Private Itemset Support Counting

Sven Laur[1], Helger Lipmaa[2,3], and Taneli Mielikäinen[4]

[1] Helsinki University of Technology, Finland
[2] Cybernetica AS, Estonia
[3] University of Tartu, Estonia
[4] University of Helsinki, Finland

Abstract. Private itemset support counting (PISC) is a basic building block of various privacy-preserving data mining algorithms. Briefly, in PISC, Client wants to know the support of her itemset in Server's database with the usual privacy guarantees. First, we show that if the number of attributes is small, then a communication-efficient PISC protocol can be constructed from a communication-efficient oblivious transfer protocol. The converse is also true: any communication-efficient PISC protocol gives rise to a communication-efficient oblivious transfer protocol. Second, for the general case, we propose a computationally efficient PISC protocol with linear communication in the size of the database. Third, we show how to further reduce the communication by using various tradeoffs and random sampling techniques.
Keywords: privacy-preserving data mining, private frequent itemset mining, private itemset support counting, private subset inclusion test.

1 Introduction

Frequent itemset mining—also known as frequent pattern mining—is a central task in data mining that has driven research in data mining for ten years. Nowadays, there are special workshops on various aspects of frequent itemset mining [BGZ04]. The goal in frequent itemset mining is to find all frequent itemsets in a given transaction database. Many kinds of data can be viewed as transaction databases and various data mining tasks arising in document analysis, web mining, computational biology, software engineering and so on can be modelled as frequent itemset mining. For example, one can use frequent itemset mining to find which items are usually bought together in a supermarket, or to analyse the correlation between various patterns in the genome database. The mining of frequent itemsets is a very challenging problem, and it is clearly even more challenging in the scenarios when one encounters privacy issues. Several researchers have studied the distributed case with multiple servers, all having a part of the database, who need to mine frequent itemsets in the joint database without revealing each other too much extra information.

We are concerned with a slightly different scenario where the database is owned by a single party, Server, who sells the result of frequent itemset mining (either the collection of all frequent itemsets or the support of a fixed itemset) to others. That is, we consider the itemset support counting (ISC) problem, which is often used as a building block of frequent itemset mining or association rules mining, but is important also by itself. As an example of ISC, Server could maintain a commercial citation database, and Client could want to find out how many people cite both herself and Shannon. Other possible examples include Internet search engines, mining in medical databases, etc. In most of such applications, some form of privacy must be guaranteed. On the one hand, it is not in Server's interests that Client obtains more information than she has paid for; moreover, in some cases like the medical databases, giving out more information might even be illegal. On the other hand, Client also does not necessarily want Server to know which itemset interests her.

To define the private itemset support counting, let us first describe the setting more formally. Server owns a $m \times n$ Boolean database $\mathcal{D}$ that can be considered as a multiset of m subsets of the set $[n] = \{1, \ldots, n\}$. Every row is $\mathcal{D}$ is called a transaction; it might correspond to a transaction in supermarket, with $j \in \mathcal{D}[i]$ if jth item was purchased during the ith transaction. A subset of $[n]$ is called an itemset, it corresponds to the set of items that can be in the ith transaction (e.g., the set of items that were bought together). The goal of Client is to determine the support $\operatorname{supp}_{\mathcal{D}}(\mathcal{Q}) := |\{i : \mathcal{Q} \subseteq \mathcal{D}[i]\}|$ of an itemset $\mathcal{Q} \subseteq [n]$ in $\mathcal{D}$, i.e., to find out how many of Server's transactions contain $\mathcal{Q}$. In an $(m \times n)$-*private itemset support counting* (PISC) protocol, Client retrieves $\operatorname{supp}_{\mathcal{D}}(\mathcal{Q})$, so that (1) she will get no other information about the database $\mathcal{D}$ (server-privacy) and (2) Server gets no information about $\mathcal{Q}$ (client-privacy). In the scope of this paper, we require server-privacy to be information-theoretical and client-privacy to be computational.

The data mining setting implies a few non-standard considerations, mostly due to the large amounts of the handled data. First, m and n can be very large (e.g., $m, n \geq 10\,000$), so whenever possible, it is desirable to have communication and computation of order $o(mn)$. Second, again due to the large amount of data, it is impractical to have protocols that are verifiable or even provide correctness. Therefore, we only focus on the privacy issues of the PISC protocols. Thus, we use relaxed security definitions, standard in the case of computationally-private information retrieval, oblivious transfer and oblivious keyword search protocols, where the security of the client is only defined by requiring that his query will remain private. Moreover, we construct protocols that are private in the semi-honest model since they are usually efficient and may suffice in the practice. Protocols, private in the malicious model, can be constructed by adding standard zero-knowledge proofs. In all cases, we put emphasis both on the efficiency of the protocols and on the provable security.

First, we show a close correspondence between PISC and CPIR by providing tight two-way reductions between these two problems. More precisely: (a) Given a $\binom{s}{1}$-CPIR protocol CPIR of ℓ-bit strings with communication $C_{\mathsf{CPIR}}(s, \ell)$, we show how to construct a $(2^n \times n)$-PISC protocol $\mathsf{CPIR\text{-}PISC}$ with communication $C_{\mathsf{CPIR}}(2^n, n)$. Taking the recent $\binom{s}{1}$-oblivious transfer protocol for ℓ-bit strings of Lipmaa [Lip05] with communication $\Theta(\log^2 s + \ell \cdot \log s)$, this results in communication $\Theta(n^2)$. (The use of a very recent CPIR protocol by Gentry and Ramzan [GR05] results in communication $\Theta(n)$.) However, in the case of $\mathsf{CPIR\text{-}PISC}$, Server needs to store a table of $2^n \cdot n$ bits and then execute the CPIR protocol on 2^n elements, which is infeasible when say $n \geq 20$, while in a realistic data mining application, n might be larger than $10\,000$. (b) Given a $(m \times n)$-PISC protocol PISC with communication $C_{\mathsf{PISC}}(m, n)$, we show how to construct a $\binom{2^n-1}{1}$-CPIR protocol $\mathsf{PISC\text{-}CPIR}$ on 1-bit strings with communication $C_{\mathsf{PISC}}(2^n, n)$. This enables us to carry over several standard results on the CPIR and oblivious transfer protocols to the PISC scenario. Moreover, the reductions increase communication at most by factor of n, therefore the optimal communication of the CPIR and PISC protocols can differ only by a logarithmic term in the database size.

For databases with many attributes, we describe an alternative $(m \times n)$-PISC protocol $\mathsf{PSI\text{-}PISC}$ that uses a new *private subset inclusion protocol* PSI, also described in this paper, as a subroutine. The resulting protocol has communication $(n + m + 1) \cdot k$, where k is the bit-length of a ciphertext, and is private in the semi-honest model assuming that the used homomorphic cryptosystem (a) has plaintext space with prime cardinality, and (b) is IND-CPA secure; the Decisional Diffie-Hellman Assumption is sufficient here. The protocol can be made secure in the malicious model by using standard (non-interactive) zero-knowledge proofs. The $\mathsf{PSI\text{-}PISC}$ protocol is computationally feasible even when $n \approx 10\,000$, since the computational work of Server is of order $\Theta(n + m + w(\mathcal{D}))$ encryptions and decryptions, where $w(\mathcal{D})$ is the number of 1-bits in the usually very sparse database $\mathcal{D}$.

In addition, we study imprecise protocols: we discuss the problem of just detecting whether the given itemset is frequent and study sampling techniques. Random sampling of the database and approximating the itemset support based on the support in the sample allows us to cheaply extend the $\mathsf{PSI\text{-}PISC}$ protocol to huge databases, supposing that Client is willing to accept approximate answers.

2 Preliminaries

For an integer s, denote $[s] := \{1, 2, \ldots, s\}$. For a nonnegative integer X, let $\mathsf{len}(X) := \lceil \log_2(X + 1) \rceil$ denote the number of bits it takes to store and transfer X. The statistical difference of two distributions X and Y over the discrete support Z is defined as $\mathsf{Dist}\,(X||Y) := \max_{S \subseteq Z} |\Pr[X \in S] - \Pr[Y \in S]|$.

Data mining setting. Our setting is the following, very common one in data mining. The Server has a transaction database $\mathcal{D}$ over n attributes (or items) $A_1, A_2, \ldots, A_n$ and the database consists of m transactions. A transaction is a subset of the attributes. Alternatively, a transaction database $\mathcal{D}$ of m transactions over n attributes can be considered as a $m \times n$ binary matrix $\mathcal{D}$ where the entry (i, j) is one iff $A_j \in \mathcal{D}[i]$. In a realistic setting, the resulting 0-1 matrix can have e.g. 100 000 transactions (rows) and 50 000 attributes (columns).

The frequent itemset mining task is, given a transaction database $\mathcal{D}$ of m rows and a minimum frequency threshold $\sigma \in (0, 1]$, to find the subsets of attributes that are contained in σ-fraction of the transactions, i.e., to determine the collection $\mathcal{F} = \{X \subseteq \{A_1, \ldots, A_n\} : \mathrm{freq}_{\mathcal{D}}(X) \geq \sigma\}$ of σ-frequent itemsets in $\mathcal{D}$ where $\mathrm{freq}_{\mathcal{D}}(X) = |\{i \in [m] : X \subseteq \mathcal{D}[i]\}| / m = \mathrm{supp}_{\mathcal{D}}(X)/m$. Alternatively, the set of frequent itemsets can be specified by the support threshold as $\mathcal{F} = \{X \subseteq \{A_1, \ldots, A_n\} : \mathrm{supp}_{\mathcal{D}}(X) \geq \sigma \cdot m\}$. Usually also the frequencies or the supports of the frequent itemsets are required. We assume that attribute labels $A_1, \ldots, A_n$ are public and thus can be substituted with canonical labelling $\{1, \ldots, n\}$.

Although frequent itemset mining can be done in time $\mathcal{O}(mn\kappa)$ [AMS+96], where κ is the number of frequent itemsets, the running time can easily become intractable, since κ itself is exponential in the cardinality of the largest frequent itemset due to the monotonicity of the support. Therefore, various output compaction techniques are known from the literature, see [BGZ04] for an up-to-date overview of frequent itemset mining algorithms and references.

Sampling bounds. Let $X_1, \ldots, X_k$ be independent random 0-1 variables that are drawn from the same distribution with the expectation $\mu = \Pr[X_i = 1]$. Let X be the average $(X_1 + \cdots + X_k)/k$. Then the Chernoff bound $\Pr[(1-\varepsilon)\mu \leq X \leq (1+\varepsilon)\mu] \leq 2 \cdot \exp\left(-\frac{k\mu\varepsilon^2}{4}\right)$ describes the distribution of relative error and the Hoeffding bound $\Pr[|X - \mu| \geq \varepsilon] \leq 2 \cdot \exp\left(-2k\varepsilon^2\right)$ the distribution of absolute error.

IND-CPA secure homomorphic cryptosystems. A public-key cryptosystem is a triple $\Pi = (G, E, D)$, where G is the key generation algorithm that returns $(\mathsf{sk}, \mathsf{pk})$ consisting of a secret key sk and a public key pk, E is the encryption algorithm and D is the decryption algorithm. For a fixed public-key cryptosystem Π and a secret key sk, let $\mathcal{C}$ be the ciphertext space, let $\mathcal{R}$ be the randomness space and let $\mathcal{P}$ be the plaintext space. Then, $E_{\mathsf{pk}} : \mathcal{P} \times \mathcal{R} \rightarrow \mathcal{C}$ and $D_{\mathsf{sk}} : \mathcal{C} \rightarrow \mathcal{P}$. Define $\mathsf{Adv}^{\mathsf{indcpa}}_{\Pi}(A) := 2 \cdot |\Pr[(\mathsf{sk}, \mathsf{pk}) \leftarrow G, (m_0, m_1) \leftarrow A(\mathsf{pk}), b \leftarrow \{0,1\} : A(m_0, m_1, E_{\mathsf{pk}}(m_b; \mathcal{R})) = b] - \frac{1}{2}|$. We say that Π is (τ, ε)-*secure in the sense of IND-CPA* if $\mathsf{Adv}^{\mathsf{indcpa}}_{\Pi}(A) \leq \varepsilon$ for any probabilistic algorithm A that works in time τ.

Cryptosystem Π is *homomorphic* if for any key pair $(\mathsf{sk}, \mathsf{pk})$, any $m, m' \in \mathcal{P}$ and any $r, r' \in \mathcal{R}$, $E_{\mathsf{pk}}(m; r) \cdot E_{\mathsf{pk}}(m'; r') = E_{\mathsf{pk}}(m + m'; r \circ r')$, where $+$ is a group operation in $\mathcal{P}$, and $\circ$ is a groupoid operation in $\mathcal{R}$. A few homomorphic cryptosystems are proven to be secure in the sense of IND-CPA under reasonable complexity assumptions.

Definitions of Client and Server privacy. Assume that Client and Server want to securely compute a functionality f, so that Client receives $f(\mathcal{Q}, \mathcal{D})$ and Server receives nothing, where $\mathcal{Q}$ is Client's private input and $\mathcal{D}$ is Server's private input. In our case, Server's input is potentially so huge that all currently known cryptographic techniques fail to provide correctness in tractable time. Therefore, we consider only privacy issues, i.e., we use relaxed security definitions. Thus, we do not require Server to commit to or even "know" a database to which Client's search is effectively applied. Such a relaxation is standard in the case of protocols like oblivious transfer, computationally-private information retrieval and oblivious keyword search; our security definitions correspond closely to the formalisation given in [FIPR05]. Moreover, in a semi-honest case, all proposed protocols have two messages and therefore, standard security definitions can be somewhat simplified.

Denote by Client an honest Client and by Server an honest Server. Let $\mathsf{Client}_{\mathsf{sk}}(\cdot; \cdot)$ denote Client's (first) message and $\mathsf{Server}_{\mathsf{pk}}(\cdot; \cdot)$ denote Server's (second) message. Let $\mathcal{R}_{\mathsf{Client}}$ (resp. $\mathcal{R}_{\mathsf{Server}}$) be the randomness space of an honest Client (resp. Server). Then we say that a two-message protocol Π is (τ, ε)-*client-private* (in the malicious model), if for any probabilistic algorithm A with the working time τ, $\mathsf{Adv}^{\mathsf{c-privacy}}_{\Pi}(A) := 2 \cdot \max_{(\mathcal{Q}_0, \mathcal{Q}_1)} \left|\Pr[b \leftarrow \{0,1\} : A(\mathcal{Q}_0, \mathcal{Q}_1, \mathsf{Client}(\mathcal{Q}_b; \mathcal{R}_{\mathsf{Client}})) = b] - \frac{1}{2}\right| \leq \varepsilon$. Here,

$\mathcal{Q}_0$ and $\mathcal{Q}_1$ are assumed to be valid client-side inputs, and the probability is taken over the coin tosses of Client, A and over the choices of b.

We define information-theoretical server-privacy in the semihonest model by requiring that for every unbounded honest-but-curious algorithm A, one can define a simulator Sim that, given solely A's private input $\mathcal{Q}$, A's random coins r, and A's private output $f(\mathcal{Q}, \mathcal{D})$, generates output that is statistically indistinguishable from the view $(\mathsf{msg}_1, \mathsf{msg}_2)$ of A that reacts with the honest Server, where $\mathsf{msg}_1 \leftarrow A(\mathcal{Q}; r)$ and $\mathsf{msg}_2 \leftarrow \mathsf{Server}(\mathcal{D}, \mathsf{msg}_1; \mathcal{R}_{\mathsf{Server}})$. More precisely, the advantage of A is defined $\mathsf{Adv}_{\Pi}^{\mathsf{s-privacy}}(A) := \max_{(\mathcal{Q},\mathcal{D})} \mathsf{Dist}\left(\mathsf{Sim}_{\mathsf{pk}}(\mathcal{Q}, r, f(\mathcal{Q}, \mathcal{D}))||(\mathsf{msg}_1, \mathsf{msg}_2)\right)$. Here, $\mathcal{D}$ is assumed to be a valid Server-side input. Protocol is *ε-server-private* (in the semihonest model), if for all unbounded honest-but-curious A, $\mathsf{Adv}_{\Pi}^{\mathsf{s-privacy}}(A) < \varepsilon$. Security in the malicious model is defined as usually.

Computationally-private information retrieval (CPIR) and oblivious transfer (OT). During a single-server $\binom{m}{1}$*-computationally-private information retrieval* protocol, Client fetches $\mathcal{D}[\mathcal{Q}]$ from the database $\mathcal{D} = (\mathcal{D}[1], \dots, \mathcal{D}[m])$, $\mathcal{D}[i] \in \mathbb{Z}_\ell$ for some fixed domain $\mathbb{Z}_\ell$, so that a computationally bounded Server does not know which entry Client is learning. In the case of a two-message CPIR protocol, we can used the previously previously given client-privacy definition. An (τ, ε)-client-private $\binom{m}{1}$-CPIR is an *(computationally) (τ, ε)-client-private and (information-theoretically) ε'-server-private* $\binom{m}{1}$*-OT protocol* if it is additionally ε'-server-private. A recent $\binom{m}{1}$-CPIR protocol by Lipmaa [Lip05], $\binom{m}{1}$-$\mathsf{LipmaaCPIR}$, has asymptotic communication $\Theta(\log^2 m + \log m \cdot \ell)$ (assuming that the security parameter is a constant). Based on the Aiello-Ishai-Reingold CPIR-to-OT transform [AIR01], Lipmaa also described an $\binom{m}{1}$-OT protocol with the same asymptotic communication. Lipmaa's protocols are client-private assuming that the underlying Damgård-Jurik homomorphic cryptosystem is IND-CPA secure, or equivalently, if the Decisional Composite Residuosity Problem is hard. Lipmaa's protocols are unconditionally server-private. A very recent $\binom{m}{1}$-CPIR protocol by Gentry and Ramzan [GR05] has communication $\Theta(\log m + \ell)$.

Private Keyword Search. In many data-mining applications, the data is indexed by a relatively small subset of keys $\mathcal{K} \subseteq \{1, \dots, m\}$, where the set $\mathcal{K}$ itself is private. Therefore, if a Client wants to privately access $\mathcal{D}[\mathcal{Q}]$ the straightforward solution, a $\binom{m}{1}$-OT to the database where empty slots are filled with dummy elements is suboptimal. Several solutions that improve communication and computation costs in this situation [CGN97,OK04,FIPR05] have been proposed. Such solutions either combine hash tables and oblivious transfer, or use oblivious evaluation of pseudo-random functions.

3 Basic Cryptographic Tool: Private Subset Inclusion Protocol

In a private subset inclusion (PSI) protocol, Client has a set $\mathcal{Q} \subseteq [n]$, Server has a set $\mathcal{S} \subseteq [n]$, and Client must establish whether $\mathcal{Q} \subseteq \mathcal{S}$ or not without neither of the parties obtaining any additional information. More precisely, the protocols must satisfy client-privacy and server-privacy as formalised in Sect. 2, where for the ease of implementation we define $f(\mathcal{Q}, \mathcal{S}) = 0$, if $\mathcal{Q} \subseteq \mathcal{S}$, and $f(\mathcal{Q}, \mathcal{S}) \neq 0$, otherwise. We use the fact that $\mathcal{Q} \subseteq \mathcal{S} \iff |\mathcal{Q} \cap \mathcal{S}| = |\mathcal{S}|$. Let $\mathcal{Q}$ (resp. $\mathcal{S}$) also denote the characteristic function of $\mathcal{Q}$ (resp. $\mathcal{S}$). That is, $\mathcal{Q}[i] = 1 \iff i \in \mathcal{Q}$ and $\mathcal{S}[i] = 1 \iff i \in \mathcal{S}$.

PRIVATE INPUT: Client has a set $\mathcal{Q}$ and Server has a set $\mathcal{S}$.
PRIVATE OUTPUT: Client knows whether $\mathcal{Q} \subseteq \mathcal{S}$ or not.

Message 1, Client Generate a new key pair $(\mathsf{sk}, \mathsf{pk}) \leftarrow G$. Send pk to Server.
For any $i \in [n]$, generate a new nonce $r_i \leftarrow_r \mathcal{R}$. Send $c_i \leftarrow E_{\mathsf{pk}}(\mathcal{Q}[i]; r_i)$ to Server.
Message 2, Server Draw $s \leftarrow_r \mathcal{P}$, $r \leftarrow_r \mathcal{R}$ uniformly at random. Set $c \leftarrow (\prod_{i:\mathcal{S}[i]=0} c_i)^s \cdot E_{\mathsf{pk}}(0; r)$. Send c to Client.
Post-processing by Client Set $t \leftarrow D_{\mathsf{sk}}(c)$. Accept that $Q \subseteq \mathcal{S}$ iff $t = 0$.

Protocol 1: Private homomorphic subset inclusion test protocol

To solve PSI, we could use a recent private set intersection cardinality protocol by Freedman, Nissim and Pinkas [FNP04]. However, their solution requires a costly secure-circuit evaluation since the intersection cardinality must remain private. Protocol 1, based on ideas from [AIR01,Lip03], is a conceptually simpler and more efficient alternative, especially when security either in the malicious model is required or the protocol is used in the context of itemset counting as later in Protocol 3. Here, we explicitly assume that the plaintext length is at least $\mathsf{len}(n)$ bits, where $n \geq |\mathcal{Q} \cup \mathcal{S}|$ is the *a priori* fixed domain size. This assumption is always true in practice.

Theorem 1. *Let Π be a (τ, ε) IND-CPA secure homomorphic cryptosystem and let n be smaller than any prime divisor of $|\mathcal{P}|$. Then Protocol 1 is $(\tau - \mathcal{O}(n), n\varepsilon)$-client-private and 0-server-private in the semi-honest model. Protocol 1 is correct with probability $1 - |\mathcal{P}|^{-1}$.*

Proof. First, $\mathcal{Q} \subseteq \mathcal{S}$ iff $w := \sum_{i:\mathcal{S}[i]=0} \mathcal{Q}[i] = 0$. Therefore, homomorphic properties of Π assure that c is a random encryption of zero, if $\mathcal{Q} \subseteq \mathcal{S}$. If $\mathcal{Q} \not\subseteq \mathcal{S}$, then $w \leq |\mathcal{Q}| \leq n$ is not a divisor of $|\mathcal{P}|$ and thus c is a random encryption of a random plaintext. Consequently, the probability that $\mathcal{Q} \not\subseteq \mathcal{S}$ if c is an encryption of zero is $|\mathcal{P}|^{-1}$. Computational client-privacy follows directly from the IND-CPA security of Π. As Server sees only n ciphertexts, any adversary A that can distinguish two vectors of ciphertexts can be used for distinguishing only two ciphertexts. The corresponding hybrid argument is fairly standard. Server-privacy is guaranteed as the second message depends only on whether $\mathcal{Q} \subseteq \mathcal{S}$ or not. $\square$

Security in the malicious model. A standard way to make the described protocol private in the malicious model is to let Client to prove the correctness of her actions; that means proving that (a) pk is a valid public key and that (b) every c_i encrypts either 0 or 1. This can be done by using (non-interactive) zero-knowledge or non-interactive zero-knowledge proofs of knowledge.

4 Exact Private Itemset Support Counting Protocols

Let $\mathcal{Q} \subseteq [n]$ be Client's query, $\mathcal{D}$ be the database and m be the number of the rows in the database; that is, $\mathcal{D}[i] \subseteq [n]$ for $i \in [m]$. More precisely, we treat $\mathcal{Q}$ as a binary n-tuple corresponding to the characteristic function of Client's input and $\mathcal{D}$ as an $m \times n$ binary matrix. Recall that in a PISC protocol, Client has to compute, in a privacy-preserving manner, the value $\mathrm{supp}_{\mathcal{D}}(\mathcal{Q}) := |\{i : \mathcal{Q} \subseteq \mathcal{D}[i]\}|$.

4.1 Relation Between PISC and CPIR

We first show that there are tight reductions between oblivious transfer and PISC protocols even if n is relatively small. For precise quantification, denote by $C_{\mathsf{CPIR}}(s, \ell)$ the communication of a $\binom{s}{1}$-computationally private information retrieval protocol CPIR on ℓ-bit strings. Similarly, let us denote by $C_{\mathsf{PISC}}(m, n)$ the communication of an $(m \times n)$-PISC protocol PISC.

Theorem 2. *(a) Let* CPIR *be a* $\binom{2^n}{1}$*-computationally private information retrieval protocol on* ℓ*-bit strings. Assume that* $\mathsf{len}(m) \leq \ell$*. Then there exists a client-private* $(m \times n)$*-PISC protocol* $\mathsf{CPIR\text{-}PISC}$ *with communication* $C_{\mathsf{CPIR}}(2^n, \ell)$*. Server has to pre-compute and store a table of* $2^n \cdot \mathsf{len}(m)$ *bits; this table can be computed in time* $\Theta(2^n \cdot m)$ *ignoring logarithmically-small multiplicands.*
(b) Let PISC *be a client-private* $(2^n \times n)$*-PISC protocol. Then there exists a* $\binom{2^n-1}{1}$*-CPIR protocol* $\mathsf{PISC\text{-}CPIR}$ *on* 1*-bit strings with communication* $C_{\mathsf{PISC}}(2^n, n)$*. Server has to pre-compute and store a table of* $\leq 2^n \cdot n$ *bits; this table can be computed in time* $\Theta(2^{2n} \cdot 2^n)$ *ignoring logarithmically-small multiplicands.*

Proof. (a) Server computes off-line the support of all 2^n itemsets in the database $\mathcal{D}$, and stores them in a new database $\mathcal{D}'$. Note that the database $\mathcal{D}'$ contains 2^n elements, each $\mathsf{len}(m)$ bits. After that, Client and Server use the $\binom{2^n}{1}$-CPIR protocol to retrieve the $\mathcal{Q}$th element of $\mathcal{D}'$. Clearly, Client learns the support of $\mathcal{Q}$, while Server obtains no new knowledge. If we use an oblivious transfer protocol instead of a CPIR protocol, then we get a server-private version of the $\mathsf{CPIR\text{-}PISC}$ protocol,

(b) Let $\mathcal{S} = \mathcal{S}[1] \dots \mathcal{S}[2^n - 1]$ be Server's $(2^n - 1)$-bit input, and let i be Client's query in the $\binom{2^n-1}{1}$-CPIR protocol. We construct a specific $2^n \times n$ binary database $\mathcal{D}$ such that itemset supports in it encode $\mathcal{S}$. More precisely, let $\chi(a) := (a_1, \dots, a_n)$ be a Boolean vector corresponding to the binary representation of a, that is, $a = \sum 2^{j-1} a_j$. The next algorithm builds a database $\mathcal{D}$ such that $\mathrm{supp}_{\mathcal{D}}(\chi(a)) \equiv \mathcal{S}[a] \mod 2$ for every $a \in \{1, \dots, 2^n - 1\}$:

1. Initialise $\mathcal{D}$ as a $2^n \times n$ all-zero matrix.
2. For $w = n$ downto 1 do
   For all a s.t. the vector $(a_1, \dots, a_n) \in \mathbb{Z}_2^n$ has Hamming weight w do
   (a) Set $v \leftarrow \mathrm{supp}_{\mathcal{D}}(a_1, \dots, a_n)$.
   (b) If $v \not\equiv \mathcal{S}[a] \mod 2$
   then replace the first all-zero row of $\mathcal{D}$ with $(a_1, \dots, a_n)$.

Since this algorithm considers itemsets in the order of decreasing cardinality, subsequent changes do not alter the already computed supports; thus, at the end all bits of $\mathcal{S}$ are correctly encoded. Moreover, the number of replaced rows is not greater than $2^n - 1$ and thus step 2b never fails to find an all-zero row. It is straightforward to derive the complexity bounds for this step.

Let $\mathcal{D}$ be the final version of this database. Now, when Client wants to make a CPIR query i, he instead forms the corresponding PISC query $\mathcal{Q} := \chi(i)$ and obtains $\mathrm{supp}_{\mathcal{D}}(\mathcal{Q})$ by executing PISC. Then, he computes $\mathcal{S}[i] \leftarrow \mathrm{supp}_{\mathcal{D}}(\mathcal{Q}) \mod 2$. Clearly, the client-privacy of PISC-CPIR follows directly from the client-privacy of the original PISC protocol. $\square$

By using similar but more complicated techniques, one can directly construct an oblivious transfer protocol based on a PISC protocol. In this case, the number of rows of the database $\mathcal{D}$ is still polynomial w.r.t. the number of encoded bits.

Corollary 1. *Assume that the Decisional Composite Residuosity Problem is hard. Assume that $n = \mathrm{polylog}(m)$. There exists a private $(m \times n)$-PISC protocol* CPIR-PISC *with communication $\Theta(n^2 \cdot \log_2 |\mathcal{P}| + n \cdot \mathsf{len}(m))$ and Server's online work $\Theta(2^n \cdot m)$.*

The use of a very recent $\binom{m}{1}$-CPIR protocol by Gentry and Ramzan [GR05] would result in communication $\Theta(n + \mathsf{len}(m))$.

As the communication complexity of non-private itemset support count is roughly $n + \mathsf{len}(m)$, Corollary 1 provides an almost communication-optimal solution. On the other hand, Thm. 2 indicates that any PISC protocol with optimal communication $\mathcal{O}(n + \log m)$ gives a rise to a $\binom{s}{1}$-CPIR protocol with communication $O(\log s)$. Moreover, the known lower and upper bounds on the CPIR protocols can be used to get lower and upper bounds for the $(m \times \mathrm{polylog}(m))$-PISC protocols. For example, given a trapdoor permutation, there exists an $(m \times \mathrm{polylog}(m))$-PISC protocol with communication $m - o(m)$. On the other hand, an $(m \times \mathrm{polylog}(m))$-PISC protocol with communication $m - o(m)$ implies the existence of one-way functions.

4.2 Oblivious Keyword Search-Based PISC

As a serious drawback, note that CPIR-PISC is practical only for small values of n, e.g., when $n \leq 20$, as the pre-computation step becomes quickly infeasible. The same applies for the CPIR step, as Server's workload is at least linear in the size of database in all CPIR protocols.

However, in specific settings the online complexity of CPIR can be drastically reduced. The first efficient protocol for oblivious keyword search was proposed by Ogata and Kurosawa [OK04]; their approach was extended in [FIPR05]. In these two protocols, during the first phase Server transfers the whole database, in an encrypted form, to Client. In the second phase, Client and Server invoke an oblivious pseudo-random function evaluation protocol. As the result, Client obtains a secret key that allows her to decrypt one database element. Though the initial communication of the protocol is linear in the database size, the second phase has poly-logarithmic communication and computation in the database size. Such a protocol is especially appealing if the second phase are repeated many times as it is commonly done in data-mining algorithms.

Thm. 2 can be used to transform an oblivious keyword search protocol to a PISC protocol. If one is interested in the frequencies of all different supports, then the resulting protocol is not really practical since the transfered PISC database must have 2^n elements. However, in data-mining applications, Client is often only interested in

PRIVATE INPUT: Client has a query $\mathcal{Q}$ and Server has a database $\mathcal{D}$.
PRIVATE OUTPUT: Client learns $\mathrm{supp}_{\mathcal{D}}(\mathcal{Q})$ if $\mathrm{freq}_{\mathcal{D}}(\mathcal{Q}) \geq \sigma$.

Setup Phase

Server runs a frequent itemset mining algorithm that outputs $\mathcal{F} = \{X : \mathrm{freq}_{\mathcal{D}}(X) \geq \sigma\}$.
Server chooses a secret key sk and for all $X \in \mathcal{F}$ computes $E_i \leftarrow \mathsf{ObPrf}_{\mathsf{sk}}(\mathsf{code}(X)) \oplus (0^\ell || \mathrm{supp}_{\mathcal{D}}(X))$.
Send list $\{E_i\}$, in a randomly permuted order, to Client.

Interactive Phase

Client and Server invoke Π_{ObPrf}. At the end Client obtains $\mathsf{mask} \leftarrow \mathsf{ObPrf}_{\mathsf{sk}}(\mathsf{code}(X))$.

Post-processing by Client

If there is an E_i such that $E_i \oplus \mathsf{mask} = (0^\ell || c)$ then output c;
else decide that $\mathrm{supp}_{\mathcal{D}}(\mathcal{Q}) < \sigma$.

Protocol 2: Protocol for PISC based on oblivious pseudo-random function evaluation

the supports of frequent itemsets. In such a case, Server can first run any conventional frequent itemset mining algorithm on the database using an appropriate minimum frequency threshold σ, and then encrypt supports of the obtained σ-frequent itemsets. In practice, the minimum frequency threshold σ is chosen to be as low as possible, so that the mining is still feasible, to get a good approximation of the supports of all itemsets.

Protocol 2 combines this idea with oblivious keyword search. This is relatively straightforward, but we have included the protocol for the sake of completeness. To read it, first recall that a two-argument pseudo-random function ObPrf is an OPRF, if it can be obliviously evaluated by Client and Server [FIPR05]. In other words, there exist a secure protocol Π_{ObPrf} such that after executing it on Client's private input x and Server's private input sk, Client learns $\mathsf{ObPrf}_{\mathsf{sk}}(x)$, while Server learns nothing. Second, we assume that each itemset $\mathcal{Q}$ has a short unique code $\mathsf{code}(\mathcal{Q})$, this can be a cryptographic hash of $\mathcal{Q}$.

Theorem 3. *Let* ObPrf *be* (τ, ε_1') *secure pseudo-random function with appropriate domain and range. Let the protocol* Π_{ObPrf} (τ, ε_1) *client-private and* (τ, ε_2') *server-private. Then Protocol 2 is* (τ, ε_1)*-client-private and* $(\tau - \mathcal{O}(1), \varepsilon_1' + \varepsilon_2')$*-server-private PISC protocol. Protocol 2 yields an incorrect end-result with probability* $2^{-\ell} \cdot |\mathcal{F}|$. *The interactive phase can be repeated over the same initially transformed encrypted database with a linear drop in concrete security.*

Proof (Sketch). We do not provide a complete proof, see [FIPR05] for details. Client-privacy and correctness are evident. Server-privacy follows from the next hybrid argument. First, consider a protocol Π_1, where Π_{ObPrf} is substituted with its ideal implementation. Let Π_2 be the protocol, where also ObPrf is substituted with a random function. It is clear that Π_2 is 0-server-private. The claim follows, since the protocols Π_2 and Π_1 are computationally ε_1'-close and Π_1 and Protocol 2 are computationally ε_2'-close. □

Ogata and Kurosawa used the RSA blinded signature to construct an ObPrf, i.e., there $\mathsf{ObPrf}_{\mathsf{sk}}(x) = \mathsf{Prf}(\mathsf{BlindSign}_{\mathsf{sk}}(x))$ for a pseudo-random function Prf. However, the Ogata-Kurosawa protocol is *a priori* secure only in the random-oracle model. On the other hand, their protocol has only two messages and Server's actions are verifiable. Indeed, any two-message server-verifiable ObPrf can be converted to a blind signature scheme. Thus, the Ogata-Kurosawa construction is optimal in that sense. Freedman et al [FIPR05] proposed an alternative OPRF construction that is secure in the standard model under the Decisional Diffie-Hellman assumption. Unfortunately, their two-round solution is not *a priori* server-verifiable; that is, Client cannot test whether the obtained value is equal to $\mathsf{ObPrf}_{\mathsf{sk}}(x)$. Therefore, a malicious Server can mount undetectable denial of service attacks by deviating from Π_{ObPrf}.

If the OPRF is verifiable and the Setup phase is done correctly, then the whole protocol is verifiable; this is since Server cannot change the committed values E_i. As the Setup phase is relatively short and well-specified, its correctness can be guaranteed by non-cryptographic methods. The later does not hold for query-transfer phase as queries can arrive during a long time period.

Generally, Protocol 2 is well suited for static databases, where Server has to run frequent itemsets mining algorithm rarely, say once in a month. Otherwise, the large initial complexity over-weights its possible advantages.

4.3 On-line Computation with Subset Inclusion

As stated before, the pre-computation cost of CPIR-PISC protocol is too large even in the case of the databases of moderate size. Limiting the answers only to frequent itemsets, like in Sect. 4.2, extends the applicability of CPIR-PISC to larger databases but limits the possible queries. To answer support queries also in large databases, we give Protocol 3. In this protocol, Server does not perform any pre-computation. Instead, when Server gets Client's query as an encrypted binary vector $(c_1, \dots, c_n)$, he runs a private subset inclusion test (a version of Protocol 1 that is secure in the semi-honest model) for every row of $\mathcal{D}$, and finally returns the replies in a randomised order. As a result, Client receives $D_{\mathsf{sk}}(C_i) = 0$ for every row where $\mathcal{Q}$ was present and $D_{\mathsf{sk}}(C_i)$ is random element of $\mathcal{P}$ for every row where $\mathcal{Q}$ was not present. After that, she decrypts the results and then counts the number of nonzero values. Together with Protocol 1, the communication of this protocol (in the semi-honest model) is $(n+1+m) \cdot \mathsf{len}(|\mathcal{P}|)$ bits. (Note that here we implicitly need that $|\mathcal{P}| > n$.) No off-line work is done; Client performs $\Theta(n+m)$ and Server does $\Theta(w(\mathcal{D})+m)$ units of online computation, where $w(\mathcal{D})$ denotes the number of 1-s in the whole database. Again, in the data-mining scenarios, the database is usually very sparse and therefore $w(\mathcal{D})$ is small.

Theorem 4. *Let Π be (τ, ε) IND-CPA secure homomorphic cryptosystem. Then Protocol 3 is $(\tau - \mathcal{O}(n), n\varepsilon)$ client-private and 0-server-private in the semi-honest model.*

Proof. As Server sees only n encryptions, client-privacy follows from standard hybrid argument: any adversary A that can distinguish two vectors of ciphertexts can be used for distinguishing only two ciphertexts. Simulation of Client's view is straightforward, the simulator must send $\mathrm{supp}_{\mathcal{D}}(\mathcal{Q})$ encryptions of 0's and $m - \mathrm{supp}_{\mathcal{D}}(\mathcal{Q})$ encryptions of random elements. □

Security against malicious Clients is achieved by letting Client to prove in zero-knowledge, for every $i \in [n]$, that c_i is an encryption of either 0 or 1. This is usually feasible since in reality, $n \leq 100\,000$.

PRIVATE INPUT: Client has a set $\mathcal{Q}$ and Server has a database $\mathcal{D}$.
PRIVATE OUTPUT: Client knows $\mathsf{count} = \mathrm{supp}_{\mathcal{D}}(\mathcal{Q})$

Message 1, Client Generate $(\mathsf{sk}, \mathsf{pk}) \leftarrow G$ and send pk to Server.
For any $i \in [n]$, send $c_i \leftarrow E_{\mathsf{pk}}(\mathcal{Q}[i]; r_i)$ with $r_i \leftarrow_r \mathcal{R}$ to Server.
Message 2, Server Generate a random permutation $\pi : [m] \mapsto [m]$.
Set $d \leftarrow \prod_{i=1}^{n} c_i$.
For every transaction $j \in [m]$
Draw $s_j \leftarrow_r \mathcal{P}$ and $r'_j \leftarrow_r \mathcal{R}$ uniformly at random.
Send $C_j \leftarrow (d/\prod_{i:\mathcal{D}[\pi(j),i]=1} c_i)^{s_j} \cdot E_{\mathsf{pk}}(0; r'_j)$ to Client.
Post-processing by Client
Set $\mathsf{count} \leftarrow 0$.
For every row $j \in [m]$
If $D_{\mathsf{sk}}(C_j) = 0$ then $\mathsf{count} \leftarrow \mathsf{count} + 1$.
Return count.

Protocol 3: Protocol for PISC, based on the private inclusion test.

5 Imprecise Private Itemset Counting Protocols

In practice, it is not usually necessary to give exact supports but accurate approximations. Sometimes it is even sufficient to know whether a set is frequent or not. In this section, we consider two approaches to approximate frequency queries. First, we study protocols to decide whether a given itemset is frequent or not. That gives rise to deterministic support approximation techniques. Second, we show how random sampling can be used together with Protocol 3 to obtain support approximations with quality guarantees.

5.1 Private frequent itemset detection

The simplest approximation of the support of a frequent itemset is to tell whether or not the itemset is frequent. At the end of a *private frequent itemset detection (PFID) protocol*, Client learns whether $\mathrm{freq}_{\mathcal{D}}(\mathcal{Q}) \geq \sigma$, and nothing else. PFID is a common subtask in pattern discovery [MT97,GKM+03]. Moreover, PFID can be used as subprotocol in different approximate PISC protocols.

One straightforward solution is to use Prot. 2 on a database where one stores some fixed integer instead of the cardinality of the support. However, this does not decrease communication or computation significantly.

A more interesting alternative is to modify the database so that it contains only maximal frequent sets, i.e., frequent sets that are not subsets of other frequent sets. That is, every maximal frequent set is added to a new database $\mathcal{D}'$ as a transaction. Afterwards, Client and Sever execute a PISC protocol on $\mathcal{D}'$. If $\mathrm{supp}_{\mathcal{D}'}(\mathcal{Q}) \geq 0$, then $\mathcal{Q}$ is frequent in the original database. Since the number of maximal itemsets can be exponentially smaller than the number of frequent itemsets, this idea might give us a significant win in the practice. Note that instead of just the support of $\mathcal{Q}$, the resulting protocol also outputs both the number ℓ of maximal itemsets and the number k of maximal itemsets that contain $\mathcal{Q}$. This additional information is sometimes desired in data mining. Even if undesired, however, such a leak is not necessarily very dangerous for the privacy of the database, since in practice there are always many different alternatives that could be the collection of the ℓ maximal itemsets, k of them containing the itemset $\mathcal{Q}$. Namely, any anti-chain (i.e., a collection $\mathcal{A}$ such that $(X \subseteq Y \vee Y \subseteq X) \Rightarrow X = Y$ for all $X, Y \in \mathcal{A}$) of ℓ itemsets on the attributes of the database such that k of the itemsets contain $\mathcal{Q}$ and $\ell - k$ do not contain $\mathcal{Q}$ could be that collection of maximal itemsets.

To show that the number of such possible collections of maximal itemsets is large, we give a rough lower bound in the case the query contains less than $n/2$ elements and there is no prior information on the database content. Let $\mathcal{M}$ be a collection of all itemsets of cardinality $\lfloor n/2 \rfloor$. Let $c_{\mathcal{Q}}$ be the cardinality of $\{X \in \mathcal{M} : \mathcal{Q} \subseteq X\}$ and c cardinality of $\mathcal{M}$, i.e. $c = \binom{n}{\lfloor n/2 \rfloor}$. Now, there is exactly $\binom{c_{\mathcal{Q}}}{k}$ ways to choose maximal sets from $\mathcal{M}$ containing $\mathcal{Q}$, and $\binom{c - c_{\mathcal{Q}}}{\ell - k}$ ways to choose other $\ell - m$ elements that cannot contain $\mathcal{Q}$. Therefore, the lower bound on consistent configurations is huge,

$$\binom{c_{\mathcal{Q}}}{k}\binom{c - c_{\mathcal{Q}}}{\ell - k} = \binom{\binom{n-|\mathcal{Q}|}{\lfloor n/2 \rfloor - |\mathcal{Q}|}}{k}\binom{\binom{n}{\lfloor n/2 \rfloor} - \binom{n-|\mathcal{Q}|}{\lfloor n/2 \rfloor - |\mathcal{Q}|}}{\ell - k}.$$

Furthermore, the exact numbers of maximal itemsets can be hidden by adding adding some subsets of the maximal itemsets to the database. This does not change the outcome of the protocol since if an itemset is contained in some subset of a maximal itemset, then it is obviously contained also in the maximal itemset itself. Note that also the supports of the itemsets leak information about other itemsets and the underlying database since each acquired support further restricts the collection of databases compatible with the known supports [Mie03].

A PFID protocol can be used to answer exact and approximate support queries. Let $0 < \sigma_1 < \cdots < \sigma_k \leq 1$ be the minimum frequency thresholds for which the frequent itemset detection problem can be solved. Then Client can find out to which of the intervals $(0, m\sigma_1], \ldots, (m\sigma_k, m]$ the support of $\mathcal{Q}$ belongs. Note that there are at most m different possible supports and in practice the number is often strictly smaller than that.

5.2 Sampling

A randomly chosen subset of the transactions provides usually a very good summary of the database. This is true also in the case of frequency estimation. Recall that the frequency of an itemset in a transaction database is the fraction of the transactions containing that itemset. Thus, the frequency of an itemset can be estimated from a set of randomly chosen transactions. Furthermore, the relative and absolute errors of such an estimation can be bounded by the Chernoff bound and the Hoeffding bound, respectively. (Note that Protocol 3 can be applied as well to a randomly chosen subset of transactions of the database as to the database itself.)

Let us assume for a moment that we know that the frequency of the itemset is at least σ. In that case the Chernoff bound gives us the following bound for the relative error ε:

Theorem 5. *Let $\mathcal{S}$ be a random k-row sample of an n-row database $\mathcal{D}$ with replacement. Then* $\Pr\left[(1-\varepsilon)\operatorname{freq}_{\mathcal{D}}(\mathcal{Q}) \le \operatorname{freq}_{\mathcal{S}}(\mathcal{Q}) \le (1+\varepsilon)\operatorname{freq}_{\mathcal{D}}(\mathcal{Q})\right] \le 2\exp\left(-\varepsilon^2 k\sigma/4\right)$ *, when the itemset is frequent, i.e.* $\operatorname{freq}_{\mathcal{D}}(\mathcal{Q}) \ge \sigma$.

Proof. Let $I_{\mathcal{Q}}(\mathcal{S}[i])$ be $I_{\mathcal{Q}}$ be an indicator variable, i.e. $I_{\mathcal{Q}}(\mathcal{S}[i]) = 1$ if $\mathcal{Q} \subseteq \mathcal{S}[i]$ and 0 otherwise. Then the frequency estimator $\operatorname{freq}_{\mathcal{S}}(\mathcal{Q}) = \sum_{i=1}^{k} I_{\mathcal{Q}}(\mathcal{S}[i])/k$ is a random variable with the expectation $\mathbf{E}(\operatorname{freq}_{\mathcal{S}}(\mathcal{Q})) = \operatorname{freq}_{\mathcal{D}}(\mathcal{Q})$. As $\operatorname{freq}_{\mathcal{S}}(\mathcal{Q})$ is a sum of k i.i.d. random zero-one variables and $\operatorname{freq}_{\mathcal{D}}(\mathcal{Q}) \ge \sigma$ by assumption, we can apply the Chernoff bound and underestimate $\operatorname{freq}_{\mathcal{D}}(\mathcal{Q})$ by σ. This proves the claim. $\square$

Theorem 5 allows us to estimate the sufficient number of transactions to bound the relative error ε and the failure probability δ:

Corollary 2. *To guarantee that the relative error is at most ε and the failure probability is at most δ, it is sufficient to randomly choose $k \ge 4(\ln 2/\delta)/(\varepsilon^2\sigma)$ transactions from the database.*

Moreover, if we are interested in bounding the absolute error, the number of rows sufficient to guarantee a certain error bound is even smaller without assuming anything about the frequencies:

Theorem 6. *Let $\mathcal{S}$ be a random k-row sample of an n-row database $\mathcal{D}$ with replacement. Then* $\Pr\left[|\operatorname{freq}_{\mathcal{S}}(\mathcal{Q}) - \operatorname{freq}_{\mathcal{D}}(\mathcal{Q})| \ge \varepsilon\right] \le 2\exp\left(-2\varepsilon^2 k\right)$.

Proof. As $\operatorname{freq}_{\mathcal{S}}(\mathcal{Q})$ is a sum of k i.i.d. random zero-one variables, we can apply the Hoeffding bound which proves the claim. $\square$

The number of transaction sufficient in this case is as follows:

Corollary 3. *To guarantee that the relative error is at most ε and the failure probability is at most δ, it is sufficient to randomly choose $k \ge (\ln 2/\delta)/(2\varepsilon^2)$ transactions from the database.*

For example, with failure probability 10^{-3} and absolute error $1/100$, it is sufficient to have $38\,500$ rows in the sample and thus PSI-PISC protocol is efficient enough. Hence, the sampling technique provides an approximation protocol such that the computation and communication complexity are independent from database size. The complexity depends only on the desired approximation error ε and the failure probability δ.

The approximation error bounds given above can be used also to obtain approximations for the frequent itemset detection, i.e., *property testers* for itemsets being frequent. More specifically, we can use the above bounds to decide correctly with high probability whether the itemset is frequent or not when the correct frequency of the itemset is above or below the minimum frequency threshold σ at least by error ε; if the correct frequency is within the error ε from the minimum frequency threshold σ, we might answer incorrectly with non-negligible probability.

Chernoff and Hoeffding bounds are quite tight when estimating the frequency of a single itemset from one sample databases, i.e. we re-sample the database before each query. This is not the case when several frequencies are estimated, i.e., when Server generates a single database by sampling the transactions to answer all or many Client's queries. In this case, Server might even verify the approximation precision of all frequent itemsets $\mathcal{F}$ (or some other collection of itemsets of interest).

The straightforward generalisation of of the Hoeffding bound to a collection $\mathcal{F}$ of itemsets with maximum absolute error guarantee ε for an arbitrary sequence of frequency queries to $\mathcal{F}$ yields to sample complexity of $(\ln 2\,|\mathcal{F}|\,/\delta)/(2\varepsilon^2)$ [Toi96]. However, this is a worst-case bound. In practice, transaction databases are not as difficult to sample as possible but have a lot structure. One important measure of complexity of a transaction database is its Vapnik-Chervonenkis dimension (VC-dimension) w.r.t. the itemsets whose frequencies want to be able to estimate accurately. If the VC-dimension is k, then the number of transactions is sufficient to guarantee maximum absolute error ε with probability $1-\delta$ is $\mathcal{O}(k \ln 1/(\delta\varepsilon)/\varepsilon^2)$.

For example, the VC-dimension of the database w.r.t. frequent itemsets can be bounded above by $\log|\mathcal{C}|$, where $\mathcal{C}$ is the collection of closed frequent itemsets in the database [Mie04]. (An itemset is closed in collection $\mathcal{F}$ if its support is strictly higher than the largest support of its supersets.) Using this upper bound for the Vapnik-Chervonenkis dimension, it is possible to show that a sample of $\mathcal{O}(\ln|\mathcal{C}| \ln 1/(\delta\varepsilon))/\varepsilon^2$ transactions suffices to guarantee with failure probability δ that the maximum absolute error is at most ε. As the number of closed frequent itemsets can be exponentially smaller than the number of frequent itemsets, the VC-based sample bound can be $\mathcal{O}(\ln\ln|\mathcal{F}| \ln 1/(\delta\varepsilon))/\varepsilon^2$ at smallest. In practice, one can compute both the Hoeffding bound and the VC-bound, and take the minimum of them. Also, if the collection of the itemsets for which the approximation guarantees are required is small enough, then one can maintain the frequency estimates for all itemsets of interest and stop the sampling immediately when the frequency estimates are sufficiently accurate.

Acknowledgements. We would like to thank Bart Goethals for proposing this problem and for many useful discussions and comments, and Aggelos Kiayias and Antonina Mitrofanova for comments. The first author was partially supported by the Finnish Academy of Sciences. The second author was partially supported by the Estonian Science Foundation, grant 6096.

References

[AIR01] William Aiello, Yuval Ishai, and Omer Reingold. Priced Oblivious Transfer: How to Sell Digital Goods. In Birgit Pfitzmann, editor, *Advances in Cryptology — EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 119–135, Innsbruck, Austria, May 6–10, 2001. Springer-Verlag.

[AMS+96] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and A. Inkeri Verkamo. Fast Discovery of Association Rules. In Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI/MIT Press, 1996.

[BGZ04] Roberto J. Bayardo Jr., Bart Goethals, and Mohammed Javeed Zaki, editors. *FIMI '04, Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations*, volume 126 of *CEUR Workshop Proceedings*, Brighton, UK, November 1, 2004.

[CGN97] Benny Chor, Niv Gilboa, and Moni Naor. Private information retrieval by keywords. Technical Report TR CS0917, Department of Computer Science, Technion, 1997.

[FIPR05] Michael J. Freedman, Yuval Ishai, Benny Pinkas, and Omer Reingold. Keyword Search and Oblivious Pseudorandom Functions. In Joe Kilian, editor, *The Second Theory of Cryptography Conference, TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 303–324, Cambridge, MA, USA, February 10–12, 2005. Springer Verlag.

[FNP04] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient Private Matching and Set Intersection. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology — EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 1–19, Interlaken, Switzerland, May 2–6, 2004. Springer-Verlag.

[GKM+03] Dimitrios Gunopulos, Roni Khardon, Heikki Mannila, Sanjeev Saluja, Hannu Toivonen, and Ram Sewak Sharma. Discovering All Most Specific Sentences. *ACM Transactions on Database Systems*, 28(2):140–174, 2003.

[GR05] Craig Gentry and Zulfikar Ramzan. Single-Database Private Information Retrieval with Constant Communication Rate. In Luis Caires, Guiseppe F. Italiano, Luis Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *The 32nd International Colloquium on Automata, Languages and Programming, ICALP 2005*, volume 3580 of *Lecture Notes in Computer Science*, pages 803–815, Lisboa, Portugal, 2005. Springer-Verlag.

[Lip03] Helger Lipmaa. Verifiable Homomorphic Oblivious Transfer and Private Equality Test. In Chi Sung Laih, editor, *Advances on Cryptology — ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 416–433, Taipei, Taiwan, November 30–December 4, 2003. Springer-Verlag.

[Lip05] Helger Lipmaa. An Oblivious Transfer Protocol with Log-Squared Communication. In Jianying Zhou and Javier Lopez, editors, *The 8th Information Security Conference (ISC'05)*, volume 3650 of *Lecture Notes in Computer Science*, pages ?–?, Singapore, September 20–23, 2005. Springer-Verlag. To appear.

[Mie03] Taneli Mielikäinen. On Inverse Frequent Set Mining. In *2nd Workshop on Privacy Preserving Data Mining (PPDM)*, pages 18–23, Melbourne, Florida, USA, November 19, 2003. IEEE Computer Society.

[Mie04] Taneli Mielikäinen. Separating Structure from Interestingness. In Honghua Dai, Ramakrishnan Srikant, and Chengqi Zhang, editors, *Advances in Knowledge Discovery and Data Mining, 8th Pacific-Asia Conference, PAKDD 2004*, volume 3056 of *Lecture Notes in Computer Science*, pages 476–485, Sydney, Australia, May 24–28, 2004. Springer.

[MT97] Heikki Mannila and Hannu Toivonen. Levelwise Search and Borders of Theories in Knowledge Discovery. *Data Mining and Knowledge Discovery*, 1(3):241–258, 1997.

[OK04] Wakaha Ogata and Kaoru Kurosawa. Oblivious Keyword Search. *Journal of Complexity*, 20(2–3):356–371, 2004.

[Toi96] Hannu Toivonen. Sampling Large Databases for Association Rules. In T. M. Vijayaraman, Alejandro P. Buchmann, C. Mohan, and Nandlal L. Sarda, editors, *VLDB'96, Proceedings of 22th International Conference on Very Large Data Bases*, pages 134–145, Mumbai, India, September 3–6, 1996. Morgan Kaufmann.