ISSN 2186-7437

NII Shonan Meeting Report

No. 2012-1

Large-Scale Distributed Computation

Graham Cormode
S. Muthukrishnan
Ke Yi

January 11–15, 2012

National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-Ku, Tokyo, Japan

Large-Scale Distributed Computation

Organizers:
Graham Cormode (AT&T Labs)
S. Muthukrishnan (Rutgers University)
Ke Yi (Hong Kong University of Science and Technology)

January 11–15, 2012

As the amount of data produced by large scale systems such as environmental monitoring, scientific experiments and communication networks grows rapidly, new approaches are needed to effectively process and analyze such data. There are many promising directions in the area of large-scale distributed computation, that is, where multiple computing entities work together over partitions of the (huge) data to perform complex computations. Two important paradigms in this realm are distributed continuous monitoring, which continually maintains an accurate estimate of a complex query, and MapReduce, a primarily batch approach to large cluster computation. The aim of this meeting is to bring together computer scientists with interests in this field to present recent innovations, find topics of common interest and to stimulate further development of new approaches to deal with massive data.

If computer science in the 20th century was defined by the inception and astounding growth of computing power, in the 21st century it will be defined by the huge growth in data. The ability to capture vast quantities of data (from network traffic, scientific experiments, online activity) is expanding rapidly, and at a rate far higher than the increase in our ability to process it. While Moore's law has held for many decades, tracking an exponential growth in computational power, the corresponding law for data has recently shown an even faster growth in data production. Therefore, the new challenge for computer science in the 21st century is how to deal with such a data deluge. The following promising new directions have emerged in recent years:

- *Distributed Continuous Monitoring.* In many settings the new data is observed locallyat a router in a network, at a sensor in a larger sensor network. The volume of observations is too large to move to a central location and process together; instead, it is necessary to perform distributed computation over the data. Since the new observations are continually arriving, we must produce a continual answer to complex monitoring queries, all while ensuring that the communication cost necessary to maintain the result, and the computational cost of the tracking, are minimized to meet the data throughput demands.

- *Cluster Computing.* As data sizes increase while the power of individual computing cores begin to saturate, there has been a move to adopt cluster computing: harnessing multiple computing nodes to work together to

solve huge data processing tasks in parallel. The best known example of this is the MapReduce paradigm, and its open source Hadoop implementation. Computationally speaking, the approach is for each compute node to process data which is stored local to it in a distributed file system, and Map this data by deriving a new data set indexed by a key value. The system then collects all tuples with the same key together at a second node, which proceeds to Reduce these tuples to obtain the (partial) output. This paradigm has proved highly successful for many large scale computations, such as search engine log-analysis, building large machine learning models and data analysis such as clustering.

The aim of this workshop is to bring together researchers active in the areas of distributed data processing, to address these fundamental issues. We hope to encourage greater cooperation and interaction between currently separate communities, ultimately leading to new advances in these important developing areas.

Overview of Talks

Transitive Closure and Recursive Datalog Implemented on Clusters

Foto Afrati

Implementing recursive algorithms using a distributed computational environment that is an extension of map-reduce presents new challenges. A central problem is that recursive tasks cannot deliver their output only at the end, which makes recovery from failures much more complicated than in map-reduce and its nonrecursive extensions. Another is the endgame problem: later rounds of a recursion often transfer only small amounts of data, causing high overhead for interprocessor communication. I will focus in the endgame problem. One way to deal with it is to use an algorithm that reduces the number of rounds of the recursion. I will present a number of algorithms for transitive closure that are efficient also as regards the amount of data transmitted among the compute nodes because they have the "unique decomposition" property. Then I will present results about reducing the number of rounds for more general Datalog programs and show also that there are cases where you can do so efficiently and cases where you can reduce the number of rounds but only at the expense of increasing the arity of the intermediately derived predicates.

(This is joint work with Jeff Ullman.)

Continuous Distributed Monitoring: A short survey

Graham Cormode

In the model of continuous distributed monitoring, a number of observers each see a stream of observations. Their goal is to work together to compute a function of the union of their observations. This can be as simple as counting the total number of observations, or more complex non-linear functions such as tracking the entropy of the induced distribution. Assuming that it is too costly to simply centralize all the observations, it becomes quite challenging to design solutions which provide a good approximation to the current answer, while bounding the communication cost of the observers, and their other resources such as their space usage. This survey introduces this model, and describe a selection results in this setting, from the simple counting problem to a variety of other functions that have been studied.

From the Trenches of Continuous Monitoring of Multitenant DBMSs for Consolidation and Load-balancing

Amr El Abbadi

Cloud platforms handle large numbers of applications with small data footprints, different types of workloads, and unpredictable load characteristics. A multitenant database management system (DBMS) storing and serving the data for these applications (tenants) is therefore an important component of the platform's software stack. Continuous monitoring of tenant and node performance

is critical for resource provisioning and elastic load balancing to ensure effective resource utilization and minimize operational cost. A self-managing system controller faces multiple algorithmic and system level challenges; including, characterizing a tenant given the variety of workloads a tenant might issue, reducing the impact of co-locating multiple tenants, and adapting to changes in tenant behavior. In this presentation, I will discuss some of the challenges we have faced in the trenches of designing a self-managing coordinating component that monitors system performance, controls tenant placement, and manages elasticity. Our approach attempts to "learn" tenant behavior through observation and analysis of the tenants. We use a collection of system level and DBMS agnostic database level performance measures and machine learning techniques to characterize tenant and node behavior. We are also developing a combination of reactive mechanisms and pro-active actions (when possible) to detect and mitigate performance crises arising from unpredictable workloads and behavior.

This work was done jointly with Divy Agrawal, Sudipto Das and Aaron Elmore.

Geometric Query Tracking Using Sketches and Models

Minos Garofalakis

The geometric method offers a generic methodology for non-linear distributed threshold monitoring tasks. Still, the approach relies on maintaining complete state at each site and assumes static estimates of the data, thus raising efficiency concerns for massive, dynamic data streams. In this talk, I will discuss novel query tracking and threshold monitoring schemes that rely on extending the geometric method with compact sketch summaries and dynamic prediction models for the local data streams. Time permitting, I will also discuss some of our current research directions and open problems in this space.

Algorithms for Distributed Stream Processing

Ashish Goel

An Active Distributed Hash Table (Active DHT) is a distributed system that stores key-value pairs in distributed main memory and allows arbitrary User Defined Functions (UDFs) to be executed on any key-value pair. An Active DHT can be viewed as a streaming version of the popular Map-Reduce paradigm. We will outline algorithms for Locality Sensitive Hashing, Social Search, PageRank computation, and Graph Sparsification in this model. In each case, we will show a significant improvement over the straightforward distributed implementations.

Safe-Zones for Distributed Monitoring

Daniel Keren

Many monitoring tasks over distributed data streams can be formulated as a continuous threshold query on a function defined over the global average of the stream values. A fundamental problem in efficient monitoring is to correctly identify all threshold crossings while minimizing communication. We present

a novel scheme for communication reduction in distributed monitoring using local constraints. Communication is required only in the event that the local constraints are violated by the incoming data. As opposed to previous work on geometric monitoring, we present here local constraints which are tailored to fit the local data distribution at each stream. Also, we attempt to define local constraints which are as simple as possible. The result is a substantial decrease in the required volume of communication compared to previous state of the art, up to two orders of magnitude in experiments with real-life data. Further theoretical analysis reveals that the reduction can sometimes be unbounded, and that it typically improves with the dimensionality of the data, which is borne out in the experiments.

Large-scale Parallel Best-First Search for Optimal Planning

Akihiro Kishimoto

Large-scale, parallel clusters composed of commodity processors are increasingly available, enabling the use of vast processing capabilities and distributed RAM to solve hard search problems. I investigate a parallel algorithm for optimal sequential planning, with an emphasis on exploiting distributed memory computing clusters. The scaling behavior of the algorithm is evaluated experimentally on clusters using up to 1024 processors. I show that this approach scales well, allowing us to effectively utilize the large amount of distributed memory to optimally solve problems which require a terabyte of RAM to solve.

Filtering: A Method for Solving Graph Problems in MapReduce

Silvio Lattanzi

The MapReduce framework is currently the de facto standard used throughout both industry and academia for petabyte scale data analysis. As the input to a typical MapReduce computation is large, one of the key requirements of the framework is that the input cannot be stored on a single machine and must be processed in parallel. In this talk we describe a general algorithmic design technique in the MapReduce framework called "filtering". The main idea behind "filtering" is to reduce the size of the input in a distributed fashion so that the resulting, much smaller, problem instance can be solved on a single machine. Using this approach we give new algorithms in the MapReduce framework for a variety of fundamental graph problems for suffficiently dense graphs. Specifically, we present algorithms for minimum spanning trees, maximal matchings, approximate weighted matchings, approximate vertex and edge covers and minimum cuts. In all of these cases, we parameterize our algorithms by the amount of memory available on the machines allowing us to show tradeoffs between the memory available and the number of MapReduce rounds. For each setting we will show that even if the machines are only given substantially sublinear memory, our algorithms run in a constant number of MapReduce rounds. To demonstrate the practical viability of our algorithms we implement the maximal matching algorithm that lies at the core of our analysis and show that it achieves a significant speedup over the sequential version.

Biconnected components in MapReduce, using Graph (Navigational) Sketches

Luigi Laura

Graph Sketches

Andrew McGregor

Overlapping Clustering and Distributed Computation

Vahab Mirrokni

I will discuss overlapping clustering and distributed computation for large-scale social networks. The talk has two parts. One part discusses the use of overlapping clustering techniques in distributed computation and in particular the computation of PageRank vectors in a distributed infrastructure. The other part describes implementation and results of a large-scale overlapping clustering algorithm. For example we discuss an application of such community discovery over Youtube videos using the co-watched graph in which we manage to cluster a graph of 150 million nodes in 8 hours.

Fast Clustering Using MapReduce

Benjamin Moseley

Clustering problems have numerous applications and are becoming more challenging as the size of the data increases. In this paper, we consider designing clustering algorithms that can be used in MapReduce, the most popular programming environment for processing large datasets. We focus on the practical and popular clustering problems, k-center and k-median. We develop fast clustering algorithms with constant factor approximation guarantees. From a theoretical perspective, we give the first analysis that shows several clustering algorithms are in MRC^0, a theoretical MapReduce class introduced by Karloff et al. Our algorithms use sampling to decrease the data size and they run a time consuming clustering algorithm such as local search or Lloyds algorithm on the resulting data set. Our algorithms have sufficient flexibility to be used in practice since they run in a constant number of MapReduce rounds. We complement these results by performing experiments using our algorithms. We compare the empirical performance of our algorithms to several sequential and parallel algorithms for the k-median problem. The experiments show that our algorithms solutions are similar to or better than the other algorithms solutions. Furthermore, on data sets that are sufficiently large, our algorithms are faster than the other parallel algorithms that we tested.

Appeared in KDD 2011. Joint work with Alina Ene and Sungjin Im

Mergeable Summaries

Jeff M. Phillips

We study the *mergeability* of data summaries. Informally speaking, mergeability requires that, given two summaries on two data sets, there is a way to merge the two summaries into a single summary on the union of the two data sets, while preserving the error and size guarantees. This property means that the summaries can be merged in a way like other algebraic operators such as sum and max, which is especially useful for computing summaries on massive distributed data. Several data summaries are trivially mergeable by construction, most notably all the sketches that are linear functions of the data sets. But some other fundamental ones like those for heavy hitters and quantiles, are not (known to be) mergeable. In this paper, we demonstrate that these summaries are indeed mergeable or can be made mergeable after appropriate modifications. Specifically, we show that for eps-approximate heavy hitters, there is a deterministic mergeable summary of size $O(1/\epsilon)$; for ϵ-approximate quantiles, there is a deterministic summary of size $O((1/\epsilon)\log(\epsilon n))$ that has a restricted form of mergeability, and a randomized one of size $O((1/\epsilon)\log^{3/2}(1/\epsilon))$ with full mergeability. We also extend our results to geometric summaries such as eps-approximations and eps-kernels.

We also achieve two results of independent interest: (1) we provide the best known randomized streaming bound for eps-approximate quantiles that depends only on ϵ, of size $O((1/\epsilon)\log^{3/2}(1/\epsilon))$, and (2) we demonstrate that the MG and the SpaceSaving summaries for heavy hitters are isomorphic.

Joint work with: Pankaj K. Agarwal, Graham Cormode, Zengfeng Huang, Zhewei Wei, and Ke Yi.

Geometric Monitoring of Distributed Streams

Assaf Schuster

Monitoring data streams in a distributed system has been the focus of much recent research. Most of the proposed schemes, however, deal with monitoring simple aggregated values, such as the frequency of appearance of items in the streams. More involved challenges, such as feature selection (e.g., by monitoring the information gain of various features), still require very high communication overhead using naive, centralized algorithms. I will present a geometric approach which reduces monitoring the value of a function to a set of constraints applied locally on each of the streams. The constraints are used to locally filter out data increments that do not affect the monitoring outcome, thus avoiding unnecessary communication. I will discuss both the basic approach we introduced in 2007 and some recent extensions.

Monitoring the Median and Related Functions

Izchak Sharfman

The median function has numerous applications in computer science and robust estimation. In the context of stream monitoring, it is applied to the

output of sketches in order to obtain tight and reliable approximations to the underlying stream data. I will present a computationally efficient algorithm for monitoring the median of linear and quadratic functions defined over streams, which correspond to the problems of tracking range queries, self-joins, and joins in streaming data.

Scheduling a Google data center

Cliff Stein

Complexity of Finding a Duplicate in a Stream

Jun Tarui

Continuous Distributed Monitoring and Related Models

Srikanta Rirthapura

The recently introduced distributed continuous monitoring model seems a good fit for cyber-security applications, among others. I will discuss some additional requirements in such applications, and how the model maybe enhanced or modified to meet these requirements. For concreteness, I will discuss these in the context of a basic monitoring problem, that of maintaining a random sample of the stream.

One-Pass Map-Reduce Algorithms

Jeffrey Ullman

I shall review a number of algorithms that use a single round of map-reduce to solve an interesting problem. Such algorithms can be evaluated both in terms of their processing cost at the mappers and reducers and their communication cost (the total number of key-value pairs that must be passed from mappers to reducers). Depending on the computing environment, either can be the dominant cost. For the problem of fuzzy joins (finding pairs of elements that are within some minimum distance of each other), we offer a spectrum of algorithms that offer different communication/computation trade-offs. For the problem of enumerating triangles or more complex structures ("sample graphs") in a large data graph, we use a one-round map-reduce implementation of conjunctive queries from Afrati and U. (EDBT, 2010) to minimize communication cost. To minimize computation cost for the same class of problems, we offer "convertible algorithms" (serial algorithms that can be implemented at the reducers without an order-of-magnitude increase in total computation cost). For general data graphs, there are convertible algorithms that meet the lower bounds established by Noga Alon (IJM, 1981) and for all connected sample graphs there is a convertible algorithm that runs in time $O(m^{p/2})$, where p is the number of nodes in the sample graph and m is the number of edges in the data graph.

[Work variously coauthored with Foto Aftrati, Anish das Sarma, Dimitris Fotakis, David Menestrina, and Aditya Parameswaran]

Fast Algorithms for Big-data

Takeaki Uno

One of recent business/research topic is big data. The big data helps us analysis of the data, for example, increase of accuracy, clarifying minorities. One of the main difficulties of big data analysis is the heavy computation. There would be many approaches, including distributed computation. In this talk, I would like to talk about algorithmic approaches to big data analysis in practically short time. The algorithms are for pattern mining, similarity analysis, compression, etc, and I also show the efficiency of the algorithms.

MapReduce Algorithmics

Sergei Vassilivitskii

Protocols for Distributed Learning

Suresh Venkatasubramanian

We consider the problem of learning classifiers for labeled data that has been distributed across several nodes. Our goal is to find a single classifier, with small approximation error, across all datasets while minimizing the communication between nodes. This setting models real-world communication bottlenecks in the processing of massive distributed datasets. We present several very general sampling-based solutions as well as some two-way protocols which have a provable exponential speed-up over any one-way protocol. We focus on core problems for noiseless data distributed across two or more nodes. The techniques we introduce are reminiscent of active learning, but rather than actively probing labels, nodes actively communicate with each other, each node simultaneously learning the important data from another node.

Continuous Distributed Counting for Non-Monotonic Streams

Milan Vojnovic

I will discuss the problem of tracking a sum of values in distributed environments where input values arrive from k distributed streams. It is required that an estimate of the sum is maintained by a coordinator node that is within a prescribed relative error tolerance, at any time with high probability. It is desired that this is achieved with small total communication with the coordinator node, in order to minimize the use of network bandwidth that may be a scarce resource. This is a basic distributed computation problem that is of interest for various applications including data aggregation and iterative solving of various computational problems. I will show that a sub-linear total communication with respect to the number of updates can be achieved for input values that are selected by an adversary but arrive in a random order, and show matching lower bounds. This result stands in between the previously known results that for monotonic partial sums (all value updates either positive or negative), the expected total communication is logarithmic in the number of updates, and

that for general worst-case inputs, the total required communication is linear in the number of updates. I will also discuss applications of the sum tracking module for tracking the second frequency moment and for solving a Bayesian linear regression.

Joint work with Zhenming Liu and Bozidar Radunovic.

Computing Statistical Summaries over Massive Distributed Data

Ke Yi

Facing today's big data challenge, it is often impossible and unnecessary to examine the entire data set for many analytical tasks. What is more useful is various concise statistical summaries extracted from the underlying data that capture certain data characteristics. In this talk, I will present simple and efficient algorithms for computing some classical statistical summaries over distributed data, in particular the frequency estimation problem and quantiles. These algorithms can be easily implemented in distributed systems like MapReduce, Dremel, and sensor networks.

Tight bounds for Distributed Continuous Monitoring

Qin Zhang

In this talk we will discuss recent developments in distributed continuous monitoring. We will introduce several new techniques to get tight bounds for central problems in this model, including frequency moments, heavy hitters and empirical entropy. In particular we will talk about how to get (almost) tight lower bounds for F0 (number of distinct elements) and F2 (size of self-join), based on multiparty number-in-hand communication complexity. These results resolve several major theoretical open problems in the area of distributed continuous monitoring.

Participants

Afrati, Foto, National Technical University of Athenes
Arge, Lars, Madalgo, Aarhus University
Cheng, Siu-Wing, HKUST
Cormode, Graham, AT&T Labs
El Abbadi, Amr, UCSB
Firmani, Donatella, Universita degli studi di Roma "La Sapienza"
Garofalakis, Minos, Technical University of Crete
Goel, Ashish, Stanford University
Harb, Boulos, Google
Karloff, Howard, AT&T Labs
Keren, Daniel, Haifa
Kishimoto, Akihiro, Tokyo Institute of Technology
Lattanzi, Silvio, Google
Laura, Luigi, Universita degli studi di Roma "La Sapienza"
McGregor, Andrew, Univ. of Massachusetts, Amherst
Mirrokni, Vahab, Google
Moseley, Benjamin, University of Illinois
Muthukrishnan, S., Rutgers, the State University of New Jersey
Phillips, Jeff M., University of Utah
Schuster, Assaf, Technion, Israel Institute of Technology
Sharfman, Izchak, Technion, Israel Institute of Technology
Stein, Cliff, Columbia University
Tarui, Jun, Univ. of Electro-Communications
Tirthapura, Srikanta, Iowa State Univ.
Ullman, Jeffrey, Stanford
Uno, Takeaki, NII
Vassilivitskii, Sergei, Yahoo!
Venkatasubramanian, Suresh, University of Utah
Vojnovic, Milan, Microsoft Research Cambridge
Yi, Ke, HKUST
Yu, Cong, Google
Zhang, Qin, Madalgo, Aarhus University