Parallel Skyline Computation Exploiting the Lattice Structure

Markus Endres, Department of Computer Science, University of Augsburg, Augsburg, Germany

Werner Kießling, Department of Computer Science, University of Augsburg, Augsburg, Germany

ABSTRACT

The problem of Skyline computation has attracted considerable research attention in the last decade. A Skyline query selects those tuples from a dataset that are optimal with respect to a set of designated preference attributes. Since multicore processors are going mainstream, it has become imperative to develop parallel algorithms, which fully exploit the advantages of such modern hardware architectures. In this paper, the authors present high-performance parallel Skyline algorithms based on the lattice structure generated by a Skyline query. For this, they propose different evaluation strategies and compare several data structures for the parallel evaluation of Skyline queries. The authors present novel optimization techniques for lattice based Skyline algorithms based on pruning and removing one unrestricted attribute domain. They demonstrate through comprehensive experiments on synthetic and real datasets that their new algorithms outperform state-of-the-art multicore Skyline techniques for low-cardinality domains. The authors’ algorithms have linear runtime complexity and fully play on modern hardware architectures.

Keywords: Multicore, Parallelization, Pareto, Preference, Skyline

INTRODUCTION

The Skyline operator (Börzsönyi, Kossmann, and Stocker, 2001) has emerged as an important and very popular summarization technique for multi-dimensional datasets. A Skyline query selects those objects from a dataset \( D \) that are not dominated by any others. An object \( p \) having \( d \) attributes (dimensions) dominates an object \( q \), if \( p \) is better than \( q \) in at least one dimension and not worse than \( q \) in all other dimensions, for a defined comparison function. This dominance criterion defines a partial order and therefore transitivity holds. The Skyline is the set of points which are not dominated by any other point of \( D \). Without loss of generality, we consider the Skyline with the \( \text{min} \) function for all attributes.

The most cited example on Skyline queries is the search for a hotel that is cheap and close to the beach (Börzsönyi et al., 2001). Unfortunately, these two goals are complementary, as the
hotels near the beach tend to be more expensive. In Figure 1 each hotel is represented as a point in the two-dimensional space of price and distance to the beach. Interesting are all hotels that are not worse than any other hotel in both dimensions.

The hotels \( p_6, p_7, p_8 \) are dominated by hotel \( p_3 \). The hotel \( p_9 \) is dominated by \( p_4 \), while the hotels \( p_1, p_2, p_3, p_4, p_5 \) are not dominated by any other hotels and build the Skyline. From the Skyline one can now make the final decision, thereby weighing the personal preferences for price and distance to the beach.

Most of the previous work on Skyline computation has focused on the development of efficient sequential algorithms (Chomicki, Ciaccia, and Meneghetti, 2013). The most prominent algorithms are characterized by a tuple-to-tuple comparison-based approach. Based on this, several algorithms have been published in the last decade, e.g., NN (Nearest Neighbor) (Kossmann, Ramsak, and Rost, 2002), BBS (Branch and Bound Skyline) (Papadias, Tao, Fu, and Seeger, 2003), SFS (Sort-Filter Skyline) (Chomicki, Godfrey, Gryz, and Liang, 2003), or LESS (Linear Elimination-Sort for Skyline) (Godfrey, Shipley, and Gryz, 2007), just to name a few.

However, the datasets to be processed in real-world applications are of considerable size, i.e., there is the need for improved query performance, and parallel computing is a natural choice to achieve this performance improvement, since multicore processors are going mainstream (Mattson and Wrinn, 2008). This is due to the fact that Moore’s law of doubling the density of transistors on a CPU every two years – and hence also doubling algorithm’s performance – may come to an end in the next decade due to thermal problems. Thus, the chip manufacturers tend to integrate multiple cores into a single processor instead of increasing the clock frequency. In upcoming years, we will see processors with more than 100 cores, but not with much higher clock rates. However, since most applications are build on using sequential algorithms, software developers must rethink their algorithms to take full advantage of modern multicore CPUs (Mattson and Wrinn, 2008). The potential of parallel computing is best described by Amdahl’s law (Gustafson, 1988): the speedup of any algorithm using multiple processors is strictly limited by the time needed to run its sequential fraction. Thus, only high parallel algorithms can benefit from modern multicore processors.

\[ \text{Figure 1. Skyline example} \]
Fuzzy Database Modeling
www.igi-global.com/chapter/fuzzy-database-modeling/11155?camid=4v1a

Looking for Information in Fuzzy Relational Databases Accessible Via Web
Carmen Martínez-Cruz, Ignacio José Blanco and Maria Amparo Vila (2009).
Database Technologies: Concepts, Methodologies, Tools, and Applications (pp. 2448-2471).
www.igi-global.com/chapter/looking-information-fuzzy-relational-databases/8046?camid=4v1a