IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON BIG DATA LEARNING AND DISCOVERY

Received August 7, 2018, accepted September 18, 2018, date of publication September 24, 2018, date of current version October 17, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2871668

Streaming Algorithms for News and Scientific Literature Recommendation: Monotone Submodular Maximization With a d-Knapsack Constraint

QILIAN YU[1], (Student Member, IEEE), LI XU[2], (Member, IEEE), AND SHUGUANG CUI[1], (Fellow, IEEE)

[1]Department of Electrical and Computer Engineering, University of California at Davis, Davis, CA 95616, USA
[2]Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China

Corresponding author: Shuguang Cui (sgcui@ucdavis.edu)

This work was supported in part by DoD under Grant HDTRA1-13-1-0029, in part by NSFC under Grant 61328102, Grant 61629101, and Grant 91746301, and in part by NSF under Grants DMS-1622433, AST-1547436, ECCS-1508051/1659025, and CNS-1343155.

ABSTRACT Submodular optimization plays a significant role in combinatorial problems, since it captures the structure of the edge cuts in graphs, the coverage of sets, and so on. Many data mining and machine learning problems can be cast as submodular maximization problems with applications in recommendation systems and data diversification. In this paper, we focus on the problem of maximizing a monotone submodular function subject to a d-knapsack constraint, for which we propose a streaming algorithm that achieves a $((1/1+2d)-\epsilon)$-approximation of the optimal value, while it only needs one single pass through the data set without storing all the data in the memory. In our experiments, we extensively evaluate the effectiveness of our proposed algorithm via two applications: news recommendation and scientific literature recommendation. It is observed that the proposed streaming algorithm achieves both execution speedup and memory saving by several orders of magnitude, compared with existing approaches.

INDEX TERMS Submodular optimization, streaming algorithm, news recommendation, scientific literature recommendation.

I. INTRODUCTION AND BACKGROUND

As data volumnes grow exponentially due to the increasingly developed multimedia techniques, smart phones, and the dramatically expanded online social networks, our society enters the big data era. One of the main challenges that data scientists and data engineers are facing is how to process and analyze the unprecedented large datasets. Different from a decade ago, data sources nowadays are more heterogeneous, consisting of plain texts, colorful images, and high-definition videos. These challenges require more computationally efficient design of data processing and analyzing algorithms. Submodular optimization plays an important role in combinatorial optimization since it captures the structure of edge cuts in graphs, the coverage of sets, and so on. Given the rich theoretical and practical features of submodular optimization, submodular functions have been adopted to preprocess massive data in order to reduce the computational complexity. For example, in the kernel-based machine learning [1], [2], the most representative subset of data is first selected in order to decrease the dimension of the feature space, by solving a submodular maximization problem under a cardinality constraint. Besides, such submodular optimization models have been extended to address data summarization problems [3].

Although many tasks in data mining and machine learning problems can be posed as submodular optimization problems, such problems with certain constraints like the cardinality constraints and the knapsack constraints are NP-hard. A simple greedy algorithm to solve the submodular maximization problem under a cardinality constraint was developed in [4], which achieves a $(1-e^{-1})$-approximation of the optimal solution with a much lower computational complexity. When the main computing memory can store the whole dataset, such a greedy algorithm can be easily implemented in a wide range of applications. Nevertheless, as it requires the full

access to the whole dataset, such a greedy algorithm is less practical when it is applied to large-scale problems, since the computation and memory resources are often limited. In addition, when data samples arrive much faster than the speed that the main memory stores data, such a traditional greedy algorithm is far from being adequate to process the data.

To levitate challenges described above, it is necessary to propose an algorithm that operates in a streaming fashion. Such a streaming algorithm is required to perform data processing and analyzing by storing a limited proportion of the data into the memory, where the final result and solution can be provided right at the end of the data stream. The main characteristic of this streaming fashion algorithm is that the full access to the dataset is not required, which dramatically saves memory and computation resources. Based on this idea, the Sieve-Streaming algorithm [5] was introduced to solve the submodular maximization problem with a cardinality constraint. Here the cardinality constraint is a special case of a d-knapsack constraint [6], where each element weight is set to be one and $d = 1$. However, the problem with non-uniform element weights or multiple knapsack constraints cannot be solved by such a streaming algorithm.

In this paper, subject to a general d-knapsack constraint, a novel streaming fashion algorithm is proposed to solve the monotone submodular maximization problem. By only accessing data in one pass, it provides an approximation solution with a $\left(\frac{1}{1+2d} - \epsilon\right)$ approximation factor for any $\epsilon > 0$. Independent of the dataset size, the proposed algorithm only requires $O\left(\frac{b \log b}{d\epsilon}\right)$ memory and $O\left(\frac{\log b}{\epsilon}\right)$ computation per element with b being the standardized d-knapsack capacity. In our experiments, compared with the classical greedy algorithm developed in [7], the proposed streaming algorithm achieves over 10,000 times running time reduction with a similar performance.

The rest of our paper is organized as follows. The problem formulation and the main contributions are introduced in Section II. In Section III we present the streaming algorithm by introducing some mild assumptions. In Section IV we implement our proposed streaming algorithms with two applications in news and scientific literature recommendation. We finally make our conclusions in Section V.

II. FORMULATION AND MAIN CONTRIBUTIONS

A. PROBLEM FORMULATION

Let $V = \{1, 2, \ldots, n\}$ be the ground set and $f : 2^V \to [0, \infty)$ be a nonnegative set function on the subsets of V. For any subset S of V, we denote the characteristic vector of S by $\mathbf{x}_S = (x_{S,1}, x_{S,2}, \ldots, x_{S,n})$, where for $1 \leq j \leq n$, $x_{S,j} = 1$, if $j \in S$; $x_{S,j} = 0$, otherwise. For $S \subseteq V$ and $r \in V$, the marginal gain of f with respect to S and r is defined to be

$$\Delta_f(r|S) \triangleq f(S \cup \{r\}) - f(S),$$

which quantifies the increase in the utility function $f(S)$ when r is added into subset S. A function f is submodular if it satisfies that for any $A \subseteq B \subseteq V$ and $r \in V \setminus B$, the diminishing returns condition holds:

$$\Delta_f(r|B) \leq \Delta_f(r|A).$$

Also, f is said to be a monotone function, if for any $S \subseteq V$ and $r \in V$, $\Delta_f(r|S) \geq 0$. For now, we adopt the common assumption that f is given in terms of a black box that computes $f(S)$ for any $S \subseteq V$. In Sections III-A, III-B, III-C, we will discuss the case when the submodular function is independent [8] of the ground set V (*i.e.,* for any $S \subseteq V$, $f(S)$ depends on only S, not $V \setminus S$), and in Section III-D, we will discuss the setting where the value of $f(S)$ depends on not only the subset S but also the ground set V.

Next, we introduce the d-knapsack constraint. Let $\mathbf{b} = (b_1, b_2, \ldots, b_d)^T$ be a d-dimensional budget vector, where for $1 \leq i \leq d$, $b_i > 0$ is the budget corresponding to the i-th resource. Let $C = (c_{i,j})$ denote a $d \times n$ matrix, whose (i, j)-th entry $c_{i,j} > 0$ is the weight of the element $j \in V$ with respect to the i-th knapsack resource constraint. Then the d-knapsack constraint can be expressed by $C\mathbf{x}_S \leq \mathbf{b}$. Note that "$C\mathbf{x}_S \leq \mathbf{b}$" means that $C\mathbf{x}_S$ is element-wise less than or equal to $\mathbf{b}$. The problem for maximizing a monotone submodular function $f : 2^V \to [0, \infty)$ subject to a d-knapsack constraint can be formulated as

$$\begin{aligned} &\underset{S \subseteq V}{\text{maximize}}\ f(S) \\ &\text{subject to } C\mathbf{x}_S \leq \mathbf{b}. \end{aligned} \tag{1}$$

We aim to **MA**ximize a monotone **S**ubmodular set function subject to a d-**K**napsack constraint, which is called d-**MASK** for short. Without loss of generality, for $1 \leq i \leq d, 1 \leq j \leq n$, we assume that $c_{i,j} \leq b_i$. That is, no entry in C has a larger weight than the corresponding knapsack budget, since otherwise the corresponding element is never selected into S.

For the sake of simplicity, we here standardize Problem (1). Let

$$b \triangleq \max_{1 \leq i \leq d} b_i \quad \text{and } c' \triangleq \min_{1 \leq i \leq d, 1 \leq j \leq n} bc_{i,j}/b_i.$$

For $1 \leq i \leq d$, $1 \leq j \leq n$, we replace each $c_{i,j}$ with $bc_{i,j}/b_i c'$ and b_i with b/c'. We then create a new matrix D by concatenating C and $\mathbf{b}$ over columns. That is, $D = (d_{i,j})$ is a $d \times (n+1)$ matrix, such that, for $1 \leq i \leq d$, $d_{i,j} = c_{i,j} \geq 1$ if $1 \leq j \leq n$; $d_{i,j} = b$ if $j = n+1$. The standardized problem has the same optimal solution as Problem (1). In the rest of the paper, we only consider the standardized version of the d-MASK problem.

B. RELATED WORK AND MAIN RESULTS

Streaming fashion algorithms make submodular optimization more powerful when the problem of data mining and machine learning becomes massive and large-scale, since the streaming algorithm only accesses a small proportion of the dataset at each operation and is able to provide the solution right after the algorithm goes through the whole dataset in a fixed number of times (usually just one round).

TABLE 1. Comparison of approximation guarantees and computation costs.

	Best Performance Known Algorithms		Proposed Streaming Algorithms	
	Greedy Algorithm[7]	Greedy Algorithm [14]		
	Approx. Factor	Comput. Cost	Approx. Factor	Comput. Cost
1-Knapsack Constraint	$1-e^{-1}$	$O(n^5)$	$1/(1+2d)-\epsilon$	$O(n\log b/\epsilon)$
d-Knapsack Constraint	$1-e^{-1}-\epsilon$	Polynomial		

A greedy approximation algorithm was first developed in [4] to solve the special case of Problem (1) with $d = 1$ and all entries of C being uniform. Such a special case is equivalent to maximizing a monotone submodular function subject to a cardinality constraint, which has been proved to be NP-hard [4]. Later, another greedy algorithm [7] was suggested to solve a more general problem, which is maximizing a monotone submodular function under a single knapsack constraint, where each entry of C can take any positive value. Such a problem (1-MASK) is known as the budgeted submodular maximization problem, which has been proved to be NP-hard, and the algorithm [7] can achieve a $(1 - e^{-1})$-approximation of the optimal value with $O(n^5)$ computation cost. In this paper, we study d-MASK problem, a more general setting of the problems described above, which maximizes a submodular function under multiple budgeted or knapsack constraints. As one of pioneer works, an approximation algorithm was proposed in [6] with a $(1-e^{-1}-\epsilon)$ approximation factor for any $\epsilon > 0$.

To address the issue that above greedy algorithms are not capable to handle the dataset whose size is larger than the main memory capacity, streaming fashion algorithms were proposed [5], [13] to solve the submodular maximization problem under a cardinality constraint and a single knapsack constraint respectively. For the most general d-MASK problem, although an improved algorithm was proposed in [15], which runs for $O(1/\delta)$ rounds in MapReduce [16] for a constant δ, and provides an $\Omega(1/d)$-approximation, an $O(\log n)$ blowup communication complexity among various parts in the algorithm prevents it from being practical [17]. Another $\Omega(1/d)$-approximation MapReduce approach was mentioned in [15] that can be executed in a streaming fashion, but detailed analysis was not provided. Table 1 shows the comparison among the approximation guarantees and computational complexity of the aforementioned algorithms against our proposed algorithm.

To our best knowledge, we are the first to propose an efficient streaming algorithm to maximize a monotone submodular function subject to a d-knapsack constraint, with

- a constant-factor approximation guarantee;
- no full access required to the dataset;
- a single pass accessto the dataset;
- $O(b\log b/\epsilon)$ memory consumption;
- $O(n\log b/\epsilon)$ computation cost.

In the following section, we present the proposed algorithm by introducing some mild assumptions, which could be removing later.

III. STREAMING ALGORITHMS FOR MAXIMIZING MONOTONE SUBMODULAR FUNCTIONS

A. SPECIAL CASE: ONE CARDINALITY CONSTRAINT

We first consider a special case of the d-MASK problem: maximizing a submodular function subject to one cardinality constraint:

$$\begin{aligned} &\underset{S\subseteq V}{\text{maximize}}\, f(S) \\ &\text{subject to } |S| \le k. \end{aligned} \tag{2}$$

Nemhauser [4] proved this problem is NP-hard and proposed a classical greedy algorithm. At each step of the algorithm, as we explained earlier, the element with the largest marginal value is added to the solution set. This operation, in fact, reduces the "gap" to the optimal solution by a significant amount. Formally, if element j is added to the current solution set S by the greedy algorithm, the marginal value $\Delta_f(j|S)$ of this picked element should be at least above certain threshold. Badanidiyuru *et al.* [5] developed the so-called Sieve-Streaming algorithm, where the threshold for the marginal value is set to be $(\text{OPT}/2 - f(S))/(k - |S|)$, where S is the current solution set, k is the maximum allowed number of elements in S, and OPT is the optimal value of the optimization problem. In our paper, for this submodular maximization problem under a single cardinality constraint, we first introduce a simple streaming algorithm under the assumption that we have the knowledge of the optimal value of the problem.

Algorithm 1 Simple Streaming Algorithm

1: Input: v such that $\alpha\text{OPT} \le v \le \text{OPT}$, for some $\alpha \in (0, 1]$.
2: $S := \emptyset$.
3: **for** $j := 1$ **to** n
4: **if** $f(S \cup \{j\}) - f(S) \ge \frac{v}{2k}$ and $|S| \le k$ **then**
5: $S := S \cup \{j\}$.
6: **end if**
7: **end for**
8: **return** S.

Theorem 1: Given an estimated optimal value $v \in [\alpha OPT, OPT]$, where $\alpha \in (0, 1]$. The simple streaming algorithm (Algorithm 1) produces a solution set S such that

$$f(S) \ge \frac{\alpha}{2} OPT.$$

Proof: Given $v \in [\alpha\text{OPT}, \text{OPT}]$, by selecting elements that satisfying the condition in Line 4 of Algorithm 1, at the end of the algorithm, we might have the solution set S with the

cardinality less than k or exactly equal to k. In the following, let us discuss these two cases.

Case 1: $|S| = k$. For $1 \leq i \leq k$, let a_i be the element added to S in the i-th iteration of the for-loop. Then we obtain

$$\begin{aligned} f(S) &= f(\{a_1, a_2, \ldots, a_k\}) \geq f(\{a_1, a_2, \ldots, a_k\}) - f(\emptyset) \\ &= \sum_{i=1}^{k} \left[f(\{a_1, a_2, \ldots, a_i\}) - f(\{a_1, a_2, \ldots, a_{i-1}\}) \right]. \end{aligned}$$

By the condition in Line 4 of Algorithm 1, for $1 \leq i \leq k$, we have

$$f(\{a_1, a_2, \ldots, a_i\}) - f(\{a_1, a_2, \ldots, a_{i-1}\}) \geq \frac{v}{2k},$$

and hence

$$f(S) \geq \frac{v}{2k} \cdot k \geq \frac{\alpha}{2}\text{OPT}.$$

Case 2: $|S| < k$. Let $\bar{S} = S^* \backslash S$, where S^* is the optimal solution to the Problem (2). For each element $a \in \bar{S}$, we have

$$f(S \cup \{a\}) - f(S) < \frac{v}{2k}.$$

Since f is monotone submodular, we obtain

$$\begin{aligned} f(S^*) - f(S) &= f(S \cup \bar{S}) - f(S) \\ &\leq \sum_{a \in \bar{S}} [f(S \cup \{a\}) - f(S)] < \frac{v}{2k} \cdot k \leq \frac{1}{2} f(S^*), \end{aligned}$$

which implies that

$$f(S) > \frac{1}{2} f(S^*) = \frac{1}{2}\text{OPT} \geq \frac{\alpha}{2}\text{OPT}.$$

■

This simple streaming algorithm produces a solution by visiting every element in the ground set only once. But it requires the knowledge of the optimal value of the problem. Besides, when the elements have non-uniform weights, this algorithm does not work. To deal with the problem with non-uniform weights and more than one constraint, we are going to modify the greedy rule and take the weight-dependent marginal values into account in a streaming fashion.

B. GENERAL CASE: MULTIPLE KNAPSACK CONSTRAINTS

In order to get the desirable output, in this subsection, we first assume we have some knowledge of OPT, and then remove this assumption by estimating OPT based on the maximum value per weight of any single element. At the end, we will remove all assumptions to develop the final version of the streaming algorithm for the general case of a d-MASK problem.

Suppose that we know a value v such that $\alpha\text{OPT} \leq v \leq \text{OPT}$ for some $0 < \alpha \leq 1$. That is, we know an approximation of OPT up to a constant factor α. We then construct the following algorithm to choose a subset S with the knowledge of the optimal value of the problem. The main idea behind this algorithm is to select each element that satisfies certain requirements such that the utility of the selected elements in the knapsack can be as close to v as possible.

Algorithm 2 OPT-KNOWN-d-MASK

1: Input: v such that $\alpha\text{OPT} \leq v \leq \text{OPT}$, for some $\alpha \in (0, 1]$.
2: $S := \emptyset$.
3: **for** $j := 1$ **to** n
4: **if** $c_{i,j} \geq \frac{b}{2}$ and $\frac{f(\{j\})}{c_{i,j}} \geq \frac{2v}{b(1+2d)}$ for some $i \in [1, d]$ **then**
5: $S := \{j\}$.
6: **return** S.
7: **end if**
8: **if** $\sum_{l \in S \cup \{j\}} c_{i,l} \leq b$ and $\frac{\Delta_f(j|S)}{c_{i,j}} \geq \frac{2v}{b(1+2d)}$ for all $i \in [1, d]$ **then**
9: $S := S \cup \{j\}$.
10: **end if**
11: **end for**
12: **return** S.

At the beginning of the algorithm, the solution set S is set to be an empty set. The algorithm will terminate when either we find an element $j \in V$ satisfying

$$c_{i,j} \geq \frac{b}{2} \quad \text{and} \quad \frac{f(\{j\})}{c_{i,j}} \geq \frac{2v}{b(1+2d)} \text{ for some } i \in [1, d], \tag{3}$$

or we finish one pass through the dataset. Here we define that an element $j \in V$ is a *big element* if it satisfies (3). When the algorithm finds a big element a, it simply outputs $\{a\}$ and terminates. The following lemma shows that $\{a\}$ is already a good enough solution.

Lemma 1: Assume the input v satisfies $\alpha OPT \leq v \leq OPT$, and V has at least one big element. The output S of Algorithm 2 satisfies

$$f(S) \geq \frac{\alpha}{1+2d} OPT.$$

Proof: Let a be the first big element that Algorithm 2 finds. Then according to Algorithm 2, $\{a\}$ is output and the algorithm terminates. Therefore, by (3), we have

$$f(S) = f(\{a\}) \geq \frac{2v}{b(1+2d)} \cdot \frac{b}{2} = \frac{v}{1+2d} \geq \frac{\alpha}{1+2d}\text{OPT}.$$

■

When V does not contain any big elements, during the data streaming, an element j is added to the solution set S if 1) the marginal value per weight for each knapsack constraint $\Delta_f(j|S)/c_{i,j}$ is at least $\beta v/b$ for $1 \leq i \leq d$, and 2) the overall d-knapsack constraint is still satisfied. In this paper, we set $\beta = \frac{2d}{1+2d}$, which gives us the best approximation guarantee as shown in the proof of Theorem 2. The following lemma shows the property of the output of Algorithm 2.

Lemma 2: Assume that V has no big elements. The output S of Algorithm 2 has the following two properties:

1) *There exists an ordering $a_1, a_2, \ldots, a_{|S|}$ of the elements in S, such that for all $0 \leq t < |S|$ and $1 \leq i \leq d$,*

we have

$$\frac{\Delta_f(a_{t+1}|S_t)}{c_{i,a_t}} \geq \frac{2v}{b(1+2d)}, \tag{4}$$

where $S_t = \{a_1, a_2, \ldots, a_t\}$.

2) *Assume that for* $1 \leq i \leq d$, $\sum_{t=1}^{|S|} c_{i,a_t} \leq b/2$. *Then for each* $a_j \in V$, *there exists an index* $\mu(a_j)$, *with* $1 \leq \mu(a_j) \leq d$ *such that*

$$\frac{\Delta_f(a_j|S)}{c_{\mu(a_j),a_j}} < \frac{2v}{b(1+2d)}.$$

Proof: 1) For $0 \leq t < |S|$, at the $(t+1)$-th step of the algorithm, assume that a_{t+1} is the element added to the current solution set $S_t = \{a_1, a_2, \ldots, a_t\}$. Then $a_1, a_2, \ldots, a_{|S|}$ forms an ordering satisfying (4).

2) By contradiction, assume that there exists $j \in V$ such that for $1 \leq i \leq d$, we have

$$\frac{f(S \cup \{j\}) - f(S)}{c_{i,j}} \geq \frac{2v}{b(1+2d)}.$$

Since j is not a big element and f is submodular, we have $c_{i,j} < b/2$, for $1 \leq i \leq d$. Then j can be added into S, where a contradiction occurs. ∎

We then establish the following theorem to show that Algorithm 2 produces an $\left(\frac{\alpha}{1+2d}\right)$-approximation of the optimal solution to Problem (1).

Theorem 2: Assuming that the input v *satisfies* $\alpha OPT \leq v \leq OPT$, *Algorithm 2 has the following properties:*

- *It outputs* S *that satisfies* $f(S) \geq \frac{\alpha}{1+2d} OPT$;
- *It only goes one pass over the dataset, stores at most* $O(b)$ *elements, and has* $O(d)$ *computational complexity per element.*

Proof: If V contains at least one big element, by Lemma 1, we have

$$f(S) \geq \frac{\alpha}{1+2d} \text{ OPT};$$

otherwise, we discuss the following two cases:

Case 1: $\sum_{j \in S} c_{i,j} \geq b/2$, for some $i \in [1, d]$. By the submodularity of f and Property 1) in Lemma 2, we have

$$f(S) \geq \frac{2v}{b(1+2d)} \sum_{j \in S} c_{i,j} \geq \frac{v}{1+2d} \geq \frac{\alpha}{1+2d} \text{ OPT}.$$

Case 2: $\sum_{j \in S} c_{i,j} < b/2$, for all $i \in [1, d]$. Let S_i^* be the set of elements $a_j \in S^* \backslash S$ such that $\mu(a_j) = i$, for $1 \leq i \leq d$. Then we have $S^* \setminus S = \bigcup_{1 \leq i \leq d} S_i^*$. With the help of the submodularity of f and Property 2) in Lemma 2, we obtain

$$f(S \cup S_i^*) - f(S) \leq \frac{2v}{b(1+2d)} \sum_{a_j \in S_i^*} c_{\mu(a_j),a_j} < \frac{v}{1+2d},$$

for $1 \leq i \leq d$. Then we have

$$\begin{aligned} f(S^*) - f(S) &= f(S \cup (S^* \setminus S)) - f(S) \\ &\leq \sum_{1 \leq i \leq d} [f(S \cup S_i^*) - f(S)] < \frac{dv}{1+2d}, \end{aligned}$$

and further,

$$f(S) > f(S^*) - \frac{dv}{1+2d} \geq \frac{1}{1+2d} \text{OPT}.$$

In both cases, we conclude

$$f(S) \geq \frac{\alpha}{1+2d} \text{ OPT}.$$

Since we have $c_{i,j} \geq 1$ for all $i \in [1, d], j \in [1, n]$, we store at most $O(b)$ elements during the algorithm. In the for-loop, we compare the values at most d times. Then the computation cost per element in the algorithm is $O(d)$. ∎

We can obtain an approximation of the optimal value OPT by solving the d-MASK problem via Algorithm 2. But in certain scenarios, requiring the knowledge of an approximation to the optimization problem and utilizing the approximation in Algorithm 2 lead to a chicken and egg dilemma. That is, we have to first estimate OPT and then use it to compute OPT. Fortunately, even in such scenarios, we still have the following lemma to estimate OPT if we know $m \triangleq \max_{1 \leq i \leq d, 1 \leq j \leq n} f(\{j\})/c_{i,j}$, the maximum value per weight of any single element.

Lemma 3: Let

$$Q = \Big\{ [1+(1+2d)\epsilon]^l \mid l \in \mathbb{Z}, \frac{m}{1+(1+2d)\epsilon} \leq [1+(1+2d)\epsilon]^l \leq bm \Big\}$$

for some ϵ *with* $0 < \epsilon < \frac{1}{1+2d}$. *Then there exists at least some* $v \in Q$ *such that* $[1-(1+2d)\epsilon]OPT \leq v \leq OPT$.

Proof: First, choose $i' \in [1, d]$, $j' \in [1, n]$ such that $f(\{j'\})/c_{i',j'} = m$. Since $c_{i',j'} \geq 1$, we have

$$\text{OPT} \geq f(\{j'\}) = m c_{i',j'} \geq m.$$

Also, let $\{j_1, j_2, \ldots, j_t\}$ be a subset of V such that $f(\{j_1, j_2, \ldots, j_t\}) = \text{OPT}$. Then by the submodularity of f,

$$\begin{aligned} \text{OPT} &= f(\emptyset) + \sum_{i=1}^{t} [f(\{j_1, j_2, \ldots, j_i\}) - f(\{j_1, j_2, \ldots, j_{i-1}\})] \\ &\leq f(\emptyset) + \sum_{i=1}^{t} [f(\{j_i\}) - f(\emptyset)] \\ &\leq \sum_{i=1}^{t} f(\{j_i\}) \leq m \sum_{i=1}^{t} c_{1,j_i} \leq bm. \end{aligned}$$

Setting $v = [1+(1+2d)\epsilon]^{\lfloor \log_{1+(1+2d)\epsilon} \text{OPT} \rfloor}$, we then obtain

$$\frac{m}{1+(1+2d)\epsilon} \leq \frac{1}{1+(1+2d)\epsilon} \text{OPT} \leq v \leq \text{OPT} \leq bm,$$

and

$$v \geq \frac{1}{1+(1+2d)\epsilon} \text{OPT} \geq [1-(1+2d)\epsilon] \text{OPT}.$$

∎

Based on Lemma 3, we propose the following algorithm that gets around the chick and egg dilemma. The idea of this algorithm is to utilize m to estimate the candidate optimal value, where we could use Algorithm 2 to find the solution

Algorithm 3 m-KNOWN-d-MASK

Input: m.
$Q := \{[1+(1+2d)\epsilon]^l | l \in \mathbb{Z},$
$\frac{m}{1+(1+2d)\epsilon} \leq [1+(1+2d)\epsilon]^l \leq bm\}$.
for $v \in Q$
  $S_v := \emptyset$.
end for
for $v \in Q$
  for $j := 1$ **to** n
    if $c_{i,j} \geq \frac{b}{2}$ and $\frac{f(\{j\})}{c_{i,j}} \geq \frac{2v}{b(1+2d)}$ for some $i \in [1,d]$ **then**
      $S_v := \{j\}$.
      break.
    end if
    if $\sum_{l \in S \cup \{j\}} c_{i,l} \leq b$ and $\frac{\Delta_f(j|S)}{c_{i,j}} \geq \frac{2v}{b(1+2d)}$ for all $i \in [1,d]$ **then**
      $S_v := S_v \cup \{j\}$.
    end if
  end for
end for
$S := \underset{S_v, v \in Q}{\operatorname{argmax}} f(S_v)$.
return S.

set corresponding to the candidate optimal value. At the end of Algorithm 3, we output the solution set which provides the largest utility.

Then we establish the following theorem to show that the above algorithm achieves a $\left(\frac{1}{1+2d} - \epsilon\right)$-approximation guarantee, and requires $O\left(\frac{b \log b}{\epsilon}\right)$ memory and $O\left(\frac{\log b}{\epsilon}\right)$ computational complexity per element. Since the ground set V contains n elements, the overall computational complexity of the proposed algorithm is $O(n \log b/\epsilon)$.

Theorem 3: With m known, Algorithm 3 has the following properties:

- *It outputs S that satisfies* $f(S) \geq \left(\frac{1}{1+2d} - \epsilon\right)$ *OPT;*
- *It goes one pass over the dataset, and has* $O\left(\frac{n \log b}{\epsilon}\right)$ *computational complexity.*

Proof: By Lemma 3, we choose $v \in Q$ such that $[1-(1+2d)\epsilon]\text{OPT} \leq v \leq \text{OPT}$. Then by Theorem 2, the output S satisfies

$$f(S) \geq \frac{1-(1+2d)\epsilon}{1+2d}\text{OPT} = \left(\frac{1}{1+2d} - \epsilon\right)\text{OPT}.$$

Notice that there are at most $\lceil \log_{1+(1+2d)\epsilon} b \rceil + 1$ (of order $\frac{\log b}{d\epsilon}$) elements in Q. At the end of the algorithm, S_v with the largest function value will be picked to be the output. Since S contains at most b elements, Algorithm 3 stores at most $O\left(\frac{b \log b}{d\epsilon}\right)$ elements and has $O\left(\frac{\log b}{\epsilon}\right)$ computational complexity per element. ■

Introducing the maximum marginal value per weight m avoids the chicken and egg dilemma in Algorithm 2. With m known, Algorithm 3 needs only one pass over the dataset. However, we need an extra pass through the dataset to obtain the value of m. Such an algorithm is practically useful for applications where multiple-round scans are allowed. In the following, we will develop our final one-pass streaming algorithm with m unknown for applications where multiple scans through a large-scale dataset are prohibited.

We modify the estimation candidate set Q into $\{[1+(1+2d)\epsilon]^l | l \in \mathbb{Z}, \frac{m}{1+(1+2d)\epsilon} \leq [1+(1+2d)\epsilon]^l \leq 2bm\}$, and maintain the variable m that holds the current maximum marginal value per weight of all single element. During the data streaming, if a big element a is observed, the algorithm simply outputs $\{a\}$ and terminates. Otherwise, the algorithm will update m and the estimation candidate set Q. If the marginal value per weight for each knapsack constraint $\Delta_f(j|S)/c_{i,j}$ is at least $2v/b(1+2d)$ for $1 \leq i \leq d$, and the overall d-knapsack constraint is still satisfied, then an element j is added to the corresponding candidate set. Then we establish the following theorem, which shows the property of the output of Algorithm 4. Its proof follows the same lines as the proof of Theorem 3.

Algorithm 4 d-KNAPSACK-STREAMING

$Q := \{[1+(1+2d)\varepsilon]^l | l \in \mathbb{Z}\}$.
for $v \in Q$
  $S_v := \emptyset$.
end for
$m := 0$.
for $j := 1$ **to** n
  for $i := 1$ **to** d
    $m := \max\{m, f(\{j\})/c_{i,j}\}$.
  end for
  $Q := \{[1+(1+2d)\epsilon]^l | l \in \mathbb{Z},$
  $\frac{m}{1+(1+2d)\epsilon} \leq [1+(1+2d)\epsilon]^l \leq 2bm\}$.
  for $v \in Q$
    if $c_{i,j} \geq \frac{b}{2}$ and $\frac{f(\{j\})}{c_{i,j}} \geq \frac{2v}{b(1+2d)}$ for some $i \in [1,d]$ **then**
      $S_v := \{j\}$.
      continue.
    end if
    if $\sum_{l \in S \cup \{j\}} c_{i,l} \leq b$ and $\frac{\Delta_f(j|S)}{c_{i,j}} \geq \frac{2v}{b(1+2d)}$ for all $i \in [1,d]$ **then**
      $S_v := S_v \cup \{j\}$.
    end if
  end for
end for
$S := \underset{S_v, v \in Q}{\operatorname{argmax}} f(S_v)$.
return S.

Theorem 4: Algorithm 4 has the following properties:

- *It outputs S that satisfies that* $f(S) \geq \left(\frac{1}{1+2d} - \epsilon\right)$ *OPT;*
- *It goes one pass over the dataset, and has* $O\left(\frac{n \log b}{\epsilon}\right)$ *computational complexity per element.*

C. ONLINE BOUND

To evaluate the performance of our proposed algorithms, we need to compare the function values obtained by our streaming algorithm against OPT, by calculating their relative difference. Since OPT is unknown, we could use an upper bound of OPT to evaluate the performance of the proposed algorithms.

By Theorem 4, we obtain

$$\mathrm{OPT} \leq \frac{1+2d}{1-(1+2d)\epsilon} f(S). \tag{5}$$

Then $\frac{1+2d}{1-(1+2d)\epsilon} f(S)$ is an upper bound of the optimal value to the d-MASK problem. In most of cases, this bound is not tight enough for the true optimal value. In the following, we provide a much tighter bound derived by the submodularity of f.

Theorem 5: Consider a subset $S \subseteq V$. For $1 \leq i \leq d$, let $r_{i,s} = \Delta_f(s|S)/c_{i,s}$, and $s_{i,1}, \ldots, s_{i,|V\setminus S|}$ be the sequence such that $r_{i,s_{i,1}} \geq r_{i,s_{i,2}} \geq \cdots \geq r_{i,s_{i,|V\setminus S|}}$. Let k_i be the integer such that $\sum_{j=1}^{k_i-1} c_{i,s_{i,j}} \leq b$ and $\sum_{j=1}^{k_i} c_{i,s_{i,j}} > b$. And let $\lambda_i = \left(b - \sum_{j=1}^{k_i-1} c_{i,s_{i,j}}\right) \Big/ c_{i,s_{i,k_i}}$. Then we have

$$OPT = \max_{C\mathbf{x}_{S'} \leq \mathbf{b}} f(S') \leq f(S) + \min_{1\leq i\leq d} \left[\sum_{j=1}^{k_i-1} \Delta_f(s_{i,j}|S) + \lambda_i \Delta_f(s_{i,k_i}|S)\right]. \tag{6}$$

Proof: Here we use a similar proof as the proof in [18, Th. 8.3.3] , where the author deals with the submodular maximization problem under one knapsack constraint. Let S^* be the optimal solution to Problem (1). First we consider the 1-MASK problem, which has the same objective function as Problem (1) but only with the i-th knapsack constraint. Assume S_i^* is its optimal solution. Since this 1-MASK problem has fewer constraints than Problem (1), we have $f(S^*) \leq f(S_i^*)$. Hence,

$$f(S^*) \leq \min_{1\leq i\leq d} f(S_i^*). \tag{7}$$

Since f is monotone submodular, for $1 \leq i \leq d$,

$$f(S_i^*) \leq f(S \cup S_i^*) \leq f(S) + \sum_{s\in S_i^*} \Delta_f(s|S). \tag{8}$$

We first assume that all weights $c_{i,j}$ and knapsack b are rational numbers. For the i-th 1-MASK problem, we can multiply all $c_{i,j}$ and b by the least common multiple of their denominators, making each weight and budget be an integer. We then replicate each element s in V into $c_{i,s}$ copies. Let s_i' denote any one copy of s, and let V_i' and $S_i^{*\prime}$ be the sets of the copies of all elements in V and S_i^*, respectively. Also, define $\Delta_f'(s_i'|S) \triangleq \Delta_f(s|S)/c_{i,s}$. Then

$$\sum_{s\in S_i^*} \Delta_f(s|S) = \sum_{s_i'\in S_i^{*\prime}} \Delta_f'(s_i'|S) \leq \max_{K'\subseteq V_i', |K'|\leq b} \sum_{s_i'\in K'} \Delta_f'(s_i'|S). \tag{9}$$

To find the value of the right-hand side of (9), we actually need to solve a unit-cost modular optimization problem as follows. We first sort all elements s' in V_i' such that the corresponding values $\Delta_f'(s'|S)$ form a non-increasing sequence. In this sequence, the first b elements are $c_{i,s_{i,j}}$ copies of $s_{i,j}$ for $1 \leq j \leq k_i - 1$, and $\left(b - \sum_{j=1}^{k_i-1} c_{i,s_{i,j}}\right)$ copies of s_{i,k_i}. Therefore, we obtain

$$\max_{\substack{K'\subseteq V_i' \\ |K'|\leq b}} \sum_{s_i'\in K'} \Delta_f'(s_i'|S) = \sum_{j=1}^{k_i-1} \Delta_f(s_{i,j}|S) + \lambda_i \Delta(s_{i,k_i}|S). \tag{10}$$

Combining (7), (8), (9) and (10), we obtain (6).

For irrational weights and knapsacks, let $\left\{c_{i,s_{i,j},t}\right\}_{t=1}^{\infty}$ and $\{b_t\}_{t=1}^{\infty}$ be two rational sequences with limits $c_{i,s_{i,j}}$ and b, respectively. And further let $k_{i,t}$ be the integer such that $\sum_{j=1}^{k_{i,t}-1} c_{i,s_{i,j},t} \leq b_t$ and $\sum_{j=1}^{k_{i,t}} c_{i,s_{i,j},t} > b_t$, and let

$$\lambda_{i,t} = \left(b_t - \sum_{j=1}^{k_{i,t}-1} c_{i,s_{i,j},t}\right) \Big/ c_{i,s_{i,k_{i,t}},t}.$$

Then $\left\{\lambda_{i,t}\right\}_{t=1}^{\infty}$ is a rational sequence with limit λ_i. According to the above argument, we obtain for each t,

$$\max_{C\mathbf{x}_{S'} \leq \mathbf{b}} f(S') \leq f(S) + \min_{1\leq i\leq d} \left[\sum_{j=1}^{k_i-1} \Delta_f(s_{i,j}|S) + \lambda_{i,t} \Delta_f(s_{i,k_i}|S)\right].$$

By letting t go to infinity, we then finish the proof. ■

A bound is called to be *offline* [18] if it can be stated before we run the algorithm; otherwise, it is an *online* one [18]. Here, we obtain an offline bound (5) and an online bound (6), the latter of which can be calculated by the following algorithm.

Algorithm 5 Online Bound of the d-MASK Problem

Input: S.
for $i := 1$ *to* d
  $S_i' := \emptyset$.
  for s **in** V
    $r_{i,s} := \Delta_f(s|S)/c_{i,s}$.
  end for
  while $\{s \in V \setminus (S \cup S_i') | \sum_{j\in S\cup S_i'\cup\{s\}} c_{i,j} \leq b\} \neq \emptyset$
    $s' := \operatorname{argmax}_{s\in V\setminus(S\cup S_i'), \sum_{j\in S\cup S_i'\cup\{s\}} c_{i,j}\leq b} r_{i,s}$.
    $S_i' := S_i' \cup \{s'\}$.
  end while
  $s' := \operatorname{argmax}_{s\in V\setminus(S\cup S_i'), \sum_{j\in S\cup S_i'\cup\{s\}} c_{i,j}\leq b} r_{i,s}$.
  $\lambda_i := (b - \sum_{s\in S_i'} c_{i,s})/c_{i,s'}$.
  $\delta_i := \sum_{s\in S_i'} \Delta_f(s|S) + \lambda_i \Delta_f(s'|S)$.
end for
return $f(S) + \min_{1\leq i\leq d} \delta_i$.

D. PROBLEMS WITH GROUND-SET DEPENDENT SUBMODULAR FUNCTIONS

In the previous sections, we have discussed the case when the submodular function f is independent of the ground set V. In the following, we will discuss the setting where f is additively decomposable [8], and the value of $f(S)$ depends on not only the subset S but also the ground set V. Here a function f is called to be *additively decomposable* [8] over the ground set V, if there exists a family of functions $\{f_i\}_{i=1}^{|V|}$ with $f_i : 2^V \to [0, \infty)$ independent of the ground set V such that

$$f(S) = \frac{1}{|V|} \sum_{i \in V} f_i(S). \tag{11}$$

Algorithm 4 is still useful for the case when f is dependent on the ground set but additively decomposable. To reduce the computational complexity, we randomly choose a small subset $\widetilde{V}$ of V, and use

$$f_{\widetilde{V}}(S) \triangleq \frac{1}{|\widetilde{V}|} \sum_{i \in \widetilde{V}} f_i(S)$$

instead of f in Algorithm 4. It can be proved that with a high probability, we can still obtain a good approximation to the optimal solution, when f_i's are bounded. The accuracy of the approximation is quantified by the following theorem.

Theorem 6: Assume that for $S \subseteq V$ and $1 \leq i \leq n$, $|f_i(S)| \leq 1$. We uniformly choose a subset $\widetilde{V}$ from V, with

$$|\widetilde{V}| \geq 2\epsilon^{-2} b^2 \left(b \log |V| + \log(2/\delta)\right),$$

and use $f_{\widetilde{V}}$ instead of f in Algorithm 4. Then with probability of at least $1 - \delta$, the output S of Algorithm 4 satisfies

$$f_{\widetilde{V}}(S) \geq \left(\frac{1}{1+2d} - \epsilon\right)(OPT - \epsilon).$$

Its proof follows the similar argument as the proof in [5, Th. 6.2], where the authors deal with the submodular maximization problem under one cardinality constraint. Now we adopt a two-pass streaming algorithm for the d-MASK problem with ground-set dependent submodular objective functions: in the first pass, we utilize reservoir sampling [19] to sample an evaluation set $\widetilde{V}$ randomly; in the second pass, we run Algorithm 4 with the objective function $f_{\widetilde{V}}$ instead of f.

IV. APPLICATIONS

In this section, we discuss two real-world applications for Algorithm 4: news recommendation and scientific literature recommendation. In the first application, we conduct experiments to show the efficiency and scalability of our proposed streaming algorithm, and we perform the sensitivity analysis with the second application.

A. NEWS RECOMMENDATION

Nowadays, people are facing many news articles on the daily basis, which highly stresses their limited reading time. A news recommendation system helps people quickly fetch the information they need, given their limited reading time. Specifically, it provides the most relevant and diversified news to people by exploiting their behaviors, considering their reading preferences, and learning from their previous reading histories.

However, the vast amount of news articles in the dataset are hard to be processed efficiently. Raman *et al.* [20] modeled the user behavior as a submodular maximization problem. Based on the learning result, a classical greedy algorithm [4] was implemented to provide a set of relevant articles to the users. However, the large amount of data in the dataset prevents the classical greedy algorithm from producing the solution in time due to its expensive computation cost. Besides, the reading behavior of the users was oversimplified in [20], where it is assumed that each user reads a fixed number of articles per day. Since the time spent on different news articles varies, it is more reasonable to use the number of words of the articles as the measure of the reading behavior. Hence, we can formulate this question into a 1-MASK problem (here $d = 1$) as follows:

$$\begin{aligned} &\underset{S \subseteq V}{\text{maximize}}\, f(S) = \mathbf{w}^T \mathbf{F}(S) \\ &\text{subject to} \sum_{j \in S} c_j \leq b, \end{aligned}$$

where c_j is the number of words in article j. Here $\mathbf{F} : 2^V \to [0, \infty)^m$, where m is the number of features. We require the total number of words in the selected articles not to exceed a specified budget b, due to the limitation of the user reading time. In addition, we assume that the non-negative parameter vector $\mathbf{w}$ is learnt by a statistical learning algorithm, based on the historical user preference (three such learning algorithms can be found in [20], [21], and [22], respectively). Let $(\phi_1(p), \ldots, \phi_m(p))$ be the characteristic vector of article p, where for $1 \leq j \leq m$, $\phi_j(p) = 1$ if p has feature j, $\phi_j(p) = 0$, otherwise. We then define $\mathbf{F}(S) = (F_1(S), \ldots, F_m(S))$; here for $1 \leq j \leq m$, $F_j(S)$ is the aggregation function of S with respect to feature j and defined by

$$F_j(S) \triangleq \log\left(1 + \sum_{s \in S} \phi_j(s)\right).$$

This choice of function F_j guarantees both precision and coverage of the solution set. On one hand, the monotonicity of $F_j(S)$ encourages feature j to be selected if its corresponding weighting parameter w_j (the j-th coordinate of the vector $\mathbf{w}$) is relatively large. On the other hand, the diminishing return property of F_j prevents too many items with feature j from being selected.

Notice that function F_j is a monotone submodular function. To see this, let

$$G_j(S) \triangleq \sum_{s \in S} \phi_j(s).$$

Obviously, $G_j(S)$ is a non-decreasing modular function. With the fact that $\zeta(x) \triangleq \log(1 + x)$ is an increasing concave function, we can conclude that $F_j(S) = \zeta(G_j(S))$ is

a monotone submodular function. Since both monotonicity and submodularity are closed under the non-negative linear combinations [23], f is a monotone submodular function as well. The solution based on Algorithm 4 to this 1-MASK problem provides the user a quick news recommendation.

As an illustration, we analyze the dataset collected in [24], which contains over 7, 000 feedback entries from 25 people with around 8, 000 news articles. In the dataset, news articles are classified into 480 types; and we use $m = 480$ type indicators as features in our experiments. We do manually set $b = 20$. Each entry of c is the average amount of time[1] every user spends, and normalized into $[1, 5]$. The learning algorithm proposed in [20] is used to calculate $\mathbf{w}$. We then compare Algorithm 4 with the greedy algorithm in [7].

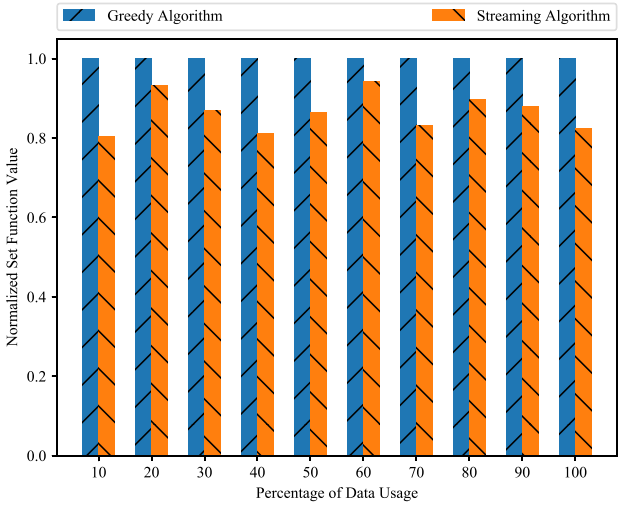


FIGURE 1. Comparison of Utilities between the Greedy and Streaming Algorithms.

In Fig. 1, we conduct 10 experiments on different portions of the dataset (*i.e.,* 10%, 20%, . . . , 100%) with the constraint fixed. For every experiment, we set the objective function value (utility), obtained by the classical greedy algorithm to be 1, by normalizing the objective value corresponding to our streaming algorithm. It has been shown that our streaming algorithm achieves over 80% the greedy algorithm utility in average. Thus, our proposed algorithm can provide competitive results for datasets with different sizes.

In Fig. 2, we compare the computation time between Algorithm 4 and the greedy algorithm. As the size of the dataset grows linearly, the computation time of the greedy algorithm increases exponentially, while our proposed streaming algorithm requires much less time. This shows that given a large dataset, the computation time for the greedy algorithm is much larger than that of our proposed streaming algorithm. When the size of the dataset is dramatically large,

[1]It is usually hard to know the average amount of time that each user spends on each article in the database, while the dataset from [24] provides this information. In a real-world application, we often use the number of words of the articles as the measure to quantify reading behaviors.

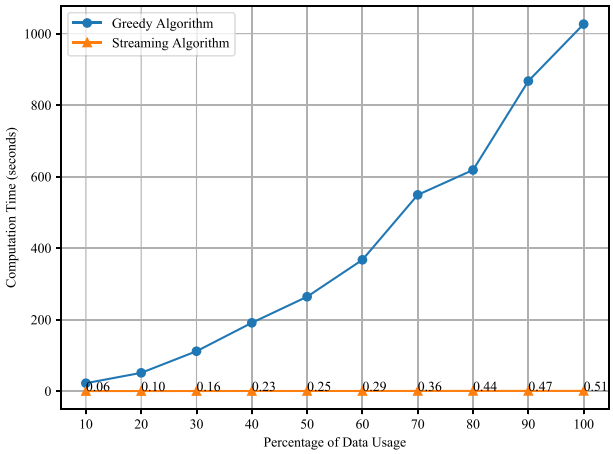


FIGURE 2. Comparison of Time Consumptions between the Greedy and Streaming Algorithms.

it is more suitable to apply our algorithm since its computation time increases only linearly with the size of the dataset.

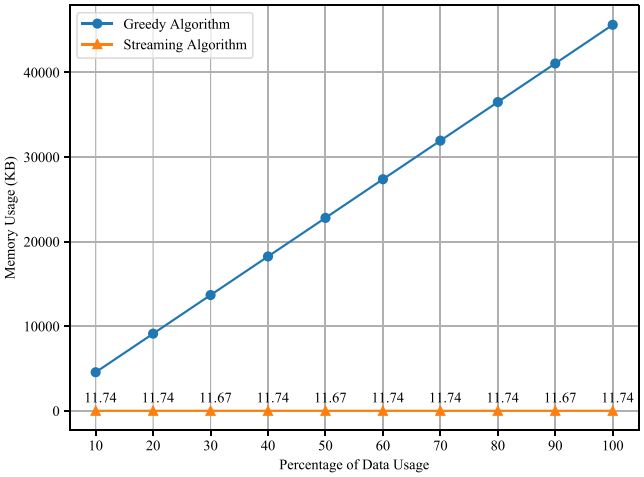


FIGURE 3. Comparison of Memory Costs between the Greedy and Streaming Algorithms.

In Fig. 3, we compare the memory usage of Algorithm 4 and the greedy algorithm. The latter one needs to load all data into the memory at the very beginning; thus its memory consumption grows linearly over the size of the dataset. On the other hand, the memory consumption of our proposed algorithm is independent of the size of the dataset, but dependent on the knapsack parameters (d and b) as shown in Theorem 4. This shows that our proposed algorithm only consumes a fixed amount of memory regardless of the data size, and therefore our proposed algorithm is more practically useful, especially for large datasets.

B. SCIENTIFIC LITERATURE RECOMMENDATION

Next, we introduce an application in scientific literature recommendation. Nowadays, the researchers have to face an enormous amount of articles and collect information that they

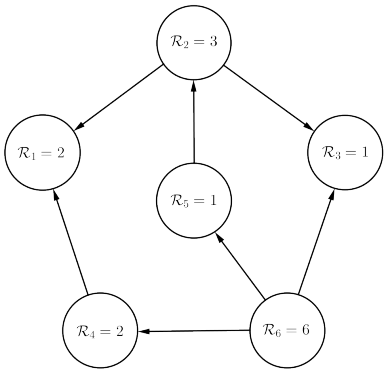


FIGURE 4. An example of a citation network.

are interested in, where they have to filter the massive existing scientific literatures and pick the most useful ones. A common approach to locate the targeted literatures is based on the so-called citation networks [25]. McNee *et al.* [25] mapped a citation network onto a rating matrix to filter research papers. In [26], an algorithm utilizing the random-walker properties was proposed. It transforms a citation matrix into a probability transition matrix and outputs the entries with the highest biased PageRank scores.

We here propose a new scientific literature recommendation system based on the citation networks and the newly proposed streaming algorithm (Algorithm 4). Consider a directed acyclic graph $G = (V, E)$ with $V = \{1, 2, \ldots, n\}$, where each vertex in V represents a scientific article. Let $\mathcal{R}_i$ denote the number of references contained in article i. The arcs between papers represent their citation relationship. For two vertices $i, j \in V$, arc $(i, j) \in E$ if and only if paper i cites paper j. The information spreads over the reverse directions of the arcs. As an example, Fig. 4 presents a citation network, which contains six vertices and seven arcs. Each of six papers cites a certain number of references.

The information initiates from a set of vertices (source papers), and then spreads across the network. Let A be the collection of the source papers. Our target is to select a subset S out of V to quickly detect the information spreading of A. For example, $A = \{1, 3, 4\}$ in Fig. 4. If we choose $S = \{6\}$, we can detect the source papers 1, 3, 4 by paths $6 \to 4 \to 1$, $6 \to 3$ and $6 \to 4$, respectively. This problem can be formulated as a monotone submodular maximization under a 3-knapsack constraint[2]:

$$\begin{aligned} &\underset{S \subseteq V}{\text{maximize}}\ R(S) \\ &\text{subject to } C\mathbf{x}_S \leq \mathbf{b}, \end{aligned} \tag{12}$$

where $C = (c_{i,j})$ is a $3 \times n$ matrix and $\mathbf{b} = (b_1, b_2, b_3)^T$.

Observe that the papers in A transfer their influence through the citation network, but this influence becomes less as it spreads through more hops. Let $T(s, a)$ be the length of the shortest directed path from s to a. Then the shortest path length from any vertex in S to a is defined as

$$T(S, a) \triangleq \min_{s \in S} T(s, a).$$

Let $W(a)$ be a pre-assigned weight to each vertex $a \in A$ such that $\sum_{a \in A} W(a) = 1$. Then our goal is to minimize the expected penalty

$$\pi(S) \triangleq \sum_{a \in A} W(a) \min\{T(S, a), T_{\max}\},$$

or maximize the expected penalty reduction

$$R(S) \triangleq T_{\max} - \pi(S) = \sum_{a \in A} W(a)[T_{\max} - T(S, a)]^+,$$

where $[x]^+ \triangleq \max\{x, 0\}$ and $T_{\max}$ is a given maximum penalty. Note that R is a monotone submodular function. To see this, for two subsets $B \subseteq C \subseteq V$, we have $T(B, a) \geq T(C, a)$ for any $a \in A$, such that $R(B) \leq R(C)$; $T_{\max} - T(S, a)$ is a submodular function with respect to S since

$$\begin{aligned} &T(B, a) - T(B \cup \{v\}, a) \\ &\quad = [T(B, a) - T(v, a)]^+ \\ &\quad \geq [T(C, a) - T(v, a)]^+ = T(C, a) - T(C \cup \{v\}, a), \end{aligned}$$

with $v \in V \setminus C$. Then $R(S)$ is also submodular, since it is a convex combination of $T_{\max} - T(S, a)$ for $a \in A$.

We construct three constraints in (12) from the aspects of the recency, the biased PageRank score, and reference number respectively. The first aspect is from the fact that readers prefer to read the recently published papers. Let $c_{1,j}$ be the time difference between the publishing date of paper j and the current date, and b_1 be the corresponding limit.

For the second aspect, the classical PageRank algorithm [27] could be used to compute an important score for every vertex in the graph: a vertex will be assigned a higher score if it is connected to a more important vertex with a lower out-degree. Gori and Pucci [26] introduced a so-called biased PageRank score. It is a measure of the significance of each paper, not only involving the propagation and attenuation properties of the network, but also taking the set of source vertices into account. Let $\rho(j)$ be the biased PageRank score of article j. We further choose a function $\xi(x) \triangleq 1 + \frac{1}{1+x}$ to map the PageRank score onto $(1, 2]$. Then paper j with the smaller value $c_{2,j} \triangleq \xi(\rho(j))$ is more valuable for the researchers. Also we set b_2 to be corresponding budget.

Thirdly, we assume that more references listed in the paper, more time the reader spends on picking the valuable information. Then we set $c_{3,j}$ to be the number $\mathcal{R}_j$ of references in paper j and b_3 be the budget of the total number of references.

To evaluate the performance of Algorithm 4, for scientific literature recommendation, we utilize a dataset collected in [28]. This dataset includes more than 20,000 papers in the Association of Computational Linguistics (ACL). There are two methods to evaluate the performance of an algorithm for literature recommendation: online evaluation and offline evaluation. In the online evaluation, some volunteers are

[2]The reason why we set $d = 3$ will be explained later in this section; the number of knapsack constraints and the corresponding budgets can be changed accordingly based on different needs of the applications.

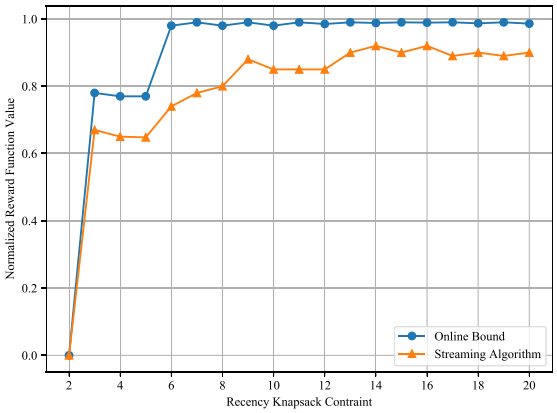


FIGURE 5. Optimal Function Values corresponding to Different Recency Constraints.

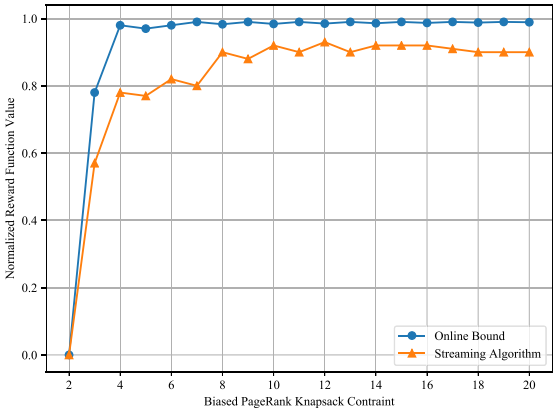


FIGURE 6. Optimal Function Values corresponding to Different Biased PageRank Constraints.

invited to test the performance of the recommendation system and express their opinions. Here we use the offline evaluation to compare the function values obtained by our proposed algorithm (Algorithm 4) and the PageRank algorithm proposed in [26].

We perform the sensitivity analysis over different knapsack constraints. With the other two constraints fixed, we change the value of the budget corresponding to the recency, biased PageRank score or reference number, respectively. Here we randomly select five nodes as the source papers. We set $T_{\max} = 50$ and $W(a) = 0.2$ for each source paper a. The results for the optimal objective values are shown in Fig. 5 (with fixed $b_2 = 10$, $b_3 = 20$), Fig. 6 (with fixed $b_1 = 20$, $b_3 = 20$) and Fig. 7 (with fixed $b_1 = 20$, $b_2 = 10$), respectively. Note that parameters b_1, b_2 and b_3 are chosen to satisfy certain profiles of the users in our scientific literature recommendation system. Empirically, researchers are expected to read the papers published within the past 5 years, and we assume each user wants to see around 4 relevant papers from end past year. We, therefore, set $b_1 = 20$. Following the similar reason, we set $b_2 = 10$ and $b_3 = 20$. In practice, we may customize these parameters based on the system needs.

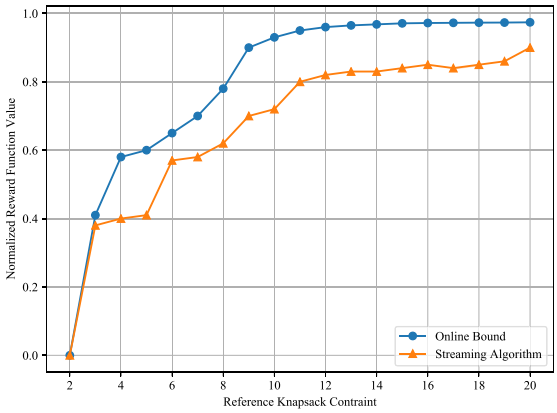


FIGURE 7. Optimal Function Values corresponding to Different Reference Knapsack Constraints.

It can be observed that the relative difference is around 10% between the function values obtained by our streaming algorithm (orange lines) and the corresponding online bounds (blue lines).

Based on the experimental results shown in the applications of news and scientific literature recommendation systems, it is clear that our proposed streaming algorithm is able to provide an approximated solution efficiently, especially over large-scale datasets, where the utility of the solution provided by our proposed algorithm is close to the online bound and that provided by the conventional greedy algorithm. Therefore, our proposed algorithm is more practically useful for handling the large datasets.

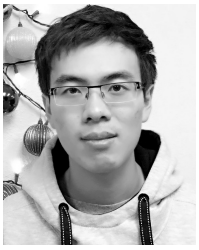
V. CONCLUSIONS

In this paper, a single-pass streaming algorithm was proposed with a $\left(\frac{1}{1+2d} - \epsilon\right)$ approximation factor to solve the monotone submodular maximization problem with a d-knapsack constraint. By conducting experiments on two applications including news and scientific literature recommendations, it can be observed that our proposed algorithm can achieve a similar performance to the classical greedy algorithm, while the computation time is 10,000 times faster under limited memory usage.

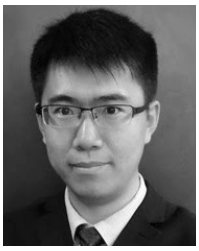
REFERENCES

[1] H. Lin and J. Bilmes, "How to select a good training-data subset for transcription: Submodular active selection for sequences," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Brighton, U.K., Sep. 2009, pp. 1–5.
[2] Y. Liu, K. Wei, K. Kirchhoff, Y. Song, and J. Bilmes, "Submodular feature selection for high-dimensional acoustic score spaces," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2013, pp. 7184–7188.
[3] S. Chakraborty, O. Tickoo, and R. Iyer, "Adaptive keyframe selection for video summarization," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2015, pp. 702–709.
[4] G. Nemhauser, L. Wolsey, and M. Fisher, "An analysis of approximations for maximizing submodular set functions—I," *Math. Programm.*, vol. 14, no. 1, pp. 265–294, 1978.
[5] A. Badanidiyuru, B. Mirzasoleiman, A. Karbasi, and A. Krause, "Streaming submodular maximization: Massive data summarization on the fly," in *Proc. 20th ACM Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2014, pp. 671–680.

[6] A. Kulik, H. Shachnai, and T. Tamir, "Maximizing submodular set functions subject to multiple linear constraints," in *Proc. 20th Annu. ACM-SIAM Symp. Discrete Algorithms*, New York, NY, USA, Jan. 2009, pp. 545–554.
[7] M. Sviridenko, "A note on maximizing a submodular set function subject to a knapsack constraint," *Oper. Res. Lett.*, vol. 32, no. 1, pp. 41–43, 2004.
[8] B. Mirzasoleiman, A. Karbasi, R. Sarkar, and A. Krause, "Distributed submodular maximization: Identifying representative elements in massive data," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2013, pp. 2049–2057.
[9] A. Badanidiyuru and J. Vondrák, "Fast algorithms for maximizing submodular functions," in *Proc. 25th Annu. ACM-SIAM Symp. Discrete Algorithms*, Portland, OR, USA, Oct. 2014, pp. 1497–1514.
[10] B. Mirzasoleiman, A. Badanidiyuru, A. Karbasi, J. Vondrák, and A. Krause, "Lazier than lazy greedy," in *Proc. 29th AAAI Conf. Artif. Intell.*, Austin, TX, USA, Jan. 2015, pp. 1812–1818.
[11] H. Lin and J. Bilmes, "Multi-document summarization via budgeted maximization of submodular functions," in *Proc. NAACL HLT*, Los Angeles, CA, USA, Jun. 2010, pp. 912–920.
[12] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, San Jose, CA, USA, Aug. 2007, pp. 420–429.
[13] Q. Yu, H. Li, Y. Liao, and S. Cui. (2017). "Fast budgeted influence maximization over multi-action event logs." [Online]. Available: https://arxiv.org/abs/1710.02141
[14] G. Papachristoudis, "Theoretical guarantees and complexity reduction in information planning," Ph.D. dissertation, Dept. Elect. Eng. Comput. Sci., MIT, Cambridge, MA, USA, Jun. 2015.
[15] R. Kumar, B. Moseley, S. Vassilvitskii, and A. Vattani, "Fast greedy algorithms in mapreduce and streaming," in *Proc. 25th Annu. ACM Symp. Parallelism Algor. Archit.*, Montreal, QC, Canada, Jun. 2013, pp. 1–10.
[16] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," in *Proc. 6th USENIX Symp. Oper. Syst. Des. Implement.*, San Francisco, CA, USA, Dec. 2004, pp. 137–150.
[17] R. Kiveris, S. Lattanzi, V. Mirrokni, V. Rastogi, and S. Vassilvitskii, "Connected components in MapReduce and beyond," in *Proc. ACM Symp. Cloud Comput.*, Seattle, WA, USA, Jun. 2014, pp. 1–13.
[18] J. Leskovec, "Dynamics of large networks," Ph.D. dissertation, School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, USA, Sep. 2008.
[19] J. S. Vitter, "Random sampling with a reservoir," *ACM Trans. Math. Softw.*, vol. 11, no. 1, pp. 37–57, 1985.
[20] K. Raman, P. Shivaswamy, and T. Joachims, "Online learning to diversify from implicit feedback," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Beijing, China, Aug. 2012, pp. 705–713.
[21] L. Li, D. Wang, T. Li, D. Knox, and B. Padmanabhan, "SCENE: A scalable two-stage personalized news recommendation system," in *Proc. 34th Annu. Int. ACM SIGIR Conf.*, Beijing, China, Jul. 2011, pp. 125–134.
[22] W. IJntema, F. Goossen, F. Frasincar, and F. Hogenboom, "Ontology-based news recommendation," in *Proc. EDBT/ICDT Workshops*, Lausanne, Switzerland, Mar. 2010, pp. 1–6.
[23] S. Fujishige, *Submodular Functions and Optimization*, 2nd ed. Amsterdam, The Netherlands: Elsevier, 2005.
[24] S. R. Wolfe and Y. Zhang, "Interaction and personalization of criteria in recommender systems," in *Proc. Int. Conf. User Modeling, Adaptation, Personalization*. Berlin, Germany: Springer, Jun. 2010, pp. 183–194.
[25] S. M. McNee *et al.*, "On the recommending of citations for research papers," in *Proc. ACM Conf. Comput. Supported Cooperat. Work (CSCW)*, New Orleans, LA, USA, Nov. 2002, pp. 116–125.
[26] M. Gori and A. Pucci, "Research paper recommender systems: A random-walk based approach," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell.*, Dec. 2006, pp. 778–781.
[27] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the Web," Stanford Univ., Stanford, CA, USA, Tech. Rep. 68, Jan. 1998.
[28] M. T. Joseph and D. R. Radev, "Citation analysis, centrality, and the ACL anthology," Univ. Michigan, Anna Arbor, MI, USA, Tech. Rep. CSE-TR-535-07, Oct. 2007.



QILIAN YU (S'16) received the B.S. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2014. He is currently pursuing the Ph.D. degree with the University of California at Davis, Davis, CA, USA. His current research interests focus on data processing and learning over large graphs, social network analysis, submodular optimization, and streaming algorithm.



LI XU received the B.S. degrees in mathematics and computer science from Peking University, Beijing, China, and Ph.D. degree in mathematics from The University of Hong Kong, Hong Kong. He was a Post-Doctoral Research Associate with the Department of Electrical and Computer Engineering, Texas A&M University. He was also a Post-Doctoral Fellow with the Department of Electrical Engineering and Computer Science, University of Michigan, and also with the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen. He is currently an Associate Professor with the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include statistical learning, coding theory, and data analytics. He was a TPC member of the IEEE GLOBECOM 2016, 2017, and 2018, the IEEE GlobalSIP 2016 and 2017, the IEEE WCNC 2017 and 2018, the IEEE 5GWF 2018, and the IEEE COMNETSAT 2018. He was the Local Arrangement Co-Chair of MIIS 2016. He was the Webmasters for WCI 2013 and CAM 2016. He was an Exemplary Reviewer of the IEEE COMMUNICATIONS LETTERS in 2014.

SHUGUANG CUI (S'99–M'05-SM'12–F'14) received the Ph.D. degree in electrical engineering from Stanford University, CA, USA, in 2005. He received the Amazon AWS Machine Learning Award in 2018. He is currently an Assistant Professor, Associate Professor, and a Full Professor in electrical and computer engineering with The University of Arizona and Texas A&M University. He is also the Child Family Endowed Chair Professor in electrical and computer engineering with the University of California at Davis, Davis. His current research interests focus on data driven large-scale system control and resource management, large data set analysis, IoT system design, energy harvesting-based communication system design, and cognitive network optimization. He was an Elected Member of the SPCOM Technical Committee, IEEE Signal Processing Society (2009–2014). He was a recipient of the Best Paper Award from the IEEE Signal Processing Society 2012. He was selected as the Thomson Reuters Highly Cited Researcher and listed in the Worlds' Most Influential Scientific Minds by ScienceWatch in 2014. He received the Amazon AWS Machine Learning Award in 2018.He was the general co-chair and the TPC co-chair of many IEEE conferences. He was the Elected Chair of the IEEE ComSoc Wireless Technical Committee (2017–2018). He is also an Area Editor of the *IEEE Signal Processing Magazine*, and an Associate Editor of the IEEE TRANSACTIONS ON BIG DATA, the IEEE TRANSACTIONS ON SIGNAL PROCESSING, the IEEE JSAC Series on Green Communications and Networking, and the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. He is a member of the Steering Committee for both the IEEE TRANSACTIONS ON BIG DATA and the IEEE TRANSACTIONS ON COGNITIVE COMMUNICATIONS AND NETWORKING. He is also a member of the IEEE ComSoc Emerging Technology Committee. He was an IEEE ComSoc Distinguished Lecturer in 2014.