Text Search Using Database Systems Revisited
— Some Experiments —

Helmut Kaufmann and Hans-Jörg Schek

Swiss Federal Institute of Technology (ETH Zürich)
Department of Computer Science—Databases
{kaufmann, schek}@inf.ethz.ch

Abstract. With the increasing availability of information in electronic form, the integration of textual data into database systems is becoming more and more important. Motivated by recent technology development, we describe how a preprocessor for simple text retrieval can be realized on top of a relational database system. This approach shows a surprisingly good performance compared to a commercially available information retrieval system and compared to another relational preprocessor product for text search.

1 Introduction and Motivation

In the late seventies and early eighties a considerable amount of research was devoted to the comparison of database systems (DBS) and information retrieval systems (IRS) and to a synthesis of these. Principle differences between both are described in [Rij81]. Specifically, discussions of data model issues are given in, for example, [SP82, DKA+86]; architectural issues in [Bil82, Sch84, LDE+85, LKD+88, Fuh89]. Prototype systems built on top of relational systems are described in [Mac79, SSL+83]. Our own early experience was gained by a prototype system [Pol80] built on top of IBM’s SQL/DS by using the reference string indexing method [Sch77]. Although these early attempts showed some promise, the consensus was more or less that relational systems were not really suitable for text retrieval. Many groups started to build new next generation database systems, e.g. [SR86, SW86, SPSW90], rather than putting a document management and search preprocessor on top of existing database systems.

Now, more than ten years later, we believe that this discussion must be resumed for the following reasons:

- **Technology Evolution:** Relational database technology has evolved together with the dramatic changes in hardware and communication technology. Sophisticated client-server architectures and transaction technology provide parallel search and update of many users in a scalable way. They more and more use multi-processor hardware and provide not only inter- but also intra-transaction parallelism.

- **Higher Level of Abstraction:** Increasingly, relational database systems are being used as storage managers upon which sophisticated object managers and application-oriented tools are built. Some consider the relational
interface as a replacement of the file system that, among many advantages, enables parallelisation at the I/O level.

- **More Text:** Many data items in current database systems contain textual attributes. Over eighty percent of the information generated in a business environment is full text, and most of the documents are enriched by structured information such as the type of the documents, their creation dates, authors, keywords, and addressees.

Our general question therefore is whether, and to what extent, we can utilise existing technology and support primitive text search in today's relational databases by building a simple preprocessor on top of these systems. Our initial expectations were the following:

- An information retrieval system should outperform any relational preprocessor solution in the single-user mode.
- A preprocessor solution on top of a relational database system should outperform an information retrieval system in case of multiple parallel users.

In this paper, we will report on our findings but we are far from giving final answers. The contribution of this paper is the presentation of some carefully measured experiments and observations that we made. It shows that our simple preprocessor solution on top of Oracle outperforms a specialised IRS (BASIS-Plus [Inf90]), especially in the case where many parallel searchers have to be supported. We also show that it is necessary to introduce a simple query optimisation by comparing our own preprocessor solution with the commercially available product SQL*TextRetrieval [SQL92]. The test data are taken from a real application. Although we restricted ourselves to a single application, we think that the observations gained there point out that the existing widely-held opinion that it is inappropriate to support an IRS on a DBS should not be taken for granted.

The structure of this paper is as follows: In Sec. 2, we present inverted lists as an access structure for text and show how they can be integrated into a relational database system using a simple preprocessor. In Sec. 3, we introduce BASISPlus and and two preprocessor solutions implemented on top of Oracle. After briefly introducing the real world application PHONO+ in Sec. 4, we conclude with a presentation and comparison of our measurement results.

## 2 Simple Access Structures for Relational Database Systems

In order to support full text search in a relational database system efficiently, textual access structures must be introduced. In this section, we present inverted lists as a simple access structure for boolean text retrieval and show how they can be included in a relational database system using a simple preprocessor. A document — in the context of this paper but without loss of generality — is a tuple of a relation with one full text attribute. By extracting the single