Sublinear Algorithms in the External Memory Model[*]

Alexandr Andoni
Princeton University, CCI

Piotr Indyk
MIT

Krzysztof Onak
MIT

Ronitt Rubinfeld
MIT, Tel Aviv University

February 16, 2010

Abstract

We initiate the study of sublinear-time algorithms in the external memory model [Vit01]. In this model, the data is stored in blocks of a certain size B, and the algorithm is charged a unit cost for each block access. This model is well-studied, since it reflects the computational issues occurring when the (massive) input is stored on a disk. Since each block access operates on B data elements in parallel, many problems have external memory algorithms whose number of block accesses is only a small fraction (e.g. $1/B$) of their main memory complexity.

However, to the best of our knowledge, no such reduction in complexity is known for *any* sublinear-time algorithm. One plausible explanation is that the vast majority of sublinear-time algorithms use random sampling and thus exhibit no locality of reference. This state of affairs is quite unfortunate, since both sublinear-time algorithms and the external memory model are important approaches to dealing with massive data sets, and ideally they should be combined to achieve best performance.

We show that such combination is indeed possible. In particular, we consider three well-studied problems: testing of *distinctness*, *uniformity* and *identity* of an empirical distribution induced by data. For these problems we show random-sampling-based algorithms whose number of block accesses is up to a factor of $1/\sqrt{B}$ smaller than the main memory complexity of those problems. We also show that this improvement is optimal for those problems.

Since these problems are natural primitives for a number of sampling-based algorithms for other problems, our tools improve the external memory complexity of other problems as well.

1 Introduction

Random sampling is one of the most fundamental methods for reducing task complexity. For a wide variety of problems, it is possible to infer an approximate solution from a random sample containing only a small fraction of the data, yielding algorithms with sublinear running times. As a result, sampling is often the method of choice for processing massive data sets. Inferring properties of data from random sample has been a major subject of study in several areas, including statistics, databases [OR86, Olk93], theoretical computer science [Fis01, Ron01, Gol98, BKS01], . . .

However, using random sampling for massive data sets encounters the following problem: typically, massive data sets are not stored in main memory, where each element can be accessed at a unit cost. Instead, the data is stored on external storage devices, such as a hard disk. There,

[*] The research was supported in part by David and Lucille Packard Fellowship, by MADALGO (Center for Massive Data Algorithmics, funded by the Danish National Research Association), by Marie Curie IRG Grant 231077, by NSF grants 0514771, 0728645, and 0732334, and by a Symantec Research Fellowship.

the data is stored in blocks of certain size (say, B), and each disk access returns a block of data, as opposed to an individual element. In such models [Vit01], it is often possible to solve problems using roughly T/B disk accesses, where T is the time needed to solve the problem in main memory. The $1/B$ factor is often crucial to the efficiency of the algorithms, given that (a) the block size B tends to be large, on the order of thousands and (b) each block access is many orders of magnitude slower than a main memory lookup. Unfortunately, implementations of sampling algorithms typically need to perform[1] one block access per each sampled element [OR86]. Effectively, this means that out of B data elements retrieved by each block access, $B-1$ elements are discarded by the algorithm. This makes sampling algorithms a much less attractive option for processing massive data sets.

Is it possible to improve the sampling algorithms by utilizing the *entire* information stored in each accessed block? At the first sight, it might not seem so. For example, consider the following basic sampling problem: the input data is a binary sequence such that the fraction of ones is either at most f or at least $2f$, and the goal is to detect which of these two cases occurs. A simple argument shows that any sampling algorithm for this problem requires $\Omega(1/f)$ samples to succeed with constant probability, since it may take that many trials to even retrieve one 1. It is also easy to observe that the same lower bound holds even if all elements within each block are equal (as long as the total number of blocks is $\Omega(1/f)$), in which case sampling blocks is equivalent to sampling elements. Thus, even for this simple problem, sampling blocks does not yield any reduction in the number of accesses.

2 Our Results

Contrary to the above impression, we show that there are natural problems for which it is possible to reduce the number of sampled blocks. Specifically, we consider the problem of testing properties of empirical distributions induced by the data sets. Consider a data set of size m with support size (i.e., the number of distinct elements) equal to n. Let p_i be the fraction of times an element i occurs in the data set. The vector p then defines a probability distribution over a set of distinct elements in the data set. We address the following three well-studied problems:

- Distinctness: are all data elements distinct (i.e., $n = m$), or are there at least ϵm duplicates?
- Uniformity: is p uniform over its support, or is it ϵ-far[2] from the uniform distribution?
- Identity: is p identical to an explicitly given distribution q, or is it ϵ-far from q?

Note that testing identity generalizes the first two problems. However, the algorithms for distinctness and uniformity are simpler and easier to describe.

It is known [GR00, Bat01, BFR$^+$00] that, if the elements are stored in main memory, then $\tilde{\Theta}(\sqrt{n})$ memory accesses are sufficient and necessary to solve both uniformity and identity testing. We give an external memory algorithm which uses only $\tilde{O}(\sqrt{m/B})$ block accesses. Thus, for m comparable to n, the number of accesses is reduced by a factor of $\sqrt{B}$. It also can be seen that this bound cannot be improved in general: if $B = m/n$, then each block could consist of equal elements, and thus the $\tilde{\Theta}(\sqrt{n}) = \tilde{\Theta}(\sqrt{m/B})$ main memory lower bound would apply.

[1] It is possible to retrieve more samples per block if the data happens to be stored in a random order. Unfortunately, this is typically not guaranteed.

[2] We measure the distance between distribution using the standard variational distance, which is the maximum probability with which a statistical test can distinguish the two distributions. Formally, a distribution p is ϵ-far from a distribution q, if $\|p - q\|_1 \geq \epsilon$, where p and q are interpreted as vectors.

From the technical perspective, our algorithms mimic the sampling algorithms of [BFR$^+$00, BFF$^+$01, Bat01]. The key technical contribution is a careful analysis of those algorithms. In particular, we show that the additional information obtained from sampling blocks of data (as opposed to the individual elements) yields a substantial reduction of the variance of the estimators used by those algorithms.

3 Applications to Other Problems

The three problems from above are natural primitives for a number of other sampling-based problems. Thus, our algorithms improve the external memory complexity of other problems as well. Below we describe two examples of problems where our algorithms and techniques apply immediately to give improved guarantees in the external memory model.

The first such problem is testing graph isomorphism. In this problem, the tester is to decide, given two graphs G and H on n vertices, whether G are H are isomorphic or at least ϵn^2 edges of the graphs must be modified to achieve a pair of isomorphic graphs. Suppose one graph, G, is known to the tester (for instance, it is a fixed graph with an easily computable adjacency relation), and the other graph, H, is described by the adjacency matrix written in the row-major order on the disk. Then, our algorithm for identity testing improves the sample complexity of the Fischer and Matsliah algorithm [FM08] by essentially a factor of $\sqrt{B}$. Formally, in the main memory, the Fischer and Matsliah algorithm uses $O(\sqrt{n} \cdot \mathrm{poly}(\log n, 1/\epsilon))$ queries to H. Combined with our external memory identity tester, algorithm will use only $O((\sqrt{n/B}+1) \cdot \mathrm{poly}(\log n, 1/\epsilon))$ samples.

The second application is a set of questions on testing various properties of metric spaces, such as testing whether a metric is a tree-metric or ultra-metric. In [Ona08], Onak considers several such properties, for which he gives algorithms whose sampling complexity in main memory is of the form $O(\alpha/\epsilon + n^{(\beta-1)/\beta}/\epsilon^{1/\beta})$, where $\alpha \geq 1$ and $\beta \geq 2$ are constant integers. The additive term $n^{(\beta-1)/\beta}/\epsilon^{1/\beta}$ corresponds to sampling for a specific β-tuple. Using our techniques for distinctness testing, it can easily be shown that whenever an algorithm from [Ona08] requires $O(\alpha/\epsilon + n^{(\beta-1)/\beta}/\epsilon^{1/\beta})$ samples, the sample complexity in external memory can be improved to $O(\alpha/\epsilon + (n/B)^{(\beta-1)/\beta}/\epsilon^{1/\beta})$, provided a single disk block contains B points.

References

[Bat01] Tugkan Batu. *Testing Properties of Distributions.* PhD thesis, Cornell University, August 2001.

[BFF$^+$01] Tugkan Batu, Lance Fortnow, Eldar Fischer, Ravi Kumar, Ronitt Rubinfeld, and Patrick White. Testing random variables for independence and identity. In *FOCS*, pages 442–451, 2001.

[BFR$^+$00] Tugkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren D. Smith, and Patrick White. Testing that distributions are close. In *FOCS*, pages 259–269, 2000.

[BKS01] Ziv Bar-Yossef, Ravi Kumar, and D. Sivakumar. Sampling algorithms: lower bounds and applications. In *STOC*, pages 266–275, 2001.

[Fis01] Eldar Fischer. The art of uninformed decisions: A primer to property testing. *Bulletin of the European Association for Theoretical Computer Science*, 75:97–126, 2001.

[FM08] Eldar Fischer and Arie Matsliah. Testing graph isomorphism. *SIAM J. Comput.*, 38(1):207–225, 2008.

[Gol98] Oded Goldreich. Combinatorial property testing—a survey. In *Randomization Methods in Algorithm Design*, pages 45–60, 1998.

[GR00] Oded Goldreich and Dana Ron. On testing expansion in bounded-degree graphs. *Electronic Colloqium on Computational Complexity*, 7(20), 2000.

[Olk93] Frank Olken. *Random Sampling from Databases.* PhD thesis, 1993.

[Ona08] Krzysztof Onak. Testing properties of sets of points in metric spaces. In *ICALP (1)*, pages 515–526, 2008.

[OR86] Frank Olken and Doron Rotem. Simple random sampling from relational databases. In *VLDB*, pages 160–169, 1986.

[Ron01] Dana Ron. Property testing (a tutorial). In S. Rajasekaran, P. M. Pardalos, J. H. Reif, and J. D. P. Rolim, editors, *Handbook on Randomization, Volume II*, pages 597–649. Kluwer Academic Press, 2001.

[Vit01] Jeffrey Scott Vitter. External memory algorithms and data structures. *ACM Comput. Surv.*, 33(2):209–271, 2001.