Optimizing Ratio of Monotone Set Functions[*]

Chao Qian[1], Jing-Cheng Shi[2], Yang Yu[2], Ke Tang[1], Zhi-Hua Zhou[2]
[1]UBRI, School of Computer Science and Technology,
University of Science and Technology of China, Hefei 230027, China
[2]National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
{chaoqian, ketang}@ustc.edu.cn, {shijc, yuy, zhouzh}@lamda.nju.edu.cn

Abstract

This paper considers the problem of minimizing the ratio of two set functions, i.e., f/g. Previous work assumed monotone and submodular of the two functions, while we consider a more general situation where g is not necessarily submodular. We derive that the greedy approach GreedRatio, as a fixed time algorithm, achieves a $\frac{|X^*|}{(1+(|X^*|-1)(1-\kappa_f))\gamma(g)}$-approximation ratio, which also improves the previous bound for submodular g. If more time can be spent, we present the PORM algorithm, an anytime randomized iterative approach minimizing f and $-g$ simultaneously. We show that PORM using reasonable time has the same general approximation guarantee as GreedRatio, but can achieve better solutions in cases and applications.

1 Introduction

Minimizing the ratio of two set functions, i.e., f/g, can be found useful in many applications. For examples, in machine learning tasks, optimizing F-measure [Rijsbergen and Joost, 1974], linear discriminant analysis [McLachlan, 2004], normalized cut [Shi and Malik, 2000], etc., involve ratio optimization. Recently, Bai *et al.* [2016] studied the ratio minimization problem where the functions are monotone and submodular. We will denote this problem as RS minimization.

In [Bai *et al.*, 2016], several algorithms with bounded approximation guarantees were proposed for RS minimization. The GreedRatio algorithm iteratively selects one element that makes the ratio of the marginal gain by this element minimized. Other methods first connect RS minimization to the problem of minimizing the difference between submodular functions [Iyer and Bilmes, 2012] or connect it to the problem of submodular optimization with submodular constraints [Iyer and Bilmes, 2013], and then apply the existing techniques of the related problems. GreedRatio was also shown to obtain the best empirical performance in the application of F-measure maximization.

In this paper, we consider a more general situation, i.e., the function g can be non-submodular. We first prove that GreedRatio obtains a $\frac{|X^*|}{1+(|X^*|-1)(1-\hat{\kappa_f}(X^*))} \cdot \frac{1}{\gamma_{\emptyset,|X^*|}(g)}$-approximation guarantee (**Theorem 1**), where $|X^*|$ is the size of an optimal solution X^*, $\hat{\kappa_f}(X^*)$ is the curvature of f, and $\gamma_{\emptyset,|X^*|}(g)$ is the submodularity ratio of g. Note that when g is submodular, our derived bound becomes $\frac{|X^*|}{1+(|X^*|-1)(1-\hat{\kappa_f}(X^*))}$, which improves the previous known bound $\frac{1}{1-e^{\kappa_f-1}}$ [Bai *et al.*, 2016]. Particularly, the approximation bound is improved from ∞ to $|X^*|$ for $\kappa_f = 1$, and improved from $\frac{e}{e-1}$ to 1 for $\kappa_f = 0$.

Note that the greedy nature of GreedRatio results in an efficient fixed time algorithm, but meanwhile may limit its performance. We then propose a Pareto optimization [Qian *et al.*, 2015b; 2016] method, PORM, which is an anytime algorithm that can use more time to find better solutions. PORM first reformulates the original problem f/g as a bi-objective optimization problem that minimizes f and maximizes g simultaneously, then employs a randomized iterative algorithm to solve it, and finally selects the solution with the smallest ratio from the maintained set of solutions. Compared with single-bit forward search by GreedRatio, PORM can perform backward search, multi-path search and multi-bit search, which may help alleviate the issue of getting trapped in local optima. Our main theoretical results for PORM are that, within reasonable time,

- PORM can achieve the same general approximation guarantee as GreedRatio (**Theorem 2**).
- In a case of F-measure maximization, PORM can escape the local optimum by backward search, multi-path search and multi-bit search to find an optimum, while GreedRatio cannot (**Theorem 3**).

Experimental results on F-measure maximization exhibit the superior performance of PORM.

2 Minimizing Ratio of Monotone Functions

Given a finite set $V = \{v_1, v_2, \dots, v_n\}$, we study the functions $f : 2^V \to \mathbb{R}$ defined on subsets of V. A set function $f : 2^V \to \mathbb{R}$ is monotone if for any $X \subseteq Y$, $f(X) \leq f(Y)$. Without loss of generality, we assume that monotone functions are normalized, i.e., $f(\emptyset) = 0$. A set function $f : 2^V \to$

[*]This work was supported by the NSFC (61603367, 61333014), the JiangsuSF (BK20160066) and the Collaborative Innovation Center of Novel Software Technology and Industrialization.

$\mathbb{R}$ is submodular [Nemhauser *et al.*, 1978] if for any $X \subseteq Y$,

$$f(Y) - f(X) \leq \sum\nolimits_{v \in Y \setminus X} \big(f(X \cup \{v\}) - f(X)\big). \quad (1)$$

We then introduce two concepts, which will be used in our analysis. The submodularity ratio as presented in Definition 1 characterizes how close a set function f is to submodularity. It is easy to see from Eq. (1) that f is submodular iff $\gamma_{X,k}(f) = 1$ for any X and k. The curvature as presented in Definition 2 characterizes how close a monotone submodular set function f is to modularity. It is easy to verify that $\hat{\kappa_f}(X) \leq \kappa_f(X) \leq \kappa_f$. Note that f is modular iff $\kappa_f = 0$.

Definition 1 (Submodularity Ratio [Das and Kempe, 2011])**.** *Let f be a non-negative set function. The submodularity ratio of f with respect to a set X and a parameter $k \geq 1$ is*

$$\gamma_{X,k}(f) = \min_{L \subseteq X, S: |S| \leq k, S \cap L = \emptyset} \frac{\sum_{v \in S} \big(f(L \cup \{v\}) - f(L)\big)}{f(L \cup S) - f(L)}.$$

Definition 2 (Curvature [Conforti and Cornuéjols, 1984; Vondrák, 2010; Iyer *et al.*, 2013])**.** *Let f be a monotone submodular set function. The total curvature of f is*

$$\kappa_f = 1 - \min_{v \in V: f(v) > 0} \Big(f(V) - f(V \setminus \{v\})\Big) \Big/ f(v).$$

The curvature with respect to a set $X \subseteq V$ is

$$\kappa_f(X) = 1 - \min_{v \in X: f(v) > 0} \Big(f(X) - f(X \setminus \{v\})\Big) \Big/ f(v),$$

and an alternative notion is

$$\hat{\kappa_f}(X) = 1 - \Big(\sum_{v \in X} \big(f(X) - f(X \setminus \{v\})\big)\Big) \Big/ \Big(\sum_{v \in X} f(v)\Big).$$

We study the problem in Definition 3 that minimizes the ratio of a monotone submodular function f and a monotone function g. We can assume that $\forall v \in V$, $f(v) > 0$, because for any v with $f(v) = 0$, we can simply add it into the final subset, which will not increase the ratio. Note that we only consider minimization since maximizing f/g is equivalent to minimizing g/f.

Definition 3 (The General Problem)**.** *Given a monotone submodular function $f : 2^V \to \mathbb{R}^+$ and a monotone function $g : 2^V \to \mathbb{R}^+$, the task is as follows:*

$$\arg\min\nolimits_{\emptyset \subset X \subseteq V} \quad f(X)/g(X). \quad (2)$$

Maximizing the F-measure in information retrieval is a special instance of our studied problem with g being submodular, which will also be studied in this paper. Given a bipartite graph $G(V, W, E)$, where V is a set of objects, W is a set of words and each edge $(v, w) \in E$ means that the object v contains the word w, we define the function $\Gamma : 2^V \to 2^W$ as for any $X \subseteq V$, $\Gamma(X) = \{w \in W \mid \exists v \in X, (v, w) \in E\}$, i.e., $\Gamma(X)$ is the set of words contained by the objects in X. Then, the information retrieval problem of finding a subset of objects that exactly cover a target set of words $O \subseteq W$ can be formulated as maximizing the F-measure of the coverage on O, as shown in Definition 4. It is easy to verify that both $|\Gamma(X) \cap O|$ and $|O| + |\Gamma(X)|$ are monotone and submodular, where $|\cdot|$ denotes the size of a set.

Algorithm 1 GreedRatio Algorithm

Input: monotone submodular functions $f, g : 2^V \to \mathbb{R}^+$
Output: a subset $X \subseteq V$
Process:

```
1: Let $X_0 = \emptyset$, $R = V$ and $i = 0$.
2: repeat
3:     $v \in \arg\min_{v \in R} \frac{f(X_i \cup \{v\}) - f(X_i)}{g(X_i \cup \{v\}) - g(X_i)}$.
4:     $X_{i+1} = X_i \cup \{v\}$.
5:     $R = \{v \mid g(X_{i+1} \cup \{v\}) - g(X_{i+1}) > 0\}$.
6:     $i = i + 1$.
7: until $R = \emptyset$
8: return $X_{i^*}$ with $i^* \in \arg\min_i f(X_i)/g(X_i)$
```

Definition 4 (F-measure Maximization)**.** *Given a bipartite graph $G(V, W, E)$ and a target subset $O \subseteq W$, the task is as follows:*

$$\arg\max\nolimits_{\emptyset \subset X \subseteq V} \quad \left(F(X) = \frac{2|\Gamma(X) \cap O|}{|O| + |\Gamma(X)|}\right).$$

3 The GreedRatio Method

Bai *et al.* [2016] have recently investigated the problem of minimizing the ratio of monotone submodular functions, i.e., the function g in Definition 3 is submodular. They proved that the GreedRatio algorithm can obtain a $\frac{1}{1 - e^{\kappa_f - 1}}$-approximation guarantee, and also conducted experiments to show that GreedRatio achieves the best performance on F-measure maximization. As shown in Algorithm 1, GreedRatio iteratively selects one element v such that the ratio of the marginal gain on f and g by adding v is minimized.

We theoretically analyze the performance of GreedRatio for our studied general problem, i.e., g is not necessarily submodular. We prove its general approximation bound in Theorem 1, where OPT denotes the optimal function value of Eq. (2). Let X^* be an optimal solution with the minimum size, i.e., $f(X^*)/g(X^*) = OPT$ and $|X^*| = \min\{|X| \mid f(X)/g(X) = OPT\}$. The proof idea is that the best single element v^* obtains the desired approximation bound (as shown in Lemma 2), and the subset output by GreedRatio (i.e., line 8 of Algorithm 1) obviously satisfies that $f(X_{i^*})/g(X_{i^*}) \leq f(v^*)/g(v^*)$.

Lemma 1. *[Iyer* et al.*, 2013] Given a monotone submodular function $f : 2^V \to \mathbb{R}^+$, it holds that, for any $X \subseteq V$,*

$$\sum\nolimits_{v \in X} f(v) \leq \frac{|X|}{1 + (|X| - 1)(1 - \hat{\kappa_f}(X))} f(X).$$

Lemma 2. *For minimizing the ratio f/g where f is monotone submodular and g is monotone, there exists $v^* \in V$ such that*

$$\frac{f(v^*)}{g(v^*)} \leq \frac{|X^*|}{1 + (|X^*| - 1)(1 - \hat{\kappa_f}(X^*))} \frac{1}{\gamma_{\emptyset, |X^*|}(g)} \cdot OPT.$$

Proof. Let $v^* \in \arg\min_{v \in V} f(v)/g(v)$. By the definition of submodularity ratio (i.e., Definition 1), we get

$$g(X^*) \leq \frac{\sum_{v \in X^*} g(v)}{\gamma_{\emptyset, |X^*|}(g)} \leq \frac{1}{\gamma_{\emptyset, |X^*|}(g)} \frac{g(v^*)}{f(v^*)} \sum_{v \in X^*} f(v)$$

$$\leq \frac{1}{\gamma_{\emptyset,|X^*|}(g)} \frac{g(v^*)}{f(v^*)} \cdot \frac{|X^*| \cdot f(X^*)}{1 + (|X^*|-1)(1-\hat{\kappa_f}(X^*))},$$

where the last inequality is derived by Lemma 1. Thus, the lemma holds. □

Theorem 1. *For minimizing the ratio f/g where f is monotone submodular and g is monotone, GreedRatio finds a subset $X \subseteq V$ with*

$$\frac{f(X)}{g(X)} \leq \frac{|X^*|}{1 + (|X^*|-1)(1-\hat{\kappa_f}(X^*))} \frac{1}{\gamma_{\emptyset,|X^*|}(g)} \cdot OPT.$$

For the special case of g being submodular, $\gamma_{X,k}(g) = 1$ for any X and k, and thus the obtained bound in Theorem 1 becomes $\frac{|X^*|}{1+(|X^*|-1)(1-\hat{\kappa_f}(X^*))}$, as in Corollary 1. This improves the previous bound $\frac{1}{1-e^{\kappa_f - 1}}$ [Bai *et al.*, 2016], since

$$\frac{|X^*|}{1+(|X^*|-1)(1-\hat{\kappa_f}(X^*))} \leq \frac{1}{1-\hat{\kappa_f}(X^*)} \leq \frac{1}{1-\kappa_f} \leq \frac{1}{1-e^{\kappa_f - 1}},$$

where the first inequality is by $\hat{\kappa_f}(X^*) \in [0,1]$, the second inequality is by $\hat{\kappa_f}(X^*) \leq \kappa_f(X^*) \leq \kappa_f$, and the third is by $\kappa_f = 1-(1-\kappa_f) \leq e^{\kappa_f - 1}$. Particularly, the previous known bound becomes vacuous when $\kappa_f = 1$, while our derived bound has an upper limit $|X^*|$. Note that when f is modular (i.e., $\kappa_f = 0$), our derived bound discloses that GreedRatio finds an optimal solution, while the known bound in [Bai *et al.*, 2016] shows that GreedRatio only achieves a $\frac{e}{e-1}$-approximation guarantee.

Corollary 1. *For minimizing the ratio f/g where both f and g are monotone and submodular, GreedRatio finds a subset $X \subseteq V$ with $\frac{f(X)}{g(X)} \leq \frac{|X^*|}{1+(|X^*|-1)(1-\hat{\kappa_f}(X^*))} \cdot OPT$.*

4 The PORM Method

The greedy nature of GreedRatio may limit its performance. To alleviate the issue of getting trapped in local optima, we propose a new approach PORM adopting the Pareto Optimization [Qian *et al.*, 2015a; 2015b] idea. Pareto optimization is a recently emerged framework that uses bi-objective optimization as an intermediate step to solve single-objective optimization problems. It was previously applied for constrained optimization [Qian *et al.*, 2015a; 2015b], where the degree of constraint violation is optimized with the original objective simultaneously. Differently, PORM is for unconstrained optimization, where two objectives are created by dividing the original objective.

PORM reformulates the original problem Eq. (2) as a bi-objective minimization problem

$$\arg\min_{\boldsymbol{x} \in \{0,1\}^n} \quad \big(f(\boldsymbol{x}), \ -g(\boldsymbol{x})\big).$$

That is, PORM minimizes f and maximizes g simultaneously. Note that we use a Boolean vector $\boldsymbol{x} \in \{0,1\}^n$ to represent a subset $X \subseteq V$, where the i-th bit x_i is a binary indicator of the membership of v_i in X. In this paper, we will not distinguish $\boldsymbol{x} \in \{0,1\}^n$ and its corresponding subset X.

In the bi-objective setting, both the two objective values have to be considered for comparing two solutions $\boldsymbol{x}$ and $\boldsymbol{x}'$. $\boldsymbol{x}$ *weakly dominates* $\boldsymbol{x}'$ (i.e., $\boldsymbol{x}$ is *better* than $\boldsymbol{x}'$, denoted as

Algorithm 2 PORM algorithm

Input: a monotone submodular function $f : \{0,1\}^n \to \mathbb{R}^+$ and a monotone function $g : \{0,1\}^n \to \mathbb{R}^+$
Parameter: the number T of iterations
Output: a solution $\boldsymbol{x} \in \{0,1\}^n$
Process:

```
1: Select $\boldsymbol{x}$ from $\{0,1\}^n$ uniformly at random.
2: Let $P = \{\boldsymbol{x}\}$ and $t = 0$.
3: while $t < T$ do
4:     Select $\boldsymbol{x}$ from $P$ uniformly at random.
5:     Generate $\boldsymbol{x}'$ by flipping each bit of $\boldsymbol{x}$ with prob. $1/n$.
6:     if $\nexists \boldsymbol{z} \in P$ such that $\boldsymbol{z} \prec \boldsymbol{x}'$ then
7:         $P = (P \setminus \{\boldsymbol{z} \in P \mid \boldsymbol{x}' \preceq \boldsymbol{z}\}) \cup \{\boldsymbol{x}'\}$.
8:         $Q = \{\boldsymbol{z} \in P \mid |\boldsymbol{z}| = |\boldsymbol{x}'|\}$.
9:         $\boldsymbol{z}_1 = \arg\min_{\boldsymbol{z} \in Q} f(\boldsymbol{z})$, $\boldsymbol{z}_2 = \arg\max_{\boldsymbol{z} \in Q} g(\boldsymbol{z})$,
           $\boldsymbol{z}_3 = \arg\min_{\boldsymbol{z} \in Q} f(\boldsymbol{z})/g(\boldsymbol{z})$.
10:        $P = (P \setminus Q) \cup \{\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3\}$.
11:    end if
12:    $t = t + 1$.
13: end while
14: return $\arg\min_{\boldsymbol{x} \in P} f(\boldsymbol{x})/g(\boldsymbol{x})$
```

$\boldsymbol{x} \preceq \boldsymbol{x}'$) if $f(\boldsymbol{x}) \leq f(\boldsymbol{x}') \wedge g(\boldsymbol{x}) \geq g(\boldsymbol{x}')$; $\boldsymbol{x}$ *dominates* $\boldsymbol{x}'$ (i.e., $\boldsymbol{x}$ is *strictly better*, denoted as $\boldsymbol{x} \prec \boldsymbol{x}'$) if $\boldsymbol{x} \preceq \boldsymbol{x}'$ and either $f(\boldsymbol{x}) < f(\boldsymbol{x}')$ or $g(\boldsymbol{x}) > g(\boldsymbol{x}')$. But if neither $\boldsymbol{x}$ is better than $\boldsymbol{x}'$ nor $\boldsymbol{x}'$ is better than $\boldsymbol{x}$, they are *incomparable*.

The procedure of PORM is described in Algorithm 2. It starts from a random solution (line 1) and then iteratively tries to improve the solutions in the archive P (lines 3-13). In each iteration, a new solution $\boldsymbol{x}'$ is generated by randomly flipping bits of an archived solution $\boldsymbol{x}$ selected from the current P (lines 4-5); if $\boldsymbol{x}'$ is not dominated by any previously archived solution (line 6), it will be added into P and meanwhile those previously archived solutions weakly dominated by $\boldsymbol{x}'$ will be removed from P (line 7).

Note that although the domination-based comparison makes the archive P contain only incomparable solutions, the size of P can be large, which may reduce the efficiency of PORM. In order to control the size of P and meanwhile avoid the loss of useful information, we keep three informative solutions for each subset size $|\boldsymbol{x}| \in \{0, 1, \ldots, n\}$ (lines 8-10): $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are two boundary solutions with the smallest f value and the largest g value, respectively, and $\boldsymbol{z}_3$ is the solution with the smallest ratio. Note that $\boldsymbol{z}_3$ can be the same as $\boldsymbol{z}_1$ or $\boldsymbol{z}_2$. Thus, it is easy to see that $|P|$ is upper bounded by $1 + 3(n-1) + 1 = 3n - 1$.

PORM repeats for T iterations. The value of T affects the quality of the produced solution, which will be analyzed in the next section. After the iterations, the solution having the smallest ratio in P is selected (line 14).

Compared with GreedRatio, PORM can escape local optima by three different ways: (1) *backward search* that flips one bit value from 1 to 0; (2) *multi-path search* that maintains several (incomparable) solutions in the archive P; (3) *multi-bit search* that flips more than one 0-bits to 1-bits simultaneously. These advantages of PORM over GreedRatio will be theoretically shown in the next section.

5 Approximation Guarantee of PORM

We first prove the general approximation bound of PORM in Theorem 2, where $\mathbb{E}[T]$ denotes the expected number of iterations. We can see that PORM reaches the same approximation guarantee as GreedRatio. Note that $\mathbb{E}[T]$ depends on the f value of the initial solution generated in line 1 of Algorithm 2 (denoted as f_{init}) and the minimum f value $f_{\min} = \min\{f(v) \mid v \in V\}$.

Theorem 2. *For minimizing the ratio f/g where f is monotone submodular and g is monotone, PORM with $\mathbb{E}[T] \leq en(3n-1)(1+\frac{1}{1-\kappa_f}(1+\log\frac{f_{init}}{f_{min}}))$ finds a solution $\boldsymbol{x}$ with*

$$\frac{f(\boldsymbol{x})}{g(\boldsymbol{x})} \leq \frac{|X^*|}{1+(|X^*|-1)(1-\hat{\kappa_f}(X^*))}\frac{1}{\gamma_{\emptyset,|X^*|}(g)} \cdot OPT.$$

The proof idea is to follow the behavior of GreedRatio. That is, the optimization process is divided into two phases: (1) starts from an initial random solution and finishes until finding the special solution $\{0\}^n$ (i.e., $\emptyset$); (2) starts after phase (1) and finishes until finding a solution with the desired approximation guarantee. The expected number of iterations of phase (1) as shown in Lemma 4 is derived by using Lemma 3, a recently proposed approach for analyzing the hitting time of a random process. Note that $\log$ here is the natural logarithm.

Lemma 3. *[Doerr* et al.*, 2012] Let $S \subseteq \mathbb{R}^+$ be a finite set of positive numbers with minimum s_{min}. Let $\{X_t\}_{t\in\mathbb{N}}$ be a sequence of random variables over $S \cup \{0\}$. Let τ be the random variable that denotes the first point in time $t \in \mathbb{N}$ for which $X_t = 0$. Suppose that there exists $\delta > 0$ such that $\mathbb{E}[X_t - X_{t+1} \mid X_t = s] \geq \delta s$ holds for all $s \in S$. Then for all $s_0 \in S$, we have $\mathbb{E}[\tau \mid X_0 = s_0] \leq \big(1+\log(s_0/s_{min})\big)/\delta$.*

Lemma 4. *For minimizing the ratio f/g where f is monotone submodular and g is monotone, the expected number of iterations until PORM finds the solution $\{0\}^n$ is at most $\frac{en(3n-1)}{1-\kappa_f}(1+\log\frac{f_{init}}{f_{min}})$.*

Proof. We use Lemma 3 to prove it. Let $X_t = \min\{f(\boldsymbol{x}) \mid \boldsymbol{x} \in P\}$ after t iterations of PORM. Note that $X_t = 0$ implies that the solution $\{0\}^n$ is found, since $f(\boldsymbol{x}) > 0$ for any nonempty subset $\boldsymbol{x}$. Thus, the variable τ in Lemma 3 is just the number of iterations required by PORM for finding $\{0\}^n$.

We investigate $\mathbb{E}[X_t - X_{t+1}|X_t]$. Let $\hat{\boldsymbol{x}}$ be the corresponding solution with $f(\hat{\boldsymbol{x}}) = X_t$. We first show that X_t cannot increase, i.e., $X_{t+1} \leq X_t$. If $\hat{\boldsymbol{x}}$ is not deleted, X_t obviously will not increase. Note that there are two possible cases for removing $\hat{\boldsymbol{x}}$ from P. If $\hat{\boldsymbol{x}}$ is deleted in line 7 of Algorithm 2, the newly included solution $\boldsymbol{x}'$ must weakly dominate $\hat{\boldsymbol{x}}$, implying $f(\boldsymbol{x}') \leq f(\hat{\boldsymbol{x}})$. If $\hat{\boldsymbol{x}}$ is deleted in line 10, $|\hat{\boldsymbol{x}}| = |\boldsymbol{x}'|$ and $f(\boldsymbol{x}')$ must be smaller than $f(\hat{\boldsymbol{x}})$, since the solution in Q with the smallest f value is kept. Thus, X_t will not increase.

We then show that X_t can decrease by flipping only one 1-bit of $\hat{\boldsymbol{x}}$. Let $P_{\max}$ denote the largest size of P during the optimization process. In the $(t+1)$-th iteration, we consider that $\hat{\boldsymbol{x}}$ is selected in line 4 of Algorithm 2, which happens with probability at least $\frac{1}{P_{\max}}$ due to uniform selection; and in line 5, only the i-th bit of $\hat{\boldsymbol{x}}$ (i.e., $\hat{x}_i$) is flipped, which happens with probability $\frac{1}{n}(1-\frac{1}{n})^{n-1} \geq \frac{1}{en}$. If $\hat{x}_i = 1$, the newly generated solution $\boldsymbol{x}' = \hat{\boldsymbol{x}} \setminus \{v_i\}$. By the monotonicity of f, we have $f(\boldsymbol{x}') = f(\hat{\boldsymbol{x}} \setminus \{v_i\}) \leq f(\hat{\boldsymbol{x}})$. If the inequality strictly holds, $\boldsymbol{x}'$ now has the smallest f value and will be added into P, which leads to $X_{t+1} = f(\hat{\boldsymbol{x}} \setminus \{v_i\}) < X_t$. If $f(\hat{\boldsymbol{x}} \setminus \{v_i\}) = f(\hat{\boldsymbol{x}})$, obviously $X_{t+1} = X_t$; but we can still write $X_{t+1} = f(\hat{\boldsymbol{x}} \setminus \{v_i\})$.

Thus, we have shown that X_t does not increase, and can decrease by flipping only one 1-bit of $\hat{\boldsymbol{x}}$. We then get

$$\mathbb{E}[X_{t+1}|X_t] \leq \sum_{i:\hat{x}_i=1}\frac{f(\hat{\boldsymbol{x}}\setminus\{v_i\})}{enP_{\max}} + \left(1-\frac{|\hat{\boldsymbol{x}}|}{enP_{\max}}\right)X_t,$$

which implies that

$$\mathbb{E}[X_t - X_{t+1} \mid X_t] = X_t - \mathbb{E}[X_{t+1} \mid X_t]$$
$$\geq \sum_{i:\hat{x}_i=1}\frac{X_t - f(\hat{\boldsymbol{x}}\setminus\{v_i\})}{enP_{\max}} = \frac{\sum_{v\in\hat{\boldsymbol{x}}}\big(f(\hat{\boldsymbol{x}})-f(\hat{\boldsymbol{x}}\setminus\{v\})\big)}{enP_{\max}}.$$

By the definition of curvature (i.e., Definition 2), we have

$$1-\hat{\kappa_f}(\hat{\boldsymbol{x}}) = \frac{\sum\limits_{v\in\hat{\boldsymbol{x}}}\big(f(\hat{\boldsymbol{x}})-f(\hat{\boldsymbol{x}}\setminus\{v\})\big)}{\sum_{v\in\hat{\boldsymbol{x}}}f(v)} \leq \frac{\sum\limits_{v\in\hat{\boldsymbol{x}}}\big(f(\hat{\boldsymbol{x}})-f(\hat{\boldsymbol{x}}\setminus\{v\})\big)}{f(\hat{\boldsymbol{x}})},$$

where the inequality is by Eq. (1), i.e., $f(\hat{\boldsymbol{x}}) = f(\hat{\boldsymbol{x}}) - f(\emptyset) \leq \sum_{v\in\hat{\boldsymbol{x}}}(f(v)-f(\emptyset)) = \sum_{v\in\hat{\boldsymbol{x}}}f(v)$. Thus,

$$\mathbb{E}[X_t - X_{t+1}|X_t] \geq \frac{1-\hat{\kappa_f}(\hat{\boldsymbol{x}})}{enP_{\max}}f(\hat{\boldsymbol{x}}) \geq \frac{1-\kappa_f}{en(3n-1)}X_t,$$

where the last inequality is by $P_{\max} \leq 3n-1, X_t = f(\hat{\boldsymbol{x}})$, and $\hat{\kappa_f}(\hat{\boldsymbol{x}}) \leq \kappa_f(\hat{\boldsymbol{x}}) \leq \kappa_f$. That is, the condition of Lemma 3 holds with $\delta = \frac{1-\kappa_f}{en(3n-1)}$. Note that $X_0 = f_{\text{init}}$ and $s_{\min} = f_{\min}$. By Lemma 3, we get

$$\mathbb{E}[\tau \mid X_0 = f_{\text{init}}] \leq \frac{en(3n-1)}{1-\kappa_f}\left(1+\log\frac{f_{\text{init}}}{f_{\min}}\right). \qquad \square$$

Proof of Theorem 2. We analyze phase (2) after finding $\{0\}^n$. Note that once $\{0\}^n$ is found, it will always be in the archive P, because it has the smallest f value and no other solutions can dominate it. By selecting $\{0\}^n$ in line 4 of Algorithm 2 and flipping only the bit corresponding to the best single element $v^* \in \arg\min_{v\in V} f(v)/g(v)$ in line 5, which happens with probability at least $\frac{1}{P_{\max}}\cdot\frac{1}{n}(1-\frac{1}{n})^{n-1} \geq \frac{1}{en(3n-1)}$, the new solution $\boldsymbol{x}' = \{v^*\}$ is generated. Then, $\boldsymbol{x}'$ is used to update P (i.e., lines 6-11). This will make P always contain a solution which either weakly dominates $\boldsymbol{x}'$ or is incomparable with $\boldsymbol{x}'$ but has a smaller ratio. That is, P will always contain a solution $\boldsymbol{x}$ with

$$\frac{f(\boldsymbol{x})}{g(\boldsymbol{x})} \leq \frac{f(v^*)}{g(v^*)} \leq \frac{|X^*|}{1+(|X^*|-1)(1-\hat{\kappa_f}(X^*))}\frac{OPT}{\gamma_{\emptyset,|X^*|}(g)},$$

where the last inequality is by Lemma 2. Thus, phase (2) needs at most $en(3n-1)$ expected number of iterations.

By combining the two phases, we get that the total expected number of iterations for finding a solution with a $\frac{|X^*|}{1+(|X^*|-1)(1-\hat{\kappa_f}(X^*))}\frac{1}{\gamma_{\emptyset,|X^*|}(g)}$-approximation ratio is

$$\mathbb{E}[T] \leq en(3n-1)\left(1+\frac{1}{1-\kappa_f}\left(1+\log\frac{f_{\text{init}}}{f_{\min}}\right)\right). \qquad \square$$

By using an illustrative example of F-measure maximization in information retrieval, we then prove in Theorem 3 that GreedRatio will get trapped in a local optimal solution, while PORM can avoid local optima by three different ways, and finally find a global optimal solution. As shown in Definition 5, this example has a unique global optimal solution $\{1\}^{n-1}0$ (i.e., $\{v_1,\dots,v_{n-1}\}$). The proof idea is mainly that GreedRatio will first select the object v_n due to the greedy nature and will be misled by it, while PORM can avoid v_n by backward search, multi-path search or multi-bit search, which will be shown in the proof, respectively.

Definition 5 (An Example of F-measure Maximization)**.** *Let $V=\{v_1,v_2\dots,v_n\}$. The function Γ and the target set O satisfy that*

(1) $\forall 1\le i,j\le n-1:\Gamma(v_i)\cap\Gamma(v_j)=\emptyset$,
$|\Gamma(v_i)|=|\Gamma(v_j)|=n^2$;

(2) $\forall 1\le i\le n-1:|O\cap\Gamma(v_i)|=n^2-1$,
$|O|=(n-1)(n^2-1)$;

(3) $\forall 1\le i\le n-1:\Gamma(v_n)\cap\Gamma(v_i)\subseteq O\cap\Gamma(v_i)$,
$|\Gamma(v_n)\cap\Gamma(v_i)|=n+2$, $|\Gamma(v_n)|=|\Gamma(v_n)\cap O|+1$.

Theorem 3. *For the F-measure maximization example as in Definition 5, PORM with $\mathbb{E}[T]\le en(3n-1)(2+2\log n)$ finds the optimal solution $\{1\}^{n-1}0$, while GreedRatio cannot.*

Proof. By the definition of $F(\boldsymbol{x})=(2|\Gamma(\boldsymbol{x})\cap O|)/(|O|+|\Gamma(\boldsymbol{x})|)$, we derive that, for any $\boldsymbol{x}$ with $|\boldsymbol{x}|=i\wedge x_n=0$, $F(\boldsymbol{x})=\frac{2(n^2-1)i}{n^3-n^2-n+1+n^2i}$, which increases with i and reaches the maximum $1-\frac{1}{2n^2-1}$ when $i=n-1$; for any $\boldsymbol{x}$ with $|\boldsymbol{x}|=i\wedge x_n=1$, $F(\boldsymbol{x})=\frac{4n+2+2(n^2-n-3)i}{n^3-n^2+n+2+(n^2-n-2)i}$, which increases with i and reaches the maximum $1-\frac{1}{2n^2-2n-1+2/n}$ when $i=n$. Thus, $\{1\}^{n-1}0$ is the unique optimal solution.

For GreedRatio, it first selects one object v with the best ratio of the marginal gain, i.e., the largest $(2|\Gamma(v)\cap O|)/(|O|+|\Gamma(v)|-|O|)$ value. For any $1\le i\le n-1$, we have

$$\frac{2|\Gamma(v_i)\cap O|}{|\Gamma(v_i)|}=\frac{2(n^2-1)}{n^2}<\frac{2(n^2+n-2)}{n^2+n-1}=\frac{2|\Gamma(v_n)\cap O|}{|\Gamma(v_n)|}.$$

Thus, GreedRatio will first select v_n. From line 8 of Algorithm 1, it is easy to see that the final output subset by GreedRatio will always contain v_n. Thus, GreedRatio cannot find the optimal solution $\{v_1,\dots,v_{n-1}\}$.

For the PORM algorithm, the problem of F-measure maximization is implemented as minimizing $f(\boldsymbol{x})=|O|+|\Gamma(\boldsymbol{x})|$ and maximizing $g(\boldsymbol{x})=|\Gamma(\boldsymbol{x})\cap O|$ simultaneously. We then prove that PORM can find the optimal solution $\{v_1,\dots,v_{n-1}\}$ by three different ways.

[Backward search] The idea is that PORM first efficiently finds all the objects V, and then simply deleting v_n from V can produce the optimal solution.

Let $G_t=\max\{g(\boldsymbol{x})\mid\boldsymbol{x}\in P\}$ after t iterations of Algorithm 2. We first use Lemma 3 to derive the number of iterations (denoted as T_1) until G_t reaches the maximum $|O|$. Let $X_t=|O|-G_t$. Then, the random variable τ in Lemma 3 is just T_1, because $X_t=0$ is equivalent to $G_t=|O|$. Since $\mathbb{E}[X_t-X_{t+1}|X_t]=\mathbb{E}[G_{t+1}-G_t|G_t]$, we only need to analyze the change of G_t. Let $\hat{\boldsymbol{x}}$ be the corresponding solution with $g(\hat{\boldsymbol{x}})=G_t$. As in the proof of Lemma 4, we can similarly show that G_t does not decrease, and can increase by flipping only one 0-bit of $\hat{\boldsymbol{x}}$. We then get

$$\mathbb{E}[G_{t+1}-G_t\mid G_t]\ge\sum\nolimits_{i:\hat{x}_i=0}\frac{g(\hat{\boldsymbol{x}}\cup\{v_i\})-g(\hat{\boldsymbol{x}})}{enP_{\max}}$$
$$=\frac{\sum_{v\in V\setminus\hat{\boldsymbol{x}}}\big(g(\hat{\boldsymbol{x}}\cup\{v\})-g(\hat{\boldsymbol{x}})\big)}{enP_{\max}}\ge\frac{g(V)-g(\hat{\boldsymbol{x}})}{enP_{\max}}\ge\frac{X_t}{en(3n-1)},$$

where the second inequality is by the submodularity of g, i.e., Eq. (1). That is, the condition of Lemma 3 holds with $\delta=\frac{1}{en(3n-1)}$. Note that $X_0=|O|-G_0\le|O|=(n-1)(n^2-1)$ and $s_{\min}=n^2-1-(n+2)$. By Lemma 3, we get

$$\mathbb{E}[T_1]=\mathbb{E}[\tau\mid X_0]\le en(3n-1)\,(1+2\log n)\,.$$

From Definition 5, we know that a solution $\boldsymbol{x}$ with $g(\boldsymbol{x})=|\Gamma(\boldsymbol{x})\cap O|=|O|$ must contain $v_1,\dots,v_{n-1}$. Then, there are two possible solutions: $\{1\}^{n-1}0$ and $\{1\}^n$. To derive an upper bound on the number of iterations for finding the optimal solution $\{1\}^{n-1}0$, we pessimistically assume that $\{1\}^{n-1}0$ is not found. For the case that P contains $\{1\}^n$, the optimal solution $\{1\}^{n-1}0$ can be generated by selecting $\{1\}^n$ in line 4 and flipping the last 1-bit in line 5, which happens with probability at least $\frac{1}{P_{\max}}\cdot\frac{1}{n}(1-\frac{1}{n})^{n-1}\ge\frac{1}{en(3n-1)}$. Denote the number of iterations in this phase as T_2. We then have

$$\mathbb{E}[T_2]\le en(3n-1).$$

Therefore, we get that the expected number of iterations for PORM finding the optimal solution $\{1\}^{n-1}0$ is

$$\mathbb{E}[T]\le\mathbb{E}[T_1]+\mathbb{E}[T_2]\le en(3n-1)(2+2\log n).$$

[Multi-path search] Let $\boldsymbol{x^i}$ (where $1\le i\le n-1$) denote any solution such that $|\boldsymbol{x^i}|=i\wedge x^i_n=0$, i.e., there are i 1s in the first $n-1$ bits and the last bit is 0. The idea is that PORM first efficiently finds the empty set $\{0\}^n$, and then follows the path $\boldsymbol{x^1}\to\boldsymbol{x^2}\to\cdots\to\boldsymbol{x^{n-1}}$ to produce the optimal solution. Note that although $\boldsymbol{x^1}$ is worse than the solution $\{0\}^{n-1}1$ (i.e., $\{v_n\}$) on the original objective f/g, they are incomparable in the bi-objective setting, and thus $\boldsymbol{x^1}$ will be kept in P, which allows PORM to follow a path different from that by GreedRatio to find the optimal solution.

Let T_1 denote the number of iterations for finding the solution $\{0\}^n$. Due to the fact that the $f(\boldsymbol{x})$ value increases with the number of 1-bits of $\boldsymbol{x}$, we can derive a better upper bound for $\mathbb{E}[T_1]$ than Lemma 4. Let j denote the minimum number of 1-bits of the solutions in P. First, j cannot increase, because the smallest f value of the solutions in P cannot increase. Second, j can decrease by 1 in each iteration with probability at least $\frac{1}{P_{\max}}\cdot\frac{j}{n}(1-\frac{1}{n})^{n-1}\ge\frac{j}{en(3n-1)}$, since it is sufficient to select a solution with j 1s in line 4 and flip only one of its j 1-bits in line 5. Thus, for reaching $j=0$,

$$\mathbb{E}[T_1]\le\sum\nolimits_{j=1}^{n}\frac{en(3n-1)}{j}\le en(3n-1)(1+\log n).$$

Starting from $\{0\}^n$, let T_2 denote the number of iterations for following the path $\boldsymbol{x^1}\to\boldsymbol{x^2}\to\cdots\to\boldsymbol{x^{n-1}}$ (i.e., the optimal solution). Note that for $1\le i\le n-1$, $\boldsymbol{x^i}$ cannot be weakly dominated by any other solution, and there are only two different objective vectors for $|\boldsymbol{x}|=i$. Thus, according to the updating procedure of PORM (lines 6-11 of Algorithm 2), we know that once $\boldsymbol{x}^{\boldsymbol{i}}$ is found, it will be always kept in P. The probability of $\boldsymbol{x}^{\boldsymbol{i}} \to \boldsymbol{x}^{\boldsymbol{i+1}}$ is at least $\frac{1}{P_{\max}} \cdot \frac{n-1-i}{n}(1-\frac{1}{n})^{n-1} \geq \frac{n-1-i}{en(3n-1)}$, since it is sufficient to select $\boldsymbol{x}^{\boldsymbol{i}}$ in line 4 and then flip only one of its first $n-1-i$ 0-bits. Thus,

$$\mathbb{E}[T_2] \leq \sum_{i=0}^{n-2} \frac{en(3n-1)}{n-1-i} \leq en(3n-1)(1+\log n).$$

Combining the above two phases leads to the total expected number of iterations for finding the optimal solution:

$$\mathbb{E}[T] \leq \mathbb{E}[T_1] + \mathbb{E}[T_2] \leq en(3n-1)(2+2\log n).$$

[Multi-bit search] The idea is that flipping two 0s in the first $n-1$ bits of $\{0\}^n$ simultaneously can find the solution $\boldsymbol{x}^{\boldsymbol{2}}$, which has a better f/g value than the solution $\boldsymbol{x}$ with $|\boldsymbol{x}| = 2 \wedge x_n = 1$ found by GreedRatio; then following the path $\boldsymbol{x}^{\boldsymbol{2}} \to \boldsymbol{x}^{\boldsymbol{3}} \to \cdots \to \boldsymbol{x}^{\boldsymbol{n-1}}$ can find the optimal solution.

Using the same analysis as multi-path search, PORM can first find the solution $\{0\}^n$ in at most $en(3n-1)(1+\log n)$ expected number of iterations. After that, selecting $\{0\}^n$ in line 4 and only flipping any two 0s in its first $n-1$ bits can generate the solution $\boldsymbol{x}^{\boldsymbol{2}}$ with probability at least $\frac{1}{P_{\max}} \cdot \frac{\binom{n-1}{2}}{n^2}(1-\frac{1}{n})^{n-2} \geq \frac{(n-1)(n-2)}{2e(3n-1)n^2}$. Compared with the solution $\boldsymbol{x}$ with $|\boldsymbol{x}| = 2 \wedge x_n = 1$, $\frac{f(\boldsymbol{x}^{\boldsymbol{2}})}{g(\boldsymbol{x}^{\boldsymbol{2}})} = \frac{(n+1)n^2-n+1}{2n^2-2} < \frac{(n+1)n^2-n-2}{2n^2-5} = \frac{f(\boldsymbol{x})}{g(\boldsymbol{x})}$. PORM then can easily follow the path $\boldsymbol{x}^{\boldsymbol{2}} \to \boldsymbol{x}^{\boldsymbol{3}} \to \cdots \to \boldsymbol{x}^{\boldsymbol{n-1}}$ to find the optimal solution. The total expected number of iterations is at most

$$en(3n-1)(1+\log n) + \frac{2e(3n-1)n^2}{(n-1)(n-2)} + \sum_{i=2}^{n-2} \frac{en(3n-1)}{n-1-i}$$
$$\leq en(3n-1)(2+2\log n).$$

Taking the minimum of the expected number of iterations for finding the optimal solution by backward search, multi-path search and multi-bit search, the theorem holds. $\square$

6 Empirical Study

We conducted experiments on F-measure maximization to investigate the actual performance of PORM. We compare PORM only with GreedRatio, since it is the previous algorithm achieving the best empirical performance [Bai *et al.*, 2016]. The generalized form of F-measure is used for evaluation, i.e., $F_p(X) = (|\Gamma(X) \cap O|)/(p|O| + (1-p)|\Gamma(X)|)$. We will test $p = \{0.2, \ldots, 0.8\}$. Note that the F-measure in Definition 4 is just $F_{0.5}$. For PORM, the number T of iterations is set to $\lfloor 3en^2(2+\log|\Gamma(X_{\text{init}})|)\rfloor$ (where X_{init} is the initial subset generated by PORM), as suggested by Theorem 2. Note that we have used the lower bound 1 for $\frac{1}{1-\kappa_f}$.

We use synthetic (*syn-100*, *syn-1000*) as well as real-world (*ldc-100*, *ldc-1000*) data sets. For *syn-100*, a bipartite graph $G(V, W, E)$ with $|V| = |W| = 100$ is randomly generated by adding one edge between $v \in V$ and $w \in W$ independently with probability 0.05; the target set O with $|O| = 20$ is randomly chosen from W. For *syn-1000*, $|V| = |W| = 1000$, $|O| = 100$ and each edge is added with probability 0.01. *ldc-100* and *ldc-1000* contain 100 and 1000 English sentences, respectively, which are randomly sampled from the LDC2002E18 text data (`https://www.ldc.upenn.edu/`).

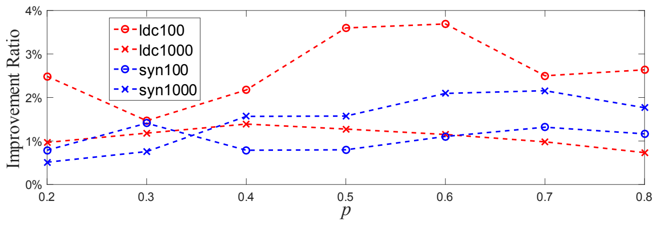

Figure 1: Ratio of improvement of PORM to GreedRatio.

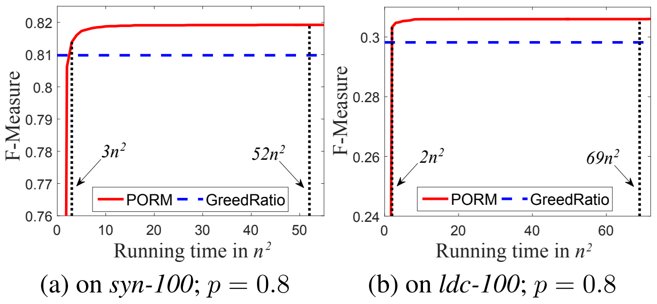

Figure 2: Performance v.s. running time of PORM.

Their target sets contain 1000 randomly chosen words. For each data set, we generate ten random instances and report the average results. Note that since PORM is a randomized algorithm, its run is further repeated 10 times independently for each data set instance. Figure 1 shows the percentages of the solution quality that PORM improves from GreedRatio, where we can observe that PORM is always better than GreedRatio and can have more than 3% improvement.

Comparing the running time (in the number of function calculations), GreedRatio takes the time in the order of n^2; PORM is set to use $3en^2(2+\log|\Gamma(X_{\text{init}})|)$ time according to the theoretical upper bound (i.e., a worst case) for PORM being good. We empirically examine how effective PORM is in practice. By selecting GreedRatio as the baseline, we plot the curve of the F-measure over the running time for PORM on *syn-100* and *ldc-100* with $p = 0.8$, as shown in Figure 2. The x-axis is in n^2, the running time of GreedRatio. We can observe that PORM takes about only 6% (3/52) and 3% (2/69) of the worst-case running time to achieve a better performance, respectively. This implies that PORM can be efficient in practice.

7 Conclusion

In this paper, we study the problem of minimizing the ratio f/g, where f is monotone submodular and g is monotone. We prove the approximation bound of GreedRatio for the problem, which even improves the previous result for g being submodular. We then propose a new algorithm PORM, and prove that PORM can achieve the same general approximation guarantee as GreedRatio, but can have a better ability of avoiding local optima. The empirical results on the application of F-measure maximization verify the superior performance of PORM.

References

[Bai *et al.*, 2016] W. Bai, R. Iyer, K. Wei, and J. Bilmes. Algorithms for optimizing the ratio of submodular functions. In *Proceedings of the 33rd International Conference on Machine Learning (ICML'16)*, pages 2751–2759, New York, NY, 2016.

[Conforti and Cornuéjols, 1984] M. Conforti and G. Cornuéjols. Submodular set functions, matroids and the greedy algorithm: Tight worst-case bounds and some generalizations of the Rado-Edmonds theorem. *Discrete Applied Mathematics*, 7(3):251–274, 1984.

[Das and Kempe, 2011] A. Das and D. Kempe. Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection. In *Proceedings of the 28th International Conference on Machine Learning (ICML'11)*, pages 1057–1064, Bellevue, WA, 2011.

[Doerr *et al.*, 2012] B. Doerr, D. Johannsen, and C. Winzen. Multiplicative drift analysis. *Algorithmica*, 64(4):673–697, 2012.

[Iyer and Bilmes, 2012] R. Iyer and J. Bilmes. Algorithms for approximate minimization of the difference between submodular functions, with applications. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI'12)*, pages 407–417, Catalina Island, CA, 2012.

[Iyer and Bilmes, 2013] R. Iyer and J. Bilmes. Submodular optimization with submodular cover and submodular knapsack constraints. In *Advances in Neural Information Processing Systems 26 (NIPS'13)*, pages 2436–2444, Lake Tahoe, NV, 2013.

[Iyer *et al.*, 2013] R. Iyer, S. Jegelka, and J. Bilmes. Curvature and optimal algorithms for learning and minimizing submodular functions. In *Advances in Neural Information Processing Systems 26 (NIPS'13)*, pages 2742–2750, Lake Tahoe, NV, 2013.

[McLachlan, 2004] G. J. McLachlan. *Discriminant Analysis and Statistical Pattern Recognition*. Wiley Interscience, 2004.

[Nemhauser *et al.*, 1978] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions – I. *Mathematical Programming*, 14(1):265–294, 1978.

[Qian *et al.*, 2015a] C. Qian, Y. Yu, and Z.-H. Zhou. On constrained Boolean Pareto optimization. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)*, pages 389–395, Buenos Aires, Argentina, 2015.

[Qian *et al.*, 2015b] C. Qian, Y. Yu, and Z.-H. Zhou. Subset selection by Pareto optimization. In *Advances in Neural Information Processing Systems 28 (NIPS'15)*, pages 1765–1773, Montreal, Canada, 2015.

[Qian *et al.*, 2016] C. Qian, J.-C. Shi, Y. Yu, K. Tang, and Z.-H. Zhou. Parallel Pareto optimization for subset selection. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*, pages 1939–1945, New York, NY, 2016.

[Rijsbergen and Joost, 1974] V. Rijsbergen and C. Joost. Foundation of evaluation. *Journal of Documentation*, 30(4):365–373, 1974.

[Shi and Malik, 2000] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[Vondrák, 2010] J. Vondrák. Submodularity and curvature: The optimal algorithm. *RIMS Kokyuroku Bessatsu B*, 23:253–266, 2010.