Data Streams: Algorithms and Applications

Data Streams: Algorithms and Applications

S. Muthukrishnan

Rutgers University
New Brunswick
NJ, USA
muthu@cs.rutgers.edu

now
the essence of knowledge

Foundations and Trends® in Theoretical Computer Science

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
USA
Tel. +1 781 871 0245
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

A Cataloging-in-Publication record is available from the Library of Congress

Printed on acid-free paper

ISBN: 1-933019-14-X; ISSNs: Paper version 1551-305X; Electronic version 1551-3068

Contents

1

Introduction

We study the emerging area of algorithms for processing data streams and associated applications, as an applied algorithms research agenda. We begin with three puzzles.

1.1 Puzzle 1: Finding Missing Numbers

Let π be a permutation of $\{1,\ldots,n\}$. Further, let π_{-1} be π with one element missing. Paul shows Carole $\pi_{-1}[i]$ in increasing order i. Carole's task is to determine the missing integer. It is trivial to do the task if Carole can memorize all the numbers she has seen thus far (formally, she has an n-bit vector), but if n is large, this is impractical. Let us assume she has only a few – say $O(\log n)$ – bits of memory. Nevertheless, Carole must determine the missing integer.

This starter puzzle has a simple solution: Carole stores

$$s = \frac{n(n+1)}{2} - \sum_{j \leq i} \pi_{-1}[j],$$

which is the missing integer in the end. Each input integer entails one subtraction. The total number of bits stored is no more than $2\log n$. This is nearly optimal because Carole needs at least $\log n$ bits in the worst case since she needs to output the missing integer. (In fact, there exists the following optimal algorithm for Carole using $\log n$ bits. For each i, store the parity sum of the ith bits of all numbers seen thus far. The final parity sum bits are the bits of the missing number.) A similar solution will work even if n is unknown, for example by letting $n = \max_{j \le i} \pi_{-1}[j]$ each time.

Paul and Carole have a history. It started with the "twenty questions" problem solved in [200]. Paul, which stood for Paul Erdos, was the one who asked questions. Carole is an anagram for Oracle. Aptly, she was the one who answered questions. Joel Spencer and Peter Winkler used Paul and Carole to coincide with Pusher and Chooser respectively in studying certain chip games in which Carole chose which groups the chips falls into and Paul determined which group of chips to push. In the puzzle above, Paul permutes and Carole cumulates.

Generalizing the puzzle a little further, let π_{-2} be π with two elements missing. The natural solution would be for Carole to store $s = \frac{n(n+1)}{2} - \sum_{j \le i} \pi_{-2}[j]$ and $p = n! - \prod_{j \le i} \pi_{-2}[j]$, giving two equations with two unknown numbers, but this will result in storing large number of bits since $n!$ is large. Instead, Carole can use far fewer bits tracking

$$s = \frac{n(n+1)}{2} - \sum_{j \le i} \pi_{-2}[j] \quad \text{and} \quad ss = \frac{n(n+1)(2n+1)}{6} - \sum_{j \le i} (\pi_{-2}[j])^2$$

In general, what is the smallest number of bits needed to identify the k missing numbers in π_{-k}? Following the approach above, the solution is to maintain *power sums*

$$s_p(x_1, \ldots, x_k) = \sum_{i=1}^{k} (x_i)^p,$$

for $p = 1, \ldots, k$ and solving for x_i's. A different method uses *elementary symmetric polynomials* [169]. The ith such polynomial $\sigma_i(x_1, \ldots, x_k)$ is the sum of all possible i term products of the parameters, i.e.,

$$\sigma_i(x_1, \ldots, x_k) = \sum_{j_1 < \cdots < j_i} x_{j_1} \cdots x_{j_i}.$$

Carole continuously maintains σ_i's for the missing k items in field F_q for some prime $n \le q \le 2n$, as Paul presents the numbers one after the other (the details are in [169]). Since

$$\prod_{i=1,\dots,k}(z - x_i) = \sum_{i=0}^{k}(-1)^i \sigma_i(x_1,\dots,x_k) z^{k-i},$$

Carole needs to factor this polynomial in F_q to determine the missing numbers. No deterministic algorithms are known for the factoring problem, but there are randomized algorithms take roughly $O(k^2 \log n)$ bits and time [213]. The elementary symmetric polynomial approach above comes from [169] where the authors solve the *set reconciliation problem* in the communication complexity model. The *subset* reconciliation problem is related to our puzzle.

Generalizing the puzzle, Paul may present a multiset of elements in $\{1,\dots,n\}$ with a single missing integer, i.e., he is allowed to *re*-present integers he showed before; Paul may present updates showing which integers to insert and which to delete, and Carole's task is to find the integers that are no longer present; etc. All of these problems are no longer (mere) puzzles; they are derived from motivating data stream applications.

1.2 Puzzle 2: Fishing

Say Paul goes fishing. There are many different fish species $U = \{1,\dots,u\}$. Paul catches one fish at a time, $a_t \in U$ being the fish species he catches at time t. $c_t[j] = |\{a_i \mid a_i = j, i \le t\}|$ is the number of times he catches the species j up to time t. Species j is *rare* at time t if it appears precisely once in his catch up to time t. The *rarity* $\rho[t]$ of his catch at time t is the ratio of the number of rare j's to u:

$$\rho[t] = \frac{|\{j \mid c_t[j] = 1\}|}{u}.$$

Paul can calculate $\rho[t]$ precisely with a $2u$-bit vector and a counter for the current number of rare species, updating the data structure in $O(1)$ operations per fish caught. However, Paul wants to store only as many bits as will fit his tiny suitcase, i.e., $o(u)$, preferably $O(1)$ bits.

Suppose Paul has a deterministic algorithm to compute $\rho[t]$ precisely. Feed Paul any set $S \subset U$ of fish species, and say Paul's algorithm stores only $o(u)$ bits in his suitcase. Now we can check if any $i \in S$ by simply feeding Paul i and checking $\rho[t+1]$: the number of rare items *de*creases by one if and only if $i \in S$. This way we can recover entire S from his suitcase by feeding different i's one at a time, which is impossible in general if Paul had only stored $o(|S|)$ bits. Therefore, if Paul wishes to work out of his one suitcase, he can not compute $\rho[t]$ exactly. This argument has elements of lower bound proofs found in the area of data streams.

However, proceeding to the task at hand, Paul can *approximate* $\rho[t]$. Paul picks k random fish species each independently, randomly with probability $1/u$ at the beginning and maintains the number of times each of these fish types appear in his bounty, as he catches fish one after another. Say $X_1[t],\ldots,X_k[t]$ are these counts after time t. Paul outputs $\hat{\rho}[t] = \frac{|\{i \mid X_i[t]=1\}|}{k}$ as an estimator for ρ. We have,

$$\mathsf{Pr}(X_i[t] = 1) = \frac{|\{j \mid c_t[j] = 1\}|}{u} = \rho[t],$$

for any fixed i and the probability is over the fish type X_i. If $\rho[t]$ is large, say at least $1/k$, $\hat{\rho}[t]$ is a good estimator for $\rho[t]$ with arbitrarily small ε and significant probability.

However, typically, ρ is unlikely to be large because presumably u is much larger than the species found at any spot Paul fishes. Choosing a random species from $\{1,\ldots,u\}$ and waiting for it to be caught is ineffective. We can make it more realistic by redefining rarity with respect to the species Paul in fact sees in his catch. Let

$$\gamma[t] = \frac{|\{j \mid c_t[j] = 1\}|}{|\{j \mid c_t[j] \neq 0\}|}.$$

As before, Paul would have to approximate $\gamma[t]$ because he can not compute it exactly using a small number of bits. Following [28], define a family of hash functions $\mathcal{H} \subset [n] \to [n]$ (where $[n] = \{1,\ldots,n\}$) to be *min-wise independent* if for any $X \subset [n]$ and $x \in X$, we have

$$\mathsf{Pr}_{h\in\mathcal{H}}[h(x) = \min\ h(X)] = \frac{1}{|X|},$$

where, $h(X) = \{h(x)\colon\ x \in X\}$. Paul chooses k min-wise independent hash functions $h_1,h_2,\ldots,h_k$ for some parameter k to be determined later and maintains $h_i^*(t) = \min_{j \le t} h_i(a_j)$ at each time t, that is, min hash value of the multi-set $\{\ldots, a_{t-2}, a_{t-1}, a_t\}$. He also maintain k counters $C_1(t), C_2(t), \ldots, C_k(t)$; $C_i(t)$ counts an item with (the current value of) hash value $h_i^*(t)$ in $\{\ldots, a_{t-2}, a_{t-1}, a_t\}$. It is trivial to maintain both $h_i^*(t)$ and $C_i(t)$ as t progresses and new items are seen. Let

$$\hat{\gamma}[t] = \frac{|\{i \mid 1 \le i \le k,\ C_i(t) = 1\}|}{k}.$$

Notice that $\mathsf{Pr}(C_i(t) = 1)$ is the probability that $h_i(t)$ is the hash value of one of the items that appeared precisely once in $a_1, \ldots, a_t$ which equals $\frac{|\{j \mid c[j]=1\}|}{|\{j \mid c[j] \neq 0\}|} = \gamma[t]$. Hence, $\hat{\gamma}[t]$ is a good estimator for $\gamma[t]$ provided $\gamma[t]$ is large, say at least $1/k$. That completes the sketch of the Paul's algorithm.

The remaining detail is that Paul needs to pick h_i's. If Paul resorts to his tendency to permute, i.e., if he picks a randomly chosen permutation π over $[u] = \{1, \ldots, u\}$, then h_i's will be min-wise hash functions. However, it requires $\Theta(u \log u)$ bits to represent a random permutation from the set of all permutations over $[u]$. Thus the number of bits needed to store the hash function will be close to u which is prohibitive!

To overcome this problem, Paul picks a family of *approximate* min-hash functions. A family of hash functions, $\mathcal{H} \subset [n] \to [n]$ is called ϵ-*min-wise independent* if for any $X \subset [n]$ and $x \in X$, we have

$$\mathsf{Pr}_{h \in \mathcal{H}}[h(x) = \min\ h(X)] = \frac{1}{|X|}(1 \pm \epsilon).$$

Indyk [135] presents a family of ϵ-min-wise independent hash functions such that any function from this family can be represented using $O(\log u \log(1/\epsilon))$ bits only and each hash function can be computed efficiently in $O(\log(1/\epsilon))$ time. This family is a set of polynomials over $GF(u)$ of degree $O(\log(1/\epsilon))$. Plugging this result into the solution above, Paul uses $O(k \log u \log(1/\epsilon))$ bits and estimates $\hat{\gamma}[t] \in (1 \pm \epsilon)\gamma[t]$, provided $\gamma[t]$ is large, that is, at least $1/k$. It will turn out that in applications of streaming interest, we need to only determine if $\gamma[t]$ is large, so this solution will do.

As an aside, the problem of estimating the rarity is related to a different problem. Consider fishing again and think of it as a random sampling process. There is an unknown probability distribution P on the countable set of fish types with p_t being the probability associated with fish type t. A *catch* is a sample S fishes drawn independently from fish types according to the distribution P. Let $c[t]$ be the number of times fish type t appears in S. The problem is to estimate the probability of fish type t being the next catch. Elementary reasoning would indicate that this probability is $c[t]/|S|$. However, it is unlikely that all (of the large number of) fish types in the ocean are seen in Paul's catch, or even impossible if the number of fish types is infinite. Hence, there are fish types t^* that do *not* appear in the sample (i.e., $c[t^*] = 0$) and the elementary reasoning above would indicate that they have probability 0 of being caught next. This is a conundrum in the elementary reasoning since t^* is indeed present in the ocean and has nonzero probability of being caught in a given probability distribution P. Let $m = \sum_{t^* \notin S} p^*_t$. The problem of estimating m is called the *missing mass problem*. In a classical work by Good (attributed to Turing too) [113], it is shown that m is estimated by $s[1]/|S|$, where $s[k]$ is the number of fish types that appear k times in S, provably with small bias; recall that our rarity γ is closely related to $s[1]/|S|$. Hence, our result here on estimating rarity in data streams is of independent interest in estimating the missing mass. See recent work on Turing-Good estimators in [185].

Once we generalize the fishing puzzle – letting the numerator be more generally $|\{j \mid c_t[j] \leq \alpha\}|$ for some α, letting Carole go fishing too, or letting Paul and Carole throw fish back into the Ocean as needed – there are some real data streaming applications [71]. In the reality of data streams, one is confronted with fishing in a far larger domain, that is, u is typically very large.

1.3 Puzzle 3: Pointer and Chaser

We study yet another game between Paul and Carole. There are $n+1$ positions numbered $1,\ldots,n+1$ and for each position i, Paul points to some position given by $P[i] \in \{1,2,\ldots,n\}$. By the Pigeonhole principle, there must exist at least two positions i,j such that $P[i] = P[j] = D$ say, for a *duplicate*. Several different duplicates may exist, and the same duplicate may appear many times since P is an arbitrary many-to-one function. Carole's goal is to find *any* one duplicate using $O(\log n)$ bits of storage and using no more than $O(n)$ queries to Paul.

The trivial solution to this problem is to take each item $i \in \{1,\ldots,n\}$ in turn and count how many positions j have $P[j] = i$ by querying $P[1], P[2], \ldots, P[n+1]$ in turn. This solution takes $O(\log n)$ bits of extra storage but needs $\Theta(n)$ queries per i in the worst case for a total of $O(n^2)$ queries in all. One of the interesting aspects of this solution is that P is accessed in *passes*, that is, we query $P[1], P[2], \ldots, P[n+1]$ in order and repeat that many times.

One suspects that this problem should be solvable along the lines in Section 1.1 using a few passes. The only such solution we know is not optimally efficient and works as follows. In the first pass by querying $P[1], P[2], \ldots, P[n+1]$ in order, Carole counts the number of items below $n/2$ and those above $n/2$. Whichever counter is strictly larger than $n/2$ (one such counter exists) shows a range of size $n/2$ with a duplicate. Now we recurse on the range with a duplicate in the next pass and so on. This method uses only $O(\log n)$ bits of storage (2 counters together with the endpoints of the range of interest in the current pass), but needs $O(n)$ queries per pass, taking $O(n \log n)$ queries in all. Further, this solution uses $O(\log n)$ passes. This solution can be generalized to use $O(k \log n)$ bits of storage and use only $O(\log_k n)$ passes. As it is, this approach similar to Section 1.1 does not meet the desired bound of $O(n)$ queries.

Jun Tarui [204] has presented a lower bound on the number of passes needed to solve this problem. Consider odd n and a restricted class of inputs such that the numbers in the first half (and likewise the second half) are distinct. Hence the duplicate item must appear once in the first half and once in the second half. A solution with s bits of memory and r passes implies a two-party protocol with r rounds and s communication bits in each round for the game where Paul and Carole get $(n+1)/2$-sized subsets PA and CA respectively of $\{1,\ldots,n\}$ and the task is to find and agree on some w that is in the intersection of PA and CA. Such a protocol corresponds to a monotone (i.e., AND/OR) circuit computing the Majority function with depth at most r and fan-in at most 2^s. This leads to a lower bound of $\Omega(\log n / \log\log n)$ passes for $s = O(\log n)$.

In the final solution, Carole does not count, but chases pointers. Start chasing the pointers from $X = n+1$ going successively to the location pointed to by $P[X]$ for current X. Now the problem of finding a duplicate is the same as that of finding if a "linked list" has a loop not containing the start "node" $n+1$. This is easy to solve in $O(n)$ queries to P with 2 pointers. Notice that this solution makes use of the random access given by $P[X]$. (As an aside, the problem is interesting if $P[i] \in S$ where S is *some* set of size n, not necessarily $\{1,2,\ldots,n\}$. Then, we do not know of an algorithm that takes less than $O(n^2)$ time within our space constraints.)

This puzzle is related to Pollard's rho method for determining the greatest common divisor between integers [4]. The primary focus here is on separating the complexity of the problem using passes vs using random accesses.

1.4 Lessons

The missing-number puzzle in Section 1.1 shows the case of a data stream problem that can be deterministically solved precisely with $O(\log n)$ bits (when $k = 1,2$, etc.). Such algorithms – deterministic and exact – are uncommon in data stream processing. In contrast, the puzzle in Section 1.2 is solved only up to an approximation using a randomized algorithm in polylog bits. This – randomized and approximate solution – is more representative of currently known data stream algorithms. Further, the estimation of γ in Section 1.2 is accurate *only* when it is large; for small γ, the estimate $\hat{\gamma}$ is arbitrarily bad. This points to a feature that generally underlies data stream algorithmics. Such features – which applied algorithmicists need to keep in mind while formulating problems to address data stream issues – will be discussed in more detail later. In Section 1.3, the puzzle demonstrates the difference between passes (the best known solution using only passes takes $O(n \log n)$ time) and random access (solution takes $O(n)$ time using random access).

References

[1] http://www.tpc.org/. Details of transactions testing at http://www.tpc.org/tpcc/detail.asp.

[2] http://www2.ece.rice.edu/~duarte/compsense/.

[3] http://www7.nationalacademies.org/bms/Massive_Data_Workshop.html.

[4] "Lance Fortnow," Blog on 03/2005.

[5] D. Abadi, D. Carney, U. Cetintemel, M. Cherniack, C. Convey, S. Lee, M. Stonebraker, N. Tatbul, and S. Zdonik, "Aurora: A new model and architecture for data stream management," *VLDB Journal*, vol. 12, no. 2, pp. 120–139, 2003. See also: "Aurora: A data stream management system", *Proc. ACM SIGMOD* 2003, Demo.

[6] A. Aboulnaga and S. Chaudhuri, "Self-tuning histograms: Building histograms without looking at data," *Proc. ACM SIGMOD*, pp. 181–192, 1998.

[7] D. Achlioptas and F. McSherry, "Fast computation of low rank approximation," *Proc. ACM STOC*, pp. 611–618, 2001.

[8] L. Adamic, "Zipf," power-law, pareto – a ranking tutorial, http://www.hpl.hp.com/research/idl/papers/ranking/, 2000.

[9] P. K. Agarwal, S. Har-Peled, and K. Varadarajan, "Geometric Approximation via Coresets," Survey. Available at http://valis.cs.uiuc.edu/~sariel/papers/04/survey/.

[10] G. Aggarwal, N. Mishra, and B. Pinkas, "Secure computation of the K'th-ranked element," *Advances in Cryptology – Eurocrypt*, LNCS 3027, Springer-Verlag, pp. 40–55, May 2004.

[11] M. Ajtai, T. Jayram, S. R. Kumar, and D. Sivakumar, "Counting inversions in a data stream," *Proc. ACM STOC*, pp. 370–379, 2002.

[12] N. Alon, N. Duffield, C. Lund, and M. Thorup, "Estimating sums of arbitrary selections with few probes," *Proc. ACM PODS*, 2005.

[13] N. Alon, P. Gibbons, Y. Matias, and M. Szegedy, "Tracking join and self-join sizes in limited storage," *Proc. ACM PODS*, pp. 10–20, 1999.

[14] N. Alon, Y. Matias, and M. Szegedy, "The space complexity of approximating the frequency moments," *Proc. ACM STOC*, pp. 20–29, 1996.

[15] A. Arasu, B. Babcock, S. Babu, M. Datar, K. Ito, I. Nishizawa, J. Rosenstein, and J. Widom, "STREAM: The stanford stream data manager," *Proc. ACM SIGMOD*, 2003, Demo.

[16] A. Arasu and G. Manku, "Approximate Counts and Quantiles over Sliding Windows," *Proc. ACM PODS*, pp. 286–296, 2004.

[17] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and issues in data stream systems," *Proc. ACM PODS*, pp. 1–16, 2002.

[18] A. Bagchi, A. Chaudhary, D. Eppstein, and M. Goodrich, "Deterministic sampling and range counting in geometric data streams," *Proc. ACM SOCG*, pp. 144–151, 2004.

[19] S. Balakrishnan and D. Madigan, "A one-pass sequential Monte Carlo method for Bayesian analysis of massive datasets," *Manuscript*, 2004.

[20] M. Balazinska, H. Balakrishnan, and M. Stonebraker, "Load management and high availability in the Medusa distributed stream processing system," *Proc. ACM SIGMOD*, pp. 929–930, 2004.

[21] Z. Bar-yossef, T. Jayram, R. Kumar, and D. Sivakumar, "Information statistics approach to data stream and communication complexity," *Proc. IEEE FOCS*, pp. 209–218, 2002.

[22] Z. Bar-Yossef, T. Jayram, R. Kumar, D. Sivakumar, and L. Trevisan, "Counting distinct elements in a data stream," *Proc. RANDOM*, pp. 1–10, 2000.

[23] Z. Bar-Yossef, R. Kumar, and D. Sivakumar, "Reductions in streaming algorithms, with an application to counting triangles in graphs," *Proc. ACM-SIAM SODA*, pp. 623–632, 2002.

[24] T. Batu, S. Guha, and S. Kannan, "Inferring mixtures of markov chains," *Proc. COLT*, pp. 186–199, 2004.

[25] K. Beauchamp, "Walsh functions and their applications," 1975.

[26] M. Bender, A. Fernandez, D. Ron, A. Sahai, and S. Vadhan, "The power of a pebble: Exploring and mapping directed graphs," *Proc. ACM STOC*, pp. 269–278, 1998.

[27] G. Blelloch, B. Maggs, S. Leung, and M. Woo, "Space efficient finger search on degree-balanced search trees," *Proc. ACM-SIAM SODA*, pp. 374–383, 2003.

[28] A. Broder, M. Charikar, A. Freize, and M. Mitzenmacher, "Min-wise independent permutations," *Proc. ACM STOC*, pp. 327–336, 1998.

[29] L. Buriol, D. Donato, S. Leonardi, and T. Matzner, "Using data stream algorithms for computing properties of large graphs," *Workshop on Massive Geometric Datasets*, With *ACM SoCG* 2005, Pisa.

[30] A. R. Calderbank, A. Gilbert, K. Levchenko, S. Muthukrishnan, and M. Strauss, "Improved range-summable random variable construction algorithms," *Proc. ACM-SIAM SODA*, pp. 840–849, 2005.

[31] A. Chakrabarti, S. Khot, and X. Sun, "Near-optimal lower bounds on the multi-party communication complexity of set disjointness," *IEEE Conference on Computational Complexity*, pp. 107–117, 2003.

[32] J. Chambers, C. Mallows, , and B. Stuck, "A method for simulating stable random variables," *Journal of the American Statistical Association*, vol. 71, no. 354, pp. 340–344, 1976.

[33] T. Chan, "Data stream algorithms in Computational Geometry," *Workshop on New Horizons in Computing* (Kyoto), 2005.

[34] T. Chan and E. Chen, "Multi-pass geometric algorithms," *Proc. ACM SoCG*, pp. 180–189, 2005.

[35] M. Charikar, C. Chekuri, T. Feder, and R. Motwani, "Incremental clustering and dynamic information retrieval," *Proc. ACM STOC*, pp. 626–635, 1997.

[36] M. Charikar, K. Chen, and M. Farach-Colton, "Finding frequent items in data streams," *Proc. ICALP*, pp. 693–703, 2002.

[37] M. Charikar, L. O'Callaghan, and R. Panigrahy, "Better streaming algorithms for clustering problems," *Proc. ACM STOC*, pp. 693–703, 2003.

[38] S. Chaudhuri, R. Motwani, and V. Narasayya, "Random sampling for histogram construction: How much is enough?," *Proc. SIGMOD*, pp. 436–447, 1998.

[39] J. Chen, D. DeWitt, F. Tian, , and Y. Wang, "NiagaraCQ: A scalable continuous query system for internet databases," *Proc. ACM SIGMOD*, pp. 379–390, 2000.

[40] S. Chen, A. Gaur, S. Muthukrishnan, and D. Rosenbluth, "Wireless in loco sensor data collection and applications," *Workshop on Mobile Data Access* (MOBEA) II, Held with *WWW Conf*, 2004.

[41] S. Chien, L. Rasmussen, and A. Sinclair, "Clifford algebras and approximating the permanent," *Proc. ACM STOC*, pp. 222–231, 2002.

[42] E. Cohen and H. Kaplan, "Spatially-decaying aggregation over a network: Model and algorithms," *Proc. ACM SIGMOD*, pp. 707–718, 2004.

[43] R. Cole, "On the dynamic finger conjecture for splay trees, part II, The proof," *Technical Report TR1995-701*, Courant Institute, NYU, 1995.

[44] R. Cole and U. Vishkin, "Deterministic coin tossing and accelerating cascades: micro and macro techniques for designing parallel algorithms," *Proc. ACM STOC*, pp. 206–219, 1986.

[45] D. Coppersmith and R. Kumar, "An improved data stream algorithm for frequency moments," *Proc. ACM-SIAM SODA*, pp. 151–156, 2004.

[46] G. Cormode, "Stable distributions for stream computations: It's as easy as 0,1,2," *Workshop on Management and Processing of Massive Data Streams (MPDS) at FCRC*, 2003.

[47] G. Cormode, M. Datar, P. Indyk, and S. Muthukrishnan, "Comparing data streams using hamming norms (How to zero in)," *Proc. VLDB*, pp. 335–345, 2002.

[48] G. Cormode and M. Garofalakis, "Sketching streams through the net: Distributed approximate query tracking," *Proc. VLDB*, pp. 13–24, 2005.

[49] G. Cormode, M. Garofalakis, S. Muthukrishnan, and R. Rastogi, "Holistic aggregates in a networked world: Distributed tracking of approximate quantiles," *Proc. ACM SIGMOD*, pp. 25–36, 2005.

[50] G. Cormode, T. Johnson, F. Korn, S. Muthukrishnan, O. Spatscheck, and D. Srivastava, "Holistic UDAFs at streaming speeds," *Proc. ACM SIGMOD*, pp. 35–46, 2004.

[51] G. Cormode, F. Korn, S. Muthukrishnan, and D. Srivastava, "Finding hierarchical heavy hitters in data streams," *Proc. VLDB*, pp. 464–475, 2003.

[52] G. Cormode, F. Korn, S. Muthukrishnan, and D. Srivastava, "Diamond in the rough: Finding hierarchical heavy hitters in multi-dimensional data," *Proc. ACM SIGMOD*, pp. 155–166, 2004.

[53] G. Cormode and S. Muthukrishnan, "The string edit distance matching problem with moves," *Proc. ACM-SIAM SODA*, pp. 667–676, 2002.

[54] G. Cormode and S. Muthukrishnan, "Estimating dominance norms on multiple data streams," *Proc. ESA*, pp. 148–160, 2003.

[55] G. Cormode and S. Muthukrishnan, "What is hot and what is not: Tracking most frequent items dynamically," *Proc. ACM PODS*, pp. 296–306, 2003.

[56] G. Cormode and S. Muthukrishnan, "Radial histograms for spatial streams," *DIMACS Technical Report*, 2003-11.

[57] G. Cormode and S. Muthukrishnan, "What is new: Finding significant differences in network data streams," *Proc. INFOCOM*, 2004.

[58] G. Cormode and S. Muthukrishnan, "Space efficient mining of multigraph streams," *Proc. ACM PODS*, 2005.

[59] G. Cormode and S. Muthukrishnan, "Summarizing and mining skewed data streams," *Proc. SIAM SDM*, 2005.

[60] G. Cormode and S. Muthukrishnan, "An improved data stream summary: The count-min sketch and its applications," *J. Algorithms*, vol. 55, no. 1, pp. 58–75, April 2005.

[61] G. Cormode, S. Muthukrishnan, and I. Rozenbaum, "Summarizing and mining inverse distributions on data streams via dynamic inverse sampling," *Proc. VLDB*, pp. 25–36, 2005.

[62] G. Cormode, S. Muthukrishnan, and C. Sahinalp, "Permutation editing and matching via embeddings," *Proc. ICALP*, pp. 481–492, 2001.

[63] G. Cormode, M. Paterson, S. Ṣahinalp, and U. Vishkin, "Communication complexity of document exchange," *Proc. ACM-SIAM SODA*, pp. 197–206, 2000.

[64] C. Cortes, K. Fisher, D. Pregibon, and A. Rogers, "Hancock: A language for extracting signatures from data streams," *Proc. KDD*, pp. 9–17, 2000.

[65] C. Cortes and D. Pregibon, "Signature-based methods for data streams," *Data Mining and Knowledge Discovery*, vol. 5, no. 3, pp. 167–182, 2001.

[66] C. Cortes, D. Pregibon, and C. Volinsky, "Communities of interest," *Proc. of Intelligent Data Analysis*, pp. 105–114, 2001.

[67] C. Cranor, T. Johnson, V. Shkapenyuk, and O. Spatscheck, "The gigascope stream database," *IEEE Data Engineering Bulletin*, vol. 26, no. 1, pp. 27–32, 2003. See also: C. Cranor, Y. Gao, T. Johnson, V. Shkapenyuk and

O. Spatscheck, “Gigsacope: High performance network monitoring with an SQL interface,” *ACM SIGMOD* 2002, Demo.

[68] T. Dasu and T. Johnson, *Exploratory Data Mining and Data Quality*. Vol. ISBN: 0-471-26851-8, Wiley, May 2003.

[69] T. Dasu, T. Johnson, S. Muthukrishnan, and V. Shkapenyuk, “Mining database structure or how to build a data quality browser,” *Proc. ACM SIGMOD*, pp. 240–251, 2002.

[70] M. Datar, A. Gionis, P. Indyk, and R. Motwani, “Maintaining stream statistics over sliding windows,” *Proc. ACM-SIAM SODA*, pp. 635–644, 2002.

[71] M. Datar and S. Muthukrishnan, “Estimating rarity and similarity in window streams,” *Proc. ESA*, pp. 323–334, 2002.

[72] G. Davis, S. Mallat, and M. Avellaneda, “Greedy adaptive approximation,” *Journal of Constructive Approximation*, vol. 13, pp. 57–98, 1997.

[73] R. DeVore and G. Lorentz, *Constructive Approximation*. New York: Springer-Verlag, 1993.

[74] D. Donoho, “High-dimensional data analysis: The curses and blessings of dimensionality,” *Manuscript*, 2000. http://www-stat.stanford.edu/˜donoho/.

[75] D. Donoho, “Compressed sensing,” *Manuscript*, 2004. http://www-stat.stanford.edu/˜donoho/Reports/2004/CompressedSensing091604.pdf.

[76] P. Drineas and R. Kannan, “Pass efficient algorithms for approximating large matrices,” *Proc. ACM-SIAM SODA*, pp. 223–232, 2003.

[77] P. Drineas, R. Kannan, and M. Mahoney, “Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication,” *Yale Technical Report* YALEU/DCS/TR-1269, 2004. To appear in *SIAM J. Computing*.

[78] P. Drineas, R. Kannan, and M. Mahoney, “Fast Monte Carlo algorithms for matrices II: Computing low-rank approximations to a matrix,” *Yale Technical Report* YALEU/DCS/TR-1270, 2004. To appear in *SIAM J. Computing*.

[79] P. Drineas, R. Kannan, and M. Mahoney, “Fast Monte Carlo algorithms for matrices III: Computing an efficient approximate decomposition of a matrix,” *Yale Technical Report* YALEU/DCS/TR-1271, To appear in *SIAM J. Computing*, 2004.

[80] D. Du and F. Hwang, *Combinatorial Group Testing and Its Applications*. 2nd ed., World Scientific Singapore, 2000.

[81] N. Duffield, C. Lund, and M. Thorup, “Flow sampling under hard resource constraints,” *Sigmetrics*, pp. 85–96, 2004.

[82] M. Elkin and J. Zhang, “Efficient algorithms for constructing $(1+\varepsilon,\beta)$-spanners in the distributed and streaming models,” *Proc. ACM PODC*, pp. 160–168, 2004.

[83] C. Estan, S. Savage, and G. Varghese, “Automatically inferring patterns of resource consumption in network traffic,” *Proc. SIGCOMM*, pp. 137–148, 2003.

[84] C. Estan and G. Varghese, “New directions in traffic measurement and accounting: Focusing on the elephants, ignoring the mice,” *ACM Transactions on Computer System*, vol. 21, no. 3, pp. 270–313, 2003.

[85] R. Fagin, M. Naor, and P. Winkler, "Comparing information without leaking it: Simple solutions," *Communications of the ACM*, vol. 39, no. 5, pp. 77–85, 1996.

[86] U. Feige, "A threshold of ln n for approximating set cover," *Journal of ACM*, vol. 45, no. 4, pp. 634–652, 1998.

[87] J. Feigenbaum, "Massive graphs: Algorithms, applications, and open problems," *Invited Lecture,* Combinatorial Pattern Matching, 1999.

[88] J. Feigenbaum, Y. Ishai, T. Malkin, K. Nissim, M. Strauss, and R. Wright, "Secure multiparty computation of approximations," *Proc. ICALP*, pp. 927–938, 2001.

[89] J. Feigenbaum, S. Kannan, A. McGregor, S. Suri, and J. Zhang, "On graph problems in a semi-streaming model," *Proc of ICALP*, pp. 531–543, 2004.

[90] J. Feigenbaum, S. Kannan, A. McGregor, S. Suri, and J. Zhang, "Graph distances in the streaming model: The value of space," *Proc. ACM-SIAM SODA*, pp. 745–754, 2005.

[91] J. Feigenbaum, S. Kannan, M. Strauss, and M. Viswanathan, "An approximate L_1 difference algorithm for massive data streams," *Proc. IEEE FOCS*, pp. 501–511, 1999.

[92] J. Feigenbaum, S. Kannan, and J. Ziang, "Computing diameter in the streaming and sliding window models," *Algorithmica*, vol. 41, no. 1, pp. 25–41, 2004.

[93] M. Fischer and S. Salzberg, "Finding a majority among N votes: Solution to problem 81-5," *J. Algorithms*, vol. 3, pp. 376–379, 1982.

[94] P. Flajolet and G. Martin, "Probabilistic counting," *Proc. FOCS*, pp. 76–82, 1983.

[95] G. Frahling, P. Indyk, and C. Sohler, "Sampling in dynamic data streams and applications," *Proc. ACM SoCG*, pp. 142–149, 2005.

[96] G. Frahling and C. Sohler, "Coresets in dynamic geometric data streams," *Proc. ACM STOC*, pp. 209–217, 2005.

[97] M. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection," *Advances in Cryptology – Eurocrypt*, LNCS 3027, Springer-Verlag, pp. 1–19, May 2004.

[98] S. Ganguly, M. Garofalakis, and R. Rastogi, "Tracking set-expression cardinalities over continuous update streams," *VLDB Journal*, vol. 13, no. 4, pp. 354–369, 2004.

[99] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[100] M. Garofalakis and A. Kumar, "Deterministic wavelet thresholding for maximum-error metrics," *Proc. ACM PODS*, pp. 166–176, 2004.

[101] L. Gasieniec and S. Muthukrishnan, "Determinsitic algorithms for estimating heavy-hitters on Turnstile data streams," *Manuscript*, 2005.

[102] D. Geiger, V. Karamcheti, Z. Kedem, and S. Muthukrishnan, "Detecting malicious network traffic using inverse distributions of packet contents," *Proc. MineNet*, 2005, Held with *Proc. ACM SIGCOMM*, 2005.

[103] P. Gibbons and S. Trithapura, "Estimating simple functions on the union of data streams," *Proc. ACM SPAA*, pp. 281–291, 2001.

[104] A. Gilbert, S. Guha, Y. Kotidis, P. Indyk, S. Muthukrishnan, and M. Strauss, "Fast, small space algorithm for approximate histogram maintenance," *Proc. ACM STOC*, pp. 389–398, 2002.

[105] A. Gilbert, Y. Kotidis, S. Muthukrishnan, and M. Strauss, "Surfing wavelets on streams: One pass summaries for approximate aggregate queries," *VLDB Journal*, pp. 79–88, 2001.

[106] A. Gilbert, Y. Kotidis, S. Muthukrishnan, and M. Strauss, "QuickSAND: Quick summary and analysis of network data," *DIMACS Technical Report*, 2001-43.

[107] A. Gilbert, Y. Kotidis, S. Muthukrishnan, and M. Strauss, "How to summarize the universe: Dynamic maintenance of quantiles," *Proc. VLDB*, pp. 454–465, 2002.

[108] A. Gilbert, S. Muthukrishnan, and M. Strauss, "Approximation of functions over redundant dictionaries using coherence," *Proc. ACM-SIAM SODA*, pp. 243–252, 2003.

[109] A. Gilbert, S. Muthukrishnan, and M. Strauss, "Improved time bounds for near-optimal sparse Fourier representations," *SPIE Conf*, Wavelets, 2005, See also: A. Gilbert and S. Guha and P. Indyk and S. Muthukrishnan and M. Strauss, "Near-optimal sparse fourier estimation via sampling", *Proc. ACM STOC*, 152–161, 2002.

[110] A. Gilbert, S. Muthukrishnan, M. Strauss, and J. Tropp, "Improved sparse approximation over quasi-coherent dictionaries," *Intl Conf on Image Processing (ICIP)*, pp. 37–40, 2003.

[111] L. Golab, D. DeHaan, E. Demaine, A. Lopez-Ortiz, and I. Munro, "Identifying frequent items in sliding windows over on-line packet streams," *Internet Measurement Conference*, pp. 173–178, 2003.

[112] O. Goldreich, "Secure multiparty computation," Book at http://philby.ucsd.edu/cryptolib/BOOKS/oded-sc.html, 1998.

[113] I. Good, "The population frequencies of species and the estimation of population parameters," *Biometrika*, vol. 40, no. 16, pp. 237–264, 1953.

[114] J. Gray and T. Hey, "In search of petabyte databases," http://www.research.microsoft.com/˜Gray/talks/.

[115] J. Gray, P. Sundaresan, S. Eggert, K. Baclawski, and P. Weinberger, "Quickly generating billion-record synthetic databases," *Proc. ACM SIGMOD*, pp. 243–252, 1994.

[116] M. Greenwald and S. Khanna, "Space-efficient online computation of quantile summaries," *Proc. ACM SIGMOD*, 2001.

[117] S. Guha, "Space efficiency in synopsis construction algorithms," *Proc. VLDB*, 2005.

[118] S. Guha and B. Harb, "Waveletssynopsis for data streams: Minimizing noneuclidean error," *Proc. ACM KDD*, 2005.

[119] S. Guha, P. Indyk, S. Muthukrishnan, and M. Strauss, "Histogramming data streams with fast per-item processing," *Proc. ICALP*, pp. 681–692, 2002.

[120] S. Guha, N. Koudas, and K. Shim, "Approximation and streaming algorithms for histogram construction problems," Journal version.

[121] S. Guha, N. Koudas, and K. Shim, "Data streams and histograms," *Proc. ACM STOC*, pp. 471–475, 2001.

[122] S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams," *Proc. IEEE FOCS*, pp. 359–366, 2000.

[123] S. Guha, K. Mungala, K. Shankar, and S. Venkatasubramanian, "Application of the two-sided depth test to CSG rendering," *Proc. I3d, ACM Interactive 3D graphics*, 2003.

[124] A. Gupta and F. Zane, "Counting inversions in lists," *ACM-SIAM SODA*, pp. 253–254, 2003.

[125] A. Haar, "Zur theorie der orthogonalen functionsysteme," *Math Annal.*, vol. 69, pp. 331–371, 1910.

[126] M. Hansen, "Slogging," Keynote plenary talk at *SIAM Conf. Data Mining*, 2005.

[127] M. Henzinger, P. Raghavan, and S. Rajagopalan, "Computing on data stream," *Technical Note 1998-011*, Digital systems research center, Palo Alto, May 1998.

[128] J. Hershberger, N. Shrivastava, S. Suri, and C. Toth, "Space complexity of hierarchical heavy hitters in multi-dimensional data streams," *Proc. ACM PODS*, 2005.

[129] J. Hershberger and S. Suri, "Adaptive sampling for geometric problems over data streams," *Proc. ACM PODS*, pp. 252–262, 2004.

[130] D. Hirschberg, "A linear space algorithm for computing maximal common subsequences," *Comm. ACM*, vol. 18, no. 6, pp. 341–343, 1975.

[131] M. Hoffman, S. Muthukrishnan, and R. Raman, "Location streams: Models and Algorithms," *DIMACS TR*, 2004-28.

[132] G. Humphreys, M. Houston, Y. Ng, R. Frank, S. Ahern, P. Kirchner, and J. Klosowski, "Chromium: A stream processing framework for interactive rendering on clusters," *Proc. ACM SIGGRAPH*, pp. 693–702, 2002.

[133] P. Indyk, "Streaming Algorithms for Geometric Problems," Invited talk at CCCG'04.

[134] P. Indyk, "Stable distributions, pseudorandom generators, embeddings and data stream computation," *Proc. IEEE FOCS*, pp. 189–197, 2000.

[135] P. Indyk, "A small approximately min-wise independent family of hash functions," *Journal of Algorithms*, vol. 38, no. 1, pp. 84–90, 2001.

[136] P. Indyk, "Better algorithms for high dimensional proximity problems via asymmetric embeddings," *Proc. ACM-SIAM SODA*, pp. 539–545, 2003.

[137] P. Indyk, "Stream-based geometric algorithms," *Proc. ACM/DIMACS Workshop on Management and Processing of Data Streams (MPDS)*, 2003.

[138] P. Indyk, "Algorithms for dynamic geometric problems over data streams," *Proc. ACM STOC*, pp. 373–380, 2004.

[139] P. Indyk and D. Woodruff, "Tight lower bounds for the distinct elements problem," *Proc. IEEE FOCS*, 2003.

[140] P. Indyk and D. Woodruff, "Optimal approximations of the frequency moments of data streams," *Proc. ACM STOC*, pp. 202–208, 2005.

[141] G. Jagannathan and R. Wright, "Privacy-preserving distributed k-means clustering over arbitrarily partitioned data," *Proc. ACM KDD*, 2005.

[142] T. Johnson, S. Muthukrishnan, and I. Rozenbaum, "Sampling algorithms in a stream operator," *Proc. ACM SIGMOD*, pp. 1–12, 2005.

[143] T. Johnson, S. Muthukrishnan, O. Spatscheck, and D. Srivastava, "Streams, security and scalability," Keynote talk, appears in *Proc.of 19th Annual IFIP Conference on Data and Applications Security*, Lecture Notes in Computer Science 3654, Springer-Verlag, pp. 1–15, 2005.

[144] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with ZebraNet," *ASPLOS-X Conference*, pp. 96–107, 2002.

[145] S. Kannan, "Open problems in streaming," Ppt slides. Personal communication.

[146] R. Karp, C. Papadimitriou, , and S. Shenker, "A simple algorithm for finding frequent elements in sets and bags," *ACM Transactions on Database Systems*, pp. 51–55, 2003.

[147] J. Kleinberg, "Bursty and hierarchical structure in streams," *Proc. ACM KDD*, pp. 91–101, 2002.

[148] D. Knuth, *The art of computer programming, Volume III: Sorting and searching.* Addison-Wesley, 1973.

[149] E. Kohler, J. Li, V. Paxson, and S. Shenker, "Observed structure of addresses in IP traffic," *Internet Measurement Workshop*, pp. 253–266, 2002.

[150] F. Korn, J. Gehrke, and D. Srivastava, "On computing correlated aggregates over continual data streams," *Proc. ACM SIGMOD*, pp. 13–24, 2001.

[151] F. Korn, S. Muthukrishnan, and D. Srivastava, "Reverse nearest neighbor aggregates over data streams," *Proc. VLDB*, pp. 814–825, 2002.

[152] F. Korn, S. Muthukrishnan, and Y. Wu, "Model fitting of IP network traffic at streaming speeds," *Manuscript*, 2005.

[153] F. Korn, S. Muthukrishnan, and Y.Zhu, "Checks and balances: Monitoring data quality in network traffic databases," *Proc. VLDB*, pp. 536–547, 2003.

[154] B. Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen, "Sketch-based change detection: Methods, evaluation, and applications," *Proc. ACM SIGCOMM Internet Measurement Conference*, pp. 234–247, 2003.

[155] S. Krishnamurthy, S. Chandrasekaran, O. Cooper, A. Deshpande, M. Franklin, J. Hellerstein, W. Hong, S. Madden, F. Reiss, and M. Shah, "TelegraphCQ: An Architectural Status Report," *IEEE Data Engineering Bulletin*, vol. 26, no. 1, pp. 11–18, 2003.

[156] R. Kumar and R. Rubinfeld, "Sublinear time algorithms," Algorithms column in SIGACT News 2003.

[157] E. Kushilevitz and N. Nisan, *Communication Complexity.* Cambridge University Press, 1997.

[158] K. Levchenko and Y. Liu, "Counting solutions of polynomial equations," *Manuscript*, 2005.

[159] K. Levchenko, R. Paturi, and G. Varghese, "On the difficulty of scalably detecting network attacks," *ACM Conference on Computer and Communications Security*, pp. 12–20, 2004.

[160] C. Lund and M. Yannakakis, "On the hardness of approximating minimization problems," *Journal of ACM*, vol. 41, pp. 960–981, 1994.

[161] M. Magdon-Ismail, M. Goldberg, W. Wallace, and D. Siebecker, "Locating hidden groups in communication networks using hidden Markov models," *Proc. ISI*, pp. 126–137, 2003.

[162] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless sensor networks for habitat monitoring," *Proc. WSNA*, pp. 88–97, 2002.

[163] A. Manjhi, V. Shkapenyuk, K. Dhamdhere, and C. Olston, "Finding (recently) frequent items in distributed data streams," *Proc. ICDE*, pp. 767–778, 2005.

[164] G. Manku and R. Motwani, "Approximate frequency counts over data streams," *Proc. VLDB*, pp. 346–357, 2002.

[165] G. Manku, S. Rajagopalan, and B. Lindsay, "Random sampling techniques for space efficient online computation of order statistics of large datasets," *Proc. ACM SIGMOD*, pp. 251–262, 1999.

[166] D. Manocha, "Interactive geometric computations using graphics hardware. Course," *ACM SIGGRAPH*, 2002.

[167] Y. Matias and D. Urieli, "Optimal workload-based wavelet synopses," *Proc. ICDT*, pp. 368–382, 2005.

[168] A. Metwally, D. Agrawal, and A. E. Abbadi, "Efficient computation of frequent and top-k elements in data stream," *Proc. ICDT*, pp. 398–412, 2005.

[169] Y. Minsky, A. Trachtenberg, and R. Zippel, "Set reconciliation with nearly optimal communication complexity," *Technical Report* 2000-1796, Cornell Univ.

[170] N. Mishra, D. Oblinger, and L. Pitt, "Sublinear time approximate clustering," *Proc. ACM-SIAM SODA*, pp. 439–447, 2001.

[171] J. Misra and D. Gries, "Finding repeated elements," *Science of Computer Programming*, pp. 143–152, 1982.

[172] R. Motwani and P. Raghavan, *Randomized Algorithms.* Cambridge University Press, 1995.

[173] K. Mulmuley, *Computational Geometry: An Introduction through Randomized Algorithms.* Prentice Hall, 1993.

[174] I. Munro and M. Paterson, "Selection and sorting with limited storage," *Proc. IEEE FOCS*, pp. 253–258, 1978, Also, *Theoretical Computer Science*, vol. 12, pp. 315–323, 1980.

[175] S. Muthukrishnan, "Nonuniform sparse approximation with Haar wavelet basis," *DIMACS TR*, 2004–42.

[176] S. Muthukrishnan, V. Poosala, and T. Suel, "On rectangular partitionings in two dimensions: Algorithms, complexity and applications," *Proc. ICDT*, pp. 236–256, 1999.

[177] S. Muthukrishnan and S. Ṣahinalp, "Approximate nearest neighbors and sequence comparison with block operations," *Proc. STOC*, pp. 416–424, 2000.

[178] S. Muthukrishnan, R. Shah, and J. Vitter, "Finding deviants on data streams," *Proc. SSDBM*, pp. 41–50, 2004.

[179] S. Muthukrishnan and M. Strauss, "Approximate histogram and wavelet summaries of streaming data," *DIMACS TR* 2004-52, Survey.

[180] S. Muthukrishnan and M. Strauss, "Maintenance of multidimensional histograms," *Proc. FSTTCS*, pp. 352–362, 2003.

[181] S. Muthukrishnan and M. Strauss, "Rangesum histograms," *ACM-SIAM SODA*, pp. 233–242, 2003.

[182] S. Muthukrishnan, M. Strauss, and X. Zheng, "Workload-optimal histograms on streams," *Proc. ESA*, 2005.

[183] B. Natarajan, "Sparse approximate solutions to linear systems," *SIAM J. Computing*, vol. 25, no. 2, pp. 227–234, 1995.

[184] S. Nath, H. Yu, P. Gibbons, and S. Seshan, "Synopsis diffusion for robust aggregation in sensor networks," *Intel Tech Report, IRP-TR-04-13*, 2004.

[185] A. Orlitsky, N. Santhanam, and J. Zhang, "Always good turing: Asymptotically optimal probability estimation," *Proc. IEEE FOCS*, pp. 179–188, 2003.

[186] M. Parseval http://encyclopedia.thefreedictionary.com/Parseval's+theorem 1799.

[187] B. Pinkas, "Cryptographic techniques for privacy-preserving data mining," *SIGKDD Explorations*, the newsletter of the *ACM Special Interest Group on Knowledge Discovery and Data Mining*, January 2003.

[188] I. Pohl, "A minimum storage algorithm for computing the median," *IBM TR* 12713, 1969.

[189] P. Raghavan, "Graph structure of the web: A survey," *Proc. LATIN*, pp. 123–125, 2000.

[190] A. Razborov, A. Wigderson, and A. Yao, "Read-once branching programs, rectangular proofs of the pigeonhole principle and the transversal calculus," *Proc. STOC*, pp. 739–748, 1997.

[191] O. Reingold, "Undirected ST-connectivity in logspace," *Proc. STOC*, pp. 376–385, 2005.

[192] S. Ṣahinalp and U. Vishkin, "Symmetry breaking for suffix tree construction," *Proc. of 26th Symposium on Theory of Computing*, pp. 300–309, 1994.

[193] S. Ṣahinalp and U. Vishkin, "Data compression using locally consistent parsing," *Technical Report*, University of Maryland Department of Computer Science, 1995.

[194] S. Ṣahinalp and U. Vishkin, "Efficient approximate and dynamic matching of patterns using a labeling paradigm," *Proc. IEEE FOCS*, pp. 320–328, 1996.

[195] M. Saks and X. Sun, "Space lower bounds for distance approximation in the data stream model," *Proc. ACM STOC*, pp. 360–369, 2002.

[196] S. Sarawagi, "Query processing in tertiary memory databases," *Proc. VLDB*, pp. 585–596, 1995.

[197] R. Seidel and C. Aragon, "Randomized search trees," *Algorithmica*, vol. 16, pp. 464–497, 1996.

[198] D. Shah, S. Iyer, B. Prabhakar, and N. McKeown, "Maintaining statistics counters in router line cards," *IEEE Micro*, pp. 76–81, 2002.

[199] N. Shrivastava, C. Buragohain, D. Agrawal, and S. Suri, "Medians and beyond: New aggregation techniques for sensor networks," *Proc. ACM SenSys*, 2004.

[200] J. Spencer and P. Winkler, "Three thresholds for a liar," *Combinatorics, Probability and Computing*, vol. 1, no. 1, pp. 81–93, 1992.

[201] H. Subramaniam, R. Wright, and Z. Yang, "Experimental analysis of privacy-preserving statistics computation," *Proc. of the Workshop on Secure Data Management* (held in conjunction with VLDB), Springer, LNCS 3178, 2004.

[202] S. Suri, C. Toth, and Y. Zhou, "Range counting over multidimensional data streams," *Proc. ACM SoCG*, pp. 160–169, 2004.

[203] M. Szegedy, "Naer optimality of the priority sampling procedure," *ECCC TR05-001*, 2005.

[204] J. Tarui Finding duplicates in passes. *Personal Communication* and http://weblog.fortnow.com/2005/03/finding-duplicates.html#comments.

[205] V. Temlyakov, "The best m-term approximation and greedy algorithms," *Advances in Computational Math.*, vol. 8, pp. 249–265, 1998.

[206] N. Thaper, S. Guha, P. Indyk, and N. Koudas, "Dynamic multidimensional histograms," *Proc. ACM SIGMOD*, pp. 428–439, 2002.

[207] J. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. Inform. Theory*, vol. 50, no. 10, pp. 2231–2242, 2004.

[208] G. Varghese Detecting packet patterns at high speeds. Tutorial at *ACM SIGCOMM*, 2002.

[209] S. Venkataraman, D. Song, P. Gibbons, and A. Blum, "New streaming algorithms for superspreader detection," *Network and Distributed Systems Security Symposium*, 2005.

[210] S. Venkatasubramanian, "The graphics card as a stream computer," *Proc. ACM/DIMACS Workshop on Management and Processing of Data Streams (MPDS)*, 2003, See also: http://www.research.att.com/˜suresh/papers/mpds/index.html.

[211] L. Villemoes, "Best approximation with walsh atoms," *Constructive Approximation*, vol. 133, pp. 329–355, 1997.

[212] J. Vitter, "External memory algorithms and data structures: Dealing with massive data," *ACM Computing Surveys*, vol. 33, no. 2, pp. 209–271, 2001.

[213] J. von zur Gathen and J. Gerhard, *Modern Computer Algebra*. Cambridge University Press, 1999.

[214] A. Wong, L. Wu, P. Gibbons, and C. Faloutsos, "Fast estimation of fractal dimension and correlation integral on stream data," *Inf. Process. Lett.*, vol. 93, no. 2, pp. 91–97, 2005.

[215] D. Woodruff, "Optimal space lower bounds for all frequency moments," *Proc. ACM-SIAM SODA*, pp. 167–175, 2004.

[216] A. Yao, "Protocols for secure computations," *Proc. IEEE FOCS*, pp. 160–164, 1982.

[217] Y. Zhang, S. Singh, S. Sen, N. Duffield, and C. Lund, "Online identification of hierarchical heavy hitters: Algorithms, evaluation, and applications," *Proc. of the Internet Measurement Conference (IMC)*, pp. 101–114, 2004.

[218] G. Zipf, *Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology*. Addison Wesley, 1949.