K-Anonymity for Crowdsourcing Database

Sai Wu, Xiaoli Wang, Sheng Wang, Zhenjie Zhang and Anthony K.H. Tung

Abstract—In crowdsourcing database, human operators are embedded into the database engine and collaborate with other conventional database operators to process the queries. Each human operator publishes small HITs (Human Intelligent Task) to the crowdsourcing platform, which consist of a set of database records and corresponding questions for human workers. The human workers complete the HITs and return the results to the crowdsourcing database for further processing. In practice, published records in HITs may contain sensitive attributes, probably causing privacy leakage so that malicious workers could link them with other public databases to reveal individual private information.

Conventional privacy protection techniques, such as *K-Anonymity*, can be applied to partially solve the problem. However, after generalizing the data, the result of standard K-Anonymity algorithms may render uncontrollable information loss and affects the accuracy of crowdsourcing. In this paper, we first study the tradeoff between the privacy and accuracy for the human operator within data anonymization process. A probability model is proposed to estimate the lower bound and upper bound of the accuracy for general K-Anonymity approaches. We show that searching the optimal anonymity approach is NP-Hard and only heuristic approach is available. The second contribution of the paper is a general feedback-based K-Anonymity scheme. In our scheme, synthetic samples are published to the human workers, the results of which are used to guide the selection on anonymity strategies. We apply the scheme on Mondrian algorithm by adaptively cutting the dimensions based on our feedback results on the synthetic samples. We evaluate the performance of the feedback-based approach on US census dataset, and show that given a predefined K, our proposal outperforms standard K-Anonymity approaches on retaining the effectiveness of crowdsourcing.

1 Introduction

Current crowdsourcing platforms, such as Amazon AMT[1] and Crowdflower[2], adopt the new LaaS (Labor as a Service) model. After the employer submits his job to the crowdsourcing platform, thousands of registered workers will become his candidate employees to provide the labor on demand. Similar to the Cloud system, the crowdsourcing platforms charge their users by the pay-as-you-go model. They are now considered as the largest online human resource providers.

The LaaS model in crowdsourcing allows us to exploit the unlimited human workers to complete the complex jobs, which are hard for the computers. The idea has been introduced into the design of database systems [1][2][3][4] to process similarity join, fuzzy search and aggregation. In those systems, new database operators involving human labors are implemented to utilize the power of the *crowd*. The main function of the human operator is to generate crowdsourcing jobs for the database tuples and collect the answers from human workers, which will be transformed and passed to the other database operators for processing. The crowdsourcing jobs usually contain one or multiple database tuples and a question for the human workers. The answers to the question are collected from all the participated workers and the human operator can adopt different models to merge the answers [4].

As an example, the HR agents, such as 51Job [3] and ChinaHR [4], receive thousands of new requests from both the users and companies per week. In particular, millions of users register their curriculum vitae (CV) in the database, and thousands of companies submit their job positions to the agents. The HR agents need to link the users to the appropriate positions based on their education level, working experiences and other personal information. After simple rule-based pruning on the CVs, the HR agents need to go through a tedious process on every candidate CV personally. It is challenging to design a good computer algorithm to process the job linkage automatically, as some attributes of the CV data (e.g., working experience) contain complex semantics and different job positions could pose different requirements. With the emergence of crowdsourcing database techniques, such process could be replaced by an alternative solution, by publishing the CVs and available positions on a crowdsourcing platform. This potentially reduces the huge cost of these HR agents, since human workers on the Internet may provide equally good service on candidate qualification review but with a tiny payment.

To support such HR applications, the human operator is supposed to disclose individual information in the process of crowdsourcing, which may lead to increasing concerns on the privacy. Given the curriculum vitae data in Table 1 (eduction level is mapped to a numeric value and the

- *Sai Wu is with the College of Computer Science, Zhejiang University, Hangzhou, P.R. China, 310027 .*
  E-mail: wusai@zju.edu.cn
- *Xiaoli Wang, Sheng Wang and Anthony K.H. Tung are with School of Computing, National University of Singapore, Singapore, 117417 .*
  E-mail: {xiaoli,wangsh,atung}@comp.nus.edu.sg
- *Zhenjie Zhang is with Advanced Digital Sciences Center, Illinois at Singapore Pte. Ltd.*
  E-mail: zhenjie@adsc.com.sg

1. http://aws.amazon.com/mturk
2. http://crowdflower.com
3. http://www.51job.com
4. http://www.chinahr.com/

TABLE 1
An example database with CV records

ID	Age	Gender	Zipcode	Education	Workclass	Married	Children	Income($)
1001	23	F	345010	3	5	S	0	17,287
1002	28	F	345055	2	2	M	1	10,057
1003	31	M	333239	4	8	S	0	22,308
1004	35	F	333123	3	1	D	2	10,483
1005	43	M	333120	5	4	M	3	38,218
1006	38	F	333460	2	4	M	3	10,257

TABLE 2
The anonymized database after running K-Anonymity algorithm

ID	Age	Gender	Zipcode	Education	Workclass	Married	Children	Income($)
1001	[20-30]	F	[345xxx]	3	5	S	0	17,287
1002	[20-30]	F	[345xxx]	2	2	M	1	10,057
1003	[30-45]	M	[333xxx]	4	8	S	0	22,308
1004	[30-45]	F	[333xxx]	3	1	D	2	10,483
1005	[30-45]	M	[333xxx]	5	4	M	3	38,218
1006	[30-45]	F	[333xxx]	2	4	M	3	10,257

workclass only shows the previous work level), the human operator needs to reveal the sensitive attributes, such as *age, gender* and *education*, to the human workers in order to get correct results. This may cause unexpected information leakage and the malicious workers could retrieve the full details of a specific person by joining the record in the job with certain public database. The potential privacy threat curbs the real systems on adopting such crowdsourcing techniques.

In conventional database systems, K-Anonymity techniques [5][6][7][8] are proposed to protect the privacy of published data. After grouping and generalizing the data, K-Anonymity guarantees that any tuple in the release cannot be distinguished from at least another K-1 tuples. In this way, the attacker cannot join the published data with other public databases to reveal the identify of a specific person. However, K-Anonymity affects the performance of crowdsourcing, as generalization and grouping leads to information loss. If we provide the anonymized data to the human workers, they may fail to return the correct answer. The system needs to address the tradeoff between the privacy and accuracy. For example, Table 2 is one anonymized version of Table 1. The records have been grouped based on the *age* and *zip* attribute. If a position requires to hire a female below 25, the human worker cannot do a correct decision for anonymized data. If the database system optimistically recommends the first two records, only one out of the two candidates is actually qualified.

In this paper, we first study the effect of K-Anonymity on the crowdsourcing results. We formulate the problem using a matrix representation. Each entry of the matrix denotes the probability that human workers return certain answer on a particular record. This model enables us to estimate the lower bound and upper bound of the accuracy for the K-Anonymity results. These bounds provide overall guidelines on the possible reduction on the utility of K-Anonymity before crowdsourcing the records.

To generalize the records to enforce the privacy requirement as well as maximize the utility of crowdsourcing, we show that our problem is consistent with previous K-Anonymity approaches, i.e. [8][9], which target at minimizing the information loss. Unfortunately, the problem is proved to be NP-hard and only heuristic approach is available. To provide a high quality anonymity strategy, the second contribution of the paper is to propose a feedback based K-Anonymity approach. Figure 1 summarizes the idea. In particular, before anonymizing the records in the database, data-independent random samples are generated and sent to crowdsourcing platform for testing. Our scheme thus exploits the results to the synthetic samples from the crowd, during the optimization process when anonymizing the real tuples in the database. In particular, we combine our scheme with Mondrian Algorithm [8] to partition the dimension (domain of each attribute) iteratively, where each partition represents an anonymization group. We evaluate the performance of the feedback-based approach on US census dataset, and show that given a predefined K, our proposal outperforms standard K-Anonymity approaches on retaining the effectiveness of crowdsourcing.

The remainder of the paper is organized as follows. In Section 2, we introduce the human operator and formalize the K-anonymity requirement for the human operator. In Section 3, we analyze the effect of anonymity on the results of crowdsourcing. We present our matrix-based probability model. In Section 4, we propose our feedback-based approach, which adaptively partitions the space based on the crowdsourcing results. Section 5 evaluates the proposed approach using real dataset and Section 6 reviews previous work on crowdsourcing and K-anonymity. We conclude the paper in Section 7.

2 PRELIMINARIES

2.1 AMT and Human Operator

AMT is a crowdsourcing platform, allowing the users to publish and accept jobs. The job in AMT is called HIT

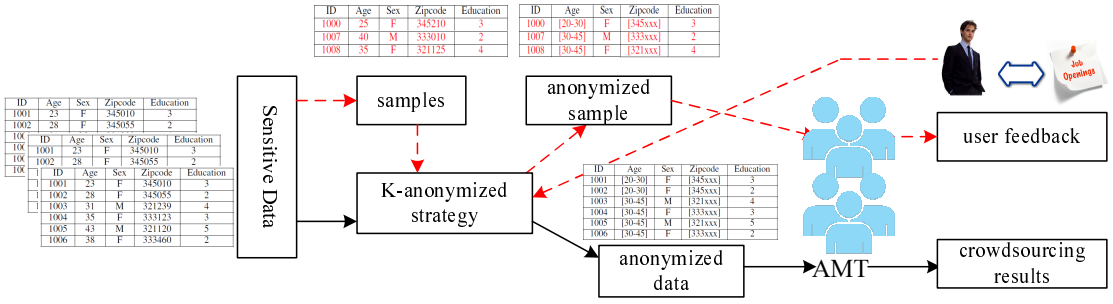


Fig. 1. Work Flow of Feedback-based K-Anonymity

(Human Intelligent Task) and currently, AMT has more than 265K available HITs. One HIT can be assigned to multiple workers and the HIT publisher is required to pay the workers for accepting their answers. Besides, AMT also charges the publisher some service fees for each HIT.

By default, the published HIT is accessible for all human workers. However, the worker can only check the details of the HIT after he accepts the job. The user can set the number of workers for each HIT, limiting the information leakage. In the example of job recommendation, we partition the CV records into several groups. Each group contains N records, which are published together as a HIT. Before accepting the HIT, the workers only know that the HIT is to recommend people some job positions, but they cannot view the detailed CV data. The HIT is designed based on the AMT's template. In the head of the HIT, we list M available positions and the corresponding requirements. In the main part of the HIT, we list N questions, asking the workers to select one most feasible position for each record. Because different HITs have different set of workers and the data are partitioned into multiple groups, each worker only sees a small portion of the CV data and it is impossible for him to recover the whole dataset. Even a group of malicious workers collaborate to crawl some CV data, they are unable to identify the record of a specific person, as the CV data are protected by K-Anonymity property.

Generating and publishing HITs can be abstracted into the human operator, which is built as a basic database operator interacting with other operators. Given the user data in Table 1, the corresponding query can be written as:

SELECT id, getposition(*) FROM user

getposition is a user-defined function (UDF) involving the human operator, which can be defined formally as:

$$OP_h : (T \times q) \rightarrow A$$

T denotes a set of tuples or partial results from other database operators. A is the subset of the answers and q is the question to the worker. In UDF *getposition*, T contains a single tuple from the user table; q equals to "recommending a job for the user"; and A just has one answer, selected from a list of jobs.

Besides the simple select query, we can also apply the human operator to handle the aggregation query:

SELECT getcandidate(, Job) FROM user GROUP BY zip*

The above query returns one user in each area, who is the best candidate for a job. In UDF *getcandidate*, T contains all tuples with the same zip code; q equals to "selecting one best candidate for the Job"; and A provides a single answer for the user ID.

The introduction of human operator does not affect the database engine. It first generates the query plan, where the human operator is normally used as the root of the expression tree. The traditional database operators are processed as before. Their partial results are then used as the input for the human operators to generate questions to the workers. The database engine is blocked to wait for the workers' answers from the AMT. When all HITs complete, the database engine resumes the plan (if necessary) for further processing.

2.2 K-Anonymity for Crowdsourcing Data

The human operator publishes a set of tuples to the crowdsourcing platform. If the tuple contains sensitive attributes, due to privacy concern, we cannot adopt the crowdsourcing techniques. In Table 1, *age*, *gender* and *zipcode* can be used to join with other public databases to reveal the identity of the user. They are quasi-identifiers. The rest attributes (*education*, *workclass*, *married* and *children*) are sensitive attributes. To guarantee the privacy requirement, we can apply the K-Anonymity techniques [5]. Instead of publishing the original data, the human operator generates the crowdsourcing jobs using the anonymized data. For example, Table 2 is the 2-anonymity results for Table 1. As anonymity causes information loss, the accuracy of crowdsourcing is affected. We need to design a new K-Anonymity algorithm to achieve the optimal anonymity strategy.

Definition 1: **Optimal K-Anonymity for Crowdsourcing**
Given a dataset D, let S be its K-Anonymity version. S is the optimal K-Anonymity strategy for D, if there is no other K-Anonymity strategy S', which leads to a better accuracy than S for the human operator.

The accuracy of a K-Anonymity strategy is computed as $\frac{|A \cap A'|}{|A|}$, where A and A' denote the answer sets of human workers by publishing the original dataset and the anonymized dataset respectively. If the accuracy is 100%,

TABLE 3
Notations

Parameter	Descriptions
G_i	a group of tuples
N_i	the number of tuples in G_i
r_i	a possible crowdsourcing result returned by tuples in G_i
P_{ij}	the probability of returning r_j as a result by G_i
S_i	a multi-dimension data space
$\theta(t)$	the anonymized form of a tuple t
$P_{S_i}(t \to r, \theta(t) \to r)$	the probability that workers return the same answer r for both t and $\theta(t)$ in S_i

the anonymity does not affect the decision of the human workers.

2.2.1 Data Publication

In fact, not all attributes are required to be published. In the example above, *age, gender, zip, edu* and *work experience* are highly correlated with the job recommendation, while *married* and *children* are not. To reduce the information leakage, we do not need to publish the later two attributes. In fact, this relationship can be caught by the function dependency. Let *job* be the missing attribute of the table. We have:

$$age, gender, zipcode, education, workclass \to job$$

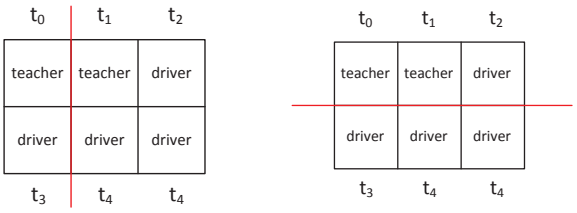
Formally, we define the core attribute set as:

Definition 2: **Core Attribute Set**
Let C_r be the result attribute. Attribute set $\mathcal{C}$ is the core attribute set, if $\mathcal{C} \to C_r$ and there is no $\mathcal{C}' \subset \mathcal{C}$ satisfying $\mathcal{C}' \to C_r$.

The function dependency is predefined before the query is being processed and the human operator only needs to publish the core attributes to the crowdsourcing platform. In the following discussion, we only keep the core attributes.

2.2.2 Prior Knowledge

Prior knowledge can help us improve the accuracy of the result. For example, if we know that a specific job requires females at 18-25 with height above 5.5 feet, we can just apply the rule to find the candidates. However, if prior knowledge is available or we can generate such knowledge via data mining and machine learning algorithms, we do not need to adopt the crowdsourcing approach. A sophisticated computer algorithm can provide good enough results with less cost. Therefore, in the rest of the paper, we assume that no prior knowledge is available and the association rules between the core attributes and the results are difficult to discover (e.g, the relationship between work experiences, educations and a specific job). For reference, Table 3 shows the parameters used in this paper.



(a) Strategy 1 (b) Strategy 2

Fig. 2. Demonstration of Anonymity Strategy

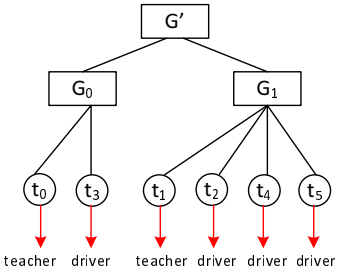


Fig. 3. Effect of Grouping

3 A Probability Model

In this section, we analyze how the anonymized data affect the decision of human workers. We give a lower bound and upper bound for the accuracy of crowdsourcing results. It indicates that although previous K-Anonymity approaches provide privacy guarantees, they may prevent the human workers from generating the right answer. Therefore, in the next section, we propose our new anonymity approach.

As mentioned before, different anonymity strategies have various effects on the crowdsourcing accuracy. Figure 2 illustrates the idea. Tuples t_0 to t_5 represent the records of Table 1. Suppose t_0 and t_1 are good candidates for the *teacher* job, while the rest persons can work as *drivers*. Let K be 2. Figure 2(a) generates two groups $\{t_0, t_3\}$ and $\{t_1, t_2, t_4, t_5\}$. The corresponding tuples will be anonymized accordingly. For example, t_0 will be transformed into *(1001, [23-35], F, 3xxxxx, 3, ...)*. Figure 2(a) is not a good anonymity strategy, as each anonymized tuple refers to the tuples that may have different recommended jobs. Instead, Figure 2(b) shows a better solution by generalizing tuples differently. In fact, the strategy in Table 2 is the optimal solution. Our job is to search for the optimal anonymity strategy given a predefined K. However, note that the answer distribution (job distribution in our example) is unknown before crowdsourcing. Therefore, we need a model to estimate the quality of a K-Anonymity strategy.

3.1 Matrix Model

In K-Anonymity technique, tuples are grouped and generalized, so that one tuple cannot be distinguished from at least K-1 other tuples. Such transformation causes information loss and may lead to incorrect result in crowdsourcing. Figure 3 illustrates the effect of grouping tuples by the

strategy of Figure 2(a). We have two groups, G_0 and G_1. For a random tuple t in G_1, t's answer follows the distribution of (25%: *teacher*, 75%: *driver*). If we further generalize the tuples to G', the answer distribution will change to (50%: *teacher*, 50%: *driver*). Even if the human worker knows the ground-truth answer for each tuple, he cannot provide an accurate answer for the corresponding group. Suppose the human worker is reliable and willing to provide the correct answer (e.g., his answer is not anticorrelated to the probability distribution), he has two selections:

1) Always return the answer with the highest probability.
2) Return the answer based on the same probability distribution in the grouping strategy.

For G_1, in the first strategy, the worker will always return *driver*. The probability of providing a correct answer is:

$$0.25 \times 0 + 0.75 \times 1 = 0.75$$

In the second strategy, the worker returns *teacher* and *driver* with the probabilities of 25% and 75%, respectively. Thus, the probability of providing a correct answer is:

$$0.25 \times 0.25 + 0.75 \times 0.75 = 0.625$$

More formally, for a group G_i and possible answers $(r_0, ..., r_{m-1})$, let P_{ji} denote the probability of a random tuple in G_i returning r_j as the result. The estimated accuracy of the first strategy is $\bar{P}_i = P_{xy}$, where $P_{xy} \geq P_{ji}$ for any j and i, while The estimated accuracy of the second strategy is:

$$P_i = \sum_{j=0}^{m-1} P_{ji}^2$$

$\bar{P}_i$ is always higher than P_i. In fact, $\bar{P}_i$ and P_i are the upper-bound and lower-bound of the estimated accuracy in the real systems, as human workers normally prefer a comprised strategy between the above two strategies. In this paper, we use P_i to estimate the accuracy. Our intuition is to guarantee a good performance for the worst case.

By extending the idea, we can model the accuracy of crowdsourcing as a probability matrix. Figure 4 shows the idea. The matrix has m rows and n columns. m is the number of possible answers and n is the number of groups generated in the anonymity process. To guarantee the K-anonymity property, each group should contain at least K tuples. For the element P_{ij}, it denotes the probability that tuples in the jth group use the ith answer as the answer.

Obviously, there are two properties for the matrix. First, the sum for the elements of the same column is 1. Namely,

$$\sum_{i=0}^{m-1} P_{ij} = 1$$

Second, suppose $\mathcal{P}_i$ denote the probability of selecting the ith answer. Let N and N_j denote the total number of tuples and the number of tuples in the jth group, respectively. Given a random tuple, it belongs to the jth group with a probability of $\frac{N_j}{\sum_{x=0}^{n-1} N_x} = \frac{N_j}{N}$. By summing up all probabilities in a row, we have

	G_0	G_1	G_2	G_3	G_4	G_5
teacher	P_{00}	P_{01}	P_{02}	P_{03}	P_{04}	P_{05}
driver	P_{10}	.....	.....	.....	.....	.....
programmer	P_{20}	.....	.....	.....	.....	.....
researcher	P_{30}	.....	.....	.....	.....	.....
chef	P_{40}	.....	.....	.....	.....	.....
waiter	P_{50}	.....	.....	.....	.....	.....

Fig. 4. Probability Matrix

$$\sum_{j=0}^{n-1} \frac{N_j P_{ij}}{N} = \mathcal{P}_i$$

The above two properties will be applied to estimate the lower bound and upper bound of the accuracy.

3.2 Accuracy Bound

Based on our previous discussion, the crowdsourcing accuracy of the ith answer can be estimated as:

$$X_i = \sum_{j=0}^{n-1} \frac{N_j P_{ij}^2}{N} \tag{1}$$

X_i follows the same form as $\mathcal{P}_i$. In fact, we can apply the Jensen's inequality to link to two parameters.

We define a continuous function $f : (0, \infty) \rightarrow (0, \infty) = x \rightarrow x^2$. The second derivative of f exists and satisfies $f'' = 2x^0 > 0$ for all $x > 0$. So f is a strictly convex function and according to Jensen's inequality,

$$\begin{aligned} \mathcal{P}_i^2 &= (\sum_{j=0}^{n-1} \frac{N_j P_{ij}}{N})^2 = f(\sum_{j=0}^{n-1} \frac{N_j P_{ij}}{N}) \\ &\leq \sum_{j=0}^{n-1} \frac{N_j}{N} f(P_{ij}) = \sum_{j=0}^{n-1} \frac{N_j P_{ij}^2}{N} \end{aligned} \tag{2}$$

The accuracy of all the answers is estimated as:

$$Acc = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} \frac{N_j P_{ij}^2}{N} \geq \sum_{i=0}^{m-1} \mathcal{P}_i^2 \tag{3}$$

Equation 3 shows how to estimate the lower bound of the accuracy. If we know the probability distribution of the answers, we can compute the lower bound for any K-anonymity algorithm. Interestingly, if no prior knowledge is known, the lower bound is not correlated to K. It is only affected by the number of possible answers. Suppose we have two answers and the answer distribution is (80%, 20%). The lower bound of any K-anonymity algorithm is 68%. However, if the two answers follow the uniform distribution (50%, 50%), the lower bound is 50%.

Equation 3 also shows that if there are too many possible answers, the K-anonymity algorithm may lead to a very low

accuracy for the crowdsourcing. Therefore, in that case, we need to carefully design our anonymity strategy.

We then apply the first property of the matrix model to estimate the upper bound. The accuracy can be rewritten as:

$$\begin{aligned}\alpha &= \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\frac{N_j P_{ij}^2}{N} = \sum_{j=0}^{n-1}\sum_{i=0}^{m-1}\frac{N_j P_{ij}^2}{N} \\ &= \sum_{j=0}^{n-1}\frac{N_j}{N}\sum_{i=0}^{m-1}P_{ij}^2 \leq \sum_{j=0}^{n-1}\frac{N_j}{N} = 1 \end{aligned} \quad (4)$$

The upper bound of the accuracy is obtained when for any group G_i, there is a θ satisfying that

$$P_{ij} = \begin{cases} 1 & \text{if } j = \theta \\ 0 & \text{otherwise} \end{cases}$$

The above requirement, in fact, requires a perfect anonymity strategy, where each group only contains the tuples with the same answer. Without prior knowledge, perfect anonymity strategy is not possible and if such knowledge is available, we can apply the classification techniques instead of crowdsourcing. Hence, in this paper, we assume that the perfect anonymity strategy is not possible.

3.3 Effect of Anonymity

In this section, we study how the anonymity affects the accuracy. We follow the idea of hierarchical anonymity approach. The tuples are generalized into groups and small groups are combined into larger ones, until all groups satisfy the K-Anonymity property. Consider the most basic operation in the approach. Given n groups, we find that there are two groups breaking the K-Anonymity and we group them together. Let G_0 and G_1 be the corresponding group. We use $\bar{G}$ and $\bar{P}_i$ to denote the new group and its element in the matrix. We have:

$$\bar{P}_i = \frac{N_0}{N_0+N_1}P_{i0} + \frac{N_1}{N_0+N_1}P_{i1} \quad (5)$$

Before grouping, the accuracy of crowdsourcing is estimated as:

$$\alpha = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\frac{N_j P_{ij}^2}{N} \quad (6)$$

After grouping, it changes to

$$\alpha' = \sum_{i=0}^{m-1}(\sum_{j=2}^{n-1}\frac{N_j P_{ij}^2}{N} + \frac{N_0+N_1}{N}\bar{P}_i^2) \quad (7)$$

So we need to estimate the effect ($\delta = \alpha - \alpha'$) of grouping.

Lemma 1: The group operation decreases the accuracy of crowdsourcing.

Proof: Combining Equation 6 and 7, we have

$$\begin{aligned}\delta N &= \sum_{i=0}^{m-1}(N_0 P_{i0}^2 + N_1 P_{i1}^2 - (N_0+N_1)\bar{P}_i^2) \\ &= \frac{N_0 N_1}{N_0+N_1}\sum_{i=0}^{m-1}(P_{i0}^2 + P_{i1}^2 - 2P_{i0}P_{i1}) \\ &\geq 0 \end{aligned} \quad (8)$$

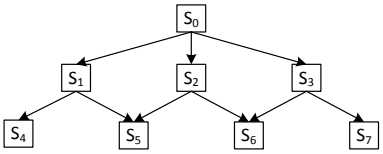


Fig. 5. Grouping Tree

$\delta \geq 0$ indicates that after combining the two groups, the expected accuracy of crowdsourcing decreases. □

Theorem 1: Given two anonymity strategy S and S', if S' can be generated from S by a set of grouping operations, S will lead to a better accuracy for crowdsourcing than S'.

Proof: By applying the grouping operations, we can show a path from S to S' : $S \rightharpoonup S_0 \rightharpoonup S_1 ... \rightharpoonup S_n \rightharpoonup S'$. We iteratively apply Lemma 1 and the accuracies follow $acc(S') \leq acc(S_n) \leq ... \leq acc(S)$. Therefore, in the K-Anonymity algorithm, once we satisfy the K-Anonymity property, we should stop the anonymity process to provide a good accuracy. □

Let S_0 be the original dataset and S_i be one of its anonymity strategy. We can organize all the anonymity strategies as a tree (strictly speaking, it is a directed graph). Figure 5 shows an example. In the tree, the ancestor nodes provide higher accuracies for the human operator than their child nodes based on Theorem 1. But it is still difficult to compare the qualities of sibling nodes (e.g., node S_4 and S_5). If the probability matrix is available, we can estimate the quality of different anonymity strategies via Equation 1. However, that requires us to know the ground-truth results of each tuple, which conflicts with our intuitions.

We find that the accuracy is, in fact, correlated to the information loss. The human workers cannot provide the correct answer, as some information are missing or fuzzy. The goal of optimal K-Anonymity is consistent with previous K-Anonymity algorithm: finding the anonymity strategy with least information loss. There are many cost models defining the information loss, such as discernability metric [9], normalized average equivalence class size metric [8] and classification metric [10]. However, for all K-Anonymity algorithms investigated in the literature, the optimal K-Anonymity problem in general settings is proven NP-hard [7][11].

In the following, we analyze the difficulty of K-Anonymity under crowdsourcing. By employing an appropriate strategy, a conventional summation-based K-Anonymity problem can be transformed to an equivalent K-Anonymity crowdsourcing problem under our setting. This intuition is formalized by the following theorem.

Theorem 2: Any standard summation-based K-Anonymity problem in continuous space can be reduced to a K-Anonymity problem under crowdsourcing by building virtual human workers with designed answering strategy.

Proof: We model a standard K-Anonymity problem as a bipartite graph. On the right side, each node represents a personal record from $\{x_1, x_2, \ldots, x_N\}$, each of which is

a person in the database. On the left side, each node is a grouping S_j, which is connected to at least $n_j \geq k$ personal records on the right. We assume that the nodes on the left side cover all possible groupings on the database. Each grouping node is also associated with a cost C_j. Different K-Anonymity problem may adopt different cost assignment based on the criterion, e.g. the average distance between the records in the group. The problem of K-Anonymity is finding a subset of groups covering all records with minimal cost summation on the selected groups. We then construct the question-answering strategy as follows. When a question on grouping S_j is asked on the crowdsourcing platform, the human worker returns answer r_0(resp. r_1) with probability P_{0j}(resp. probability $1 - P_{0j}$) such that

$$P_{0j}^2 + \left(1 - P_{1j}^2\right)^2 = \frac{C_j}{n_j \max_l C_l}.$$

There are two feasible solutions for P_{0j} satisfying the equation above. The sum of these two solutions is exactly 1. Either of the solutions works for our construction. By applying Equation 1, it is straightforward to show that the total error for prediction is $\sum_{S_{\phi_j}} \frac{C_{\phi_j}}{\max_l C_l}$, if our algorithm picks up t groupings as $\{S_{\phi_1}, \ldots, S_{\phi_t}\}$. Therefore, the error of K-Anonymity under crowrdsourcing is proportional to the cost of original K-Anonymity problem. this completes the proof of the theorem. □

The theorem above implies the NP-hardness of the K-Anonymity problem under crowdsourcing. In the rest of the paper, we will show to exploit human workers' feedbacks and existing heuristics on conventional K-Anonymity problems to improve the accuracy of the results.

4 Feedback-based Approach

As mentioned in previous sections, K-Anonymity may significantly degrade the quality of crowdsourcing results and finding optimal K-Anonymity strategy is NP-complete. Therefore, in this section, a heuristic approach, which exploits the human workers' feedbacks, is introduced to maximize the accuracy of crowdsourcing.

4.1 Samples and K-Anonymity

We adopt the DataSynth [12] to generate samples based on the original dataset. DataSynth is a tool to generate synthetic data for data masking. The samples follow the same data distribution and correlations as the original dataset. We publish the samples into the AMT to collect the ground-truth results (e.g., which job position should be recommended to the user). Publishing the samples will not reveal the identifies of the user record, as they are dummy records. As an example, Table 4 shows the dataset that includes real tuples from Table 1 and the generated synthetic samples (the red tuples are the synthetic samples). In this table, the *age*, *gender* and *zipcode* are quasi-identifiers and others are sensitive attributes. Table 5 is the 2-Anonymity result for Table 4.

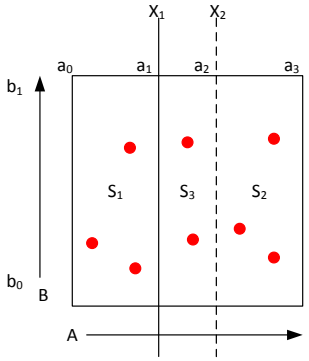


Fig. 6. Alternative Partitions

The samples are anonymized and published into the AMT to collect the new answers. The answers will be compared to the ground-truth results to estimate the quality of the anonymity approach. Suppose there is one job position, which is specially appropriate for the women above 35. If we publish the samples in Table 5, the human workers cannot decide how to recommend the job position, as the samples have the age range [30-45]. If the human workers assume that the tuples are distributed in the range evenly, they may recommend the job position in a probability of 0.667. By comparing the worker's answer and the ground-truth result, we can estimate the accuracy of the anonymity strategy.

Generally, for a tuple t with m quasi-identifiers, we define $\theta(t)$ as its anonymity result.

$$\theta(t) = [l_1, u_1] \times [l_2, u_2] \times ...[l_m, u_m]$$

where $[l_i, u_i]$ represents the partitioning strategy and if the tuple is not partitioned in the ith dimension, we use $[t[i]]$ to denote it. In the above example, the quasi-identifiers of the first sample are anonymized as $[20, 30] \times [F] \times [345000, 345999]$. A straightforward solution is to iterate all possible K-Anonymity strategies and evaluate each strategy based on the samples. The one with maximal estimated accuracy is adopted. However, due to the exponential number of strategies and high crowdsourcing cost, the simple strategy is impractical. Therefore, we adopt a heuristic approach by combining the sample-based feedbacks and the multidimensional K-Anonymity approach [8].

4.2 Cost Model for Partitioning

In the multidimensional K-Anonymity approach, each dimension is partitioned iteratively, until no allowable cut (cut that still guarantees the K-Anonymity property) is available. However, for each dimension, we may have multiple cut options. Figure 6 illustrates the idea for the 2-dimensional space. In the following discussion, we will use Figure 6 as our example. But the techniques can be applied to the more general case as well. Suppose we want to generate 2-Anonymity results. Line X_1 and X_2 represent two possible cut strategies for the A attribute. X_1 generates two partitions, $S_1 = [a_0, a_1) \times [b_0, b_1]$ and $S_2 \cup S_3 = [a_1, a_3] \times [b_0, b_1]$, while X_2 generates

TABLE 4
CV Data With Samples

ID	Age	Gender	Zipcode	Education	Workclass	Married	Children	Income($)
1000	25	F	345210	3	3	S	0	17,287
1001	23	F	345010	3	5	S	0	17,287
1002	28	F	345055	2	2	M	1	10,057
1003	31	M	333239	4	8	S	0	22,308
1004	35	F	333123	3	1	D	2	10,483
1005	43	M	333120	5	4	M	3	38,218
1006	38	F	333460	2	4	M	3	10,257
1007	40	M	333010	2	3	M	2	10,483
1008	35	F	333125	4	5	D	1	21,387

TABLE 5
Anonymized CV Data With Samples

ID	Age	Gender	Zipcode	Education	Workclass	Married	Children	Income($)
1000	[20-30]	F	[345xxx]	3	3	S	0	17,287
1001	[20-30]	F	[345xxx]	3	5	S	0	17,287
1002	[20-30]	F	[345xxx]	2	2	M	1	10,057
1003	[30-45]	M	[333xxx]	4	8	S	0	22,308
1004	[30-45]	F	[333xxx]	3	1	D	2	10,483
1005	[30-45]	M	[333xxx]	5	4	M	3	38,218
1006	[30-45]	F	[333xxx]	2	4	M	3	10,257
1007	[30-45]	M	[333xxx]	2	3	M	2	10,483
1008	[30-45]	F	[333xxx]	4	5	D	1	21,387

another two partitions, $S_1 \cup S_2 = [a_0, a_2) \times [b_0, b_1]$ and $S_3 = [a_2, a_3] \times [b_0, b_1]$. In [8], the cut that balances the size of generated partitions is selected. But in our case, we need to select the partitions that maximize the accuracy of the crowdsourcing.

In particular, suppose the possible answer set is R, the accuracy of partition S_1 is estimated as:

$$Acc(S_1) = \sum_{\forall r \in R} P_{S_1}(t \to r, \theta(t) \to r) \quad (9)$$

Given a sample t in S_1, $P_{S_1}((t \to r) \wedge (\theta(t) \to r))$ denotes the probability that t leads to the answer r in our ground-truth result and its crowdsourcing result is also r for the anonymity strategy $\theta(t)$. $Acc(S_1)$ can be computed via the ground-truth results and the crowdsourcing feedbacks. Similarly, we can compute $Acc(S_2 \cup S_3)$, $Acc(S_1 \cup S_2)$ and $Acc(S_3)$. To compare the two partitions (X_1 and X_2), we need to compute:

$$\delta = Acc(S_1) + Acc(S_2 \cup S_3) - Acc(S_1 \cup S_2) - Acc(S_3) \quad (10)$$

The first two terms estimate the K-anonymity accuracy by partitioning the space via X_1, while the second two terms estimate the partitioning accuracy of X_2. If $\delta \geq 0$, X_1 generates a better anonymity result. Otherwise, X_2 is better.

It is challenging to precisely estimate the value of δ. However, note that as δ is only applied to guide our partitioning algorithm, an approximation is good enough. We first transform Equation 9 into

$$\begin{aligned} Acc(S_1) \quad = \quad & \sum_{\forall r \in R} (P_{S_1}(t \to r) P_{S_1}(\theta(t) \to r) \\ & + Cov_{S_1}(t \to r, \theta(t) \to r)) \end{aligned} \quad (11)$$

where $P_{S_1}(t \to r)$ is the probability that the sample in S_1 has r as its ground-truth result, $P_{S_1}(\theta(t) \to r)$ denotes the probability that the human workers return r as the result given the anonymity form of $\theta(t)$, and $Cov_{S_1}(t \to r, \theta(t) \to r)$ is the correlation between the tuple's true result and the crowdsourcing result in S_1. For simplicity, $Cov_{S_1}(t \to r, \theta(t) \to r)$ is estimated through all available samples and thus is set to a constant. Combining Equations 10 and 11, we have:

$$\begin{aligned} \delta \quad = \quad & \sum_{\forall r \in R} (P_{S_1}(t \to r) P_{S_1}(\theta(t) \to r) \\ & + P_{S_2 \cup S_3}(t \to r) P_{S_2 \cup S_3}(\theta(t) \to r) \\ & - P_{S_1 \cup S_2}(t \to r) P_{S_1 \cup S_2}(\theta(t) \to r) \\ & - P_{S_3}(t \to r) P_{S_3}(\theta(t) \to r)) \end{aligned}$$

When computing the $P_{S_2 \cup S_3}(t \to r) P(\theta(t) \to r)$, we have $\theta(t) = [a_1, a_3] \times [t[1]]$. Without prior knowledge, the human workers assume that the values of the first attribute are uniformly distributed. Therefore, the first value of t lies in $[a_1, a_2]$ and $[a_2, a_3]$ with probabilities of $\alpha_1 = \frac{a_2 - a_1}{a_3 - a_1}$ and $\alpha_2 = \frac{a_3 - a_2}{a_3 - a_1}$. We then estimate $P(\theta(t) \to r)$ as:

$$\alpha_1 P_{S_2}([a_1, a_2][t[1]] \to r) + \alpha_2 P_{S_3}([a_2, a_3][t[1]] \to r) \quad (12)$$

Similarly, we can compute $P(\theta(t) \to r)$ for $theta(t) = [a_0, a_2] \times [t[1]]$. Let $\beta_1 = \frac{a_1 - a_0}{a_2 - a_0}$ and $\beta_2 = \frac{a_2 - a_1}{a_2 - a_0}$. The probability is calculated as:

$$\beta_1 P_{S_1}([a_0, a_1][t[1]] \to r) + \beta_2 P_{S_2}([a_1, a_2][t[1]] \to r) \quad (13)$$

Combing Equation 10 to 13, we get the final estimation for δ. Let x_r, y_r and z_r denote $P_{s_1}([a_0, a_1] \times [t[1]] \to r)$, $P_{s_2}([a_1, a_2] \times [t[1]] \to r)$ and $P_{s_3}([a_2, a_3] \times [t[1]] \to r)$,

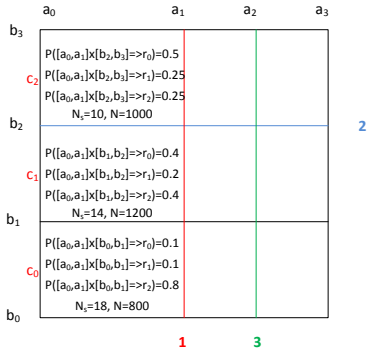


Fig. 7. Feedback Histogram

respectively. We have:

$$\begin{aligned}\delta \quad = \quad & \sum_{\forall r \in R} (x_r(P_{s_1}(t \to r) - \beta_1 P_{s_1 \cup s_2}(t \to r)) \\ & + y_r(\alpha_1 P_{s_2 \cup s_3}(t \to r) - \beta_2 P_{s_1 \cup s_2}(t \to r)) \\ & + z_r(\alpha_2 P_{s_2 \cup s_3}(t \to r) - P_{s_3}(t \to r))) \qquad (14)\end{aligned}$$

Except x_r, y_r and z_r which are estimated from the crowdsourcing feedbacks, all the rest can be computed via the ground-truth results of the samples. To find the optimal cut, we need to maximize δ. However, given limited budget for crowdsourcing, we cannot compute the three parameters for all possible cuts. Therefore, we propose a feedback histogram.

4.3 Feedback Histogram

In the feedback histogram, we pre-generate some cuts and anonymize the samples based on the cuts. The results are published to the crowdsourcing platform to get the users' feedbacks. The feedback is then used to estimate the three parameters, x_r, y_r and z_r.

Figure 7 illustrates the idea of feedback histogram. Each cell of the histogram keeps two types of information. First, it records the crowdsourcing results for all possible results. Formally, for a cell $c_i = [l_1, u_1] \times [l_2, u_2] \times ...[l_m, u_m]$ and the result set R, we will keep the feedback of $P_{c_i}([l_1, u_1] \times [l_2, u_2] \times ...[l_m, u_m] \to r_j)$ for all $r_j \in R$. $P_{c_i}([l_1, u_1] \times [l_2, u_2] \times ...[l_m, u_m] \to r_j)$ is estimated using the samples in the cell and their crowdsourcing answers. Second, we also maintain the number of samples (N_s) and the number of total tuples (N) in the cell.

The histogram cells can be combined using Equation 12. For example, the three cell of the first column in Figure 7 can be used to estimate $P_{c_0 \cup c_1 \cup c_2}([a_0, a_1] \times [b_0, b_3] \to r_0)$:

$$0.1 \times \frac{b_1 - b_0}{b_3 - b_0} + 0.4 \times \frac{b_2 - b_1}{b_3 - b_0} + 0.5 \times \frac{b_3 - b_2}{b_3 - b_0}$$

In the partitioning process, we can exploit the histograms to compute Equation 14. After iterating all possible cuts, the optimal one is selected. In this strategy, the granularity of the histogram affects the performance of the partitioning. As a fine-grained histogram incurs too much overhead, we adopt a greedy-based approach to build the histogram.

In particular, we cut the space iteratively by different dimensions. Each cut will result in a set of new cells. The samples in those cells are then anonymized based on the cell ranges. We publish the anonymized samples as a crowdsourcing job to collect the feedbacks for the cells. If we generate a job for each cell, given limited budget, we can only maintain a fixed number of cells.

The ground-truth results are correlated to the crowdsourcing results. Therefore, to build a good histogram, we use the ground-truth results of the samples to partition the space. For each cell, we have the following observation: $V = (v_0, v_1, ..., v_n)$, where v_i is the number of samples that return r_i as the result. The variance of the observation is calculated as:

$$var(V) = \frac{\sum_{i=0}^{n} v_i^2}{n+1} - (\frac{\sum_{i=0}^{n} v_i}{n+1})^2$$

The variance indicates how skewness the result distribution is in the cell. When only one v_i has the none-zero value, the variance reaches its maximal value. This is also our optimal case for K-anonymity. Namely, all tuples in the cell are expected to generate the same result.

Algorithm 1 `BuildHistogram(double[] space, Set samples, int cellLimit)`

```
1: i = 0;
2: Vector[] cut = new Vector[space.length];
3: while existsCut(space, cut, cellLimit) do
4:     d=i % space.length
5:     Set sorted = sort(samples, d)
6:     max=0
7:     for j=0 to sorted.size() do
8:         List cell = getAllCells(sorted.get(j))
9:         if cell.size()≤ cellLimit then
10:            double variance = computeVariance(cell)
11:            if variance > max then
12:                idx = j, max = variance
13:    cut[d].add(idx)
14:    i++
15: List cell = getAllCells(sorted.get(j))
16: publishJobs(cell)
```

Algorithm 1 summarizes the process. If there is available allowable cut for samples and the total cell number is less than the predefined threshold (line 3), we will iteratively partition the space by different dimensions (line 4-14). The definition of allowable cut is extended from [8]. For each cut, we first sort the samples by the dth dimension (line 5). Then, every sample's value in the dth dimension is used to partition the space. The cut with maximal variances is selected (line 7-12). Finally, we publish a job for each histogram cell to collect the feedback (line 15-16).

Definition 3: **Allowable Cut for Samples**
In d-dimensional space, a cut perpendicular to the ith axis is allowable, if there are at least K samples in the newly generated cells.

4.4 Adaptive Partitioning

Finally, we present the idea of our adaptive partitioning strategy in Algorithm 2. The process is split into two

Algorithm 2 `Partition(double[] space, Histogram H)`

```
1: Set P = new Set(space);
2: while isSuperSetOfHistogram(P, H) do
3:     Partition p = selectWorstPartition(P)
4:     Set newpartition = getBestPartition(p, H)
5:     if newpartition ≠ null then
6:         P.replaceWithNewCells(newpartition)
7: Set result = new Set()
8: for i = 0 to P.size do
9:     result.add(MondrianAlgorithm(P.get(i)))
10: return result
```

phases. In the first phase, the space is partitioned by the feedback histogram (line 2-6). In particular, the partition results are first initialized as the whole space (line 1). If there is a partition that contains multiple histogram cells, we can apply the feedbacks to further partition it (line 2). In the partitioning process, we estimate the accuracy of existing partitions and select the one with worst accuracy (line 3). Based on the Equation 14, we find the optimal partition for each dimension and the best partitioning strategy among all dimensions is adopted (line 4). The old partition is then replaced by the newly generated one (line 6).

In the second phase, the partial results of the first phase are further partitioned by applying the conventional Mondrian Algorithm [8] (line 7-9). All the generated partitions are used to guide the anonymity process.

5 Experiments

5.1 Experimental Settings

To evaluate the performance of our proposed approach, we use three datasets. Two real datasets, the US census dataset in 1990 and the IPUMS census dataset, are obtained from the UC Irvine Machine Learning Repository[5]. We configure the two datasets similarly as the experiments reported in [8]. E.g., using eight regular attributes, removing tuples with missing values, and selecting the attribute ranges which contain highly dense tuples. The resulting US dataset contains 10,000 records[6] and the IPUMS dataset contains 4,178 records. For the partitioning experiments, we impose an intuitive ordering on each attribute, and eliminated all hierarchical constraints for both approaches.

Another dataset is a synthetic dataset, derived from the US census dataset in 1990. The data space is partitioned into regions based on the ground-truth results. Tuples in the same region must have the same ground-truth values. The unsatisfied tuples are considered as noises and discarded. The synthetic dataset has 10,000 tuples and denotes the idea case, where the optimal K-anonymity approach can maximize the crowdsourcing accuracy.

Our target application is the recommendation service. Given a person's profile, we ask the human workers in Amazon AMT to recommend a job. In our experiments, we use the five categories of occupations from the census dataset as our ground-truth results for the people's jobs. The categories are described in Table 6. We list some descriptions for the human workers to understand the requirement for different occupations. Note that even without anonymity, the crowdsourcing answers may be different from the ground-truth ones. However, the accuracy model of crowdsourcing is beyond the scope of this paper. We only focus on the effect of anonymity. Our metric is the accuracy ratio, which is defined as:

$$r = \frac{\sum_{i=0}^{n} f(g(t_i), g'(t_i))}{n+1}$$

where t_i is a tuple in our dataset, $g(t_i)$ and $g'(t_i)$ denote the crowdsourcing answers of t_i before and after anonymity respectively, and $f(x, y)$ returns 1 or 0 depending on whether x equals to y.

In our model, we use the samples to build the feedback histogram. The samples are randomly selected from the datasets, and we tag the ground-truth result for each sample using the original value of the occupation attribute. In the diagram, we use **FKA** (Feedback K-Anonymity) to denote our approach. For comparison, we also implement the original **Mondrian** Algorithm [8]. Table 7 lists the experiment parameters and their corresponding ranges. The default value is marked in bold font.

Based on the five occupation categories, we summarize the detailed statistics of the real datasets respectively in Tables 8 and 9. The statistics show the average value of each attribute in the datasets. For example, the average age in *CAT 0* of the US dataset is 43 and the average value of gender is 0.3 that means 70% are males. The two datasets demonstrate different data distributions. In the IPUMS dataset, all categories have similar average values, except for the gender attribute, whereas in the US dataset, we can clearly distinguish each category from the others via the average values. This difference leads to different experiment results as discussed below.

TABLE 6
Occupation Labels

CAT ID	Occupation categories
CAT 0	Managerial and professional specialty occupations
CAT 1	Technical, officical, sales, and support occupations
CAT 2	Service occupations
CAT 3	Repair occupations
CAT 4	Operators, fabricators, and labors

TABLE 7
Parameter settings

Parameter	Range
K	50,**100**,150,200,300
Cell number	100,200,300,**400**
Sample ratio	5,**10**,15,20,30
Attribute number	2,**3**,4,5
Result number	2,3,4,**5**

5.2 Accuracy of Varied K

We first study the effect of K in the anonymity process. Parameter K represents the privacy requirement, but a larger

5. http://archive.ics.uci.edu/ml/datasets.html

6. We select limited number of records due to the high monetary cost of crowdsourcing.

TABLE 8
Statistics in US dataset

Occup.	Age	Gender	Education	Workclass	Income($)
CAT 0	43	0.3	13.3	2.3	38,218
CAT 1	33	0.5	11	1.9	22,308
CAT 2	25	0.5	9.5	1.6	10,483
CAT 3	38	0.1	9.6	1.6	17,287
CAT 4	26	0.4	9.1	1.2	10,057

TABLE 9
Statistics in IPUMS dataset

Occup.	Age	Gender	Education	Workclass	Income($)
CAT 0	38	0.4	8.8	1.8	22,673
CAT 1	42	0.1	8.5	1.8	28,376
CAT 2	38	0.5	7.4	1.9	18,680
CAT 3	38	0.03	6.3	1.8	19,986
CAT 4	36	0.3	5.4	1.9	15,314

K always causes a lower accuracy for the crowdsourcing results. In the experiment, we fix the values of other parameters as the default values shown in Table 7. For comparison purpose, besides **Mondrian**, we also include the results of the **Datafly** [13] and **Incognito** [14]. Interestingly, we observe very different results for the three datasets.

In Figure 8, although the accuracies of both approaches drop for a larger K, our **FKA** performs much better than the other approaches. This is because the synthetic dataset represents the idea case, where the histogram can effectively catch the distribution of crowdsourcing results. On the contrary, in Figure 10, **FKA** only obtains a slightly higher accuracy than the other anonymity approaches. It is because in the IPUMS dataset, the tuples of different categories follow the similar distribution as shown in Table 9. The results of US dataset lie between the synthetic dataset and the IPUMS dataset. It represents a more realistic dataset, where people can be grouped into different categories, but different categories do have some overlaps.

Except the synthetic dataset, **Mondrian** performs better than **Datafly** and **Incognito**. **Datafly** achieves a good performance for the synthetic dataset, as data follow uniform distribution in the synthetic dataset and **Datafly** generates the anonymity groups uniformly. In the remaining experiments, we use the results of **Mondrian**, as it performs better for the real datasets.

5.3 Accuracy of Varied Cell Number

The key technique in our **FKA** approach is to exploit the feedback histogram to evaluate the quality of different anonymity approaches. The histogram guides the anonymity algorithm to partition the space adaptively. If the histogram cannot provide an accurate description for the answer distribution, our approach may fail to generate an anonymity strategy, which is crowdsourcing friendly. In these experiments, we study how the histogram affects the performance of crowdsourcing.

In Figure 11 to 13, we vary the number of histogram cells and test the result accuracy for different datasets. More histogram cells lead to a better understanding of the answer distribution, but it also incurs high overhead for the crowdsourcing, as we need to publish a job for each cell to collect the feedbacks.When the histogram cell number increases, both the synthetic dataset and the US dataset get a better accuracy. This is because the histograms for the two datasets can precisely reflect the answer distribution. On the other hand, the histogram granularity does not affect the performance of IPUMS dataset, as in this dataset, the crowdsourcing answer is not only determined by the quasi-identifiers. The values of sensitive attributes are more correlated to the crowdsourcing answers. Another observation is that the when the histogram is precise enough, increasing the histogram cells will not benefit the anonymity strategy any more.

5.4 Accuracy of Different Sample Ratios

Besides the cell number, another parameter that affects the histogram quality is the sample size. We randomly select the samples from the datasets and tag them with ground-truth results. The samples are then published into the crowdsourcing platform to collect the feedbacks. Similar to the cell number, more samples lead to a better histogram. In Figure 14 to Figure 16, the sample rate varies from 5% to 20%. For the IPUMS dataset, we find that its performance is not affected much by the samples. For the other two datasets, the accuracy improves significantly for a larger sample size. This, in fact, is consistent with the observation in the last experiment on the cell number. A precise histogram can help the anonymity algorithm find a better anonymity strategy.

Figure 17 shows the monetary costs in the sampling process. We group 100 sample questions into one HIT (Human Intelligent Task) and assign each HIT to 5 workers. We pay each worker \$0.2 for a HIT and 30% management fee for the AMT. More budgets lead to more feedback answers for **FKA** and thus a better anonymity result.

5.5 Accuracy of Varied Attribute Number

The anonymized attributes cause information loss. When publishing the data for crowdsourcing jobs, the human workers must infer the results based on the incomplete information. In these experiments, we study how the anonymized attributes affect the accuracy of crowdsourcing.

In Figure 18 and 19, we vary the number of attributes used in anonymity processing. In particular, besides the quasi-identifiers, we also anonymize the sensitive attributes, such as *education* and *income*. In the anonymity process, we always adopt the same K value. Namely, the privacy level is not changed. We observe that the human worker's performance degrades, when the number of anonymized attributes increases. This indicates that anonymized attributes generate ambiguous information for the workers and may confuse them for their decisions. To provide a high accuracy crowdsourcing results, we should anonymize as fewer attributes as possible.

The performance gap between **FKA** and **Mondrian** becomes larger as well. This might be caused by the different partitioning strategies. The **FKA** approach preserves some

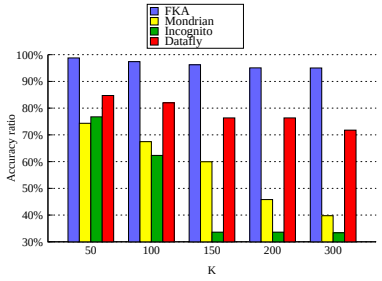


Fig. 8. Accuracy of Varied K (Synthetic dataset)

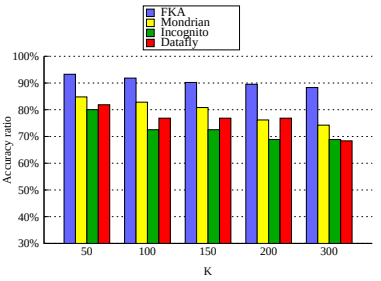


Fig. 9. Accuracy of Varied K (US dataset)

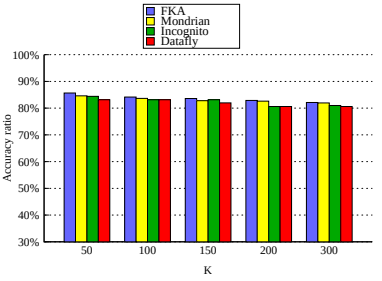


Fig. 10. Accuracy of Varied K (IPUMS dataset)

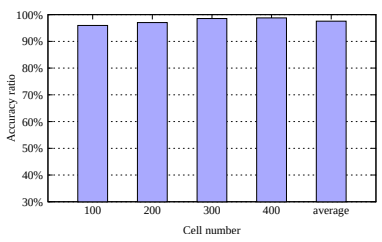


Fig. 11. Accuracy of Varied Cell Number (Synthetic dataset)

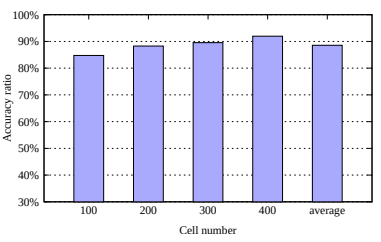


Fig. 12. Accuracy of Varied Cell Number (US dataset)

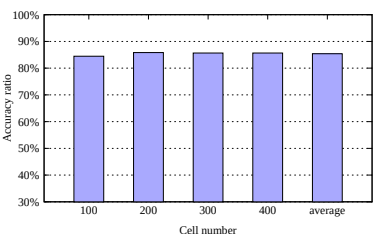


Fig. 13. Accuracy of Varied Cell Number (IPUMS dataset)

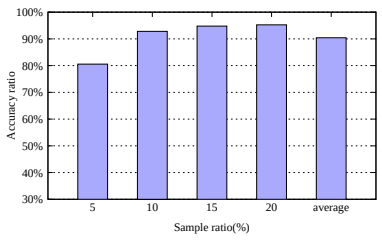


Fig. 14. Effect of Sample Ratios (Synthetic dataset)

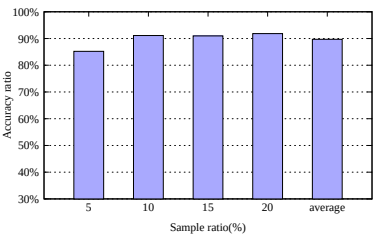


Fig. 15. Effect of Sample Ratios (US dataset)

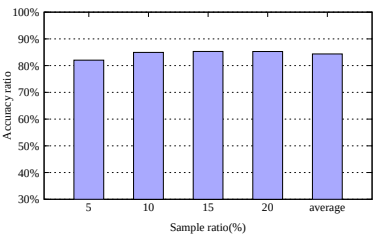


Fig. 16. Effect of Sample Ratios (IPUMS dataset)

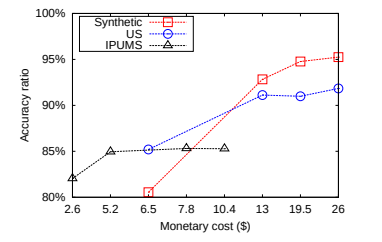


Fig. 17. Monetary Cost for Sampling

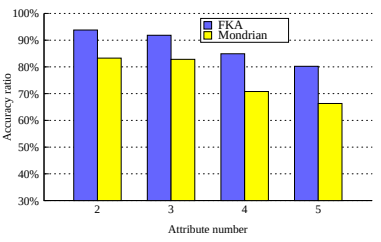


Fig. 18. Effect of Varied Attributes (US dataset)

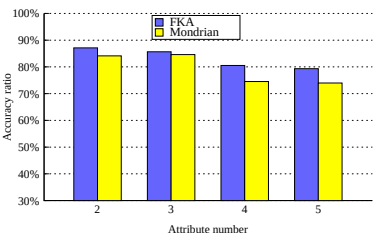


Fig. 19. Effect of Varied Attributes (IPUMS dataset)

distribution features for the anonymized attributes by the histogram, while the **Mondrian** blindly groups the tuples. We discard the diagram of the synthetic data, as in the idea case, the accuracy is not affected much by the attribute number.

Another observation is that if an attribute, which has low correlation with the ground-truth values, is selected in the anonymity process, the result accuracy will not be affected significantly. It suggests that if we know the correlations between the crowdsourcing result and different attributes, we can select the attributes with lower correlation to anonymize. This strategy can provide a higher accuracy in the crowdsourcing.

5.6 Accuracy vs Result Number

Based on our analysis in Section III, the accuracy lower bound of any K-Anonymity approach is determined by the result distribution. If the human operator only has few valid results and the result distribution is skewed, we can get a good lower bound. Otherwise, the estimated lower bound indicates that some K-Anonymity strategies can lead to extremely low accuracies in the crowdsourcing. In this experiment, we filter the datasets by discarding the tuples of some occupations. In this way, the datasets only include tuples of limited ground-truth results. Correspondingly, when designing the crowdsourcing job, only the involved jobs are used as the selections for the human workers. Figure 20 and 21 show the accuracies for the two datasets. In both datasets, when increasing the number of possible results, the accuracy of crowdsourcing drops, which verifies our estimation for the lower bound. The **FKA** performs better than the **Mondrian**, as it can catch the result distribution via the feedback histograms. The diagram of the synthetic data is discarded for the same reason as the last experiment

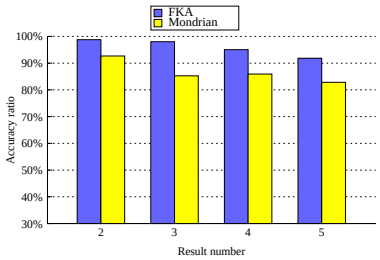


Fig. 20. Lower Bound Test (US dataset)

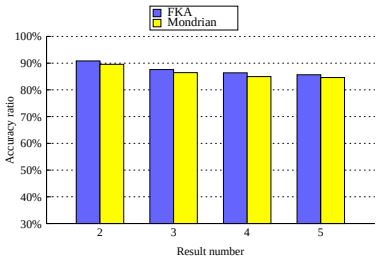


Fig. 21. Lower Bound Test (IPUMS dataset)

on the attribute number.

6 Related Work

6.1 Crowdsourcing System

Instead of designing sophisticated computer algorithms, the crowdsourcing system leverages human workers to solve the problems with rich semantics. Example applications include image annotation [15], natural language processing [16][17] and sentiment analysis [18]. In crowdsourcing systems, the quality of the results relies on the human workers. Mason and Watts [19] studied the effect of compensation on performance. They found that increased incentives increase the number of returned answers, but not the quality of the answers. In [20], an economic model is proposed to catch the relationship between the workers' wages and their working hours. The model found that the wage follows the normal distribution in log scale approximately. The crowdsourcing system normally assigns the same job to multiple human workers, who may return different answers. To resolve the conflictions between human workers, different models [4][21] are introduced, which are all based on the probability theory.

In database system, crowdsourcing techniques are embedded as a specific database operator [1][2], which collaborates with other database operators to process queries. To enable the users to express their crowdsourcing tasks, a new query language, hQuery [22], is introduced by extending the original SQL. Basic database algorithms, such as sort and join, can be redesigned with the help of human computations [23]. Some more complicated algorithms, such as graph search [24], also benefit from the crowdsourcing techniques.

6.2 K-Anonymity

To protect the privacy of individuals, the outsourced data are anonymized before publishing. The most popular technique is K-Anonymity [5], which guarantees that each person contained in the release cannot be distinguished from at least k-1 individuals. In K-Anonymity, the basic operations are generalization and suppression. Generalization can be conducted on the attribute level [9] and cell level [25], while suppression can be applied to the tuple level [10] and cell level [26]. Besides the privacy concern, the K-Anonymity approach tries to reduce the information loss as much as possible. But as shown in many studies [26][27], finding the optimal K-Anonymity strategy is a NP-hard problem. Most solutions adopt the greedy-based heuristic approaches. In this paper, we tailor a new heuristic approach based on the workers' feedbacks for the crowdsourcing system. Our approach exploits the human worker's computation to improve the quality of anonymity.

Sometimes, K-Anonymity is not enough for guaranteeing the privacy and hence, l-diversity is proposed to address the problem [28]. Our solution is orthogonal to the l-diversity and in fact, we can enforce the l-diversity in our feedback algorithm. We will study the effect of the l-diversity to the crowdsourcing accuracy in our future work.

7 Conclusion

To integrate the crowdsourcing techniques into the database engine, we must address the privacy concern, as each crowdsourcing job requires us to publish some sensitive data to the anonymous human workers. In this paper, we study how to guarantee the data privacy in the crowdsourcing scenario. A probability-based matrix model is introduced to estimate the lower bound and upper bound of the crowdsourcing accuracy for the anonymized data. The model shows that K-Anonymity approach needs to solve the trade-off between the privacy and the accuracy. Different from the conventional K-Anonymity approaches, the anonymity scheme for the crowdsourcing system must maximize the expected accuracy of crowdsourcing. Therefore, we propose a novel K-Anonymity approach, which exploits the crowdsourcing answers from the human workers progressively. In particular, we build a feedback histogram by repeatedly submitting the crowdsourcing jobs to collect the human's opinions. We then adaptively adjust the anonymity approach to maximize the estimated accuracy. Experiments on three different datasets show that our solution can maintain high accuracy results for the crowdsourcing jobs.

Acknowledgment

The work of Sai Wu is supported by National Natural Science Foundation of China (Grant No. 61202047). The work of Sai Wu, Xiaoli Wang and Anthony K.H. Tung was carried out at the SeSaMe Centre. It is supported by the Singapore NRF under its IRC@SG Funding Initiative and administered by the IDMPO.

References

[1] A. Feng, M. J. Franklin, D. Kossmann, T. Kraska, S. Madden, S. Ramesh, A. Wang, and R. Xin, "Crowddb: Query processing with the vldb crowd," *PVLDB*, vol. 4, no. 12, pp. 1387–1390, 2011.

[2] A. Marcus, E. Wu, S. Madden, and R. C. Miller, "Crowdsourced databases: Query processing with people," in *CIDR*, 2011, pp. 211–214.

[3] A. Marcus, E. Wu, D. R. Karger, S. Madden, and R. C. Miller, "Demonstration of qurk: a query processor for humanoperators," 2011, pp. 1315–1318.

[4] X. Liu, M. Lu, B. C. Ooi, Y. Shen, S. Wu, and M. Zhang, "Cdas: A crowdsourcing data analytics system," vol. 5, no. 10, pp. 1040–1051, 2012.

[5] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 557–570, 2002.

[6] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Incognito: Efficient full-domain k-anonymity," in *SIGMOD Conference*, 2005, pp. 49–60.

[7] A. Meyerson and R. Williams, "On the complexity of optimal k-anonymity," in *PODS*, 2004, pp. 223–228.

[8] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Mondrian multidimensional k-anonymity," in *ICDE*, 2006, p. 25.

[9] R. J. B. Jr. and R. Agrawal, "Data privacy through optimal k-anonymization," in *ICDE*, 2005, pp. 217–228.

[10] V. S. Iyengar, "Transforming data to satisfy privacy constraints," in *KDD*, 2002, pp. 279–288.

[11] V. Ciriani, S. D. C. di Vimercati, S. Foresti, and P. Samarati, "*k*-anonymity," in *Secure Data Management in Decentralized Systems*, 2007, pp. 323–353.

[12] A. Arasu, R. Kaushik, and J. Li, "Datasynth: Generating synthetic data using declarative constraints," *PVLDB*, vol. 4, no. 12, pp. 1418–1421, 2011.

[13] L. Sweeney and L. Sweeney, "Achieving k-anonymity privacy protection using generalization and suppression," *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, vol. 10, p. 2002, 2002.

[14] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Incognito: efficient full-domain k-anonymity," in *SIGMOD*, 2005, pp. 49–60.

[15] A. Sorokin and D. Forsyth, "Utility data annotation with amazon mechanical turk," in *First IEEE Workshop on Internet Vision at CVPR*, 2008.

[16] J. Ledlie, B. Odero, E. Minkov, I. Kiss, and J. Polifroni, "Crowd translator: on building localized speech recognizers through micropayments," *Operating Systems Review*, vol. 43, no. 4, pp. 84–89, 2009.

[17] I. McGraw, "Crowd-supervised training of spoken language systems," *PhD Thesis, MIT CSAIL*, 2012.

[18] C. Akkaya, A. Conrad, J. Wiebe, and R. Mihalcea, "Amazon mechanical turk for subjectivity word sense disambiguation," in *NAACL HLT*, 2010, pp. 195–203.

[19] W. Mason and D. J. Watts, "Financial incentives and the "performance of crowds"," *SIGKDD Explor. Newsl.*, vol. 11, no. 2, pp. 100–108, May 2010.

[20] J. J. Horton and L. B. Chilton, "The labor economics of paid crowdsourcing," in *Proceedings of the 11th ACM conference on Electronic commerce*, ser. EC '10, 2010, pp. 209–218.

[21] S. Guo, A. Parameswaran, and H. Garcia-Molina, "So who won?: dynamic max discovery with the crowd," in *SIGMOD*, 2012, pp. 385–396.

[22] A. G. Parameswaran and N. Polyzotis, "Answering queries using humans, algorithms and databases," in *CIDR*, 2011, pp. 160–166.

[23] A. Marcus, E. Wu, D. Karger, S. Madden, and R. Miller, "Human-powered sorts and joins," *PVLDB*, vol. 5, no. 1, pp. 13–24, Sep. 2011.

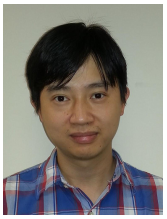
[24] A. G. Parameswaran, A. D. Sarma, H. Garcia-Molina, N. Polyzotis, and J. Widom, "Human-assisted graph search: it's okay to ask questions," *PVLDB*, vol. 4, no. 5, pp. 267–278, 2011.

[25] H. Park and K. Shim, "Approximate algorithms with generalizing attribute values for k-anonymity," *Inf. Syst.*, vol. 35, no. 8, pp. 933–955, 2010.

[26] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu, "Anonymizing tables," in *ICDT*, 2005, pp. 246–258.

[27] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, and D. Thomas, "Approximation algorithms for k-anonymity," *Journal of Privacy Technology*, 2005.

[28] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "L-diversity: Privacy beyond k-anonymity," *ACM Trans. Knowl. Discov. Data*, vol. 1, no. 1, Mar. 2007.



Sai Wu received his Ph.D. degree from National University of Singapore (NUS) in 2011 and now is an assistant professor at College of Computer Science, Zhejiang University. His research interests include P2P systems, distributed database, cloud systems and indexing techniques. He has served as a Program Committee member for VLDB, ICDE and CIKM.

Xiaoli Wang is current a Ph.D. student and Research Assistant in School of Computing, National University of Singapore. Her research interests are mainly in indexing and query processing on the complex structure, such as sequence, tree and graph.

Sheng Wang is current a Ph.D. student in School of Computing, National University of Singapore. His research interests are mainly in Cloud databases, log-structured file systems and scientific databases.

Zhenjie Zhang is a Research Scientist at ADSC. He received his Ph.D. from the School of Computing, National University of Singapore, in 2010. His research interests cover a variety of different topics, including clustering analysis, non-metric indexing and data privacy. He has published more than 20 research papers in database and data mining venues, including SIGMOD, VLDB, and ICML. He has served as a Program Committee member for WWW 2010, VLDB 2010 and KDD 2010.

Anthony K.H. Tung received the PhD degree in computer sciences from Simon Fraser University (SFU) in 2001. He is currently an Associate Professor in the Department of Computer Science, National University of Singapore. His research interests involve various aspects of databases and data mining (KDD) including buffer management, frequent pattern discovery, spatial clustering, outlier detection, and classification analysis.