Adaptive and Automated Index Selection in RDBMS

Martin R. Frank, Edward R. Omiecinski and Shamkant B. Navathe

College of Computing, Georgia Institute of Technology, Atlanta GA 30332-0280, USA
{martin,edwaxdo,sham}@cc.gatech.edu

We present a novel approach for a tool that assists the database administrator in designing an index configuration for a relational database system. A new methodology for collecting usage statistics at run time is developed which lets the optimizer estimate query execution costs for alternative index configurations. Defining the workload specification required by existing index design tools may be very complex for a large integrated database system. Our tool automatically derives the workload statistics. These statistics are then used to efficiently compute an index configuration. Execution of a prototype of the tool against a sample database demonstrates that the proposed index configuration is reasonably close to the optimum for test query sets.

1 Introduction

Relational database management systems (RDBMS) are by far the most popular database systems today. Despite their known shortcomings in non-traditional applications like engineering and image processing, no clear alternatives have evolved despite substantial research. Thus RDBMS are likely to dominate the commercial arena for years to come, especially for business applications.

Relational databases use indices to provide fast access to data. The presence of an index reduces the search time for indexed data items but also complicates update operations since the tuples as well as the indices must be updated. Hence there is a tradeoff involved in selecting indices and indexing every column is rarely a good idea\(^1\). This tradeoff decision will be referred to as the index selection problem (ISP).

In the context of this paper the ISP refers to tailoring the index configuration to the database usage profile, not to selecting an index set to process a single query. The single-relation ISP denotes the ISP reduced to selecting an index configuration for a single relation, which is much easier than the general ISP since join queries pose the hardest problems. The single-index ISP refers to choosing an index configuration consisting of single indices, combined (concatenated) indices are excluded from consideration.

The decision as to which attributes to index is influenced by numerous factors, such as database usage, characteristics of the database system and the underlying operating system. Due to this complexity, it is difficult for the unassisted database administrator (DBA) to choose a good index configuration for a large integrated database. Tools were

\(^1\) In the context of B-tree indices, not Grid files.
proposed which relieve some of this burden from the DBA, however even the most sophisticated ones like DBDSGN [FST88] still require the DBA to manually specify the workload. The designer has to specify the workload as a small set of weighted representative queries. It is unclear how the DBA can condense the transactions on a huge DBMS (consider 100 tables and 1000 transactions per hour) to just – say – 20 representing the original workload. Hence mechanisms have to be found which derive the workload information for those tools from the database system itself.

Throughout this paper the workload specification (or usage input) denotes the part of the input for index selection tools which contains information about the workload for which the index configuration is optimized. The workload specification may consist of statistics (e.g. how often a certain column was referenced or updated during the last time period) or representative queries.

Research in the area of automated index selection has treated statistics gathering and statistics evaluation separately. Consequently: (1) the existing statistics gathering mechanisms – originally intended for debugging – consume too much overhead to continuously collect usage data, and (2) the existing index selection tools are rarely used in practice because they require hard-to-derive statistics as input. It is our belief that a successful tool must integrate both aspects.

Our tool requires no usage input specified by the designer, it derives all its input automatically during regular database usage. The output is an optimum\(^2\) index configuration for the queries during the recording period in the sense that it minimizes the average query execution time.

2 Previous Research

We discuss previous work on the index selection problem (ISP) with respect to the workload specification. The list is by no means complete. Research in the area not discussed here has been done by Palermo [Pal70], King [Kin74] and many others.

Stonebraker [Sto74] constructs a probabilistic model for database activity and solves the single-index single-relation ISP for certain special cases in polynomial time. The formalization of the index selection problem provides insight into its difficulty, but the results are valid for special cases only and there is no methodology presented for finding an index configuration for the general case. The usage input parameters are (1) the probability that a query is a non-retrieval query (Insert, Delete, Update) as opposed to a retrieval query (Select) and (2) for all columns i the probability that column \(c_i\) appears restrictively in a query. These statistics have to be specified as input to the tool. It is obvious that these statistics are not trivial to derive. There are also some general problems with analytical approaches to the ISP. First, substantial simplifications have to be made to derive an analytical solution. Second, the model becomes obsolete if there are changes to the query processing strategy or to other modeled aspects of the DBMS.

Schkolnick [Sch75] presents a more general probabilistic model and an algorithm that solves the single-index single-relation ISP significantly faster than the naive approach. As in [Sto74] a cost function is derived that gives the expected average query execution cost depending on the index configuration. An algorithm is presented that finds the optimal

\(^2\) Truly optimal only for a restricted version of the index selection problem.