Indonesian Journal of Electrical Engineering and Computer Science
Vol. 10, No. 2, May 2018, pp. 798~806
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v10.i2.pp798-806

A Model for Processing Skyline Queries in Crowd-sourced Databases

Marwa B. Swidan, Ali A. Alwan*, Sherzod Turaev, Yonis Gulzar
Department of Computer Science, Kulliyyah of Information and Communication Technology, International Islamic University Malaysia, Kuala Lumpur, 53100 Malaysia

Article Info

Article history:

Received Nov 15, 2017
Revised Jan 29, 2018
Accepted Feb 11, 2018

Keywords:

Crowd-sourcing
Estimating missing values
Incomplete data
Skyline query

ABSTRACT

Nowadays, in most of the modern database applications, lots of critical queries and tasks cannot be completely addressed by machine. Crowd-sourcing database has become a new paradigm for harness human cognitive abilities to process these computer hard tasks. In particular, those problems that are difficult for machines but easier for humans can be solved better than ever, such as entity resolution, fuzzy matching for predicates and joins, and image recognition. Additionally, crowd-sourcing database allows performing database operators on incomplete data as human workers can be involved to provide estimated values during run-time. Skyline queries which received formidable attention by database community in the last decade, and exploited in a variety of applications such as multi-criteria decision making and decision support systems. Various works have been accomplished address the issues of skyline query in crowd-sourcing database. This includes a database with full and partial complete data. However, we argue that processing skyline queries with partial incomplete data in crowd-sourcing database has not received an appropriate attention. Therefore, an efficient approach processing skyline queries with partial incomplete data in crowd-sourcing database is needed. This paper attempts to present an efficient model tackling the issue of processing skyline queries in incomplete crowd-sourcing database. The main idea of the proposed model is exploiting the available data in the database to estimate the missing values. Besides, the model tries to explore the crowd-sourced database in order to provide more accurate results, when local database failed to provide precise values. In order to ensure high quality result could be obtained, certain factors should be considered for worker selection to carry out the task such as workers quality and the monetary cost. Other critical factors should be considered such as time latency to generate the results.

Corresponding Author:

Ali A. Alwan,
Department of Computer Science,
International Islamic University Malaysia IIUM,
P.O. Box 10, 50728 Kuala Lumpur, Malaysia.
Email: aliamer@iium.edu.my

1. INTRODUCTION

In the recent years, crowd-sourcing paradigm has become a powerful tool to bring human intelligence into information processing. This means use of human knowledge, idea, experience, and skills to solve many problems that are difficult or very expensive to answer correctly using machine-based algorithms. Examples of such problems include translation, handwriting recognition, photo tagging and audio transcription. The process of integrating those people to carry out the computation with the software systems known as a hyped human/ machine systems [1], [2]. The crowd-sourcing database (CSDB) is one of most modern databases that contains a large amount of heterogeneous data. The data in crowd-source database

might be in many forms and varies between web data, text, image, audio, personal data and so on [2]. CSDB uses many aspects of traditional database systems and SQL statements for processing complex queries. Furthermore, it adds the crowd functionality into a DBMS. The performance and cost of database query rely on crowd workers [1], [2].

Crowd-sourced databases promise a powerful solution rely on hundreds of thousands of ordinary workers (the crowd). Those workers attempt to provide information which is either unavailable or inaccurate exploiting the available data in the crowd database. This information help in performing various difficult computation functions such as matching, ranking, or aggregating results based on fuzzy criteria [5]. Accomplishing tasks by crowd workers might incur a monetary cost, latency, and the accuracy of the queries subject to the worker's quality [2].

In most of the contemporary database applications, the issue of missing data become a frequent phenomenon, especially when databases are generated automatically using various information extraction or information integration approaches. There are many reasons that make the database imprecise, exemplify the data integration from different huge databases. Furthermore, users might provide incomplete input by ignoring some data whether intentionally or unintentionally when working on the database. These activities influence negatively on the database contents and deteriorate the quality of the database contents. These reasons impact on the completeness and the correctness of the query result [3], [8]-[10].

In this regard, a skyline queries have gained considerable attention for assisting in multi-criteria decision making and decision support applications [4], [23], [24], [27], [28]. Skylines are those tuples which are the superior among all tuples in the database. The main issue concern when working with skyline queries is reducing the search space as small as possible and avoid the unnecessary exhaustive pairwise comparisons to determining the skylines. When operating on huge database such as crowd-sourcing database the advanced query operators, such as skyline queries help to filter unnecessary information efficiently before connecting with the crowd. Nevertheless, incomplete data has a negative effect on the performance of the skyline query process which might raise some issues including cyclic dominance and losing the transitivity property of skyline technique which is held on all existing skyline techniques. Besides, inaccurate data might also lead decline the quality of the results produced from the crowd-sourced database [4].

Our contribution in this paper can be summarized as follows:

a. To propose an approach that is able to return skylines from partially complete crowd-sourcing database with the intention of reducing the number of pairwise comparisons by eliminating the unnecessary tuples before applying the skyline technique.
b. To propose an approach that estimates the missing values of skylines utilizing crowd-sourcing database. The approach handles this issue taking into account the relative error produced when generating the estimated values, the monetary cost of estimating the values, reduce the time latency, and ensure high quality of the skyline results.
c. To propose a model of skyline queries for incomplete crowd-sourcing database.

The rest of this paper is organized as follows. Some necessary basic definitions and notations, which are used throughout the paper, are reported in section 2. The previous works related to this research are presented and examined in section 3. Moreover, the detail steps of the proposed technique are explained in section 4. Section 5 draws the conclusion of this paper.

2. PRELIMINARIES

This section gives some necessary definitions and annotations that are related to skylines queries in partially complete crowd-sourcing databases. These definitions and notations are important to explain the details of our proposed model. We also describe the concepts related to crowd-sourcing database, while concentrating on challenges of incomplete data on database. This include explaining the reasons that make the database to be incomplete and the methods used to manipulate the missing data.

2.1. Crowd-Sourcing Database

Crowd-sourcing database is a hybrid database system that automatically uses crowd-sourcing to integrate human input for processing queries that a normal database system cannot answer [5]. The crowd-sourcing system comprises of three components work to provide services to the users and ensure returning a high-quality result to the user. The components are: (i) requester, who submit the queries to the crowd and waits for the answers, (ii) platform (query execution engine), which is responsible to manage the given query by the user and retrieve back the answers generated by the workers, (iii) workers, who is responsible to run the assigned tasks and preparing and delivered the results to the target user [5], [25].

Crowd-sourcing database consists of a collection of enormous heterogeneous databases that are stored on different places on the internet and can be accessed through the crowd. Crowd-sourced databases

extend traditional databases to support more types of queries by means of the power of people, such as CrowdDB [5], Qurk [6], Deco [7], and so on. These crowd-sourced databases are associated with certain crowd-sourcing marketplaces, such as AMT (Amazon Mechanical Turk) [5], CrowdFlower [2] and so on, to attract crowds to work for them. Furthermore, crowd-sourcing database leverages many aspects of traditional database systems, however, there are also some differences between them which are explained as follows. Traditional databases are considered as a closed-world for query processing. Information that is not in the database is considered to be false or non-existent. Besides, traditional databases are extremely literal. For instance, users expect that the data has been properly cleaned and validated before inserting into database and do not negatively tolerate inconsistencies in data or queries [5].

There are three main critical factors that influence the data processing in crowd-sourcing system. This encompasses, monetary cost, time latency, and the accuracy of the results [1], [2], [8], [9], [10], [11]. These factors are further elaborated below.

Monetary Cost: Every task given by the user should be paid as the crowd is not free, therefore, requester needs to know in advance the estimated cost of the given task before asking the question and send it to the worker. Hence, if the number of tasks are very large, then crowd-sourcing can be very expensive [11].

Time Latency: This factor represents the running time to obtain answers from the crowd. For each task, the requester can set the time bound (e.g., 60 seconds) to answer it, and the worker must answer it within the stipulated time bound. The requester can also set the expiration time of the tasks if necessary, i.e., the maximum time that the tasks will be available to workers in the platform (e.g., 24 hours) [2].

Accuracy: Crowd-sourcing relies on human workers to achieve the job and the quality of the results obtained is highly influenced by the worker quality. Humans most often tend to make errors, particularly for heavy tasks that need long time and high attention. This, in turn, leads in the worst case to provide inaccurate results to the requester. Besides, some malicious workers are interested to obtain rewards and intentionally submit random answers to all questions or give wrong answers. This can significantly decline the quality of the results. Moreover, sometimes the worker misunderstands the task assigned which eventually negatively impact on the quality of the task accomplished. Lastly, workers may have different levels of expertise, and untrained worker may be incapable of accomplishing certain complicated tasks [2].

2.2. Incomplete Data

The database with incomplete data or missing value (also known as null value) are becoming very common in real world applications such as web database [26]. For example, a census database that allows null values for some attributes. A survey database where answers to one question cause other questions to be skipped. Also, a medical database that relates human body analyst (a substance that can be measured in the blood or urine) measurements to a number of diseases, or patient risk factors to a specific disease. The definition of incomplete database is given below [14].

Incomplete Database: given a database $D(R_1, R_2, ..., R_n)$, where R_i is a relation denoted by $R_i(d_1, d_2, ..., d_m)$, D is said to be *incomplete* if and only if it contains at least a tuple p_j with missing values in one or more dimensions d_k (attributes); otherwise, it is *complete*.

There are many reasons that make the database to be imperfect due to missing values. The data integration from different huge databases, depend on worker to input the data that also lead to missing data whether intentionally or unintentionally. Additionally, some types of databases in which data constantly changing such as temporal database. These causes might impact of the completeness and correctness of the result produced [12], [13].

Missing data is always unwanted and problematic, especially during query processing. Even if the rate of the missing values is small it might seriously bias inference. From the database literature it can be concluded that missing data mechanisms are often categorized into three different categories. First, missing completely at random (MCAR) which means there are no systematic differences between missing and observed values or put more simply, missing values appear to be completely random. Second, missing at random (MAR) where missing data occur based on some observed variables but not on the variable of interest or the variable under study. In particular this means that for each tuple most attribute values are available, while just some are randomly missing. Third, missing not at random (MNAR), in this category missing data occur based on n variable of interest or the dependent variable in statistical terms [14], [12].

The missing values have negatively influence on the quality of the database as more missing values lead to high rate of preference data. Therefore, estimating the missing values can enhance the quality of the database contents by transforming the database state from incomplete to complete. In this section we highlight on the prediction methods of handling incomplete data. We classify the prediction methods into two main categories, namely: statistical methods and machine learning methods [13], [14].

2.2.1. Statistical Methods\

Statistical methods are the base for many of the prediction approaches. The main aim of this method is preserving the distribution of the whole data by avoiding bias on the data distribution. The predicted values may benefit user at the database level but not at the tuple level. In many cases users are more concern on the individual value of the attribute rather than the whole data. Thus, these statistical methods may not be appropriate for preference queries [12], [14]. The statistical methods are as follows:

a. Single Imputation (SI) - In this method, the whole missing values for a given attribute are substituted by a single value. The methods that involve single imputation include mean, median, mode, maximum value in the attribute range, minimum value in the attribute range, and expectation maximization. This method is not satiable when the relevant error is very high.
b. Multiple Imputations (MI) - In this method, more than one estimated values are derived in order to fill in the missing values. The computation cost of this method is higher than SI, but it is more accurate and productive. This method also needs an appropriate mechanism to reflect the uncertainty of missing data and precisely estimate the missing values. An important argument in favour of this approach is that, frequently, attributes have relationships (correlations) among themselves. In this way, those correlations could be used to create a predictive model for classification or regression (depending on the attribute type with missing data, being, respectively, nominal or continuous).

2.2.2. Machine Learning Methods

Machine learning (ML) methods learn prediction models to handle the database with missing values based on the complete database instances. Some of the ML methods involve probabilistic approach to predict the missing values. ML methods are non-parametric as they do not rely on data distribution during the training or testing process. These methods are also called *parameter-free* methods or *distribution-free* methods. This method needs long time before getting the optimal values because it repeats work many times until it catches the lowest relevant error. There are several approaches that have been proposed which belong to the ML methods which include C4.5, Autoclass, Fractioning Cases, CN2, k-nearest neighbor (kNN) [14], [29].

2.3. Skyline Queries

Skyline queries have been widely used as an attractive operator in multi-criteria decision making applications, where many criteria are involved in the query statement to select the most suitable answer that fits the user requirements. Another domain that applied the skyline queries is the decision support system and recommender system, where these systems combine various interests to help users by recommending a strategic decision. Some definitions related to skyline queries are given in the following [4], [8], [11], [14], [27].

Dominance: given two tuples p_i and $p_j \in D$ data set with d dimensions, p_i dominates p_j (the greater is better) (denoted by $p_i > p_j$) if and only if the following condition holds: $\forall d_k \in d, p_i.d_k \geq p_j.d_k \wedge \exists d_l \in d, p_i.d_l > p_j.d_l$

Skyline: skyline technique retrieves the skyline, S in a way such that any skyline in S is not dominated by any other tuples in the data set.

Skyline queries: select a tuple p_i from the set of data set D if and only if p_i is as good as p_j (where $i \neq j$) in all dimensions (attributes) and *strictly* in at least one dimension (attribute). We use $S_{skyline}$ to denote the set of skyline tuples, $S_{skyline} = \{p_i \mid \forall\ p_i, p_j \in D, p_i \quad p_j\}$.

Comparable: Let the tuples a_i and $a_j \in R$, a_i and a_j are comparable (denoted by $a_i \varepsilon a_j$) if and only if they have no missing values in at least one identical dimension; otherwise a_i is incomparable to a_j (denoted by $a_i \notin a_j$).

Most of the skyline algorithms rely on the assumption of completeness of the data, i.e. all values of points are present and known. However, in many cases, this assumption does not hold. Thus, conventional skyline algorithms cannot be applied directly. Besides, the incompleteness of data leads to loss transitivity property of skyline technique which is held on all existing skyline techniques. This further leads to cyclic dominance between the tuples as some tuples are incomparable with each other and thus no tuple is considered as skyline [4].

3. RELATED WORK

This section presents and reviews the previous approaches related to skyline queries in complete traditional database. Furthermore, skyline queries in incomplete traditional database have been discussed.

Lastly, the previous strategies for skyline queries in crowd-sourcing databases have been explained and discussed deliberately.

3.1. Skyline Queries for Complete Data

First topic of skyline queries in complete database have been addressed with two different solutions the Block-Nested-Loop (BNL) and Divide-and-Conquer (D&C) [15]. In 2002, [16] presented a new online skyline algorithm called Nearest Neighbor (NN). It is different about the last algorithms that computed the Skyline in batch, but the NN algorithm returns the first results immediately, and find more results continuously. Furthermore, many approaches have been suggestion for skyline queries in complete database included: Sort-Filter-Skyline (SFS) [17], Branch-Bound-Skyline (BBS) [18], Linear Elimination Sort for Skyline (LESS) [19], Sort and Limit Skyline algorithm (SaLSa) [20], Cache Based Constrained Skyline (CBCS) [21], and others.

3.2. Skyline Queries for Incomplete Data

There are numerous algorithms which applied skyline queries for incomplete database. This includes but not limited the first work contributed in [4] proposed two algorithms for handling the skyline queries in incomplete data, namely: Bucket and Iskyline. Moreover, Replacement Based Sets Skyline Computation (RBSSQ) [22], sort-based Incomplete Data Skyline (SIDS) [23], Incoskyline [14] have been proposed to solve the issue of processing skyline queries in incomplete database. The previous works has only focused on how to tackle the issue of processing skyline queries in incomplete data, aiming at solving the issue of cyclic dominance and losing the transitivity property of skyline technique. However, not much attention has been paid on improving the quality of the skyline results by providing estimated values for the skylines with incomplete data. However, the only work that addresses the issue of predicting the missing values of the skylines is contributed in [14] aiming at manipulating the missing values before returning the skylines to the user. Additionally, the most of works designed to be used in traditional databases only.

3.3. Skyline in Crowd-Sourcing Database

Crowd-sourcing databases has various unique features compared with the traditional databases. Therefore, it might not be possible to directly apply approaches produced for traditional database in crowd-sourcing databases. To the best of our knowledge, there is not much attention given on processing of skyline queries in crowd-sourcing databases with incomplete data. Most of these studies concentrate on designing skyline strategies to estimate the missing values of skyline results ensuring low processing cost and less time latency and high accuracy for the estimated results.

In [8], [9] they have proposed a hybrid approach combining dynamic crowd-sourcing with heuristic techniques. The idea of the proposed approach relies on utilizing a set of heuristic techniques in order to predict the missing values for all tuples. It also exploits the contents of the crowd for some cases to improve accuracy of the estimated values. A new algorithm named CrowdSky has been proposed [11]. The algorithm considers three key factors that influence the process of skyline which are minimizing the monetary cost, reducing the latency, and improving the accuracy of a crowd-sourced skyline. Although the work assumed that the process of value estimation by the crowd will be applied on a set of virtual attributes. Which means the initial database is complete, but with insufficient data that might not help to provide an accurate answers for skyline queries. Therefore, utilizing the crowd database by generating some relevant virtual attributes and their values are predicted by the crowd workers to support the query results.

4. RESEARCH METHOD

In this section we present our proposed model to compute skylines in incomplete crowd-sourcing database. Fig 1. outlines the details structure of the model that consists of eight phases. In this framework, we assume a traditional local data set R contains tuples with some missing values and the user will submit a skyline query with some predefined preferences. The user query may also contain additional meta-data for describing the required result quality, maximal query budget, response time requirements, etc. Further details related to each component are explaining in the following subsection.

4.1. Data Filtration

This component analyzes the initial database in order to identify and remove the unnecessary tuples in the initial database. This process leads to eliminate the dominated tuples by applying the skyline process on the incomplete database. Therefore, this process helps to reduce the amount of data in the next processes. Many dominated tuples can be safely removed as they have no contribution in the skyline results. Doing so results into minimizing the monetary cost and the time latency to estimate the missing values from the crowd.

4.2. Sample Selector and Attribute Analyzer

The aim of this component is to generate a sample from the initial database to be used for analyzing the relationships between attributes. Analyzing the relationship between the attributes helps to estimate the missing values in the tuples. It is impractical to do this process over the whole data since the number of tuples in the initial database are very huge. Therefore, analyzing the attribute correlation directly on the whole data might incur high overhead and consume significant amount of time. Hence, extracting a small portion of the database for attribute analysis helps to speed up the process and reduce the cost without compromising the quality of the estimated values. The sample will be generated randomly and within a predefined range. Attribute analysis assists in generating the approximate functional dependencies between attributes. This process is further explained in the next subsection.

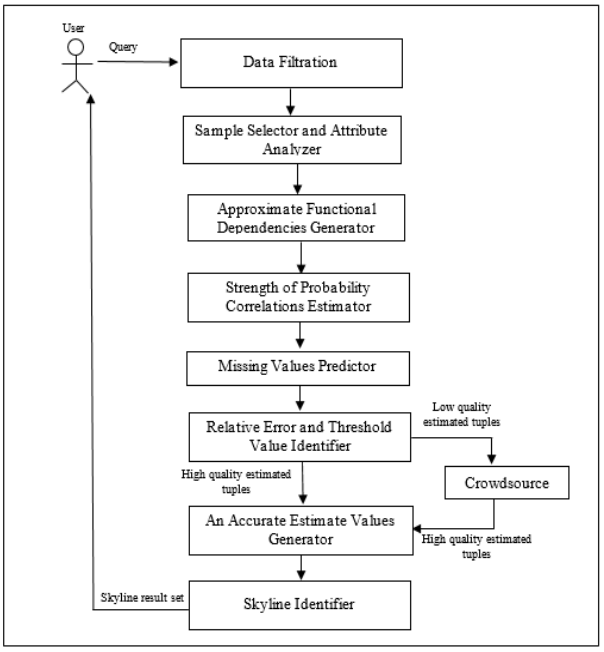

Figure 1. The Proposed Model of Skyline Queries in Crowd-Sourcing Incomplete Databases

4.3. Approximate Functional Dependencies Generator

The aim of this component is to derive the relationship between the attributes. For this purpose, the system analyzing the values on the selected sample trying to capture the relationships between attributes and determine how the value in one attribute affects the values of the other attributes.

Definition: Approximate Functional Dependency (AFD) Given a relation *R*, a subset *X* of its dimensions, and a single dimension d_i of *R*, we say that there is an approximate functional dependency (ADF) between X and di, denoted by Xf $\rightarrow$ d_i, if the corresponding functional dependency $X \rightarrow di$ holds on all but a small fraction of the tuples of *R.* The set of dimensions *X* is called a *determining set* of di denoted by $DtrSet(d_i)$ [14].

To generate the AFD, we first divide the sample set with missing values into two sets. The first set contains all values that missing in the identical attribute that needs to be estimated, and the second set contains the remaining values (complete). The values of the second set are analyzed to capture the relationships between the attributes of the relation. Then the AFD is generated, that help us in specifying how the value of one attribute influences the values of the other attributes. Then to ensure that the derived AFD reflects the relationships between attributes [14].

4.4. Strength of Probability Correlations Estimator

The aim of this component is to choose the attribute that has high influence on other attributes rely on the strength of probability correlations between attributes, for instance d_i and d_j, and to which extent the value of d_i influences the value of d_j. The strength of probability correlations between d_i and d_j is denoted as $P(d_i, d_j)$ and is formulated as follows [14]:

$$P(d_i, d_j) = \frac{|d_j|}{|d_j, d_i|}$$

Where |.| refers to the number of distinct values. The value of $P(d_i, d_j)$ indicates the strength that every distinct value in d_j is associated with a unique value in d_i.

4.5. Missing values predictor

This component is responsible to substitute the missing values of the database with some imputed values derived using the available data of the database. This is achieved based on counting the frequency of the value exists with several alternative values by replace the missing values with the estimated values. In this process there might be many estimated values for a dimension that need to be considered. However, in some cases there might be more than one AFD that can be generated between the dimensions. In this case, the results of the AFDs are combined to estimate the missing value.

4.6. Relative error and threshold value identifier

This component concerns on checking the accuracy of the estimated values generated using the existing data of the database. After all missing values has been estimated in last stage of the proposed model, the system computes the relative error which indicates the error rate between the obtained estimated values when the database has some missing values in one or more dimensions and the real values when the database is complete with no missing values. If the relative error rate is larger than the user defined threshold value, then the estimated values will be discarded. Otherwise, accept the estimated values and use them in the skyline results. In one case, this helps us to benefit from the existing data in predicating the missing values and reduce the amount of data to be generated from the crowd. Moreover, it also helps to reduce the monetary cost by avoiding sending many request to the crowd workers. In other case, if the quality of estimated values is low, then we may refer to the crowd-sourced database to obtain better results that might be within the user defined threshold value. Hence, we can conclude that our model has considered both scenarios and ensuring the cost and the quality of the results have been maintained.

4.7. An accurate estimate values generator

This component is responsible to generate estimated values for those skylines with incomplete data. The process starts by evaluating the quality of the estimated values through comparing the relative error between the real missing value and the estimated one against the user defined threshold value. If the relative error produced less than the threshold value, then the estimated values will be accepted for replacement. Otherwise, the estimated values will be ignored and attempt to exploit the data in the crowd to generate more accurate values. We argue that utilizing the local data in the database helps to estimate a large number of estimated values with high quality. This is because the value estimated are generated based on exploiting the explicit relationship between the dimensions and via identifying the strength of the probability correlation between the dimensions. However, for certain cases, particularly for data with high missing rate or the data is independent, then it might be challenging to estimate accurate values. Therefore, we request the crowd to generate more precise estimated values which results into a relative error below the given threshold value. Basically, this component has two sub-components which are local estimation and crowd-sourcing estimation. These sub-components are further explained in the following.

4.8. Skyline identifier

This component is the last component in the proposed framework which aims to identify the final skylines of the database and get the query result. The pairwise comparison process will be applied on final complete database to get skylines. This helps users in selecting the relevant skylines from several candidate skylines.

5. CONCLUSION

Skyline query is the process of identifying the non-dominated tuples in the database and returning them to the end user based on the user given preference. It is one of the most predominant type of preference queries that have been introduced into the database in the last decade. This paper attempts to introduce a model to process skyline queries in crowd-sourcing databases with partially complete data. The model describes the necessary components to process skyline query and return skylines of the database with the intention of reducing the number of pairwise comparisons between tuples which results into reducing the cost of skyline query processing and minimize the time latency. Most importantly, the model concentrates on

providing skylines of incomplete database with complete data derived from the crowd-sourcing databases. This is accomplished through estimating the missing values in the database which in consequence provide more precise results helping user to make the most appropriate decision. We plan to implement this model and evaluate its performance and efficiency in provide precise values. Factors such as relative errors, monetary cost, accuracy of the results, time latency will be taken into consideration when develop the model.

ACKNOWLEDGEMENTS

This research is supported by the project FRGS15-205-0491, Ministry of Education, Malaysia.

REFERENCES

[1] A. Parameswaran, and N. Polyzotis, *"Answering queries using humans, algorithms and databases,"* Presented at the 5th Biennial Conference on Innovative Data Systems Research (CIDR'11), Asilomar, California,USA, 2011.
[2] G. Li, J. Wang, Y. Zheng, and M. J. Franklin," Crowdsourced data management: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 28(9), 2016, pp. 2296-2319.
[3] G.Wolf, A. Kalavagattu, H. Khatri, R. Balakrishnan, B. Chokshi, J. Fan, and S. Kambhampati, "Query processing over incomplete autonomous databases: query rewriting using learned data dependencies," *VLDB Journal-The International Journal on Very Large Data Bases*, 18(5), 2009, pp.167-1190.
[4] M. E. Khalefa, M. F. Mokbel and J. J. Levandoski, "*Skyline query processing for incomplete data*," the 24th International Conference on Data Engineering (ICDE), Cancun, Mexico, 2008, pp.556- 565.
[5] M. J. Franklin, D. Kossmann, T. Kraska, S. Ramesh and R. Xin, *"CrowdDB: answering queries with crowdsourcing,"* ACM SIGMOD International Conference on Management of data SIGMOD'11, Athens, Greece, 2011, pp. 61–72.
[6] A. Marcus, E. Wu, D. R. Karger, S. Madden and R. C. Miller, *"Crowdsourced databases: query processing with people,"* the 5th Biennial Conference on Innovative Data Systems Research (CIDR'11), Asilomar, California, USA, 2011, pp.211-214.
[7] H. Park, H. Garcia-Molina, R. Pang, N. Polyzotis, A. Parameswaran, and J. Widom, *"Deco: A system for declarative crowdsourcing,"* the VLDB Endowment, 5(12), Istanbul, Turkey, 2012, pp.1990-1993.
[8] C. Lofi, K. El Maarry and WT. Balke, *"Skyline queries over incomplete data-error models for focused crowd-sourcing,"* the 32th International Conference on Conceptual Modeling, Hong Kong, China, 2013, pp.298-312.
[9] C. Lofi, K. El Maarry and WT. Balke, *"Skyline queries in crowd-enabled databases,"* the 16th International Conference on Extending Database Technology EDBT/ICDT '13, Genoa, Italy, 2013, pp. 465-476.
[10] K. El Maarry, C. Lofi, and WT. Balk, "Crowdsourcing for query processing on web data: a case study on the skyline operator," *Journal of Computing and Information Technology – CIT*, 23(1), 2015, pp.43-60.
[11] J. Lee, D. Lee, and S.W. Kim, *"CrowdSky: skyline computation with crowdsourcing,"* the 19th International Conference on Extending Database Technology (EDBT), Bordeaux, France, 2016, pp.125-136.
[12] M. A. Soliman, I. F. Ilyas and S. Ben-David, "Supporting ranking queries on uncertain and incomplete data," *the Very Large Database Journal, VLDB*, 19(4), 2010, pp.477- 501.
[13] L. Wang and R. Jones, "Big data analytics for disparate data," *American Journal of Intelligent Systems*, 7(2), 2017, pp.39-46.
[14] A. A. Alwan, H. Ibrahim, N. I. Udzir and F. Sidi, *"Estimating missing values of skylines in incomplete database,"* the The Second International Conference on Digital Enterprise and Information Systems (DEIS2013), Kuala Lumpur, Malaysia, 2013, pp.220-229.
[15] S. Borzsony, D. Kossmann and K. Stocker, *"The skyline operator*," the 17th International Conference on Data Engineering (ICDE01), Heidelberg, Germany, vol. 3896, 2001, pp. 421-430.
[16] D. kossmann, F.Ramsak and S. Rost, *"Shooting stars in the Sky: an online algorithm for skyline queries,"* the 28th International Conference on Very Large Data Bases (VLDB'02), Hong Kong, China, 2002, pp. 275- 286.
[17] J. Chomicki, P. Godfrey, J. Gryz and D. Liang, *"Skyline with presorting,"* the 19th International Conference on Data Engineering (ICDE03), Bangalore, India, 2003, pp.717-816.
[18] D. Papadias, Y. Tao, G. Fu and B. Seeger, *"An optimal and progressive algorithm for skyline queries,"* ACM Conference on the Management of Data (SIGMOD), San Diego, California, USA, 2003, pp. 443-454.
[19] P. Godfrey, R. Shipley and J. Gryz, "*Maximal vector computation in large data sets,"* the 31st International Conference on Very Large Data Bases (VLDB'05), Trondheim, Norway, 2005, pp. 229-240.
[20] I. Bartolini, P. Ciaccia and M. Patella, *"SaLSa: computing the skyline without scanning the whole sky,"* the 15th International Conference on Information and Knowledge Management (ICIKM06), Arlington, Virginia, USA, 2006, pp. 405- 414.
[21] M. L. Mortensen, S. Chester, I.Assent and M. Magnani, *"Efficient caching for constrained skyline queries,"* the 18th International Conference on Extending Database Technology (EDBT), 2015, pp.337- 348.
[22] M. S. Arefin and Y. Morimoto, "Skyline sets queries for incomplete data," *International Journal of Computer Science & Information Technology*, 4(5), 2012, pp. 67-80.
[23] R. Bharuka and S. P. Kumar, *"Finding skylines for incomplete data,"* the 24th Australasian Database Conference, Adelaide, Australia, vol.137, 2013, pp. 109-117.
[24] A. A. Alwan, H. Ibrahim, N. I. Udzir and F. Sidi, "An efficient approach for processing skyline queries in incomplete multidimensional database," *Arabian Journal for Science and Engineering*, 41(8), 2016, pp.2927-2943.

[25] E. D. Difallah, M. Catasta, G. Demartini, G. P. Ipeirotis and P. Cudré-Mauroux, *"The dynamics of Micro-Task crowdsourcing,"* the 24th International World Wide Web Conference (WWW 2015), Florence, Italy, 2015, pp. 238-247.
[26] G. Canahuate, M. Gibas and H. Ferhatosmanoglu, *"Indexing incomplete databases,"* the 10th International Conference on Extending Database Technology (EDBT), Munich, Germany, 2006, pp.884-901.
[27] Y. Gulzar, A. A. Alwan, N. Salleh, I. F. Al-Shaikhli and S. I. Mairaj Alvi, *"A Framework for Evaluating Skyline Queries over Incomplete Data,"* Procedia Computer Science, 94, 2016, pp.191-198.
[28] Y. Gulzar, A. A. Alwan, N. Salleh, and I. F. Al-Shaikhli, *"Skyline query processing for incomplete data in cloud environment in Zulikha, J. & N. H. Zakaria (Eds.),"* the 6th International Conference on Computing & Informatics, (ICOCI), Kuala Lumpur, 2017, pp.567-576.
[29] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, "A survey of machine learning for big data processing," *EURASIP Journal on Advances in Signal Processing*, 2016, DOI 10.1186/s13634-016-0355-x.

BIOGRAPHIES OF AUTHORS

Marwa B. Swidan is a PhD student at International Islamic University Malaysia, IIUM, Malaysia. She earned her B.Sc. and M.Sc. degrees in Computer Science from Tripoli University, Libya in 2004 and 2009 respectively. Her research interests include crowdsourcing, data mining/exploration, database systems and skyline queries.

Ali A. Awan is currently an assistant professor at Kulliyyah (Faculty) of Information and Communication Technology, International Islamic University Malaysia (IIUM), Malaysia. He received his Master of Computer Science (2009) and Ph.D in Computer Science (2013) from Universiti Putra Malaysia (UPM), Malaysia. His research interests include preference queries, skyline queries, probabilistic and uncertain databases, query processing and optimization and management of incomplete data, data integration, location-based social networks (LBSN), recommendation systems, and data management in cloud computing.

Sherzod Turaev obtained his PhD in Computer Science in 2010 from University Rovira i Virgili, Tarragona, Spain and PhD in Mathematics in 2001 from National University of Uzbekistan. He was a postgraduate researcher in University Putra Malaysia from 2009 to 2012. He is a faculty member at Department of Computer Science, Faculty of Information and Communication Technology, International Islamic University Malaysia since 2012. His research interests include graph theory, regulated rewriting systems, formal languages and automata, bio-computing, machine learning and information security.

Yonis Gulzar is currently a Ph.D. student at International Islamic University Malaysia (IIUM), Malaysia. He has received his Master of Computer Applications (2013) from Bangalore University, India. His research interests include preference queries, skyline queries, probabilistic and uncertain databases, query processing and optimization and management of incomplete data, data integration, and data management in cloud computing.