Compact Summaries over Large Datasets

Graham Cormode
University of Warwick
G.Cormode@Warwick.ac.uk

ABSTRACT

A fundamental challenge in processing the massive quantities of information generated by modern applications is in extracting suitable representations of the data that can be stored, manipulated and interrogated on a single machine. A promising approach is in the design and analysis of compact summaries: data structures which capture key features of the data, and which can be created effectively over distributed data sets. Popular summary structures include the count distinct algorithms, which compactly approximate item set cardinalities, and sketches which allow vector norms and products to be estimated. These are very attractive, since they can be computed in parallel and combined to yield a single, compact summary of the data. This tutorial introduces the concepts and examples of compact summaries.

Categories and Subject Descriptors

E.1 [**Data**]: Data Structures

General Terms

Algorithms, Theory

Keywords

summaries, sketches, approximate counting

PODS'15, May 31–June 4, 2015, Melbourne, Victoria, Australia.

http://dx.doi.org/10.1145/2745754.2745781.

1. INTRODUCTION

Business and scientific communities all agree that "big data" holds both tremendous promise, and substantial challenges [8]. There is much potential for extracting useful intelligence and actionable information from the large quantities of data generated and captured by modern information processing systems. Big data challenges involve not only the sheer volume of the data, but the fact that it can represent a complex variety of entities and interactions between them, and new observations that arrive, often across multiple locations, at high velocity. Examples of applications that generate big data include:

Physical Data from sensor deployments and scientific experiments—astronomy data from modern telescopes generates terabytes of data each night, while the data collected from a single particle physics experiment is too big to store;

Medical Data, as we can now sequence whole genomes economically, generating data sets of the order of 200TB in one example [7];

Activity Data, as human activity data is captured and stored in ever greater quantities and detail: interactions from online social networks, locations from GPS, Internet activity etc.

Across all of these disparate settings, certain common themes emerge. The data in question is large, and growing. The applications seek to extract patterns, trends or descriptions of the data. Ensuring the scalability of systems, and the timeliness and veracity of the analysis is vital in many of these applications. In order to realize the promise of these sources of data, we need new methods that can handle them effectively.

While such sources of big data are becoming increasingly common, the resources to process them (chiefly, processor speed, fast memory and slower disk) are growing at a slower pace. The consequence of this trend is that there is an urgent need for more effort directed towards capturing and processing data in many critical applications. Careful planning and scalable architectures are needed to fulfill the requirements of analysis and information extraction on big data. In response to these needs, new computational paradigms are being adopted to deal with the challenge of big data. Large scale distributed computation is a central piece: the scope of the computation can exceed what is feasible on a single machine, and so clusters of machines work together in parallel. On top of these architectures, parallel algorithms are designed which can take the complex task and break it into independent pieces suitable for distribution over multiple machines.

A central challenge within any such system is how to compute and represent complex features of big data in a way that can be processed by many single machines in parallel. A vital component is to be able to build and manipulate a *compact summary* of a large amount of data. This powerful notion of a small summary, in all

its many and varied forms, is the subject of this tutorial. The idea of a summary is a natural and familiar one. It should represent something large and complex in a compact fashion. Inevitably, a summary must dispense with some of the detail and nuance of the object which it is summarizing. However, it should also preserve some key features of the object in a very accurate fashion. Effective compact summaries are often *approximate* in their answers to queries and *randomized*.

The theory of compact summaries can be traced back over four decades. A first example is the Morris Approximate Counter, which approximately counts quantities up to magnitude n using $O(\log \log n)$ bits, rather than the $\lceil \log n \rceil$ bits to count exactly [15]. Subsequently, there has been much interest in summaries in the context of *streaming algorithms*: these are algorithms that process data in the form of a stream of updates, and whose associated data structures can be seen as a compact summary [16]. More recently, the more general notion of *mergeable summaries* has arisen: summaries that can be computed on different portions of a dataset in isolation, then subsequently combined to form a summary of the union of the inputs [1]. It turns out that a large number streaming algorithms entail a mergeable summary, hence making this class of objects a large and interesting one.

There has been much effort expended on summary techniques over recent years, leading to the invention of powerful and effective summaries which have found applications in Internet Service Providers [5], Search Engines [17, 12], and beyond.

2. TUTORIAL OUTLINE

This short tutorial will introduce the notion of summaries, and outline ideas behind some of the most prominent examples, which may include:

- Counts, approximate counts [15], and approximate frequencies [14]
- Count distinct, set cardinality, and set operations [9, 10]
- Random projections with low-independence vectors to give *sketch* data structures [3, 4, 6]
- Summaries for medians and order statistics [11, 13]
- Linear summaries for graphs: connectivity, bipartiteness and sparsification [2]
- Summaries for matrix and linear algebra operations [18]
- Problems for which no compact summary can exist, via communication complexity lower bounds.

Acknowledgments

This work supported in part by a Royal Society Wolfson Research Merit Award, funding from the Yahoo Research Faculty Research and Engagement Program, and European Research Council (ERC) Consolidator Grant ERC-CoG-2014-647557.

3. REFERENCES

[1] Pankaj Agarwal, Graham Cormode, Zengfeng Huang, Jeff Phillips, Zheiwei Wei, and Ke Yi. Mergeable summaries. In *ACM Principles of Database Systems*, 2012.

[2] Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Analyzing graph structure via linear measurements. In *ACM-SIAM Symposium on Discrete Algorithms*, 2012.

[3] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. In *ACM Symposium on Theory of Computing*, pages 20–29, 1996.

[4] M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. In *Procedings of the International Colloquium on Automata, Languages and Programming (ICALP)*, 2002.

[5] G. Cormode, F. Korn, S. Muthukrishnan, T. Johnson, O. Spatscheck, and D. Srivastava. Holistic UDAFs at streaming speeds. In *ACM SIGMOD International Conference on Management of Data*, pages 35–46, 2004.

[6] G. Cormode and S. Muthukrishnan. An improved data stream summary: The Count-Min sketch and its applications. *Journal of Algorithms*, 55(1):58–75, 2005.

[7] Kathleen Cravedi, Tera Randall, and Larry Thompson. 1000 genomes project data available on Amazon Cloud. *NIH News*, March 2012.

[8] Kenneth Cukier. Data, data everywhere. *The Economist*, February 2010.

[9] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for database applications. *Journal of Computer and System Sciences*, 31:182–209, 1985.

[10] Philippe Flajolet, É. Fusy, Olivier Gandouet, and Frederic Meunier. Hyperloglog: The analysis of a near-optimal cardinality estimation algorithm. In *International Conference on Analysis of Algorithms*, 2007.

[11] M. Greenwald and S. Khanna. Space-efficient online computation of quantile summaries. In *ACM SIGMOD International Conference on Management of Data*, 2001.

[12] Sergey Melnik, Andrey Gubarev, Jing Jing Long, Geoffrey Romer, Shiva Shivakumar, Matt Tolton, and Theo Vassilakis. Dremel: Interactive analysis of web-scale datasets. In *International Conference on Very Large Data Bases*, pages 330–339, 2010.

[13] A. Metwally, D. Agrawal, and A. El Abbadi. Efficient computation of frequent and top-k elements in data streams. In *International Conference on Database Theory*, 2005.

[14] J. Misra and D. Gries. Finding repeated elements. *Science of Computer Programming*, 2:143–152, 1982.

[15] Robert Morris. Counting large numbers of events in small registers. *Communications of the ACM*, 21(10):840–842, 1977.

[16] S. Muthukrishnan. *Data Streams: Algorithms and Applications*. Now Publishers, 2005.

[17] Rob Pike, Sean Dorward, Robert Griesemer, and Sean Quinlan. Interpreting the data: Parallel analysis with sawzall. *Dynamic Grids and Worldwide Computing*, 13(4):277–298, 2005.

[18] David Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014.