Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

Hierarchical Pareto Curve Model for Privacy Skyline

Boris Chan & Jacob Sun
R&D Department
HIS technologies limited
Hong Kong Science Park, Hong Kong
boris@histechs.com & Jacob@histechs.com

Vincent Ng
Dept. of Computing
Hong Kong Polytechnic University
Hunghom, Hong Kong
cstyng@comp.polyu.edu.hk

***Abstract*—Privacy is an essential issue in database publishing. Since the introduction of skyline operator in database community, there was a few researches working on the privacy skyline and related the privacy theory, framework and model in last few years. For those algorithms (e.g. SkylineCheck and Privacy Diagnostics), centralized database is assumed and the consideration of concurrency and parallelism is in lack. In this paper, we propose the Hierarchical Pareto Curve (HPC) model for private skyline processing. In HPC, answers to the skyline query are interpolated by Spline function and represented by a set of polynomial Pareto curves. Hence, skyline querying requests can be satisfied without i disclosing the actual data points. Moreover, the accuracy of a skyline query can be controlled by setting the order of the polynomial expression and total number of Pareto curves. The HPC model can be extended for distributed and cooperative computing environments. With privacy embedded in piecewise Pareto curves and merging operators developed, distributed skyline processing becomes practical. From our preliminary experiments, the results show supportive indications towards the HPC model.**

***Keywords*—Privacy Skyline, Pareto curve, Approximation**

I. INTRODUCTION

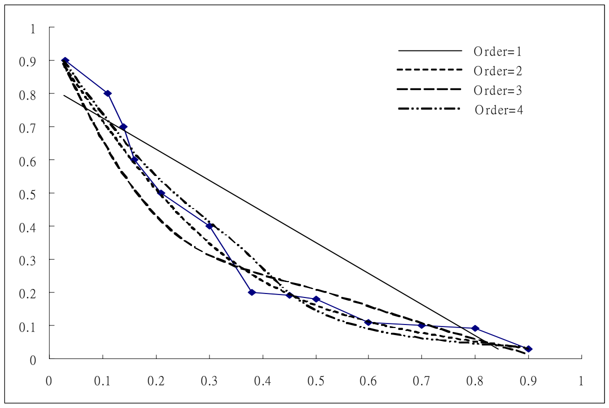

Figure 1. Privacy Skyline and Pareto curves approximation

Privacy is an essential issue in database publishing. In recent years, there were few research works on the privacy skyline [18, 19, 11, 13] and related the privacy theory, framework and model. For algorithms such as SkylineCheck in [19] and Privacy Diagnostics in [13], centralized database is assumed and the consideration of concurrency and parallelism is not present.

In this paper, we propose the Hierarchical Pareto Curve (HPC) model for privacy skyline processing. Given d-dimensional database (D_{global}), a skyline query (*sky*) returns a set of objects $\{obj_1, obj_2, obj_3, ..., obj_n\}$ which are not dominated by any other objects in D_{global}. For the case of 2-dimensional data set (i.e. $obj_1 = (x_1, y_1)$), *sky* would be a set $\{obj_1 = (x_1, y_1), obj_2 = (x_2, y_2), obj_3 = (x_3, y_3), ..., obj_n = (x_n, y_n)\}$. Here, *sky* can be in an arbitrary shape (Figure 1) of discrete data points which can be modeled approximately by Pareto curve as follows:

$$f(x) = c_0 + c_1 x + c_2 x^2 + c_3 x^3 + ... + c_n x^n$$

where c_n is the coefficient of x^n, $n = \{1, 2, ...\}$ and c_0 is the constant.

With several pieces of polynomials (Spline), it can yield a good "fitting to the original *skyline* (see Figure 2). Moreover, the accuracy (mean square error) of the Pareto curves can be controlled by the order of the polynomial expression and total number of Pareto curves.

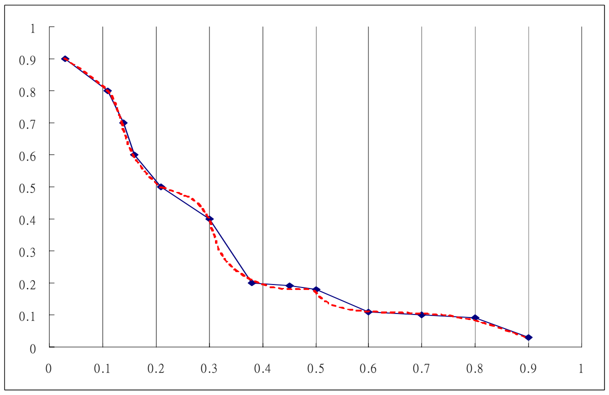

Figure 2. Piecewise Pareto curves approximation

In distributed and cooperative computing environments, we assume that database is partially replicated and allocated to various service providers (servers). In order to answer a skyline query, the trivial and traditional process is to transfer all required datasets of the distributed to a centralized site (such as agent) for computing the global *skyline*. In HPC model, servers with different subspace data can compute its own *sub-skyline* separately. All the Pareto curves of *sub-skyline* at servers are sent to a centralized site (agent). Then, the global *skyline* in the form of Pareto curves is formed by defined operations in HPC model and be able to answer the requested skyline query. The interesting point here it is not necessary to disclose any actual data point of distributed servers while a requesting agent can obtain the global *skyline*. We will report the related works in next section followed by full descriptions of HPC model in section 3. In section 4, the evaluation on the HPC model will be reported with preliminary experiments result discussion. We will make the conclusion and discussion about the possible future works in section 5 and 6 respectively.

II. RELATED WORKS

In [5], Borzsonyi et al first introduced skyline operators in the database community. They proposed a solution based on Block Nested Loop (BNL) and Divide-and-Conquer (B&C) for searching the skylines. In BNL, all objects in database will be compared with one another for eliminating those dominated objects. The final skyline is resulted of a computational expensive elimination process. The B&C approach for searching skyline by dividing objects into pieces which can fit to memory size. Hence the computation of the partial skyline according to the pieces of objects can be more efficient due to in-memory processing. The final skyline is resulted by merging those partial skylines. In [8], a variant of BNL, sort-filter-skyline (SFS) was proposed by Chomicki et al. Its basic idea is to sort objects with respect to monotonic scoring function. Thus, the resulting skyline is computed by scanning those sorted objects. Later, Tan et al improved the SFS algorithm by bitmap and index based algorithm [8].

The first progressive technique for computing skyline was proposed in [10]. It enables a skyline to be computed without scanning the whole database. Kossmann et al [6] presented another progressive algorithm based on the Nearest Neighbor searching technique. Papadias et al [7], proposed a Branch-and-Bound Skyline (BSS) algorithm which is proven to be minimized in I/O cost. Previous discussed works usually focus on enforcement of bound of data published. while privacy of skyline data is not considered.

Privacy of data modeled as query-view privacy has been studied extensively [13, 14, 15, 16, 17]. Given a set of public views of a database, the goal is to check if they reveal any information about private view of the same database, where views are defined by conjunctive queries. Views can be used to express adversarial knowledge. However, rigid definition of privacy usually requires sensitive information to be completely independent of the released data [13, 14, 16]. It does not provide flexibility to tradeoff privacy for utility. Dalvi et al. relaxed the strong requirement but described domain size goes to infinity [13]. Checking query-view safety in the general setting is NP-hard [15].

The k-Anonymity and l-diversity were proposed in [11, 3] for privacy criteria that attempt to capture adversarial knowledge in a less formal way. k-Anonymity requires that no individual be identified from a group of k individuals. Meanwhile l-Diversity requires that each QI-group (Quasi-Identifier) contains at least l "well-represented" sensitive values. Furthermore, V Chen et al incorporated the privacy criteria in [17] and proposed a multi-dimensional privacy criterion with more intuitive and flexible. The algorithm for measuring disclosure and sanitizing data improved the computational efficiency.

In the previous research works related to privacy, centralized database was assumed. Also, performance of query processing was the focus in major skyline research works while privacy enforcement during computation ignored. Thus, we believe our proposed HPC model is a novel one to handle privacy in skyline querying..

III. HIERARCHICAL PARETO CURVE (HPC) MODEL

In the HPC model, given a d-dimensional database (D_{global}), a skyline query (skyline), it will return a set of objects $\{obj_1, obj_2, obj_3, ..., obj_n\}$ which are not dominated by one another objects in D . For the case of two-dimensional data (i.e. $obj_1 = (x_1, y_1)$), a skyline is a set of objects as:

$$\{obj_1 = (x_1, y_1), obj_2 = (x_2, y_2), obj_3 = (x_3, y_3), ..., obj_n = (x_n, y_n)\}$$
.

A *skyline* can be in an arbitrary shape depicted in Figure 1a. Technically, a *skyline* of discrete data points can be modeled by a Pareto curve expressed below.

$$f(x) = c_0 + c_1 x + c_2 x^2 + c_3 x^3 + ... + c_n x^n \qquad [1]$$

where c_n is the coefficient of x^n , $n = \{1,2,...\}$ and c_0 is the constant.

Further more, a Pareto curve can be generalized and expressed as:

$$f(x) = \sum_{i=0}^{n} c_i \cdot x^i \qquad [2]$$

By extending equation [2] for multiple Pareto curves, we have :

$$\begin{bmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix} = \begin{bmatrix} x_0^n & x_0^{n-1} & \dots & x_0^0 \\ x_1^n & x_1^{n-1} & & x_1^0 \\ \vdots & & & \vdots \\ x_n^n & x_n^{n-1} & \dots & x_n^0 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix} \qquad [3]$$

Also, we can compute multiple Pareto curves with different orders. For example, we have computed four of Pareto curves with order of 1~4 respectively and resulted in graphical presentation (Figure 2).

*Mean Square Error (*mse*)*

It is obvious that single polynomial with high order results in better "fitness" to the actual data points. In fact, the "fitness" of each curve can be measured by the corresponding Mean Square Error (*mse*).

$$mse(f(x)) = \sum_{s=1}^{S} (\hat{x} - x_s)^2 \text{ for } s = 1,2,... \qquad [5]$$

where $\hat{x}$ is the estimator, x_s is the estimated value, S is the sample size.

We can consider *Spline Interpolation* is a form of interpolation where the interpolant is a special type of piecewise Pareto curve called *Spline*. With *Spline Interpolation*, piecewise Pareto curves are defined as:

$$f(x) = \begin{cases} f_1(x) & \forall x \in D_1 \mid x_0 \le x < x_1 \\ f_2(x) & \forall x \in D_2 \mid x_1 \le x < x_2 \\ \vdots & \\ f_R(x) & \forall x \in D_R \mid x_{R-1} \le x < x_R \end{cases}$$

and can be represented as:

$$f(x) = \sum_{R=1}^{R} f_R(x) \quad \forall x \in D_R \mid x_{R-1} \le x < x_R \qquad [6]$$

where there is no intersection among $D_0, D_1, \ldots D_R$.

The *Spline Interpolation* is employed to represent the skyline result set for answering skyline query. With this approach, there are two immediate properties. First, each piecewise *Spline* can interpolate (passing through) all the skyline points for a data subspace. Second, the attributes of a piecewise *Spline* can be used to represent the corresponding part of a skyline without revealing the actual data points so as to support data privacy. One other benefit is the reduction of data transfer if Spline attributes are required instead of the original data. For this approach, we called the computed result as a *privacy skyline*.

Manipulation in HPC Model

In distributed and cooperative computing environments, we assume that database is partially replicated and allocated to various service providers (servers). In other words, every single server contains subset of global database and results in *sub-skyline* of the global *skyline* if it carries out a local skyline query processing.

For a 2-dimensional skyline query, if we have two servers working on the skyline query, we may have two curves, c_1 and c_2,with respective intervals as $a \le x \le b$ and $n \le x \le m$. It is observed that there are 3 relationships between the two as shown in Figure 3.

In case 1: curves c_1 and c_2 are represented by the following functions:

$$f_1(x) = \sum_{i=0}^{n} c_i \cdot x^i \text{ for } lb_1 \le x \le ub_1 \text{ and} \qquad [8]$$

$$f_2(x) = \sum_{i=0}^{n} c_i \cdot x^i \text{ for } lb_2 \le x \le ub_2 \qquad [9]$$

where lb_1 and ub_1 are the lower-bound and upper-bound of c_1, and lb_2 and ub_2 are the lower-bound and upper-bound of c_2 respectively. If $ub_1 < lb_2$ or $lb_1 > ub_2$, two curves are separated.

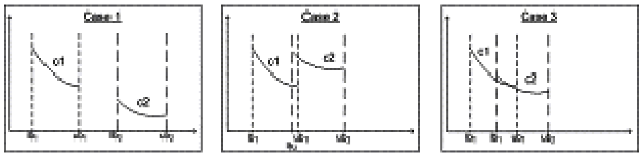

Figure 3. Relationship of piecewise Pareto curves

In case 2, assume two curves c_1 and c_2 (same as above) and find that they have no cross point if $f_1(x) - f_2(x) = 0$ is unresolved. They are overlapping at by calculating $lb_2 - ub_1 > 0$ or $ub_1 - lb_2 > 0$. If $lb_2 - ub_1 > 0$, the corresponding region of c_2 will be eliminated. Otherwise if $ub_1 - lb_2 > 0$, the corresponding region of c_1 will be eliminated.

For case 3, assume two curves c_1 and c_2 (same as above) and find that they are overlapping at $x = x'$ by calculating $f_1(x) - f_2(x) = 0$. If $f_1(a) < f_2(a)$ where $lb_2 < a < x'$, c_1 is dominating c_2 in the region of $lb_2 \le x \le x'$ and c_2 is dominating c_1 in the region of $x' \le x \le ub_1$. In order to eliminate the ambiguity, both curves (c_1 & c_2) have to be shortened according to the cross point at $x = x'$.

$$f_1(x) = \sum_{i=0}^{n} c_i \cdot x^i \text{ for } lb_1 \le x \le x' \qquad [10]$$

$$f_2(x) = \sum_{i=0}^{n} c_i \cdot x^i \text{ for } x' \le x \le ub_2 \qquad [11]$$

As a result, we the following operations in HPC model for handling pair-wise Pareto curves combination. One can easily apply the operations recursively for multiple Pareto curves combinations.

1. $f_i(x)$ co-exists with $f_j(x)$ and the MSE will be $mse(f_i(x)) + mse(f_j(x))$ [12a] it means that both $f_i(x)$ and $f_j(x)$ are in the set of multiple Pareto curves.
2. either sub-region of $f_i(x)$ or sub-region of $f_j(x)$ is eliminated and the MSE will be either

$$mse(f_i(x)) + mse(f_j(x)) \cdot \frac{(ub_1 - ub_2)}{ub_2 - lb_2} \text{ or}$$

$$mse(f_i(x)) \cdot \frac{(lb_2 - ub_1)}{ub_1 - lb_1} + mse(f_j(x)) \qquad [12b]$$

3. D_i and D_j of $f_i(x)$ and $f_j(x)$ respectively needs redefined as equations [10] and [11], and the MSE will be

$$mse(f_i(x)) \cdot \frac{(ub_1 - x')}{ub_1 - lb_1} + mse(f_j(x)) \cdot \frac{(x' - lb_2)}{ub_2 - lb_2} \qquad [12c]$$

Generic Model for Multi-dimension Skyline

For distributed processing of ***K***-dimensional skyline, we employ $f(d_1, d_2, \ldots d_K)$ to represent. According to the equation [6], we have:

$$f(d_1) = \sum_{R=1}^{R} f_R(d_1) \quad \forall d_1 \in D_{1,R} \mid d_{1,R-1} \le d_1 < d_{1,R}$$

$$f(d_2) = \sum_{R=1}^{R} f_R(d_2) \quad \forall d_2 \in D_{2,R} \mid d_{2,R-1} \le d_2 < d_{2,R}$$

Therefore, we have

$$f(d_1, d_2, \ldots d_k) = \begin{cases} \sum_{R=1}^{R} f_R(d_1) \quad \forall d_1 \in D_{1,R} \mid d_{1,R-1} \le d_1 < d_{1,R} \\ \vdots \\ \sum_{R=1}^{R} f_R(d_k) \quad \forall d_k \in D_{k,R} \mid d_{k,R-1} \le d_k < d_{k,R} \end{cases}$$

with equation [2], we have

$$f(d_1, d_2, \ldots d_k) = \begin{cases} \sum_{R=1}^{R} \sum_{i=0}^{n} c_{1,i} \cdot d_1^i \quad \forall d_1 \in D_{1,R} \mid d_{1,R-1} \le d_1 < d_{1,R} \\ \vdots \\ \sum_{R=1}^{R} \sum_{i=0}^{n} c_{k,i} \cdot d_k^i \quad \forall d_k \in D_{k,R} \mid d_{k,R-1} \le d_k < d_{k,R} \end{cases} \qquad [13]$$

and the MSE is expressed as:

$$mse(f(d_1, d_2, \ldots d_k)) = \sum_{i=1}^{sizeof(D)} (\hat{d}_i - d_i)^2 \qquad [14]$$

It is noted that the MSE is the resulting measurement. The controlling parameters for the MSE are the order of the polynomial (*I*) and the number of piecewise polynomials (*R*). As mentioned before, *Spline* can interpolate (passing through) all the data points using multiple piecewise polynomials. Therefore, the interpolation can be free of *mse* (when *R* = *sizeof(D)*).

The distributed skyline processing algorithm will be:

1. client send skyline query to agents;
2. agents forward the skyline query to connected servers;
3. each server computes its own sub-skyline according to its subset of database and discards duplicated query;
4. by equations [5] & [6], server can provide the Pareto curves representing its sub-skyline and reply the agents;
5. by equations [12a], [12b] & [12c], agent can perform the Pareto curves combinations and result in equation [13] and reply to the client.

Figure 4. Distributed Skyline Processing Algorithm

IV. Evaluations of HPC Model

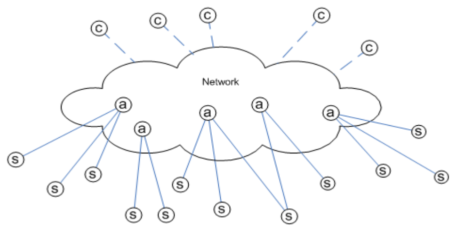

Figure 5. Generic Distributed Computing Environments

It is assumed that for a generic distributed computing environments (Figure 4a), clients (c) are loosely coupling to the network. Clients may look for and address to the service provider (or simply referred as server, s in Figure 4a) for service through service agents (a in Figure 4a). Moreover, it is assumed that database objects are partially duplicated and allocated to servers. We further assume that total number of objects in database is 10K, 50K, & 100K. We employ distribution of exponential function to generate the discrete occurrence of objects along the skyline due to its nature of continuous probability distributions.

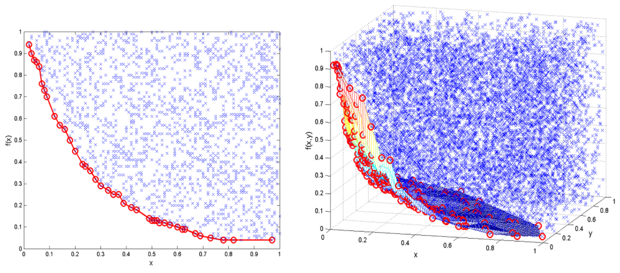

Figure 6. 2D & 3D Synthetic Skyline

In experiment I, we aimed at studying the “fitness” of HPC for modeling skyline. Thus, we synthesized 1000 of skylines from 2-dimensional data space and carried out the approximations by 1~5 pieces of piecewise polynomial to the order of 1~5 according to the equation 12. Correspondingly, the Mean Square Error (*mse*) will be measured according to the equation 13 and recorded during the simulation.

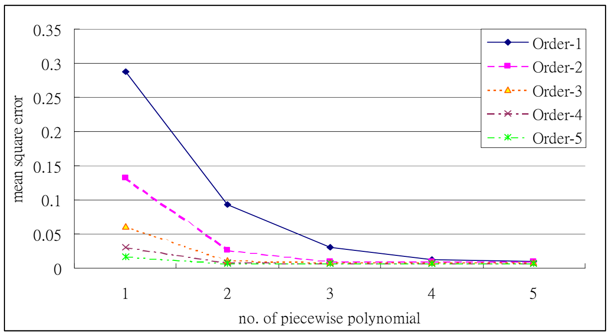

Figure 7. Results of Experiment I

From the result of experiment I (Figure 6), we observed that the mse is highest with "Order-1" of the polynomial when focus is on the no. of piecewise polynomial was 1. And the mse was reducing with order increasing. In parallel, the higher the no. of piecewise polynomial, the lower the *mse*. This observation aligned with the theoretical analysis in previous section.

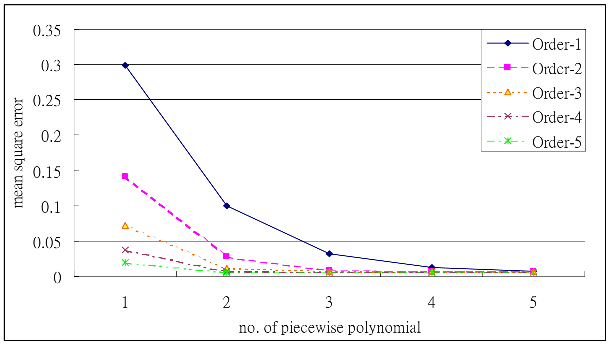

Figure 8. Results of Experiment II

In Experiment II, we repeated the experiment I with skylines synthesized from 4-dimensional data space. We observed from the results (Figure 7) when the results had no significant difference when compared with the results from Experiment I. It was mainly due to the independence of the attributes in skyline application.

In Experiment III, we aimed at studying the performance of traditional approach and HPC approach for the skyline processing in distributed computing environments.

Procedure for processing the skyline query by conventional approach in distributed computing environments is as follows:

1. client initiates skyline query and sends to service agents (agents);
2. each agent forwards the skyline query to associated service providers (servers);
3. each server sends its subset of database to the requesting agent;
4. agent unions the subset of database from servers;
5. agents communicates with one another for constructing the global database;
6. super agent (one of the agents) holds the global database and results in the global skyline;
7. super agent sends the global skyline to requesting client;

It is noted that the global skyline is in the form of actual data points. There is "zero" privacy during the skyline query processing between servers, agents and requesting client.

Procedure for processing the skyline query by HPC model in distributed computing environments is as follows:

1. client initiates skyline query and sends to service agents (agents);
2. each agent forwards the skyline query to associated service providers (servers);
3. each server processes the skyline query according to its subset of database and result in the corresponding sub-skyline;
4. server approximates the sub-skyline by Pareto curve (by equation 13) and records the *mse* (by equation 14);
5. server sends the resulting polynomial(s) & *mse* to agent;
6. agent carries out the curves resolution (by equation 12a, 12b & 12c);
7. agent sends the final polynomial(s) & *mse* to client;

It is noted that there is computation overhead for merging curves by equation 13.

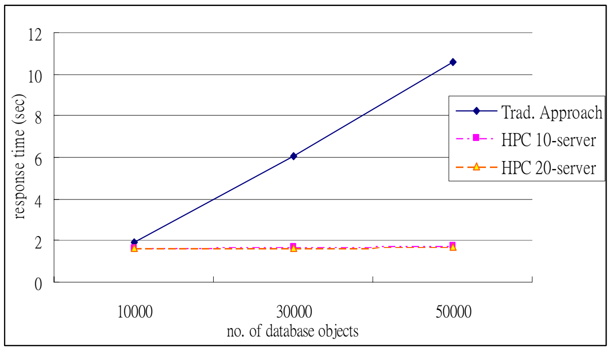

Figure 9. Results of Experiment III

We observed from the experiment III that when the database size (number of database object) increased, the performance (in terms of response time) became poor. Meanwhile, the performance of skyline processing by the HPC model with 10 servers and 20 servers were relatively static when the database size increased. The poor performance of traditional approach was bottlenecked at constructing the global database for computing the global skyline. In the contrary, HPC model enabled distributed sub-skyline processing. Although there was computation overhead for merging curves (by equation 13), it could take advantage of distributing skyline computation among servers.

V. Conclusion and Fugure Works

We have presented the Hierarchical Pareto Curve (HPC) model for the privacy skyline processing and reported the definitions and manipulations of the HPC model in distributed and cooperative computing environments. Technically, we have demonstrated Pareto curve as an answer to skyline query instead of actual data points. According to the theory of geometrical approximation, the formal deviation from the actual points can be captured as Mean Square Error (*mse*). From the preliminary experimental results, we observed that the performance of HPC model for skyline processing out performance the traditional approach in distributed and cooperative computing environments.

Further our studies, we may explore more along multiple path of communications in the distributed & cooperative computing environment. For instance, the servers (mentioned in section 4) may be communicating and cooperating among peer servers for exchange of *sub-skyline* without relaxing the privacy. In the same manner, agents can communicate among agents for better system performance. Moreover, the proposed HPC model has been demonstrated with read-only skyline processing so far. We would like to explore more regarding skyline processing with read-only as well as updating accesses. Furthermore, transaction management of skyline processing such as concurrency control protocol with different degree of privacy deserves serious attention in database research community.

References

[1] E. Dellis, A. Vlachou, I. Vladimirskiy, B. Seeger, and Y. Theodoridis, Constrained Subspace Skyline Computation, In CIKM, 2006, page 415-424.

[2] J. Pei, W. Jin, M. Ester, and Y. Tao, Catching the Best Views of Skyline: A Semantic Approach Based on Decisive Subspaces, In VLDB, 2005, page 253-264.

[3] Y. Tao, X. Xiao, and J. Pei, SUBSKY: Efficient Computation of Skylines in Subspaces, In ICDE, 2006, page 65-75.

[4] Y. Yuan, X. Lin, Q. Liu, W. Wang, J.X. Yu, and Q. Zhang, Efficient Computation of Skyline Cube, In VLDB, 2005, page 241-252.

[5] S. Borzsonyi, D. Kossmann, and K. Stocker, The Skyline Operator, In ICDE, 2001, page 421-430.

[6] Donald Kossmann, Frank Ramsak, and Steffen Rost, Shooting stars in the sky: An online algorithm for skyline queries, In VLDB, 2002, page 275-286.

[7] D. Papadias, Y. Tao, G. Fu, and B. Seeger, An optimal and progressive algorithm for skyline queries, In SIGMOD, 2003, page 467-478.

[8] Jan Chomicki, Parke Godfrey, Jarek Gryz, and Dongming Liang, Skyline with presorting, In ICDE, 2003, page 717- 719.

[9] Parke Godfrey, Ryan Shipley, Jarek Gryz1, Maximal Vector Computation in large Data Sets, In VLDB, 2005, page 229-240.

[10] K. Tan, Efficient Progressive Skyline Computation, In VLDB, 2001. page 301-310

[11] Latanya Sweeney, k-anonymity: a model for protecting privacy, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems.vol 10, no 5, page 557-570

[12] Ashwin Machanavajjhala , Johannes Gehrke , Daniel Kifer , Muthuramakrishnan Venkitasubramaniam, l-Diversity: Privacy Beyond k-Anonymity, Proceedings of the 22nd International Conference on Data Engineering, v.10 n.5, page.557-570, October 2002.

[13] Dalvi, N., Miklau, G., and Suciu, D., Asymptotic conditional probabilities for conjunctive query, In ICDT, 2005, page 289-305.

[14] Deutsch, A., Papakonstantinou, Y., Privacy in database publishing., In ICDT, 2005. page 230-245

[15] Ashwin Machanavajjhala , Johannes Gehrke, On the efficiency of checking perfect privacy, Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, June 26-28, 2006, Chicago, IL, USA. page 163-172

[16] Gerome Miklau , Dan Suciu, A formal analysis of information disclosure in data exchange, Proceedings of the 2004 ACM SIGMOD international conference on Management of data, June 13-18, 2004, Paris, France. page 507-534

[17] Bee-Chung Chen , Kristen LeFevre , Raghu Ramakrishnan, Privacy skyline: privacy with multidimensional adversarial knowledge, Proceedings of the 33rd international conference on Very large data bases, September 23-27, 2007, Vienna, Austria. page 770-781

[18] Jiexing Li , Yufei Tao , Xiaokui Xiao, Preservation of proximity privacy in publishing numerical sensitive data, Proceedings of the 2008 ACM SIGMOD international conference on Management of data, June 09-12, 2008, Vancouver, Canada page 473-486

[19] Yao, C., Wang, X.S., Jajodia, S. Checking for k-anonymity violation by views. VLDB, 2005. page 910-921