KANIS: Preserving k-Anonymity Over Distributed Data

Katerina Doka, Dimitrios Tsoumakos, Nectarios Koziris
Computing Systems Laboratory, School of Electrical and Computer Engineering
National Technical University of Athens &
Institute for the Management of Information Systems
Research Center Athena, Athens, Greece
{katerina, dtsouma, nkoziris}@cslab.ntua.gr

ABSTRACT

In this paper we describe *KANIS*, a distributed system designed to preserve the privacy of multidimensional, hierarchical data that are dispersed over a network. While allowing for efficient storing, indexing and querying of the data, our system employs an adaptive scheme that automatically adjusts the level of indexing according to the privacy constrains: Efficient roll-up and drill-down operations take place in order to guarantee k-anonymity while minimizing data distortion and inconsistency. Thus, our system manages to maintain k-anonymity of the published data in a distributed and on-line manner even under frequent updates, without affecting its ability to efficiently answer queries. The initial experimental evaluation of our prototype shows that *KANIS* manages to preserve k-anonymity while improving the data quality up to 22% compared to a popular centralized global recoding algorithm. It achieves a near-optimal distortion regardless of the network or dataset size, with a reasonable communication overhead, scattered among the participating nodes.

1. INTRODUCTION

The advent of Web 2.0 with its new, more democratized model gave individuals the ability to equally become content producers as well as consumers. This fact drastically changed the patterns of interaction between people and businesses, leading to an ever increasing demand for online processing of immense volumes of data (blogs, wikis, social networks, social bookmarking, news aggregation sites, etc.). The tools that make up the Web are abundant and constantly evolving and recombining.

An example of such evolution is given by personalized web services: It is estimated that over 80% of users prefer the numerous personalization services that businesses and social sites offer [3]. This entails the gathering of considerable amounts of sensitive information, which often raises serious privacy issues. For instance, the release of the AOL query logs in 2006 led to the tracing of anonymous users to their real names [2]. Location-based services [4, 15] offer a wide variety of information based on the clients' locations but also feature significant abuse possibilities [14].

k-Anonymity has been proposed as an approach to protect personalized data privacy against such attacks [8]. Its goal is to ensure that individuals are unidentifiable in released data by making sure each value of a sensitive subset of data attributes called *quasi-identifier attributes* (or just *QID*) appears at least k times [18]. The most common way to produce k identical tuples is to generalize values within the attributes, e.g., by dropping the least significant digit from the Zip code domain. At the same time, the utility of the published data should remain as high as possible.

Various approaches for generalization dictate the mapping of a set of attribute values to another set of values that belong to a more general domain. This mapping can be done either globally, by mapping the whole domain to a more general one (*global recoding*) [10, 11] or locally, by mapping each tuple individually to a generalized one (*local recoding*) [13, 19]. More recent works use attribute hierarchies in order to achieve k-anonymity with the less possible information loss by "climbing up" in the domain hierarchy [13].

Yet, the way that k-anonymity has been implemented so far refers to a centralized storage and processing server that is responsible of gathering all sensitive data, anonymizing it and distributing it to the numerous users. Some works [9, 20] attempting to propose distributed k-anonymity algorithms do not deal with data horizontally partitioned over multiple network nodes. As more systems and applications opt for data distribution, we believe that their efficient anonymization is of great importance to offer customized privacy according to the needs of the applications that access them.

In this paper, we investigate the problem of continuously preserving the anonymity of fully distributed data in a way that minimizes the data distortion and the communication overhead. To that end, we propose *KANIS*[1] (K-ANonymity Indexing System), an always-on DHT-based system, that guarantees real-time k-anonymization during updates. The system comprises of multiple cooperating nodes that share and serve multidimensional, hierarchical data. Individual nodes actively monitor the privacy of the data they are responsible for, in order to adjust the indexing level to the one that guarantees k-anonymity after the insertion of new tuples. Furthermore, the system does not invalidate the semantics of the stored hierarchies and allows for distributed knowledge mining. To our knowledge, this is the first attempt towards the support of distributed k-anonymity in

[1]Kanis (nobody in Greek) is how Ulysses cleverly introduced himself to cyclop Polyphemus, who had captured him and his crew. After being blinded by Ulysses, Polyphemus yelled to his fellow cyclops that "nobody" had hurt him.

PersDB 2011 Workshop, September 2, 2011, Seattle, WA, USA

DHTs. Yet, while the main focus of this work is k-anonymity, our system can be extended to support other privacy principles (e.g., l-diversity, t-closeness, m-invariance etc.).

1.1 Definitions

The goal of k-anonymization is to make every tuple of a published table identical to at least $k-1$ other tuples with respect to a set of attributes. As a motivating example, let us assume a table of patient's data (Table 1(a)). Record No. 3 is unique with respect to the attribute set {Gender, Age, Postcode}, hence the medical problem of this patient may be revealed if the table is published. To preserve his privacy, we may generalize the Postcode attribute values such that each tuple has at least two occurrences. Assuming each domain is analyzed in the hierarchies of Figure 1, we can achieve 2-anonymity by climbing up one level in the Postcode hierarchy (see Table 1(b)).

Definition 1 (Quasi-Identifier Attribute Set). *A quasi-identifier attribute set (*QID*) is a minimal set of attributes in a table that can be joined with external information to potentially identify individual records.*

The *QID* sets are selected by experts based on the specific knowledge of the domain they refer to.

Definition 2 (Equivalence Class). *An equivalence class (EC) of a table with respect to an attribute set is the set of all tuples that contain identical values for the attribute set.*

Definition 3 (Frequency Set). *The frequency set of a table with respect to an attribute set is a mapping from each EC to the total number of tuples (counts) that belong to it.*

Definition 4 (K-Anonymity). *A table satisfies k-anonymity with respect to a quasi-identifier set if its frequency set contains counts greater than or equal to k.*

`Example:` For our motivating example, the *QID* set is {Gender, Age, Postcode}. Tuples 1 and 2 from Table 1(a) form an EC with respect to the *QID*, with frequency count equal to 2. k-anonymity requires that every tuple occurrence for a given *QID* set has a frequency of at least k. For example, Table 1(a) does not satisfy 2-anonymity since tuple No.3 occurs only once.

There exist various metrics to evaluate the quality of a k-anonymous dataset. A general criterion should be the *distortion* of a table. Since we are dealing with hierarchical structures, in this paper we consider distortion as defined in [12], based on the *weighted hierarchical distance (WHD)* metric. Assuming each hierarchy level has a weight, the WHD between two levels is the fraction of the weights of the levels between them to the sum of all levels' weights. According to the definition of [12], the distortion of a generalized tuple is the sum of the WHD values of all attributes belonging to its *QID* set and the distortion of a table is the sum of the distortions of all tuples belonging to that table.

1.2 Necessary Notation

Our data spawn the d-dimensional space. Each dimension i is organized along L_i hierarchy levels: $H_{i1}, H_{i2}, \ldots, H_{iL_i}$, with H_{i1} being the special *ALL (∗)* value. We assume that our database comprises of fact-table tuples of the form: $\langle tupleID, D_{11} \ldots D_{1L_1}, \ldots, D_{d1} \ldots D_{dL_d}, fact_1, \ldots, fact_k \rangle$, where $D_{ij}, 1 \leq i \leq d$ and $1 \leq j \leq L_i$ is the value of the j^{th} level of the i^{th} dimension of this tuple and $fact_l, 0 \leq l \leq k$ are the numerical facts that correspond to it (we assume that the numeric values correspond to the most detailed level of the hierarchies). Our goal is to insert, index and update this data so that it constantly remains k-anonymous, for values of k according to the applications' requirements.

Definition 5 (Level Ordering). *We define that a hierarchy level $H_{ix}, 1 \leq x \leq L_i$ lies above (below) $H_{iy}, 1 \leq y \leq L_i$ and denote it as $H_{ix} < H_{iy}$ ($H_{ix} > H_{iy}$)* iff *$x \leq y$ ($x \geq y$), i.e., if H_{ix} corresponds to a less (more) detailed level than H_{iy}.*

Definition 6 (Level Combination Ordering). *A level combination $C = \langle c_1, c_2, \ldots, c_d \rangle$, where each element c_i can be a valid hierarchy level of the i^{th} dimension (including the special $*$ value): $c_i = H_{iy}, 1 \leq y \leq L_i$, lies above (below) a level combination $C' = \langle c'_1, c'_2, \ldots, c'_d \rangle$, denoted $C \prec C'$ ($C \succ C'$)* iff *$c_i < c'_i$ ($c_i > c'_i$), $\forall 1 \leq i \leq d$.*

`Example:` For the data of our motivating example, `city`$<$`suburb`, $\langle$`gender`, `interval`, `city`$\rangle \prec \langle$`gender`, `value`, `suburb`$\rangle$, while $\langle$`gender`, `interval`, `city`$\rangle \succ \langle *$, `interval`, `state`$\rangle$.

Property 1. *If a table T satisfies k-anonymity with respect to a level combination C, then it satisfies k-anonymity $\forall C'$, where $C' \prec C$.*

Property 2. *If a table T does not satisfy k-anonymity with respect to a level combination C, nor does it satisfy k-anonymity $\forall C'$, where $C' \succ C$.*

`Example:` The data of our motivating example is 2-anonymous with respect to $\langle$`gender`, `interval`, `city`$\rangle$. Therefore, the data is also 2-anonymous with respect to $\langle *$, `interval`, `state`$\rangle$. On the contrary, since $\langle$`gender`, `value`, `city`$\rangle$ does not ensure 2-anonymity, nor does $\langle$`gender`, `value`, `suburb`$\rangle$.

1.3 Assumptions

We assume that there exist private channels between the nodes participating in the system. By exchanging messages through these channels, the system nodes attempt to jointly k-anonymize their data. Moreover, we do not consider the existence of malicious nodes among the participating ones: Peers follow the protocol without trying to derive extra information in order to violate privacy.

2. THE SYSTEM

KANIS is a system that efficiently preserves the k-anonymity property for different multidimensional, hierarchy-annotated datasets in the face of continuous updates. The system initially chooses a level of hierarchy for each dimension and indexes tuples according to that default level combination, called *pivot* $P = \langle p_1, p_2, \ldots, p_d \rangle$ where each pivot element p_i can be a valid hierarchy level of the i^{th} dimension (including the special $*$ value). Each tuple receives an ID that equals the hashed value of the attribute combination corresponding to P. The DHT then assigns each tuple to the node with ID numerically closest to its ID. Inserted tuples are internally stored in a hierarchy-preserving manner (tree-like form). As data are shared this way among

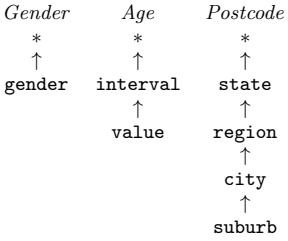

Figure 1: Concept hierarchies for `Gender`, `Age` and `Postcode`.

Table 1: (a) Raw table and (b) its 2-anonymized version through the use of hierarchies

No.	Gender	Age	Postcode	Problem
1	male	middle	4350	Flu
2	male	middle	4350	Ulcer
3	male	middle	4351	Ulcer
4	female	old	4353	Flu
5	female	old	4353	Ulcer

No.	Gender	Age	Postcode	Problem
1	male	middle	435*	Flu
2	male	middle	435*	Ulcer
3	male	middle	435*	Ulcer
4	female	old	435*	Flu
5	female	old	435*	Ulcer

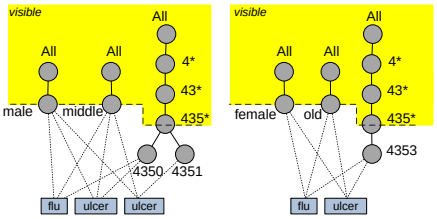

Figure 2: KANIS system for the motivating example

the overlay nodes, coordination is thus required in order to decide a switch to a more or less general level combination.

We note that our system focuses on global recoding, which usually over-generalizes a table, resulting in more information loss than local recoding. However, global recoding assures that all values of an attribute belong to the same domain. This is very important for data mining and statistical analysis, as most such tools assume domain consistency.

2.1 Insertion

Before the data are initially inserted to the system, we assume the fact table undergoes global recoding centrally (e.g., using Incognito [10]) and the appropriate P is selected so that the dataset is k-anonymous. The data are parsed tuple by tuple, hashed according to the selected P and inserted to the corresponding network nodes. Inserted tuples are internally stored in a hierarchy-preserving manner: The data of Table 1(b) would be stored as seen in Figure 2, with P being equal to ⟨`gender`, `interval`, `city`⟩[2]. Since there exist 2 distinct value combinations for pivot, two different trees are created and stored in the corresponding overlay nodes after the insertion process is over. Note that only the values above the pivot level (the yellow area) are available for user queries, in order to ensure k-anonymity.

2.2 Updates

Updates refer to the insertion of new tuples, since for most analytics applications tuples are commonly considered as read-only. Given that P is known to all nodes participating in the overlay, hashing the newly inserted tuple according to it and storing the new item is trivial. However, k-anonymity must be preserved at all times while maintaining data as useful as possible.

On one hand, new tuples might break the k-anonymity constraint of existing data. Suppose that a tuple ⟨`female, middle, 4352, Flu`⟩ is inserted in our k-anonymous distributed system. Since P is equal to ⟨`gender`, `interval`, `city`⟩, a new tree will be created, as shown in Figure 3. However, this new tuple is unique for the *QID* set and thus jeopardizes the privacy of the individual it refers to. In this case, a new global P must be selected in order to generalize the data and ensure k-anonymity.

On the other hand, new tuples that arrive and load the existing trees with new values might result in an over-generalized dataset with high distortion. If tuple ⟨`male, middle, 4351, Flu`⟩ is inserted, we observe that drilling down one level in the Postcode hierarchy preserves k-anonymity and significantly decreases data distortion (see Figure 4).

Both cases require the reindexing of the system's data according to a new P. Our system supports near real-time updates of the distributed data by dynamically adjusting its indexing to the incoming tuples without assuming any prior knowledge, solely relying on locally maintained information. By shifting to a different P we aim at guaranteeing k-anonymity while causing the least possible distortion.

Every time a new tuple or a batch of new tuples is inserted to the system, the receiving nodes check the modified trees. Note here that each tree corresponds to an EC. If the number of tuples belonging to a tree t (the count of the EC), denoted $Count_t$, is less than k, then the k-anonymity constraint is violated and the *rollup* anonymization strategy must be followed. If, on the other hand, $Count_t > 2 \cdot k$ then the *drilldown* anonymization strategy further investigates whether a P with less distortion could be chosen.

Rollup Anonymization During this procedure, the node where the privacy breach has occurred must select an alternative global P in order to ensure k-anonymity. To do so, the node requires information from the rest of the network nodes. To that end, it floods a *CollectStats* message over the network, which contains the values of all hierarchy levels above *pivot*. Upon reception, each node collates these values with each of its trees and calculates the frequency set of all possible ECs that lie above P. This frequency set is returned to the initiator.

With this process we aim to find all possible ECs that can be merged with the non-anonymized one in order to result in an EC with size of at least k. Since the new P will always be a generalization of the old one, the already anonymized ECs will remain anonymized after the reindexing.

After collecting all the node statistics, the initiator chooses among the possible rollup level combinations the one, P_{new}, that will result in an EC of k or more tuples and will cause the minimum distortion, performing the necessary calculations locally. A reindexing operation is then initiated with P_{new} (see section 2.3).

Drilldown Anonymization This procedure is performed in order to check if there exists a level combination that preserves k-anonymity while reducing distortion. It is divided in two phases, the local and the global one. During the local phase, for the specific tree t where $Count_t > 2 \cdot k$ the node

[2]The most detailed level of the *Age* hierarchy is not depicted for simplicity reasons.

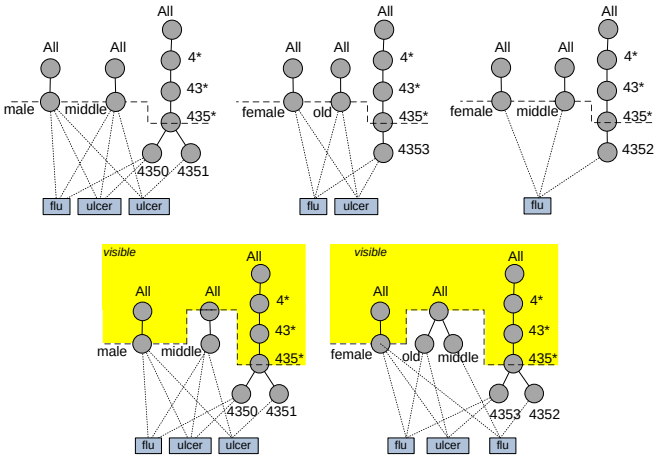

Figure 3: The insertion of ⟨female, middle, 4352, Flu⟩ causes a rollup operation.

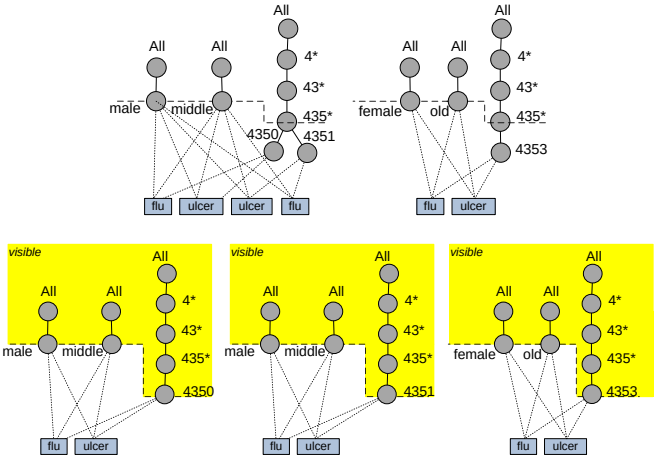

Figure 4: The insertion of ⟨male, middle, 4351, Flu⟩ causes a drilldown operation.

calculates the frequency set for all possible level combinations that lie below P. P_{cand} is the set of level combinations that result in ECs with $Count_t > k$. If the set is empty, the process stops. Otherwise, the global phase begins with P_{cand} being flooded to all network nodes. Upon reception, each node n checks for each level combination of P_{cand} if the resulting ECs are k-anonymous and sends back those that satisfy this constraint ($P_{cand,n}$). After collecting all the answers, the initiator calculates the intersection of the returned sets $\bigcap_{i=1}^{N} P_{cand,n}$ and chooses the level combination P_{new} with the minimum distortion. A reindexing operation is then initiated with P_{new}.

There is an open issue related to the continuous publishing of updated data. Since the same data might be anonymized differently in different releases, the anonymity of an individual may be compromised when cross-examining multiple releases over time. Dealing with such inferences is part of our future work. We just note that generally, depending on the scenario and the assumed knowledge of the attacker, a variety of strategies can be applied [16, 7]. For example, in the case of global recoding, if the attacker has no temporal background knowledge then a k-anonymity breach is avoided.

2.3 Reindexing

The initiating node floods a *Reindex* message to force all nodes to change their *pivot*. Each node that receives this message traverses its tuples, finds all the values of the level combination that will constitute the new reference point and rehashes them one by one, sending the tuples to the corresponding nodes. Assuming that the size of the dataset $|D| \gg N^2$, N being the size of the network, the preferred method to perform this is to send at most $N-1$ messages per node, grouping the tuples by recipient. After the node completes the procedure, it erases all its data.

3. EXPERIMENTAL RESULTS

Our proposed method has been developed using a heavily modified version of the FreePastry simulator [6], although any DHT implementation could be used as a substrate. The default number of nodes used is 16, although experiments have been conducted with up to 128 nodes. Incognito [10], the most popular global recoding method, has been implemented as well for direct comparison. For the initial evaluation of our prototype we measure the quality of our method as well as the cost of the process in terms of time, messages and bandwidth in various settings.

Table 2: Description of the datasets

Adult			APB		
Attribute	Values	Levels	Attribute	Values	Levels
age	74	5			
work class	7	3	customer	900	3
education	16	5			
marital status	7	4	product	9000	7
occupation	14	5			
race	5	3	channel	9	2
sex	2	2			
native country	41	4	time	18	4

For our experiments we utilize the adult dataset of the UCIrvine Machine Learning Repository [5], which has become the de facto benchmark for k-anonymization. Records with unknown values have been eliminated, resulting in a dataset of 45k tuples (denoted as *Adult*). To prove the scalability of our system, we have also utilized datasets produced by the APB-1 benchmark generator [1] (denoted as *APB*), which simulates a realistic business situation. More specifically, using the APB benchmark generator we produce three 4-d datasets A, B and C with densities 0.1, 0.5 and 1 and sizes 1M, 6M and 12M respectively. The characteristics of both datasets are presented in Table 2.

3.1 Varying the Size of the Update Batch

In this set of experiments, *KANIS* and Incognito initially contain the first 5k tuples of the Adult dataset, k-anonymized. Next, we continuously pose batches of updates with sizes varying from 1k to 10k tuples to both of them and record their behavior. For Incognito, we assume that it maintains the initial domain generalizations (the ones that arise from the processing of the initial table of 5k tuples), as long as k-anonymity is preserved, at the risk of potentially keeping the table over-generalized. The quality of both anonymization methods is evaluated using the *distortion* metric, as defined in Section 1.1.

Table 3 presents measurements for various k values, assuming that the *QID* set contains all the available dimensions ($|QID|= 8$). The communication cost of *KANIS* is estimated through the number of reindexings performed throughout the simulations, the number of messages required and

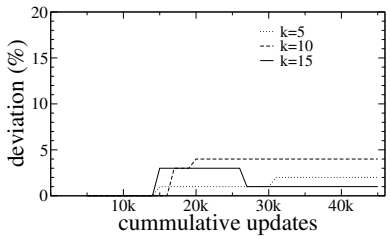

Figure 5: The distortion deviation of *KANIS* as updates arrive in 1k batches

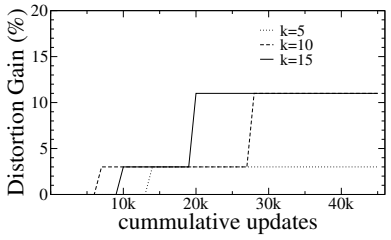

Figure 6: The gain of *KANIS* in distortion compared to Incognito as updates arrive in 1k batches

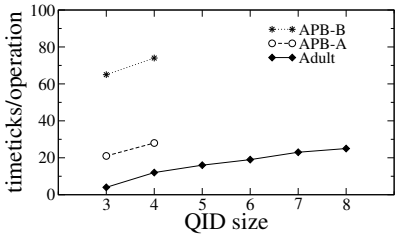

Figure 7: Time per reindexing operation vs. *QID* size for 5k update batches (k=10)

Table 3: Results for various sizes of continuous update batches and various k values (Adult dataset)

	upd size	KANIS #ReInd	msg/node	BW	distortion deviation
k=5	1k	1	5.3	1.6M	2%
	5k	1	5.1	1.7M	1%
	10k	1	5.1	1.7M	1%
k=10	1k	2	9.8	2.6M	4%
	5k	2	9.8	3.3M	4%
	10k	2	9.8	4.5M	4%
k=15	1k	2	9.8	2.5M	3%
	5k	2	9.8	3.1M	3%
	10k	2	9.8	4.4M	3%

the size of the relocated data. The quality of the anonymized dataset is captured by the maximum % difference in distortion between a resulting k-anonymized table and the optimal k-anonymized table, namely the one that would have been created, had it been anonymized by Incognito from scratch. We term this value *deviation* and the optimal case *baseline*.

We observe that our method manages to maintain the privacy of the continuously growing dataset while keeping distortions close to the baseline ones (less than 4% deviation). *KANIS* achieves this online, with less than 2 reindexing procedures and a communication cost in messages per node and bandwidth certainly affordable by current systems. For the smallest k value we observe that the communication cost as well as the distortion deviation are almost half the respective measures for k=10 and k=15. This is natural, since smaller k values translate to a more relaxed privacy policy.

In Figure 5 we plot the distortion deviation with respect to the baseline case throughout the experiment, as update batches of 1k tuples arrive at the system. This deviation is attributed to the fact that drilldown anonymization, to minimize communication costs and utilize as much local (rather than global) knowledge as possible, considers only the P combinations that strictly lie below the current P. The deviation remains less than 5% regardless of the value of k and demonstrates a steady behavior (with only minor fluctuations) despite the continuous insertion of updates. Exploring ways to enable reindexings to all possible P combinations is part of our future work.

Figure 6 presents the gain in distortion compared to Incognito, achieved by *KANIS* by virtue of its drilldown anonymization method, which effectively monitors the number of tuples belonging to each stored tree and adaptively reindexes its contents if a P resulting in less distortion is discovered. As updates arrive at the system the gain grows, reaching over 10% for the highest k values. This is natural, since the addition of tuples increases the counts for the existing ECs, thus increasing the possibility of finding new ECs with less distortion that conform to the k constraint.

Finally, Figure 7 plots the average time needed per reindexing operation in order to keep the dataset 10-anonymized for update batches of 5k tuples and for various *QID* set sizes. In this graph results for the APB datasets are added as well, but in their case the *QID* parameter cannot exceed 4 (4-d dataset). Since it is a simulation-based experiment, time is represented by simulation time-ticks. The increase in the number of the *QID* attributes complicates both the decision and the reindexing process and results in an increase in time per operation. Yet, the increase is not linear, since *KANIS* takes advantage of all of its resources, which share the processing. Indeed, each participating node handles the reindexing of the part of the dataset it hosts, parallelizing the procedure to a great extent. Moreover, the cost of the reindexing operation rises with the dataset size, due to the relocation of the data. Again, thanks to the parallelization, the increase is not directly proportional: While APB-B consists of 5 times more tuples than APB-A, its reindexing lasts only 2.6 times longer.

3.2 Scaling the Number of Tuples

In this set of experiments we aim to evaluate the performance as well as the communication cost of our system with an increasing dataset size. To that end, we have used the largest of our generated APB datasets (APB-C). To an initial k-anonymized table of 1M tuples we pose batches of 1M updates. Figure 8 shows the gain in distortion for various k values. Besides affirming the previous findings, that the gain in distortion rises with the addition of new data, we prove that our system is capable of handling increasing dataset sizes achieving more than 20% of quality improvement compared to the centralized algorithm. It is worth noting that, in this set of experiments, the distortion deviation is 0 during the biggest part of them, with a maximum value of 3% reached only under the most strict policy (k=1000).

3.3 Scaling the Number of Nodes

Finally, we evaluate the horizontal scalability of our method by varying the number of participating nodes from 16 to 128. We believe this to be a more than adequate number for our target application. Adding more nodes on one hand increases its ability to handle large volumes of data, but on the other hand imposes bigger communication costs. Utilizing the two largest APB datasets (6–12M tuples) we plot the communication cost of the reindexing process with various network sizes (see Figure 9).

As expected, the average number of messages required to perform a reindexing operation increases with the increase in network nodes, as flooding becomes more costly. However, this number is scattered over the network nodes, resulting in a decreasing average load per node. Apart from that,

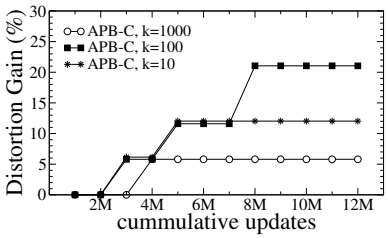

Figure 8: The gain of *KANIS* in distortion compared to Incognito as updates arrive in 1M batches

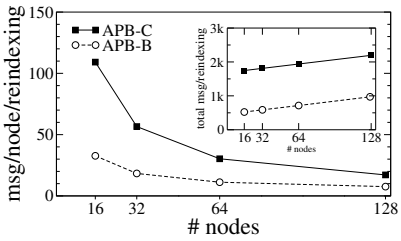

Figure 9: The communication cost of *KANIS* in messages per reindexing with varying network size

we have observed that *KANIS* manages to maintain steady gains in distortion regardless of the network size, performing the necessary reindexings.

4. RELATED WORK

The fist works to study k-anonymization were those by Samarati and Sweeney [18, 17], who proposed mechanisms to protect privacy using the ideas of suppression and generalization. Suppression removes some attribute values ore even the whole tuple from the table. Generalization, the most popular anonymization mechanism, is achieved by replacing an exact QID value with a more general one, e.g., an integer value can be substituted by a range.

A common taxonomy for generalization algorithms divides them to global and local recoding methods. Global recoding generalizes a table at the domain level, mapping all tuples of a QID value to the same EC [10, 11]. Incognito [10] exhaustively discovers the minimal full-domain generalizations of a large database table. Local recoding generalizes a table at cell levels, resulting in tuples with the same QID being mapped to different ECs [13, 19].

All the above methods deal with the anonymization of one centralized database and do not consider data distributed over multiple locations. The work in [9] attempts to propose a distributed k-anonymity algorithm but for vertically partitioned data. The works of Zhong *et al* [20, 21] consider horizontally distributed data, but focus on k-anonymization by suppression only. To the best of our knowledge, this work is the first one to concern anonymization of data horizontally partitioned and distributed among multiple network nodes through recoding.

5. CONCLUSIONS

In this paper we proposed *KANIS*, a system that preserves the anonymity of fully distributed data under continuous updates employing an adaptive scheme that adjusts the level of hierarchy generalization according to the privacy constrains. This is achieved in an online manner, through efficient rollup and drilldown operations, while minimizing data distortion with just a small communication overhead. An initial evaluation of our prototype shows that *KANIS* manages to preserve k-anonymity while improving the data quality up to 22% compared to a popular centralized global recoding algorithm. It achieves a near-optimal distortion regardless of the network or dataset size, with a communication overhead scattered among the participating nodes.

Acknowledgments

This work was supported by the European Commission in terms of the ARCOMEM FP7 ICT Project (FP7-270239).

6. REFERENCES

[1] *OLAP Council APB-1 OLAP Benchmark*. http://www.olapcouncil.org/research/resrchly.htm.

[2] M. Barbaro and T. Zeller. A Face Is Exposed for AOL Searcher No. 4417749, 2006. The New York Times.

[3] Personalization reports. http://www.choicestream.com/who/news.php.

[4] http://dropl.at/.

[5] A. Frank and A. Asuncion. UCI machine learning repository, 2010. http://archive.ics.uci.edu/ml.

[6] FreePastry, http://freepastry.rice.edu/FreePastry.

[7] B. Fung et al. Anonymity for Continuous Data Publishing. In *Proceedings of EDBT'08*.

[8] B. Gedik and L. Liu. Protecting location privacy with personalized k-anonymity: Architecture and algorithms. *IEEE TMC*, 7(1), 2008.

[9] W. Jiang and C. Clifton. A Secure Distributed Framework for Achieving k-Anonymity. *The VLDB journal*, 15(4):316–333, 2006.

[10] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Incognito: Efficient Full-Domain k-Anonymity. In *Proceedings of SIGMOD'05*.

[11] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *Proceedings of ICDE'06*.

[12] J. Li, R. Wong, A. Fu, and J. Pei. Achieving k-Anonymity by Clustering in Attribute Hierarchical Structures. In *Proceedings of DaWaK'06*.

[13] J. Li, R. Wong, A. Fu, and J. Pei. Anonymization by Local Recoding in Data with Attribute Hierarchical Taxonomies. *IEEE TKDE*, pages 1181–1194, 2008.

[14] Location and privacy: Where are we headed? http://www.microsoft.com/privacy/dpd.

[15] Nextbus inc. http://www.nextbus.com/.

[16] J. Pei et al. Maintaining K-Anonymity against Incremental Updates. In *SSDBM*, 2007.

[17] P. Samarati. Protecting Respondents' Identities in Microdata Release. *IEEE TKDE*, pages 1010–1027, 2001.

[18] L. Sweeney et al. k-anonymity: A Model for Protecting Privacy. *International Journal of Uncertainty Fuzziness and Knowledge Based Systems*, 10(5):557–570, 2002.

[19] J. Xu et al. Utility-Based Anonymization Using Local Recoding. In *Proceedings of KDD'06*.

[20] S. Zhong. On Distributed k-Anonymization. *Fundamenta Informaticae*, 92(4):411–431, 2009.

[21] S. Zhong, Z. Yang, and R. Wright. Privacy-Enhancing k-Anonymization of Customer Data. In *Proceedings of EDBT'05*.