Sidra: a Flexible Distributed Indexing and Ranking Architecture for Web Search

Miguel Costa and Mário. J. Silva

Departamento de Informática
Faculdade de Ciências da Universidade de Lisboa
1700 Lisboa, Portugal
{mcosta@xldb.di.fc.ul.pt, mjs@di.fc.ul.pt}

Abstract. This paper introduces Sidra, a new Web indexing and ranking system for tumba!, a search engine for the Portuguese Web. Sidra has a flexible distributed architecture that enables very fast response times and scalability, while providing multiple access methods that can be chosen dynamically as a function of the queries and client context. The paper discusses the main design choices and outlines the architecture of the Sidra implementation.

1 Introduction

Existing Web search engines need to serve millions of users while providing the best possible results for any query in a fraction of a second. As mobile devices become a common interface to query these systems, there is a need to provide adaptive responses. For instance, location dependent queries will demand different results to the same queries, depending on preferences expressed by the users or inferred from their computing environment.

To provide high-scalability and sub-second response times in such dynamic environment, relevance of results matching a query needs to be evaluated using multiple criteria. It is necessary to index various features, corresponding to different importance metrics for the indexed pages.

State of the art Web search engines such as Google, have an indexing structure that provides access methods where a unique ranking dimension is used to index the Web pages [1]. In the dynamic environment that we envision, user preferences change over time and the same query means different things to different users.

To answer this need, we have designed Sidra, a new indexing and ranking system for large Web data sets that meets the above requirements. Sidra maintains several distributed indexing data structures. Indexes can be organized by different ranking criteria and can be selected to find matches based on the context-data that can be associated to queries. This approach increases significantly the storage requirements for indexes, but preserves the sub-second response times currently demanded by search engines' users.

Tumba! (see http://www.tumba.pt) is a search engine for the Portuguese Web which is being offered as a public service since 2002 [2]. The implementation

of the version of Sidra presented in this paper is now in operation while under a process to evaluate scalability and eliminate some performance bottlenecks. It is presently responding to more than 10 thousand queries a day over a collection of 4 million documents, with response times under 0.5 seconds.

2 Architecture

Sidra is composed of different indexes, organized by different criteria and partitioned over many Query Servers. Each Query Server is responsible for searching a list of *sid* (Sidra document identifiers) and associated information matching the queries received. Figure 1 represents a possible configuration of Sidra's architecture. There are three different types of indexes: term-documents index, location-documents index and a topic-documents index. These enable document searches on up to three dimensions. Dimensions can vary with user needs. For instance, one dimension can be time, enabling users to search documents crawled in different periods. Indexes are partitioned by multiple Query Servers horizontally, allowing searches on different dimensions in parallel (partition parallelism) in a very short period of time. We have chosen a horizontal partition (also known as global partition) instead of a vertical partition (also known as local partition), because previous studies indicate that for bandwidths above 100 Mbits/sec it achieves better performance [3–5].

Brokers receive queries from Clients, and use their catalog with the distribution of the indexes and the location of the Query Servers, to dispatch sub-queries requesting from them matching documents and all the associated information. Brokers then merge the results received in parallel from the Query Servers as they are being produced. This provides a pipeline parallelism that enhances query performance. After that, documents are ranked according to several criteria. Clients are user programs embedded with the Client library of Sidra that provide a user interface to Sidra. Clients format Broker responses, transforming the data for presentation on user devices.

Different configurations of this architecture can be built to respond to specific needs of Web search engines. Components can be incremented, replicated or partially replicated as the data and query workloads increase. For instance, if a Broker becomes the bottleneck, we can add another Broker to the system and Clients will start sending queries in a round-robin way among the Brokers. As some indexes are much more frequently accessed than others, the workload of the Query Servers that serve these indexes can be much higher, becoming the bottleneck of the system. Sidra supports replication of these Query Servers. Brokers can then also perform load balancing by distributing requests by the less loaded servers. As replicating a full Query Server can represent a significant amount of storage resources, Sidra's architecture also enables the creation of partial replicas of Query Servers for the more frequently accessed entries.

By designing Sidra to be scalable and capable of load balancing, we also achieve fault-tolerance: Clients detect Brokers that stop responding or exhibit slow response times, and redirect their request to other available Brokers. On

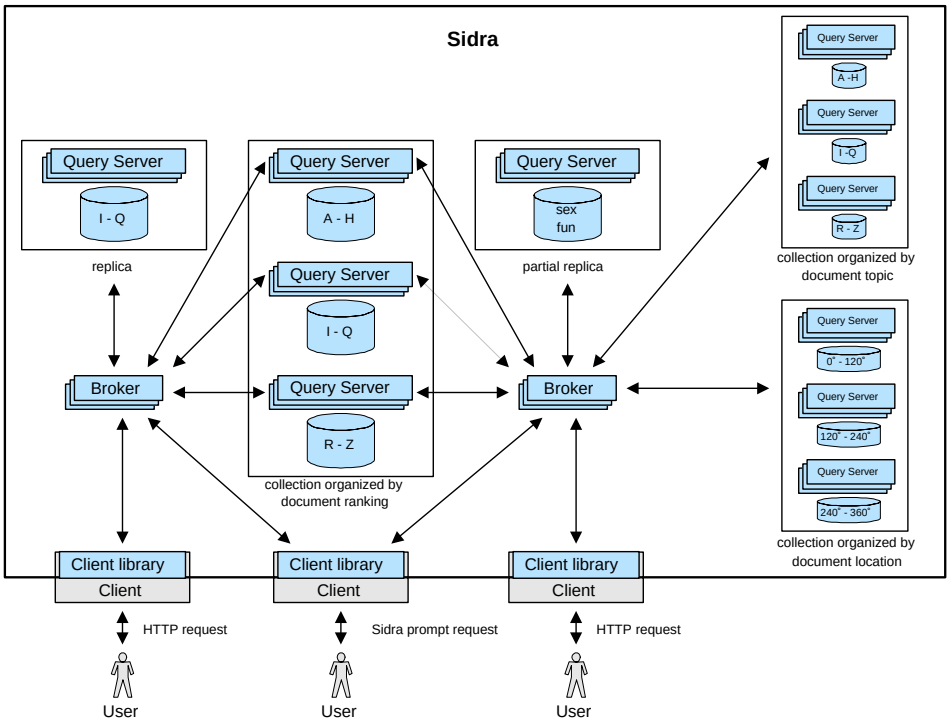

Fig. 1. one possible configuration of Sidra's architecture.

the other hand, Brokers also sense Query Servers and may redirect requests to another Query Servers with an index replica.

3 Related Work

Global Web search engines, such as Google, have similar indexing architectures despite using vertical index partitioning. However, we are not aware of efforts to provide query planning and index selection on this system [6].

Inquery is a centralized retrieval engine, whose retrieval model is based on a Bayesian inference network [7]. However, it is not designed to support Web data. Cahoon et al. built a distributed information retrieval system using a multithreaded version of the Inquery information system [8]. Clients search over multiple text collection simultaneously, with the Broker selecting the collections more relevant to each query, to restrict searches to a small percentage of data. They attempt to solve two problems: how to select the most relevant subsets of the collections in order to reduce the search space (the collection selection problem), and how to merge the results from different collections, frequently

with incomparable ranking scores (this is known as the result merging problem or the collection fusion problem). Zhihong Lu extended this architecture with partial collection replication and selection [9].

4 Conclusion

We presented the architectural design of a new indexing and ranking system for the tumba! search engine. Sidra has been designed to scale-up and provide very fast response times, comparable to state-of-the-art Web search engines. Its flexible structure provides also load balancing, fault tolerance and can support the selection of indexing structures organized by different ranking criteria.

We are presently evaluating the running implementation of Sidra as the engine of tumba!. To evaluate scalability and performance we are establishing a testing environment where a realistic set of queries extracted from tumba!'s logs, will be applied with different workloads to multiple Sidra configurations.

References

1. Brin, S., Page, L.: The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems **30** (1998) 107–117
2. Silva, M.J.: The case for a portuguese web search engine. In: IADIS WWW/Internet 2003 Conference. (2003)
3. Tomasic, A., Garcia-Molina, H.: Performance of inverted indices in distributed text document retrieval systems. In: Proceedings of the Second International Conference on Parallel and Distributed Information Systems. (1993) 8–17
4. Jeong, B.S., Omiecinski, E.: Inverted file partitioning schemes in multiple disk systems. IEEE Transactions on Parallel and Distributed Systems **6** (1995) 142–153
5. Ribeiro-Neto, B.A., Barbosa, R.A.: Query performance for tightly coupled distributed digital libraries. In: Proceedings of the third ACM Conference on Digital libraries. (1998) 182–190
6. Barroso, L.A., Dean, J., Holzle, U.: Web search for a planet: The google cluster architecture. IEEE Micro Magazine (2003) 22–28
7. Callan, J.P., Croft, W.B., Harding, S.M.: The inquery retrieval system. In: Proceedings of the Third International Conference on Database and Expert Systems Applications. (1992) 78–83
8. Cahoon, B., McKinley, K.S., Lu, Z.: Evaluating the performance of distributed architectures for information retrieval using a variety of workloads. ACM Transactions on Information Systems **18** (2000) 1–43
9. Lu, Z.: Scalable Distributed Architectures for Information Retrieval. PhD thesis, University of Massachusetts Amherst (1999)