Answering Skyline Queries over Incomplete Data with Crowdsourcing

Manchala Kavitha

Department of Computer Science, Sri Venkateswara University, Tirupati

***Abstract*—** *This query has a large application base in many real-life scenarios such as decision making, profiled based recommendation, location-based services. Extensive experiments using both real and synthetic data sets confirm the superiority of Bayes Crowd to the state-of-the-art method, in terms of execution time, monetary cost, and latency minimization. a novel query framework, termed as Bayes Crowd, which takes into account the data correlation using Bayesian network. We leverage the typical c-table model on incomplete data to represent objects. Considering budget and latency constraints, we present a suite of effective task selection strategies. Moreover, we introduce a marginal utility function to measure the benefit of crowdsourcing one task. In particular, the probability computation of each object being an answer object is at least as hard as #SAT problem.*

***Keywords: Query Processing, Skyline Query, Incomplete Data, Crowdsourcing*.**

I. INTRODUCTION

The skyline query finds the objects that are not dominated by any other object, where an object o dominates another object o ′ iff o is not worse than o ′ in all attributes, and is better than o ′ in at least one attribute. This query has a large application base in many real-life scenarios such as decision making, profiled based recommendation, location-based services [1], [2], [3], [4]. Take a dataset in movie recommendation system as an example. Each movie is represented by a vector containing ratings from audiences. For instance, for the three movies m1 = (3, 2, 1), m2 = (4, 2, 3), and m3 = (2, 3, 2), each of them has ratings from three audiences where the higher the rating, the better. It is said, m1 is dominated by m2, as m2 has equal/higher ratings to/than m1 on three attributes. Thus, m2 and m3 are the skyline points. In real-life applications, it is impossible for all audiences to watch/score a certain movie. Hence, some movie ratings are usually missing. As depicted in Table 1, there are five movies (i.e., objects) {o1, o2, o3, o4, o5} with ratings from five audiences (w.r.t. attributes) {a1, a2, a3, a4, a5}. The movie/object o2's value on the attribute a2 is missing, and thus, it is denoted by the variable V ar(o2, a2). Obviously, the real answer objects of this skyline query cannot be obtained due to the data incompleteness. Incomplete data are ubiquitous in a wide spectrum of real-life applications, owing to a variety of reasons such as A Sample Dataset ID Name Film Ratings from Audiences a1 a2 a3 a4 a5 o1 Schindler's List (1993) 5 2 3 4 1 o2 Se7en (1995) 6 Var (o2, a2) 2 2 2 o3 The Godfather (1972) 1 1 Var(o3, a3) 5 3 o4 The Lion King (1994) 4 3 1 2 1 o5 Star Wars (1977) 5 Var(o5, a2) Var(o5, a3) Var(o5, a4) 1 instable sensor networks, data integration, data loss, privacy preservation, etc. For example, users tend to skip certain fields when they fill out on-line forms; participants choose to ignore some sensitive questions on surveys; publicly viewable satellite map services contain missing map data in many mobile applications; and in privacy-preserving applications, the data is incomplete deliberately in order to preserve the sensitivity of some attribute values. As a result, incomplete data queries have been extensively explored in the past decade, including skyline queries top-k queries similarity queries and so forth. However, most of existing models and approaches for incomplete data queries only rely on the machine power. As well known, the machine has limitations in some cases where the human is powerful. Collective intelligence has become a hot topic, with the development of Web 3.0 and the emerging of artificial intelligence (AI) techniques. As a consequence, many crowdsourcing platforms emerge, such as Amazon mechanical turk (AMT)1 , FigureEight2 , and Upwork3 , each of which acts as an intermediate between requesters and workers. On crowdsourcing platforms, a requester posts a series of tasks, and workers answer those tasks and get paid. Compared with conventional trade markets, the crowdsourcing platform offers a freer employment contract where workers can come and go as their wills freely. Towards this, in this work, we resort to crowdsourcing techniques to handle skyline queries over incomplete data.

1.1 What is Data Engineering?

The key to understanding what data engineering lies in the "engineering" part. Engineers design and build things. "Data" engineers design and build pipelines that transform and transport data into a format wherein, by the time it reaches the Data Scientists or other end users, it is in a highly usable state. These pipelines must take data from many disparate sources and collect them into a single warehouse that represents the data uniformly as a single source of truth.

1.2 How Did Data Engineering Come About?

Many would say that data engineering as a profession has been around for well over a decade, maybe a couple, ever since databases, Microsoft SQL Servers and ETL came to be. Some would say ever since IBM popularized database management systems in the 1970s. With that, here's a very brief history recap.

In the 1980s the term "information engineering" was coined to largely describe database design and to include software engineering in data analysis. Somewhere after the rise of the internet in the 1990s and 2000s, 'big data" came to be. Yet DBAs, SQL Developers and IT professionals working in the field were not labelled "Data Engineers" at that time.

1.3 Why the Critical Need for Data Engineering Now?

By now you've heard/read about Gartner's determination back in 2017 that 85% of big data projects fail. This was largely due to a lack of reliable data infrastructures. Data could not be trusted enough to base key business decisions on it. Fast forward to 2019 and things had not improved. The CTO of IBM said that 87% of data science projects never make it into production. Gartner reiterated its prediction that now just 80% of projects would fail. A New Vantage Report produced similar stats.

Over the last decade, most companies have completed a digital transformation. This has produced unimaginable volumes of new types of data and much more complicated data at a higher frequency. While it was previously apparent that Data Scientists were needed to make sense of it all, it was less apparent that someone needs to organize and ensure this data's quality, security, and availability for the Data Scientists to do their jobs.

So, in the early days of big data analytics, Data Scientists were very often expected to build the necessary infrastructure and data pipelines to do their work. This was not necessarily in their skill sets or expectations for the job. The result was that data modelling would not be done correctly. There would be redundant work and inconsistency in the use of data among Data Scientists. These kinds of issues prevented companies from being able to extract optimal value from their data projects, so they failed. It also led to a high rate of Data Scientist turnover that still exists today.

Today with the onslaught of completed corporate digital transformations, the Internet of Things and the race to become AI-driven, it is crystal clear that companies need Data Engineers in abundance to provide the foundation for successful data science initiatives.

This is why will we continue to see the role of Data Engineers grow in importance and breadth. Companies need teams of people whose sole focus is to process data in a way that allows them to extract value from it.

1.4 What Is the Relationship and Difference Between Data Scientists and Data Engineers?

Much has been written about the relationships between these two roles, so we'll be brief. In the past, companies thought that they could get away with having Data Scientists do the role of Data Engineers. This is what has caused much of the "unicorn effect" and shortage in Data Scientist recruitment.

Some Data Scientists also sold themselves as being able to do a Data Engineer's job. Many fell short – see the image to the right courtesy of O'Reilly.com.

Today, the volume and speed of data have driven Data Scientist and Data Engineer to become two separate and distinct roles albeit but with some overlap.

It's now widely recognized that companies need both Data Scientists and Data Engineers in an advanced analytics team. It's pretty difficult to do any meaningful data science without Data Engineers to support this function. There's frequent collaboration between Data Engineers and Data Scientists however the priority skills and knowledge of tools are different.

Data Engineer Ability:

Data Scientists are focused on advanced analytics of data that is generated and stored in a company's databases. Data Engineers design, manage and optimize the flow of data with those databases throughout the organization. So, Data Scientists will be highly skilled in math and statistics, R, algorithms and machine learning techniques. Data Engineers will be more versed in SQL, MySQL, and NoSQL, architecture and cloud technologies and frameworks such as agile and scrum.

Both will likely know Python, visualization techniques and have other coding languages in common.

Foundation software engineering – Agile, devOps, architecture design, service-oriented architecture.

Distributed systems – This would include software engineer skills and software architect skills.

Open Frameworks – Apache Spark, Hadoop, perhaps Hive, MapReduce, Kafka and others…

SQL – This is a database staple and remains that way.

Programming – Python has become the favoured language for working with data. Java on the other hand, while still widely sought has fallen out of favour with most data scientists and engineers. Scala is another language that Apache Spark and Kafka are based on.

Pandas – a Python library for cleaning and manipulating data.

Visualization/dashboards

Cloud platforms – AWS is probably the most prevalent cloud skill set for Data Engineers to know. Google Cloud Data Engineering and Microsoft Azure are right behind.

Analytics – While mainly the realm of data scientists, statistical analysis skills or understanding of some of the different mathematical principles or probabilistic principles are necessary for being able to properly manipulate the data so that it is in a shape that is accessible for the people who are doing the end analysis on it.

Data modeling – Data modeling knowledge is quite important now in the sense that a Data Engineer needs to know how they are going to structure tables, partitions, where to normalize and deformalize data in the warehouse, etc. and how to think about retrieving certain attributes.

II. LITERATURE SURVEY:

Title: Progressive skyline computation in database systems

Author: Bernhard Seeger

Abstract: The skyline of a *d*-dimensional dataset contains the points that are not dominated by any other point on all dimensions. Skyline computation has recently received considerable attention in the database community, especially for progressive methods that can quickly return the initial results without reading the entire database. All the existing algorithms, however, have some serious shortcomings which limit their applicability in practice. In this article we develop branch-and-bound skyline (BBS), an algorithm based on nearest-neighbor search, which is I/O optimal, that is, it performs a single access only to those nodes that may contain skyline points. BBS is simple to implement and supports all types of progressive processing (e.g., user preferences, arbitrary dimensionality, etc). Furthermore, we propose several interesting variations of skyline computation, and show how BBS can be applied for their efficient processing.

Title: Finding Optimal Skyline Product Combinations under Price Promotion

Author: Zhibang Yang Keqin Li

Abstract: Nowadays, with the development of e-commerce, a growing number of customers choose to go shopping online. To find attractive products from online shopping marketplaces, the skyline query is a useful tool which offers more interesting and preferable choices for customers. The skyline query and its variants have been extensively investigated. However, to the best of our knowledge, they have not taken into account the requirements of customers in certain practical application scenarios. Recently, online shopping marketplaces usually hold some price promotion campaigns to attract customers and increase their purchase intention. Considering the requirements of customers in this practical application scenario, we are concerned about product selection under-price promotion. We formulate a constrained optimal product combination (COPC) problem. It aims to find out the skyline product combinations which both meet a customer's willingness to pay and bring the maximum discount rate. The COPC problem is significant to offer powerful decision support for customers under price promotion, which is certified by a customer study. To process the COPC problem effectively, we first propose a two list exact (TLE) algorithm. The COPC problem is proven to be NP-hard, and the TLE algorithm is not scalable because it needs to process an exponential number of product combinations. Additionally, we design a lower bound approximate (LBA) algorithm that has a guarantee about the accuracy of the results and an incremental greedy (IG) algorithm that has good performance. The experiment results demonstrate the efficiency and effectiveness of our proposed algorithms.

Title: Progressive Approaches for Pareto Optimal Groups Computation

Author: Zhibang Yang

Abstract: Group skyline query is a powerful tool for optimal group analysis. Most of the existing group skyline queries select optimal groups by comparing the dominance relationship between aggregate-based points; such feature creates difficulties for users to specify an appropriate aggregate function. Besides, many significant groups that have great attractions to users in practice may be overlooked. To address these issues, the group skyline (GSky) query is formulated on the basis of a general definition of group dominance operator. While the existing GSky query algorithms are effective, there is still room for improvement in terms of progressiveness and efficiency. In this paper, we propose some new lemmas which facilitate direct generation of the GSky query results. Consecutively, we design a layered unit-based (LU) algorithm that applies a layered optimum strategy. Additionally, for the GSky query over the data that are dynamically produced and cannot be indexed, we propose a novel index-independent algorithm, called sorted-based progressive (SP) algorithm. The experimental results demonstrate the effectiveness, efficiency, and progressiveness of the proposed algorithms. By comparing with the state-of-the-art algorithm for the GSky query, our LU algorithm is more scalable and two orders of magnitude faster.

Title: Skyline Query Processing for Incomplete Data

Author: Justin J. Levandoski

Abstract: Recently, there has been much interest in processing skyline queries for various applications that include decision making, personalized services, and search pruning. Skyline queries aim to prune a search space of large numbers of multi-dimensional data items to a small set of interesting items by eliminating items that are dominated by others. Existing skyline algorithms assume that all dimensions are available for all data items. This paper goes beyond this restrictive assumption as we address the more practical case of involving incomplete data items (i.e., data items missing values in some of their dimensions). In contrast to the case of complete data where the dominance relation is transitive, incomplete data suffer from non-transitive dominance relation which may lead to a cyclic dominance behaviour. We first propose two algorithms, namely, "Replacement" and "Bucket" that use traditional skyline algorithms for incomplete data. Then, we propose the "ISkyline" algorithm that is designed specifically for the case of incomplete data. The "ISkyline" algorithm employs two optimization techniques, namely, virtual points and shadow skylines to tolerate cyclic dominance relations. Experimental evidence shows that the "ISkyline" algorithm significantly outperforms variations of traditional skyline algorithms.

Title: Skyline Preference Query Based on Massive and Incomplete Dataset

Author: Lingfeng Sun; Baoyan Song

Abstract: Personalized recommendation and the processing of real-time data exemplify the processing of massive data which in the field of Internet-of-Things (IoT) received a great extent of attention in recent literature. The incompleteness of massive data in the IoT is widespread. Obtaining personalized information from the incomplete data set is still puzzled by searching efficient and accurate methods at present. Skyline query is a widely used data processing method, especially in the field of multi-objective decision analysis and data visualization. To eliminate the negative effects on massive data processing in IoT, a novel skyline preference query strategy based on massive and the incomplete data set is proposed in this paper. This strategy simply separates and divides massive and incomplete data set into two parts according to dimension importance and executes skyline query, respectively. The strategy mainly resolves the problem of extracting personalized information from massive and incomplete data set and improves the efficiency of skyline query on massive and incomplete data set. First, this paper presents a skyline preference query strategy based on strict clustering and implements it on dimensions that have higher importance. Second, a skyline preference query strategy based on loose clustering is implemented on dimensions that have lower importance. Finally, integrating local skyline query results, this paper calculates global skyline query results by using information entropy theory. The efficiency and effectiveness of Skyline Preference Query (SPQ) algorithm have been evaluated in terms of response time and result set size through the comparative experiments with ISkyline algorithm and sort-based incomplete data skyline algorithm. A large number of simulation results show that the efficiency of SPQ algorithm is higher than that of other common methods.

2.1 Existing System:

In real-life applications, it is impossible for all audiences to watch/score a certain movie. Hence, some movie ratings are usually missing. As a key step of measuring the utility, the probability computation is at least as hard as the model counting problem. By contrast, the closest related work to ours is the crowd skyline query. As analysed in Section 1, the work is based on unary questions to impute missing values of objects, resulting in the inaccurate result. While in, they partition attributes into the

observed attributes and the crowd ones, and assume that all values in crowd attributes are missing. Nevertheless, in real-life scenarios, it is easy to realize that, the values are usually missing over attributes randomly. Besides, both studies assume that data attributes are independent. It is worth noting that, our work studied in this paper allows attribute values missing randomly, and takes into account the data correlation. However, all the techniques above do not support the crowd skyline query with incomplete data. In addition, the work combines Bayesian network and crowdsourcing to impute missing values, which is different from our goal of optimizing the query quality with crowdsourcing.

2.2 Disadvantage:

- The performance of this encrypted format is low.
- It consuming more time and the cost.
- The security and the accuracy is less

III. PROPOSED SYSTEM

Generate the condition of each object being a query answer object for cable construction. In the crowdsourcing phase, we develop three task selection strategies in iteration policy under budget and latency constraints. The marginal utility function is introduced to quantify the benefit of crowdsourcing one task. As a key step of measuring the utility, the probability computation is at least as hard as the model counting problem.

We present a novel crowd skyline query framework BayesCrowd, which incorporates two main phases, i.e., the modeling phase and the crowdsourcing phase.

In the modeling phase, we generate the condition of each object being a query answer object for cable construction. In the crowdsourcing phase, we develop three task selection strategies in iteration policy under budget and latency constraints. The marginal utility function is introduced to quantify the benefit of crowdsourcing one task.

As a key step of measuring the utility, the probability computation is at least as hard as the model counting problem (i.e., #SAT problem). We propose an adaptive DPLL (abbrev. of Davis-PutnamLogemann-Loveland) (ADPLL for short) method to accelerate the computation.

Extensive experimental evaluation using both real and synthetic data sets demonstrates that, our proposed framework BayesCrowd is superior to the state-of-the-art method in terms of both efficiency and accuracy, under a variety of parameter settings.

3.1 Advantages

- High security and more effective.
- This system gives more accuracy compared to another systems

3.2 Algorithm

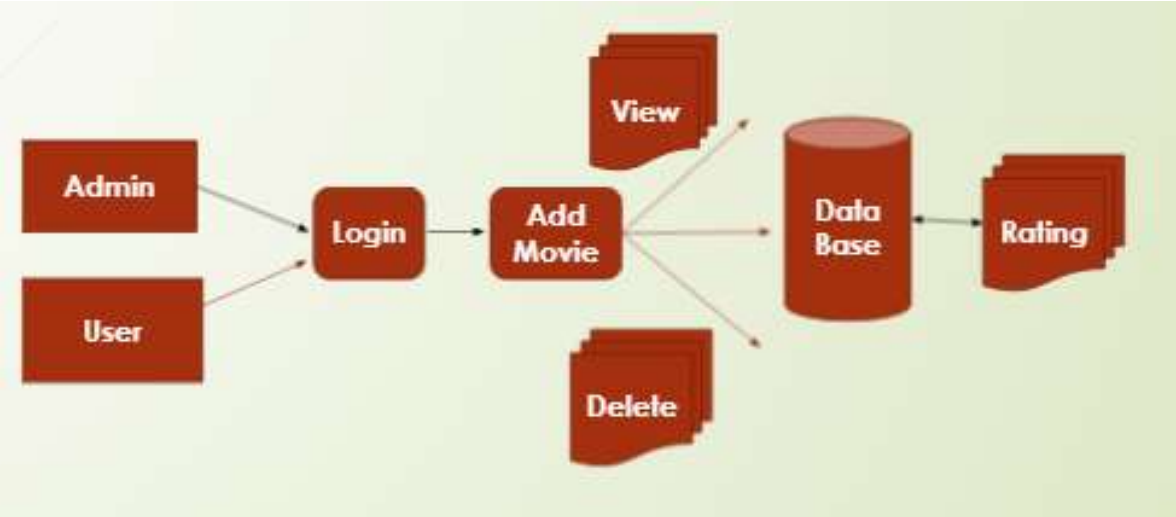
Ciphertext:

Ciphertext is also known as encrypted or encoded information because it contains a form of the original plaintext that is unreadable by a human or computer without the proper cipher to decrypt it. Decryption, the inverse of encryption, is the process of turning ciphertext into readable plaintext.

AES:

The algorithm described by AES is a symmetric-key algorithm, meaning the same key is used for both encrypting and decrypting the data. The Advanced Encryption Standard (AES) is a symmetric block cipher chosen by the U.S. government to protect classified information. AES is implemented in software and hardware throughout the world to encrypt sensitive data. It is essential for government computer security, cybersecurity and electronic data protection. Symmetric, also known as secret key, ciphers use the same key for encrypting and decrypting, so the sender and the receiver must both know -- and use -- the same secret key. The government classifies information in three categories: Confidential, Secret or Top Secret. All key lengths can be used to protect the Confidential and Secret level. Top Secret information requires either 192- or 256-bit key lengths.

IV. SYSTEM ARCHITECTURE



4.1 Module Description

A novel query framework, termed as Bayes Crowd, which takes into account the data correlation using Bayesian network Considering budget and latency constraints, we present a suite of effective task selection strategies. Moreover, we introduce a marginal utility function to measure the benefit of crowdsourcing one task.

V. FUTURE ENHANCEMENT

Nevertheless, none of the aforementioned work adopts crowdsourcing techniques to process incomplete data. In addition, it is noteworthy that, the dominance relationship analysis is based on the traditional dominance relationship definition on complete data, instead of incomplete data. Hence, the techniques for skyline computation over incomplete data cannot be applied to our studied problem.

VI. CONCLUSION

A novel crowd skyline query framework BayesCrowd. It takes into account the data correlation, and consists of two major phases, i.e., the modeling phase and the crowdsourcing maker phase. In the modeling phase, the query results are represented with the c-table model. We present an effective approach for cable construction. Since probability computation in terms of conditions in the c-table is at least as hard as #SAT problem, we develop an ADPLL algorithm to accelerate computation. For the crowdsourcing phase, we put forward a suite of effective task selection strategies, which consider budget and latency constraints. We also introduce a marginal utility function to measure the benefit of crowdsourcing a task. Extensive experiments on both real and synthetic datasets demonstrate the superiority of BayesCrowd. In the future, we intend to further explore the quality optimization problem on answering incomplete data queries.

REFERENCES

[1] S. Berzsenyi, D. Kossmann, and K. Stocker, “The skyline operator,” in ICDE, pp. 421–430, 2001.

[2] D. Papadis, Y. Tao, G. Fu, and B. Seeger, “Progressive skyline computation in database systems,” ACM Trans. Database Syst., vol. 30, no. 1, pp. 41–82, 2005.

[3] X. Zhou, K. Li, Z. Yang, and K. Li, “Finding optimal skyline product combinations under-price promotion,” IEEE Trans. Know. Data Eng., vol. 31, no. 1, pp. 138–151, 2018.

[4] X. Zhou, K. Li, Z. Yang, G. Xiao, and K. Li, “Progressive approaches for pareto optimal groups computation,” IEEE Trans. Know. Data Eng., vol. 31, no. 3, pp. 521–534, 2019.

[5] M. E. Khalifa, M. F. Mokbel, and J. J. Levandoski, “Skyline query processing for incomplete data,” in ICDE, pp. 556–565, 2008. [6] X. Miao, Y. Gao, B. Zheng, G. Chen, and H. Cui, “Top-k dominating queries on incomplete data,” IEEE Trans. Know. Data Eng., vol. 28, no. 1, pp. 252–266, 2016.

[6] Y. Wang, Z. Shi, J. Wang, L. Sun, and B. Song, “Skyline preference query based on massive and incomplete dataset,” IEEE Access, vol. 5, pp. 3183–3192, 2017.