Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 112 (2017) 811–821

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Knowledge Based and Intelligent Information and Engineering Systems, KES2017, 6-8 September 2017, Marseille, France

An efficient synchronous indexing technique for full-text retrieval in distributed databases

Fadoua Hassen[a*] , Grissa Touzi Amel[b]

[a,b]*LIPAH, FST, University of Tunis El Manar, Tunis, Tunisia*

Abstract

Despite the fact that distributed databases (DDB) have become a requirement of users and real-world applications, these DDBs are still too far from the performance of centralized DBs that have shown efficiency and robustness with their native search engines. Indeed, DDB couldn't benefit from textual search engines because of the timing issue in the distributed context and keeping data integrity among the distributed architecture. As a contribution in this research field, we propose in this paper, a new technology for indexing DDB based on the Lucene indexing engine. We prove that this technology allows, in addition to textual search, 1) the transparency of distributed architecture's scaling [8] and external search indexes, 2) the optimization of the exchange protocol data between nodes, 3) the synchronous processing of insertion, deletion or failure of a node of the distributed topology.

Peer-review under responsibility of KES International

Keywords: synchronization; distributed database; Full-text; Lucene

1. Introduction

In the last two decades, the market leading search engines such as Google and Yahoo, invested high efforts to establish the best possible algorithms and the most efficient hardware to answer the client's queries. One of the most common policies used by the leading market search engines is the datacenters decentralization. Indeed, in parallel to the evolution of DDB[10], an external layer has been added to improve the quality of search and accelerate access.

* Corresponding author.
E-mail address: hassen.fadoua@gmail.comru

1877-0509
Peer-review under responsibility of KES International
10.1016/j.procs.2017.08.071

The textual search engines offer additional capabilities to the SQL-based searches such as advanced truncation, phonetic search, thesaurus, context and proximity [1] which are mandatory features of a successful search engine. Lucene framework [17] is one of the most successful open-source API to provide such capabilities. Lucene search indexes are stored outside the database and can be shared between multiple nodes for read and update. The resulting indexes are an optimized copy of database content to enable a faster search: Indexes may contain the database exact values or may just hold a dictionary for different index keys. A specific issue related to search engines is the textual indexes update problems. In this specific area of distributed applications, the update is not restricted to database content, but it is always a matter of the textual indexes update to allow the search engines highest capabilities [9]. However, the new deal started with distributed architecture is to preserve data integrity among the different nodes of the topology. In order to afford the same result to the end-users, the duplicate copies of physical data must be synchronized[14] in the near real-time scale between all nodes. The real time copy updates[7] are so far impossible in large scale systems due to the write lock concurrency problems and the network latency. In this paper, we develop a new approach to satisfy the enrichment of the search capabilities of DDB by textual search techniques: By adding a layer of data indexes with the indexing engine Lucene. The proposed scenario offers the designer a friendly and productive interface that allows it to index the distributed databases. In addition, it offers to the final user an advanced search interface as the root word search (find all the words derived from a common root), search by synonyms (find the different word that refers to the meaning of the word the input of the research), proximity search (finding a word that sounds like the input value before), search by the same name sounds (phonetic search), etc.When the indexes distribution pattern is established, the component "Synchronizing distributed indexes" must act as a light agent, in the background, that propagates automatically and independently, for each node, the different update operations on Lucene indexes.

Besides this introduction, this paper includes seven sections. Section 2 discusses the related work on full text retrieval for DDB. Section 3 introduces the proposed solution architecture. Section 4 presents the suggested synchronization protocol. Section 5 details the use of textual search Layer in our approach. Section 6 reports the experimental tests of the implemented example and discusses the different aspects of the result. Section 7 summarizes the result of this work and opens the future perspectives for it.

2. Related works

The efficiency of distributed search indexes was tested in many different approaches and the common result of these works is always positive. The map-reduce implementation studied in [13] to generate and search among distributed semantic search indexes shows an acceptable level while generating both vertical and horizontal indexes fragments. The query time and the indexing time show that horizontal segmentation using the parallel indexes is more efficient. However, in this map-reduce implementation, only RDF were taken into consideration. There were no persistant database behind the indexes. The on-going update time is not evaluated and the live segmentation as soon as data arrives is ignored in this study. Moreover, the indexing map-reduce strategy generates too much intermediate map data causing the overall slowness of indexing [11]. A single pass map-reduce strategy was suggested in [15] but we believe that the use case is always built on a particular assumption: The input data is available and is not evolving. In our proposed solution we also take into account that data is evolving considerably among the time and act based on this assumption. In the real world, the modern analytic approach tend to adapt the output result of the real-time representation of data store. Thus, we reserved an important part of this work to the exchange protocol and its possible optimization.

The network impact of distributed index systems was studied in depth in [12] and shows the heavy impact of a misconfigured vertical replication between different nodes. The work suggests many optimization technic such as a data compression and a result paging. In this paper, the data compression is avoided for the first implementation and may be studied in further works, although we estimate that the zipping-dezipping process will add a considerable overhead to both search and generation process. The paging aspect is widely adopted and is even adapted with network available bandwidth in this work.