Deterministic K-Set Structure

Sumit Ganguly and Anirban Majumder
{sganguly,anirban}@cse.iitk.ac.in

IIT Kanpur, India

Abstract. [1] A k-set structure is a sub-linear space data structure that supports multi-set insertion and deletion operations and returns the multi-set, provided the number of distinct items with non-zero frequency does not exceed k. This is a fundamental problem with applications in data streaming [16], distributed systems [15, 17], etc. In this paper, we present the design of a deterministic k-set structure.

1 Introduction

Consider scenarios where entities with identity arrive and depart in a critical zone, for example, persons with RF-tags, TCP connections to a given site, etc.. The problem is to efficiently answer the following query: " Are there at most k distinct entities in the critical zone, and if so, what are their identities?" Clearly, if there is enough memory to track all the entities, then, an $O(n)$ space solution is obvious. The problem can be effectively solved using a k-set data structure, which is a sub-linear space data structure that (a) supports insertions and deletions of items in a multi-set, and, (b) supports a *Retrieve* operation that returns all the distinct items and their number of occurrences in the multi-set, provided, the number of distinct items is at most k. Applications of k-set structure arise in diverse areas, including data stream applications and distributed computing [15, 17]. In a distributed computing scenario, a host and a PDA may temporarily disconnect and proceed asynchronously with their computations. Later a reconciliation mechanism is needed to synchronize a specific collection of bits between the two hosts. A k-set structure can be used to give a solution requiring low communication. Analogously, in a distributed computing environment, a k-set structure can be used for reconciling changes to shared structures, such as files, transaction logs, etc., with the minimum communication necessary.

We will define a k-set structure in the data stream model of computation. A data stream σ is viewed as a sequence of records of the form (pos, i, v), where, i is the identity of the data item that is assumed to belong to the domain $\{1, 2, \ldots, n\}$ and v is the change in the frequency of i. We will abbreviate the set $\{1, 2, \ldots, n\}$ by $[1, n]$. For simplicity, we assume that v is integral, where, a positive value of v corresponds to v insertions of i, and a negative value of v corresponds to v deletions of i. The frequency $f_i(\sigma)$ of an item i is defined as the sum of the changes in the frequencies of i, that is, $f_i(\sigma) = \sum_{(pos,i,v)\in\sigma} v$. At any given time, the multi-set corresponding to the stream is defined as $\{(i, f_i(\sigma)) \mid f_i(\sigma) \neq 0\}$. When the data stream σ is understood, then, we will refer to the frequency of an item i in the

[1] Preliminary version [12] of this paper appeared with the same title in the *Proceedings of the ACM SIGACT-SIGMOD Symposium on Principles of Database Systems*, June 2006.

stream simply as f_i. Data streams that allow insertions and deletions of items and allow item frequencies to be either positive or negative are referred to as *general* streams. *Strict* streams refer to the sub-class of general streams where deletions and insertions are allowed, although, item frequencies are constrained to be non-negative. Both models are popular abstractions of diverse families of computations and are well-studied in the research literature. The size of a stream σ is defined as

$$|\sigma| = \max_{\text{prefixes } \sigma' \text{ of } \sigma} \quad \max_{i=1}^{n} |f_i(\sigma')| = \max_{\text{prefixes } \sigma' \text{ of } \sigma} \|f(\sigma')\|_\infty \ .$$

The expressions for space and time requirement of the algorithms use the parameters m and n, where, m is an upper bound on $|\sigma|$ and n is the size of the domain of the items. We now define the k-set structure in two variants, respectively called *strong* and *weak* k-sets.

Definition 1. *A strong k-set structure over a data stream is a data structure that supports the following three operations, (a) procedure* Update*, for updating the data structure corresponding to stream insertion and deletion operations, (b) procedure* Retrieve*, that returns the multi-set $S = \{(i, f_i) \mid i \in [1, n] \text{ and } f_i \neq 0\}$, provided, $|S| \leq k$, and, (c) procedure* IsCard *(Is Cardinality at most k?), that returns* TRUE *if $|S| \leq k$ and returns* FALSE *otherwise. A weak k-set structure only supports the procedures* Update *and* Retrieve*.* □

A space lower bound argument for a k-set is obtained as follows. There are $\binom{n}{k}(2m)^k$ possible multi-sets of size k over the domain $\{1, 2, \ldots, n\}$ such that $|f_i| \leq m$ and $f_i \neq 0$. Each such multi-set must map to a distinct memory pattern of a deterministic algorithm (otherwise, the algorithm makes an error in at least one of the inputs). Therefore, a deterministic k-set structure requires space $\Omega(\log(\binom{n}{k}(2m)^k)) = \Omega(k(\log \frac{n}{k} + \log m))$ bits of space. We are interested in obtaining designs that approach this lower bound.

Previous work. A randomized strong k-set structure is a structure that uses random bits in the execution of its procedures Update, Retrieve and IsCard, and whose answers are correct with high probability. The COUNTSKETCH [3] and the COUNT-MIN [5] algorithm can both be used to design a randomized strong k-set for general streams using space $O(k(\log(mn))(\log \frac{n}{\delta}))$ bits, where, δ is the error probability of the Retrieve or IsCard operation. For strict streams, the randomized k-set structure [10] uses $O(k(\log(mn))(\log \frac{k}{\delta}))$ bits. The work in [16] (Theorem 15) and [11] presents a deterministic weak k-set structure using $O(k^2(\log^2 n)(\log m)(\log^2 k))$ space. A combinatorial group-testing based approach for designing a weak k-set structure is presented in [6] that uses space $O(k^2(\log^2 n)(\log m))$ bits. A special case of the weak k-set problem, where, item frequencies are ± 1, has been studied in distributed systems, where it is called set reconciliation, and a nearly-optimal $O(k \log n)$ space solution is presented [15, 17]. The work in [9] presents a nearly optimal space solution for the weak k-set problem when item frequencies are exactly 1 or 0. *Compressed Sensing.* The works of [2, 8] together with the recent work of [14] presents a deterministic and weak k-structure using space $O(k 2^{(\log\log n)^E}(\log m))$ for general streams and is based on a construction using extractor graphs.

For strict streams, a strong 1-set is presented in [10] using nearly optimal space. Strongly selective sets [4, 7], together with a strong 1-set, can be used to construct strong k-sets using space $O(k^2 \cdot \text{polylog}(n) \log(mn))$. This line of work however cannot be used to obtain significantly more space efficient k-sparsity tests, since, there is a space lower bound of $\Omega(k^2(\log(n/k))/(\log k))$ for the size of (n, k)-strongly selective family [4].

Contributions. We present a near-optimal space construction for weak k-sets using $O(k \log(mn))$ bits for streams over $[1, n]$ and size at most m. We show that a strong k-set structure requires $\Omega(n)$ bits, for any $k \geq 0$. For strict streams over $[1, n]$ and having size at most m, we present a strong k-set structure that uses space $O(k^2 \log n + k \log m)$ bits. These are the most space-efficient weak and strong k-set structures respectively. Moreover, the structures admit efficient update processing and retrieval operations.

Comparison with compressed sensing algorithms [2, 14]. For general streams, the compressed method gives asymptotically more efficient retrieval of items as compared to the method of this paper. However, our method is simpler and more elementary, is more space efficient and requires less time for processing stream updates. Further, for strict streams, a variant of our algorithm is significantly faster than the algorithm of [14].

2 *K*-set structure

In this section, we present our design of a k-set structure.

Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, where, p will be specified later, and using the arithmetic over $\mathbb{F}_p$, we maintain the following $2k + 2$ counters, denoted by $l_0, l_1, \ldots, l_{2k+1}$.

$$l_r = \sum_{x_i \in \text{ stream}} f_i x_i^r, \quad r = 0, 1, \ldots, 2k + 1 \ . \tag{1}$$

The counters can be easily updated in the face of insertions and deletions occurring in the stream as follows. Corresponding to a stream update (pos, x_i, v), we update the r^{th} counter as follows, using the arithmetic of $\mathbb{F}$.

$$l_r := l_r + v \cdot x_i^r, \quad \text{for } r = 0, 1, \ldots, 2k + 1 \ .$$

We use the following notation in this section. Given t distinct items $x_1, x_2, \ldots, x_t$, each of which lies in the interval $1 \leq x_i \leq t$, we let $X = X(t)$ denote the $t \times t$ diagonal matrix that has x_i in its i^{th} diagonal entry and zeros elsewhere. Given a set of t frequency values, $f_1, f_2, \ldots, f_t$, we let F denote the diagonal matrix whose i^{th} diagonal entry is f_i and is zero elsewhere. Let f be the column vector $[f_1, f_2, \ldots, f_t]^T$. That is,

$$X = \begin{bmatrix} x_1 & & & \\ & x_2 & & \\ & & \ddots & \\ & & & x_n \end{bmatrix} \qquad F = \begin{bmatrix} f_1 & & & \\ & f_2 & & \\ & & \ddots & \\ & & & f_t \end{bmatrix} \qquad \text{and } f = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_t \end{bmatrix} \ .$$

For $1 \le t \le k$ and $0 \le r \le 2k - t$, let $V(r,t)$ denote the $t \times t$ matrix shown below. For a given set of values $x_1, x_2, \ldots, x_t$, for brevity, we refer to $V(0,t)$ as V, as follows.

$$V(r,t) = \begin{bmatrix} x_1^r & x_2^r & \cdots & x_t^r \\ x_1^{r+1} & x_2^{r+1} & \cdots & x_t^{r+1} \\ & \vdots & \vdots & \\ x_1^{r+t-1} & x_2^{r+t-1} & \cdots & x_t^{r+t-1} \end{bmatrix} \qquad V = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_t \\ x_1^2 & x_2^2 & \cdots & x_t^2 \\ & \vdots & \vdots & \\ x_1^{t-1} & x_2^{t-1} & \cdots & x_t^{t-1} \end{bmatrix}$$

The following identity is a direct consequence of the definition.

$$V(r,n) = VX^r \tag{2}$$

Let $w(s,r)$, B_r and C_r denote the following $r \times 1$ column matrix and $r \times r$ square matrices respectively.

$$w(s,r) = \begin{bmatrix} l_s \\ l_{s+1} \\ \cdots \\ l_{r+s-1} \end{bmatrix}, \quad B_r = \begin{bmatrix} l_0 & l_1 & \ldots & l_{r-1} \\ l_1 & l_2 & \ldots & l_r \\ \vdots & \vdots & \vdots & \vdots \\ l_{r-1} & l_r & \ldots & l_{2r-2} \end{bmatrix}, \text{ and } \quad C_r = \begin{bmatrix} l_1 & l_2 & \ldots & l_r \\ l_2 & l_3 & \ldots & l_{r+1} \\ \vdots & \vdots & \vdots & \vdots \\ l_r & l_{r+1} & \ldots & l_{2r-1} \end{bmatrix} . \tag{3}$$

Lemma 1. *Suppose that there are $t \le k$ items in the stream with non-zero frequency and the arithmetic in (1) is performed over $\mathbb{F}_p$, where $p > 2mn + 1$. Then, (a) rank$(B_{k+1}) = t$, (b) the items $x_1, x_2, \ldots, x_t$ are the eigenvalues of the matrix $B_t^{-1}C_t$ and, (c) the frequency vector is given by $f = V^{-1}w(0,t)$.*

Proof. Suppose there are $t \le k$ distinct items $x_1, x_2, \ldots, x_t$ in the stream, $x_i \in [1,n]$. Let $V(r) = V(r,t)$ and $w(r) = w(r,t)$, for $r = 0, 1, 2, \ldots, 2k + 1 - t$. Thus, equation (1) can be rewritten as follows.

$$V(r)f = VX^r f = w(r), \quad r = 0, \ldots, 2k + 1 - t \ .$$

Since, the x_i's are non-zero and distinct, $V(r) = VX^r$ is invertible for each value of $0 \le r \le 2k + 1 - t$. Therefore,

$$VXV^{-1}w(r) = VX^{r+1}((VX^r)^{-1}w(r)) = VX^{r+1}f = w(r+1), \ \ 0 \le r \le 2k + 1 - t \ .$$

Let A denote the matrix VXV^{-1}. The above set of equations can be expressed as

$$AB_t = C_t \tag{4}$$

Since A is in the eigen-decomposition form, X is the diagonal eigenvalue matrix of A. In other words, the distinct items $x_1, \ldots, x_t$ are the eigenvalues of the matrix A'. Further,

$$\begin{aligned} B_t &= [w(0), w(1), \ldots, w(t-1)] = [Vf, VXf, VX^2f, \ldots, VX^{t-1}f] \\ &= V[f, \ Xf, \ X^2f, \ \ldots \ , X^{t-1}f] = VFV^T \end{aligned} \tag{5}$$

Since, V is invertible and none of the f_i's are 0, B_t is invertible, and therefore has rank t.

Since B_t is the left $t \times t$ sub-matrix of B_{k+1}, $\text{rank}(B_{k+1}) \geq \text{rank}(B_t) = t$. Let U denote the $(k+1) \times t$ matrix $V(0, k+1)$ and let $U(j)$ be the $(k+1) \times t$ matrix $V(j, k+1)$, for $0 \leq j \leq k$. Therefore, (1) can be equivalently written as: $U(j)f = UX^j = w(j, k+1)$, for $j = 0, \ldots, k$. Thus,

$$B_{k+1} = [w(0, k+1), w(1, k+1), \ldots, w(k, k+1)] = U[f, Xf, \ldots, X^k f] = UFU^T \ .$$

Since, U and F each have rank t, it follows that $\text{rank}(B_{k+1}) \leq t$. As shown earlier, $\text{rank}(B_{k+1}) \geq t$. Therefore, $\text{rank}(B_{k+1}) = t$. $\square$

Given that the number of items with non-zero frequency in the stream is $t \leq k$, then, t can be found as $\text{rank}(B_{k+1})$ and the items themselves are the eigenvalues of $A_t = C_t B_t^{-1}$. The rank computation can be done using $O(k^3)$ arithmetic operations over the field $\mathbb{F}_p$, and the eigenvalues can be found in time $O(nk^3)$, by iterating over $x \in [1, n]$ and testing if $A_t - xI$ is singular. This is stated in Lemma 2.

Lemma 2 (Weak k-set for general data streams). *For general streams over the domain $[1, n]$ and size a-priori bounded by m, a weak k-set structure can be designed with the following characteristics: (a) space $O(k(\log(mn))$ bits, (b) time for update– $O(k)$ operations over a finite field $\mathbb{F}_p$ ($p = O(mn)$), and, (c) time for retrieving the elements– $O(nk^3)$ operations over $\mathbb{F}_p$.* $\square$

If item frequencies are non-negative, then, the standard dyadic intervals technique can be utilized as follows to retrieve the items more efficiently and without requiring the explicit computation of eigenvalues.

Finding roots of the characteristic polynomial. We now assume that item frequencies cannot assume negative values. Since there are n eigenvalues, the characteristic polynomial $F(z)$ is of the form $F(z) = \alpha \prod_{a \in S}(z - a)$, where, S is the set of items in the stream with non-zero frequency and α is a constant. Let $\mathbb{F}$ be a field with characteristic at least $2km$ and containing at least $n + 1$ elements.

Instead of maintaining a single set of $2k + 2$ counters, we maintain a collection of $L = \lceil \log |\mathbb{F}_p| / k \rceil$ sets of counters, where each set consists of $2k + 2$ counters. We set p to be a prime greater than $2mn^2 + 1$, since, the maximum absolute frequency of any dyadic interval can be at most mn, and, there are n distinct items. The s^{th} counter set is denoted as $\{l_r^s\}_{r=0,1,\ldots,2k+1}$, for $0 \leq s \leq L - 1$. For $0 \leq s \leq L - 1$, define a family of functions $h_s : \{0, 1, \ldots, 2^d - 1\} \to \{0, 1, \ldots, 2^{d-s} - 1\}$ as follows.

$$h_0(a) = a \qquad \text{and} \qquad h_s(a) = \lfloor a/2^s \rfloor$$

It follows that, for any $s \geq 0$ and given value of $c = h_s(a)$, there are exactly two distinct values of b such that $b = h_{s-1}(a)$. Corresponding to each stream update of the form (x, v), we update each of the s counter sets as follows.

$$l_r^s = l_r^s + (h(x))^r v, \quad 0 \leq r \leq 2k + 1, 0 \leq s \leq L - 1 \ .$$

Let $f_a^{(s)}$ denote the frequency of item a at level s. Then, $f_a^{(s)} = \sum_{b:h_s(b)=a} f_b$. If a has positive frequency $f_a > 0$, then, $f_{h_s(a)}^{(s)} > 0$ has positive frequency at level s, for $1 \le s \le L-1$ (vice-versa may not be true). Let n_s denote the number of distinct items with positive frequencies at level s. Let $A_{n_s}^{(s)}$, $B_{n_s}^{(s)}$ and $C_{n_s}^{(s)}$ respectively denote the corresponding matrices obtained from the counters at level s, for $s = 0, 1, \ldots, L-1$. Let $F_s(z)$ denote the characteristic polynomial of $A_n^{(s)}$, that is, $F_s(z) = \det(A_{n_s}^{(s)} - zI)$. By construction, we have the following property

$$F(a) = 0 \text{ only if } F_s(h_s(a)) = 0 \ .$$

For each value of s starting from L and counting down to 0, we obtain a set of items of size at most $2 \cdot k$ that are potentially roots of $F_s(z)$. At level L, there are at most k distinct items, each of which are then checked to see if $F_s(a_s) = 0$ (or equivalently, if $A_n^{(s)} - a_s I$ is singular). Therefore, at each level, there cannot be more than k candidates that pass the above test. Each candidate item a at level s corresponds to exactly 2 candidates at level $s-1$; therefore, the number of potential candidates at any level is no more than $2k$. Proceeding in this manner, we obtain the set of items with positive frequencies at level 0.

The data structure maintains $(2k+2)$ counters for at most $\lceil \log |\mathbb{F}|/k \rceil$ levels. Therefore, its space requirement is $O(k(\log \frac{n}{k}))(\log(kmn)))$ bits. Testing whether an item x is an eigenvalue can be done by calculating the rank of $A - xI$, which can be done in time $O(k^3)$. Since there are at most $2k$ candidate items at each level the time complexity of retrieval is $O(k^4(\log(m + \frac{n}{k})))$ field operations. We summarize this discussion in the following lemma.

Lemma 3 (Improved weak k-set for strict data streams). *For strict streams over the domain $[1, n]$ and size a-priori bounded by m, a weak k-set structure can be designed with the following characteristics: (a) space $O(k(\log(n/k))(\log(kmn)))$, (b) time required to retrieve the items is $O(k^4(\log(kmn)))$ operations over a prime field $\mathbb{F}_p$ of size $O(kmn)$, and, (c) time for processing each stream update is $O(k(\log(n/k))$ operations over $\mathbb{F}_p$.* □

3 Strong k-set structure

In this section, we present a design of a strong k-set structure. We first show that a strong k-set structure over general streams requires $\Omega(n)$ bits, for any $k \ge 1$.

Lemma 4. *For general streams and $0 \le k \le n$, a strong k-set structure requires $\Omega(n)$ bits.*

Proof. Let $k = 0$. Let F be a family of sets from $\{1, 2, \ldots, n\}$ of size $\frac{n}{2}$ such that the size of pair-wise intersection is at most $\frac{n}{8}$. Using simple counting techniques, it is easy to show that there exist such families of size $2^{\Omega(n)}$ [1]. Let $S, T \in F$. Construct two streams from S and T respectively where the item frequency is 1 for each element in the corresponding set. Consider the memory patterns of a k-set structure after processing the streams independently. We claim that the two memory patterns must be different for the following reason. Consider a sequence of deletions of all $\frac{n}{2}$ items from S. Since, S and T have at most $\frac{n}{8}$ items in common, the same sequence of deletions applied to T leaves T with at least $\frac{7n}{8}$ distinct items with non-zero frequencies. If the memory patterns of the strong k-set structure are the same after processing the streams corresponding to S and T, then, the k-set structure must make a mistake in answering whether the cardinality of the set of items with non-zero frequency is at most 0 or not. It follows that the space required by a strong k-set structure, for $k = 0$ is at least $\Omega(\log|F|) = \Omega(n)$.

A strong k-set structure implies a strong k'-set structure, for any $k' < k$, since, one can test whether the number of items in the multi-set is at most k, retrieve the elements one by one, checking and stopping if the number of items exceeds k'. Since, a strong 0-set structure requires $\Omega(n)$ bits, for every $k \geq 0$, a strong k-set structure requires $\Omega(n)$ bits. □

Lemma 5 can be used to design a strong k-set for strict streams using real arithmetic.

Lemma 5. *Suppose that real arithmetic is used to maintain the counters defined by* (1). *Then, for strict data streams there are $t > k$ distinct items in the stream with positive frequencies if and only if* rank$(B_{k+1}) = k + 1$.

Proof. The if part is the statement of Lemma 1(a). Suppose there are $t > k$ distinct items with positive frequencies. Let G denote the diagonal matrix with $G_{i,i}$ set to the positive square root of f_i. Therefore,

$$B_t = V_t F V_t^T = V_t G^2 V_t^T = (V_t G)(V_t G)^T \quad .$$

It follows that B_t is a positive definite matrix, and hence, all left most determinants of B_t are positive. Since, $t > k$, in particular, $\det(B_{k+1}) > 0$, and therefore, $\text{rank}(B_{k+1}) = k + 1$. □

Lemma 5 can be used to design a procedure to test whether the number of distinct items with non-zero frequencies is at least k or not, by computing the rank of B_{k+1}. Standard methods for finding the rank of B_{k+1}, such as Gaussian elimination and QR decomposition, may require $O(k \log n + \log m)$ bits of precision, since, the matrices B and C are derived from van der Monde matrices and can therefore be shown to have condition number $O(n^k m)$ (a well-known fact of numerical linear algebra). The counters l_j, $j = 1, 2, \ldots, r$ are themselves stored as large integers, with $\log|l_j| \leq \log(n^j m) = j \log n + \log m$. The total storage is therefore $O(k^2 \log n + k \log m)$ bits. A more space-efficient strong k-set structure may be found in a very recent work [13].

References

1. Noga Alon, Yossi Matias, and Mario Szegedy. "The space complexity of approximating frequency moments". *Journal of Computer Systems and Sciences*, 58(1):137–147, 1998.
2. Emmanuel Candès, Justin Romberg, and Terence Tao. "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information". *IEEE Trans. Inf. Theory*, 52(2):489–509, February 2006.
3. Moses Charikar, Kevin Chen, and Martin Farach-Colton. "Finding frequent items in data streams". In *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP), 2002*, pages 693–703.
4. Bogdan Cheblus and Dariusz R. Kowalski. "Almost Optimal Explicit Selectors". In *Foundations of Computing Theory (FCT) LNCS 3623*, pages 270–280, 2005.

5. Graham Cormode and S. Muthukrishnan. "An Improved Data Stream Summary: The Count-Min Sketch and its Applications". *Journal of Algorithms*, 55(1).
6. Graham Cormode and S. Muthukrishnan. "Combinatorial Algorithms for Compressed Sensing". In *Proceedings of the Thirteenth Colloquium on Structural Information and Communication Complexity (SIROCCO)*, 2006.
7. A. De Bonis, L. Gàsieniec, and U. Vaccaro. "Generalized framework for selectors with applications in optimal group testing". In *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP)*, pages 81–96, 2003.
8. David Donoho. "Compressed sensing". *IEEE Transactions on Information Theory*, 52(4):1289–1306, April 2006.
9. David Eppstein and Michael T. Goodrich. "Space-Efficient Straggler Identification in Round-Trip Data Streams via Newtons Identities and Invertible Bloom Filters". In *Proceedings of Tenth Workshop on Algorithms and Data Structures (WADS)*, 2007.
10. S. Ganguly. "Counting distinct items over update streams". In *Proceedings of the Sixteenth International Symposium on Algorithms and Computation (ISAAC)*, pages 505–514, 2005.
11. S. Ganguly and Majumder A. "CR-precis: A Deterministic Summary Structure for Update Streams". In *Proceedings of the International Symposium on Algorithms, Probabilistic and Experimental Methodologies (ESCAPE), LNCS 4614*, 2007.
12. S. Ganguly and A. Majumder. "Deterministic K-set Structure". In *Proceedings of Twentyfifth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 280–289, 2006. Detailed version available from www.cse.iitk.ac.in/users/sganguly.
13. Sumit Ganguly. "Determinstic tests for vector sparsity". Manuscript available from http://www.cse.iitk.ac.in/users/sganguly, February 2008.
14. Piotr Indyk. "Explicit Constructions for Compressed Sensing of Sparse Signals". In *Proceedings of ACM Symposium on Discrete Algorithms (SODA)*, 2008.
15. Y. Minsky, A. Trachtenberg, and R. Zippel. "Set Reconciliation with Nearly Optimal Communication Complexity". *IEEE Transactions on Information Theory*, 49(9):2213–2218, 2003.
16. S. Muthukrishnan. *"Data Streams: Algorithms and Applications"*. Foundations and Trends in Theoretical Computer Science, *Vol. 1, Issue 2*, 2005.
17. D. Starobinski, A. Trachtenberg, and S. Agarwal. "Efficient PDA synchronization". *IEEE Trans. on Mob. Comp.*, 2(1):40–51, 2003.