Optimal plan search in a rule-based query optimizer

Ming-Chien Shan

Hewlett-Packard Laboratories
1501 Page Mill Road, Palo Alto, CA 94303

Abstract

This paper describes an optimal plan search strategy adopted in a rule-based query optimizer. Instead of attempting to search for the optimal plan directly, an initial plan is first generated based upon a set of heuristic rules. Depending upon the application, the initial plan may be used either as the final plan or as a base in a subsequent search. A new concept—clustering degree of an index—is introduced to better model the I/O costs of index scans. This new statistical information facilitates the formulation of the rules. An exhaustive search based upon the A* algorithm is then invoked to guarantee the optimal property of the plan. A lower bound value is derived and used as the estimation of "remaining distance" required in the A* algorithm. Noteworthy features of our approach include the capability for dynamic control of exhaustive search for an optimal plan, and on-line performance monitoring/tuning. The preliminary results lead us to believe that the rule-based approach is a promising one to face the new challenges of the optimizer, as created by the requirements of supporting diversified applications.

1 Introduction

In most database management systems, various ad hoc mechanisms are used for query processing, so that different cases of queries are not handled in a uniform way. This makes enhancements and extensions of the system very difficult. The issues of maintainability and extensibility had been ignored until recently, when they started gaining overdue interest [2, 15, 16, 27]. The approaches vary over a wide spectrum, from just recoding the optimization procedures in conventional rule format and employing traditional AI production systems for query processing, to the building of DBMS generators capable of supporting any data model.

The Iris database management system [14] developed at Hewlett-Packard Laboratories is a research prototype of a next-generation database management system. It is intended to meet the needs of new and emerging database applications, including software engineering, computer-aided design and manufacturing, office information and knowledge-based systems.

One of our objectives is to provide "extensibility" of the system in a general sense. In other words, we aim at the design of a system which can be dynamically tuned, with reasonable effort, to
support different system configurations and/or functional requirements for various new applications. To facilitate the support of this capability, a rule-based approach to query optimization is adopted. The query optimization process is driven by a set of rules which are applied to the query tree during several recursive scans.

The design of the query optimizer plays a key role in the success of a database management system. A query optimizer evaluates a number of strategies for processing a given query and selects one that minimizes some cost measure. One problem in query optimizer design is how to search a large space of potential query execution plans efficiently. In this paper, issues related to optimal plan search in a rule-based optimizer are investigated and a new solution is proposed.

Most conventional systems, such as system R [28], make an exhaustive search using a breadth-first algorithm. Another search algorithm widely adopted in DBMSs is the hill-climbing algorithm [4, 16, 18, 33]. However, systems adopting the hill-climbing search algorithm risks choosing a suboptimal plan, due to such problems as the local maximum, plateau, and ridge problems [32]. It is not easy to enhance this kind of algorithm to correct this deficiency. Therefore, we feel that the adoption of an exhaustive search type of algorithm is still necessary for certain applications.

During the past decade, many researchers have written about query optimization, but none uses the A* algorithm [32] for optimal plan search. In order to apply the A* algorithm, one must be able, at any point within the search, to estimate the remaining distance to reach the goal. In the context of query optimization, this means that, given a current partial plan, a reasonable estimate of the cost of the additional work required to obtain a complete plan must be provided. One reason that the A* algorithm has not been widely adopted in query optimization is the difficulty of estimating this value.

We need a heuristic function that estimates the merits of each node generated in the query tree. It will enable unpromising paths to be pruned at an early stage when applying the A* algorithm. This is the essence of the A* search and the place where good heuristics should be exploited. Since an accurate value cannot be easily derived, a lower bound of the "remaining costs" is computed and used in the A* algorithm.

In our approach, a set of basic rules is first applied to produce an initial candidate plan. In general, the initial plan generated is a reasonable one and good for certain categories of applications such as ad hoc queries or queries compiled in a PC environment. However, if an optimal plan is required, as for queries in repeatedly executed applications, an exhaustive search of the entire plan space will be performed. During the search, the cost of the initial plan supplies an upper bound on the cost of the optimal plan. Additional information is also used for speeding up the pruning of unpromising plans.