Open Questions from the Workshop on Algorithms for Data Streams 2012 at TU-Dortmund

July 24, 2012

1 "For all" guarantee for computationally bounded adversaries (Martin Strauss)

There are two types of compressed sensing guarantees, illustrated using two players:

1. "For all": Charlie constructs the sensing matrix ϕ, and then Mallory constructs the signal $x = x(\phi)$ as a function of ϕ. The Compressed Sensing question is to recover the approximate signal $\tilde{x}$ from the measurement ϕx. The best guarantee possible is the following ℓ_2/ℓ_1 guarantee:
$$||\tilde{x} - x||_2 \leq \epsilon/\sqrt{k}||x_{opt} - x||_1.$$
2. "For each": Charlie construct a distribution D over sensing matrices ϕ. Then Mallory constructs a vector $x = x(D)$ dependent on the distribution only. Finally, a sensing matrix ϕ is sampled from the distribution D. The goal is again to recover $\tilde{x}$, with good probability over the choice of ϕ. It turns out a stronger guarantee, termed ℓ_2/ℓ_2, is possible:
$$||\tilde{x} - x||_2 \leq (1+\epsilon)||x_{opt} - x||_2$$

In some sense the two "worlds" are incomparable: the first one works for all x but obtains weaker error guarantee, and the second one works for each x with some probability but gets better error guarantee.

Question is: How can we get the best of both worlds ("for all" with ℓ_2/ℓ_2 error) ?

Once we require "for all", it is provably impossible to obtain ℓ_2/ℓ_2 guarantee. But what if Mallory has bounded computational resources to construct a "bad" x?

A preliminary result considers the following setting. Mallory sees ϕ and writes down a sketch of ϕ (in bounded space). Then Mallory produces x from this sketch only. Then ℓ_2/ℓ_2 is possible for such x's.

Generally, we would like to allow Mallory to be probabilistic polynomial time, and have a ϕ so that Mallory still cannot find an input $x = x(\phi)$ that breaks the recovery algorithm.

2 TSP in the streaming model (Christian Sohler)

We have n points living in $\{1, \ldots, \Delta\}^2$ space.

Question: Can we approximate the value of the TSP tour (Traveling Salesman Problem) of the n points when streaming over the points in one pass, using small space $(\log^{O(1)} \Delta)$.

One can achieve a $2-$approximation by computing a minimum spanning tree in small space, and use the MST to approximate TSP. The question is whether one can obtain an approximation factor $c < 2$ in polylog space?

There are other natural related question, such as computing the Earth-Mover Distance over the points in the stream [has appeared in previous open questions lists].

3 Homomorphic hash functions (Ely Porat)

Question: to construct a hash function $h : \mathbb{F}_p^n \to \mathbb{F}_p^m$, where $m < n$, satisfying the following properties:

- h is linear: $h(u + v) = h(u) + h(v)$ for all $u, v \in \mathbb{F}_p^n$;
- for any u, v, we have $\Pr_h[h(u) = h(v)] = \frac{c}{p^m}$ for some constant c independent of n, m.

One solution is by considering a random linear function, given by the matrix M. Then we have that $\Pr_M[Mu = Mv] = \Pr_M[M(u - v) = 0] = 1/p^m$. This function would require $O(nm \log p)$ random bits, and computing h takes $O(nm)$ time. We would like more efficient solutions.

Ely and coauthors claim a solution with $O((n + m) \log p)$ bits, and $O((n + m) \log(n + m))$ time.

If one considers Reed-Salomon codes, it seems that they would give worse bound on second property (probability of collision).

4 Faster JL Dimensionality Reduction (Jelani Nelson)

The standard Johnson-Lindenstrauss lemma states the following: for any $0 < \epsilon < 1/2$, any $x_1 \ldots x_n \in \mathbb{R}^d$, there exists $A \in \mathbb{R}^{k \times d}$ with $k = O(1/\epsilon^2 \cdot \log n)$, such that for any i, j we have $\|Ax_i - Ax_j\|_2 = (1 \pm \epsilon)\|x_i - x_j\|_2$.

The main question is to construct A's that admit faster computation time of Ax. There are several directions to try to obtain more efficient A:

- Fast JL (FFT-based). Here, the runtime is of the form $O(d \log d + \text{poly}(k))$ to compute Ax ($d \log d$ is usually the most significant term).

- Sparse JL. Here, the runtime is of the form $O(\epsilon k\|x\|_0 + k)$, where $\|x\|_0$ is the number of non-zero coordinates of x (i.e., it works well for sparse vectors).

Question: Can one obtain a JL matrix A, such that one can compute Ax in time $\tilde{O}(\|x\|_0 + k)$?

One possible avenue would be by considering a "random" k by k submatrix of the FFT matrix. This may or may not lead to the desired result.

5 Applications of Clifford Algebras in Graph Streams (He Sun)

Some of the recent results in graph streaming algorithms [7, 10] use *complex-valued* sketches to capture the graph structure. While it had been known earlier that integer-valued sketches can be used to count triangles, Kane et al. [7] developed a complex-valued sketch to count the number of occurrences of an arbitrary subgraph of constant size. These techniques also extend to variations of the subgraph counting problem, for instance counting a directed or (labelled) subgraph. However, the bounds on the space complexity which depends on the variance of the sketches are quite loose for most graph families.

It is interesting to compare these results to the framework of designing randomized algorithms for computing the permanent. Let A be a 0-1 matrix, and B be the matrix obtained from A by replacing each 1 uniformly and randomly with an element from a finite set D. With suitable choices of the set D, the determinant of B can be used to approximate the permanent of A. As shown by Chien et al. [2] and discussed by Muthukrishnan [12], by choosing elements of D from $\mathbf{Z}$, $\mathbf{C}$, or a Clifford algebra $\mathbf{CL}$, the variance of the estimator drops significantly each time when we move to a more "complex" algebra. It seems plausible that similar techniques can be used to improve the space complexity of graph streaming algorithms which are based on complex-valued random variables.

Question: Find suitable applications of Clifford algebra in designing algorithms in graph streams.

6 Efficient measures of *surprisingness* of sequences (Rina Panigrahy)

Consider a sequence of iid random bits $S \in \{0,1\}^n$.

Question: Find efficient measures of how surprising/unbelievable S appears to be. (Good heuristic for measuring how probable/improbably a string is.)

For example, if we see $0, 0, 0, \ldots$, we won't believe it is random (i.e., it is surprising.)

One existing measure is the (k^{th}-order) Shannon entropy H_k (H_0 would correspond to taking the entropy of the empirical frequencies of 0s and 1s). However, it fails to say that a string like $(0, 0, ...0, 1, 1, ...1)$ is surprising (from the point of view of densities it looks pretty random).

Ideal solution is to consider the Kolmogorov complexity, but it is hard (impossible) to compute.

A particular setting of the strings to consider may be: suppose each bit is generated from a biased independent coin, but the bias of the coin changes (slowly?) over time. Is there a good compression here?

7 Coding theory in the streaming model (Atri Rudra)

Consider the problem of "codeword testing" in the data stream model. In particular, consider a code

$$C : \Sigma^k \to \Sigma^n,$$

with distance[1] d. The specific problem is the following

> The input to the problem is a vector $\mathbf{y} \in \Sigma^n$ and integer parameters $0 \leq \tau_1 < \tau_2 \leq n$. The algorithm has to decide whether
>
> $$\Delta(\mathbf{y}, C) \leq \tau_1 \text{ or } \Delta(\mathbf{y}, C) \geq \tau_2.^2$$

Ideally, we want a one-pass, $\log^{O(1)} n$ space algorithm to solve the problem above for some *good* code C. (That is, we have $k \geq \Omega(n)$ and $d \geq \Omega(n)$.) Or if we prove a hardness result, one would like a hardness result for *every* good code C. (For the sake of simplicity, assume that the algorithm has access to some succinct description of the code C.)

The main technical motivation comes from the case when $\tau_1 = 0$ and $\tau_2 \geq \epsilon n$ for any fixed $\epsilon > 0$ but with *constant* number of queries to $\mathbf{y}$ (i.e. in the property testing world). This question is perhaps the open question in the codeword testing literature. The case of $\tau_1 > 0$ also makes sense in the property testing world and has been studied [5]. (See the paper for some potential practical motivations.)

One of the original motivation (in [13]) for the study of the data-streaming version of the question was possibly to use communication complexity results to prove the impossibility of good locally testable codes.

It was shown in [13] that for the well-known Reed-Solomon codes, the data stream version of the problem can be solved for $\tau_1 = 0$ and $\tau_2 = 1$ with one pass and logarithmic space. It can also be shown that the classical Berlekamp-Massey algorithm for decoding Reed-Solomon codes implies a solution for the case $\tau_2 = \tau_1 + 1$ with one pass and space $\tilde{O}(\tau_1)$.[3] Finally, [11] showed how to solve this problem in one pass and $O(k \log n)$ space. This question is wide open:

> Solve the problem above with one pass and $\tilde{O}(\min(k, \tau_1))$ space.

[1] The distance of a code C is the minimum Hamming distance between any two codewords, i.e. $\min_{\mathbf{x} \neq \mathbf{y} \in \Sigma^k} |\{i \in [n] | C(\mathbf{x})_i \neq C(\mathbf{y})_i\}|$.

[2] $\Delta(\mathbf{y}, C)$ is the Hamming distance of $\mathbf{y}$ from the closest codeword in C.

[3] There is a small catch: the algorithm actually computes the location of errors *if* the number of errors is at most τ_1. However, results in [13] can be used to verify if the returned error locations are indeed correct.

In fact the very special case of the problem above for $k = \tau_1 = \sqrt{n}$ with one pass and space $o(\sqrt{n})$ is also open. This is open even for the special case of Reed-Solomon codes.

8 Signatures for set equality (Rasmus Pagh)

Given $S \subseteq \{1, ..n\}$, we would like to construct a fingerprint so that later, given fingerprints of two sets, we can check the equality of the two sets.

There are (at least) two possible solutions to the problem:

1. $h(S) = \sum_{i \in S} x^i \mod p$ for random $x \in \mathbb{Z}_p$. Update time would be roughly $\log p = \Omega(n)$. One would like to obtain a better update time.
2. $h(S) = \prod_{i \in S}(x - i) \mod p$, and random x. Insertion can be done in constant time. But the fingerprint is not linear.

Question: Can we construct a fingerprint that achieves constant update time and is linear, while using $O(\log n)$ random bits? Ideally updates would include insertions and deletions.

9 Low Expansion Encoding of Edit Distance (Hossein Jowhari)

Let $T = \bigcup_{i=1}^{n} \{0,1\}^i$. For pair of strings $(x, y) \in T \times T$ let $ed(x, y)$ denote the edit distance between x and y which is defined as the minimum number of character insertion, deletion and substitution needed for converting x into y. Is there a mapping $f : T \to \{0,1\}^m$ satisfying the following conditions

- f is injective, i.e. it does not map different inputs to the same point.
- $m = O(n^c)$ for some constant $c \geq 1$.
- For strings with $ed(x, y) = 1$ we have $\mathcal{H}(f(x), f(y)) \leq C$ for $C = o(\log n)$.

The same question holds for randomized mappings as long as they map different x and y to different points with high probability. Currently the best upper bound on C is $O(\log n \log^* n)$ achieved through a randomized mapping that deploys the Locally Consistent Parsing method [3]. For non-repetitive strings (the Ulam distance) there is a deterministic mapping with $C \leq 6$ and $c = 2$. Preferably we would like to have mappings that are efficiently computable and are equipped with polynomial time decoding algorithms (x can be obtained from $f(x)$ efficiently). See [6] for motivations on the problem.

10 Single-Pass Unweighted Matchings (Andrew McGregor)

Suppose you have $O(n\text{polylog}n)$ memory and a single pass over a stream of m edges (arbitrarily ordered) on n nodes. How well can you approximate the size of the maximum cardinality matching. A trivial greedy algorithm finds a 1/2-approximation but that's still the best known algorithm in the general setting. Kapralov [8] showed that achieving better than a $1 - 1/e$ approximation is impossible. If the stream is randomly ordered, Konrad et al. [9] presented a $1/2+0.005$-approximation. Other variants of the question are also open, e.g., achieving a $(1 - \epsilon)$ approximation with multiple passes (see, e.g., Ahn and Guha [1]) or the best approximation possible for maximum weighted matching in a single pass (see, e.g., Epstein et al. [4]).

References

[1] Kook Jin Ahn and Sudipto Guha. Laminar families and metric embeddings: Non-bipartite maximum matching problem in the semi-streaming model. *CoRR*, abs/1104.4058, 2011.

[2] Steve Chien, Lars Eilstrup Rasmussen, and Alistair Sinclair. Clifford algebras and approximating the permanent. *J. Comput. Syst. Sci.*, 67(2):263–290, 2003.

[3] Graham Cormode, Mike Paterson, Süleyman Cenk Sahinalp, and Uzi Vishkin. Communication complexity of document exchange. In *SODA*, pages 197–206, 2000.

[4] Leah Epstein, Asaf Levin, Julián Mestre, and Danny Segev. Improved approximation guarantees for weighted matching in the semi-streaming model. *SIAM J. Discrete Math.*, 25(3):1251–1265, 2011.

[5] Venkatesan Guruswami and Atri Rudra. Tolerant locally testable codes. In *Proceedings of the 9th InternationalWorkshop on Randomization and Computation (RANDOM)*, pages 306–317, 2005.

[6] Hossein Jowhari. Efficient communication protocols for deciding edit distance. In *ESA*, 2012.

[7] Daniel M. Kane, Kurt Mehlhorn, Thomas Sauerwald, and He Sun. Counting arbitrary subgraphs in data streams. In *Proceedings of the 39th International Colloquium on Automata, Languages, and Programming (2)*, pages 598–609, 2012.

[8] Michael Kapralov. Improved lower bounds for matchings in the streaming model. *CoRR*, abs/1206.2269, 2012.

[9] Christian Konrad, Frédéric Magniez, and Claire Mathieu. Maximum matching in semi-streaming with few passes. In *Proceedings of 15th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, 2012.

[10] Madhusudan Manjunath, Kurt Mehlhorn, Konstantinos Panagiotou, and He Sun. Approximate counting of cycles in streams. In *Proceedings of the 19th Annual European Symposium*, pages 677–688, 2011.

[11] Andrew McGregor, Atri Rudra, and Steve Uurtamo. Polynomial fitting of data streams with applications to codeword testing. In *Proceedings of the 28th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 428–439, 2011.

[12] S. Muthukrishnan. Data streams: Algorithms and applications. *Foundations and Trends in Theoretical Computer Science*, 1(2), 2005.

[13] Atri Rudra and Steve Uurtamo. Data stream algorithms for codeword testing. In *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 629–640, 2010.