A Stab at Approximating Minimum Subadditive Join

To be published in *Algorithms and Data Structures: 10th International Workshop, WADS 2007, Halifax, Canada, August 15-17, 2007 Proceedings.* LNCS, Springer Verlag.

Staal A. Vinterbo

Decision Systems Group, Brigham and Women's Hospital, Boston
Harvard Medical School
Harvard-MIT Division of Health Sciences and Technology

staal@dsg.harvard.edu

Abstract

Let $(L, *)$ be a semilattice, and let $c : L \to [0, \infty)$ be monotone and increasing on L. We state the Minimum Join problem as: given size n sub-collection X of L and integer k with $1 \leq k \leq n$, find a size k sub-collection $(x'_1, x'_2, \ldots, x'_k)$ of X that minimizes $c(x'_1 * x'_2 * \cdots * x'_k)$. If $c(a * b) \leq c(a) + c(b)$ holds, we call this the Minimum Subadditive Join (MSJ) problem and present a greedy $(k - p + 1)$-approximation algorithm requiring $O((k - p)n + n^p)$ joins for constant integer $0 < p \leq k$. We show that the MSJ Minimum Coverage problem of selecting k out of n finite sets such that their union is minimal is essentially as hard to approximate as the Maximum Balanced Complete Bipartite Subgraph (MBCBS) problem. The motivating by-product of the above is that the privacy in databases related k-ambiguity problem over L with subadditive information loss can be approximated within $k - p$, and that the k-ambiguity problem is essentially at least as hard to approximate as MBCBS.

---

1 Introduction

In this paper we will often talk about finite collections of elements from some set. What we mean by a collection is an ordered multiset. Let $I_n = \{1, 2, \ldots, n\}$, and let U be a set. We can then represent a collection of size n of elements from U as a function $\mathcal{X} : I_n \to U$. We will denote the set of all collections of elements from U of size n as $\mathcal{C}_n(U)$. We can now formally define the Minimum Subadditive Join[1] (MSJ) problem as follows.

Problem 1 (Minimum Subadditive Join) *Let $(L, *)$ be a semilattice where $*$ is polynomial time computable, and let $c : S \to [0, \infty)$ be a monotone, increasing and polynomial time computable function on L such that*

$$c(a * b) \leq c(a) + c(b) \tag{1}$$

holds for $a, b \in L$. Given $\mathcal{X}_n \in \mathcal{C}_n(L)$ an integer k with $1 \leq k \leq n$, find a subset $S = \{s_1, s_2, \ldots, s_k\} \subset I_n$ that minimizes $c(\mathcal{X}_n(s_1) * \mathcal{X}_n(s_2) * \cdots * \mathcal{X}_n(s_k))$. We denote an instance of this problem as $((L, *), \mathcal{X}_n, c, k)$.

[1] We could equally well have chosen to call the operation $*$ "meet", but motivated by the behavior of cardinality over $(2^X, \cup)$, we chose "join".

If (1) is not required to hold, we call the problem the Minimum Join (MJ) problem.

Our study of the Minimum Join problem is principally motivated by its relation to privacy in databases. Biomedical research is dependent on sharing of data [1, 2]. Two immediate reasons for this are the principle of reproducibility of research, and the need of comparative retrospective data analysis. However, we are ethically [3], and legally [4] bound to protect the privacy of individuals, and the consequences of loss of trust in the preservation of privacy can be dire [5].

With the advent of large collections of data relatively easily available in electronic form on the web, privacy is endangered through the potential ability of linking a combination of individually seemingly "safe" data items across tables.

Consider the following data matrix in Table 1. Let each row represent what we know about a particular individual. We want to release the data contained in Table 1 without endangering the

Table 1: Example data set.

	a	b	c	d
1	1	0	1	1
2	0	1	1	1
3	0	0	0	1
4	0	1	1	0

privacy rights of the individuals in question. As we don't know what a potential adversary is capable of, we postulate the existence of a "linking machine" ϕ that when given a row in Table 1, yields the identity of the individual. Introducing ambiguity in the data is a countermeasure [6, 7, 8, 9] that can be applied to reduce the applicability of ϕ. A way to introduce this ambiguity is to define the value $\top$ to be indistinguishable from both 0 and 1, in effect representing both of these values. A row is then made indistinguishable from others by substituting values with $\top$. Consider row 1. It differs from row 2 in attributes a and b. We collect the sets of attributes in which row 1 differs from all the others it in a collection $C_1 = (\{a, b\}, \{a, c\}, \{a, b, d\})$. If we need to make row 1 indistinguishable from k rows, we can do this by selecting $k - 1$ elements from C_1 and substituting $\top$ for the values in row 1 identified by the union of the selected elements. Say $k = 3$, we can then choose the sets $\{a, b\}$ and $\{a, b, d\}$ corresponding to rows 2 and 4, respectively. The result for row 1 is $(\top\top1\top)$.

The above is an example of k-ambiguity [9] by cell suppression. A particular flavor of k-ambiguity is k-anonymity [6], in that k-anonymity requires that every record in the transformed data is equivalent to k records in the *transformed* data. The problem of introducing k-anonymity by a minimum number of cell suppressions was shown to be approximable within $O(k \log k)$ [10], and subsequently a more general version of the k-anonymity problem was shown to be approximable within $\max\{2k - 1, 3k - 5\}$ [11]. We will in the following

- present a polynomial time $(k - p + 1)$-approximation algorithm for the MSJ problem and show that this algorithm fails to provide any bound for non-subadditive instances of MJ, and
- prove that a specialization Minimum Coverage (MinC) of the MSJ problem is essentially as hard to approximate as the known NP-hard Maximum Balanced Complete Bipartite Subgraph (MBCBS) problem,
- use the connection between MJ and k-ambiguity to present a $(k-p)$-approximation algorithm for subadditive information loss instances of the k-ambiguity problem. We also show that negative approximation results obtained for MSJ are applicable to the k-ambiguity problem.

2 A Polynomial Time Approximation Algorithm

The algorithm GREEDY-MSJ in Algorithm 1 is a simple greedy algorithm that essentially works by iteratively adding the unused element that minimizes the added cost to the solution. We also include that by pre-computing an optimal solution for $p \leq k$, we can improve the upper approximation bound. Let $V^* = \mathcal{X}_n(v_1) * \mathcal{X}_n(v_2) * \cdots * \mathcal{X}_n(v_m)$ for any $V = \{v_1, v_2, \ldots, v_m\} \subseteq I_n$.

Algorithm 1: The greedy $(k - p + 1)$-approximation algorithm.

GREEDY-MSJ$(\mathcal{X}_n, c, k, p)$
$C \leftarrow I_n$
$S \leftarrow \emptyset$
$(i_1, i_2, \ldots, i_p) \leftarrow \arg\min_{1 \leq j_1 < j_2 < \cdots < j_p \leq n} c(\{j_1, j_2, \ldots, j_p\}^*)$
$S \leftarrow \{i_1, i_2, \ldots, i_p\}$
$z \leftarrow c(S^*)$
$C \leftarrow C - S$
while $k > p$
    $i \leftarrow \arg\min_{j \in C} c(z * \mathcal{X}_n(j))$
    $S \leftarrow S \cup \{i\}$
    $z \leftarrow z * \mathcal{X}_n(i)$
    $C \leftarrow C - \{i\}$
    $k \leftarrow k - 1$
return S

Theorem 1 *For constant integer $0 < p \leq k$ we have that* GREEDY-MSJ *is a polynomial time $(k - p + 1)$-approximation algorithm for the Minimum Subadditive Join problem that can be implemented to run in $O(((k-p)n + n^p)t(*))$ time, where $t(*)$ is the time it takes to compute $*$.*

Proof. Let

- $S(i) = \{s_1, s_2, \ldots, s_i\}$ be the solution returned by GREEDY-MSJ$(\mathcal{X}_n, c, i, p)$, with s_j being added to the solution before s_{j+1} for all $j > p$, and
- $O(i) = \{{}_i o_1, {}_i o_2, \ldots, {}_i o_i\}$, be an optimal solution for instance $(\mathcal{X}_n, c, i)$, and let $O(i)$ be ordered such that any elements in $O(i)$ occurring in $S(i)$ come first and in the same order as in $S(i)$. This ensures that
$$ {}_i o_i \notin S(i-1) \tag{2}$$
holds.

First of all, note that by design of the algorithm, we have that

$$c(O(p)^*) = c(S(p)^*) \ . \tag{3}$$

Next we note that because of monotonicity of c and (1) we have that for any $i \geq p$

$$c(O(i)^*) = 0 \Rightarrow c(S(i)^*) = 0 \ , \tag{4}$$

and GREEDY-MSJ produces an optimal solution in this case. Therefore now assume that $c(O(i)^*) > 0$. By the greediness of GREEDY-MSJ, (2), and (1) we also have that that for $i \geq p$

$$\begin{aligned} c(S(i+1)^*) &= c(S(i)^* * \mathcal{X}_n(s_{i+1})) \\ &\leq c(S(i)^* * \mathcal{X}_n({}_{i+1}o_{i+1})) \\ &\leq c(S(i)^*) + c(\mathcal{X}_n({}_{i+1}o_{i+1})) \end{aligned} \tag{5}$$

holds. By (3) we get

$$c(S(p)^*) + c(\mathcal{X}_n({}_{p+1}o_{p+1})) = c(O(p)^*) + c(\mathcal{X}_n({}_{p+1}o_{p+1})) \ . \tag{6}$$

Further, we have that $c(O(i)^*) \geq c(\{{}_io_1, {}_io_2, \ldots, {}_io_{i-1}\}^*) \geq c(O(i-1)^*)$ by monotonicity of c and optimality of $O(i-1)$. This, combined with (5) and (6), means that

$$\begin{aligned} \frac{c(S(p+1)^*)}{c(O(p+1)^*)} &\leq \frac{c(O(p)^*) + c(\mathcal{X}_n({}_{p+1}o_{p+1}))}{c(O(p+1)^*)} \\ &= \frac{c(O(p)^*)}{c(O(p+1)^*)} + \frac{c(\mathcal{X}_n({}_{p+1}o_{p+1}))}{c(O(p+1)^*)} \leq 1 + 1 \leq 2 \ . \end{aligned} \tag{7}$$

Furthermore, using (5) and $c(O(i)^*) \geq c(O(i-1)^*)$, we get that for $i > p$

$$\begin{aligned} \frac{c(S(i)^*)}{c(O(i)^*)} &\leq \frac{c(S(i-1)^*) + c(\mathcal{X}_n({}_io_i))}{c(O(i)^*)} \\ &= \frac{c(S(i-1)^*)}{c(O(i)^*)} + \frac{c(\mathcal{X}_n({}_io_i))}{c(O(i)^*)} \\ &\leq \frac{c(S(i-1)^*)}{c(O(i-1)^*)} + \frac{c(\mathcal{X}_n({}_io_i))}{c(O(i)^*)} \end{aligned}$$

holds. By monotonicity of c we have that $c(\mathcal{X}_n({}_io_i)) \leq c(O(i)^*)$, by which we get

$$\frac{c(S(i)^*)}{c(O(i)^*)} \leq \frac{c(S(i-1)^*)}{c(O(i-1)^*)} + 1 \ . \tag{8}$$

Using (7) and induction on (8), we get

$$\frac{c(S(k)^*)}{c(O(k)^*)} \leq k - p + 1 \ . \tag{9}$$

The computation of an optimal solution for $i = p$ in line 3 can be implemented as p nested loops over C, which yields the $O(n^p t(*))$ term. The subsequent steps are $O((k-p)nt(*))$, making the algorithm implementable in $O(((k-p)n + n^p)t(*))$ time. ■

We now investigate the non-subadditive case. We will in the following show that for any $\beta > 1$, we can construct a MJ instance such that GREEDY-MSJ produces a solution worse than β times the optimal.

Proposition 1 GREEDY-MSJ *fails to approximate MJ within any given bound* $\beta > 1$.

Proof. Let $X = \{s_1, s_2, \ldots, s_k, o_1, o_2, \ldots, o_k\} = L$, and let $*$ be defined such that the Hasse diagram of $\leq$ is Figure 1. Further, let

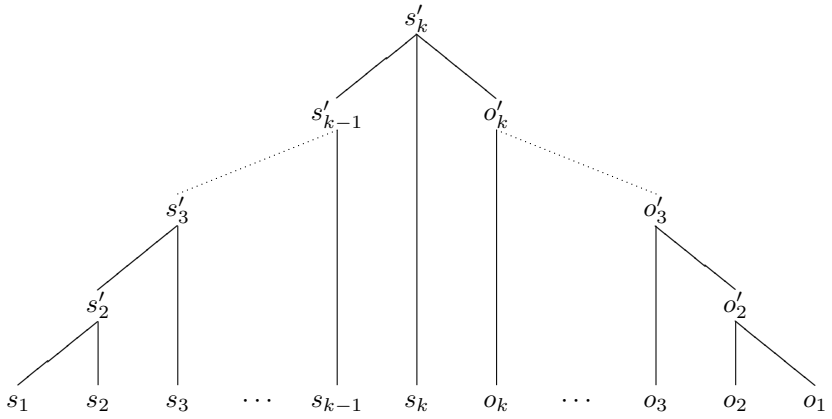

Figure 1: Example of GREEDY-MSJ failure.

- $c(s_1) = \frac{1}{3k}$ and $c(s_i) = \frac{1}{2k}$ for $2 \leq i \leq k$,
- $c(o_i) = \frac{1}{k}$ for $1 \leq i \leq k$,
- $c(s'_i) = \sum_{j=1}^{i} c(s_j)$ for $1 \leq i < k$,
- $c(o'_i) = \sum_{j=1}^{i} c(o_j)$ for $1 \leq i \leq k$, and
- $c(s'_k) = (\beta + \epsilon)$,

for some $\epsilon > 0$ and $\beta > 1$. Let $\mathcal{X}_n$ represent X, then we have that

$$\frac{c(\text{GREEDY-MSJ}(\mathcal{X}_n, c, i, 1)^*)}{c(O_i)} = 1, \text{ for } 1 \leq i < k, \text{ and}$$
$$\frac{c(\text{GREEDY-MSJ}(\mathcal{X}_n, c, k, j)^*)}{c(O_k)} > \beta$$

for $j < k$ if we let O_i be an optimal solution for $k = i$. ■

3 Minimum Coverage

Consider the problem of selecting k out of n finite subsets of some set such that their union is minimal. It is clear that this problem can be formulated as follows.

Problem 2 (Minimum Coverage) *Let U be a finite set, and let $\mathcal{X}_n \in \mathcal{C}_n(2^U)$. Find $S = \{s_1, s_2, \ldots, s_k\} \subset I_n$ that minimizes $|\mathcal{X}_n(s_1) \cup \mathcal{X}_n(s_2) \cup \cdots \cup \mathcal{X}_n(s_k))|$. We denote an instance of this problem as $(\mathcal{X}_n, k)$.*

We call the above Problem 2 Minimum Coverage (MinC) since the problem of *maximizing* $|\mathcal{X}_n(s_1) \cup \mathcal{X}_n(s_2) \cup \cdots \cup \mathcal{X}_n(s_k))|$ in Problem 2 is known as the Maximum Coverage problem [12].

It is also clear that since $(2^U, \cup)$ is a semi-lattice, that $|X \cup Y| \geq |X|$ and $|X \cup Y| \leq |X| + |Y|$ for all $X, Y \in 2^U$, and that both $\cup$ and $||$ are computable in polynomial time, we have that a Minimum Coverage instance $(\mathcal{X}_n, k)$ is the MSJ instance $((2^U, \cup), \mathcal{X}_n, c, k)$ where $c(X) = |X|$. We formulate the above as the following proposition.

Proposition 2 *The Minimum Coverage instance $(\mathcal{X}_n, k)$ is the Minimum Subadditive Join instance $((2^U, \cup), \mathcal{X}_n, c, k)$ where $c(X) = |X|$.*

Proposition 3 *Let $(L, \vee, \wedge, \bar{}, 0, 1)$ be a finite Boolean algebra such that $\vee$, $\wedge$, and $\bar{}$ all are computable in polynomial time. Let polynomial time computable function $g : L \to [0, \infty)$ be such that $g(x) = g(1) - g(\overline{x})$ for all $x \in L$, and let $h(x) = g(1) - g(x)$. Also, let $\overline{\mathcal{X}_n}$ be such that $\overline{\mathcal{X}_n}(i) = \overline{\mathcal{X}_n(i)}$. Then MSJ instances $((L, \vee), \mathcal{X}_n, g, k)$ and $((L, \wedge), \overline{\mathcal{X}_n}, h, k)$ are polynomial time equivalent.*

Proof. We start by noting that by applying DeMorgan's laws we have that

$$\mathcal{X}_n(i) \vee \mathcal{X}_n(j) = \overline{\overline{\mathcal{X}_n(i)} \wedge \overline{\mathcal{X}_n(j)}} = \overline{\overline{\mathcal{X}_n}(i) \wedge \overline{\mathcal{X}_n}(j)} \ .$$

Using this we have that

$$g(\mathcal{X}_n(i) \vee \mathcal{X}_n(j)) = g(\overline{\overline{\mathcal{X}_n}(i) \wedge \overline{\mathcal{X}_n}(j)}) = g(1) - g(\overline{\mathcal{X}_n}(i) \wedge \overline{\mathcal{X}_n}(j)) = h(\overline{\mathcal{X}_n}(i) \wedge \overline{\mathcal{X}_n}(j)) \ .$$

We also have that

$$g(x) = g(1) - g(\overline{x}) = h(\overline{x}) \ .$$

The proof then follows from the polynomial time computability of the Boolean algebra operators. ■

Now consider the problem of finding k out of n finite subsets of some set such that their intersection is maximal. This problem we call the Maximum Intersection (MaxI) problem and define it formally as follows.

Problem 3 (Maximum Intersection) *Let U be a finite set, let and let $\mathcal{X}_n \in \mathcal{C}_n(2^U)$. Find $S = \{s_1, s_2, \ldots, s_k\} \subset I_n$ that maximizes $|\mathcal{X}_n(s_1) \cap \mathcal{X}_n(s_2) \cap \cdots \cap \mathcal{X}_n(s_k))|$. We denote an instance of this problem as $(\mathcal{X}_n, k)$.*

Proposition 4 *Maximum Intersection and Minimum Coverage are polynomial time equivalent.*

Proof. We begin by noting that for a finite set U we have that $(2^U, \cup, \cap, \bar{}, \emptyset, U)$ is a Boolean algebra with polynomial time computable operations. Let $g(X) = |X|$, we then have that $g(X) = g(U) - g(\overline{X})$, and that $g(X \cup Y) \geq g(X)$ and $g(X \cup Y) \leq g(X) + g(Y)$ for all $X, Y \in 2^U$. Let $h(X) = g(U) - g(X)$. Then by Proposition 3 we have that MSJ instances $((2^U, \cup), \mathcal{X}_n, g, k)$ and $((2^U, \cap), \overline{\mathcal{X}_n}, h, k)$ are polynomially equivalent. We finish the proof by recognizing that $((2^U, \cup), \mathcal{X}_n, g, k)$ is the Minimum Coverage instance $(\mathcal{X}_n, k)$ and since maximizing $g(x) = |X|$ is equivalent to minimizing $|\overline{X}| = h(X)$, we recognize $((2^U, \cap), \overline{\mathcal{X}_n}, h, k)$ as the Minimum Intersection instance $(\overline{\mathcal{X}_n}, k)$. ■

4 Hardness

Consider the Maximum Balanced Complete Bipartite Subgraph (MBCBS) problem defined as follows.

Problem 4 (Maximum Balanced Complete Bipartite Subgraph) *Given a bipartite graph $G = ((V, W), E)$ find a maximum biclique H in G such that $|H \cap V| = |H \cap W|$.*

This problem is known to be NP-hard as the associated decision problem is NP-complete [13].

We start by noting that any vertex in a bipartite graph that has no incident edges is of no interest in the context of the MBCBS problem and can be removed in polynomial time. Hence, we can assume that any instance $G = ((V, W), E)$ of the MBCBS problem can be represented by $\mathcal{X}_n \in \mathcal{C}_n(2^W)$. Further, let $\mathcal{A}$ be an algorithm for the MaxI problem such that $\mathcal{A}(\mathcal{X}_n, k)$ is the solution given by $\mathcal{A}$ for the MaxI instance $(\mathcal{X}_n, k)$. Now consider the MBCBS algorithm given as Algorithm 2.

Algorithm 2: The MBCBS algorithm using the MaxI algorithm $\mathcal{A}$.

```
MBCBS(X_n)
i ← n
while i > 0
    S' ← A(X_n, i)
    s ← | ∩_{x∈S'} X_n(x)|
    if s ≥ i
        return S'
    i ← i − 1
return ∅
```

Lemma 1 *Let H_o be an optimal solution to MBCBS instance $\mathcal{X}_n$, and let H be the solution found by applying* MBCBS($\mathcal{X}_n$)*. Then*

$$\frac{|H_o|}{|H|} \leq r \tag{10}$$

where r is the performance ratio of the best known algorithm $\mathcal{A}$ for MaxI.

Proof. Let $\mathcal{A}_o$ be an optimal algorithm for the MaxI problem, let $a_o(i) = |\cap_{x\in\mathcal{A}_o(\mathcal{X}_n,i)} \mathcal{X}_n(x)|$, and let $a(i) = |\cap_{x\in\mathcal{A}(\mathcal{X}_n,i)} \mathcal{X}_n(x)|$. Note that

$$a_o(i) \geq a_o(i+1) \ .$$

Note that we can, without loss of generality, assume that

$$a(i) \geq a(i+1) \ .$$

This because we can construct a MaxI algorithm $\mathcal{A}'$ that for a given i in polynomial time finds $n \geq j \geq i$ such that $a(j) - a(i)$ is maximal, and return any i size subset S'' of the corresponding solution. We then have that $|S''^{\cap}| \geq a(l)$, for any $l > i$, and we can use $\mathcal{A}'$ instead of $\mathcal{A}$ in MBCBS.

Now let i_o be the largest i in MBCBS such that $a_o(i) \geq i$, and let $i_{\mathcal{A}}$ be the largest i in MBCBS such that $a(i) \geq i$. Inspecting the MBCBS algorithm we have that

$$\begin{aligned} a_o(i_o+1) &< i_o+1 \\ a_o(i_o) &\geq i_o \\ a(i_{\mathcal{A}}+1) &< i_{\mathcal{A}}+1 \\ a(i_{\mathcal{A}}) &\geq i_{\mathcal{A}} \ . \end{aligned}$$

Now note that if $i_o = i_{\mathcal{A}}$ we have that $\frac{|H_o|}{|H|} = 1$. Then the theorem holds as 1 is a lower bound for performance ratios under our (implicitly assumed standard) computational model. Therefore in the following assume that $i_o \neq i_{\mathcal{A}}$, which means that $i_o > i_{\mathcal{A}}$. Combining the above we then get

$$a(i_{\mathcal{A}}) \geq i_{\mathcal{A}} \geq a(i_{\mathcal{A}}+1) \geq a(i_o) \ ,$$

and

$$a_o(i_{\mathcal{A}}) \geq a_o(i_{\mathcal{A}}+1) \geq a_o(i_o) \geq i_o \ .$$

Using this we get that

$$\frac{|H_o|}{|H|} = \frac{i_o}{i_{\mathcal{A}}} \leq \frac{a_o(i_o)}{a(i_o)} \leq r \ . \tag{11}$$

■

Theorem 2 *MinC is NP-hard.*

Proof. We first note that if $\mathcal{A}$ runs in polynomial time, then MBCBS runs in polynomial time. Then the NP-hardness of MaxI follows from Lemma 1, as we can solve the NP-hard MBCBS problem optimally in polynomial time if we have an optimal polynomial time algorithm for MaxI. The theorem then follows from Proposition 4. ■

Corollary 1 *MJ and MSJ are NP-hard.*

We now state the main theorem of this section, that MinC is essentially as hard to approximate as the MBCBS problem.

Theorem 3 *Let n be the number of vertices in a MBCBS problem instance graph, and let g be a monotonically increasing function. Let the MBCBS problem be hard to approximate within a factor of $g(n)$ unless proposition P holds. Then MinC is hard to approximate within $g(n)$ unless proposition P holds.*

Proof. This follows from Lemma 1 and Proposition 4. ■

Corollary 2 *MJ and MSJ are essentially at least as hard to approximate as MBCBS.*

Corollary 3 *Let $\epsilon > 0$ be an arbitrarily small constant. Then there is no polynomial time algorithm for MinC that achieves an approximation ratio $n^{\epsilon'}$ where $\epsilon' = \frac{1}{2^{O(1/\epsilon \log(1/\epsilon))}}$ unless there exists probabilistic algorithm for SAT that runs in time 2^{n^ϵ} on an instance of size n. Also, assuming that $NP \not\subseteq \text{BPTIME}(2^{n^\epsilon})$ MinC allows no PTAS.*

Proof. This follows from Theorem 3 and hardness results established by Khot [14]. ■

5 Approximation of Minimum Loss k-Ambiguity

Informally, k-ambiguity is a property of a transformation Δ of a data set X such that every point in the transformed data set $\Delta(X)$ is a generalization of at least k elements in X. Following Vinterbo [9] we express this formally in Problem 5.

Problem 5 (Minimum Loss k-Ambiguity) *Let V be a set and let $\leq$ be a partial order on V with a single maximal element. For $v \in V$ and $\mathcal{X}_n \in \mathcal{C}_n(V)$ define $\sigma(v) = \{i \in I_n | \mathcal{X}_n(i) \leq v\}$ to be the "meaning" of element v with respect to $\mathcal{X}_n$. Associate with V a measure $\lambda : V \to [0, \infty)$ of information loss computable in polynomial time and let the measure λ be monotone and increasing in our partial order $\leq$. Given finite $\mathcal{X}_n \in \mathcal{C}_n(V)$ and a positive integer $k \leq n$ find $\Delta : V \to V$ such that for each $x \in I_n$ both*

$$x \in \sigma(\Delta(\mathcal{X}_n(x))) \tag{12}$$

$$|\sigma(\Delta(\mathcal{X}_n(x)))| \geq k \tag{13}$$

hold and $\lambda(\Delta(\mathcal{X}_n(x)))$ is minimized.

Vinterbo [9] calls the requirements (12) the preservation of meaning, and (13) k-ambiguity.

Theorem 4 *Any instance $\mathcal{X} = (V, \leq, \mathcal{X}_n, \lambda, k)$ of Problem 5 such that $(V, \leq)$ is a join-semilattice $(V, \vee)$ and for which*

$$\lambda(x \vee y) \leq \lambda(x) + \lambda(y)$$

holds for all $x, y \in V$, is approximable within $k - p$ in $O((2n(k-p) + p + (n-1)^p)t(\vee))$ time where $p \leq k$ is a positive integer.

Proof. Since we have that the join $\vee(x_1, x_2, \ldots, x_k)$ for any k elements $x_1, x_2, \ldots, x_k \in V$ can be written as a join $\vee(x_1 \vee x_2, x_1 \vee x_3, \ldots, x_1 \vee x_k)$ of $k-1$ elements, we have that all joins of size k elements from $\mathcal{X}_n$ containing one fixed element $x = \mathcal{X}_n(j)$ can be written as all joins of $k-1$ elements from $\mathcal{X}^x_{n-1} = \{(i, x \vee \mathcal{X}_n(y)) | y \in I_n - \{j\} \wedge i = y - I(y > j)\}$, where I is the Boolean indicator function. Computing $\mathcal{X}^x_{n-1}$ can be done in polynomial time as $\vee$ is polynomial time computable.

Let $\mathcal{Y}_x = ((V, \vee), \mathcal{X}^x_{n-1}, \lambda, k-1)$ be an instance of MSJ, and let for for all $x \in X$ $\Delta(x) =$ Greedy-MSJ$(\mathcal{Y}_x, \lambda, k-1, p)^\vee$. From the argument above and Theorem 1, we can conclude the proof. ■

Problem 6 (k-Ambiguity by Cell Suppression) *Let X be a set, let $\top \notin X$, and $V = (X \cup \{\top\})^m$. Define $*$ on V such that $(x_1 x_2 \cdots x_m) * (y_1 y_2 \cdots y_m) = (z_1 z_2 \cdots z_m)$ where $z_i = x_i$ if $x_i = y_i$ and $z_i = \top$ otherwise, and let $\mathcal{X}_n \in \mathcal{C}_n(V)$. The class of Problem 5 instances $(V, *, \mathcal{X}_n, \lambda, k)$, where $\lambda(x_1 x_2 \cdots x_m) = \sum_{i=1}^m I(x_i = \top)$ where I is the Boolean indicator function, is called the k-ambiguity by cell suppression problem.*

We see that the example of k-ambiguity given in Section 1 is an instance of k-ambiguity by cell suppression over $V = \{0, 1, \top\}^4$. We can use the connection between k-ambiguity by cell suppression and MinC to show that the former is as hard to approximate as the latter.

Theorem 5 *k-ambiguity by cell suppression is essentially as hard to approximate as the MBCBS problem.*

Proof. Let $(C = (S_1, S_2, \ldots, S_n), k)$ be an instance of the Minimum Coverage problem. We can in polynomial time construct a binary data table with $n+1$ rows in which row 1 differs from row j in attributes corresponding exactly to S_{j-1}. Let S be the set of suppressed entries in row 1 returned by a k-ambiguity by cell suppression algorithm. Then we know that S is a superset of at least k of the sets in C. We can in polynomial time find k of these. The non-approximation results then follow from Theorem 3. ■

Corollary 4 *Problem 5 is essentially at least as hard to approximate as the MBCBS problem.*

6 Discussion

The above results suggest that finding a PTAS for the Minimum Join problem is highly unlikely. However, it seems likely that the analysis of particular NP-Hard specializations of MSJ might yield better approximation algorithms for these than the generic Greedy-MSJ algorithm.

The proof of Greedy-MSJ failure on non-subadditive MJ instances begs the question whether such examples of failure can be constructed for larger classes of algorithms.

Even though our analysis of MSJ was primarily motivated by the connection to privacy in databases, there are other motivating factors as well. Minimum Coverage, in addition to being a complement

to Maximum Coverage analyzed by Hochbaum et al. [12], and its equivalent problem, Maximum Intersection, can be used to gain insight into problems from other areas as well.

One such (arguably esoteric) example is the problem of finding k out of n integers that share the most factors. As each square-free integer in a finite set can be represented by a finite set containing its prime factors, the problem restriction to square-free integers corresponds to the Maximum Intersection problem. This means that it is for instance highly unlikely that there exists a PTAS for the square-free instance and hence the problem in general.

Another example might be a location selection type problem. Assume we have k factories each of which we wish to place in one of n locations. Each placement has a set of requirements that needs to be met. The requirements are such that if it is met for one location, it is also met for all other locations. Assuming that the the fulfillment of each requirement has a positive cost associated with it, it is natural to seek out the k locations that minimize this cost.

Acknowledgments.

Thanks go to Stephan Dreiseitl for fruitful input. This work was funded by NIH grant R01 LM007273-04A1.

References

[1] Melton, L.: The threat to medical-records research. N Engl J Med **337**(20) (Nov 1997) 1466–70

[2] Dick, R.S., Steen, E.B., Detmer, D.E.: The Computer Based Patient Record: An Essential Technology for Health Care, Revised Edition. Institute of Medicine (1997)

[3] Hippocrates: The oath and law of hippocrates. Volume 38 of The Harvard Classics. P.F. Collier & Son, New York (1909-1914)

[4] United States Department of Health and Human Services: 45 CFR Parts 160 and 164 RIN 0991-AB14, Standards for Privacy of Individually Identifiable Health Information. Federal Register **67**(157) (August 2002)

[5] Walton, J., Doll, R., Asscher, W., Hurley, R., Langman, M., Gillon, R., Strachan, D., Wald, N., Fletcher, P.: Consequences for research if use of anonymised patient data breaches confidentiality. BMJ **319**(7221) (Nov 1999) 1366

[6] Sweeney, L.: k-anonymity: a model for protecting privacy. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems **10**(5) (2002) 557–570

[7] Hundepool, A.J., Willenborg, L.C.R.J.: Mu- and tau-argus: Software for statistical disclosure control. In: Third International Seminar on Statistical Confidentiality at Bled. (1996)

[8] Øhrn, A., Ohno-Machado, L.: Using boolean reasoning to anonymize databases. Artif Intell Med **15**(3) (1999) 235–54

[9] Vinterbo, S.A.: Privacy: A machine learning view. IEEE Transactions on Knowledge and Data Engineering **16**(8) (August 2004) 939–948

[10] Meyerson, A., Williams, R.: On the complexity of optimal k-anonymity. In: PODS '04: Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, New York, NY, USA, ACM Press (2004) 223–228

[11] Aggarwal, G., Feder, T., Kenthapadi, K., Motwani, R., Panigrahy, R., Thomas, D., Zhu, A.: Approximating algorithms for k-anonymity. Journal of Privacy Technology **1** (2005) 1–18

[12] Hochbaum, D., Pathria, A.: Analysis of the greedy approach in covering problems. Naval Research Quarterly **45** (1998) 615–627

[13] Garey, M.R., Johnson, D.S.: Computers and Intractability, A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, New York (1979)

[14] Khot, S.: Ruling out ptas for graph min-bisection, densest subgraph and bipartite clique. In: FOCS, IEEE Computer Society (2004) 136–145