K-nearest Neighbor Skyline Queries in Mobile Environment

Jing Nie[1*], Wei Weng[1], Linan Sun[2]

[1]*Department of Software Engineering, Xiamen Institute of Software Technology, Xiamen, 361024, China.*

[2]*College of Computer and Information Engineering, Xiamen University of Technology, Xiamen 361024, China;*

Received 1 December 2014, www.cmnt.lv

Abstract

With the popularity of portable mobile Internet device, the applications based on query are increasingly enriched. This kind of skyline query problems is not only related about the positions, but also the constantly moving queries. Range-base queries are widely used to solve the problem in recent algorithm, but focusing more on computing all skyline points. However, users are interested in nearby skyline points in mobile environments. Two different algorithms are proposed and the characteristics and applied range are analyzed in the paper to solve the problem, after researching relevant properties based on the basic concept of the skyline query.

Keywords: skyline queries; k-nearest neighbor; query optimization

1 Introduction

Skyline query is a kind of decision support technology in database management system [1]. In the scenarios of rich high dimensional data, it returns the set of points that are not dominated by any other point [2]. That is, these data serving the multiple objective needs of people have been found. In the last ten years, many theory and technology are proposed to solve it and related applications are getting enormous progress, such as continuous skyline queries [3], reverse skyline queries [4], subspace skyline queries [5], probabilistic skylines [6]. Literature [7] summarized more detail discussion about this subject.

LBS (location based services) is becoming increasingly common because of the ubiquity of mobile devices such as cell phones and iPad [8]. For example, mobile users could be interested in parking lot that are near, reasonable in pricing, and plenty of empty bays, so skyline queries are an important operator under such circumstances [9]. Mobile devices based skyline queries bring challenge because of the motility and location correlation. Specifically, we sum up in three key areas.

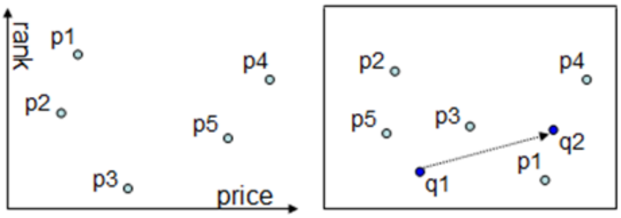
(1) Due to motility of these devices, the input of a user location is often changed, i.e., the query point is often changed. So the skyline results depend on the location of the query point and often changed.

(2) It may be difficult to get the precision query location. Some mobile users worry about their privacy of position. On the other hand, mobile devices have the limited precision in returning location. So our application systems should accept imprecise query location, and even a range.

(3) What mobile users interest in often are a few skyline points around their position. A simple solution might be computing all skyline points at first, then select the part around user's position. But if our data set is very large, this method waste computational resources obviously. Further analysis and process will be done for computing Local skyline points.

To study the problem of the above scenario, Fig.1 shows an example of five restaurants (P_1 to P_5) and a query q (q_1 and q_2 are different position of q). Fig. 1(a) shows the tow attributes of restaurant: rank and price. Restaurant with the higher the rank and the lower price is better. In general, we cannot find the best one in multi-criteria analysis, just like this example, rank and price are contradictory, so no restaurant is the best than all other rest. Skyline queries extract most preferred alternatives, i.e., the restaurants that are no one is better than it. If ignoring the spatial criteria, P_1 and P_2 are skyline points in Fig. 1(a). In mobile environment, given two points P_1 and P_2 , if P_1 is no worse than P_2 in non-spatial attributes and P_1 is closer to the query point than P_2 , we say P_1 dominates P_2 . In Fig. 1(b), a skyline query is to find all restaurants that are not dominated by any other restaurant. When query point located in q_1 , P_1 , P_2 and P_5 are all skyline points. On the other hand, when query point located in q_2 , P_1 and P_2 are skyline points. If the data set is enormous, it is not only time consuming but also unnecessary to compute all of skyline points, so our work focus on Top-K strategies.

[*]Corresponding author's Email:56564995@qq.com



(a) Price and rank (b) Location coordinates

FIGURE 1. Example of Top-K skyline queries in mobile environment

2 Related work

In essence, our work inherits the characteristics of a skyline query and range-based query. As such, we review the existing work on these two queries.

Since the skyline query processing was firstly introduced into the database community by Borzonyi et al. [10], a number of algorithms have been proposed from then on to improve efficiency of algorithm and data updating[11-13].K-Nearest Neighbor Queries is an important operation in Spatial Database. Point-KNN query is a kind of common query algorithms and the algorithm returns the closest k objects of a query from database. If k=1, the algorithm [14] can solve it by making use of Voronoi diagram in the period of O(nlogn) and if k>1, the algorithm[15-16] proposes different solutions. However, the objects studied in these literatures are static, that is to say, each attribute values of element in DES (data element set) does not change. In the mobile environment, each what I say before contains the spatial property, that is distance between query and every object and this distance ,will change when moving so that the Skyline query results also change with the moving. By the sample of query location, Zhexuan Song et al. [17] research the critical distance of k nearest neighbor points of the query and overlap between the corresponding result sets. Combining with the pre-fetching technique, they put forward the branch-and-bound algorithm, which avoid a lot of operations that repeat scanning the database. Tao et al. [19] see the query tracks as segments, and propose the algorithm to get all the nearest neighbor objects of the segments, which we call it continuous nearest neighbor queries, CNN.

The algorithm raise the concept of split points, avoiding the general problems of query sampling, that is the low speed leading to the wrong result. The algorithm can find all the split points through a traversal of database, the query tracks are separated into the smaller intervals by the split points and querying their nearest neighbor objects in each interval are the same.

The algorithm named rang nearest neighbors (RNN) proposed by Haibo HU et al. [20], expands the query range to the hyper rectangle, to return the nearest neighbor objects in any point of the hyper rectangle. It's easy to see that the data space is corresponding to PNN and CNN when it is 0 dimensional space and 1 dimensional space. In the research of other Spatial query processing [21-22], method as location cloaking has been raised in order to protect the location privacy of users. That is, when the queries based on location have been put forward by users, the agents will change current positions to the cloaking region according to the requests of users, and the servers return all the possible results of this cloaking region to the agents then, and the agents will screen out a result for users. Thus, the location of users is unknown for servicers. However, only the distance problem of objects and queries has been considered has been considered in these algorithms, without the domination problem between objects. The problem of continuous skyline queries is proposed first. It is pointed that the skyline points in lower dimensional space must be the skyline points in high dimensional space.

Therefore, the skyline set is divide into two part in the paper: SK_{ns} and SK_{chg} , and only SK_{chg} will increase or decrease with the query. In addition, if the point p is the furthest skyline point in distance query of SK_{ns} , then any other points not approach to point p cannot be the skyline points. The changing conditions of $SK_{c\ h}$ have been discussed on the above properties. A method of computing skyline sets in the objets near the mobile equipments has been raised in the literature [23], which, however, is not the skyline points on the significance of global data set, and the spatial nature has not been considered in the dominance relationship. The relevance algorithm about the spatial skyline query is proposed in literature [24-25]. The spatial skyline query concentrates on the dominance of space. Given a geo-database D (like cinema gas station etc.) and query set Q, for $P_1, P_2 \in D$, if the distance of P_1 away from all the queries is not far from P_2 , and P_1 is closer than P_2 for at least one query, so P_1 spatially dominate P_2 . It can be seen that the dominance of space is not be considered in this kind of query. Having considered the dominant of objects on the spatial and non spatial attributes, Xin Lin [26] and others proposed the range-based skyline queries algorithm. Relative to the point query PSQ and continuous query CSQ, the demands to the position precision of queries were further relaxed obviously. Given a range (like rectangular region), there may be quantities of queries in this region. So, the index-based algorithm which become I-SKY the is used in the paper. Calculate the skyline scopes of every object in advance, and create index for the ambit, thus the skyline set based on the range query can be calculated effectively. The algorithm N-SKY were put forward for the problem of hard to maintain the indexes due to the high-speed of query, which will change the range-based skyline queries to the segment-based skyline queries.Muhammad Aamir Cheema and others discuss the skyline queries on the mobile environment according to the safety region. The so-called safety region means the results of query are not change in these region. If the moving query leave the safety region, the query result must be updated and new safety region will arise.

3 Preliminaries

3.1 PROBLEM DEFINITION

If there is an object set $D=\{p_1,p_2,...,p_n\}$,every element $p_i \in D$ has m attributes as P_{ij} , $i=1,2,...,n$, $j=1,2,...,m$. Without loss of generality, we stipulate that the last attribute p_{im} is the spatial attribute and others are non spatial attribute. In the case of confusion, p_{ij} is also on behalf of the values in the j attribute of the object p_i. The value of spatial attributes are designed by the distance between object and query. And we stipulate that any two objects are absolutely different. To easily discuss, we range the query q within the two-dimensional space Q . In the practical applications, Q would be represented as a city and the q would be as the location of the mobile objects who issue queries. For example, one in a moving car is using cellphone to search for the nearest restaurant in the city, and the query position is the same as phone's GPS coordinates. Under mobile environment, distant between object p_i and query q is changing continuously, and $p_{im}=dis(p_i,q)$ is changing continuously.

Definition 1 (Non-Spatial Dominance). Set 2 objects p_i and p_j. If the spatial attributes of p_i is no worse than that of p_j, the p_i spatially dominate p_j. Representation formalism is as follows. If $p_{ik} \le p_{jk} (k=1,2,...,m-1)$, $p_i < p_j$. All the object collections of non-dominated solution p_i is as $NSD(p_i)=\{p_j \mid p_j < p_i\}$. All of the objects which are non-dominated by others in dataset D are written as $NS-Skyline(D)$.

Definition 2 (Dominance). If $p_i < p_j$, and p_i is closer to q that p_j, or $p_{im} < p_{jm}$, we call p_i dominates p_j as $p_i <_q p_j$. p_i is called dominator and p_j becomes dominatee. All of the object collections of p_i is dominated under query q is $SD_q(p_i)=\{p_j \mid p_j <_q p_i\}$. It's obvious that $SD(p_i) \subseteq NSD(p_i)$ according to the definition.

Definition 3 (skyline point). For the dataset D , in the case that the query $q \in Q$ is given, the objects has not been dominated are called skyline point of query q. All the skyline points make up the skyline set, as $Skyline_q(D)$.

Definition 4 (top-k skyline points). For the dataset D , in the case that the query $q \in Q$ is given, the first k skyline points are closest to q will be the top-k skyline set, as $TK-Skyline_q(D)$.

Definition 5 (Skyline Scope) [26]. For any data $p_i \in D$, and its outline region of two dimensional space can be represented as $SS(p_i)=\{q \mid p_i \in Skyline_q(D)\}$. In other words, in the space $SS(p_i)$, p_i is the skyline point, which means $dis(p_i,q) < dis(p_j,q)$, when $\forall p_j \in NSD(p_i)$. The skyline set can be obtained through getting the intersection of half-plane containing p_i ,which is partitioned by the perpendicular bisector in the middle point of p_i and $NSD(p_i)$. The detail refers to the literature [29]. Obviously, if $NSD(p_i)=\phi$, the outline region of p_i is the whole query area.

The point of our work is that, the query set Q consist of continuously moving is limitless and then giving a query q in random, how to find the first k skyline points under the query.

3.2 RELATED PROPERTIES

Property 1 $SD(p_i) \subseteq NSD(p_i)$
We can easily hold it up according to Definition 1 and Definition 2. In line with this properties, we can judge if the object in $NSD(p_i)$ arrange p_i under the query q by judging whether p_i is the skyline point under the query q. If any object in $NSD(p_i)$ doesn't dominate the p_i under the query q, the p_i is the contour of query q.

Property 2 For any query q, if $p_i \in NS-Skyline(D)$, $p_i \in Skyline_q(D)$.

The property is pointed out in literature [3].

Property 3 Any a non-skyline point is dominated by another skyline point.

At first, if given a object p_i is a non-skyline point, it must be dominated by another one p_j . Based on Definition 2, it's known that p_i can't dominate itself, thus $p_i \ne p_j$. Similarly, p_i can't dominate p_j, that is to say, the dominance relation is not symmetric. If p_j is not skyline point, there must exist another object p_k which dominate p_j in the same way. Time and time again, repeatedly, a dominating chain $p_k <_q p_j <_q p_i$ comes into being in the process of finding the skyline point. the finding process is limited, in the limited dataset D , because objects on the dominating chain are different to each other. A skyline point must be found in limit steps to the last object of the chain. Dominance relation is transitive, that is, if $p_k <_q p_j$ and $p_j <_q p_i$, $p_k <_q p_i$ can be obtained. Therefore, the last object in the chain is skyline point which dominate p_i.

Property 4 Sort the objects based on the distance to query q from small to large and set the sequence as

$p_1, p_2, \cdots, p_k, \cdots p_n$. If $p_k \notin NS-Skyline(D)$ and there is no $j<k$ to make the $p_j \in Skyline_q(D) \cap NSD(p_k)$ work, then $p_k \in Skyline_q(D)$.

It's known that $p_k \notin NS-Skyline(D)$, so $NSD(p_k)$ is non-null. It can be got by Property 1, if p_k is not dominated by any object in $NSD(p_k)$, $p_k \in Skyline_q(D)$. Under the circumstance that the object has been sorted, the object behind p_k in the row is further to the query q than p_k in the $NSD(p_k)$, which will not dominate p_k again. Now take the set $P\_NSD(p_k)$ into consideration which consists of the objects in $NSD(p_k)$ in front of p_k. The objects are either the skyline points or not.

According to the property 3, if non-skyline points in $P-NSD(p_k)$ dominate p_k, there must exist a skyline point in $P-NSD(p_k)$ dominating p_k. Therefore we only need to judge whether p_k is dominated by the skyline point in $P-NSD(p_k)$. If there is no $j<k$ let $p_j \in Skyline_q(D) \cap NSD(p_k)$, it's explained that all the skyline points do not dominate p_k, so $p_k \in Skyline_q(D)$. Ulteriorly, the objects behind p_k in $Skyline_q(D)$ cannot dominate p_k, thus, if objects in $Skyline_q(D)$ in front of p_k can be found before judging whether p_k is the skyline point, and if the intersection of part of skyline points and $P-NSD(p_k)$ is null, p_k is not skyline point. Otherwise, p_k is the skyline point.

4 Algorithm description

4.1 ALGORITHM BASED ON SKYLINE POINTS PROPERTIES

Given a query q, according to the Property 2 and Property 4 , its Top-k skyline points can be found as shown in Algorithm 1.

Algorithm 1 Top-k Skyline query base on the skyline points properties

Input: dataset D, query $Q=\{q_1, q_2, ..., q_3\}$, the number of closest skyline points k

Output: $TK-Skyline_q(D)$

(1) count $NS-Skyline_q(D)$ and $NSD(p)$ of each object p, and make $TK-Skyline_q(D)=\phi$;

(2) $c=1$;

(3) if $c>n$, turn to (9);

(4) $s=1$;

(5) if $s>k$,turn to (8);

(6) find unhandled object p which closest to query q_c based on the distance to the query, and mark it as handled.

(7) if $p \in NS-Skyline(D)$, merge it into the set $TK-Skyline_{qc}(D)$ and $s++$; on the contrary, if $NSD(p) \cap TK-Skyline_{qc}(D)=\phi$, merge it into the set $TK-Skyline_{qc}(D)$ and $s++$; turn to (3);

(8) $c++$;

(9) end;

Set up R-tree[24] using non-spatial attributes and use the method referring to literature [12] to compute $NS-Skyline(D)$. The weakness of algorithm1 is obvious that when the query is moving, the distances from objects to the query in different positions are changing and the corresponding skyline set is also changing so that the k-nearest-neighbor skyline points, which makes a bigger time cost. In addition, if the dataset D is large, it will take a lot to do the data-query.

4.2 CONTOUR ALGORITHM

According to the Definition 5 in Part 3.1, a query can be seen as a region rather than a point. That is to say, the whole section can be divided into different regions, the query in any one position in a region and the corresponding skyline set is constant. Meanwhile, notice that outline regions of different object may overlap. Explain it further using the example in literature [26], as shown in Fig.2. Set a object set $D=\{a,b,c,d,e\}$, in which a, d and e is non-spatial dominated by other data, so they are skyline points naturally. Other non-spatial dominance relation include $a<b<c$, $d<b$ and $e<b<c$. Thus, $NSD(a)=NSD(d)=NSD(e)=\phi$, $NSD(b)=\{a,d,e\}$, $NSD(c)=\{a,b,e\}$. The dotted line triangle is a region surrounded by the perpendicular bisectors of edge ca, cb and ce. The region is the skyline space $SS(c)$ of c, and in the same way, the solid triangle is the skyline space $SS(b)$ of b . For $SS(c)=R_1 \cup R_2$, $SS(b)=R_2 \cup R_3$, $SS(c) \cap SS(b)=R_2$ and R_3 is communal outline region of b and c. When query is in the region R_1, the skyline point are $\{a,c,d,e\}$. When query is in the region R_2, the skyline points including all the object is $\{a,b,c,d,e\}$. When query is in $R3$, the skyline points are $\{a,b,d,e\}$. Of course, if skyline points are outside $R_1 \cup R_2 \cup R_3$, the points are $\{a,d,e\}$. MX-CIF Quadtree [30] can be used to store the outline region index, the efficiency of this the data structure is higher for the overlapping spatial index than general R tree. For a query, only to search for the object outline regions where the query is in, then the skyline set is consist of the objects. Sorting this objects by their distance to the query position, the k-nearest neighbor skyline points can be got. As long as query do not leave the region, the skyline set will not change. If query moving in this region, re-select k nearest object in the

skyline set. For convenience, the public outline region the query q is in is written as $comSS(q)$.

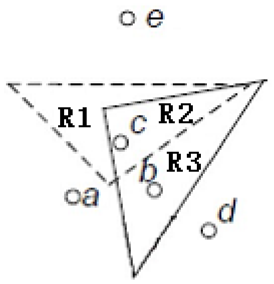


FIGURE 2 Outline region

Algorithm 2 Top-k skyline query based on outline region

Input: dataset D, query $Q=\{q_1,q_2,...,q_n\}$, the number of nearest skyline points k

Output: $TK-Skyline_q(D)$

(1) compute skyline space $SS(p)$ of every object p;

(2) $k=1$;

(3) If $k==1$, compute the corresponding public skyline region $comSS(q_k)$; other wise, judge whether q_k is in the $comSS(q_{k-1})$, if so, re-compute $comSS(q_k)$;

(4) Search first k objects nearest to q_k in the public skyline query, which make uo the $TK-Skyline_{qk}(D)$;

(5) $k++$; if $k \le n$, turn to (3);

(6) End.

5 Conclusion

Top-k skyline query algorithms are proposed under two mobile circumstances in the paper, and these two are based on different data structures. Data structure of Algorithm 1 is easier, which only save the structure of object set D. But two data structures is demand in Algorithm 2. One is to save the object set D, and the other one is to save skyline region. In addition, these two algorithms are used in different situations. Under the circumstance that parallel query rate is not high and the concurrency value is not large as the environment of extreme single users, Algorithm1 can meet needs. On the contrary, Algorithm2 is more suitable. Further studies will focus more on improving two algorithms based on comparison under different data structures, different distributions and different dimensions.

Acknowledgments

Fund Project: the Project of Xiamen Science and Technology Program under Grant Nos.3502Z20133041, 3502Z20133042, the Xiamen University of Technology’s International Cooperation and Exchange Project under Grant No.E201301300.

References

[1] Tan Kian-Lee, Eng Pin-Kwang, Ooi Beng Chin 2001 Efficient Progressive Skyline Computation *Proc. the 27th VLDB Conference (Roma, Italy)* 301-10

[2] Bhattacharya B, Bishnu A, Cheong O, Das S, Karmakar A, Snoeyink J. 2010 Computation of non-dominated points using compact Voronoi Diagrams *Proc. International workshop on Algorithms and Computation (Dhaka, Bangladesh, February 10-12)* 82-93

[3] Huang Zhigong, Lu Hua, Qqi Beng Chine,Tung Anthony KH 2006 Continuous skyline queries for moving objects *IEEE Transactions on Knowledge and Data Engineering* **18**(12) 1645-58

[4] Dellis E, Seeger B 2007 Efficient computation of reverse skyline queries. *Proc. of the 33th VLDB Conference (Vienna, Austria ,September 23-28)* 291-30

[5] Jin Wen, Tung Anthony K H, Ester Martin, Han Jiawei 2007 On Efficient Processing of Subspace Skyline Queries on High Dimensional Data *Proc. the 19th international conference on scientific and statistical database management* 12-22

[6] Jiang Bin, Pei Jian, Lin Xuemin, Yuan Yidong 2012 Probabilistic skylines on uncertain data: model and bounding pruning-refining-refining methods *Journal of Intelligent Information System* **38**(1) 1-39

[7] Chomicki J, Ciaccia P, Meneghetti N 2013 Skyline queries, Front and Back *SIGMOD Record*. **42** (3) 6-18

[8] Cai Y and Xu T 2008 Design, Analysis, and Implementation of a Largescale Real-timen Location-based Information Sharing System *ACM MobiSys* 106

[9] Yang Y, Papadopoulos S, Papadias D, Kollios G 2009 Spatial Outsourcing for Location-based Services *Proc.* of *the 2009 ACM SIGMOD International Conference on Management of data* 5-18

[10] Borzonyi S, Kossmann D, Stocker K 2001 The Skyline Operator. *International Conference on Data Engineering* 420-30

[11] Kossmann D, Ramsak F, Rost S 2002 Shooting stars in the sky: An online algorithm for skyline queries *Proc. of the 28th VLDB Conference* 275-86

[12] Papadias D, Tao Y, Fu G, and Seeger B 2003 An Optimal and Progressive Algorithm for Skyline Queries *ACM SIgMOD* 467-478

[13] Tao Y, Ding L, Lin X, Pei J 2009 Distance-based Representative Skyline *Proc. ICDE* 892-903

[14] Berg de M, Dreveld van M, Overmars M, and 1997 Schwarzkop O *Computational Geometry: Algorithms and Applications* Springer-Verlag

[15] Kriegel S and H 1998 Optimal multi-step k-nearest neighbor search. *Proc. 1998 ACM SIGMOD International Conference on Management of* Data (Seattle, USA) 154-65

[16] Kanungu T, Mount D, Netanyahu N, Piatko C, Silverman R, Wu A 1999 Computing nearest neighbors for moving points and applications to clustering. Proc. of *10th ACM-SIAM Symposium on Discrete Algorithms (Baltimore, Maryland, U. S.)* 931-932.

[17] Song Zh, Roussopoulos N 2001 K-Nearest Neighbor Search for Moving Query Point. *Proc. of the 7th International Symposium on Spatial and Temporal Databases* 79-96

[18] Gaede V, Guenther O 1998 *Multidimensional access metods* ACM Computer Surveys 30

[19] Tao Y, Papadias D, Shen Q 2002 Continuous Nearest Neighbor Search. *Proc Very large Data Bases Conf* 287-98

[20] Hu Haibo, Lee Dik Lun. 2006 Range Nearest-Neighbor Query *IEEE Transactions on knowledge and data engineering* **18**(1) 78-91

[21] Gedik B, Liu L 2008 Protecting location privacy with personalized k-anonymity: Architecture and algorithms *IEEE Transactions on Mobile Computing* **7**(1) 1-18

[22] Xu Jianliang, Du Jing, Hu Haibo 2010 Privacy-Conscious Location-Based Queries in Mobile Environments. IEEE Transactions on parallel and Distributed Systems **21**(3) 1-14

[23] Huang Zhiyong, Jensen Christian S, Lu Hua, Ooi Beng Chin 2006 Skyline Queries Against Mobile Lightweight Devices in MANETs. *Proc. of the 22nd International conference on data engineering* 66

[24] Sharifzadeh Mehdi, Shahabi Cyrus 2006 The Spatial Skyline Queries *Proc. of the 32th VLDB Conference (Seoul, Korea)* 751-62

[25] Zou Lei, Chen Lei, Qzsu Tamer M, and Zhao Dongyan 2010 Dynamic Skyline Queries in Large Graphs *Proc. of 14th International Conference on Database Systems for Advanced applications* 62-78

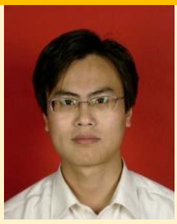
[26] Lin Xin, Xu Jianliang, Hu Haibo 2011 Range-based Skyline Queries in Mobile Environments. *IEEE Transactions on Knowledge and Data Engineering* **23**(11) 103-13

[27] Cheema Muhammad Aamir, Lin Xuemin, Zhang Wenjie, ZhangYing 2013 A Safe Zone Based Approach for Monitoring Moving Skyline Queries *The 16th International Conference on Extending Database Technology (Genoa, Italy)* 275

[28] Guttman A 1984 R-Trees: A Dynamic Index Structure for Spatial Searching *Proc. ACM SIGMOD Int'l Conf. Management of Data* 47-57

[29] Berg de M, Cheong O, Kreveld M, Overmars M 2008 *Computational Geometry: Algorithms and Applications* Heidelberg: Springer-Verlag

Authors



Jing Nie, 1981.12, Xiamen City, Fujian Province, P.R. China

Current position, grades: Director of Department of Software Engineering, Xiamen Institute of Software Technology, P.R. China.
University studies: Graduated from Fuzhou University, China in 2013, received a Master's degree in Business Administration.
Scientific interest: Augmented Reality, Mobile Application Development
Publications: more than 10 papers published in various journals.
Experience: Graduated from Fuzhou University,received a Master's degree in Business Administration ;has completed three scientific research projects; more than 10 papers published in various journals.

Wei Weng, 1979.10, Xiamen City, Fujian Province, P.R. China

Current position, grades: Lecturer of Department of Software Engineering, Xiamen Institute of Software Technology, P.R. China.
University studies: Graduated from Xiamen University, China in 2012, received a Master's degree in Computer Application.
Scientific interest: Skyline Computing, Mobile Application Development, Community Mining.
Publications: more than 20 papers published in various journals.
Experience: Graduated from Xiamen University, received a Master's degree in Computer Application ; has completed five scientific research projects; more than 20 papers published in various journals.

Li'nan Sun, 1993.12,Xiamen City, Fujian Province, P.R. China

Current position, grades: Student of Xiamen University of Technology
Scientific interest: Android Development, Data Mining
Experience: Studying in Xiamen University of Technology; join in a scientific research projects;