Space efficient indexes for the big data era
Sidirourgos, L.

Citation for published version (APA):
Sidirourgos, E. (2014). Space efficient indexes for the big data era

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (http://dare.uva.nl)
Chapter 1

Introduction

1.1 Scientific Discovery

Scientific discovery is the process of answering questions that have troubled mankind for centuries. In the early times, scientific discovery was based only on empirical observations. It was not until few centuries ago that we developed the necessary theories to explain our observations. The next breakthrough in scientific discovery was made possible the last few decades with the introduction of computers. Scientist were able to design complex mathematical models to simulate events, and with this newly acquired computational power, they were able through vast computations to examine “what if” scenarios and predict outcomes. However, the journey of scientific discovery has not yet reached an end.

Nowadays, scientific discovery has shifted from being an exercise of theory and computation, to become the exploration of an ocean of observational data. This transformation was identified by Jim Gray as the 4th paradigm of scientific discovery [35]. State-of-the-art observatories populated with astronomical data, digital sensors, and modern scientific instruments, produce every day petabytes of information. This scientific data is stored on massive datacenters for later analysis. But even from the data management viewpoint, the capture, curation, and analysis of data is not a computation-intensive process any more, but a data-intensive one. The explosion in the amount of scientific data presents a new “stress test” for database systems design. Meanwhile, the scientists are confronted with new questions: how can relevant and compact information be found from such a flood of data?

The predominant answer given by the data management community is the raw
power of big datacenter installations, complemented by new technologies focusing on scalable distribution of data and operations [2, 24]. When it comes to curating and analyzing vast amounts of data one can hardly argue that another approach is feasible. However, scientific discovery implies also a more delicate and refined task, that of forming the scientific question, a hypothesis to be later tested for correctness. Such formation of a scientific questions or hypothesis, as well as the initial proof-of-validity, constitutes an \textit{ad-hoc exploratory scientific query workload}. During an iterative and interactive query session, the scientists pose queries in order to examine the nature of the stored data, discover interesting aspects of it, formulate their hypothesis, but also test the syntactical correctness of their queries. Scientists need \textit{interactive} and \textit{low-cost} means to make an initial exploration over the daily produced data. Facing this challenge calls for a database architecture exhibiting features different from contemporary ones.

1.2 E-commerce

Besides data-intensive scientific discovery, the \textit{e-commerce surge} of the last decade has driven the need for managing huge amounts of data. The modern enterprise IT world is challenged with complex tasks aiming to get faster results and create more value out of their existing data loads and the readily available public data.

The e-commerce surge is primarily driven by the exponential evolution of the Web. Social networks, photography, videos, and blogging, as well as telecom data, web logs, even medical records, fuel the industry of decision-making applications. The collection and analysis of these data exceeds the capabilities of previous system deployments in enterprises. Data oriented frameworks are needed to provide real time services. Up until now, companies have used traditional analytic tools and data warehouses to analyze their own structured data, but more is to be discovered from the semi-structured web data. Enterprises seek to increase their profits by analyzing the impact of their services and the behavior of their users.

This is exactly where managing big amounts of data comes into play. Data analytics are needed in almost all of the big decision making capabilities for the enterprise community. New technologies are needed to overcome various technical obstacles, such as scalability, complexity of data, and speed (velocity) of the generation of data. For these reasons, we witness even more data centers and computer clusters put into use by enterprises. Big data holds great potential in the coming years to comes. Enterprises are not only looking for storing data but also want to get the best result out of it.
1.3 Big Data Analytics

The Big data term was coined following the e-commerce surge and the data-intensive scientific discovery. Big data refers to the collection of many data sets, but also, the term is used to describe the challenges posed to existing data management systems. The capture, curation, and analysis, but also the storage, transfer, and visualization of big data renders existing systems incapable of managing such loads. The challenges put forward are best described by the so called “3Vs”, volume, velocity, and variety [46, 68].

**Volume** refers to the vast amount of data that is produced and needs to be managed. Special care is needed so one will not “get lost” in such flood of data. The techniques proposed to deal with the first V are tier storage [71], extract statistical valid samples [73, 74], and identify cold spots [48] to name a few.

**Velocity** describes the increased frequency in which queries are fired to the system, as well as the demand for real time interaction, e.g., web applications. Solutions in the literature include constant reorganization of data [38] and multilayered kernel processing [41].

**Variety** of the big data includes incompatible data formats, incomplete data structures and inconsistent semantics. XML-based data formats, or semantic frameworks such as the RDF, are used to map schemas, express semantic relationships and inference new statements.

Recently, a fourth “V” was added, namely *veracity*. In traditional business intelligence and analytics applications, data are well structured. Significant amount of time and money is invested to guarantee the correctness of the data that is stored. However, in the big data era, this is no longer possible, thus veracity of the data, i.e., the *uncertainty of data*, is now a necessary evil that has to be dealt with. The data is now imprecise, inaccurate and often erroneous.

1.3.1 Big Data System Landscape

The database community, motivated by these challenges, introduced systems with innovative architecture. For example, MapReduce [24] was proposed as a new paradigm for data processing on large clusters, based on two phase execution: namely map and reduce. In addition, many systems based on Hadoop, an open source implementation of MapReduce, have been proposed to bridge the gap between this distributed programming model and SQL-based data management [2]. System designers have build
database warehouses to work on top of distributed environments, such as Hive, Pig, Impala, and Shark/Spark. The aforementioned systems harvest the raw data processing power by scaling out, i.e., when more processing power is needed, more machines are added. But, with the big data era, high-performance main memory systems have also flourished. These systems are designed to scale up. They take advantage of new and more powerful hardware by exploiting every last bit and cpu cycles available. Main memory transactional systems such as H-Store [39] and Hekaton [26], as well as read optimized systems such as MonetDB [58], Vectorwise, SAP Hana, and hybrid systems that support both transactional and analytical workloads, such as HyPer [40] belong to this category of highly optimized database systems.

1.4 Space Efficient Indexes

As mentioned before, the predominant answer of the database systems community to the big data challenge is the abundant use of computational and storage power. This is achieved with the deployment of large computer clusters and cloud computing. Even so, in most big data applications the access and the transfer of the data from slower storage units to faster memories still remains the bottleneck.

Main memory has become cheap enough to obtain large amounts, nevertheless all data can not possibly fit in main memory, thus secondary (slower) storage is still in use, such as SSDs and HDDs. During the evaluation of a query, it is necessary to access the data stored in such slower media to ensure completeness of the answer. Also, the data has to be transferred from slower storage to main memory, and from main memory to L2 cache, L1 cache, and so on. Evidently, the access and transfer of the data through different layers of the memory hierarchy is the bottleneck for data intensive applications.

In this thesis we present a collection of storage efficient indexes to overcome the data transfer bottleneck. An index is typical used to restrict access to only the relevant to the query parts of the data. By space efficient we emphasize that the index has to be significantly smaller than the original data and typically has to reside in at least one level higher in the memory hierarchy than the indexed data. During query evaluation, before accessing and transferring data from a slower memory to a faster, we consult the space efficient index which resides in much faster memory. The index will reveal which parts, if any, of the ”slower” data are relevant to the query and should be transferred. The main novelty of the indexes presented in this thesis compared to prior work is that they take advantage of the memory hierarchy to achieve optimal performance and to facilitate simple but efficient compression rates. The indexes are memory conscious.
1.4. SPACE EFFICIENT INDEXES

in the sense that they are designed to exactly fit in memory to optimize access. Moreover, they are secondary indexes, i.e., no data replication or reclustering is needed. These features make the indexes particularly useful for big data, both for systems that scale up, by exploiting the hardware, and for distributed systems that scaled out, since they can be used to reduce the amount of data transferred from one node to another.

The indexes we present here, except being space efficient, they also address different challenges of the 3Vs big data problems. Imprints is a cache-conscious index structure suitable for read intensive applications. It allows to identify which cachelines of data should be read to the cpu caches, thus reducing the time cost of bringing all data from main memory to the cpu. Imprints address the volume dimension of the big data challenge.

In the same line of thinking, the second index we introduce, named Split Bloom filters, is an index that restricts access to slower secondary storage. Besides addressing the volume problem, split Bloom filters also tackles the velocity of the big data. They allow for faster evaluation of frequent queries and more efficient updates of Bloom filters. The value indexes presented next in this thesis work for semi-structured data of any type, thus exploring solutions for the variety challenge posed by big data. Finally, the n-gram based index we present last allow for indexing of very large data sets on disks.

Space efficient indexes allow significant reduction of hardware investments. They make sure that all available power is harvest before moving to larger installations.

1.4.1 Contributions

The contributions of this thesis can be summarized as follows:

1. We introduce column imprints, a collection of many small bit vectors, each indexing the data points of a single cacheline. We introduce a compression schema for imprints, which is cpu friendly and exploits the empirical observation that data often exhibits local clustering or partial ordering as a side-effect of the construction process. We conducted an extensive experimental evaluation to assess the applicability and the performance impact of the column imprints.

2. We show how skew in access patterns can be exploited to improve Bloom filter efficiency (less space and/or lower false positive rate). We do this by splitting the filter into many smaller ones, where each filter covers only a small subset of records. We define a mathematical model and show how to optimally size the Bloom filters. We also describe how to adjust the sizing of the filters because of changes in access patterns or deterioration caused by updates. An extensive
experimental evaluation confirms that our algorithms construct better Bloom filters optimized for skewed access patterns, that they achieve a lower false positive rate or lower memory consumption depending on the settings, and that they can adapt gracefully to data and workload changes.

3. We describe a collection of indices for XML text, element, and attribute node values that (i) consume little storage, (ii) have low maintenance overhead, (iii) permit fast equi-lookup on string values, and (iv) support range-lookup on any XML typed value (e.g., double, dateTime). We evaluate the stability of the hash function, the storage overhead, and the indices creation and maintenance time in the context of MonetDB/XQuery.

4. We study methods to conserve the scalable creation time and efficient exact substring query properties of gram indices, while reducing storage space. We first propose a partial gram index based on a reduction from the problem of omitting indexed q-grams to the set cover problem. While this method is successful in reducing the size of the index, it generates false positives at query time, reducing efficiency. We then increase the accuracy of partial grams by splitting posting lists of frequent grams in a frequency-tuned set of signatures that take the bytes surrounding the grams into account. The resulting qs-gram scheme is tested on big data collections (up to 426GB) and is shown to achieve an almost 1:1 data to index size, and query performance even faster than normal gram methods.

1.4.2 Published Papers

The material in this thesis has been published in major international refereed database conferences.


2. **Memory Efficient Bloom Filters for Skewed Access Patterns.** Lefteris Sidirourgos and Per-Åke Larson. *Submitted for publication at the moment of printing this thesis.*

3. **Generic and Updatable XML Value Indices Covering Equality and Range Lookups.** Lefteris Sidirourgos and Peter Boncz. *Fourth International workshop on Database Technologies for Handling XML Information on the Web (DataX '09), collocated with EDBT/ICDT 2009 joint conference, St. Petersburg, Russia, 2009.*
1.4. SPACE EFFICIENT INDEXES

4. **Space-Economical Partial Gram Indices for Exact Substring Matching.** Nan Tang, Lefteris Sidirourgos and Peter Boncz. *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM), Hong Kong, China, 2009.*

1.4.3 Thesis Outline

Each chapter of this thesis is an independent work, each of which introduces a new index schema. We do not include in this thesis a single related work chapter, since each index has its own. Chapter 2 presents column imprints. Next, Chapter 3 presents split Bloom filters. Chapter 4 presents the XML value indexes. Finally, Chapter 5 presents the qs-grams. Chapter 6 concludes this thesis.