Dependency-Preserving Generalization for Data Publishing

Dependency-Preserving Generalization for Data Publishing

By Harika Gorla, B.E.

A Thesis Submitted to the Department of Computing and Software and the School of Graduate Studies of McMaster University in the Partial Fulfillment of the Requirements for the Degree of Master of Science

McMaster University

Master of Science (2019)

Hamilton, Ontario (Computer Science)

TITLE: Dependency-Preserving Generalization for Data Publishing

AUTHOR: Harika Gorla, B.E

SUPERVISOR: Dr.Fei Chiang

Abstract

A vast amount of microdata about individuals and entities are collected and published for different purposes, such as demographic and public health research. However, data in its original form contains sensitive information about the individuals and publishing such data violates individuals privacy. To resolve this problem, *privacy preserving data publishing* (PPDP) proposes many approaches to generate a public version of data that is practically useful and individual's privacy is protected. k-anonymity has emerged as an efficient approach to protect the individual's privacy by generalizing and/or suppressing portions of the data to make individuals indistinguishable in the released data.

Existing generalization algorithms focus on minimizing the information loss during generalization of attribute values. Any data dependencies defined over the data may be lost during this generalization step. A data dependency is a formal concept which is used to describe patterns in data. These patterns are employed during data analysis and data cleaning. A typical data dependency in a database is a *Functional Dependency (FD)*: $X \rightarrow Y$ expresses that the values of attribute X uniquely determine the values of attribute Y e.g. *postal code* $\rightarrow$ *province* means the value of *postal code* uniquely determines the value of *province*.

In this thesis, we study the problem of publishing data with two objectives. First, protecting the identity of the individuals in the published data through k-anonymity. Second, to provide high utility by preserving the instances of the data dependencies in the released data. We introduce dependency loss as a penalty measure for the anonymized public data. We define and study the problem of dependency-preserving generalization for finding a public database instance that guarantees privacy through k-anonymity and has minimum dependency loss. We present two clustering-based generalization algorithms that find such a database instance and we run experiments to show the comparable performance and improved utility in preserving data dependencies of our algorithms.

Acknowledgements

I would like to express my deepest gratitude to my supervisor Dr. Fei Chiang for the patient guidance, advice and encouragement she has provided throughout my time as her student. She directed me on the right path by providing her feedback and sharing her valuable knowledge during our technical discussions. Furthermore, I would like to thank Dr. Mostafa Milani for his support and supervision in this research.

I would also like to thank my committee members Dr. Frantisek Franek and Dr. Wenbo He for serving as my committee members. Thank you for investing time and providing valuable feedback.

It's my fortune to gratefully acknowledge my friends, Elaf Alhazmi and Zahra Ali for their support and care throughout my research tenure. Big thanks to my fellow colleagues and friends of the Data Science lab for their co-operation and support: Morteza Alipour, Zheng Zheng, Yu Huang and many others. I gained a lot from them, through their personal and scholarly interactions, suggestions at various points of my research program.

I want to express my heartfelt gratitude to my aunt, Sireesha, my uncle, Subba Reddy, my friend Marei and her family for their wonderful services all these years. They have been great support ever since I moved to Canada, and I owe them so much for their care and hospitality.

A very special word of thanks goes to my parents, Subba Reddy Gorla and Sarada Gorla for their selfless love, care, sacrifice to shape my life. Although they hardly understood what I researched on, they were willing to support any decision I made. Also, I express my thanks to my brother, Sundeep Reddy Gorla for his selfless love, care and encouragement which contributed a lot for completion of my thesis.

My heartfelt regard goes to my in-laws, Dinakar Reddy Pulimi and Sailaja Pulimi for their love and moral support. Finally, I thank my husband, Venkat Aveen Reddy Pulimi who supported me in every possible way to see the completion of my thesis. He was always around at times I thought that it is impossible to continue and helped me to keep things in perspective. I greatly value his contribution and deeply appreciate his belief in me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A number of organizations collect and publish the microdata about individuals and entities to third parties for a variety of purposes, such as demographic and public health research. However, this kind of data often contains personal details and sensitive information and publishing such data violates individual privacy. To address this challenge, *Privacy-Preserving Data Publishing* has become an active research area in which many data anonymization approaches have been proposed.

1.1 Privacy-Preserving Data Publishing

Privacy-Preserving Data Publishing (PPDP) is the problem of generating a public version of a private database while protecting the privacy of individuals and keeping the published database useful for general data analysis tasks. A relation R contains a set of attributes $A_1, \cdots, A_n$. For example, the hospital patient data in Table 2 has six rows and five attributes: age (AGE), gender (GEN), postal code

(PC), province (PRV) and diagnosis (DIAG). In PPDP, three types of attributes are considered. *Identifiers* (such as SSN, first name and last name) that uniquely identify individuals, *quasi-identifiers* (QI) (such as age, gender, zipcode, race) that can uniquely identify individuals when combined together with the external data and *sensitive attributes* contains person specific sensitive information (such as diagnosed disease, salary).

In order to protect an individual's privacy, the data publisher removes the identifiers from the data. De-identifying the data, however, provides no guarantee of anonymity. According to a study [23], approximately 87% of the United States population can be uniquely identified by joining or linking the de-identified data sets with the publicly available data sets on the basis of QI attributes. This kind of attack is known as a "*record linkage*" attack. For example, as shown in Table 1 and Table 2, an attacker is able to determine Carol's sensitive medical information by joining both tables on AGE, Gender (GEN), Postal code (PC). *k-anonymity* has emerged as an efficient anonymization approach to prevent this linkage attack.

FNAME	LNAME	AGE	GEN	PC
Andre	Brown	54	Male	T5H
Carol	Anderson	**32**	**female**	**V6J**
Beth	Hill	48	Female	B3M
Ellen	Furlan	37	Female	T9C
Dan	Johnson	52	Male	E3A
Andrew	Fuller	64	Male	B0P

TABLE 1: Voter registration data.

AGE	GEN	PC	PRV	DIAG
32	**female**	**V6J**	BC	osteoarthritis
51	male	J5B	QC	atrial flutter
39	female	J5B	QC	pain reliever
43	female	K2H	ON	seizure
54	male	R7A	MB	seizure
48	female	K2H	ON	atrial flutter

TABLE 2: Hospital patient data.

1.2 k-anonymization

The key idea of k-anonymization [23, 21] is to make an individual's data indistinguishable from $k-1$ other individual's data by substituting the specific QI attribute values with more general values and less informative according to the pre-defined attributes *value generalization hierarchies* (VGHs). The VGH of an attribute defines a tree whose leaves are the specific values that the attribute can assume and the root is the most general value. Figures 1(b)-3(b) illustrates examples of VGHs for the province (PRV), postal code (PC) and medication (MED) attributes.

Various approaches for generalization have been studied, such as global recoding [12, 13, 8, 21, 22, 24], and local recoding [14, 19, 25, 3, 16]. Global recoding anonymize the database at domain level in which an attribute's current domain is mapped to a more general domain. For example, all AGE attribute values in the database are mapped from age in years to 10-year intervals. Local recoding anonymize the database at a tuple/record level, in which some or all attributes values in a tuple are more general than the corresponding pre-mapping values. For example, {32, female, v6J} is mapped to {[30,40], female, unknown}. A general view of k-anonymity is clustering with a condition such that each cluster must have at least k tuples. In order to maximize the usefulness of the data, the tuples in each cluster must be as similar as possible. This guarantees less information loss when the tuples in a cluster are generalized to have the same QI values.

ID	PC	PRV	MED	DIAG
t_1	V6J	BC	ibuprofen	osteoarthritis
t_2	V6J	BC	ibuprofen	osteoarthritis
t_3	J5B	QC	ibuprofen	osteoarthritis
t_4	K2H	ON	ibuprofen	osteoarthritis
t_5	K2H	ON	diazepam	seizure
t_6	J5B	QC	diazepam	seizure
t_7	R7A	MB	diazepam	seizure
t_8	R7A	MB	diazepam	seizure

Table 3: Private table R.

ID	PC	PRV	MED	DIAG
r_1	V6J	BC	ibuprofen	osteoarthritis
r_2	V6J	BC	ibuprofen	osteoarthritis
r_3	[J,K,L]	Central Canada	ibuprofen	osteoarthritis
r_4	[J,K,L]	Central Canada	ibuprofen	osteoarthritis
r_5	[J,K,L]	Central Canada	diazepam	seizure
r_6	[J,K,L]	Central Canada	diazepam	seizure
r_7	R7A	MB	diazepam	seizure
r_8	R7A	MB	diazepam	seizure

Table 4: 2-anonymous public view R'.

ID	PC	PRV	MED	DIAG
m_1	V6J	BC	ibuprofen	osteoarthritis
m_2	V6J	BC	ibuprofen	osteoarthritis
m_3	J5B	QC	*	osteoarthritis
m_4	K2H	ON	*	osteoarthritis
m_5	K2H	ON	*	seizure
m_6	J5B	QC	*	seizure
m_7	R7A	MB	diazepam	seizure
m_8	R7A	MB	diazepam	seizure

Table 5: 2-anonymous public view R''.

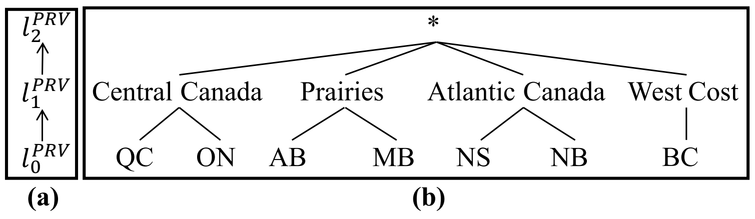


Figure 1: (a) DGH and (b) VGH of province (PRV).

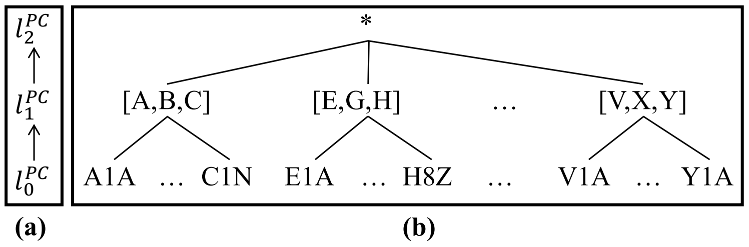


Figure 2: (a) DGH and (b) VGH of postal code (PC).

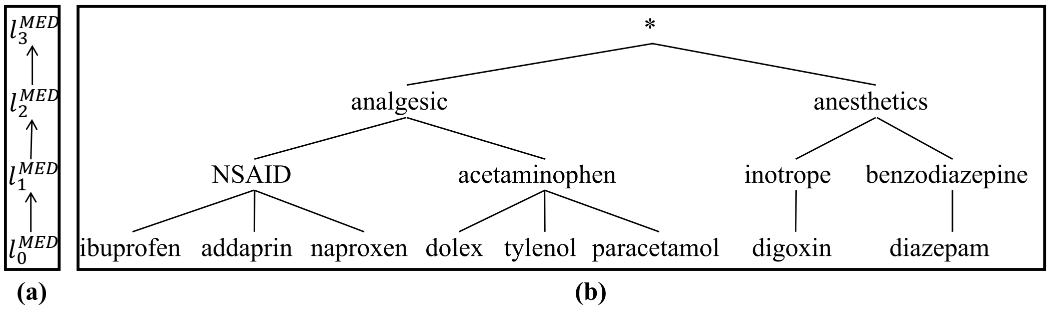


Figure 3: (a) DGH and (b) VGH of medication (MED).

Example 1.1. *Consider the de-identified private hospital patient data R, as shown in Table 3 containing the patient records with postal code* (PC), *province* (PRV), *and medication* (MED) *as* QI *and their diagnosed disease* (DIAG) *as the sensitive attribute. Relation R' in Table 4 is a 2-anonymous view of R in Table 3 since each tuple in R' is indistinguishable from at least one tuple w.r.t the QI attribute set* {PC, PRV, MED}. *A relation R might have more than one k-anonymous view by applying different methods of generalization. Relation R'' in Table 5 is also a k-anonymous view of R with $k = 2$.*

1.3 Dependency Loss Penalty

Generalization is accompanied by information loss and to evaluate the usefulness of a generalized published database, different measures are proposed in the literature [10, 25, 23, 21, 22]. These are typically general purpose penalty measures that penalize a generalized database based on the information loss incurred by the generalization process. Existing penalty measures quantify the data distortion in the entire generalized public database with respect to the data in the original specific database and do not consider the loss of data dependencies.

A database dependency is a formal concept that can be used to describe patterns in data. For instance, a functional dependency (FD): $\varphi : X \rightarrow Y$ expresses that the values of attribute X uniquely determine the values of attribute Y. These data dependencies are often used by the data recipients to perform some data analysis and data cleaning. Therefore, given an original private relation R that holds a set of data dependencies, publishing an anonymous public view R' of R that better preserves the dependencies is practically useful to the data recipients to enrich and clean their own stale and inconsistent data. Since the anonymized version of the original data is a view, duplicate records are possible.

Example 1.2. *(ex.1.1 cont.) Let* $\varphi_1 : \mathsf{PC} \rightarrow \mathsf{PRV}$ *and* $\varphi_2 : \mathsf{DIAG} \rightarrow \mathsf{MED}$ *be the two FDs that hold over relation R in Table 3 and have 6 dependency instances w.r.t both* φ_1 *and* φ_2 *:* $p_1 = (\mathsf{V6J}, \mathsf{BC})$*,* $p_2 = (\mathsf{J5B}, \mathsf{QC})$*,* $p_3 = (\mathsf{K2H}, \mathsf{ON})$*,* $p_4 = (\mathsf{R7A}, \mathsf{MB})$*,* $p_5 = (\mathsf{osteoarthritis}, \mathsf{ibuprofen})$*,* $p_6 = (\mathsf{seizure}, \mathsf{diazepam})$*. Consider a data recipient such as an external medical center or a pharmaceutical firm that owns a inconsistent subset of the original private data R as shown in Table 3. A*

2-anonymous view R'' of R in table 5 better preserves the FDs over R' in Table 4 because all the six dependency instances are preserved in R'' without any distortion. However, in R', two instances p_2 and p_3 are distorted and not completely preserved. Assume there exists a dependency instance $p_2 = (\mathsf{J5B}, \mathsf{X})$ of the FD $\varphi_1 : \mathsf{PC} \rightarrow \mathsf{PRV}$, in the data owned by the data recipient. In p_2, let X represent missing or inconsistent value. Publishing R'' in Table 5 provides a valuable commodity to the data recipient over R' in Table 4 to clean or update PRV *attribute value of p_2 in his/her data because, in R'', the dependency instance $p_2 = (\mathsf{J5B}, \mathsf{QC})$ is completely preserved. However, in R', p_2 is distorted and not completely preserved.*

The existing penalty measures do not consider data dependencies. The minimal distortion penalty measure [21, 23] assigns a penalty to an anonymous relation based on the domain level of its general values in the DGHs*. The penalty of R' is 8 and R'' is $4 \times 3 = 12$ because R' has eight general values of level 1 (four values in l_1^{PC} and four values in l_1^{PRV}) and $R^{''}$ has four general values of level 3 (l_1^{MED}) according to the generalization hierarchies in Figures 1-3. Therefore, $R^{'}$ is preferred over $R^{''}$ w.r.t minimal distortion penalty measure. In [18], the penalty of a general relation is based on the number of suppressed values ($*$). $R^{'}$ is favored over $R^{''}$ because $R^{'}$ has no suppressed values and $R^{''}$ has four suppressed values in* MED.

Therefore, we propose a penalty metric known as *Dependency Loss* (Δ) that quantifies the loss of data dependencies in a public relation by searching for the dependency instances in the public relation that are most similar to the data dependencies in the private relation. The similarity between any two instances of dependency is measured by computing the pairwise distance between the attribute values in the instances. The pairwise distance between any two values of an attribute is measured by an entropy-based distance function. If an instance from private relation appears at least once in public relation without any distortion, then the distance is zero. For example, the instance $p_4 = (\mathsf{R7A}, \mathsf{MB})$ from R appears in $R''[m_7]$ without any distortion, therefore the distance is zero. As a special case, the *Dependency Loss* of public relation is zero if all the dependency instances from the private relation appear in the public relation with no distortion. For example, the *Dependency Loss* of R'' in Table 5 is zero w.r.t R in Table 3.

1.4 Dependency Preserving Generalization

We define the dependency preserving generalization problem using the *dependency loss* measure to find a k-anonymous relation with minimal loss of dependencies. Various research works have shown that optimal anonymization is NP-hard [21, 18, 1]. In this work, we propose two simple and efficient clustering-based generalization algorithms, PAIR-ENUM and k-ASSEMBLE for dependency preserving generalization problem. A category of local recoding algorithms with high utility are clustering based generalization algorithms [25, 19, 3, 16]. The base idea is to partition the tuples in a private relation into clusters of k size and generalize the QI attribute values in each cluster to satisfy the privacy requirement. These algorithms use penalty or utility measures to guide each step of clustering and try to optimize the output against those measures. We define a utility metric that measures the quality of clustering more accurately w.r.t preserving the dependencies and use this metric to guide our algorithms towards finding the clustering that has maximal utility w.r.t dependencies.

PAIR-ENUM starts with an initial set of |R| singleton clusters, which means each cluster contains a tuple. At each iteration, PAIR-ENUM considers all the possible pairs of clusters to be merged and greedily merges a cluster pair containing a set of clusters with the highest utility. After each iteration, PAIR-ENUM seals the clusters of size k. The sealed clusters are not merged with other unsealed clusters in the next iteration since these clusters satisfy the k constraint and are published in the public relation. On the other hand, k-ASSEMBLE initializes a pool of m clusters, each of size k. A cluster in the pool is generated by randomly selecting an unsealed tuple from R and grouping with its $k - 1$ unsealed nearest neighbours in R. For each tuple in R, we find its $k - 1$ nearest neighbours using our entropy based distance measure. Among the m clusters generated in each iteration, k-ASSEMBLE seals the cluster with the highest utility and is added to the public relation since the cluster satisfies the k-constraint.

In summary, the key contributions of our work are as follows:

1. We introduce a dependency loss metric to capture the loss of dependencies via generalization.

2. We present two greedy clustering based generalization algorithms, namely PAIR-ENUM and k-ASSEMBLE that aim to minimize the dependency loss.

3. We propose optimization techniques that prune the cluster pairs and recompute the utility efficiently to improve the performance of the algorithms by 1.8× and 6.3× at a cost of 3.2% and 5.3% dependency loss.

4. An experimental evaluation that studies our algorithms comparative performance with the existing local re-coding algorithms, Top-down and k-member. We show that k-ASSEMBLE is 94% faster than k-member and the difference between Top-down and k-ASSEMBLE is about 70%. A qualitative evaluation of the dependency loss of our algorithms shows that PAIR-ENUM achieves less distortion (4.5% on an average) than k-ASSEMBLE as we vary the k value. A study on the impact of data skew on the dependency loss shows that our algorithm generated instances with less dependency loss using uniform data (6% on an average) over the original skewed data.

Chapter 2

Related Work

In this section, we briefly summarize the related research on data publishing and generalization methods.

2.1 Privacy Models

Many privacy models have been proposed to provide guarantees about the protection of an individual's privacy. These models have been developed considering different attack scenarios to the data. For example, an attacker having diverse levels of background knowledge could lead to information disclosure. Some of the well known models are k-anonymity [23, 21], *l-diversity* [17], *t-closeness* [15] and *differential privacy* [5].

Among these models, we focus on k-anonymity because it enables general purpose data publication with reasonable utility. This model is in contrast to more robust models, such as differential privacy which might restrain the usefulness of the anonymized data in order to provide more rigorous privacy guarantee. Its conceptual simplicity has made it widely discussed and adopted in a variety of domains such as healthcare [6, 11] and data mining [10, 25].

2.2 Penalty Measures

K-anonymity adopts generalization and suppression to preserve privacy. So there will be a certain degree of information loss. A specific value of an attribute can be replaced with a general value according to a given *value generalization hierarchy* (VGH). In Figure 1(b), the parent node Central Canada is more general than its child nodes QC and ON and the root node * represents the most general value of province (PRV). In order to describe quantitatively, various metrics have been proposed to measure the information loss of generalized data. A reasonable information metric is to measure the similarity between the original data and the anonymous data. According to *minimal distortion metric* (MD), a penalty is charged to each instance of a value that is generalized or suppressed in the anonymized data. For example, generalizing five instances of QC to Central Canada causes five units of distortion, and further generalizing these instances to * causes another five units of distortion.

2.2.1 Precision Metric

In [23], Sweeney defined an information theoretic metric known as *precision metric* to capture the amount of distortion in a generalized version R'. Given a

k-anonymous view $R^{'} = \{t^{'}_1, ..., t^{'}_{|R|}\}$ that generalizes original ground relation R with QIs $A_1, ..., A_n$, the *precision* of $R^{'}$ is defined as follows:

$$Prec(R^{'}) = 1 - \frac{\sum_{i=1}^{n}\sum_{j=1}^{|R|}(\frac{level(t^{'}_j[A_i])}{h^{A_i}})}{|R| \times n},$$

where h^{A_i} is the number of levels in the DGH of A_i. The precision metric is 1 when all the values in $R^{'}$ are specific/ground and it is 0 when every values in $R^{'}$ is $*$.

2.2.2 *ILoss* Metric

The *ILoss metric* [10, 25] measures the information loss incurred by replacing a ground value v' with the general value v, based on the number of ground descendant values of a general value v ($|base(v)|$) according to the attribute's VGH. That is $ILoss(v) = \frac{|base(v)|}{|base(*)|}$. For example, $ILoss$(Central Canada) $= \frac{2}{7}$ based on VGH in Figure 1. The *ILoss* for a tuple t' is defined as $\sum_{v_i \in t'} w_i \times ILoss(v_i)$, where w_i is the weight for loss in the i-th QI attribute. The *ILoss* of a generalized version R' is the sum of *ILoss* of its tuples.

2.2.3 Discernibility Metric

The *Discernibility metric* (C_{DM}) [2, 13] assigns a penalty to each tuple t' in a generalized version $R^{'}$ based on the number of tuples that are indistinguishable from t'. In specific, the penalty of tuple t' is $|G|^2$, if it belongs to QI-group G with $|G| \geq k$. If t' is in a QI-group G with size less than k ($|G| < k$), then t' is suppressed and its penalty is $|R| \times |G|$. The discernibility penalty for R' that generalizes R is defined as follows:

$$C_{DM} = \sum_{\forall G \ s.t. \ |G| \geq k} |G|^2 + \sum_{\forall G \ s.t. \ |G| < k} |R|.|G|,$$

2.2.4 Classification Metric

Iyenar [10] proposed a penalty metric to quantify the loss of a generalized relation that is used for classification. This metric is useful when the tuples in the original relation R are assigned a categorical class label in an effort to produce an anonymous public version R', where each QI-group in R' consist of tuples that are uniform with respect to the class label. No penalty is assigned to a tuple if it belongs to the majority class within its induced QI-group. All other tuples are penalized a value of 1. More precisely,

$$C_{CM}(R') = \sum_{\forall G \ s.t. \ |G| \geq k} (minority(G)) + \sum_{\forall G \ s.t. \ |G| < k} (|G|),$$

where *minority(G)* gets a QI-group with labeled tuples and returns a set of tuples that does not belong to the majority class in the QI-group.

2.2.5 Entropy Generalization Measure

For a private relation R with QI attributes $A_1, \cdots, A_n$, X_i is a random variable that equals the value of the i-th attribute A_i, $1 \leq i \leq n$. The *entropy generalization measure* [9] of R' that generalizes R is defined as,

$$\Pi(R^{'}) = \sum_{t' \in R'} \sum_{A_i} P(t'[A_i]) \times H(X_{A_i} |\, t'[A_i]). \tag{2.1}$$

in which $H(X_{A_i}|t'[A_i])$ is a conditional entropy and is defined as,

$$H(X_{A_i}|t'[A_i]) = - \sum_{b \in base(t'[A_i])} P(b) \times log\ P(b). \tag{2.2}$$

In (2.1), $P(v) \times H(H_{A_i}|v)$ defines a penalty measure for value v where $P(v) = \sum_{b \in base(v)} P(X_{A_i} = b)$. The penalty for a ground value v is 0 since the conditional entropy is 0. The penalty is maximum (i.e. $H(X_{A_i})$) when $v = *$.

These are typically general purpose penalty measures that quantify the data distortion in the entire generalized public database with respect to the data in the original specific database and do not consider the loss of data dependencies. Therefore, we propose a penalty measure known as *Dependency Loss* (Δ) that quantifies the loss of data dependencies in a generalized public database w.r.t the data dependencies in the original specific database.

2.3 Generalization Algorithms

Currently, there are many generalization algorithms, which are classified into two categories: *global recoding* and *local recoding*.

2.3.1 Global Recoding.

In global recoding, domains (Dom) of QI attributes are mapped to domains with more general values according to the generalization hierarchies of the attributes. *Single-dimensional generalization* is a special form of global recoding that specifies a mapping $\mu_i : \mathsf{Dom}^{A_i} \rightarrow \mathsf{Dom}'$ for every QI A_i. *Full-domain generalization* is a specific form of single-dimensional generalization where all the values in an attribute are in the same domain level Dom' according to the attribute's domain generalization hierarchy. Table 4 is obtained by single-dimensional generalization from Table 3 since a value in the former is not generalized to multiple values. However, Table 4 is not a full-domain generalization of Table 3 because all the values in an attribute are not from the same domain level. For example, PRV attribute values BC and Central Canada in Table 4 belong to different levels w.r.t DGH shown in Figure 1.

A form of global recoding is *multi-dimensional generalization* that extends single-dimensional generalization. In multidimensional generalization, QI attribute values are generalized according to a single mapping $\mu : \mathsf{Dom}^{A_1} \times \cdots \times \mathsf{Dom}^{A_n} \rightarrow D'$. Intuitively, multi-dimensional recoding divides the domain of QI values into a set of non-overlapping multidimensional regions, and each region will be replaced with generalized values. Representative global recoding algorithms include Datafly [22], Incognito [12], Mondrian [13].

Datafly [22]: Datafly is a greedy heuristic algorithm that performs single-dimensional full-domain generalization. The algorithm generates an array of QI-group sizes and if there exists at least one QI-group of size less than k, it generalizes the attribute in QI set having the most distinct values until k-anonymity is satisfied. Datafly algorithm guarantees to generate a k-anonymous relation, it does not provide the minimal generalization.

Incognito [12]: Incognito is an efficient single-dimensional full-domain generalization algorithm that builds a generalization lattice and traverses it using a bottom-up breadth-first search for computing k-minimal generalization. The key idea of Incognito is that if a relation R is is k-anonymous w.r.t QI attribute set of m attributes, then R is also k-anonymous with respect to any QI', where $QI' \subset QI$. The DGHs of QI attributes is traversed using bottom-up breadth-first search and checks for the k-anonymity property. In the first iteration, the algorithm checks the k-anonymity property for every single attribute in QI, removing those generalizations in each DGH of QI attribute that does not satisfy k-anonymity. In the next iteration, it combines the remaining generalization in pairs which forms generalization lattice and checks for k-anonymity using bottom-up breadth-first search, then in the triple and so on, until the entire QI attribute set is considered. In the given DGH of an attribute, when a generalization satisfies k-anonymity, then all its directed generalization also satisfy k-anonymity and therefore they will not be taken for consideration.

Mondrian [13]: Mondrian is a greedy algorithm that uses multidimensional global recoding technique. The QI attribute values are represented as a set of points in multidimensional space and each attribute represents a dimension. The algorithm achieves k-anonymity by recursively partitioning the space into regions until each region contains at least k points (tuples). In every iteration, the algorithm partitions the space by choosing a dimension d with the widest range of values and divides the region at the median value m, such that $d > m$ belongs to one region and the remaining belong to another region. The division is made if the region has greater than k points. The tuples in each region are generalized using the corresponding QI attributes VGHs.

2.3.2 Local Recoding.

In local recoding, same QI attribute values that appear in different tuples might be generalized to different ancestors in the generalized relation. For example, as shown in Table 5, MED attribute value ibuprofen remains same in tuples m_1, m_2 and is generalized to $*$ in tuples m_3 and m_4 w.r.t VGH shown in Figure 3. A category of local recoding algorithms is clustering-based generalization methods that partition tuples into subsets of tuples or clusters of size k and generalize the tuples in each cluster by unifying their QI attribute values. In local recoding, unlike global recoding, the clusters are not disjoint (overlapping clusters are allowed), i.e., any two non-distinct individual tuples may belong to different clusters and are generalized to different general values. Representative local recoding algorithms include Top-down approach [25], k-member clustering [3], One-Pass K-Means algorithm [16].

Top-down Approach [25]: The top-down approach is an efficient heuristic local recoding algorithm for k-anonymization. The general idea of the algorithm is to perform binary partitioning. That is, in each iteration, the algorithm form two groups G_x and G_y using two seed tuples $x, y \in R$ that maximize *penalty* if they are placed into the same group. The algorithm uses a heuristic method to find the seed tuples. First, the algorithm randomly picks a tuple x_1. Then, by scanning the rest of the tuples in R, the algorithm finds a tuple y_1, such that *penalty*(x_1, y_1) is maximized. Next, by scanning all the tuples again, the algorithm finds tuple x_2 that maximizes *penalty*(x_2, y_1). This process is repeated until there is no substantial growth in the *penalty*(x, y). The two seed tuples x and y form two groups G_x and G_y and the other tuples in R are assigned to the two groups in random order. The assignment of a tuple $t \in R$, depends on *penalty*(G_x, t) and *penalty*(G_y, t). Tuple t is assigned to the group that leads to a lower *penalty.* The overall partitioning costs $O|R^2|$.

k-member Clustering [3]: The greedy k-member clustering algorithm works as follows. The algorithm first randomly selects a tuple t from R as a seed to build a cluster c. Then the algorithm chooses a tuple t' that incur less information loss within the cluster c. This process is repeated until the size of cluster c reaches k. Once the size of the cluster c reaches k, the algorithm selects a tuple that is furthest from t and repeats the same process to build the next cluster. This process is repeated until there are less than k tuples left to build a cluster. Finally, as a postprocessing step, the algorithm iterates over the leftover tuples and assigns them to their closest clusters. The time complexity of the algorithm is $O|R^2|$.

One-Pass K-Means Algorithm (OKA) [16]: The greedy OKA algorithm derives from the K-Means algorithm and the algorithm proceeds in two stages: the clustering stage and the adjustment stage. In the clustering stage, the algorithm first randomly pics $K = \lfloor \frac{N}{k} \rfloor$ tuples as the seeds to build K clusters, where N is the total number of tuples in relation R and k specifies the threshold value for k-anonymity. Other tuples in R are assigned to K clusters in random order. For the tuple $t \in R$, OKA finds the cluster c that is closest t and assigns t to c and then updates the centroid of the cluster. The overall clustering costs $O(\frac{N^2}{k})$. After the clustering stage, some clusters size might be less than k. Therefore, in the adjustment stage, some records are removed from the clusters with more than k tuples and these tuples are added to their respective closest clusters with less than k tuples. This process is repeated until there exists no cluster with size less than k.

In summary, generally global recoding generalization algorithms are efficient, but the quality of information in the anonymized solution is low because anonymization is done at a domain level. On the contrary, local recoding algorithms greatly improve the quality of information because anonymization is done at the tuple level, but these algorithms are often inefficient because of massive distance computations between the tuples. In this work, we propose two simple and efficient local recoding algorithms for dependency preserving generalization problem because the information loss induced by this kind of algorithms is less than the global recoding algorithms.

Chapter 3

Preliminaries and Problem Definition

3.1 Relational Database

We assume that the data is stored in relations. A relation R with schema $\mathcal{R} = \{A_1, ..., A_n\}$ is a finite set of n-ary tuples $\{t_1, ..., t_N\}$ with attributes A_i. A database D is a finite set of relations $R_1, ..., R_m$ with database schema $\mathcal{S} = \{\mathcal{R}_1, ..., \mathcal{R}_m\}$. We use A, B, C for attributes and X, Y, Z for sets of attributes. The set of tuples in a relation R is denoted by N ($|R| = N$) and the total number of attributes in R is denoted by n ($|\mathcal{R}| = n$). For a tuple in relation R, each attribute can be associated with only one value from the set of attribute's domain values. The attribute domain is the set of all unique values permitted for an attribute, e.g. a domain of day-of-week is {Monday, Tuesday, $\cdots$, Sunday}. Each attribute value in a tuple represents a piece of information about an entity.

Example 3.1. *Table 6 is an example of a relation containing five tuples and five attributes: First Name* (FName)*, Last Name* (LName)*, Gender* (GEN)*, Postal-Code* (PC)*, Province* (PRV)*.*

FName	**LName**	**GEN**	**PC**	**PRV**
Andre	Brown	Male	J5B	QC
Carol	Anderson	Female	R7A	AB
Beth	Hill	Female	R7A	AB
Ellen	Furlan	Female	K2H	ON
Dan	Johnson	Male	K2H	ON

TABLE 6: Relational table.

3.1.1 Functional Dependency

A *functional dependency* (FD) φ over a relation R with schema $\mathcal{R}$ is denoted by $\varphi : X \rightarrow Y$, where X, Y are attribute sets in $\mathcal{R}$ ($X, Y \subseteq \mathcal{R}$). We say φ holds over R if for every pair of tuples $t_1, t_2 \in R$, $t_1[X] = t_2[X]$ implies $t_1[Y] = t_2[Y]$. We say R satisfies a set of FDs Σ, $R \models \Sigma$, if $R \models \varphi, \forall \varphi \in \Sigma$. The set of dependency instances of Σ in R is denoted by Γ.

Example 3.2. *An FD that holds true over relation in Table 6 is* $\varphi_1 : \mathsf{PC} \rightarrow \mathsf{PRV}$*, since for every two tuples* $t_1, t_2 \in R$*,* $t_1[\mathsf{PC}] = t_2[\mathsf{PC}]$ *implies* $t_1[\mathsf{PRV}] = t_2[\mathsf{PRV}]$*. Example of an instance is* $(\mathsf{J5B}, \mathsf{QC})$ *of* $\varphi_1 : \mathsf{PC} \rightarrow \mathsf{PRV}$ *and with* $\Sigma = \{\varphi_1\}$*, the* *set of dependency instances of* Σ *in* R *is* $\Gamma = \{(\mathsf{J5B}, \mathsf{QC}), (\mathsf{R7A}, \mathsf{AB}), (\mathsf{K2H}, \mathsf{ON})\}$ *and* $|\Gamma| = 3$.

3.2 Privacy Definitions

Many privacy models are proposed in PPDP [7] to prevent the re-identification of the individuals in the published relation. k-anonymity has emerged as an effective model in anonymization [23, 21, 22]. The k-anonymity model assumes the data is stored in relation and each tuple/record in the relation corresponds to a unique real-world entity, for example, an organization or a person and various tuples need not be unique. The tuple ID uniquely identifies each real-world entity.

Definition 1 (Quasi-Identifier (QI))**.** *A quasi-identifier of a relation* R *denoted as* QI*, is the set of attributes in* R *that could be linked with the external information to re-identify the individual records with high probability.*

Example 3.3. *The attribute set* {PC, PRV, MED} *in Table 7 is a quasi-identifier.*

In addition to the QI, a relation may contain publicly unknown highly sensitive attributes. The attribute Diagnosis (DIAG) in Table 7 is an example of a sensitive attribute. The quasi-identifiers are determined based on the content of externally available data and background knowledge obtained from previous data releases. The main goal of k-anonymity is to anonymize a relation so that no one can link the records in the published version to the corresponding real-world entities.

Definition 2 (QI-group)**.** *A QI-group of a relation R is a set of tuples in R that have the same set of values for the QIs.*

Example 3.4. *In Table 8, the first two rows* $\{r_1, r_2\}$ *form a QI group on* {PC, PRV, MED}.

ID	PC	PRV	MED	DIAG
t_1	V6J	BC	ibuprofen	osteoarthritis
t_2	V6J	BC	ibuprofen	osteoarthritis
t_3	J5B	QC	ibuprofen	osteoarthritis
t_4	K2H	ON	ibuprofen	osteoarthritis
t_5	K2H	ON	diazepam	seizure
t_6	J5B	QC	diazepam	seizure
t_7	R7A	MB	diazepam	seizure
t_8	R7A	MB	diazepam	seizure

TABLE 7: Private table R.

Definition 3 (k-anonymity property)**.** *A view V of a relation R is k-anonymous if the size of every QI-group in V is at least k with respect to the QI attribute set.*

Example 3.5. *Table 7 do not satisfy the 2-anonymity property since the tuples* $t_3, \cdots, t_6$ *occur only once.*

ID	**PC**	**PRV**	**MED**	**DIAG**
r_1	V6J	BC	ibuprofen	osteoarthritis
r_2	V6J	BC	ibuprofen	osteoarthritis
r_3	[J,K,L]	Central Canada	ibuprofen	osteoarthritis
r_4	[J,K,L]	Central Canada	ibuprofen	osteoarthritis
r_5	[J,K,L]	Central Canada	diazepam	seizure
r_6	[J,K,L]	Central Canada	diazepam	seizure
r_7	R7A	MB	diazepam	seizure
r_8	R7A	MB	diazepam	seizure

Table 8: 2-anonymous public view R'.

ID	**PC**	**PRV**	**MED**	**DIAG**
m_1	V6J	BC	ibuprofen	osteoarthritis
m_2	V6J	BC	ibuprofen	osteoarthritis
m_3	J5B	QC	*	osteoarthritis
m_4	K2H	ON	*	osteoarthritis
m_5	K2H	ON	*	seizure
m_6	J5B	QC	*	seizure
m_7	R7A	MB	diazepam	seizure
m_8	R7A	MB	diazepam	seizure

Table 9: 2-anonymous public view R''.

Definition 4 (k-anonymization)**.** *K-anonymization [12] is a process to modify a private table to a view that satisfies the k-anonymity property with respect to the QI attribute set. Since the anonymized version of the original data is a view, duplicate records are possible.*

Example 3.6. *Table 8 and Table 9 are two different k-anonymous views of R in Table 7 with $k = 2$ considering QI attribute set* {PC, PRV, MED} *since all QI-groups are of size* 2.

The k-anonymity model assumes that QI is known to the data publisher before data publishing. k-anonymity provides stronger privacy by including more attributes in the QI set. However, this also implies that more data distortion is needed to satisfy k-anonymity requirement because more tuples in a QI-group have to agree on more attributes.

One dilemma faced by a data publisher is how to classify attributes in a original private table into three disjoint sets, QI, sensitive attributes and non-sensitive attributes. In general, an attribute A should be included in the QI if the attacker could potentially obtain A from other external sources. There is no certain answer to the question of how a data publisher can determine whether or not an attacker can obtain an attribute A from some external sources, but it is important to understand the consequences of a mis-classification. Mis-classifying an attribute A into sensitive or non-Sensitive attribute may compromise another sensitive attribute S. An attacker or the data recipient may obtain A from other external sources and then use A to perform record linkage on attribute S.

3.3 Generalization

Samarati and Sweeney [21, 22, 23] devised mechanisms for k-anonymization using the concept of generalization and suppression. Generalization replaces QI values in a relation with less specific, but semantically consistent values according to some generalization hierarchies. For each QI attribute A, we assume a set of levels $\mathcal{L}^A = \{l_0^A, ..., l_h^A\}$ with a partial order $\leq^A$, called a *generalization relationship* on $\mathcal{L}^A$. Levels l_i^A are assigned with disjoint domain-sets $dom(l_i^A)$. The notation $l_j^A \leq l_i^A$ implies that $dom(l_i^A)$ generalizes $dom(l_j^A)$. The domain set $dom(l_h^A)$ is the top domain set without a parent and it has only one value. The domain set $dom(l_0^A)$ is the *ground domain set.* The *domain generalization hierarchy* denoted by DGH^A, is defined to be the set of domains that is totally ordered by the domain generalization relationship $\leq^A$. We use h^A to refer to the number of levels in DGH^A. The domain-set of attribute A, denote by Dom^A, is a subset of $\bigcup dom(l_i^A)$. Figure 4(a) shows the DGH for the medication (MED) attribute.

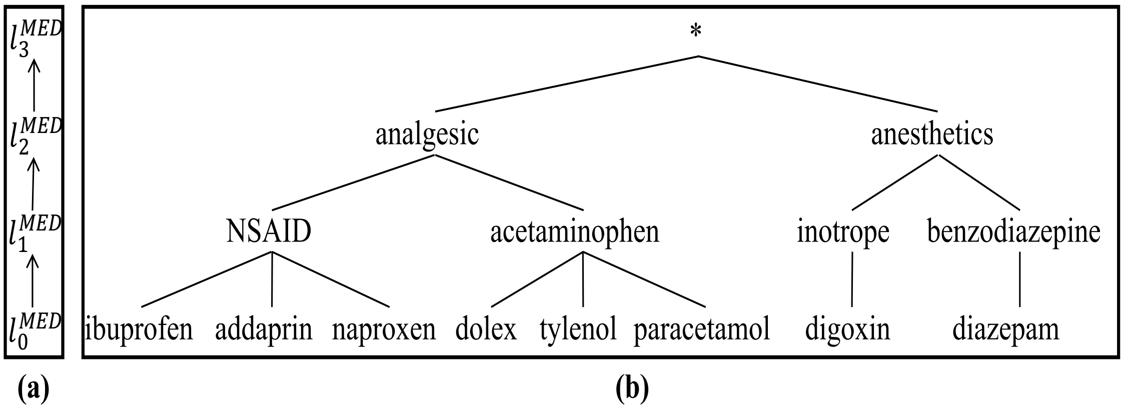


FIGURE 4: (a)DGH and (b)VGH of medication (MED).

A *value generalization relationship* for attribute A is a partial order $\preceq^A$ on $\bigcup dom(l_i^A)$. It implies a value generalization hierarchy, denoted as VGH^A, which is a tree whose leaves are ground domain-set values ($dom(l_0^A)$) and root is a single value from the top domain set ($dom(l_n^A)$) in DGH. For two values v' and v in Dom^A, $v' \preceq^A v$ implies that v' is more specific than v according to the VGH^A. Figure 4(b) shows VGH of the MED attribute. The value v' is *ground* if there is no other value more specific than v'. A value is *general* if it is not ground. For example in Figure 4(b), ibuprofen is ground and NSAID is general. A relation is *generalized* if it has some general values and it is *ground* if its values are ground. For a general value v of attribute A, we use $base(v)$ to refer to the set of descendant leaf nodes of v and $level(v)$ is its level according to the generalization hierarchies of A. For example, $base$(NSAID)= {ibuprofen, addaprin, naproxen} and $level$(NSAID)=1 and *level*(ibuprofen)=0 according to generalization hierarchy of MED in Figure 4. Suppression is a special case of generalization, replaces some values with a special value(*), indicating that the replaced value cannot be disclosed.

3.3.1 Generalization Penalty

Generalization is accompanied by information loss and to measure the loss various generalization penalties have been proposed in PPDP. These existing penalty measures such as ILoss [10, 25], penalty [18] and precision measure [23] measure the information loss considering the VGHs of the QI attribute values. Our generalization penalty measure is based on an entropy-based generalization measure [9], which considers both the VGHs and the entropy of QI values in the original ground relation R.

Example 3.7. *Consider the ground relation R in Table 10 and the general values* Central Canada *and* Prairies *in the* VGH *of the attribute* province (PRV) *in Figure 5. According to the ILoss measure,* Central Canada *and* Prairies *have the same penalty value* 2 *since they both have two ground values according to the* VGH *of* PRV*. However, generalizing* MB *to* Prairies *and* QC*,* ON *to* Central Canada *incur different information loss as the ground value* AB *does not appear in Table 3. Similarly* * *in the* VGH *of* MED *has* 8 *ground values and the ILoss penalty value is* 8*. However, only* ibuprofen *and* diazepam *appear in Table 3.*

ID	PC	PRV	MED	DIAG
t_1	V6J	BC	ibuprofen	osteoarthritis
t_2	V6J	BC	ibuprofen	osteoarthritis
t_3	J5B	QC	ibuprofen	osteoarthritis
t_4	K2H	ON	ibuprofen	osteoarthritis
t_5	K2H	ON	diazepam	seizure
t_6	J5B	QC	diazepam	seizure
t_7	R7A	MB	diazepam	seizure
t_8	R7A	MB	diazepam	seizure

Table 10: Private table R.

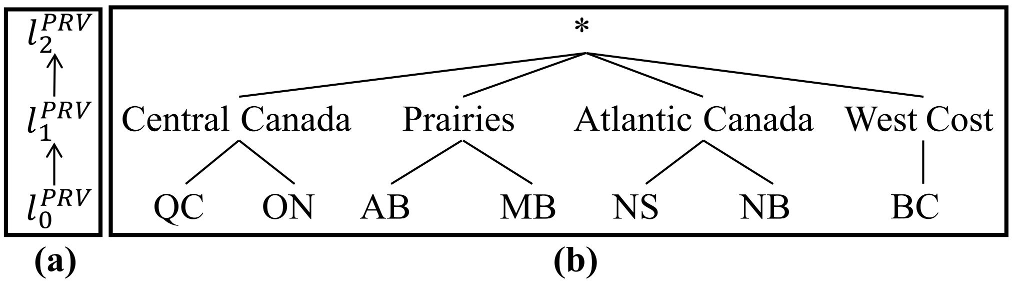


Figure 5: (a)DGH and (b)VGH of province (PRV).

Assuming that the data user has some prior knowledge about the values that appear in the original relation R, for instance, the data user based on his/her background knowledge knows that the hospital patient data in Table 10 has no patient entries from the Alberta (AB) province, then replacing MB with its parent value Prairies do not induce any uncertainty/information loss for the data user, which is not taken into consideration for measuring the information loss.

Entropy-Based Penalty. For every QI attribute A in a ground relation R, X_A is a random variable of values $t[A]$ where t is a random tuple in R. The *entropy-based penalty* [9] of a value v in the VGH ^{A} is defined as,

$$E(v) = P(v) \times H(X_A|v), \tag{3.1}$$

where $P(v) = P(X_{A_i} \in base(v))$ is the probability of values in $base(v)$ in the original ground relation R and $H(X_A|v) = -\sum_{b\in base(v)} P(b|v) \times log\ P(b|v)$ is the conditional entropy with $P(b|v) = P(X_A = b|X_A \preceq v)$. When v is a ground value, the conditional entropy $H(X_A|v) = 0$ and therefore the penalty is minimum, i.e. $E(v) = 0$. The penalty is maximum, i.e. $E(v) = H(X_A)$, for $v = *$ because $P(*) = 1$. Intuitively, $E(v)$ is an information theoretic measure that penalize v based on the entropy of values in $base(v)$ in the original ground relation R.

Example 3.8. *According to the original relation R in Table 10 and* VGH *of* PRV *in Figure 5,* $E(\text{Central Canada}) = P(\text{Central Canada}) \times H(X_{\text{PRV}}|\text{Central Canada}) \approx 0.5 \times 0.3 = 0.15$. *Here,* $P(\text{Central Canada}) = P(X_{A_i} \in base(\text{Central Canada})) = P(QC) + P(ON) = \frac{2}{8} + \frac{2}{8} = 0.5$, *because* $base(\text{Central Canada}) = \{\text{QC}, \text{ON}\}$ *according to the* province (PRV) VGH *in Figure 5 and* $P(\text{QC}) = \text{P(ON)} = \frac{2}{8}$ *in Table 10. The conditional entropy* $H(X_{\text{PRV}}|\text{Central Canada}) = -\sum_{b\in base(\text{Central Canada})} P(b|\text{Central Canada}) \times log\ P(b|\text{Central Canada}) = -(P(\text{QC}|\ \text{Central Canada}) \times log\ P(\text{QC}\ |\text{Central Canada}) + P(\text{ON}|\text{Central Canada}) \times log\ P(\text{ON}|\ \text{Central Canada})) = -(2 \times \frac{2}{4} \times log\frac{2}{4}) \approx 0.3$. *Here* $P(\text{QC}\ |\text{Central Canada}) = P(\text{ON}\ |\text{Central Canada}) = \frac{2}{4}$

w.r.t $t_3[\mathsf{PRV}], \cdots, t_6[\mathsf{PRV}]$ *in Table 10. On the other hand,* $E(\mathsf{Prairies}) = P(\mathsf{MB}) \times H(X_{\mathsf{MB}}|\mathsf{Prairies}) = \frac{2}{8} \times (-\frac{2}{2} \times log\ \frac{2}{2}) = 0$. *Here* $base(\mathsf{Prairies}) = \{\mathsf{AB}, \mathsf{MB}\}$ *according to the* province (PRV) VGH *in Figure 1. However, only* MB *appears in Table 10* (*in* $t_7[\mathsf{PRV}]$, $t_8[\mathsf{PRV}]$). *Assuming that the data user has some background knowledge about the values that appear in original relation* R*, replacing* MB *with* Prairies *do not induce any uncertainty to the data user. Therefore,* $E(\mathsf{Prairies}) = 0$.

3.4 Clustering Based Generalization

Clustering-based generalization is a class of flexible local recoding generalization algorithms with high utility [25, 19, 3, 16]. The base idea is to partition a set of tuples into groups, such that the tuples in the same group are more similar to each other than tuples in other groups with respect to some defined similarity criteria and anonymize all the tuples in a group to the same generalized tuple to satisfy the privacy requirement. These class of algorithms are more flexible than the other generalization algorithms as they generalize attribute values at the cell level. They do not overgeneralize a table, and hence, they may minimize the distortion of an anonymous relation.

Definition 5. ***(Clustering.)*** *A clustering* $\mathcal{C}$ *of a ground relation* R*, is a set of clusters* $c_1, ..., c_m$ *such that* $c_i \cap c_j = \emptyset, i \neq j$ *and* $\bigcup c_i = R$. *For a cluster* c_i, $G(c_i)$ *is the QI-group obtained from tuples in* c_i *by unifying their QI values to their lowest common ancestors according to* VGHs*. The generalized relation* $R^{\mathcal{C}} = \bigcup_{c_i \in \mathcal{C}} G(c_i)$.

Example 3.9. *One possible way of clustering tuples in Table 7 is $\mathcal{C}_1$ that contains 4 clusters $c_{1,2} = \{t_1, t_2\}$, $c_{3,4} = \{t_3, t_4\}$, $c_{5,6} = \{t_5, t_6\}$, $c_{7,8} = \{t_7, t_8\}$. R' in Table 8 is the generalization w.r.t. $\mathcal{C}_1$. In R', the QI-group of the cluster $c_{1,2}$ is $G(c_{1,2}) = \{r_1, r_2\}$. Another possible clustering is $\mathcal{C}_2$ with clusters $c_{1,2} = \{t_1, t_2\}$, $c_{3,6} = \{t_3, t_6\}$, $c_{4,5} = \{t_4, t_5\}$, $c_{7,8} = \{t_7, t_8\}$ and the generalization w.r.t $\mathcal{C}_2$ is R'' in Table 9. The generalization w.r.t the singleton clustering $\mathcal{C}$, where each tuple in R is assigned to its own cluster $\mathcal{C} = \{c_1, ..., c_8\}$, results in the same original ground relation R.*

3.5 Distance and Dependency Loss Measure

In this section, we present the dependency loss measure for quantifying the loss of a generalized relation. This measure is different from the existing penalty measures because it takes into consideration the dependency instances in the private relation and measures how much they are distorted in the generalized relation. The dependency loss is based on the distance between the values and tuples.

Definition 6. ***(Distance between two values.)*** *For any two values v, v' in the domain of a QI attribute A (dom^A), the distance $\delta(v', v)$ is defined as follows:*

$$\delta(v, v') = \begin{cases} \dfrac{|E(v') - E(v)|}{H(X_A|*)} & \textbf{if } v \preceq v' \textbf{ or } v \preceq v', \\ 1 & \textbf{otherwise}, \end{cases} \tag{3.2}$$

Proposition 1. *The normalized distance function δ has the following properties. For every v, v', v'',*

(a) $\delta(v, v) = 0$, $0 \leq \delta(v, v') \leq 1$.

(b) $\delta(v, v') = \delta(v', v)$.

(c) $\delta(v, v'') \leq \delta(v, v') + \delta(v', v'')$.

Definition 7. *(**Distance between two tuples.**) For any two tuples t, t' with shared attributes $A_1, ..., A_m$, the distance function $d(t, t')$ is defined as follows:*

$$d(t, t') = \begin{cases} \dfrac{\sum_{A_i} \delta(t[A_i], t'[A_i])}{m} & \textbf{if } t \sqsubseteq t' \textbf{ or } t' \sqsubseteq t, \\ 1 & \textbf{otherwise}, \end{cases} \tag{3.3}$$

The distance function δ calculates the entropy-based penalty $E(v)$ and the conditional entropy $H(X_A)$ on the original ground relation R and returns a normalized distance value between 0 and 1. If v, v' are from the same branch in the in VGH^A, then distance is the normalized difference between their information loss. Otherwise, v, v' are incomparable, and the distance $\delta(v, v') = 1$. Note that for a sensitive attribute, $\delta(v, v')$ is an arbitrary user-defined distance function with a value between 0 and 1. The distance between tuples is the sum of the pairwise distance between their corresponding QI attribute values.

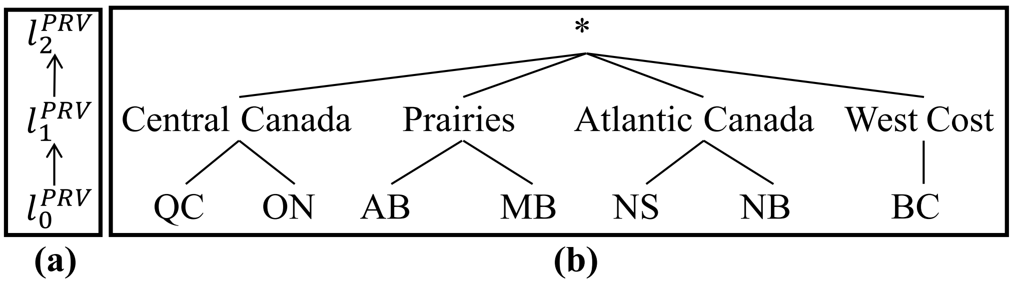


FIGURE 6: (a)DGH and (b)VGH of province (PRV).

ID	PC	PRV	MED	DIAG
t_1	V6J	BC	ibuprofen	osteoarthritis
t_2	V6J	BC	ibuprofen	osteoarthritis
t_3	J5B	QC	ibuprofen	osteoarthritis
t_4	K2H	ON	ibuprofen	osteoarthritis
t_5	K2H	ON	diazepam	seizure
t_6	J5B	QC	diazepam	seizure
t_7	R7A	MB	diazepam	seizure
t_8	R7A	MB	diazepam	seizure

TABLE 11: Private table R.

Example 3.10. *According to the ground relation R in Table 11 and* VGH *of* PRV *in Figure 6,* δ(BC, Central Canada)=1 *since* BC $\npreceq$ Central Canada *and* δ(QC, Central Canada) $= \frac{|\text{E(Central Canada)}-\text{E(QC)}|}{H(X_{\text{PRV}}|*)} \approx \frac{0.15}{0.6} = 0.25$ *since* QC $\preceq$ Central Canada, |E(Central Canada)-E(QC)|$= 0.15 - 0 = 0.15$ (E(Central Canada) *computation is shown in Example 3.8) and $H(X_{\text{PRV}}|*) \approx 0.6$. According to the* VGH *of* PRV*, base*$(*)=$ {QC, ON, AB, MB, NS, NB, BC}*. However, only four descendent leaf nodes* {BC, QC, ON, MB$\}$ *appears in Table 11* $(t_1[\mathsf{PRV}], \cdots, t_8[\mathsf{PRV}])$. *Therefore,* $H(X_{\mathsf{PRV}}|*) = -(P(\mathsf{BC}|*) \times \textit{log}\ P(\mathsf{BC}|*) + (\mathsf{QC}|*) \times \textit{log}\ P(\mathsf{QC}|*) + (\mathsf{ON}|*) \times \textit{log}\ P(\mathsf{ON}|*) + (\mathsf{MB}|*) \times \textit{log}\ P(\mathsf{MB}|*)) = -4 \times \frac{2}{8} \times \log(\frac{2}{8}) \approx 0.6$. *Similarly,* $\delta(\mathsf{J5B}, [\mathsf{J}, \mathsf{K}, \mathsf{L}]) \approx 0.25$.

Definition 8. ***(Dependency Loss**(Δ)) Consider a ground relation R and a set of FDs* Σ *that hold on R. The dependency loss of* R' *that generalizes R, i.e.* $R \sqsubseteq R'$, *is defined as follows:*

$$\Delta(R', R, \Sigma) = \sum_{p \in \Gamma} \min_{t' \in R'} (d(p, t')), \tag{3.4}$$

where Γ *is the set of dependency instances of* Σ *in R.*

The *Dependency Loss* (Δ) quantifies the information loss of a generalized version R' w.r.t to the data dependencies that hold on the relation R. The loss is measured by computing the distance between every dependency instance p in R and its closest dependency instance t' in R'. This measure is different from the existing penalty metrics because it quantifies the usefulness of an anonymous public version R' w.r.t to the data dependencies in the original private relation R. We define a new generalization problem using *Dependency loss* that aims to maximize the utility by preserving data dependencies in the anonymous public version R'.

3.6 Problem Definition

Definition 9. ***(Dependency-Preserving Generalization.)*** *Given an original private relation R, set of FDs Σ that holds on R and a value k, the dependency-preserving generalization problem is to find anonymous private version R' such that,*

(a) R' generalizes R i.e. $R \sqsubseteq R'$.

(b) R' is k-anonymous.

(c) The dependency loss $\Delta(R', R, \Sigma)$ is minimal.

The previous studies show that the problem of optimal k-anonymity is NP-hard [18, 9, 1]. In the next section, we present two efficient greedy clustering based generalization algorithms for the dependency preserving generalization problem.

Chapter 4

Dependency Preserving Generalization Algorithms

We propose two clustering based generalization algorithms, namely, PAIR-ENUM and k-ASSEMBLE to solve the dependency preserving generalization problem. In order to generate a k-anonymous public view R' with minimal *dependency loss*(Δ), we define a *utility* for clustering. This metric measures the quality of the clustering w.r.t preserving the dependencies more accurately and guides our algorithms at each step to find a clustering with maximal *utility*.

Definition 10. *Consider an original ground relation R, set of FDs Σ, and a clustering $\mathcal{C} = \{c_1, ..., c_m\}$ of R. The utility of $\mathcal{C}$ w.r.t Σ is defined as follows:*

$$utility(\mathcal{C}, R, \Sigma) = \sum_{p \in \Gamma} \max_{c \in \mathcal{C}}(preserve(c, p)). \tag{4.1}$$

where $0 \leq preserve(c, p) \leq 1$ *measures how much cluster* c *preserves instance* p *and is defined as,*

$$preserve(c, p) = (1 - \min_{t \in G(c)} (d(p, t))) \times \frac{|c|}{k}. \tag{4.2}$$

Assuming the instance p is over FD $\varphi : X \rightarrow Y$, the distance measure $0 \leq d(p, t) \leq 1$ is defined as,

$$d(p, t) = \frac{\sum_{A \in X \cup Y} \delta(p[A], t[A])}{|X \cup Y|}. \tag{4.3}$$

The *utility* of a clustering $\mathcal{C}$, is the sum of how much dependency instance p in Σ of the private relation R is preserved in the clustering $\mathcal{C}$. This is done by finding the cluster $c \in \mathcal{C}$ that maximizes $preserve(c, p)$, that is, the cluster c that best preserves the instance p. The $preserve(c, p)$ measure defined in (4.2) has two factors. The first factor $1 - \min_{t \in G(c)}(d(p, t))$ depends on the distance between the instance p and the closet tuple t in $G(c)$, where $G(c)$ is the QI-group obtained from unifying the tuples in the cluster c through generalization. The second, $\frac{|c|}{k}$ measures how close is the cluster c to satisfying the k-constraint. The distance function $d(p, t)$ in (4.3) is a normalized sum of the pairwise distance (δ) between the QI attribute values in p and t. The pairwise distance between the attribute values is computed using the distance function δ defined in (6).

ID	PC	PRV	MED	DIAG
r_1	V6J	BC	ibuprofen	osteoarthritis
r_2	V6J	BC	ibuprofen	osteoarthritis
r_3	[J,K,L]	Central Canada	ibuprofen	osteoarthritis
r_4	[J,K,L]	Central Canada	ibuprofen	osteoarthritis
r_5	[J,K,L]	Central Canada	diazepam	seizure
r_6	[J,K,L]	Central Canada	diazepam	seizure
r_7	R7A	MB	diazepam	seizure
r_8	R7A	MB	diazepam	seizure

TABLE 12: 2-anonymous public view R'.

ID	PC	PRV	MED	DIAG
m_1	V6J	BC	ibuprofen	osteoarthritis
m_2	V6J	BC	ibuprofen	osteoarthritis
m_3	J5B	QC	*	osteoarthritis
m_4	K2H	ON	*	osteoarthritis
m_5	K2H	ON	*	seizure
m_6	J5B	QC	*	seizure
m_7	R7A	MB	diazepam	seizure
m_8	R7A	MB	diazepam	seizure

TABLE 13: 2-anonymous public view R''.

Example 4.1. *(ex. 3.10 cont.) For the dependency instance $p_1 = (\mathsf{V6J}, \mathsf{BC})$ and r_1 in Table 12, $d(p_1, r_1) = 0$ since the same instance appears in R'. For $k = 2$, the best cluster in Table 12 that preserves p_1 is $c_{1,2}$ because $G(c_{1,2}) = \{r_1, r_2\}$ and the preserve$(c_{1,2}, p_1) = (1-d(p_1, r_1)) \times \frac{|c_{1,2}|}{k} = (1-0) \times \frac{2}{2} = 1$, which means the instance p_1 is completely preserved in the cluster $c_{1,2}$ of R'. For the instance $p_2 = (\mathsf{J5B}, \mathsf{QC})$ and r_3 in Table 12, $d(p_2, t_3) = \frac{\delta(\mathsf{J5B},[\mathsf{J,K,L}])+\delta(\mathsf{QC},\mathsf{Central\ Canada})}{2} = \frac{0.25+0.25}{2} = 0.25$. The best cluster in Table 12 that preserves p_2 is $c_{3,4}$, because $G(c_{3,4}) = \{r_3, r_4\}$ and preserve$(c_{3,4}, p_2) = (1 - d(p_2, r_3)) \times \frac{|c_{3,4}|}{k} = (1 - 0.25) \times \frac{2}{2} = 0.75$, which means the instance p_2 is partially preserved in the cluster $c_{3,4}$ of R' with preserve score of* 0.75*. In the singleton clustering $\mathcal{C}$, the preserve measure for all the six instances is 0.5. Therefore, utility$(\mathcal{C}, R, \Sigma) = 6 \times 0.5 = 3$. For the public view R' in Table 12, four instances (p_1, p_4, p_5, p_6) are completely preserved by the preserve score of 1 and two instance are preserved by the preserve score of 0.75. Therefore, utility$(R', R, \Sigma) = \frac{4\times1+2\times0.25}{6} = 0.75$. For the public view R'' in Table 13, all* *the six instances are completely preserved with a preserve score of 1. Therefore,* $utility(R'', R, \Sigma) = \frac{6\times 1}{6} = 1$.

4.1 PAIR-ENUM Algorithm

The pseudo-code of PAIR-ENUM is shown in Algorithm 1. Given an original private view R, set of FDs Σ and k value as the input, the algorithm returns a k-anonymous public relation $R^{\mathcal{C}}$ with minimal *dependency loss* (Δ), because the algorithm evaluates the *utility* of all the possible clusterings and merges the cluster pairs that has maximal utility at each step to generate a final k-anonymous view with minimal dependency loss.

In PAIR-ENUM, S is the subset of tuples in R that are in clusters of size k. These are called the sealed clusters. S is initially empty (line 1). The algorithm begins with a single clustering $\mathcal{C}$, where each tuple $t \in R$ is assigned to its own cluster $c \in \mathcal{C}$ (line 3). The algorithm continues as long as there are at least k tuples in $|R \setminus S|$, that can be sealed. If the number of unsealed tuples in R is less than k, then these tuples are suppressed since they cannot form a new sealed cluster of size k and the algorithm stops. The while loop in (line 4) finds the clustering $\mathcal{C}'$ with maximum $utility(\mathcal{C}', R, \Sigma)$ which is obtained from $\mathcal{C}$. To find $\mathcal{C}'$, the algorithm iterates over every possible cluster pairs c_i and c_j that can be merged to generate the clustering $\mathcal{C}'$(line 6). The merge function $merge(c_i, c_j)$ in line 7 returns the cluster $c_{i,j}$ (obtained by merging the clusters c_i and c_j) if $|c_i \cup c_j| \leq k$. Otherwise,

Algorithm 1: PAIR-ENUM (R,Σ, k)

Input: Private relation R with QI attributes $A_1, A_2, \cdots$, set of FDs $\Sigma = \{\varphi_1, \varphi_2, \cdots\}$, k value for k-anonymity.

Output: $R^{\mathcal{C}}$ with $R \sqsubseteq R'$ with maximum *utility*.

```
 1  $S := \emptyset$;  $\mathcal{C} = \emptyset$;
 2  foreach tuple $t_i \in R$ do
 3      $\mathcal{C} \leftarrow \mathcal{C} \cup \{\{t_i\}\}$;
 4  while $|R \setminus S| \geq k$ do
 5      $MaxU := 0$;  $\mathcal{C}' := \emptyset$;
 6      foreach pair $c_i, c_j \in \mathcal{C}$ such that $(c_i \cup c_j) \cap S = \emptyset$ do
 7          $c_{i,j} := merge(c_i, c_j)$;
 8          $\mathcal{C}_{i,j} := (\mathcal{C} \setminus \{c_i, c_j\}) \cup \{c_{i,j}\}$;
 9          if $|c_i \cup c_j| > k$ then
10              $c' := (c_i \cup c_j) \setminus c_{i,j}$;  $\mathcal{C}_{i,j} := \mathcal{C}_{i,j} \cup \{c'\}$;
11          $u := utility(\mathcal{C}_{i,j}, R, \Sigma)$;
12          if $u > MaxU$ then
13              $MaxU := u$;  $\mathcal{C}' := \mathcal{C}_{i,j}$;
14      foreach cluster $c$ in $\mathcal{C}' \setminus \mathcal{C}$ do
15          if $|c| = k$ then $S := S \cup c$;
16      $\mathcal{C} := \mathcal{C}'$;
17  return $R^{\mathcal{C}}$;
```

if $|c_i \cup c_j| > k$ (line 10), $merge(c_i, c_j)$ returns a subset $c_{i,j}$ of $c_i \cup c_j$ containing k closest tuples.

The clustering $\mathcal{C}_{i,j}$ in line 8 is generated by removing the clusters c_i and c_j from $\mathcal{C}$ and adding the merged cluster $c_{i,j}$. The tuples left after merging c_i, c_j form a new cluster c' which is added to the clustering $C_{i,j}$ (line 10). If c is sealed (line 15), the algorithm adds the tuples in c to S and updates the clustering C to the best clustering C'. After the while loop, the algorithm returns the final clustering $\mathcal{C}$ containing clusters of size k. Using C, we generate the final k-anonymous public view $R^{\mathcal{C}}$, by generalizing the tuples in each cluster $c_i \in \mathcal{C}$ to have the equal QI values using the attributes generalization hierarchies (VGHs). For each tuple t_i in cluster $c = \{t_1, ..., t_k\} \in \mathcal{C}$, for each QI attribute A, $t'_i[A]$ holds the least common ancestor of $t_1[A], ..., t_k[A]$ according to the VGH^A.

ID	**PC**	**PRV**	**MED**	**DIAG**
c_1	V6J	BC	ibuprofen	osteoarthritis
c_2	V6J	BC	ibuprofen	osteoarthritis
c_3	J5B	QC	ibuprofen	osteoarthritis
c_4	K2H	ON	ibuprofen	osteoarthritis
c_5	K2H	ON	diazepam	seizure
c_6	J5B	QC	diazepam	seizure
c_7	R7A	MB	diazepam	seizure
c_8	R7A	MB	diazepam	seizure

TABLE 14: Singleton clustering $\mathcal{C}$ of R.

ID	**PC**	**PRV**	**MED**	**DIAG**
$c_{1,2}$	V6J	BC	ibuprofen	osteoarthritis
$c_{1,2}$	V6J	BC	ibuprofen	osteoarthritis
c_3	J5B	QC	ibuprofen	osteoarthritis
c_4	K2H	ON	ibuprofen	osteoarthritis
c_5	K2H	ON	diazepam	seizure
c_6	J5B	QC	diazepam	seizure
c_7	R7A	MB	diazepam	seizure
c_8	R7A	MB	diazepam	seizure

TABLE 15: Clustering $\mathcal{C}_{1,2}$.

ID	PC	PRV	MED	DIAG
c_1	V6J	BC	ibuprofen	osteoarthritis
c_2	V6J	BC	ibuprofen	osteoarthritis
$c_{3,4}$	[J,K,L]	Central Canada	ibuprofen	osteoarthritis
$c_{3,4}$	[J,K,L]	Central Canada	ibuprofen	osteoarthritis
c_5	K2H	ON	diazepam	seizure
c_6	J5B	QC	diazepam	seizure
c_7	R7A	MB	diazepam	seizure
c_8	R7A	MB	diazepam	seizure

Table 16: Clustering $\mathcal{C}_{3,4}$.

ID	PC	PRV	MED	DIAG
c_1	V6J	BC	ibuprofen	osteoarthritis
c_2	V6J	BC	ibuprofen	osteoarthritis
c_3	J5B	QC	ibuprofen	osteoarthritis
c_4	K2H	ON	ibuprofen	osteoarthritis
c_5	K2H	ON	diazepam	seizure
c_6	J5B	QC	diazepam	seizure
$c_{7,8}$	R7A	MB	diazepam	seizure
$c_{7,8}$	R7A	MB	diazepam	seizure

Table 17: Clustering $\mathcal{C}_{7,8}$.

ID	PC	PRV	MED	DIAG
$c_{1,2}$	V6J	BC	ibuprofen	osteoarthritis
$c_{1,2}$	V6J	BC	ibuprofen	osteoarthritis
$c_{3,6}$	J5B	QC	ibuprofen	osteoarthritis
$c_{4,5}$	K2H	ON	ibuprofen	osteoarthritis
$c_{4,5}$	K2H	ON	diazepam	seizure
$c_{3,6}$	J5B	QC	diazepam	seizure
$c_{7,8}$	R7A	MB	diazepam	seizure
$c_{7,8}$	R7A	MB	diazepam	seizure

Table 18: Final clustering $\mathcal{C}$.

ID	PC	PRV	MED	DIAG
$c_{1,2}$	V6J	BC	ibuprofen	osteoarthritis
$c_{1,2}$	V6J	BC	ibuprofen	osteoarthritis
$c_{3,6}$	J5B	QC	*	osteoarthritis
$c_{4,5}$	K2H	ON	*	osteoarthritis
$c_{4,5}$	K2H	ON	*	seizure
$c_{3,6}$	J5B	QC	*	seizure
$c_{7,8}$	R7A	MB	diazepam	seizure
$c_{7,8}$	R7A	MB	diazepam	seizure

Table 19: 2-anonymous public view R''.

Example 4.2. *(ex. 1.2 cont.) The algorithm begins with the singleton clustering, where each tuple in R is assigned to its own cluster as shown in Table 14 and utility$(\mathcal{C}, R, \Sigma) = 6 \times 0.5 = 3$. The number of possible cluster pairs evaluated in iteration 1 is $\{c_{1,2}, c_{1,3}, \cdots, c_{7,8}\} = 28$. As shown in Table 15, $\mathcal{C}_{1,2}$*

is a candidate clustering for the first iteration generated by merging clusters c_1 *and* c_2 *and* $utility(\mathcal{C}_{1,2}, R, \Sigma) = 2 + (4 \times 0.5) = 4$ *since the instances* $p_1 = (\mathsf{V6J}, \mathsf{BC})$ *and* $p_5 = (\mathsf{ibuprofen}, \mathsf{osteoarthritis})$ *will be completely preserved by* $c_{1,2}$ *with* $preserve(c_{1,2}, p_1) = preserve(c_{1,2}, p_5) = 1$ *and the remaining 4 instances are partially preserved with preserve score of* 0.5*. Another candidate clustering* $\mathcal{C}_{7,8}$ *in Table 17 also has the* $utility(\mathcal{C}_{7,8}, R, \Sigma) = 2 + (4 \times 0.5) = 4$ *in which instances* $p_4 = (\mathsf{R7A}, \mathsf{MB}), p_6 = (\mathsf{diazepam}, \mathsf{seizure})$ *are completely preserved, i.e.* $preserve(c_{7,8}, p_4) = preserve(c_{7,8}, p_6) = 1$*. For the clustering* $\mathcal{C}_{3,4}$*,* $utility(\mathcal{C}_{3,4}, R, \sigma) = 1 + (5 \times 0.5) = 3.5$ *as for only one instance* $preserve(c_{3,4}, p_5) = 1$ *and the remaining* 5 *instances still have preserve measure* 0.5*. Among all the possible clusterings evaluated in iteration 1, the two clusterings* $\mathcal{C}_{1,2}$ *and* $\mathcal{C}_{7,8}$ *have the same maximum utility. The algorithm breaks the tie by choosing the first generated clustering* $\mathcal{C}_{1,2}$ *as the best clustering in iteration 1 and continues by replacing* $\mathcal{C}$ *with* $\mathcal{C}_{1,2}$*. The tuples in the cluster* $c_{1,2}$ *are sealed since* $|c_{1,2}| = k$*. Ignoring the tuples in the sealed cluster* $c_{1,2}$*, the possible cluster pairs evaluated in iteration 2 is* $\{c_{2,3}, c_{2,4}, \cdots, c_{7,8}\} = 15$*. Following the same steps, PAIR-ENUM generates the final clustering* $\mathcal{C}$ *as shown in Table 18 with* $utility(\mathcal{C}, R, \Sigma) = 6$*, which means all the six dependency instances* $\Gamma = \{p_1, \cdots, p_6\}$ *are completely preserved in the final clustering* $\mathcal{C}$*. Table 19 is the corresponding* k*-anonymous public view* R'' *obtained by generalizing QI values of tuples in each cluster* $c_i \in \mathcal{C}$ *using the* VGHs *shown in Figures 1-3, with dependency loss* $\Delta(R'', R, \Sigma) = 0$*.*

4.2 k-ASSEMBLE Algorithm

The pseudo-code of k-ASSEMBLE is shown in Algorithm 2. At each step, k-ASSEMBLE initializes a pool of m clusterings, with each clustering containing a cluster of size k. We compute the *utility* of each of these m clusterings at each step and greedily select the clustering among the m clusterings with the highest *utility* as an input to the next step. Given that we take a greedy approach, we can achieve utility that is locally maximal.

In the while loop (line 4), k-ASSEMBLE finds a clustering $\mathcal{C}'$ that is obtained from $\mathcal{C}$ and has the highest utility, *utility*$(\mathcal{C}', R, \Sigma)$. To find $\mathcal{C}'$, the algorithm iterates over a pool of m clusterings, with each clustering containing a new cluster of size k to form $\mathcal{C}'$. To generate a pool of m clusterings in each iteration, k-ASSEMBLE randomly picks m unsealed tuples from $\mathcal{C}$ and then adds them to the pool $\mathcal{P}$ (line 6 and 7). For each tuple t_i in the pool $\mathcal{P}$, k-ASSEMBLE creates a cluster c_i with its $k-1$ closest neighbours in $\mathcal{C}$ (lines 11 - 14). The *find_best_cluster*$(\mathcal{C}, c_i)$ in (line 12) returns the closest neighbour to t_i w.r.t distance measure defined in Definition 7. The algorithm generates $\mathcal{C}_i$ by removing $k-1$ tuples (which are added to c_i) from $\mathcal{C}$ and adding c_i to $\mathcal{C}$. k-ASSEMBLE adds the tuples in cluster c to S since $|c| = k$ (line 20) and updates the clustering $\mathcal{C}$ to the best clustering $\mathcal{C}'$. The while loop in line 4 ends by returning the final clustering $\mathcal{C}$ containing clusters of size k and we generalize the tuples in each $c_i \in \mathcal{C}$ to generate the final k-anonymous public view $R^{\mathcal{C}}$. We can tune k-ASSEMBLE via m to balance the trade-off between runtime and clustering utility. By assigning a larger value for m, the pool size increases which may allow k-ASSEMBLE to find

Algorithm 2: k-ASSEMBLE (R,Σ, k)

```
Input: Ground relation $R$ with QI attributes $A_1, A_2, \cdots$, set of FDs
       $\Sigma = \{\varphi_1, \varphi_2, \cdots\}$, value $k$ for $k$-anonymity.
Output: $R^{\mathcal{C}}$ with $R \sqsubseteq R'$ with maximum utility.
1  $S := \emptyset$;  $\mathcal{C} = \emptyset$;  $\mathcal{P} = \emptyset$
2  foreach tuple $t_i \in R$ do
3      $\mathcal{C} \leftarrow \mathcal{C} \cup \{\{t_i\}\}$;
4  while $|R \setminus S| \geq k$ do
5      $MaxU := 0$;  $\mathcal{C}' := \emptyset$;
6      Randomly select $m$ tuples from $\mathcal{C}$ such that $\{t_1, \cdots, t_m\} \cap S = \emptyset$;
7      $\mathcal{P} = \mathcal{P} \cup t_i$ for $i = 1, \cdots m$
8      $\mathcal{C} = \mathcal{C} \setminus \{t_1, \cdots, t_m\}$;
9      foreach $t_i \in \mathcal{P}$ do
10         $c_i := t_i$;
11         while $|c_i| < k$ do
12             $t_j$ =find_best_tuple($\mathcal{C}, c_i$) w.r.t d;
13             $\mathcal{C}_i := (\mathcal{C} \setminus \{t_j\})$;
14             $c_i := (c_i \cup \{t_j\})$;
15         $\mathcal{C}_i := \mathcal{C} \cup c_i$;
16         $u := utility(\mathcal{C}_i, R, \Sigma)$;
17         if $u > MaxU$ then
18             $MaxU := u$;  $\mathcal{C}' := \mathcal{C}_i$;
19     for cluster $c$ in $\mathcal{C}' \setminus \mathcal{C}$ do
20         $S := S \cup c$
21     $\mathcal{C} := \mathcal{C}'$;
22 return $R^{\mathcal{C}}$;
```

a clustering with better utility compared to the clustering generated with smaller m value. However, the complexity is the increase in the runtime because of the increased number of clusterings generated and evaluated in each iteration.

ID	**PC**	**PRV**	**MED**	**DIAG**
t_1	V6J	BC	ibuprofen	osteoarthritis
t_2	V6J	BC	ibuprofen	osteoarthritis
t_3	J5B	QC	ibuprofen	osteoarthritis
t_4	K2H	ON	ibuprofen	osteoarthritis
t_5	K2H	ON	diazepam	seizure
t_6	J5B	QC	diazepam	seizure
t_7	R7A	MB	diazepam	seizure
t_8	R7A	MB	diazepam	seizure

TABLE 20: Original table R.

ID	**PC**	**PRV**	**MED**	**DIAG**
$c_{1,2}$	V6J	BC	ibuprofen	osteoarthritis
$c_{1,2}$	V6J	BC	ibuprofen	osteoarthritis
t_3	J5B	QC	ibuprofen	osteoarthritis
t_4	K2H	ON	ibuprofen	osteoarthritis
t_5	K2H	ON	diazepam	seizure
t_6	J5B	QC	diazepam	seizure
t_7	R7A	MB	diazepam	seizure
t_8	R7A	MB	diazepam	seizure

TABLE 21: Clustering $\mathcal{C}_1$.

ID	**PC**	**PRV**	**MED**	**DIAG**
t_1	V6J	BC	ibuprofen	osteoarthritis
t_2	V6J	BC	ibuprofen	osteoarthritis
t_3	J5B	QC	ibuprofen	osteoarthritis
$c_{4,5}$	K2H	ON	*	*
$c_{4,5}$	K2H	ON	*	*
t_6	J5B	QC	diazepam	seizure
t_7	R7A	MB	diazepam	seizure
t_8	R7A	MB	diazepam	seizure

TABLE 22: Clustering $\mathcal{C}_4$.

ID	**PC**	**PRV**	**MED**	**DIAG**
$c_{1,2}$	V6J	BC	ibuprofen	osteoarthritis
$c_{1,2}$	V6J	BC	ibuprofen	osteoarthritis
$c_{3,6}$	J5B	QC	*	osteoarthritis
$c_{4,5}$	K2H	ON	*	osteoarthritis
$c_{4,5}$	K2H	ON	*	seizure
$c_{3,6}$	J5B	QC	*	seizure
$c_{7,8}$	R7A	MB	diazepam	seizure
$c_{7,8}$	R7A	MB	diazepam	seizure

TABLE 23: 2-anonymous public view R''.

Example 4.3. *(ex. 1.2 cont.) Let* $m = 2$ *and let* t_1, t_4 *be the two random tuples selected by k-ASSEMBLE from the relation R in Table 20 in the first iteration. The algorithm generates the clusterings* $\mathcal{C}_1$ *and* $\mathcal{C}_4$*, with the former containing the cluster* $c_{1,2} = \{t_1 \cup t_2\}$ *and the latter contains the cluster* $c_{4,5} = \{t_4 \cup t_5\}$*. Clusters* $c_{1,2}$ *and* $c_{4.5}$ *are formed by merging the selected tuples* t_1 *and* t_4 *with their corresponding nearest neighbours* t_1 *and* t_5 *to generate clusters of size* $k = 2$*. For the clustering* $\mathcal{C}_1$*, utility*$(\mathcal{C}_1, R, \Sigma) = 2 + (4 \times 0.5) = 4$ *since two instances* $p_1 = (\mathsf{V6J}, \mathsf{BC})$ *and* $p_5 = (\mathsf{ibuprofen}, \mathsf{osteoarthritis})$ *will be completely preserved by* $c_{1,2}$ *with preserve*$(c_{1,2}, p_1) =$ *preserve*$(c_{1,2}, p_5) = 1$ *and the remaining 4 instances are partially preserved with preserve score of* 0.5*. For the clustering* $\mathcal{C}_4$*, utility*$(\mathcal{C}_4, R, \Sigma) = 1 + (5 \times 0.5) = 3.5$ *as for only one instance* $p_3 = (\mathsf{K2H}, \mathsf{ON})$*, preserve*$(c_{4,5}, p_3) = 1$ *and the preserve score for the remaining* 5 *instances is* 0.5*. k-ASSEMBLE chooses* $\mathcal{C}_1$ *as an input to the next iteration since* $\mathcal{C}_1$ *better preserves dependency instances (utility*$(\mathcal{C}_1, R, \Sigma) = 4$*) over* $\mathcal{C}_4$ *(utility*$(\mathcal{C}_4, R, \sigma) = 3.5$*). The algorithm seals the tuples in the cluster* $c_{1,2}$ *since* $|c_{1,2}| = k$ *and continues by replacing* $\mathcal{C}$ *with* $\mathcal{C}_1$*. In iteration 2, let* t_5 *and* t_7 *be the two random tuples selected by k-ASSEMBLE from the set of unsealed tuples* $\{t_3, t_4, t_5, t_6, t_7, t_8\}$ *in Table 20 to generate the clustering* $\mathcal{C}_5$ *and* $\mathcal{C}_7$*. Following the same steps, k-ASSEMBLE generates the final k-anonymous relation* R'' *as shown in Table 23, with dependency loss* $\Delta(R'', R, \Sigma) = 0$ *since all the six dependency instances in* $\Gamma = \{p_1, \cdots, p_6\}$ *appear at least once in R" without any distortion.*

4.3 Analysis

PAIR-ENUM and k-ASSEMBLE always terminate after a finite number of iterations and returns a final k-anonymous public view $R^{\mathcal{C}}$. In PAIR-ENUM, the while loop in line 4 terminates because, in each iteration, it merges two clusters and the resulting clusters are sealed if they have at least k tuples. k-ASSEMBLE terminates as in each iteration (line 4), a cluster of size k gets sealed and is added to $\mathcal{C}$ that eventually covers all the tuples in R. The worst-case running time of PAIR-ENUM is $O(|R|^4 \times |\Gamma|)$ since there are at most $|R|$ iterations and each iteration considers $O(|R|^2)$ candidate pairs, and computes utility in $O(|R| \times |\Gamma|)$. The runtime complexity of k-ASSEMBLE is $O(m \times \frac{|R|^2}{k} \times |\Gamma|)$ since there are $\frac{|R|}{k}$ iterations and each iteration considers at most m clusterings and computes the utility of the dependency instances in $O(|R| \times |\Gamma|)$.

Chapter 5

Optimizations

We propose two optimizations to improve the performance of PAIR-ENUM and k-ASSEMBLE to $O(|R|^2 \times k \times |\Gamma|)$ and $O(m \times |R| \times |\Gamma|)$ respectively.

The worst-case running time depends on (i) the number of iterations in the for loop (line 6), that is the number of possible cluster pairs considered at each step in the PAIR-ENUM algorithm, and (ii) the cost of computing *utility* in both PAIR-ENUM (line 11) and k-ASSEMBLE (line 16). To scale-up the algorithm, we try to optimize with respect to both (i) and (ii).

5.1 Opt-1: Pruning Cluster Pairs.

For (i), we apply an optimization that reduces the number of candidate cluster pairs in line 6 of PAIR-ENUM. The idea is for each cluster $c \in \mathcal{C}$, we index its q closest clusters according to the distance measure defined in Definition 7 and consider only those candidates as possible merges with c at each iteration. This optimization reduces the $O(|R|^2)$ candidate pairs to $O(|R|)$.

5.2 Opt-2: Re-computing the Utility Efficiently.

To reduce the cost of computing the *utility* for clustering $\mathcal{C}$, for each dependency instance $p \in \Gamma$, we index $M > 2 \times k$ tuples in a map $\mathcal{U}$, holding the dependency instances similar to the instance p and use $\mathcal{U}$ to efficiently compute the utility of a new clustering $\mathcal{C}_{ij}$. Merging c_i and c_j results in new clustering $C_{i,j}$ and while computing the *utility*, we can find the new best tuples that preserve p by looking at $\mathcal{U}[p]$ and the tuples in the result of merging c_i and c_j. Since $M > 2 \times k$, the tuple that better preserves the instance p is still in $\mathcal{U}[p] \cup c_i \cup c_j$. Using $\mathcal{U}$, the worst-case running time for computing the *utility* measure will reduce from $O(|R| \times |\Gamma|)$ to $O(k \times |\Gamma|)$.

Chapter 6

Experiments

We evaluate our two clustering-based generalization algorithms using two real data sets. Our evaluation focuses on the following objectives.

1. We study the performance to effectiveness trade-off of our algorithms as we vary the size of the data set.

2. We evaluate PAIR-ENUM and k-ASSEMBLE performance as we increase the number of FDs (Σ) in the data set.

3. We compare the performance of PAIR-ENUM and k-ASSEMBLE against two existing local re-coding algorithms as we scale the number of tuples.

4. A qualitative evaluation of the dependency loss of our algorithms as we vary the k value. We also study the impact of the data skew on the dependency loss as we vary the k value.

6.1 Datasets

Setup: All the experiments were performed on a cluster of Intel core i7-7740X CPU @ 4.30GHz with 32GB of RAM. The operating system on the machine is Red Hat Enterprise Linux and the implementation was built and run in Python 3 platform.

Table 24 provides the characteristics of two datasets w.r.t the numbers of entities (N), attributes (n), QIs ($|QI|$), set of FDs ($|\Sigma|$) and set of dependency instances (Γ).

Drug Deaths. The accidental drug deaths dataset [4] contains a listing of accidental deaths associated with drug overdose in Connecticut from 2012 to 2017. For k-anonymization, we considered sex, race, age, zipcode, residence state, residence county, death city, death county as the QIs and injury location, description of injury as the sensitive attributes. The functional dependencies (Σ) that hold over drug deaths data set are φ_1 : zipcode $\rightarrow$ residence state, φ_2 : residence county $\rightarrow$ residence state, φ_3 : death city $\rightarrow$ death county and φ_4 : location $\rightarrow$ description of injury. The VGHs of drug deaths QI attributes can be seen in Appendix A1.

Food Inspections. The New York Restaurants Food Inspections data set [20] contains inspection results of the retail food establishments in New York. Nine attributes: borough, address, violation code, violation description, zipcode, cuisine description, action, inspection type are considered as the QIs and critical flag, score, grade are the sensitive attributes. The functional dependencies (Σ) that hold over Food Inspections dataset are φ_1 : address $\rightarrow$ borough, φ_2 : violation code $\rightarrow$ violation description, φ_3 :zipcode $\rightarrow$ borough, φ_4 : cuisine description, address $\rightarrow$ borough, φ_5 : inspection type, critical flag $\rightarrow$ action and φ_6 : critical flag, score $\rightarrow$ grade. The VGHs of food inspections QI attributes can be seen in Appendix A2.

	Drug Deaths	**Food Inspections**
N	6000	30,000
n	10	11
$\|QI\|$	8	8
$\|\Sigma\|$	4	6
$\|\Gamma\|$	998	1866

Table 24: Data characteristics.

	Attribute	**Distinct Values**	**# levels in DGH^A**
1	sex	2	2
2	race	9	3
3	age	62	6
4	zipcode	220	4
5	residence state	16	3
6	residence county	55	3
7	death city	182	3
8	death county	8	3
9	injury location	4	-
10	description of injury	4	-

Table 25: Description of drug deaths dataset

	Attribute	**Distinct Values**	**# levels in DGH^A**
1	borough	5	2
2	address	506	4
3	violation code	84	4
4	zipcode	139	4
5	violation description	81	4
6	cuisine description	63	3
7	action	5	2
8	critical flag	3	-
9	inspection type	28	3
10	score	100	-
11	grade	4	-

TABLE 26: Description of food inspections dataset

6.2 Performance to Effectiveness Trade-off

We run the algorithms three times and take the average as the reported running time we set $k = 5$ as default. The implementation of Baseline refers to PAIR-ENUM without optimizations. PAIR-ENUM includes Opt-1 to reduce the number of possible cluster pairs considered at each step. Both PAIR-ENUM and k-ASSEMBLE include Opt-2 to reduce the cost of computing utility.

Figure 7 reports the execution time behaviors of our algorithms for various cardinalities on the food inspections dataset. The Baseline and Baseline+Opt2 runtime increase exponentially because these algorithms evaluate all the possible pairwise clusters in each iteration to generate an optimal output. We terminate these algorithms after 24 hours. Opt-1 performs better over Baseline and Baseline+Opt2 because the algorithm prunes the search space and evaluates only reduced candidate cluster pairs in each iteration. The PAIR-ENUM and k-ASSEMBLE runtime increase linearly with the size of the data set because of the increased number of dependency instances that must be preserved. These algorithms are about 1.8× and 6.3× faster than Opt-1 because they apply both Opt-1 and Opt-2 that prunes the cluster pairs and recomputes the utility of the clusterings efficiently.

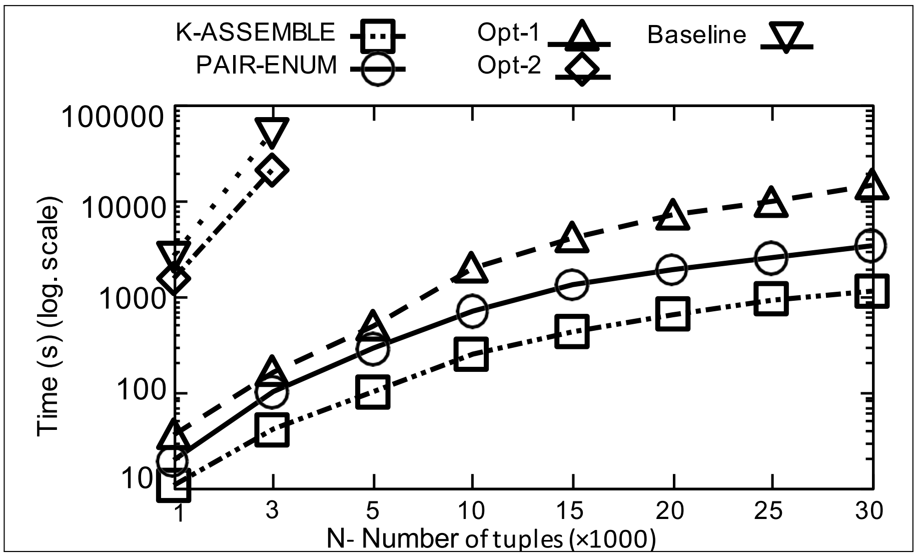


FIGURE 7: Performance to effectiveness trade-off - food inspections dataset.

Figure 8 shows the loss of dependency instances (Δ) in R' from R for various cardinalities on food inspections data. As shown in the figure, the dependency loss (Δ) decreases with the increase in the size of the dataset because larger the number of records, greater the flexibility to select the clusters that better preserve the dependency instances and thereby minimizing the dependency loss(Δ). The figure also highlights the trade-off between performance and data quality. The improvement in runtime comes at a cost of 3.2% dependency loss for PAIR-ENUM and k-ASSEMBLE suffers a dependency loss of 5.3%.

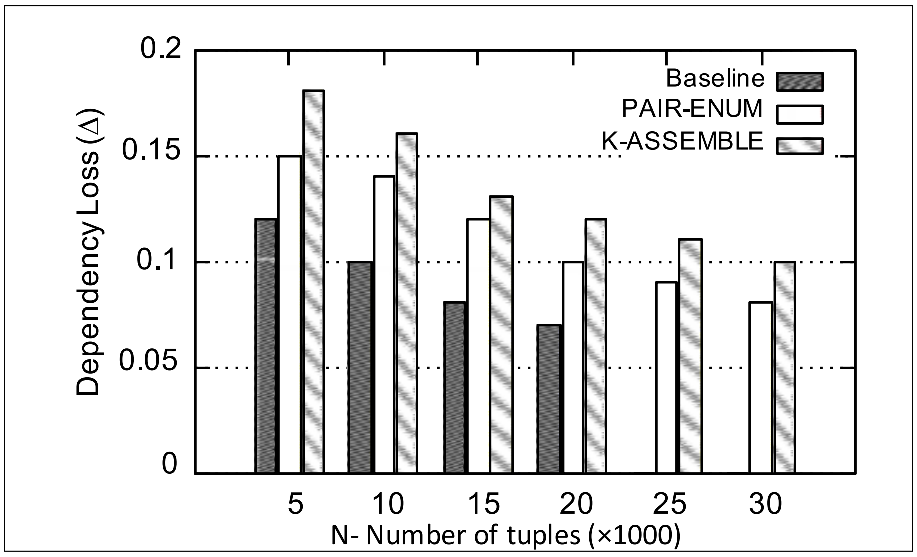


FIGURE 8: Performance to effectiveness trade-off - food inspections dataset.

6.3 Performance

Figure 9 reports the performance of k-ASSEMBLE and PAIR-ENUM for the increasing the number of functional dependencies (Σ) using food inspections data. As shown, the runtime for both algorithms increases with larger $|\Sigma|$ since a larger number of dependency instances needs to be evaluated and preserved with minimal loss. The PAIR-ENUM is 2× slower than k-ASSEMBLE because in each iteration, PAIR-ENUM enumerates all the possible cluster pairs and re-computes the utility of each cluster pair.

In Figure 10, we compare the performance of k-ASSEMBLE and PAIR-ENUM against the existing algorithms: *k*-member [3] and Top-down [25] using the food

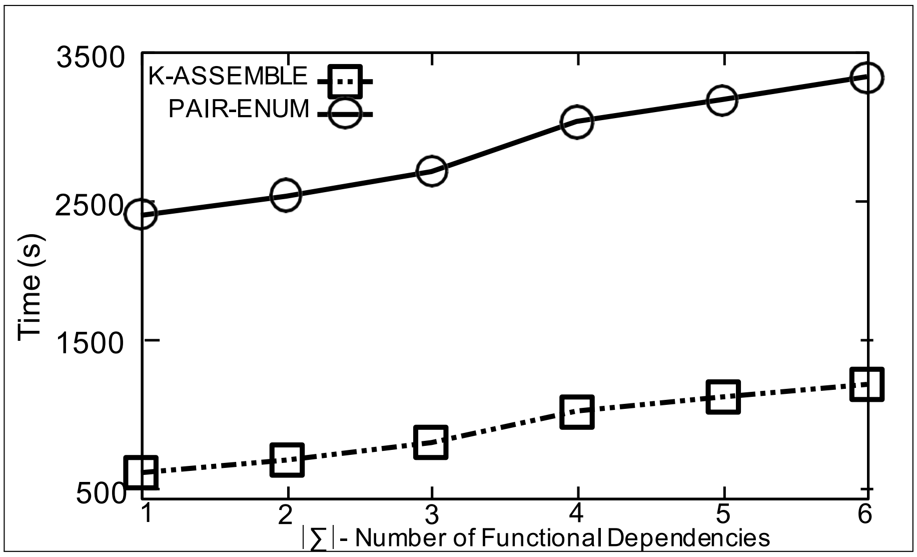


FIGURE 9: Performance evaluation by varying number of functional dependencies - food inspections dataset.

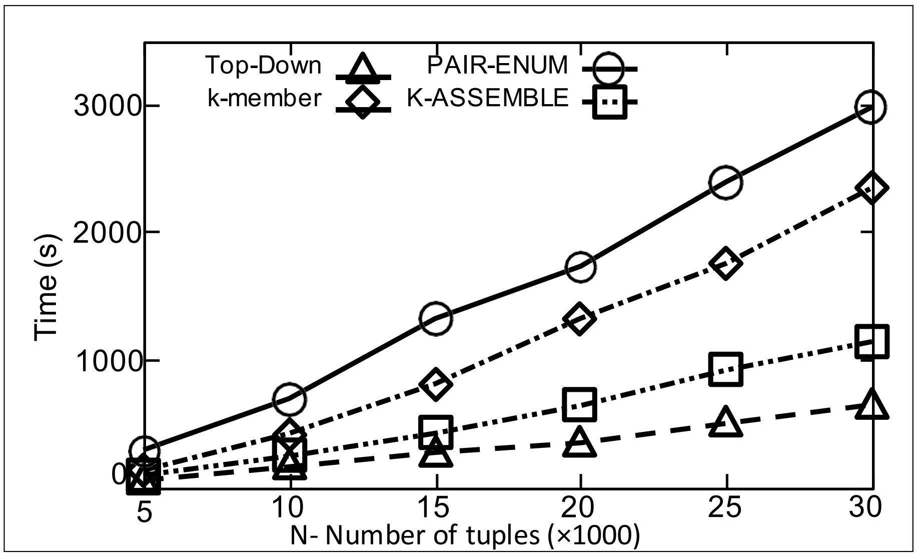


FIGURE 10: Performance comparison with varied number of tuples - food inspections dataset.

inspections data. As shown, the runtime increase almost linearly w.r.t the size of the dataset for all the four algorithms. The Top-Down performs better than the other algorithms because it focuses only on generalizing R to generate a k-anonymous public view R' with no effort in minimizing the loss of dependencies during anonymization. As a result, the difference between Top-Down and k-ASSEMBLE is about 70%. k-ASSEMBLE is 94% faster than k-member. We also note that in k-ASSEMBLE, pool size parameter m acts as a trade-off between the data quality and runtime. For the larger m value, a large number of clusters are considered and evaluated in each iteration, thereby providing greater flexibility to generate the output with minimal dependency loss. PAIR-ENUM is about 34% slower than k-member due to the enumeration of pairwise cluster candidates and re-computation of their *utility*.

6.4 Minimizing Dependency Loss

Figure 11 shows the dependency loss (Δ) of PAIR-ENUM and k-ASSEMBLE for different values of k using the drug deaths data. As shown, the dependency loss increases with the increase in the k value because more generalization is needed to satisfy the more restrictive privacy requirement imposed by the larger k values and thereby increasing the dependency loss (Δ) in R'. PAIR-ENUM is able to generate instances with less distortion (4.5% on an average) than k-ASSEMBLE for $k > 5$ because in each iteration, PAIR-ENUM extensively evaluates possible candidate cluster pairs.

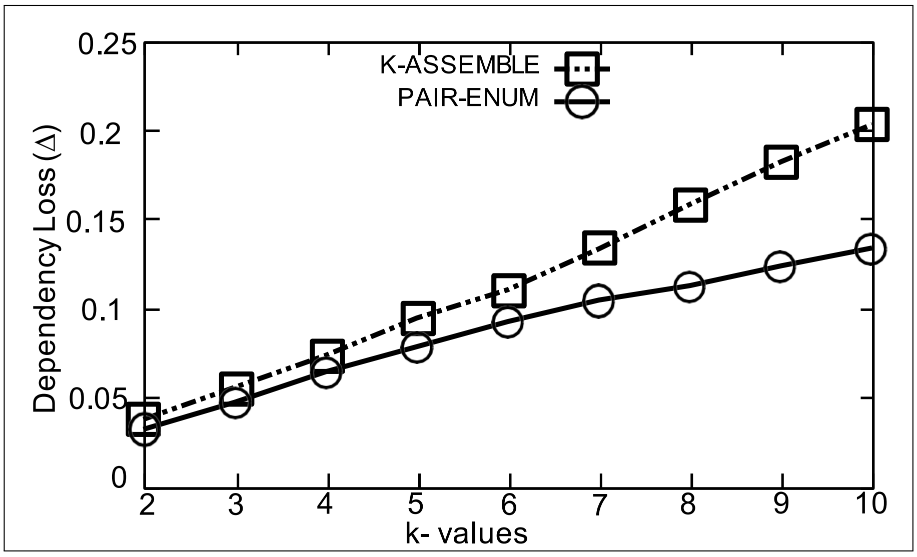


FIGURE 11: Dependency Loss for varied values of k - drug deaths dataset.

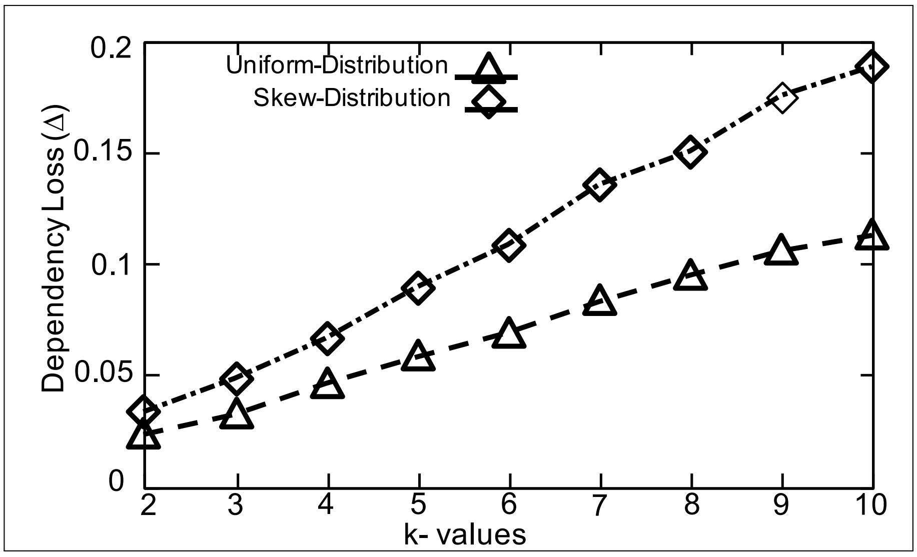


FIGURE 12: Skewed data impact on Dependency Loss for varied values of k - drug deaths dataset.

In Figure 12, we study the impact of the data skew on the dependency loss (Δ) for different values of k by comparing the original skewed drug deaths data to a synthetically generated data with the uniform distribution of data values. As expected, PAIR-ENUM generated instances with less dependency loss using uniform data (6% on an average) over the original skewed data. This is because, in the skewed data, the distribution of the dependency instances is not uniform and therefore, it is hard to preserve those infrequent dependency instances during anonymization with minimal loss.

Chapter 7

Conclusion and Future Work

In this thesis, we presented a new problem in the area of PPDP that aims to preserve data dependencies while publishing a dataset. We defined the dependency loss penalty measure that quantifies the loss of dependencies due to generalization in the published dataset. We proposed two cluster-based generalization algorithms to solve the dependency preserving generalization problem and optimizations to improve the performance of the algorithms. Our experimental evaluation shows the algorithms comparative performance and minimal distortion w.r.t *dependency loss* metric using two real datasets.

As next steps, we intend to study:

1. We aim to extend the dependency loss penalty measure to general penalty definition that considers the loss of general patterns (e.g. values) that are specified by queries. The penalty in such case depends on the distortion in the query answers when the query is answered on the anonymized public version. Dependency loss will be a special case when the query is a projection

query over the attributes that participate in the functional dependencies.

2. The clusters/QI-groups generated by our algorithms overcome record linkage attack. However, if the entire cluster has the same sensitive value, it may aid an adversary to compromise an individual's sensitive information. To overcome this limitation, a future extension is towards l-diversity. A public view of a relation R is said to have l-diversity if every cluster/QI-group of this view has at least l well represented distinct sensitive attribute values. In the k-anonymous public view generated by our algorithm, first, we would remove the clusters that do not satisfy l-diversity requirement. Next, add the tuples in these clusters to other clusters that already satisfy l-diversity requirement which cause least dependency loss. The other way is injecting noise to the sensitive attribute values in these clusters such that we obtain l-distinct values.

3. In this work, we studied the problem of dependency-preserving generalization of data that holds functional dependencies. In the future, we will consider the data that holds Multivalued Dependency (MVD), a type of functional dependency that exists when there are at least three attributes (X, Y, Z) in a relation R and for a single value of X, there is a defined set of Y values ($X \rightarrow\rightarrow Y$) and a defined set of Z values ($X \rightarrow\rightarrow Z$). However, Y and Z are independent of each other. The set of dependency instances/patterns is a projection over the attributes $\{X, Y, Z\}$ in a relation R that we need to preserve during generalization. As a future work, we will also consider Conditional Functional Dependencies (CFDs) that extend standard functional dependencies by enforcing patterns of semantically related constants, which

are used as rules for data cleaning.

4. Our model assumes that QI is known to the data publisher before data publishing. However, if an attacker has access to additional attributes/columns (that do not appear in the original private data), then this information may aid him to compromise an individual's privacy if there exists some correlation between attributes from the published view of the original private data and the additional attributes possessed by the attacker. Let QI ={Gender, Age} and Disease be the sensitive attribute that appears in the published k-anonymous view of the original private data. If the attacker also has access to the public data with attributes Street, Disease, then he/she could possibly compromise an individual's privacy using this information along with some additional background information.

   For example, from the available public data, the attacker knows that people who live on Main Street are predisposed to Cancer. The attacker also knows Rita age to be 27 living on the Main Street. If the k-anonymous public view generated by our model has an entry {Female, [20-30], Cancer}, then this information could enable the attacker to conclude Rita most likely has Cancer since there exists a correlation between the information possessed by the attacker and the age attribute in the k-anonymous public view. To address this limitation, we could change our model to take the publicly available information into consideration while generalizing the attributes from the original private data that could be correlated. Given the publicly available information, we could condition on the attributes from the original private data that could be correlated to compute prior probability, which could possibly change the way we generalize those attributes. The impact on the performance of the model solely depends on how we make use of the publicly available information. There could be some additional cost involved if we compute conditional probability distribution on attributes from original private data that could be correlated with the publicly available data.

Bibliography

[1] Gagan Aggarwal, Feder, Kenthapadi, Motwani, Panigrahy, Thomas, and Zhu. Anonymizing tables. In *International Conference on Database Theory*, pages 246–258, 2005.

[2] Roberto J Bayardo and Rakesh Agrawal. Data privacy through optimal k-anonymization. In *International Council for Open and Distance Education*, pages 217–228, 2005.

[3] Ji-Won Byun, Ashish Kamra, Elisa Bertino, and Ninghui Li. Efficient k-anonymization using clustering techniques. In *International Conference on Database Systems for Advanced Applications*, pages 188–200, 2007.

[4] Connecticut Open Data. Accidental drug related deaths 2012-2017. `https://data.ct.gov`, 2018.

[5] Cynthia Dwork. Differential privacy. In *International Colloquium on Automata, Languages, and Programming*, pages 1–12, 2006.

[6] Khaled El Emam and Dankar. A globally optimal k-anonymity method for the de-identification of health data. *Journal of the American Medical Informatics Association*, 16(5):670–682, 2009.

[7] Benjamin Fung, Ke Wang, and S Yu Philip. Anonymizing classification data for privacy preservation. *Transactions on Knowledge and Data Engineering*, 19(5):711–725, 2007.

[8] Benjamin Fung, Ke Wang, and Philip S Yu. Top-down specialization for information and privacy preservation. In *International Conference on Data Engineering*, pages 205–216, 2005.

[9] Aristides Gionis and Tamir Tassa. k-anonymization with minimal loss of information. *Transactions on Knowledge and Data Engineering*, 21(2):206–219, 2009.

[10] Vijay Iyengar. Transforming data to satisfy privacy constraints. In *Special Interest Group on Knowledge Discovery and Data Mining*, pages 279–288, 2002.

[11] Florian Kohlmayer and Prasser. Flash: efficient, stable and optimal k-anonymity. In *Privacy, Security, Risk and Trust*, pages 708–717, 2012.

[12] Kristen LeFevre, David J DeWitt, and Raghu Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *Special Interest Group on Management of Data*, pages 49–60, 2005.

[13] Kristen LeFevre, David J DeWitt, and Raghu Ramakrishnan. Mondrian multidimensional k-anonymity. In *International Council for Open and Distance Education*, pages 25–25, 2006.

[14] Jiuyong Li, Raymond Chi-Wing Wong, Ada Wai-Chee Fu, and Jian Pei.

Anonymization by local recoding in data with attribute hierarchical taxonomies. *Transactions on Knowledge and Data Engineering*, 20(9):1181–1194, 2008.

[15] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *International Conference on Data Engineering*, pages 106–115, 2007.

[16] Jun-Lin Lin and Meng-Cheng Wei. An efficient clustering method for k-anonymization. In *Privacy and Anonymity in Information Society*, pages 46–50, 2008.

[17] Ashwin Machanavajjhala, Kifer, Gehrke, and Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *International Conference on Data Engineering*, 1(1), 2007.

[18] Adam Meyerson and Ryan Williams. On the complexity of optimal k-anonymity. In *Principles of Database Systems*, pages 223–228, 2004.

[19] Ercan Nergiz and Chris Clifton. Thoughts on k-anonymization. *Transactions on Knowledge and Data Engineering*, 63(3):622–645, 2007.

[20] NYC OpenData. Dohmh new york city restaurant inspection results. `https://opendata.cityofnewyork.us`, 2018.

[21] Pierangela Samarati. Protecting respondents identities in microdata release. *Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.

[22] Latanya Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):571–588, 2002.

[23] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[24] Ke Wang, Philip S Yu, and Sourav Chakraborty. Bottom-up generalization: A data mining solution to privacy protection. In *International Conference on Data Mining*, pages 249–256, 2004.

[25] Jian Xu, Wei Wang, Jian Pei, Xiaoyuan Wang, Baile Shi, and Ada Wai-Chee Fu. Utility-based anonymization using local recoding. In *Special Interest Group on Knowledge Discovery and Data Mining*, pages 785–790, 2006.

Appendix A1

Drug Deaths Dataset

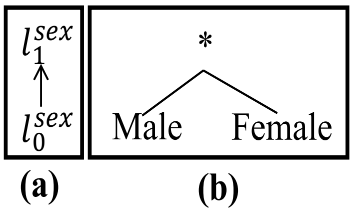


Figure A1.1: DGH and VGH of sex.

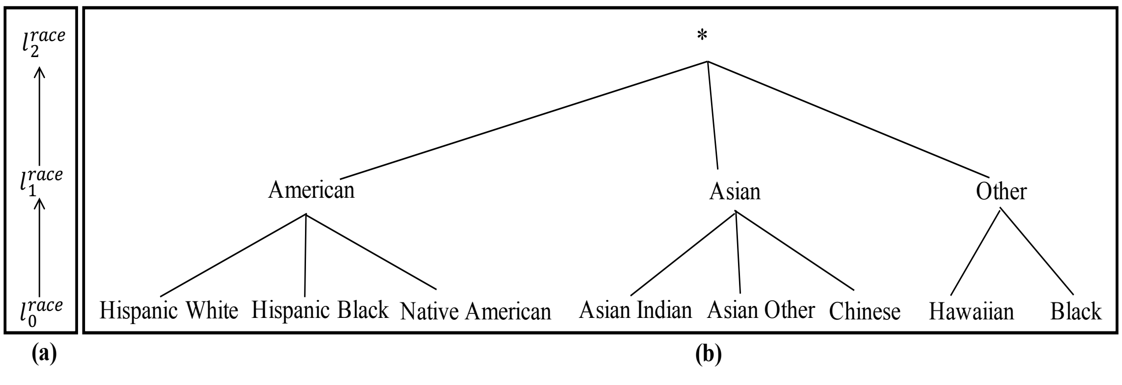


Figure A1.2: DGH and VGH of race.

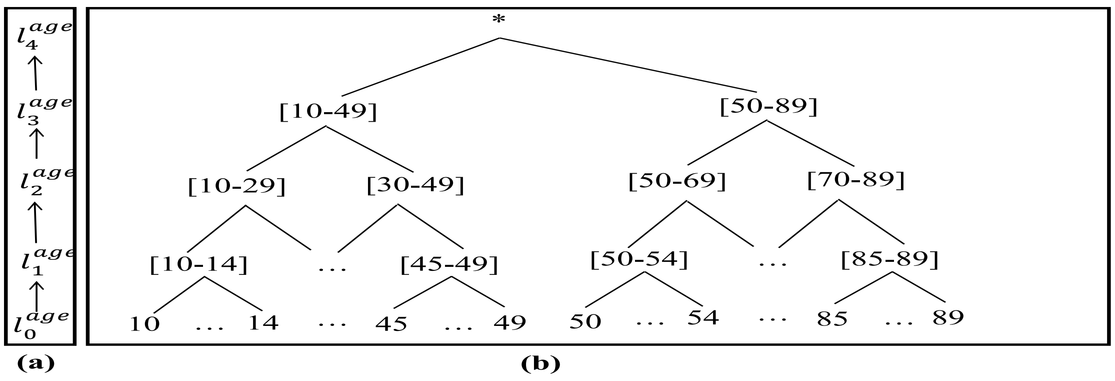


Figure A1.3: DGH and VGH of age.

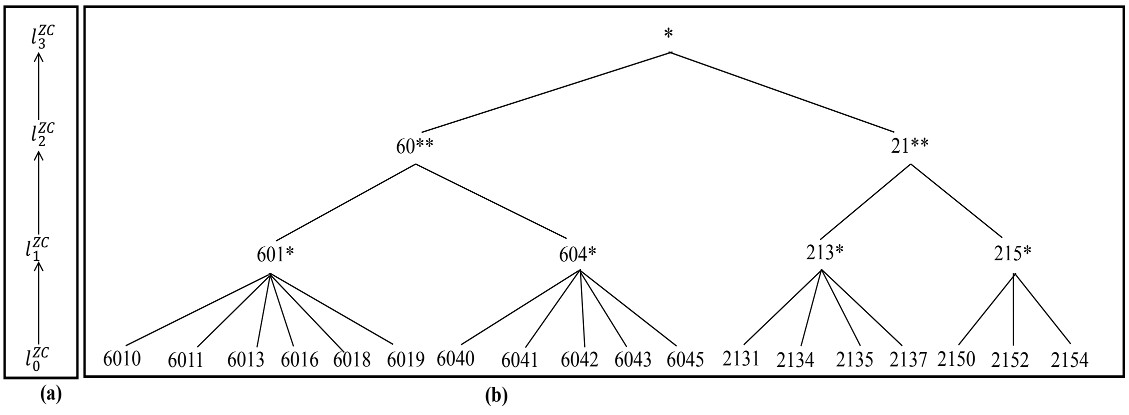


Figure A1.4: DGH and VGH of zipcode (ZC).

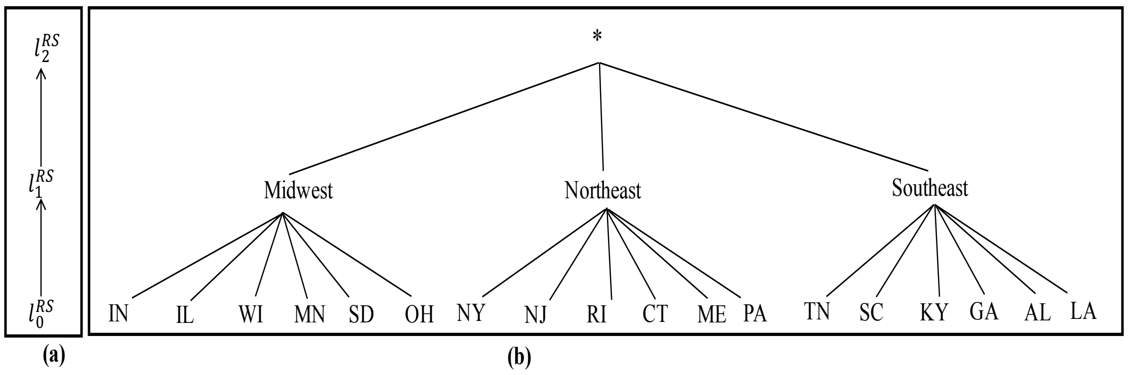


Figure A1.5: DGH and VGH of residence state (RS).

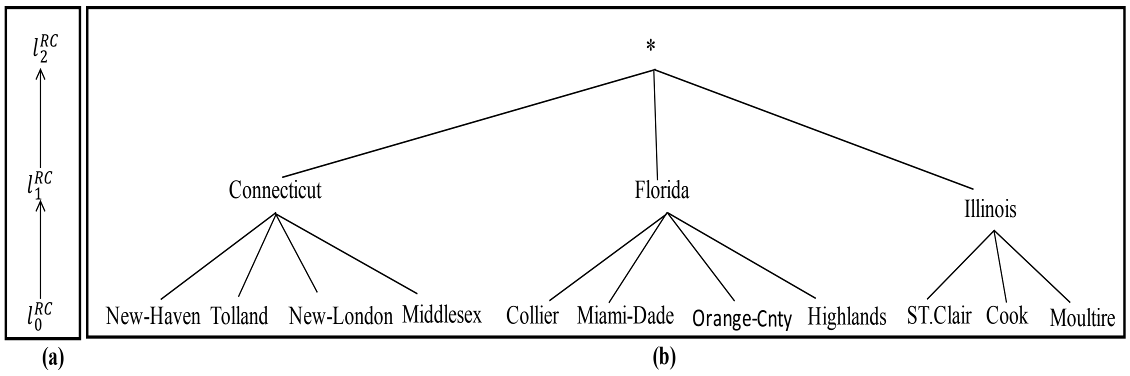


Figure A1.6: DGH and VGH of residence county (RC).

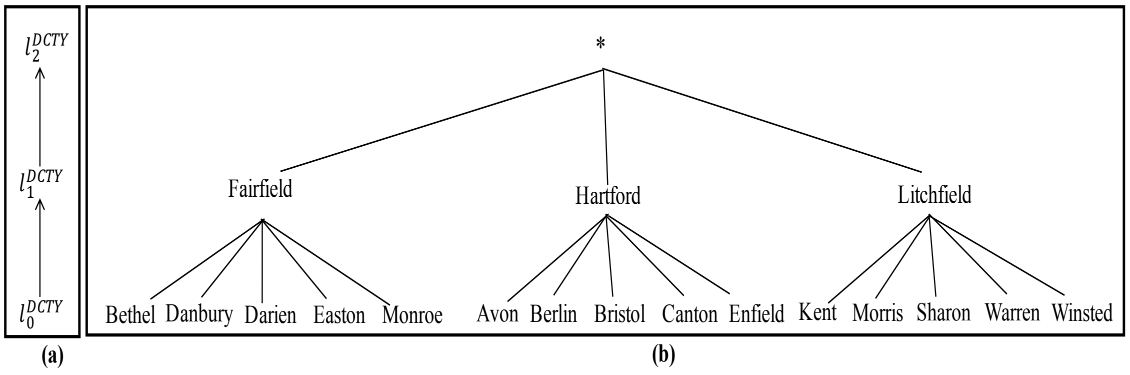


Figure A1.7: DGH and VGH of death city (DCITY).

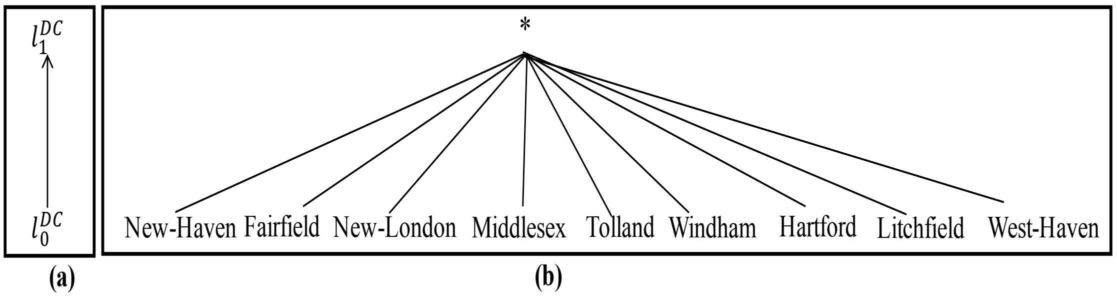


Figure A1.8: DGH and VGH of death county (DC).

Appendix A2

Food Inspections Dataset

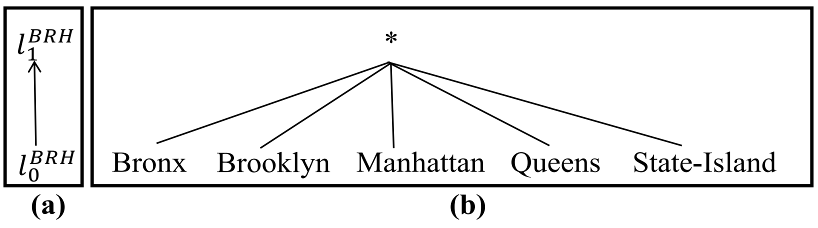


FIGURE A2.1: DGH and VGH of borough (BRH).

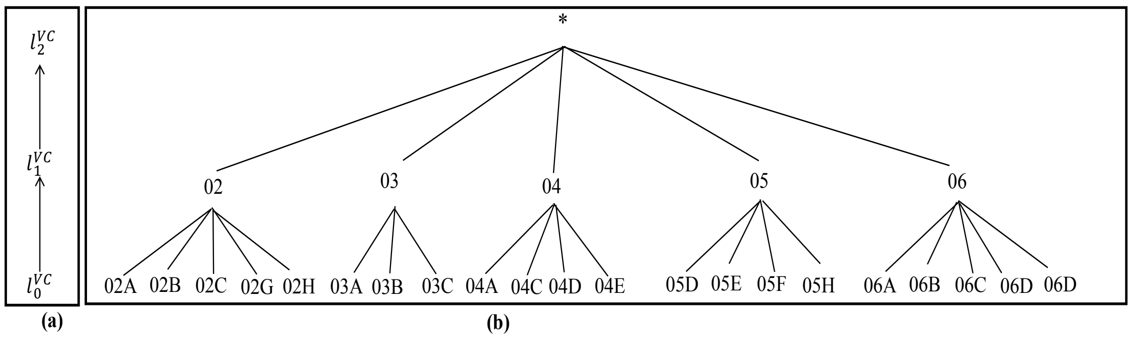


FIGURE A2.2: DGH and VGH of violation code (VC).

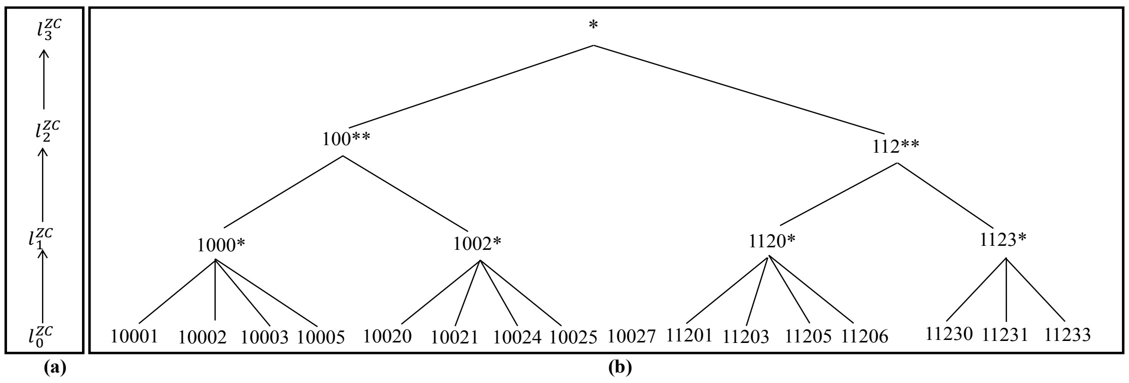
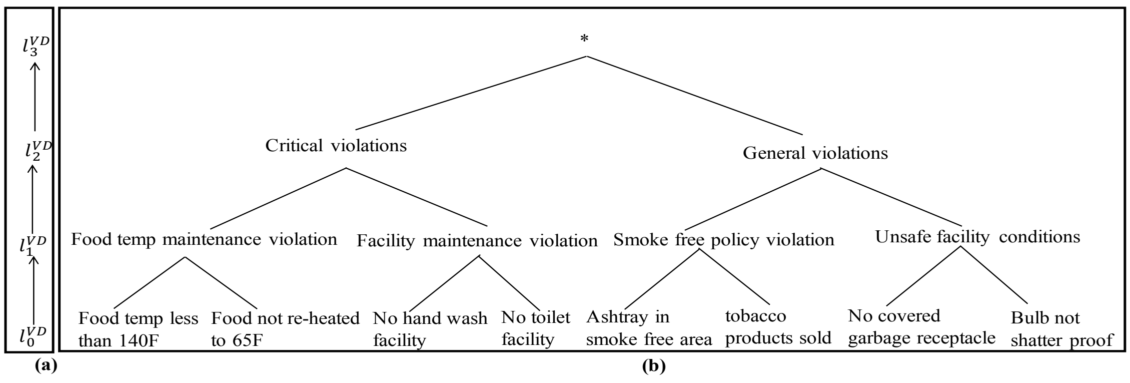
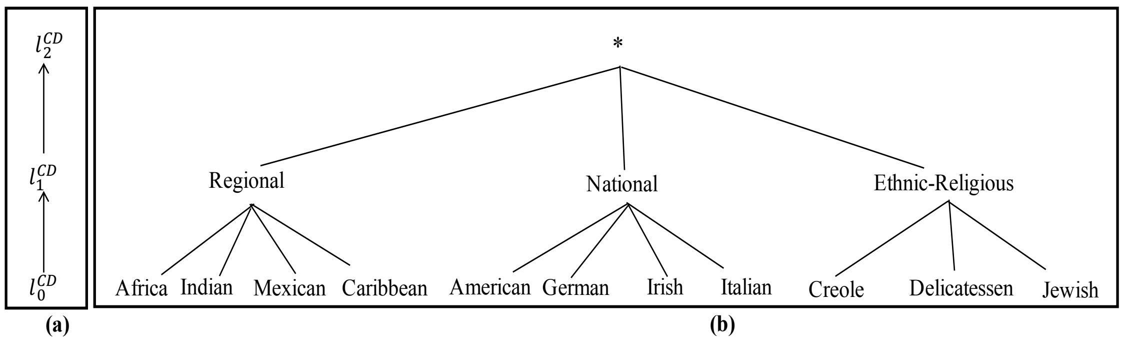


FIGURE A2.3: DGH and VGH of zipcode (ZC).



l_3^{VD}
l_2^{VD}
l_1^{VD}
l_0^{VD}
(a)
*
Critical violations
General violations
Food temp maintenance violation
Facility maintenance violation
Smoke free policy violation
Unsafe facility conditions
Food temp less than 140F
Food not re-heated to 65F
No hand wash facility
No toilet facility
Ashtray in smoke free area
tobacco products sold
No covered garbage receptacle
Bulb not shatter proof
(b)

FIGURE A2.4: DGH and VGH of violation description (VD).



l_2^{CD}
l_1^{CD}
l_0^{CD}
(a)
*
Regional
National
Ethnic-Religious
Africa Indian Mexican Caribbean American German Irish Italian Creole Delicatessen Jewish
(b)

FIGURE A2.5: DGH and VGH of cuisine description (CD).

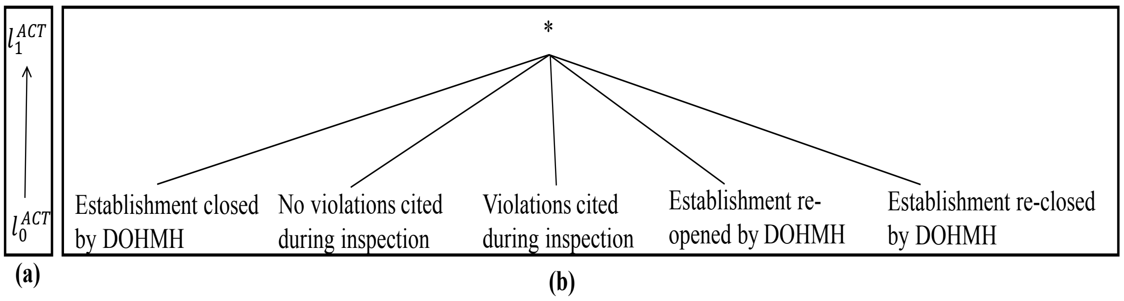

l_1^{ACT}
l_0^{ACT}
(a)
*
Establishment closed by DOHMH
No violations cited during inspection
Violations cited during inspection
Establishment re-opened by DOHMH
Establishment re-closed by DOHMH
(b)

Figure A2.6: DGH and VGH of action (ACT).

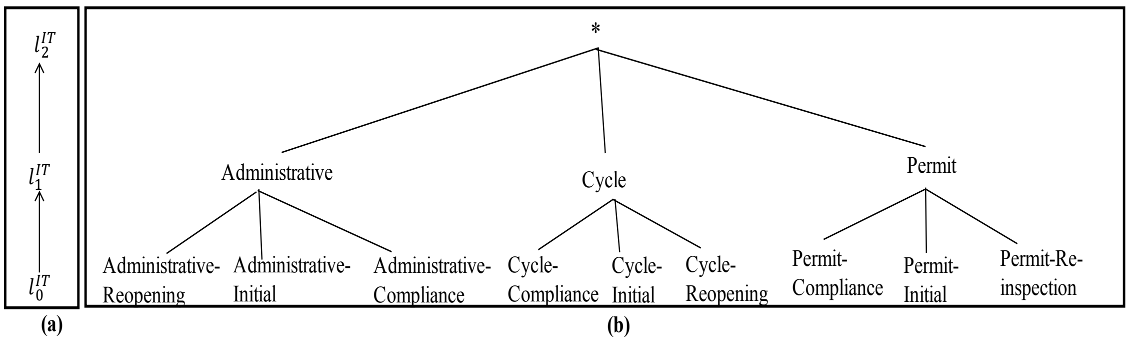


Figure A2.7: DGH and VGH of inspection type (IT).