From Query Complexity to Computational Complexity

Shahar Dobzinski*
Department of Computer Science
Cornell University, Ithaca, NY, USA
shahar@cs.cornell.edu

Jan Vondrák
IBM Almaden Research Center
San Jose, CA, USA
jvondrak@us.ibm.com

ABSTRACT

We consider submodular optimization problems, and provide a general way of translating oracle inapproximability results arising from the *symmetry gap* technique to computational complexity inapproximability results, where the submodular function is given explicitly (under the assumption that $NP \neq RP$). Applications of our technique include an optimal computational hardness of $(\frac{1}{2}+\epsilon)$-approximation for maximizing a symmetric nonnegative submodular function, an optimal hardness of $(1-(1-1/k)^k+\epsilon)$-approximation for welfare maximization in combinatorial auctions with k submodular bidders (for constant k), super-constant hardness for maximizing a nonnegative submodular function over matroid bases, and tighter bounds for maximizing a monotone submodular function subject to a cardinality constraint. Unlike the vast majority of computational inapproximability results, our approach does not use the PCP machinery or the Unique Games Conjecture, but relies instead on a direct reduction from Unique-SAT using list-decodable codes.

Categories and Subject Descriptors

F.2.2 [**Non-numerical Algorithms and Problems**]: Computations on Discrete Structures

Keywords

Inapproximability, submodular function, symmetry gap

1. INTRODUCTION

In this paper we consider the approximability of various submodular maximization problems. This class includes problems such as finding the maximum of a monotone submodular function subject to a cardinality constraint [10, 2], maximization of a nonnegative submodular function [3], and welfare maximization in combinatorial auctions where bidders have submodular valuations [8, 7, 12].

*Supported by NSF award AF-0910940.

STOCŠ12, May 19–22, 2012, New York, New York, USA.

Let f be a set function defined on a ground set N, $|N| = n$. f is called *submodular* if for every S and T we have that $f(S) + f(T) \geq f(S \cup T) + f(S \cap T)$. A set function f is called monotone (non-decreasing) if for every $T \subseteq S$ we have that $f(T) \leq f(S)$. Our goal is to prove hardness results for problems of the form $\max\{f(S) : S \in \mathcal{F}\}$, where $\mathcal{F}$ is the family of feasible solutions.

Notice that a naive representation of the submodular function would require us to specify 2^n values, one for each possible subset of the items. However, we would like our algorithms to run in time that is polynomial in n. Thus, there are two main approaches for accessing the function. The first one is to assume that the valuation is represented by an oracle that can answer a certain type of queries. The simplest oracle is a *value oracle*: given a set S, what is $f(S)$? To prove hardness results in the value oracle model, we have to show that if the algorithm achieves a certain approximation ratio, then it must make a superpolynomial number of value queries. Another model assumes that the function is succinctly represented, i.e., the representation size is polynomial in n and for each set S the value $f(S)$ can be computed in polynomial time. To prove hardness results in this model we have to show that no polynomial-time algorithm can guarantee a certain approximation ratio under standard complexity assumptions, such as $P \neq NP$.

By now, one can say that the value oracle model is quite well understood. Optimal or near-optimal hardness results are known for many of the problems that have been considered [3, 5, 9]. Moreover, in [14] a unified approach for obtaining inaproximability results was presented: it defines a notion of *symmetry gap* and shows how to systematically obtain value-oracle hardness results that match this gap. However, the situation is less bright as far as succinctly represented functions are concerned. For example, consider the problem of maximizing a nonnegative submodular function: an inapproximability factor of $\frac{1}{2} + \epsilon$ is known in the value oracle model [3] (even for symmetric submodular functions, which is optimal). However, for succinctly represented functions it is only known that a $(\frac{3}{4} + \epsilon)$-approximation is impossible unless $P = NP$ [3], and 0.695-approximation is impossible assuming the Unique Games Conjecture [1].

In this paper we show how to "mechanically convert" oracle hardness results for submodular optimization into computational hardness results. Specifically, we show that every hardness result obtained via the symmetry gap machinery can also be obtained for succinctly represented functions. In a sense, this shows that the difference between oracle hardness results and computational complexity hardness re-

sults is not too big here: one can concentrate on proving a hardness result in the value oracle model, and get the same hardness result when the function is explicitly represented "for free". Another way of interpreting our results, is that we face *both* computation and communication barriers when designing algorithms for submodular functions, and removing one barrier does not suffice.

Theorem: (informal) Let $\max\{f(S) : S \in \mathcal{F}\}$ be an instance of submodular maximization with a symmetry gap $\gamma \in (0,1)$. Then for any constant $\epsilon > 0$, there is no $(\gamma+\epsilon)$-approximation algorithm for a succinctly represented problem "related" to $\max\{f(S) : S \in \mathcal{F}\}$, unless $NP = RP$.

We remark that the main oracle hardness result in [13] contains a technical error which has been recently discovered and corrected in [14]. This error does not affect any currently known concrete applications of the theorem. We state our main hardness result (Theorem 4.4) in a form consistent with the new corrected theorem in [14].

Applications of the Main Theorem

Let us explicitly mention some of the applications of the main theorem (analogous to the value-oracle hardness results in [3, 9, 13, 5]).

Corollary: Let f denote a succinctly represented nonnegative submodular function. Unless $NP = RP$,

1. There is no $(\frac{1}{2}+\epsilon)$-approximation for the problem $\max f(S)$, even if f is symmetric.
2. There is no $(1-(1-1/k)^k+\epsilon)$-approximation for the problem $\max\{f(S) : |S| \leq n/k\}$, even if f is monotone and k is constant (n denotes the size of the ground set).
3. There is no $(1-(1-1/k)^k+\epsilon)$-approximation for welfare maximization in combinatorial auctions with k bidders that have (succinctly represented) monotone submodular valuations, even if k is constant.
4. There is no constant-factor approximation for the problem $\max\{f(S) : S \in \mathcal{B}\}$, where $\mathcal{B}$ is the collection of bases in a succinctly represented matroid.
5. There is no 0.491-approximation for the problem $\max\{f(S) : |S| \leq \ell\}$.
6. There is no 0.478-approximation for the problem $\max\{f(S) : |S| \in \mathcal{I}\}$, where $\mathcal{I}$ is the collection of independent sets in a succinctly represented matroid.

Related work. As noted above, for the first application (maximizing a succinctly represented nonnegative submodular function) it was previously known that there is no $(\frac{3}{4}+\epsilon)$-approximation unless $P = NP$ [3] and that there is no 0.695-approximation assuming the Unique Games Conjecture [1]. Our result holds even in the more restrictive setting where f is symmetric (for this case the papers [3, 1] provide inapproximability results with worse factors).

For the problem of monotone submodular maximization subject to a cardinality constraint, Feige [2] shows that that there is no $(1-\frac{1}{e}+\epsilon)$-approximation, unless $P = NP$ (while $1-1/e$ is the ratio guaranteed by the greedy algorithm [10]). While Feige's result hold even for coverage valuations, a subclass of submodular valuations, it requires the cardinality bound to be a very small (constant) fraction of n. Our hardness result holds for the constraint $|S| \leq n/k$, with any constant $k \geq 2$. The tightness of our result in this regime is implied by the recent work of Feldman, Naor and Schwartz [4].

The situation is similar when considering the problem of welfare maximization in combinatorial auctions. It was known that there is no $(1-\frac{1}{e})$-approximation unless $P = NP$, [7] (for coverage valuations), and that a $(1-\frac{1}{e})$ approximation can be achieved using value queries only [12]. The hardness result of [7] requires the number of players k to be a large constant, while our result holds for any constant $k \geq 2$. The tightness of our result is implied again by [4].

For the other problems mentioned above, no non-trivial hardness results in the computational complexity model were previously known.

Our Approach

In contrast to a large body of existing work on the hardness of approximation of explicitly posed problems, our proofs are not based on Probabilistically Checkable Proofs or other sophisticated machinery. Instead we present an alternative approach that is based on encoding a submodular function using a Unique-SAT formula and list decodable codes. List decodable codes are a generalization of error correcting codes; the difference is that, given an encoded (possibly corrupted) message, instead of decoding a single possible message, the list decoding algorithm outputs a list of possible messages, one of which is guaranteed to be correct. This allows list decoding algorithms to handle a greater number of errors than that allowed by unique decoding.

We sketch here at a high level the main ideas of our approach. For illustration, we consider the problem of nonnegative submodular maximization, for which it was proved in [3] that any $(\frac{1}{2}+\epsilon)$-approximation would require an exponential number of value queries[1]. We start by explaining the main ideas of this result.

Oracle hardness. Consider a ground set partitioned into two "hidden parts", $X = A \cup B$, and define two possible set functions on X. One is independent of A, B, and is defined as $f(S) = |S| \cdot |X \setminus S|$. Another one, f_A, is defined to coincide with f whenever S is "balanced" in the sense that $|S \cap A| - |S \cap B| \in [-\epsilon|X|, \epsilon|X|]$. For sets that are unbalanced, $f_A(S)$ depends on A in such a way that A (and also B) has "high value". Indeed, A and B are the intended optimal solutions for f_A. The gap between the optima for f and f_A is close to 2. The core of the proof is that the functions f and f_A cannot be distinguished by subexponentially many value queries, because every query with high probability falls in the region where f and f_A coincide. On such a query, the two functions not only give the same answer, but the answer does not give any information about (A, B).

How to hide a set. The main challenge in turning this into a computational hardness result is to present the input function f_A explicitly without revealing the partition (A, B). A natural description of f_A contains information about A in some form, and if an algorithm learns A, it knows the optimal solution.

[1]This proof can be seen as a germ of the symmetry gap technique.

The solution is to hide the set A using an *error-correcting code*: Let A_x be the codeword encoding a bit string x, using an error-correcting code $\mathcal{C}$ (we use a natural correspondence between sets and bit-strings here). This gives an exponential-size family of candidate sets $\{A_x : x \in \{0,1\}^m\}$. Only one of them is the true optimal solution A: this set $A = A_{x^*}$ is determined by a *distinguished string* x^*. However, the distinguished string x^* is not given explicitly - it is given for example by a uniquely satisfiable formula ϕ whose satisfying assignment is x^*. Thus, the formula ϕ implicitly describes an objective function $f_{A_{x^*}}$.

In order to interpret ϕ as a description of the function $f_\phi = f_{A_{x^*}}$, we have to make sure that we are able to evaluate $f_\phi(S)$, given ϕ. Here, we use the property that $f_A(S)$ only depends on A when S is unbalanced with respect to A, or in other words, when S is closer to A in Hamming distance than a random set would be. If $\mathcal{C}$ is a suitable *list-decodable code*, this is sufficient to determine A, by finding the list of potential codewords close to S, and checking if any of them corresponds to the satisfying assignment x^*. If we determine that S is indeed close to A_{x^*}, we are able to evaluate $f_\phi(S) = f_{A_{x^*}}(S)$. If S is not close to A_{x^*}, it means that the defining formula does not depend on A_{x^*} and we are again able to evaluate $f_\phi(S) = f(S)$. Thus, ϕ can be interpreted as a succinct description of f_ϕ.

Uniqueness and NP-hardness. To summarize: We have a problem whose input is assumed to be a Unique-SAT formula ϕ. This formula describes a submodular function $f_{A_{x^*}}$, where x^* is the satisfying assignment to ϕ (or f in case there is no satisfying assignment). The optima in the two cases differ by a factor close to 2. Therefore, approximating the optimum within a factor better than 2 would allow us to distinguish whether the formula is satisfiable or not.

The only remaining issue is the assumption that the formula has a unique assignment. This can be dealt with, following the work of Valiant and Vazirani [11]. They showed that every formula ϕ can be transformed into a random formula ϕ' of size polynomial in ϕ, such that if ϕ is satisfiable, then with constant probability ϕ' has a unique satisfying assignment. If ϕ is not satisfiable, then ϕ' is not satisfiable either. Therefore, we can feed the resulting formula ϕ' (repeatedly) into our presumed algorithm for submodular maximization, and determine by checking the returned solution whether there is a unique satisfying assignment. We allow our algorithm to perform arbitrarily when the input formula has multiple satisfying assignments, since such a formula does not encode a submodular function. Still, we are able to detect whenever a unique solution exists, and this will happen eventually with high probability. Therefore, we are able to solve SAT with one-sided error, and this implies $NP = RP$.

2. PRELIMINARIES

2.1 List Decodable Codes

DEFINITION 2.1 (LIST DECODABLE CODES). *A pair of functions (E, D), $E : \Sigma^m \to \Sigma^n$, $D : \Sigma^n \to (\Sigma^m)^\ell$, $\ell = poly(n)$ is called an (n, m, d)-*list decodable code *if:*

1. *E is injective.*
2. *For $c \in \Sigma^n$ and $x \in \Sigma^m$, $x \in D(c)$ if and only if $d_H(c, E(x)) \leq d$.*

Here, d_H denotes Hamming distance, $d_H(x, y) = |\{i : x_i \neq y_i\}|$. Of particular interest will be list decodable codes over $\Sigma = \{0, 1\}$, but we will also use larger alphabets. In this paper we are only interested in cases where E and D can be computed in polynomial time.

2.2 Unique-SAT and RP

DEFINITION 2.2. *Unique-SAT is a problem whose input is a formula ϕ and the goal is to:*

- *asnwer NO, if ϕ has no satisfying assignment,*
- *answer YES, if ϕ has exactly 1 satisfying assignment,*
- *return an arbitrary answer otherwise.*

DEFINITION 2.3. *RP is the class of decision problems that can be decided by randomized polynomial-time algorithms that always answer NO if the correct answer is NO, and answer YES with probability at least $1/2$ if the correct answer is YES.*

THEOREM 2.4 (VALIANT-VAZIRANI [11]). *Unique-SAT is NP hard under one-sided error randomized reductions.*

3. WARMUP: NONNEGATIVE SUBMODULAR MAXIMIZATION

In this section, we consider the problem of non-monotone submodular maximization. In this problem we are given a non-negative submodular function $f : 2^X \to \mathbb{R}_+$ and we are interested in finding a set S maximizing $f(S)$. The result we prove is implied by our more general result for symmetry gap hardness. We present it separately to convey the main ideas more clearly.

THEOREM 3.1. *There is a class $\mathcal{C}$ of succinctly represented nonnegative submodular functions, such that if there is a polynomial-time $(\frac{1}{2} + \epsilon)$-approximation algorithm for maximizing functions in $\mathcal{C}$ then $NP = RP$. The result holds even for symmetric functions.*

Indistinguishable submodular functions. We define submodular functions on a ground set $X = [2n]$, following [3]. Sometimes we will also identify X with $[n] \times \{0, 1\}$. We start by defining a function $f_C : 2^{[2n]} \to \mathbb{R}_+$ for every set $C \subset [2n]$ of size n. In the definition we use the notation $a = |S \cap C|$ and $b = |S \cap \overline{C}|$.

$$f_C(S) = \begin{cases} (a+b)(2n-a-b) & \text{if } |a-b| \leq 2\epsilon n, \\ 2a(n-b) + 2(n-a)b & \text{if } |a-b| > 2\epsilon n. \\ \quad +4\epsilon^2 n^2 - 8\epsilon n|a-b| & \end{cases}$$

We also define the function $f(S) = |S| \cdot (2n - |S|)$. Notice the both functions are symmetric. In [3] it is shown that all the above functions are submodular. Moreover, for any $C \in \binom{[2n]}{n}$, $\max_S f_C(S) = f_C(C) = 2n^2 + 4\epsilon^2 n^2 - 8\epsilon n^2$ and $\max_S f(S) = n^2$. Therefore, the ratio between the two maxima is $2 - O(\epsilon)$.

In [3], it is shown that these two functions cannot be distinguished by a polynomial number of value queries. However, we take a different approach here and design a way to present the above functions *explicitly* on the input, while guaranteeing that distinguishing between the two cases implies deciding Unique-SAT.

SAT-submodular functions. We encode the functions above as follows. First, we fix a suitable list decodable code: more precisely a family of binary $(n, m, (\frac{1}{2} - \epsilon)n)$-list decodable codes for all $n \in \mathbb{Z}_+$ (over $\Sigma = \{0,1\}$, with $n = poly(m)$ and constant $\epsilon > 0$; such codes are described for instance in [6]). We denote the encoding function by $E : \{0,1\}^m \to \{0,1\}^n$ and the decoding function by $D : \{0,1\}^n \to (\{0,1\}^m)^\ell$. This code is fixed in the following.

Each function in $\mathcal{C}$ is described by a boolean formula ϕ, which is assumed to have at most one satisfying assignment. (If ϕ has multiple satisfying assignments, it may not encode a submodular function and an algorithm run on such input can behave arbitrarily.) Given ϕ on m variables, we define $f_\phi : \{0,1\}^{2n} \to \mathbb{R}_+$ as follows. Let $x = (x_1, \ldots, x_m)$ be the unique satisfying assignment to the variables (if it exists). Let $y = E(x) \in \{0,1\}^n$. We identify the ground set $X = [2n]$ with $[n] \times \{0,1\}$ and we define an expansion procedure $exp : \{0,1\}^n \to 2^X$ as follows: $(i,0) \in exp(x)$ and $(i,1) \notin exp(x)$ if $y_i = 0$, and $(i,0) \notin exp(x)$ and $(i,1) \in exp(x)$ if $x_i = 1$. Note that $|exp(x)| = n$. We define the function corresponding to ϕ to be $f_\phi = f_C$ where $C = exp(x)$ and x is the unique satisfying assignment to ϕ (with some overloading of notation which the reader will hopefully forgive). If ϕ is not satisfiable then we interpret the function as $f_\phi = f$. As we will see, this is a natural choice, considering that the "high-value set" for f_ϕ is determined by a satisfying assignment for ϕ. If there is no satisfying assignment, there is also no high-value set.

We now show that ϕ is indeed a legitimate representation of f_ϕ in the sense that value queries can be answered efficiently (and thus one can use the algorithms of [3] to obtain good approximation ratios if the input is a SAT-submodular function given by ϕ). In particular, we do not need to know whether ϕ is satisfiable in order to evaluate $f_\phi(S)$.

LEMMA 3.2. *Given ϕ, the value of $f_\phi(S)$ can be calculated in polynomial time for any $S \subseteq [2n]$.*

PROOF. In this proof we view subsets of $X = [2n]$ as strings in $\{0,1\}^{2n}$; the Hamming distance d_H is then equivalent to the symmetric difference between sets. Observe that if S is *balanced*, meaning $||S \cap C| - |S \setminus C|| \leq 2\epsilon n$, we can calculate the value of $f_C(S)$ without having to know what C is (or even without knowing whether the function is f or f_C). Therefore, to calculate the value for given S, we have to show a polynomial-time procedure that finds out whether S is balanced or unbalanced, and finds C in the unbalanced case. In the following we show a procedure that identifies whether $|S \cap C| - |S \setminus C| > 2\epsilon n$ and in that case finds C. The other case is similar.

CLAIM 3.3. *If $|S \cap C| - |S \setminus C| > 2\epsilon n$ then $d_H(S, C) < n - 2\epsilon n$.*

PROOF. Using the assumption and $|C| = n$,

$$2\epsilon n < |S \cap C| - |S \setminus C| = n - |C \setminus S| - |S \setminus C| = n - d_H(S, C).$$

□

For $a \in \{0,1\}$, we define a contracting procedure $con^a : 2^X \to \{0,1\}^n$ that takes a set $C \subseteq X = [n] \times \{0,1\}$ and produces the following n-bit string $y = con^a(C)$: If $(i,0) \notin C$ and $(i,1) \in C$ then $x_i = 0$. If $(i,0) \in C$ and $(i,1) \notin C$ then $x_i = 1$. Otherwise, $x_i = a$. Note that $con^a(exp(x)) = x$ for both $a \in \{0,1\}$. We now need the following simple claim:

CLAIM 3.4. *If $d_H(S, C) < n - 2\epsilon n$ then there exists $a \in \{0,1\}$ such that $d_H(con^a(S), con^a(C)) < \frac{n}{2} - \epsilon n$.*

PROOF. Observe that two "01" bits in S that do not agree with two "10" bits in C (or vice versa) will result in one bit of disagreement between $con^a(S)$ and $con^a(C)$. If two "10" bits (or "01" bits) in C do not agree with "11" (or "00") bits in S, then we have one bit of disagreement between $con^a(S)$ and $con^a(C)$ for either $a = 0$ or $a = 1$. Therefore, a block with two disagreements between S and C translates to one bit of disagreement between $con^a(S)$ and $con^s(C)$, and a block with one disagreement translates to a bit with $\frac{1}{2}$ probability of disagreement (when choosing $a \in \{0,1\}$ uniformly at random). Thus the claim holds for some value of $a \in \{0,1\}$. □

We are now ready to complete the proof. Observe that if $d_H(con^a(S), con^a(C)) < \frac{n}{2} - \epsilon n$ then the list-decodable property of the code implies that the unique satisfying assignment x must be in one of the polynomially many strings that $D(con^a(S))$ returns. We can check if x is one of the strings of $D(con^a(S))$ simply by testing the satisfiability of each of the assignments that $D(con^a(S))$ returns, for each of the two possible values of a. If none of the assignments is satisfying, we know that S is balanced with respect to C and we do not need the knowledge of C to compute $f_C(S)$. If we find the set C, computing $f_C(S)$ is straightforward.

PROOF OF THEOREM 3.1. Assume that we have a $(\frac{1}{2} + 8\epsilon)$-approximation algorithm for maximizing a nonnegative submodular function. The specific class of submodular functions that we use is the class of SAT-submodular functions, where f_ϕ is encoded by giving the formula ϕ. Given a formula ϕ, we simply copy it on the input for the submodular maximization algorithm; i.e. the reduction is trivial.

As we showed, the function f_ϕ can be evaluated efficiently, given ϕ. Hence the presumed submodular maximization algorithm will be able to find the optimum within a factor of $\frac{1}{2} + 8\epsilon$. If the input formula is uniquely satisfiable, then the true optimum is at least $(2 - 8\epsilon)n^2$ and hence the algorithm returns a solution of value strictly above n^2. On the other hand, if the input formula is not satisfiable, there is no solution of value more than n^2. Therefore, we can distinguish the two cases and solve Unique-SAT. By Theorem 2.4, this implies $NP = RP$. □

4. INAPPROXIMABILITY FROM SYMMETRY GAP

In this section, we present a computational-complexity version of the general hardness result for submodular optimization based on the *symmetry gap* [14]. This encapsulates the hardness result for nonnegative submodular maximization as a special case (see [14] for more details). First, we summarize the symmetry gap technique and formulate our main result.

Symmetry gap. The starting point is a fixed instance $\max\{f(S) : S \in \mathcal{F}\}$, where $f : 2^X \to \mathbb{R}_+$ is a submodular function and $\mathcal{F} \subset 2^X$ is the set of feasible solutions. We assume that this instance is symmetric with respect to a certain group $\mathcal{G}$ of permutations of X, in the following sense.

DEFINITION 4.1. *We call an instance $\max\{f(S) : S \in \mathcal{F}\}$ on a ground set X totally symmetric with respect to a group*

of permutations $\mathcal{G}$ on X, if $f(S) = f(\sigma(S))$ for all $S \subseteq X$ and $\sigma \in \mathcal{G}$, and $S \in \mathcal{F} \Leftrightarrow S' \in \mathcal{F}$ whenever $\mathbf{E}_{\sigma\in\mathcal{G}}[\mathbf{1}_{\sigma(S)}] = \mathbf{E}_{\sigma\in\mathcal{G}}[\mathbf{1}_{\sigma(S')}]$.

Remark. The symmetry assumption on the family of feasible sets here is stronger than the one given in [13]. This is due to a mistake in [13] which the author only recently found and corrected in a revised manuscript [14]. This mistake affects the formulation of the general hardness theorem (see [14]), but fortunately not any of the concrete applications given in [13] and [5].

We note that $\sigma \in \mathcal{G}$ is a permutation of X, but we also use it for the naturally induced map on subsets of X and vectors indexed by X. For $\mathbf{x} \in [0,1]^X$, we define the *symmetrization operation* as $\bar{\mathbf{x}} = \mathbf{E}_{\sigma\in\mathcal{G}}[\sigma(\mathbf{x})]$. (Whenever we use this notation, σ is drawn uniformly from the group $\mathcal{G}$.) A fractional solution $\mathbf{x}$ is called symmetric, if $\bar{\mathbf{x}} = \mathbf{x}$.

We consider the *multilinear extension* $F(\mathbf{x}) = \mathbf{E}[f(\hat{\mathbf{x}})]$, where $\hat{\mathbf{x}} \in \{0,1\}^X$ is a random vector whose i-th coordinate is rounded to 1 independently with probability x_i and 0 with probability $1 - x_i$. We also define $P(\mathcal{F}) = \mathrm{conv}(\{\mathbf{1}_S : S \in \mathcal{F}\})$, the polytope associated with $\mathcal{F}$. Then, the symmetry gap is defined as follows.

DEFINITION 4.2 (SYMMETRY GAP). *Let $\max\{f(S) : S \in \mathcal{F}\}$ be an instance totally symmetric under a group of permutations $\mathcal{G}$ of the ground set X. Define $\bar{\mathbf{x}} = \mathbf{E}_{\sigma\in\mathcal{G}}[\sigma(\mathbf{x})]$. The symmetry gap of this instance is defined as $\gamma = \overline{OPT}/OPT$ where*

$$OPT = \max\{F(\mathbf{x}) : \mathbf{x} \in P(\mathcal{F})\},$$

$$\overline{OPT} = \max\{F(\bar{\mathbf{x}}) : \mathbf{x} \in P(\mathcal{F})\}$$

DEFINITION 4.3 (REFINEMENT). *Let $\mathcal{F} \subseteq 2^X$, $|X| = k$ and $|N| = n$. A refinement of $\mathcal{F}$ is $\tilde{\mathcal{F}} \subseteq 2^{N\times X}$, where*

$$\tilde{\mathcal{F}} = \{\tilde{S} \subseteq N \times X \;\big|\; (x_1, \ldots, x_k) \in P(\mathcal{F}) \text{ where } x_j = \frac{1}{n}|\tilde{S} \cap (N \times \{j\})|\}$$

The hardness result of [14] states that there is a class of instances of submodular maximization "similar to $\max\{f(S) : S \in \mathcal{F}\}$", namely in the form $\max\{\tilde{f}(S) : S \in \tilde{\mathcal{F}}\}$ where $\tilde{\mathcal{F}}$ is a refinement of $\mathcal{F}$, for which achieving an approximation better than γ requires an exponential number of value queries. Here, we turn this into the following computational hardness result.

THEOREM 4.4. *Let $\max\{f(S) : S \in \mathcal{F}\}$ be an instance of nonnegative (optionally monotone) submodular maximization, totally symmetric with respect to $\mathcal{G}$ with symmetry gap $\gamma = \overline{OPT}/OPT$, and $\epsilon > 0$. Then there is a collection $\mathcal{C}$ of succinctly represented instances in the form $\max\{\tilde{f}(S) : S \in \tilde{\mathcal{F}}\}$, where $\tilde{f}$ is nonnegative (optionally monotone) submodular and $\tilde{\mathcal{F}}$ is a refinement of $\mathcal{F}$, and there is no $(\gamma + \epsilon)$-approximation algorithm for $\mathcal{C}$ unless $NP = RP$.*

Applications. We mention a few applications of this theorem, that arise from the symmetric instances given in [13]. These are computational-complexity analogues of the results given in [13].

COROLLARY 4.5. *Unless $NP = RP$, there is no $(1-(1-1/k)^k + \epsilon)$-approximation for the problem $\max\{f(S) : |S| \leq n/k\}$, where f is a succinctly represented monotone submodular function (and k is constant).*

COROLLARY 4.6. *Unless $NP = RP$, there is no $(1-(1-1/k)^k + \epsilon)$-approximation for welfare maximization in combinatorial auctions with k bidders that have (succinctly represented) monotone submodular valuations (and k is constant).*

Corollary 4.5 follows immediately from Theorem 4.4 and the following symmetric instance on k elements: $\max\{f(S) : |S| \leq 1\}$ where $f(S) = \min\{|S|, 1\}$, as in [13]. Corollary 4.6 requires some additional justification; we discuss it in Section 5.

Another application due to a symmetric instance given in [13] is the following.

COROLLARY 4.7. *Unless $NP = RP$, there is no constant-factor approximation for the problem $\max\{f(S) : S \in \mathcal{B}\}$, where f is a succinctly represented nonnegative (non-monotone) submodular function, and $\mathcal{B}$ is the collection of bases in a succinctly represented matroid.*

Additional applications follow from the symmetric instances given in [5].

COROLLARY 4.8. *Unless $NP = RP$, there is no 0.491-approximation for the problem $\max\{f(S) : |S| \leq \ell\}$, where f is a succinctly represented nonnegative submodular function.*

COROLLARY 4.9. *Unless $NP = RP$, there is no 0.478-approximation for the problem $\max\{f(S) : |S| \in \mathcal{I}\}$, where f is a succinctly represented nonnegative submodular function and $\mathcal{I}$ is the collection of independent sets in a succinctly represented matroid.*

4.1 The proof

Here we proceed to prove Theorem 4.4. Up to a certain point, the approach is analogous to that of [13]: Based on the symmetric instance, we produce pairs of instances with a ratio of optimal values corresponding to the symmetry gap, that are provably hard to distinguish. The difference is in how we present these instances on the input. In contrast to the oracle model of [13], we encode the instances succinctly in a way that still makes them hard to distinguish. The main issue is how to present the objective function succinctly on the input, without revealing too much about its structure.

We appeal to the following technical lemma in [14] (the full version of [13]).

LEMMA 4.10. *Consider a function $f : 2^X \to \mathbb{R}_+$ invariant under a group of permutations $\mathcal{G}$ on the ground set X. Let $F(\mathbf{x}) = \mathbf{E}[f(\hat{\mathbf{x}})]$, $\bar{\mathbf{x}} = \mathbf{E}_{\sigma\in\mathcal{G}}[\sigma(\mathbf{x})]$, and fix any $\epsilon > 0$. Then there is $\delta > 0$ and functions $\hat{F}, \hat{G} : [0,1]^X \to \mathbb{R}_+$ (which are also symmetric with respect to $\mathcal{G}$), satisfying:*

1. *For all $\mathbf{x} \in [0,1]^X$, $\hat{G}(\mathbf{x}) = \hat{F}(\bar{\mathbf{x}})$.*
2. *For all $\mathbf{x} \in [0,1]^X$, $|\hat{F}(\mathbf{x}) - F(\mathbf{x})| \leq \epsilon$.*
3. *Whenever $||\mathbf{x} - \bar{\mathbf{x}}||^2 \leq \delta$, $\hat{F}(\mathbf{x}) = \hat{G}(\mathbf{x})$ and the value depends only on $\bar{\mathbf{x}}$.*

4. *The first partial derivatives of $\hat{F}, \hat{G}$ are absolutely continuous*[2].

5. *If f is monotone, then $\frac{\partial \hat{F}}{\partial x_i} \geq 0$ and $\frac{\partial \hat{G}}{\partial x_i} \geq 0$ everywhere.*

6. *If f is submodular, then $\frac{\partial^2 \hat{F}}{\partial x_i \partial x_j} \leq 0$ and $\frac{\partial^2 \hat{G}}{\partial x_i \partial x_j} \leq 0$ almost everywhere.*

Observe that we can choose $\epsilon > 0$ arbitrarily small (but constant). Then, the lemma provides another constant, $\delta > 0$ (which will be important in the following), and functions $\hat{F}, \hat{G}$ such that $\hat{F}$ is close to the multilinear extension F of the original objective function, while $\hat{G}$ is its symmetrized version $\hat{G}(\mathbf{x}) = \hat{F}(\bar{\mathbf{x}})$. Hence the gap between the optimal values of $\hat{F}$ and $\hat{G}$ is arbitrarily close to the symmetry gap (as in [13]; see [14] for more details). We remark that in addition, the functions $\hat{F}, \hat{G}$ have explicit forms (see [14]) that can be evaluated efficiently, given the function f. Recall that f is a fixed particular function and $|X|$ is a fixed ground set, and hence we can present f explicitly by its list of values. Although the formulas defining $\hat{F}$ and $\hat{G}$ in [14] involve a number of terms exponential in $|X|$, the number of terms is constant since $|X|$ is constant.

We also use the following lemma from [14], which allows us to use the continuous functions above to define discrete submodular functions.

LEMMA 4.11. *Let $F : [0,1]^X \to \mathbb{R}$ be a function with absolutely continuous first partial derivatives. Let $N = [n]$, $n \geq 1$, and define $f : N \times X \to \mathbb{R}$ so that $f(S) = F(\mathbf{x})$ where $x_i = \frac{1}{n}|S \cap (N \times \{i\})|$. Then*

1. *If $\frac{\partial F}{\partial x_i} \geq 0$ everywhere for each i, then f is monotone.*

2. *If $\frac{\partial^2 F}{\partial x_i \partial x_j} \leq 0$ almost everywhere for all i, j, then f is submodular.*

In this manner, we obtain pairs of discrete instances associated with $\hat{F}$ and $\hat{G}$, such that the ratio between their optima is close to the symmetry gap. The new contribution is how we encode these instances on the input. At a high level, we encode the submodular functions arising from Lemma 4.10 as follows. The functions are defined on a refined ground set $N \times X$, which can be viewed as partitioned into clusters $N \times \{i\}$ corresponding to individual elements $i \in X$ of the original symmetric instance. We hide this partition by associating exponentially many possible partitions with the codewords of a list decodable code. The "correct" partition is associated with a codeword x^* that encodes a unique satisfying assignment to a certain formula ϕ. (See Section 3 for an explanation of this idea in a special case where the partition has just 2 parts.) Intuitively, if an algorithm gives a good approximation to the respective maximization problem, it must be able to determine the underlying partition and hence by decoding the codeword find a satisfying assignment to the formula.

4.2 Element-transitive groups

First, we explain the case of *element-transitive* permutation groups: These are groups $\mathcal{G}$ of permutations on X such that for any two elements $i, j \in X$, there is $\sigma \in \mathcal{G}$ such that $\sigma(i) = j$. Alternatively, such groups can be described by saying that they induce a single *orbit* on X.

DEFINITION 4.12. *Given a group $\mathcal{G}$ of permutations on X, we define a relation $\sim$ where $i \sim j$ if there is $\sigma \in \mathcal{G}$ such that $\sigma(i) = j$.*

Since for each $\sigma, \sigma' \in \mathcal{G}$, the group also contains σ^{-1} and $\sigma \circ \sigma'$, it is quite obvious that $\sim$ is an equivalence relation.

DEFINITION 4.13. *An orbit of X is an equivalence class of the relation $\sim$. We say that $\mathcal{G}$ is element-transitive, if $i \sim j$ for all $i, j \in X$; i.e. X is a single orbit.*

In this case, all the elements are in a certain sense equivalent (although the group could still have a non-trivial structure). Furthermore, we have the following useful (known) property.

LEMMA 4.14. *For each orbit $O \subseteq X$, the symmetrized vector $\bar{\mathbf{x}} = \mathbf{E}_{\sigma \in \mathcal{G}}[\sigma(\mathbf{x})]$ satisfies $\bar{x}_j = \frac{1}{|O|} \sum_{i \in O} x_i$ for all $j \in O$.*

PROOF. Consider $i, j \in O$ and fix $\pi \in \mathcal{G}$ such that $\pi(i) = j$. Then we have

$$\bar{x}_j = \mathbf{E}_{\sigma \in \mathcal{G}}[x_{\sigma(j)}] = \mathbf{E}_{\sigma \in \mathcal{G}}[x_{\sigma \circ \pi(i)}]$$

Since the distribution of $\sigma \circ \pi$ is the same as that of σ (uniform over $\mathcal{G}$), we get that $\bar{x}_j = \bar{x}_i$. Furthermore, $|O|\bar{x}_j = \sum_{i \in O} \bar{x}_i = \sum_{i \in O} x_i$, because each permutation in $\mathcal{G}$ maps O onto itself. □

In particular, if $\mathcal{G}$ is element-transitive, the symmetrized vector $\bar{\mathbf{x}}$ has all coordinates equal. We remark that this is the case in the symmetric instances that lead to Corollary 4.5 and Corollary 4.6 (see [9, 13]).

Our goal is to encode succinctly the submodular functions arising from $\hat{F}$ and $\hat{G}$. In particular, we would like to have a hidden partition of the ground set $N \times X$ into sets $(Y_i : i \in X)$, and we want to define submodular functions $\hat{f}(S) = \hat{F}(\mathbf{x})$, $\hat{g}(S) = \hat{G}(\mathbf{x})$, where $x_i = \frac{|S \cap Y_i|}{|Y_i|}$. We want to be able to evaluate the functions $\hat{f}(S), \hat{g}(S)$ efficiently, and yet it is important that the partition $(Y_i : i \in X)$ should not be easy to determine.

Recall Lemma 4.10. If $||\mathbf{x} - \bar{\mathbf{x}}||^2 \leq \delta$, then $\hat{F}(\mathbf{x}) = \hat{G}(\mathbf{x})$. Moreover, $\hat{G}(\mathbf{x}) = \hat{F}(\bar{\mathbf{x}})$ and hence the value depends only on $\bar{\mathbf{x}}$. Since all the coordinates of $\bar{\mathbf{x}}$ are equal, the value depends only on $\sum x_i$ and we do not need to know anything about the hidden partition. Knowledge about the partition is needed only in the case where the set S is *unbalanced* in the sense that the respective vector $x_i = \frac{|S \cap Y_i|}{|Y_i|}$ has some coordinates that are significantly larger than others. Therefore, we can proceed as follows.

Hiding a partition. Let $N = [n]$ and $X = [k]$. We fix a $(n, m, (1 - \frac{1+\gamma}{k})n)$-list-decodable code over the alphabet X, where $\gamma = \sqrt{\delta/k}$ with δ from Lemma 4.10 (see [6]). With each codeword $y \in X^n$, we associate the following partition of $N \times X$: for each $i \in [k]$, $Y_i = \{(a, y_a + i \pmod k) : 1 \leq a \leq n\}$. We have $\bigcup_{i=1}^k Y_i = N \times X$, because each pair (a, j) appears in Y_i for exactly one $i \in [k]$. One of these partitions will be distinguished by some property of the codeword $y \in X^n$ that can be checked efficiently, but

[2] A function $F : [0,1]^X \to \mathbb{R}$ is absolutely continuous, if $\forall \epsilon > 0; \exists \delta > 0; \sum_{i=1}^t ||\mathbf{x}_i - \mathbf{y}_i|| < \delta \Rightarrow \sum_{i=1}^t |F(\mathbf{x}_i) - F(\mathbf{y}_i)| < \epsilon$.

the distinguished codeword cannot be found easily — we use Unique-SAT for this task. We consider a formula ϕ over m variables in X, which has either 0 or 1 satisfying assignment. In case the formula has a unique satisfying assignment, it defines a partition $(Y_1, \ldots, Y_k)$ as above, using the codeword $y = E(z)$, where z is the unique satisfying assignment to ϕ. If ϕ does not have any satisfying assignment, it is interpreted as encoding the symmetric case where the input instance does not depend on the partition $(Y_1, \ldots, Y_k)$.

Encoding the feasibility constraint. In line with the definition of refinement (Definition 4.3) and also with the concept of a hidden partition $(Y_1, \ldots, Y_k)$, let us now define the feasible sets in the refined instance:

$$\tilde{\mathcal{F}} = \{S \subseteq N \times X : \xi(S) \in P(\mathcal{F}) \text{ where } \xi_i(S) = \frac{1}{n}|S \cap Y_i|\}$$

Observe the following: since the membership condition $S \in \mathcal{F}$ for the original instance is assumed to depend only on the symmetrized vector $\overline{\mathbf{1}_S} = \mathbf{E}_{\sigma \in \mathcal{G}}[\mathbf{1}_{\sigma(S)}]$, membership in $\tilde{\mathcal{F}}$ depends only on the symmetrized vector $\overline{\xi(S)}$. In fact we are now considering the case where the symmetrized vector has all coordinates equal, and hence membership in $\tilde{\mathcal{F}}$ depends only on $\sum_j \xi_j(S) = \frac{1}{n}|S|$. Hence an algorithm does not need to know the codeword y and the partition $(Y_1, \ldots, Y_k)$ in order to check the feasibility of a set S. The feasibility constraint can be described explicitly on the input by a boolean table indexed by $|S|$.

Encoding the submodular function. We encode a submodular function f_ϕ by providing a value table of the initial objective function $f : 2^X \to \mathbb{R}_+$ (which is a constant-size object), and a formula ϕ over m variables in X, which has either 0 or 1 satisfying assignment. In case the formula has a unique satisfying assignment, it defines a partition $(Y_1, \ldots, Y_k)$ as above, using the codeword $y = E(z)$, where z is the unique satisfying assignment to ϕ, and E is the encoding function of the list-decodable code that we fixed beforehand. Such a formula ϕ is then interpreted as encoding the function $f_\phi(S) = \hat{F}(\xi(S))$ where $\xi_i(S) = \frac{1}{n}|S \cap Y_i|$. If ϕ does not have any satisfying assignment, it is interpreted as encoding the function $f_\phi(S) = \hat{G}(\xi(S)) = F(\bar{\xi}(S))$ (which depends only on $\sum_j \xi_j(S) = \frac{1}{n}|S|$, and hence the partition $(Y_1, \ldots, Y_k)$ is irrelevant).

Evaluating the submodular function. The main technical point is to prove that given ϕ, we are able to evaluate the function $f_\phi(S)$ efficiently. As in Section 3, the trick is that we don't necessarily need to determine the partition $(Y_1, \ldots, Y_k)$ in order to evaluate f_ϕ, even if we are in the satisfiable case where the partition matters. If S is balanced in the sense that the vector defined by $x_i = \xi_i(S) = \frac{1}{n}|S \cap Y_i|$ satisfies $||\mathbf{x} - \bar{\mathbf{x}}||^2 \leq \delta$, then $\hat{F}(\mathbf{x}) = \hat{G}(\mathbf{x}) = \hat{F}(\bar{\mathbf{x}})$ and we can evaluate $f_\phi(S) = \hat{F}(\mathbf{x})$ without knowing the partition (essentially we can assume that we are in the unsatisfiable case and evaluate $\hat{F}(\bar{\mathbf{x}})$).

However, we must be able to detect whether $||\mathbf{x} - \bar{\mathbf{x}}||^2 > \delta$ and if so, reconstruct the partition $(Y_1, \ldots, Y_k)$. Recall that we consider the case where $\bar{\mathbf{x}}$ has all coordinates equal to $\bar{x} = \frac{1}{n}\sum x_i$; hence we want to detect whether $||\mathbf{x} - \bar{\mathbf{x}}||^2 = \sum_i (x_i - \bar{x})^2 > \delta$. This implies that for some coordinate, $x_i - \bar{x} > \sqrt{\delta/k} = \gamma$. Recall that $x_i = \frac{1}{n}|S \cap Y_i|$, and $\bar{x} = \frac{1}{kn}|S|$. Therefore, we want to be able to detect that S intersects some part Y_i more than others. In fact, since the parts $(Y_1, \ldots, Y_k)$ are generated from Y_1 by cyclic rotation of the alphabet X, it is enough to be able to detect this event for the first part, Y_1.

The following lemma is crucial for accomplishing that. It states that if we have a set $S \subset [n] \times X$ that correlates with a codeword $y \in X^n$ more than a random set would, then we can convert S into a codeword $s \in X^n$ that is closer to y in Hamming distance than a random codeword would be.

LEMMA 4.15. *Let $y \in X^n$ be a hidden codeword, $|X| = k$. If $S \subseteq [n] \times X$ satisfies*

$$|\{i \in [n] : (i, y_i) \in S\}| \geq \frac{1}{k}|S| + \gamma n,$$

then we can transform S (without knowing y) into a list of $n|X|$ strings $s^{(\ell)} \in X^n$ such that for at least one of them,

$$d_H(s^{(\ell)}, y) \leq \left(1 - \frac{1+\gamma}{k}\right) n$$

PROOF. First we show how to find $s \in X^n$ probabilistically. Let $S_i = \{j \in X : (i,j) \in S\}$. For each $i \in [n]$, we choose s_i as a random element of X, where each element of S_i is chosen with probability $\frac{2}{k} - \frac{|S_i|}{k^2}$, and each element of $X \setminus S_i$ is chosen with probability $\frac{1}{k} - \frac{|S_i|}{k^2}$. (The reader can check that these probabilities add up to $|S_i|(\frac{2}{k} - \frac{|S_i|}{k^2}) + (k - |S_i|)(\frac{1}{k} - \frac{|S_i|}{k^2}) = 1$.)

What is the expected number of coordinates where s and y disagree? We call $i \in [n]$ good if S_i contains y_i, and bad otherwise. By assumption, the number of good coordinates is at least $\frac{1}{k}|S| + \gamma n$. On each good coordinate, we have $\Pr[s_i = y_i] = \frac{2}{k} - \frac{|S_i|}{k^2}$, while for each bad coordinate, $\Pr[s_i = y_i] = \frac{1}{k} - \frac{|S_i|}{k^2}$. So the expected Hamming distance between s and y is

$$\begin{aligned}
\mathbf{E}[d_H(s,y)] &= \sum_{i=1}^{n}(1 - \Pr[s_i = y_i]) \\
&= \sum_{\text{good } i}\left(1 - \frac{2}{k} + \frac{|S_i|}{k^2}\right) + \sum_{\text{bad } i}\left(1 - \frac{1}{k} + \frac{|S_i|}{k^2}\right) \\
&= \sum_{i=1}^{n}\left(1 - \frac{1}{k} + \frac{|S_i|}{k^2}\right) - \frac{1}{k}\#\text{ good coordinates} \\
&\leq \left(n - \frac{n}{k} + \frac{|S|}{k^2}\right) - \frac{1}{k}\left(\frac{1}{k}|S| + \gamma n\right) \\
&= \left(1 - \frac{1+\gamma}{k}\right) n.
\end{aligned}$$

We can derandomize this construction by noting that due to linearity of expectation, we can have arbitrary dependencies between different coordinates s_i. Hence we can implement the randomization by subdividing $[0,1]$ into $|X|$ intervals for each $i \in [n]$, drawing a single random number $\Theta \in [0,1]$ for all $i \in [n]$, and then listing the $n|X|$ possible events. □

Using this lemma, we can evaluate the function $f_\phi(S)$ as follows.

- Using Lemma 4.15 with $\gamma = \sqrt{\delta/k}$, convert S into a list of strings $s^{(\ell)} \in X^n$. Make k copies of each string by considering all cyclic rotations of the alphabet X, $i \to i + j \bmod k$.

- Using the list-decodable code, get a list of all strings $z \in X^m$ such that $d_H(s^{(\ell)}, E(z)) \leq \left(1 - \frac{1+\gamma}{k}\right) n$ for some string $s^{(\ell)}$ on the list above.
- For each $z \in X^m$ on this list, check whether the respective assignment satisfies the formula ϕ. If so, let $(Y_1, \ldots, Y_k)$ be the partition of $N \times X$ associated with $y = E(z)$, as above. Otherwise, use a default partition such as $Y_j = \{(i,j) : i \in N\}$.
- Compute $x_j = \frac{|S \cap Y_j|}{|Y_j|}$ and evaluate $f_\phi(S) = \hat{F}(\mathbf{x})$. (We refer to [14] for an explicit formula for $\hat{F}(\mathbf{x})$.)

CLAIM 4.16. *The above is a valid efficient procedure to evaluate the function f_ϕ.*

PROOF. All the steps are efficient, due to Lemma 4.15, properties of list decoding, and the construction of $\hat{F}$. We remark that although the definition of $\hat{F}$ in [14] involves multilinear polynomials with 2^k terms, the number of terms is still constant since k is a constant.

For correctness, we need to check that we find the correct partition $(Y_1, \ldots, Y_k)$ whenever it is necessary for evaluating the function. If the input formula ϕ is uniquely satisfiable, let z be the unique assignment, $y = E(z)$ and $(Y_1, \ldots, Y_k)$ the associated partition.

If the set S is such that the respective vector $x_i = \frac{1}{n}|S \cap Y_i|$ satisfies $||\mathbf{x} - \bar{\mathbf{x}}||^2 > \delta$, we showed above that this implies $x_i - \bar{x} > \sqrt{\delta/k} = \gamma$ for some coordinate i. In terms of the set S and partition $(Y_1, \ldots, Y_k)$, this means $\frac{1}{n}|S \cap Y_i| - \frac{1}{kn}|S| > \gamma$ for some i. By considering cyclic rotations of X, we can assume that $\frac{1}{n}|S \cap Y_1| - \frac{1}{kn}|S| > \gamma$; this is exactly the assumption of Lemma 4.15. Therefore, we will generate a list of strings $s^{(\ell)}$ such that for one of them, $d_H(s^{(\ell)}, y) \leq (1 - \frac{1+\gamma}{k})n$. By the properties of the list-decodable code, we will be able to decode y and a string z such that $y = E(z)$ will appear on the list of decoded messages. Then we will check the formula ϕ and discover that z is a satisfying assignment - in this case, we determine correctly the partition $(Y_1, \ldots, Y_k)$, and we evaluate the correct function $\hat{F}(\mathbf{x})$.

If the set S is such that $||\mathbf{x} - \bar{\mathbf{x}}||^2 \leq \delta$, then we may not discover a satisfying assignment. The same will happen if no satisfying assignment exists. In both cases, we will use the default partition $Y_j = \{(i,j) : i \in N\}$. However, we still compute the correct value in both cases because in the region where $||\mathbf{x} - \bar{\mathbf{x}}||^2 \leq \delta$, we have $\hat{F}(\mathbf{x}) = \hat{G}(\mathbf{x}) = \hat{F}(\bar{\mathbf{x}})$, and this function does not depend on the partition $(Y_1, \ldots, Y_k)$. It depends only on $\sum x_i = \frac{1}{n}|S|$, and hence we will again evaluate correctly. □

This completes the proof of Theorem 4.4 in the case where the group $\mathcal{G}$ is element-transitive: We encoded the possible input functions $\hat{f}(S) = \hat{F}(\mathbf{x})$ and $\hat{g}(S) = \hat{G}(\mathbf{x})$ in a way that reduces Unique-SAT to the two cases, and the ratio between the two optima can be made arbitrarily close to the symmetry gap. The general case is treated in the next section.

4.3 General Symmetry Groups

In this section we extend the proof of Theorem 4.4 to the case of arbitrary symmetry groups $\mathcal{G}$. Compared to Section 4.2, the added difficulty is the presence of multiple orbits in the ground set X (see Definition 4.13). In general, we have to deal with the phenomenon of multiple orbits, for the following reason. The refined instances are partitioned into clusters $(Y_i : i \in X)$, and the argument in Section 4.2 was that this partition should be "hidden" from the algorithm. This is possible to achieve, if all elements in the original instance are equivalent and hence the objective function treats them the same way. However, in general we cannot claim that the algorithm cannot learn anything about the partition $(Y_i : i \in X)$. Elements in different orbits can affect the objective function in different ways. We can only claim that the subpartition corresponding to each orbit remains hidden and cannot be determined by an algorithm. Therefore, we have to adjust the way we associate the partition with a codeword.

Encoding a partition with multiple orbits. Let $\mathcal{G}$ be a symmetry group that induces a partition of X into orbits $O_1, \ldots, O_r$. We consider a refinement of the ground set that is also partitioned into parts corresponding to the orbits: $\tilde{X} = \tilde{O}_1 \cup \ldots \cup \tilde{O}_r$. This partitioning will be made explicit and known to the algorithm. What we want to hide is the partitioning of each set $\tilde{O}_i$ into clusters corresponding to the elements of O_i. We could use a separate error-correcting code for each orbit; however, to connect this construction with the construction of the functions $\hat{F}, \hat{G}$, it is more convenient to use a single error-correcting code for all the orbits. We do this as follows.

Let κ be the smallest common multiple of the sizes of all orbits $|O_i|$, and let Σ be an alphabet of size κ. Note that κ is still a constant, depending only on the initial instance. For each orbit, the refinement of O_i will be defined on the set $N \times \Sigma$. Each "column" $\{i\} \times \Sigma$ will be partitioned into $|O_i|$ pieces of size $\kappa/|O_i|$, and the location of these pieces is determined by certain coordinates of an error-correcting code. The error-correcting code will map strings in Σ^m to codewords in Σ^{rn} (for all r orbits). The i-th orbit will use the respective portion of n symbols of the codeword. The refinement of the full ground set X will be $\tilde{X} = [r] \times N \times \Sigma$, where r is the number of orbits. The product structure given by $[r] \times N \times \Sigma$ will be explicitly known to an algorithm.

Given a distinguished codeword $y \in \Sigma^{rn}$, we define a partition of $\tilde{X} = [r] \times N \times \Sigma$, indexed by X. For convenience, we denote elements of X by (i,j), where (i,j) is the j-th element of orbit O_i. Then, the partition consists of parts $Y_{i,j}$ where $1 \leq i \leq r$ and $1 \leq j \leq |O_i|$. We define

$$Y_{i,j} = \{(i, a, y_a + j\kappa/|O_i| + \ell \pmod{\kappa}) : a \in N, 0 \leq \ell < \kappa/|O_i|\}$$

In other words, $Y_{i,j}$ is a subset of $\{i\} \times N \times \Sigma$, which for each $a \in N$ contains a block of $\kappa/|O_i|$ elements, whose location is determined by the codeword y. Together, these blocks for elements in O_i form the set $\{i\} \times N \times \Sigma$. Taking a union over all orbits O_i, we obtain the ground set $\tilde{X} = [r] \times N \times \Sigma$.

Encoding the feasibility constraint. By a natural extension of the previous section, the feasible sets in the refined instance are the following:

$$\tilde{\mathcal{F}} = \{S \subseteq N \times X : \xi(S) \in P(\mathcal{F}) \text{ where } \xi_{i,j}(S) = \frac{1}{n}|S \cap Y_{i,j}|\}$$

Observe that $\xi(S)$ is now a vector indexed by i, j, where O_i is an orbit and j is an element in the orbit. We know that membership in $\mathcal{F}$ depends only on the symmetrized vector $\overline{\mathbf{1}_S}$, and similarly membership in $P(\mathcal{F})$ depends on the symmetrized vector $\bar{\mathbf{x}}$. Therefore, $S \in \tilde{\mathcal{F}} \Leftrightarrow \xi(S) \in P(\mathcal{F}) \Leftrightarrow \overline{\xi(S)} \in P(\mathcal{F})$. The effect of symmetrization on

$\xi(S)$ is that the coordinates on each orbit are made equal (see Lemma 4.14). Therefore, the condition $S \in \tilde{\mathcal{F}}$ depends only on the parameters $|S \cap (\{i\} \times N \times \Sigma)|$ which can be computed by the algorithm (since the product structure of $[r] \times N \times \Sigma$ is explicit and known). We only need a description of the polytope $P(\mathcal{F})$, which can be described succinctly by $\mathcal{F} \subseteq 2^X$, a constant-size object.

Encoding the objective function. Finally, we show how we encode the objective function in the general case. Again, we provide a value table of the original objective function $f : 2^X \to \mathbb{R}_+$, and a formula ϕ over m variables in Σ, which has either 0 or 1 satisfying assignment. In case the formula has a unique satisfying assignment, it defines a partition $(Y_{i,j} : (i,j) \in X)$ as above, using the codeword $y = E(z)$, where z is the unique satisfying assignment to ϕ. Such a formula ϕ is then interpreted as encoding the function $f_\phi(S) = \hat{F}(\xi(S))$ where $\xi_{i,j}(S) = \frac{1}{n}|S \cap Y_{i,j}|$. If ϕ does not have any satisfying assignment, it is interpreted as encoding the function $f_\phi(S) = \hat{G}(\xi(S)) = F(\xi(S))$ (which depends only on $\sum \xi_{i,j}(S) = \frac{1}{n}|S|$, and hence the partition $(Y_{i,j} : (i,j) \in X)$ is irrelevant).

Evaluating the submodular function. Again, the main point is to show how to evaluate $f_\phi(S)$, given the formula ϕ. Now we have a vector (indexed by pairs (i,j)) defined by $x_{i,j} = \xi_{i,j}(S) = \frac{|S \cap Y_{i,j}|}{|Y_{i,j}|}$. If $\mathbf{x}$ satisfies $||\mathbf{x} - \bar{\mathbf{x}}||^2 \leq \delta$, then S is balanced, $\hat{F}(\mathbf{x}) = \hat{G}(\mathbf{x}) = \hat{F}(\bar{\mathbf{x}})$ and we can evaluate $f_\phi(S) = \hat{F}(\mathbf{x})$ without knowing the partition (again we can consider a default partition $Y_{i,j} = \{(i, a, j\kappa/|O_i| + \ell) : 0 \leq \ell < \kappa/|O_i|\}$, since the answer does not depend on it).

We must be able to reconstruct the partition $(Y_{i,j} : (i,j) \in X)$ whenever $||\mathbf{x} - \bar{\mathbf{x}}||^2 > \delta$. This means that $x_{i,j} - \bar{x}_{i,j} > \sqrt{\delta/k} = \gamma$ for some (i,j). Here, $\bar{\mathbf{x}}$ has coordinates equal on each orbit O_i, i.e. $\bar{x}_{i,j}$ depends only on i. This means that S intersects $Y_{i,j}$ more than the average $Y_{i,j'}$ over $j' \in O_i$:

$$|S \cap Y_{i,j}| > \frac{1}{|O_i|}(|S \cap \bigcup_{j' \in O_i} Y_{i,j'}| + \gamma\kappa n)$$

As $Y_{i,j}$ is a subset of $\{i\} \times N \times \Sigma$, we can restrict S to the same set and define $S' = S \cap (\{i\} \times N \times \Sigma) = S \cap \bigcup_{j' \in O_i} Y_{i,j'}$. It still holds that $|S' \cap Y_{i,j}| > \frac{1}{|O_i|}(|S'| + \gamma\kappa n)$. Now, $Y_{i,j}$ consists of a block of $\kappa/|O_i|$ cyclically shifted copies of a particular codeword $y \in \Sigma^{rn}$; by an averaging argument S' also intersects one of these copies, let's call it y', more than the average: $|S' \cap y'| \geq \frac{|O_i|}{\kappa}|S' \cap Y_{i,j}| > \frac{1}{\kappa}|S'| + \gamma n$. By Lemma 4.15, we can generate a list of codewords $s^{(\ell)}$ such that for at least one of them, $d_H(s^{(\ell)}, y') < (r - \frac{1+\gamma}{\kappa})n$. By list decoding, we are able to find such a codeword and determine the distinguished partition $(Y_{(i,j)} : (i,j) \in X)$. Then we are able to evaluate the function f_ϕ.

The rest of the proof is identical to the case of element-transitive groups. If we are able to approximate the optimum better than the symmetry gap, we must be able to distinguish the instances $\max\{\hat{f}(S) : S \in \tilde{\mathcal{F}}\}$ and $\max\{\hat{g}(S) : S \in \tilde{\mathcal{F}}\}$, and hence solve Unique-SAT.

5. WELFARE MAXIMIZATION FOR k AGENTS

As an application of Theorem 4.4, we stated Corollary 4.6 on the hardness of welfare maximization for k agents. This follows fairly easily from the discussion in Section 4.2, although some additional considerations are necessary due to the fact that the welfare maximization problem is formally in a different format than $\max\{f(S) : S \in \mathcal{F}\}$. In this section, we present a sketch of the proof of Corollary 4.6 — both to explain the additional arguments, and also to show a concrete application of the somewhat abstract machinery of symmetry gap.

In fact, we show Corollary 4.6 in the special case where all agents have the same valuation function, and hence the problem can be reformulated as the following:

$$\max\left\{\sum_{i=1}^{k} f(S_i) : S_i \subseteq X \text{ are disjoint}\right\}$$

We argue about the hardness of the problem as follows. We start from a symmetric instance where $f(S) = \min\{|S|, 1\}$, on a ground set of size k. Note that this is symmetric under every permutation of the ground set, and hence the group is element-transitive as in Section 4.2. The multilinear extension of this function is $F(\mathbf{x}) = 1 - \prod_{i=1}^{k}(1 - x_i)$. The symmetrized version is $G(\mathbf{x}) = 1 - (1 - \frac{1}{k}\sum_{i=1}^{k} x_i)^k$. Lemma 4.10 gives two slightly modified functions $\hat{F}(\mathbf{x}), \hat{G}(\mathbf{x})$, very close to the above, such that $\hat{F}(\mathbf{x}) = \hat{G}(\mathbf{x})$ whenever $||\bar{\mathbf{x}} - \mathbf{x}||^2 \leq \delta$. By discretizing these functions on a ground set partitioned into $(Y_1, Y_2, \ldots, Y_k)$, $|Y_1| = \ldots = |Y_k| = n$, we obtain monotone submodular functions $\hat{f}(S) = \hat{F}(\frac{1}{n}|S \cap Y_1|, \ldots, \frac{1}{n}|S \cap Y_k|)$ and $\hat{g}(S) = \hat{G}(\frac{1}{n}|S \cap Y_1|, \ldots, \frac{1}{n}|S \cap Y_k|)$.

By the discussion in Section 4.2, these functions can be encoded on the input (while hiding the partition $(Y_1, \ldots, Y_k)$) in such a way that we cannot distinguish the two cases, unless $NP = RP$. Now assume for a moment that the input functions are derived from $F(\mathbf{x})$ or $G(\mathbf{x})$ rather than $\hat{F}(\mathbf{x}), \hat{G}(\mathbf{x})$. When the input formula is $f(S) = F(\frac{1}{n}|S \cap Y_1|, \ldots, \frac{1}{n}|S \cap Y_k|) = 1 - \prod_{i=1}^{k}(1 - \frac{1}{n}|S \cap Y_i|)$, then one can achieve welfare k, by allocating Y_i to the agent i. If the input function is $g(S) = G(\frac{1}{n}|S \cap Y_1|, \ldots, \frac{1}{n}|S \cap Y_k|) = 1 - (1 - \frac{1}{kn}|S|)^k$, then the optimal solution is any partition into sets of equal size, which gives welfare $k(1 - (1 - 1/k)^k)$. The actual valuations on the input are derived from $\hat{F}(\mathbf{x}), \hat{G}(\mathbf{x})$ and hence slightly different from the above, but the relative difference of the two optimal values can be made arbitrarily small. Therefore, achieving a $(1 - (1 - 1/k)^k + \epsilon)$-approximation would imply $NP = RP$.

6. REFERENCES

[1] Per Austrin. Improved inapproximability for submodular maximization. In *APPROX*, pages 12–24, 2010.

[2] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

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

[4] Moran Feldman, Seffi Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In *IEEE FOCS*, pages 570–579, 2011.

[5] Shayan Oveis Gharan and Jan Vondrák. Submodular maximization by simulated annealing. In *ACM-SIAM SODA*, pages 1098–1116, 2011.

[6] Venkatesan Guruswami and Atri Rudra. Soft decoding, dual BCH codes, and better list-decodable eps-biased codes. In *ECCC 15(036)*, 2008.

[7] 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.

[8] Benny Lehmann, Daniel Lehmann, and Noam Nisan. Combinatorial auctions with decreasing marginal utilities. *Games and Economic Behavior*, 55:2:270–296, 2006.

[9] Vahab Mirrokni, Michael Schapira, and Jan Vondrák. Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. In *ACM EC*, pages 70–77, 2008.

[10] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions - I. *Mathematical Programming*, 14:265–294, 1978.

[11] Leslie G. Valiant and Vijay V. Vazirani. NP is as easy as detecting unique solutions. *Theoretical Computer Science*, 47:1:85–93, 1986.

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

[13] Jan Vondrák. Symmetry and approximability of submodular maximization problems. In *IEEE FOCS*, pages 651–670, 2009.

[14] Jan Vondrák. Symmetry and approximability of submodular maximization problems. 2011. Full version, arXiv:1110.4860v1.