IJDCST @Jan-2015, Issue- V-3, I-2, SW-08
ISSN-2320-7884 (Online)
ISSN-2321-0257 (Print)

Effective Skyline Maintenance on Big Data

K Shireesha[1], A.Soumya[2]

[1] Assistant Professor, CJITS,Jangaon,Telangana,India.

[2] Sr. Assistant Professor, CJITS,Jangaon,Telangana,India.

ABSTRACT: Skyline is an important operation in many applications to return a set of interesting points from a potentially huge data space. Given a table, the operation finds all tuple's that are not dominated by any other tuple's. It is found that the existing algorithms cannot process skyline on big data efficiently. This paper presents a novel skyline algorithm SSPL on big data. SSPL utilizes sorted positional index lists which require low space overhead to reduce I/O cost significantly. We present a new indexing method named ZINC (for Z-order indexing with Nested Code) that supports efficient skyline computation for data with both totally and partially ordered attribute domains. By combining the strengths of the Z-order indexing method with a novel nested encoding scheme to represent partial orders, ZINC is able to encode partial orders of varying complexity in a concise manner while maintaining a good clustering of the PO domain values. Our experimental results have demonstrated that ZINC outperforms the state-of-the-art TSS technique for various settings.

INTRODUCTION

In order to specify the process of big data representation a Collaborative Filtering Skyline (CFS), a general framework that combines the advantages of CF with those of the skyline operator. CFS generates a personalized skyline for each user based on scores of other users with similar behavior. The personalized skyline includes [1] objects that are good on certain aspects, and eliminates the ones that are not interesting on any attribute combination. Although the integration of skylines and CF has several attractive properties, it also involves rather expensive computations. We face this challenge through a comprehensive set of algorithms and optimizations that reduce the cost of generating personalized skylines. In addition to exact skyline processing, we develop an approximate method that provides error guarantees.

We refer to such system as *Data-Intensive Super Computer* systems [8][9], or "DISC." We believe that DISC systems could yield breakthroughs in a number of scientific disciplines and other problems of societal importance. Much of our inspiration for DISC comes from the server infrastructures that have been developed to support search over the worldwide web. Google and its competitors have created DISC systems providing very high levels of search quality, response time, and availability.

LITERATURE REVIEW

In this article we observe different research people's decisions and their requirements in useful format for developing Skyline Computation over large amount of data representation. In this scenario we will take number of author's suggestion [1] with specified format for developing my application independently. Botolini decisions on big data representation explains following things efficiently Collaborative filtering (CF) systems exploit previous ratings and similarity in user behavior to recommend the top-k objects/ records.

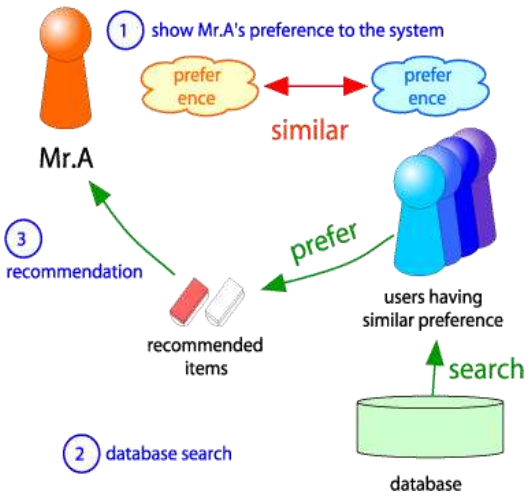

Figure 1: Data representation on collaborative filtering with missing values.

In order to specify the process of big data representation a Collaborative Filtering Skyline (CFS), a general framework that combines the advantages of CF with those of the skyline operator. CFS generates a personalized skyline for each user based on scores of other users with similar behaviour, we develop an approximate method that provides error guarantees.

.

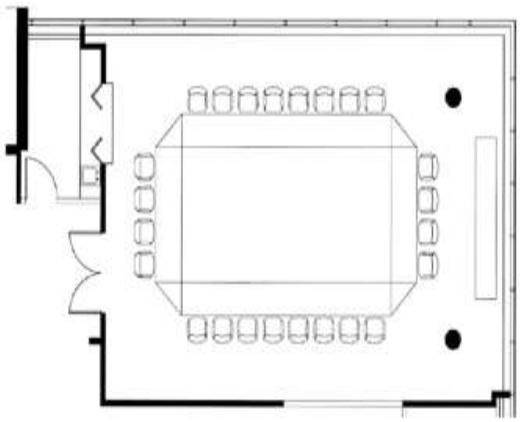

Figure 2: Space search of the users in skyline process generation.

***k*-Dominant Skyline and its Variants**

The primary cause for the skyline set size blow up is the definition [3] of *dominance*, which is rather stringent in that *p* must be better or equal to *q* in all dimensions before *q* can be eliminated from the skyline. Using our movie rating as an example, this means that movie *p* is only considered better than movie *q* if and only if *p* is rated higher or equal to *q* by all the users.

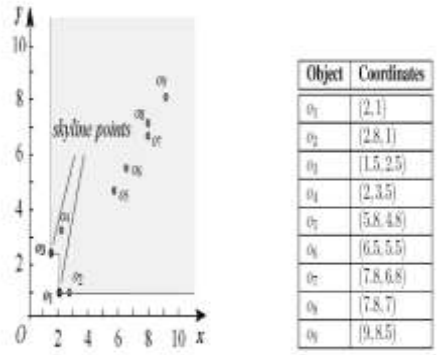

Figure 3: Object Representation in skyline queries.

The skyline is also known under other names such as the *pare to set*, the set of *admission points*, and the set of *maximal vectors:*

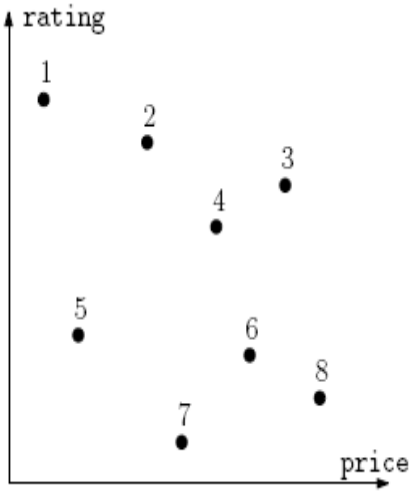

Figure 4: The Skyline is {1, 5, and 7}.

Ever since its debut in the database literature a decade ago skyline computation has generated considerable interests in the database area for a brief survey. This is, at least in part, due to the relevance of skylines to *multi criteria optimization*. Figure 4 illustrates an example with 8 hotels, of which the

skyline is *{*1, 5, 7*}*. Every hotel *not* in the skyline is worse than at least one hotel in the skyline on both dimensions, i.e., more expensive [3] and rated worse at the same time. In general, for any scoring function that is monotone on all dimensions, the skyline always contains the *best* (top-1) point that minimizes the function. Skyline queries are capable of retrieving interesting points from a large data set according to multiple criteria[3], from a large *n*-dimensional data set, skyline queries are well suitable or applications like decision making and multiple criteria optimization.

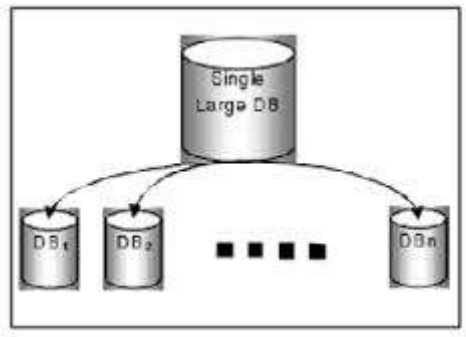

Figure 5: Introducing skyline with distributed systems.

For instance, a tourist can issue a skyline query on a hotel relation to get those ones with good recommendations and cheap prices. Another real-life example is online comparative shopping, in which a search engine needs to get good bargains from many different shopping sites according to multiple criteria like *price*, *quality*, *guarantee*, etc. Clearly, such multiple criteria are best captured by a skyline query[3]. In such cases as mentioned above, however, directly applying existing techniques to process skyline queries would incur large overhead. We make the following contributions on the problem of distributed constrained skyline queries in a network environment without assuming any overlay structures. First, we propose a specific partition algorithm that divides all relevant sites into groups, such that a given query can be executed in parallel among all those site groups. Second, we elaborate the query execution within any single group of sites, which includes deciding a query forwarding plan between sites and designating a group head responsible for query results merger. Third, we give a parallel distributed skyline algorithm based on the first two contributions, together with a cost model to estimate query costs. Fourth, we detail how to use multiple filtering points in distributed query processing such that the network data transmission is more efficient. Two simple yet efficient heuristics are developed for selecting multiple filtering points on a processing site. Finally, we conduct extensive experiments on both synthetic and real datasets, and the results demonstrate the efficiency and robustness of our proposals.

EXISTING SYSTEM

Skyline is an important operation in many applications to return a set of interesting points from a potentially huge data space. For providing skyline computation process on each data set traditionally used index-based algorithms utilize the pre-constructed data-structures to avoid scanning the entire data set. It pre-constructs data-structures with low space overhead. By the data-structures, the algorithm only involves a small part of table to return the skyline results. Index-based algorithms have serious limitations and the used indexes can only be built on a small and selective set of attribute combinations. So the better system is required for doing efficiency of skyline computation on mathematical analysis.

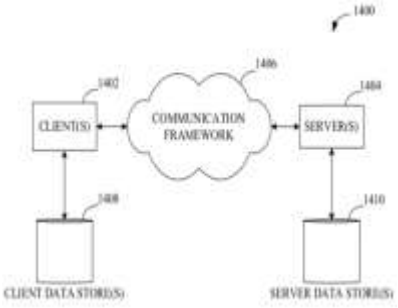

Figure 6: Data monitoring in various data application event generation

Their algorithm first retrieves values in every dimension from remote data sites using sorted access in round-robin on all dimensions. This continues until all dimension values of an object, called the terminating object, have been retrieved. Wu et al. proposed a parallel execution of constrained skyline queries in a CAN based distributed environment

Problem Statement

In this paper we consider the related problem of finding the expected number of maximal elements in a given set. We give a solution to that problem under a very general probability distribution and then apply that answer to related problems. A maximal vector is one that is not less than any other vector in all components. More precisely, we say that a vector P dominates the vector Q if P is greater than Q in every component; then a vector is maximal if it is not dominated by any other vector in the set.

A Fast Expected-Time Maxima Algorithm

This problem has received much attention recently. Each vector is initially represented as a pair of integers which define the top and bottom endpoints of a segment in the array. Division into further subsets can be accomplished by taking the arithmetic mean of the endpoints as defining two new segments, etc.; note that *the division preserves randomness* and can be accomplished in constant time.

The SSPL Algorithm

This section first introduces the data-structures required by SSPL, then describes the overview of the SSPL algorithm , next shows how to perform pruning, followed that presents the implementation and analysis of SSPL, and finally introduces how to extend SSPL to cover other cases.

Pruning

In this section we describe early pruning and late pruning SSPL splits coordinate space into 2m regions by the computed scan depth.

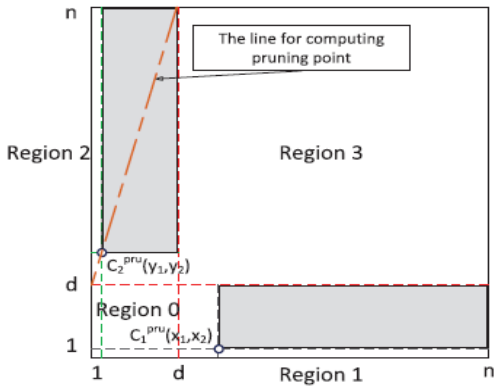

Figure 7: An instance of early pruning.

It reflects the value of SSPL on processing skyline on big data. In uniform distribution, the experimental results verify the theoretical analysis in the paper

PRAPOSED SYSTEMS

In this paper we propose a novel skyline algorithm that is Skyline Sorted Positional Index List algorithm; it defines following things for big data assessment process. The proposed frame work contains following things as follows, This can utilize some small preconstructed data-structures to reduce I/O cost significantly. To determine scan depth of the sorted positional index lists. It devises pruning operation on the candidate positional indexes, and the mathematical analysis for pruning. SSPL consists of two phases. The main scenario of this document is to

improve the performance of the all the requirement specifications.

ZINC

In this section, we present our proposed indexing method named ZINC (for Z-order Indexing with Nested Code) that supports efficient skyline computation for data with both TO as well as PO attribute domains.

Nested Encoding Scheme

We introduce a novel encoding scheme, called *nested encoding* (or NE, for short), for encoding values in PO domains. The encoding scheme is designed to be amenable to Z- order indexing such that when the encoded values are indexed with a ZB-tree, the two desirable properties of monotonicity and clustering of ZB-tree are preserved.

Horizontal, Vertical, and Irregular Regions

Note that a vertical region corresponds to a collection of total orders while a horizontal region corresponds to a weak order2. We have defined a regular region to be a maximal subgraph in order to have as large a regular structure as possible to be encoded concisely.

Partial Order Reduction Algorithm

We present an algorithm, termed PO-Reduce, that takes a partial order *G*0 as input and computes a *reduction sequence*, denoted by *G*0 *! G*1 _ _ _ *! Gn*□1 *! Gn*, *n* _ 0, that transforms *G*0 into a total order *Gn*, where each *Gi*+1 is derived from *Gi* by replacing some regions in *Gi* by virtual nodes. The reduction sequence will be used by our nested encoding scheme to encode each node in *G*0. Given an input partial order*Gi*, algorithm PO-Reduce operates as follows: (1) Let *S* = *fS*1*; _ _ _ Skg* be the collection of regular regions in *Gi*; (2) If *S* is empty, then let *S* = *fS*1*g*, where *S*1 is an irregular region in *Gi* that has the smallest size (in terms of the number of nodes) among all the irregular regions in*Gi*. (3) Create a new partial order *Gi*+1 from *Gi* as follows. First, initialize *Gi*+1 to be *Gi*. For each region *Sj* in *S*, replace *Sj* in *Gi*+1 with a virtual node *v0j* such that *parent*(*v0j*) = *parent*(*v*) with *v 2 min*(*Sj*) and *child*(*v0j*) = *child*(*v*) with *v 2 max*(*Sj*). (4) If *Gi*+1 is a total order, then the algorithm terminates; otherwise, invoke the PO-Reduce algorithm with *Gi*+1 as input.

Skyline query processing has attracted a lot of research. For generating process of the sequential execution in data set constructions. Efficient skyline computation in recent data generation is the main complex task in sorted positional Index list algorithm. So we propose to extend SSPL with indexing method named ZINC (for Z-order Indexing with Nested Code) that supports efficient skyline computation for data with both totally and partially ordered attribute domains.

CONCLUSION:

We proposed to use Skyline query processing has attracted a lot of research. For generating process of the sequential execution in data set constructions. Efficient skyline computation in recent data generation is the main complex task in sorted positional Index list algorithm. So we propose to extend SSPL with indexing method named ZINC (for Z-order Indexing with Nested Code) that supports efficient skyline computation for data with both totally and partially ordered attribute domains. Our experimental results show that using the nested encoding scheme, ZINC significantly outperforms TSS as well as two ZB-tree variants based on different encoding schemes.

REFFERENCES:

[1]. I. Bartolini, Z. Zhang, and D. Papadias, "Collaborative Filtering with Personalized Skylines," IEEE Trans. Knowledge Data Eng., vol. 23, no. 2, pp. 190-203, Feb. 2011.

[2] Randal E. Bryant, "Data-Intensive Supercomputing: The case for DISC", Early experiences on the journey towards self-* storage. *IEEE Data Eng. Bulletin*, 29(3):55–62, 2006.

[3] Chee-Yong Chan1, H.V. Jagadish2, Kian-Lee Tan1, Anthony K.H. Tung1, Zhenjie Zhang, "Finding k-Dominant Skylines in High Dimensional Space", SIGMOD 2006,June 27–29, 2006, Chicago, Illinois, USA. Copyright 2006 ACM 1-59593-256-9/06/0006 ...$5.00.

[4] L. Chen and X. Lian, "Efficient Processing of Metric Skyline Queries," IEEE Trans. Knowledge Data Eng., vol. 21, no. 3, pp. 351- 365, Mar. 2009.

[5] Seagate, "Barracuda xt: No Compromise. Speed and Capacity for High-Performance Desktop Systems,"http://www.seagate.com/ docs/pdf/datasheet/disc/ds_barracuda_xt.pdf, 2012.