition of our framework is imposed both by the nature of the ordering in the priority model, which we describe later on, as well as by restricting the algorithm to make only *relevant* oracle queries. To avoid ambiguity, assume the oracles are given in terms of f and $\bar{f}$, with ρ and $\bar{\rho}$ being used purely for notational simplicity. That is, when we say that the algorithm queries $\rho(u|S)$, we assume that it queries both $f(S \cup \{u\})$ and $f(S)$. At iteration i, define (respectively) X_{i-1} and Y_{i-1} to be the currently accepted and rejected sets, and u_i to be the next item considered by the algorithm. We introduce three models of relevant oracle queries in order of increasing information.[1]

Next attainable partial solution query (**Q-Type 1**)
The algorithm is permitted to only query $f(X_{i-1} \cup \{u_i\})$ and $\bar{f}(Y_{i-1} \cup \{u_i\})$, *i.e.* the values of the next possible partial solution. In all models, the algorithm can then use this information in any way. For example, the deterministic algorithm of Buchbinder *et al.* greedily chooses to add u_i to X_{i-1} if $\rho(u_i|X_{i-1}) = a \geq b = \bar{\rho}(u_i|Y_{i-1})$. In our model, the decision about u_i can be any (even non-computable) function of a and b and the "history"of the algorithm thus far.

Already attained partial solutions query (**Q-Type 2**)
In this model we allow queries of the form $\rho(u_i|X_j)$ and $\bar{\rho}(u_i|Y_j)$ for all $j < i$.

All subsets query (**Q-Type 3**)
Here, the algorithm can query $\rho(u_i|S)$ and $\bar{\rho}(u_i|S)$ for every $S \subseteq X_{i-1} \cup Y_{i-1}$. Note that in this very general model, the algorithm can potentially query exponentially many sets so that in principle such algorithms are not subject to the $\frac{1}{2}$ inapproximation result of Feige et al [9].

[1] See Section 5 for further motivation.

2.3 Internal Memory or History

We define an algorithm's *internal memory* or *history* as a record of the following:

- All previously considered items and the decisions made for these items.
- The outcomes of all previous relevant query results.
- Anything deducible from previously considered items, decisions and relevant queries. Consequently, the algorithm can rule out all submodular functions contradicting the observed marginal values for previously considered items.

At the start of iteration i, allow the algorithm to perform any possible relevant queries. Let $\mathcal{N}_{i-1} = X_{i-1} \cup Y_{i-1}$ be the set of all previously seen (and decided upon) items. The algorithm's internal memory under each query restriction type contains the following:

Q-Type 1 myopic model
- The decision made for every $u \in \mathcal{N}_{i-1}$.
- $\rho(u_j|X_{j-1})$ and $\bar{\rho}(u_j|Y_{j-1})$ for $0 < j \leq i$.

Q-Type 2 myopic model
- The decision made for every $u \in \mathcal{N}_{i-1}$.
- $\rho(u_j|X_k)$ and $\bar{\rho}(u_j|Y_k)$ for $0 \leq k < j \leq i$.

Q-Type 3 myopic model
- The decision made for every $u \in \mathcal{N}_{i-1}$.
- $\rho(u|S)$ and $\bar{\rho}(u|S)$ for all $u \in \mathcal{N}_i$ and all $S \subseteq \mathcal{N}_i$.

2.4 Priority Models

Finally, we specify the order in which input items are considered. We categorize double-sided myopic algorithms into the following subclasses: **online**, **fixed** priority, and **adaptive** priority. For all templates, the algorithm makes an irrevocable decision for the current item based on the history of relevant queries. Initially, $|\mathcal{N}|$ is the only accessible information. Revealing the input length allows for a broader and potentially more powerful class of algorithms compared to the priority framework. That is, an adversary cannot abruptly end a computation. We begin with a description of the online model.

Online 2-Sided Myopic Template

while not empty($\mathcal{N}$)

next := lowest index (determined by adversary) of items in $\mathcal{N}$
Relevant Query: Perform any set of relevant queries and update the *internal memory*
Decision: As a function of the *internal memory*, irrevocably accept or reject u_{next}, and remove u_{next} from $\mathcal{N}$

A *priority function* is any injection $\pi : Q(u) \to \mathbb{R}$, where $Q(u)$ is the vector of item u's marginals accessible under the appropriate relevant query model. The priority function determines the ordering of the input items. Specifically, at each iteration the item $u_{next} \in \mathcal{N}$ with the highest priority (*i.e.* $u_{next} = \text{argmin}_{v \in \mathcal{N}}[\pi(Q(v))]$) is given to the algorithm. This leads to the more general models of fixed and adaptive priority algorithms.

Fixed Priority 2-Sided Myopic Template

Ordering: Specify a priority function $\pi : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$
while not empty($\mathcal{N}$)

next := index i of the item in $\mathcal{N}$ that minimizes $\pi(\rho(u_i|\emptyset), \bar{\rho}(u_i|\emptyset))$
Relevant Query: Using u_{next} as the next item, perform any set of relevant queries and update the *internal memory*
Decision: As a function of the *internal memory*, irrevocably accept or reject u_{next}, remove u_{next} from $\mathcal{N}$ and update internal memory.

Adaptive Priority 2-Sided Myopic Template

while not empty($\mathcal{N}$)

Ordering: Given the *internal memory*, specify a priority function π
next := index i of the item in $\mathcal{N}$ that minimizes $\pi(Q(u_i))$
Relevant Query: Using u_{next} as the next item, perform any set of relevant queries and update the *internal memory*
Decision: As a function of the *internal memory*, irrevocably accept or reject u_{next}, remove u_{next} from $\mathcal{N}$ and update internal memory.

A fixed priority algorithm determines π before it makes any oracle queries, and this ordering remains unchanged. For all relevant query models, $Q(u)$ initially corresponds to the pair of u's marginal gains in f and $\bar{f}$ w.r.t. the empty set. Thus we write $\pi : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$.

On the other hand, an adaptive priority algorithm may specify a new ordering function after each item is processed. Observe that in Q-Type 2 and 3, the length of Q increases with the iterations.

In both fixed and adaptive models, updating the internal memory also means deducing that certain marginal descriptions cannot exist and applying relevant queries (for the given Q-type) to obtain additional information. In particular, if $u_{next} \in \mathcal{N}$ is the item with the minimum π value, then any item u_j with $\pi(Q(u_j)) < \pi(Q(u_{next}))$ cannot appear later. We note that our inapproximability arguments result solely from information theoretic arguments. That is, the complexity or even computability of the ordering and decision steps is arbitrary, and there is no limitation on the size of the memory.

3 A 0.450 Inapproximation for Fixed Priority Algorithms

We design an LP whose solution gives the complete mapping of a submodular function $f : 2^{\mathcal{N}} \to \mathbb{R}$, over a small ground set $\mathcal{N}$. In the construction of f, we require certain items to be indistinguishable to the algorithm - allowing the adversary to control the input ordering. Denote by k the number of initial steps taken by the algorithm that the adversary will anticipate. That is, for 2^k possible partial solutions, the adversary prepares an input ordering (consistent with the algorithm's queries in these k steps) such that any extendible solution has a bad approximation ratio. Since f may not have a succinct representation, we exemplify the adversarial strategy on a small Max-Dicut instance, albeit with a worse lower bound.

Theorem 4. *For unweighted* Max-Dicut*, no fixed order Q-Type 3 double-sided myopic algorithm can achieve an approximation ratio greater than* $\frac{2}{3}$*.*

Proof. Let $v_1, ..., v_6$ be the vertices along a directed 6-cycle G with unit edge weights. Clearly, $OPT = 3$ is achieved by $\{v_1, v_3, v_5\}$ or $\{v_2, v_4, v_6\}$. The regularity of G ensures that $\pi(v_1) = ... = \pi(v_6)$ for any π, allowing the adversary to specify any input ordering. Suppose the algorithm accepts (*resp.* rejects) v_1 in the first step, the adversary fixes $u_{i_2} = \{v_3, v_4\}$ as the next set of possible inputs in the sequence. At this point, v_3 and v_4 are indistinguishable to the algorithm, since $\rho(v_3|S) = \rho(v_4|S) = 1$ and $\bar{\rho}(v_3|S) = \bar{\rho}(v_4|S) = 1$ for any $S \subseteq \mathcal{N}_1 = \{v_1\}$. If the algorithm accepts u_{i_2}, then the adversary sets $u_{i_2} = v_4$ (*resp.* $u_{i_2} = v_3$); otherwise it sets $u_{i_2} = v_3$ (*resp.* $u_{i_2} = v_4$). Both cases contradict the optimal solutions, and the maximum cut value is now at most 2. □

3.1 Construction of the LP for Theorem 2

Lemma 1. *No fixed Q-Type 3 double-sided myopic algorithm can achieve an approximation ratio greater than* $\frac{1}{c}$ *if there exists a (non-negative) submodular function* f *satisfying the following conditions:*

Cond. 1 $f(\{u\}) = f(\{v\}), \bar{f}(\{u\}) = \bar{f}(\{v\}), \forall u, v \in \mathcal{N}$.

Cond. 2 *There exist subsets* $A = \{a_1, ..., a_k\} \subseteq \mathcal{N}$, $R = \{r_1, ..., r_k\} \subseteq \mathcal{N}$, $A \cap R = \emptyset$*, such that for every* $0 < i < k$ *and every* $\mathcal{C}_i \in \{a_1, r_1\} \times \{a_2, r_2\} \times ... \times \{a_i, r_i\}$ *and every subset* $S \subseteq \mathcal{C}_i$*,*

$$f(S \cup \{a_{i+1}\}) = f(S \cup \{r_{i+1}\})$$
$$\bar{f}(S \cup \{a_{i+1}\}) = \bar{f}(S \cup \{r_{i+1}\})$$

Semantically, A (resp. R) *is the set of items that the algorithm is tricked into accepting (*resp. *rejecting). This is achievable if* a_j, r_j *are indistinguishable to the algorithm at round* j*, in the Q-Type 3 restricted oracle model.*

Cond. 3 *For every* $\mathcal{C}_k \in \{a_1, r_1\} \times \{a_2, r_2\} \times ... \times \{a_k, r_k\}$*, any solution* $S \subseteq \mathcal{N}$ *such that* $\mathcal{C}_k \cap A \subseteq S$ *and* $S \cap \mathcal{C}_k \cap R = \emptyset$ *(*i.e. S *is an extension of* $\mathcal{C}_k$*) must have* $f(S) \leq 1$.

Cond. 4 *There exists a set* $S^* \in 2^{\mathcal{N}}$ *such that* $f(S^*) \geq c$.

We formulate these conditions as an LP that maximizes c. For fixed n and k, define $\mathcal{N} = \{s_1, ..., s_{\lfloor \frac{n}{2} \rfloor}, o_1, ..., o_{\lceil \frac{n}{2} \rceil}\}$ as the ground set, and designate $O = \{o_1, ..., o_{\lceil \frac{n}{2} \rceil}\}$ as the optimal solution. Set $A = \{s_1, ..., s_k\}$ and $R = \{o_1, ..., o_k\}$ to deter the algorithm from O as much as possible. For every $S \subseteq \mathcal{N}$, we abuse notation and refer to $f(S)$ directly as the LP variable corresponding to the value of f on S. As this construction entails exponentially many variables, this is indeed only feasible in practice when restricted to a small $\mathcal{N}$. For interpretability, we may refer to the variable $\bar{f}(S_i)$ as alias for the variable $f(\mathcal{N} \setminus S_i)$ (following the definition of $\bar{f}$). Define the linear program as follows:

Objective

$$\max \quad f(\{o_1, ..., o_{\lceil \frac{n}{2} \rceil}\}) \tag{Obj}$$

Constraints

$$f(S \cup \{v\}) + f(S \cup \{u\}) - f(S \cup \{v, u\}) \geq f(S) \quad \forall S \subseteq \mathcal{N}, (v, u) \in \mathcal{N} \setminus S \tag{Sub}$$

$$\begin{aligned} f(\{v\}) - f(\{u\}) &= 0, \\ \bar{f}(\{v\}) - \bar{f}(\{u\}) &= 0 \qquad \forall v, u \in \mathcal{N} \qquad \text{(C1)} \\ f(S \cup \{a_{i+1}\}) - f(S \cup \{r_{i+1}\}) &= 0, \\ \bar{f}(S \cup \{a_{i+1}\}) - \bar{f}(S \cup \{r_{i+1}\}) &= 0 \qquad \forall S \subseteq \mathcal{C}_i, \\ &\qquad \forall \mathcal{C}_i \in \{a_1, r_1\} \times ... \times \{a_i, r_i\}, \\ &\qquad 0 < i < k \qquad \text{(C2)} \\ f(S) &\leq 1 \qquad \forall S \text{ s.t. } \exists \mathcal{C}_k, \mathcal{C}_k \cap A \subseteq S \\ &\qquad \wedge S \cap \mathcal{C}_k \cap R = \emptyset \qquad \text{(C3)} \\ f(S) &\geq 0 \qquad \forall S \qquad \text{(C4)} \\ f(\emptyset) &= 0 \qquad \text{(C5)} \end{aligned}$$

Inequality (Sub) is a necessary and sufficient condition for submodularity [20], and (C4) and (C5) constrain f to be non-negative and normalized. The remaining constraints correspond to conditions 1-3 of Lemma 1. If the LP is feasible, then the objective value in (Obj) is a lower bound for c in condition 4.

Proof of **Theorem 2**. Using $n = 8$ and $k = 4$, the LP described above produces a solution of $c = 2.2222$. By Lemma 1, this demonstrates a $\frac{1}{2.2222} \approx 0.450$ inapproximation. □

4 A 0.432 Inapproximation for Adaptive Priority Algorithms

Lemma 2. *No Q-Type 2 adaptive double-sided myopic algorithm can achieve an approximation ratio greater than* $\frac{1}{c}$ *if there exists a (non-negative) submodular function* f *that satisfies* **Cond. 3** *and* **4** *of Lemma 1, as well as the following:*

Cond. 2* *There exist subsets* $A = \{a_1, ..., a_k\} \subseteq \mathcal{N}$, $R = \{r_1, ..., r_k\} \subseteq \mathcal{N}$, $A \cap R = \emptyset$, *such that for every* $i < k$ *and every* $\mathcal{C}_i \in \{a_1, r_1\} \times \{a_2, r_2\} \times ... \times \{a_i, r_i\}$ *and all pairs* $v, u \in \mathcal{N} \setminus \mathcal{C}_i$,

$$f((\mathcal{C}_i \cap A) \cup \{u\}) = f((\mathcal{C}_i \cap A) \cup \{v\})$$
$$\bar{f}((\mathcal{C}_i \cap R) \cup \{u\}) = \bar{f}((\mathcal{C}_i \cap R) \cup \{v\})$$

Cond. 2* is the analog of **Cond. 2** of Lemma 1. Indistinguishability w.r.t. the weaker Q-Type 2 queries is now imposed on *all* remaining items in the first k steps (subsuming **Cond. 1**). As in Section 3.1, this leads to an LP representation.

Proof of **Theorem 3**. Using $n = 8$ and $k = 4$, the LP described above produces a solution of $c = 2.3158$, giving us an inapproximability of $\frac{1}{c} \approx 0.432$. □

5 Further Discussion of the Double-Sided Myopic Model

Adapting the priority framework [3], we define the class of double-sided myopic algorithms and show the Buchbinder *et al.* algorithm can be realized as an online double-sided myopic algorithm. Similar to Poloczek's [23] Max-Sat inapproximation, our inapproximation results for deterministic double-sided myopic algorithms provide evidence that the randomized $\frac{1}{2}$-approximation double greedy for USM cannot be de-randomized even if the algorithms allow reasonable input orderings beyond the online constraint. Our double-sided interpretation of the greedy algorithm of [4] satisfies the deterministic model of Bar-Noy and Lampis [2], for which they give a $\frac{2}{3\sqrt{3}}$ online inapproximability for Max-Dicut.

Allowing randomized decisions, our myopic model also captures the oblivious algorithms of Feige and Jozeph [9] for the Max-Dicut problem. Specifically, they define an oblivious algorithm as an online randomized algorithm that independently accepts each vertex v as a randomized function of the bias of the vertex, where $bias(v) = \frac{w_{in}(v)}{w_{in}(v)+w_{out}(v)}$. We note that the w_{in} (respectively, w_{out}) values, as defined by [9], are precisely the values of $\bar{\rho}(v|\emptyset)$ (resp. $\rho(v|\emptyset)$). Hence, the bias of every node can be determined within our Q-Type 2 model. In fact, we could have included the initial marginals $\bar{\rho}(v|\emptyset)$ and $\rho(v|\emptyset)$ within the Q-Type 1 model as all our inapproximations ensure that all initial marginals are identical. We could have also introduced a Q-Type 0 model allowing only access to the initial marginals which suffices to model oblivious algorithms as randomized online myopic algorithms. However, to capture the Feige and Jozeph inapproximation bound, we would have to further restrict our algorithms so that decisions are not memory dependent. We note that the Paul *et al.* [21] derandomization of the Feige and Jozeph oblivious algorithm results in an online non-oblivious algorithm that goes beyond our myopic framework as it is specific to a graph input model where each vertex is represented by its bias as well as the bias of its neighbors. The myopic framework does not have access to individual edges and hence cannot determine the adjacent nodes.

We introduced a hierarchy of query types (respectively priority orderings) to illustrate the restricted nature of the Buchbinder *et al.* algorithm (Q-Type

1 and online) in contrast to more general input and priority models which can reasonably be said to follow the same double sided greedy approach introduced by Buchbinder *et al.* We believe that Q-Type 3 is perhaps the most general value oracle model that one can allow in a myopic algorithm. The fact that Q-Type 3 allows exponential time and memory only strengthens the inapproximation results. From a positive point of view, an efficient myopic algorithm would only make a small (e.g. linear or polynomial) number of oracle calls but would have exponentially many from which to choose. Q-Type 2 provides a natural way to restrict algorithms to polynomially many query calls.

Our inapproximations follow from an LP formulation of possible algorithmic decisions, and at present does not yield a succinctly defined problem. However, we provide a $\frac{2}{3}$-inapproximation for Max-Dicut for fixed priority double-sided myopic algorithms. We again emphasize the generality of the myopic framework as it allows very general input orderings without imposing greediness in the decisions (as to rejecting or accepting an input). We also observe that *non-greediness* appears to be essential in both the randomized double-greedy algorithm of Buchbinder *et al.* as well as the deterministic approximation of Bar-Noy and Lampis for Max-Dicut on DAGs; in contrast, the Buchbinder *et al.* deterministic algorithm does make greedy decisions.

Acknowledgments

The authors thank Yuval Filmus for the idea of employing LP, and Matthias Poloczek and Charles Rackoff for their helpful suggestions.

References

1. N. Alon, B. Bollobás, A. Gyárfás, J. Lehel, and A. Scott, *Maximum directed cuts in acyclic digraphs*, Journal of Graph Theory, 55 (2007), pp. 1–13.
2. A. Bar-Noy and M. Lampis, *Online maximum directed cut*, Journal of Combinatorial Optimization, 24 (2012), pp. 52–64.
3. A. Borodin, M. N. Nielsen, and C. Rackoff, *(Incremental) priority algorithms*, in Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms, SODA '02, Philadelphia, PA, USA, 2002, Society for Industrial and Applied Mathematics, pp. 752–761.
4. N. Buchbinder, M. Feldman, J. S. Naor, and R. Schwartz, *A tight linear time (1/2)-approximation for unconstrained submodular maximization*, in Proceedings of the 2012 IEEE 53rd Annual Symposium on Foundations of Computer Science, FOCS '12, Washington, DC, USA, 2012, IEEE Computer Society, pp. 649–658.
5. G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, *Maximizing a submodular set function subject to a matroid constraint*, in Integer programming and combinatorial optimization, Springer, 2007, pp. 182–196.
6. G. Cornuejols, M. Fisher, and G. L. Nemhauser, *On the uncapacitated location problem*, in Studies in Integer Programming, B. K. P.L. Hammer, E.L. Johnson and G. Nemhauser, eds., vol. 1 of Annals of Discrete Mathematics, Elsevier, 1977, pp. 163 – 177.

7. G. Cornuejols, M. L. Fisher, and G. L. Nemhauser, *Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms*, Management Science, 23 (1977), pp. 789–810.
8. U. Feige, *A threshold of ln n for approximating set cover*, J. ACM, 45 (1998), pp. 634–652.
9. U. Feige and S. Jozeph, *Oblivious algorithms for the maximum directed cut problem*, arXiv preprint arXiv:1010.0406, (2010).
10. U. Feige, V. S. Mirrokni, and J. Vondrak, *Maximizing non-monotone submodular functions*, in Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science, FOCS '07, Washington, DC, USA, 2007, IEEE Computer Society, pp. 461–471.
11. M. Feldman, J. S. Naor, and R. Schwartz, *Nonmonotone submodular maximization via a structural continuous greedy algorithm*, in Automata, Languages and Programming, Springer, 2011, pp. 342–353.
12. Y. Filmus and J. Ward, *A tight combinatorial algorithm for submodular maximization subject to a matroid constraint*, in Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on, IEEE, 2012, pp. 659–668.
13. S. O. Gharan and J. Vondrák, *Submodular maximization by simulated annealing*, in Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, 2011, pp. 1098–1116.
14. M. X. Goemans and D. P. Williamson, *Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming*, J. ACM, 42 (1995), pp. 1115–1145.
15. E. Halperin and U. Zwick, *Combinatorial approximation algorithms for the maximum directed cut problem*, in Proc. of 12th SODA, 2001, pp. 200–1.
16. N. Huang and A. Borodin, *Bounds on double-sided myopic algorithms for unconstrained non-monotone submodular maximization*, arXiv preprint arXiv:1312.2173, (2013).
17. D. Kempe, J. M. Kleinberg, and É. Tardos, *Maximizing the spread of influence through a social network*, in KDD, 2003, pp. 137–146.
18. J. Lee, V. S. Mirrokni, V. Nagarajan, and M. Sviridenko, *Non-monotone submodular maximization under matroid and knapsack constraints*, in Proceedings of the 41st annual ACM symposium on Theory of computing, ACM, 2009, pp. 323–332.
19. G. L. Nemhauser and L. A. Wolsey, *Best algorithms for approximating the maximum of a submodular set function*, Mathematics of Operations Research, 3 (1978), pp. 177–188.
20. G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, *An analysis of approximations for maximizing submodular set functions-I*, Mathematical Programming, 14 (1978), pp. 265–294.
21. A. Paul, *Unconstrained submodular maximization priority algorithms.* Cornell University, Unpublished manuscript.
22. A. Paul, M. Poloczek, and D. Willamson, *Priority algorithms for MAX DICUT.* Cornell University, Unpublished manuscript, January 1, 2014.
23. M. Poloczek, *Bounds on greedy algorithms for MAX SAT*, in Algorithms–ESA 2011, Springer, 2011, pp. 37–48.
24. M. Sviridenko, *A note on maximizing a submodular set function subject to a knapsack constraint*, Operations Research Letters, 32 (2004), pp. 41–43.