Faculty of Science and Technology

MASTER'S THESIS

Study program/ Specialization: Computer Science	Spring semester, 2013 Open / Restricted access
Writer: Yohannes Kifle Russom	…… (Writer's signature)
Faculty supervisor: Prof. Chuming Rong Tomasz Wlodarczyk	
Title of thesis: **Privacy preserving for Big Data Analysis**	
Credits (ECTS): 30	
Key words: data anonymity , primary identifiers, quasi-identifiers, privacy, K-anonymity, Hadoop, MapReduce	Pages: 69 + enclosure: CD Stavanger, 30 June, 2013 Date/year

Front page for master thesis
Faculty of Science and Technology
Decision made by the Dean October 30th 2009

UNIVERSITY OF STAVANGER

MASTER THESIS

Privacy preserving for Big Data Analysis

Author:
Yohannes Kifle Russom

Supervisor:
Prof. Chunming Rong
Dr. Tomasz Wlodarczyk
Antorweep Chakravorty

A thesis submitted in fulfillment of the requirements
for the degree of Master in Computer Science

June 30, 2013

Acknowledgements

I would like to thank my supervisors, professors Chunming Rong and Tomasz Wlodarczyk for their valuable and interesting discussion during the weekly friday meetings and for providing us with the necessary facilities to accomplish our Master's theses.

I would like to express my deepest appreciation to Antorweep, PhD student at UiS, for his abundant help and making himself available whenever I needed him. Last but not least, I would like to thank my family, friends and specially my wife Rahel Ghebru for being patient and helpfull. Thanks as well to Stephen Michael Jothen and Kjetil Endresen for being such great friends.

Contents

List of Figures

List of Tables

Definitions

Quasi-Identifier Attribute set: Is set of attributes $(X_1, ..., X_d)$ in table T that can be joined with external information to re-identify an individual. We assume the set of quasi-identifier are defined by the administrator.

Equivalence Class: An original table T consists of a multiset of tuples. An equivalence class for T with respect to quasi-identifier attributes $(X_1, ..., X_d)$ is the set of all tuples in T containing identical values $(v_1, ..., v_d)$ for $(X_1, ..., X_d)$.

UNIVERSITY OF STAVANGER

Abstract

Department of Electrical Engineering and Computer Science

Master Of Computer Science

Privacy Preserving for Big Data Analysis

by Yohannes Kifle Russom

The Safer@Home [6] project at the University of Stavanger aims to create a smart home system capturing sensor data from homes into it's data cluster. To provide assistive services through data analytic technologies, sensor data has to be collected centrally in order to effectively perform knowledge discovery algorithms. This Information collected from such homes is often very sensitive in nature and needs to be protected while processing or sharing across the value chain. Data has to be perturbed to protect against the disclosure and misuse by adversaries. Anonymization is the process of perturbing data by generalizing and suppresing identifiers which could be a potential threat by linking them with publicly available databases. There is a great challenge of maintaining privacy while still retaining the utitlity of the data.

This thesis evaluates various anonymization methods that suits our requirements. We present the software requirement specification of an anonymization framework and provide the practical implementation of a well accepted privacy preserving anonymization algorithm called Mondrian [7]. To quantify the information loss during the anonymization process, a framework is proposed to evaluate the anonymized dataset. Moreover, it proposes the distributed method for solving the anonymization process using the Hadoop MapReduce framework to make a scalable system for big data analysis.

1

Introduction

It is expected to see the elderly population to grow rapidly in the coming few years. If the elderly are going to receive the same treatment and quality of service as today, the number of professional personnel delivering these services should be doubled. Often elderly people also prefer to live at home for several reasons. The efficient way of solving these issue would be to extend the traditional health care services to residential homes using sensor networks which monitor the movement and activity of users of the system. There have been various research about analyzing sensor from smart home [8–12]. The Safer@Home project at the University of Stavanger is one of them. This thesis paper is part of this project which specifically works on preserving the privacy of personal informations. The disclosure of personal information without using proper means of hidding informations could lead to misuse/abuse of personal information and be used by third parties for their purpose, for example advertising companies. Since data collected from users of the system could have personal informations which could track individuals out of the number of users, it becomes necessary to devise a software which could anonymize personal information and preserve its privacy.

To avoid the disclosure of personal informations unique personal identifiers like personal numbers, social security number or any other unique numbers can easily be deleted from datasets before releasing them publicly. Does deleting this unique identifers prevent the disclosure of personal informations from being de-identified? Personal data can be protected by using cryptographic algorithms to hide them from adversaries. Can we publish these data to be used by researchers or analysts? Researchers and analysts require data which is consistent and coherent, encrypting these data with cryptographic algorithms will not give them the data with their completeness/truthfulness. Other than the unique personal identifiers, datasets could hold attributes which could be a threat after being linked with other publicly available data (referred as quasi-identifiers). This data need to be properly studied on how much information could be discovered by linking this data with other publicly available informations. A better way of hiding those unique and joined attributes (quasi-identifiers) from identifying individuals is to use anonymization methods. Though there are different kinds of anonymization methods, their central objective is to protect the De-identifyability of individuals from datasets, and keep the data utility for further studies.

Some companies have inattentively published incorrectly de-identified datasets in the past years. For example, Netflix is the world's largest on-line movie rental service provider with millions of customers. The company uses movie rental and ranking records to suggest movies to users. Netflix sponsored a data-mining competition where teams with the best recommendation engine wins [13]. Unfortunately privacy concerns were discovered, and if movie recommendations are combined individuals could be traced. Moreover, AOL has also previously published a large dataset of their customers' for search queries, without knowing that most of the search queries could be traced back to a unique individual via personalized queries [14]. As a result, many companies and privacy researchers have been working for different ways of ensuring the best anonymity results without perturbing the original dataset too much.

Similar problems can occur when private organizations are working with smart home applications. These organizations use personal informations to process the number of customers using the smart home application. This thesis paper specifically studies on how to anonymize the data that is securely collected by sensors from the patients house. A number of different privacy preserving algorithms have been invented to solve the issues of linking quasi-identifiers that uniquely identifies customers. But the first issues with such kind of customer vs service provider issues are to secure the network between the two parties, storing and processing them. This part of the project is done in my previous

work, where I proposed a security architecture (Figure 2.1) and show how personal informations (personal number, age, zip code ...etc) can be processed in different modules and the other non informations to be processed in different modules. Beside that the paper clearly explains the different kinds of practical functional algorithms which have been applicable in the privacy preserving solutions. K-anonymity algorithm perturbs data in a scientifical way. The primary objective of k-anonymity is to protect the privacy of the individuals to whom the data pertains. It provides protection by making each row identical to at least k-1 other rows. However, the released data should still remain as "useful" as possible. Thus it prevents definite database linkages and at worst the data released narrows down an individual entry to a group of k individuals. Unlike other perturbation models, K-anonymity guarantees the data released is accurate.

Smart home applications require a real time sensor response algorithm which can perform data processing and give response in real time. The previous way of solving linking problems does not comply with the real time requirement of smart home applications. Most of the algorithms presented in [15] shows that k-anonymity algorithms are NP hard. To be able to use privacy preserving for smart home applications, a real time response is required.

To deal with the real time anonymization system, this thesis proposes the software engineering requirements for building an anoymization software and practical implementation of the latest K-anonymity algorithm called the Mondrian algorithm. The Mondrian algorithm uses the greedy algorithm for K-anonymization which has a number of advantages over the existing practical algorithms. These are:

- Greedy algorithms are more efficient than other practical algorithms such as Datafly, μ-Argus and optimal k-anoymization. The running time of greedy algorithm is O(nlogn), whereas optimal k-anonymity algorithms are exponential in the worst case [7].
- Greedy multidimensional algorithm produce higher quality results than the optimal single-dimensional algorithms, single-dimensional heuristic [[5, 16]] and stochastic search [17] algorithms

1.0.1 Contribution

Privacy preserving for Big data analyis proposes the software enginnering requirement for building an anonymization software. Presents the design and implementation of the latest k-anonymity algorithm. To overcome delay in anonymization process I propose a new way of implementing Mondrian in distributed way using Hadoop and MapReduce. Last but not least it quantifies the information loss using different quality metrics and evaluates the results briefly.

The thesis is organized in the following way

Chapter 2. Explains the classification of anonymiziation algorithms and the need to anonymize datasets.

Chapter 3. Surveys the existing practical K-anonymity algorithms in privacy preserving data publishing, with a focus on different models of privacy.

Chapter 4. Describes the software requirement specification of anonymization software.

Chapter 5. Briefly describes the evaluation criteria for quantifying anonymized datasets.

Chapter 6. Presents the Mondrian algorithm design and implementation steps that are followed .

Chapter 7. Discuss the results obtained from the practical implementation and evaluate the information losses.

Chapter 8. Presents the conclusions of the whole master thesis paper and gives suggestion on how it can be improved in the future works.

2

Background

The Safer@Home project focuses on delivering a non-invasive solution. It analyses movement data based on optical light sensors in a household to recognize behavioral patterns. Though there exist a number of smart home projects, however they fail to the privacy and security concerns related with such projects. In my previous work on the Safer@Home project I have proposed a security architecture, which gets data from sensors, and classifies them into different modules based on their attribute type. Those attributes with type *Identifiers* and *sensitive* go to the encrypted identifiers module and the *non Identifiers* attributes go to the synonymized module. [15]

As shown in Fig 2.1 the blue dotted line indicates that users with the Doctor/Nurse profile have the access to see data which is queried from the two modules of the system. The output of those queries is, personal data included with the rest of the data. This means the profile Doctor/Nurse has the privilege to see personal identifiers with the rest of their data. But the red dotted line indicates the Researcher profile. They have limited privileges, so they can only query data from the set of Synonimized Sensor Data module. Data which is going to be released for researchers or shared with other organizations will have to be first anonymized.

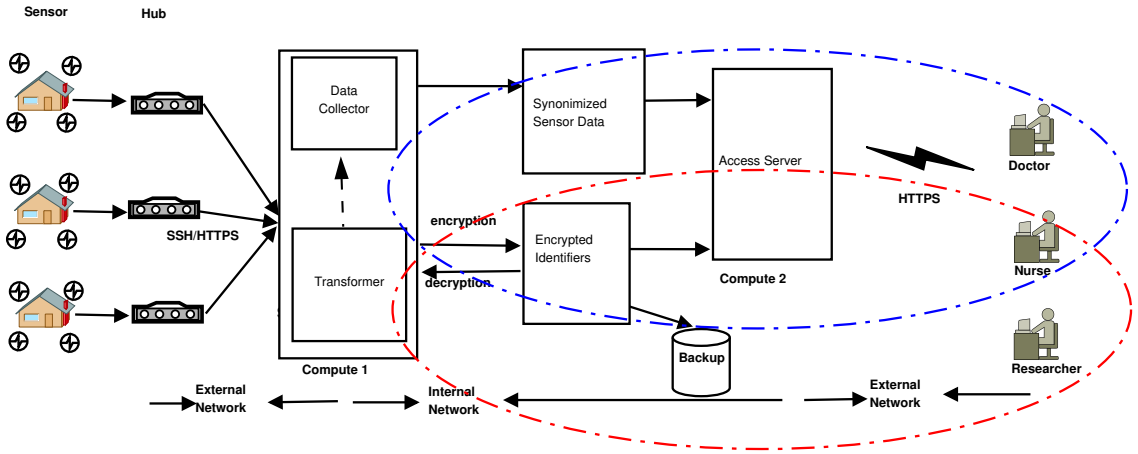

FIGURE 2.1: Secure System architecture for collecting data from sensors

There exist a number of different De-identifying anonymity models and techniques, it is important to classify the anonymization algorithms by the problems they tend to solve. There are three characteristics that can be used to classify these. Privacy, data models and quality metrics. The privacy models describe what type of data records are being anonymized, thus data depends on the data modules, it could be numeric or categorical data. These also show which attributes need to be anonymized before data is published and it can be made robust against attacks from adversaries with or with out background knowledge. Quality metrics measure the information loss after the different anonymization algorithms are applied. Currently, there is no metric that is widely accepted by the research community [18].

2.1 Privacy

There are different privacy objectives when anonymizing a dataset, These objectives are:

- Unique identity disclosure:
  - If data is published there must be no record that could identify an individual person.
- Sensitive attributes disclosure:
  - Adversaries cannot learn any sensitive information about any individual in the dataset via the disclosed attributes.[19]

2.1.1 Unique Identity Disclosure

Anonymization on a dataset is meant to protect datasets from disclosure of identities. Released datasets should not contain attributes which can uniquely identify a specific person.

Personal data can be shared with other parties or made it publicly available for research and other purposes. However such disclosure of personal information raises serious privacy concerns. To avoid such concerns, there are different kinds of privacy preserving techniques which hide the personal information which can identify individual uniquely. These techniques can be divided as heuristic and cryptographic approaches. Heuristic based techniques are designed to work for centralized data sets, whereas cryptography approaches are for the distributed scenario. Choosing the appropriate method is most important and crucial. Efficiency and scalability are the two most important factors for choosing among the set of privacy preserving algorithms in data mining. The cryptographic algorithm is found to be infeasible for our project as it has an efficiency problem. Encrypting data attributes increases data privacy it is computationally expensive to apply in big sensor data. Instead, I chose the heuristic based approach. Because the heuristic based approach can also be used in distributed systems, provided that issues that could be raised due to security reasons are covered. In my previous work on the project [15] I have shown the security architecture which could be used in this scenario.

2.1.1.1 K-anonymity

K-anonymity is one of the algorithms, tailored to solve the problem of identity disclosure. Samarati and Sweeney proposed a model for privacy protection called k-anonymity [20]. A dataset satisfies k-anonymity, if every record in the data set is identical to at least (k−1) other tuples with respect to the set of quasi-identifier attributes, and such a dataset is so-called k-anonymous. As a result, an individual is indistinguishable from at least (k−1) individuals in a k-anonymous dataset. The K-anonymity algorithm limits the ability to link or match published data with existing external information. Those attributes in the private information that could be used for linking with external data are known as the quasi-identifier. The dataset does not only include explicit unique identifiers such as personal number, security number but also include attributes such as birth date, age, ZIP Code and gender [21].

The important feature of K-anonymity is the protection techniques to preserve the data integrity and keep the data utility level high. Datasets which are anonymized reflects the anonymization of the original dataset. There is always a trade off between privacy and data utility. As data becomes more privacy preserved, the anonymized data could be less important for further research or analysis. Whereas if utility is required, then the privacy level has to be little relaxed to allow data to be used by other parties.

K-anonymity uses the generalization and suppression methods to mask the identifiability of a unique person from set of data. Generalization is replacing a value with a less specific data and without losing its meaning. Suppression is the replacing of original value by some other character or symbol which totaly replacies it. It can be used for replacing unique identifiers [5]. There are a number of advantages on using both generalization and suppression methods as a means of perturbing information. One of the most important advantages is that, data which has been changed using these methods still reflects the original data distribution. Meaning, data utility is high that these data could be used for further research or analysis by other parties.

As the name generalization indicates, the process of generalizing attributes is generalizing values by hierarchies into value which ranges from the minimum to the maximum value of the attribute, and make each tuples in same level much alike. Fig 2.2 shows the generalization sequence or a functional generalization sequence of an attribute birth date. Until the required level of K-anonymity is achieved, each of the levels of the attribute are the generalized. Fig 2.3 shows the suppression method for anonymizing the zip code attribute. In each level of the iteration, the values are made to be equal by changing the last numbers by '*'. The goal of K-anonymity is how to limit the linking of those personal informations collected from the sensors with data which are accessible from public sites. When referring to a table, each row or tuple is an order of n-values $<v_1$, v_2 ,...,$v_n >$.

2.1.2 Sensitive Attribute Disclosure

Data which is ready for release should be resistant enough that adversaries could not link them with information from public. For example , if we have a dataset of attributes such as Name, Age, Sex, Zip and Disease. The Name and disease attribute are sensitive ones , because if adversaries have a prior knowledge of the a persons Name/Disease, they may track down the individual. Protecting

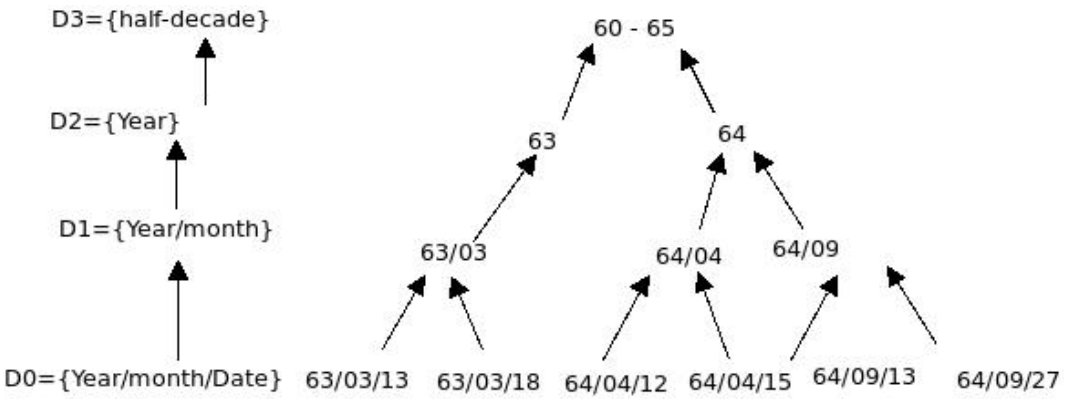

FIGURE 2.2: Birth date generalization hierarchy level

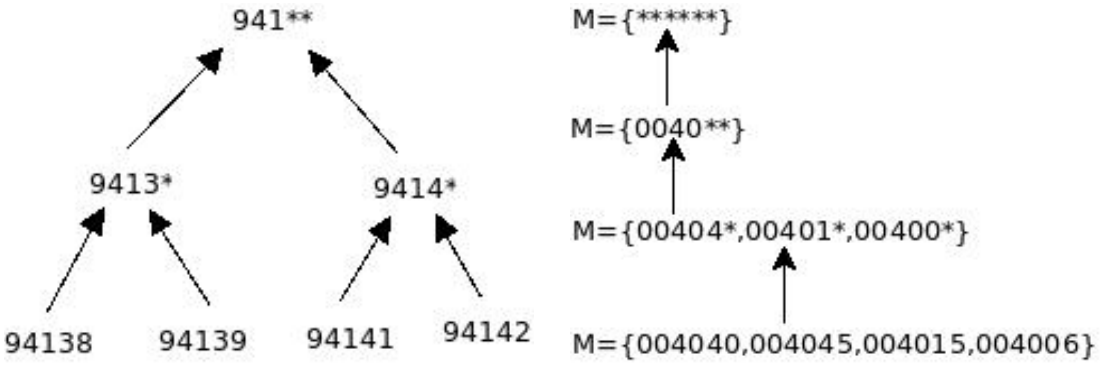

FIGURE 2.3: Zip code suppression

against sensitive data disclosure is more difficult than protecting against identity disclosure.

Published data should be prevented from an adversary using publicly available data in conjunction with a published dataset to discover sensitive data about

Name	Age	Sex	Zip Code
Martha	23	F	4041
Ken	27	M	4045
Linda	65	F	4041
Peter	35	M	4045

TABLE 2.1: Sample: Publicly available data

Attributes			2-Anonymous		Sensitive Attributes
Age	Gender	Zip Code	Gender	Zip Code	Disease
23	F	4041	F	4041	HIV
27	M	4045	M	4045	Diabetic
65	F	4041	F	4041	HIV
35	M	4045	M	4045	Flu

TABLE 2.2: Published data with sensitive attributes

an individual. For example Table 2.1 contains the public available data. If an organization releases the data in Table 2.2, an attacker could infer that Linda was treated for HIV. Because the published dataset with it's sensitive attributes have the same value of quasi-identifiers as the publicly available dataset.

Some of the existing K-anonymity algorithms that anonymize based on their sensitivity and diversity are :

- P-sensitive k-Anonymity extends the features of k-anonymity to prevent sensitive data disclosure. Each dataset in addition to fulfilling the property of k-anonymity, the sensitive attribute value must also appear at least p times in each equivalence class [22].
- Bayes-Optimal Privacy [23] is one of the privacy models known for being strong. The Bayes-Optimal formulation calculates the privacy loss after data published as the difference between the prior and posterior beliefs of a person. Their difference must be minimal.
- l-diversity address the Bayes-Optimal privacy problem that requires each equivalence class in each attribute has at least l 'well-represented' value [23]. For example lets assume we have a group of k different records that all share a particular quasi-identifier. An attacker cannot identify the individual based on the quasi-identifiers. But if the adversary is interested in knowing the sensitive attribute and all the groups have the same value then the data has leaked. This is called the homogeneity attack. The distribution of target values within a group is called l-diversity.

2.2 Data Model

One of the key factors in choosing anonymization algorithms is the data model. Data can be numeric or categorical. Examples of numerical data are such as age, salary, zip code. It is easier and more precise to anonymize datasets in numeric than the categorical datasets. Anonymization in numeric can be done using partitioning. Whereas categorical datasets are datasets with ranges of values which are discrete and ordering is not possible but they can be arranged hierarchically. The tree shows the categorical hierarchies of occupation and how it is generalized.

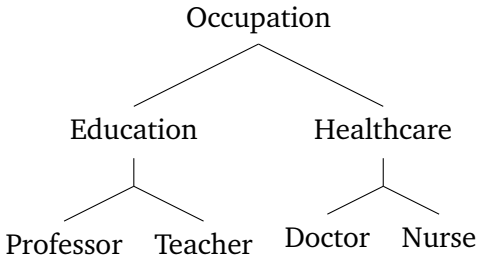

FIGURE 2.4: Categorical data type of Occupation

2.3 Quality metrics

Dataset which are anonymized need to be quantified to measure the information loss during the process of anonymization. There are a number of metrics which are proposed to quantify the information loss that anonymization introduces into a dataset. A brief description of quality metrics is given in Chapter 5.

2.4 Hadoop

Hadoop [24] is an open-source Java framework for large-scale data processing and querying vast amount of data across clusters of computers. It is an Apache project initiated and led by Yahoo. It is mostly inspired by Googles MapReduce and GFS (Google File System). Hadoop is immensely used by big companies like Yahoo, Facebook, Microsoft, and Amazon.

Hadoop partitions large amounts of data and performs computation across the nodes in parallel. By using Hadoop we can harness the performance in computing very large amouns of data and preserve network bandwidth.

Some of the key features of Hadoop are :

- **Cost effective:** Hadoop computes in parallel commodity servers. This results in lowering the cost per terabyte of storage, which in turn makes Hadoop more affordable.
- **Scalable:** New nodes can be added as per the requirment of the system.
- **Flexible:** There is no defined schema in Hadoop. It can process any type of data, whether structured or not from any number of sources. Data from multiple sources can be merged and aggregated to give fast and reliable statistical analysis.

- **Fault tolerant** If a node stops working the system redirects the work to another location of the data and pursues its processing without losing any data.

Under Apache Hadoop there are two main subprojects, which have made the Hadoop framework much more suitable for practical use. These are: MapReduce and HDFS (Hadoop Distributed File System).

2.4.1 Hadoop Distributed File System (HDFS)

HDFS [25] is a distribued file system which is designed to run in commodity hardware. It is highly fault-tolerant and can be deployed with low-cost hardware. It has alot of similarities with other distributed file systems. But it has some features which make it unique. These are HDFS's [25] write-once-read-many model that relaxes concurrency control requirements, simplifies data coherency, and enables high-throughput access.

2.4.2 MapReduce

MapReduce [24] is a programming model and software framework first developed by Google (Google's MapReduce paper submitted in 2004). MapReduce facilitates and simplifies the processing of big amounts of data in parallel on large-scale clusters of commodity hardware in a robust, reliable and fault-tolerant manner. It can handle petabytes of data with thousands of nodes.

MapReduce has two operations [26] :

- A map function, based on a unique key, maps each key with value.
  - **Input:** Key, value
  - **output:** list (Key2, value2)
  - Map function $\rightarrow$ Map(Key, Value) $\rightarrow$ Output (list (Key2, Value2))
- . A reduce function, input values are output values of map function
  - **Input:** (key2, list(value2))
  - **output:** $\rightarrow$ (list(key3, value3))

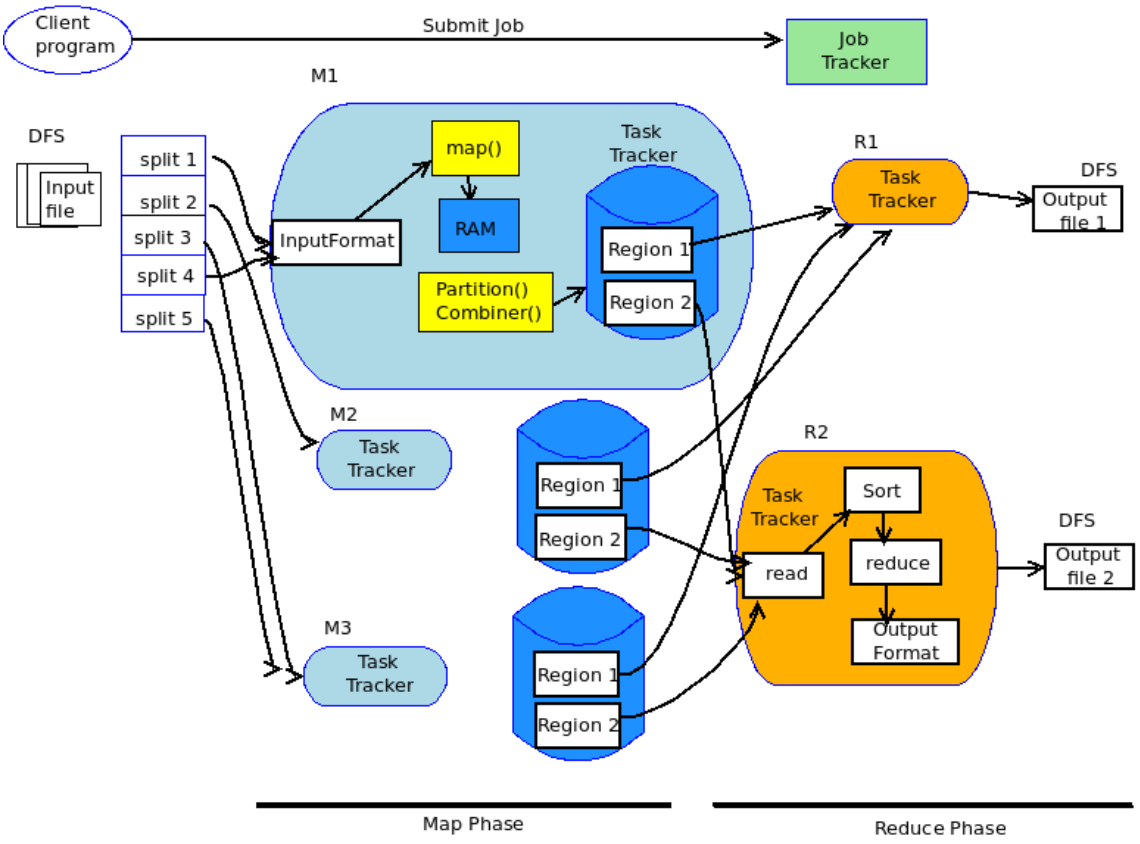

FIGURE 2.5: Hadoop MapReduce process [1]

Fig 2.5 depicts that HDFS splits the large data files into chunks which are managed by different nodes in the cluster. Each of these chunks are replicated across several machines to protect from losing data incase a machine failure occurs. For each of these chunks the mapping process is performed to allocate each record with key, value pairs. Data is distributed among the different nodes, and later on sorted and combined by their key value pairs to be reduced to one output.

3

Related Works

In a table T with attributes ranging from A_1, A_2,...,A_n, a quasi-identifier is a minimal set of attributes (A_i,...,A_{i+k}) where (k ϵ 1,...,n) and($1 \leq i \leq n$). These records can be joined with external information to re-identify individuals. In this thesis paper, the choice of quasi-identifier is maintained by the administrator based on the background knowledge.The main foucus is to anonymize the dataset based on the requirements of the k-anonymity algorithm.

A table T is said to be K-anonymous if every tuple t in T, there exist at least $(k-1)$ tuples which have the same projection with respect to a quasi-identifier, i.e if we have a dataset with attributes (A_1, A_2,...,A_n), tuples $t_{1_{A_{i_1},...,A_{i_l}}} = t_{k-1_{A_{i_1},...,A_{i_l}}}$. All those tuples which have same projection of quasi-identifiers are grouped in one, and they are know as an equivalence class.

I **Datafly**

The Datafly algorithm is one of the first practical applications of K-anonymity [5]. The algorithm approaches the K-anonymity by using full-domain generalization until every combination of the quasi-identifier is at least K times. Besides the method of generalization, the Datafly algorithm uses suppression method, which removes some tuples from the anonymous data set [5].

Following tables 3.1, 3.2 show how the Datafly algorithm performs its generalization and suppression steps to make data ready for release.

There are a few steps in Datafly: *Step 1*, constructs the frequency of unique values in each of the quasi-identifier attributes (Birthdate, Gender, and Zip), along with the number of occurrences of each sequence. *Step 2* is generalizing values starting from the attributes with the highest frequencies. The generalizing is a recursive task until the required level of k or less tuples have distinct sequences in frequency. *Step 3* is to suppress those tuples who have a frequency of less than k. The final step is to construct the table based on the number of occurences.

Table 3.1, demonstrates the Datafly algorithm having 12 unique tuples with attributes birth date, gender, zip code and disease (The last tuple shows the frequence of unique values in each column). The quasi-identifier attributes are birth date, gender and zip. The domain generalization hierarchy is done on the attribute of the birth date. Table 3.2 shows tuples with the same quasi-identifier values are grouped together and at least each group is less than or equal to K-1. Table 3.3 shows the final 2-anonymous dataset.

Birthdate	Gender	Zip code	No Of Occurs	Tuple No
16/05/1971	Male	4041	1	t1
10/08/1971	Male	4041	1	t2
19/05/1971	Female	4041	1	t3
20/04/1971	Female	4026	1	t4
03/05/1970	Female	4026	1	t5
01/10/1970	Female	4026	1	t6
18/05/1970	Male	4026	1	t7
10/03/1971	Female	4029	1	t8
09/08/1970	Male	4029	1	t9
01/07/1970	Male	4029	1	t10
09/02/1975	Male	4026	1	t11
19/03/1975	Male	4026	1	t12
12	2	3		

TABLE 3.1: Datafly algorithms table example [5]

II μ **Argus**

The second practical implementation of k-anonymity was done by the Statistics Netherlands [5]. The objective of the project was to develop

Birthdate	Gender	Zip	No Of Occurs	Tuple No
1971	Male	4041	2	t1,t2
1971	FeMale	4026	2	t3,t4
1970	Male	4026	2	t5,t6
1970	Male	4026	1	t7
1971	FeMale	4029	1	t8
1970	Male	4029	2	t9,t10
1975	Male	4026	2	t10, t11
3	2	3		

TABLE 3.2: Generalized output of original dataset based on quasi-identifiers. [5]

Birthdate	Gender	Zip	Disease
1971	Male	4041	fever
1971	Male	4041	HIV
1971	Female	4026	bronchitis
1971	Female	4026	obesity
1970	Female	4026	HIV
1970	Female	4026	vomiting
1970	Male	4029	Allergic
1970	Male	4029	Back pain
1975	Male	4026	Cancer
1975	Male	4026	Wheezing

TABLE 3.3: Datafly algorithm k =2, with quasi-identifier Birthdate, Gender, Zip, final result. [5]

software that protects the disclosure of individuals' identity from being recognized in the released data. μ-Argus makes use of the generalization and suppression method like the Datafly algorithm, but it operates in different way. μ-Argus makes use of the k value, assignes values to each of the attributes in the original table. Values are between 0 and 3, which corresponds to not identifying, most identifying, more identifying and identifying. Then, μ-Argus makes a combination by testing 2- and 3 combinations of attributes. Those combinations which are not safe are eliminated by generalizing attributes within combinations and by cell suppression. Despite the erase of entire tuples like in Datafly, here values are suppressed at the cell level. Thus, the output of this algorithm contains the contents of the original data but values could be missing in some cell locations. Fig 3.3 shows the algorithm of μ-Argus.

IIII **Optimal K-anonymity**

Optimal K-anonymity is one of the practical methods of K-anonymity algorithms for determining an optimal-anonymization of a given dataset. An optimal anonymization is one which depersonalize the input dataset

Input: Private Table **PT**; quasi-identifier QI = (A_1, …, A_n),
disjoint subsets of QI known as *Identifying*, *More*, and
Most where QI = *Identifying* ∪ *More* ∪ *Most*, k constraint;
domain generalization hierarchies DGH_{Ai}, where i=1,…,n.
Output: MT containing a generalization of PT[QI]
Assumes: |PT| ≥ k
Method:

1. freq ← a frequency list containing distinct sequences of values of PT[QI], along with the number of occurrences of each sequence.
2. Generalize each A_i∈ QI in freq until its assigned values satisfy k.
3. Test 2- and 3- combinations of *Identifying*, *More* and *Most* and **let** *outliers* store those cell combinations not having k occurrences.
4. Data holder decides whether to generalize an A_j∈ QI based on *outliers* and if so, identifies the A_j to generalize. freq contains the generalized result.
5. **Repeat** steps 3 and 4 until the data holder no longer elects to generalize.
6. Automatically suppress a value having a combination in *outliers*, where precedence is given to the value occurring in the most number of combinations of *outliers*.

FIGURE 3.1: μ-Argus Algorithm [2]

as little as possible to achieve k-anonymity, where "as little as possible" is typically quantified by a given cost metric. There exist several different cost metrics proposed[[2, 17, 20, 27]]. Their aim is to minimize the amount of information loss resulting from the generalization and suppression operations that are applied to produce the transformed dataset.

The approach used in this algorithm was a bit different than Datafly and μ-Argus. The differeces can be seen as:

1 Previous proposals start generalizing from the original dataset and systematically or greedily generalize it into one that is K-anonymous, whereas the K-optimal algorithm starts with a fully generalized dataset (every tuple is identical to every other tuple) and systematically specializes the dataset into one that is minimally K-anonymous.

2 K-optimal algorithm uses a tree-search strategy for cost-based pruning and dynamic search rearrangement.

3 They proposed data management strategies to reduce the cost of evaluating a given anonymization.

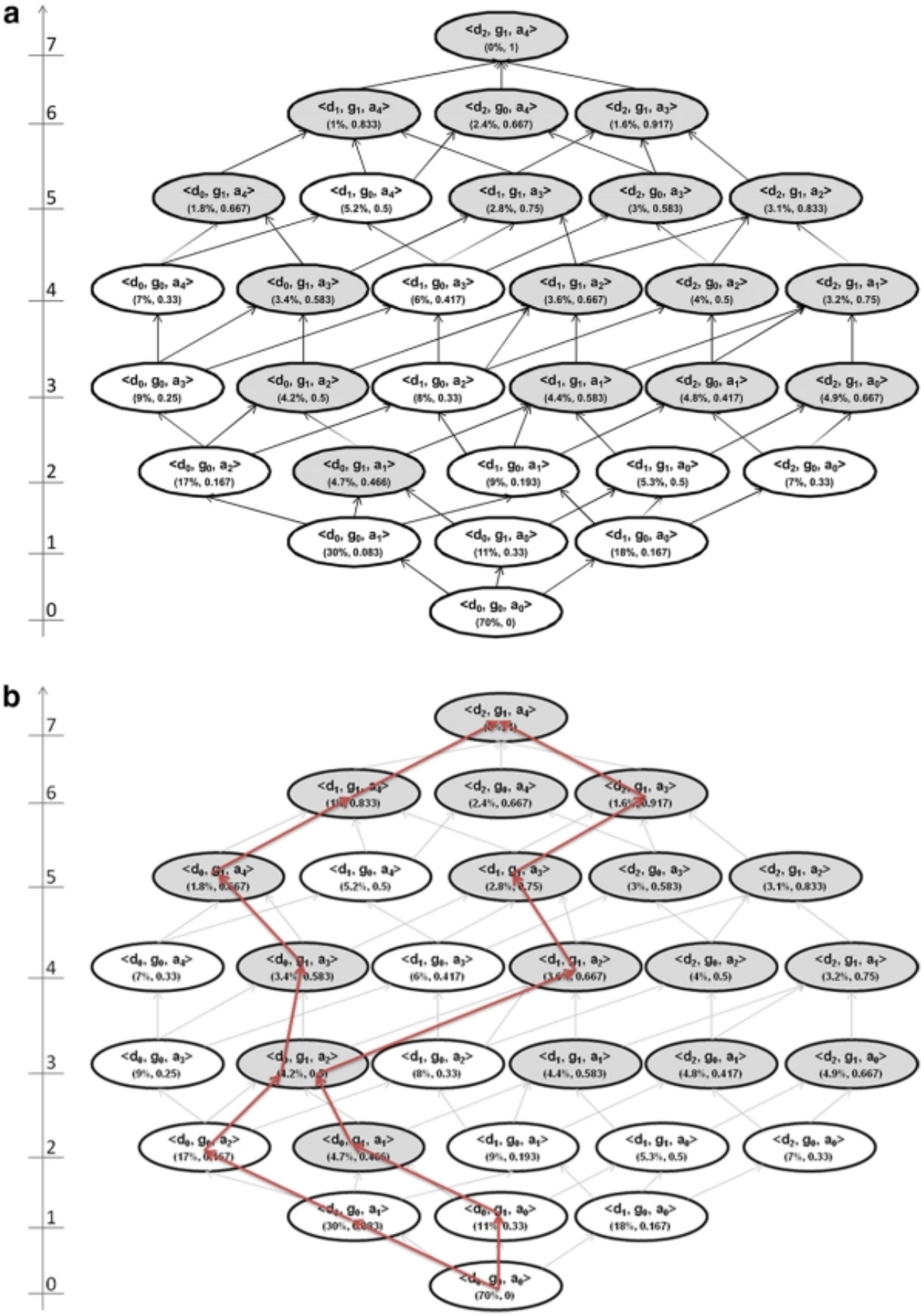

FIGURE 3.2: Optimal K-Anonymity [3]

Fig 3.2 (a) shows the tree structure of the optimal k-anonymity algorithm. There are 3 attributes ((d) for admission date, (g) for gender and (a) age in years). Each node indicates the generalization level for each of the three attributes. To find the optimal k-anonymity, it starts with the most suppresed value,$<d_0,g_0,a_0>$having 70% of the records suppresed and 0 precision of information loss. At the higher levels the nodes are more generalized, the suppression goes lower and loss of information is lowered. Fig 3.2 (b) shows how the optimal K-anonymity algorithm finds the node with the optimal solution. The optimal value is found by comparing the value of percentage of suppression and the value of the precision information loss metric.

Optimal k-anonymity searches through the whole power set to find the lowest cost. Searching the set of all subsets of a given dataset is a problem, and it's an NP-hard problem. To challenge this issue, optimal k-anonymity uses the search method such as the OPUS framework [28]. OPUS framework is built on top of the set-enumeration search strategy[28] with a dynamic tree arrangement and cost-based pruning for solving the optimization problem.

4

Software Requirement Specification

As with any software development process, one of the most important steps in carrying out a successful software engineering methodology is to identify the basic requirement of the system. Systems which require security and privacy, need to follow some standard methodologies to accomplish their goal. There are a number of software engineering methodologies. Each methodology represents a different approach to evaluation. The fact that there are so many approaches in common use simply reflects the fact that no single methodology is 'the best'. Since this paper of the Safer@Home project is focused on security and privacy issues, a methodology called PriS is found to be the most relevant of all the others.

PriS [29] is a security requirements engineering method, which includes privacy requirements early in the development stage. PriS considers the different privacy requirement goals that need to be achieved and describes the effect of privacy requirements and facilitates the identification of the system architecture. Privacy goals are generated to satisfy the issues of 1) authentication, 2) authorization, 3) identification, 4) data protection, 5) anonymity, 6) pseudonymity, 7) unlinkability.

4.1 Basic Requirements

- **User Profile:** The system has different types of users with different privacy levels. Thus, the user profile will be given according to their privacy levels. For example doctor profile should have a maximum privacy level with authorization to access the information without perturbation, whereas a researcher profile should have less access. The system should be accessible using a predifined username and a password.
- **User Authorization:** Users are only allowed to query information based on their privacy level. Users should be allowed to see or surf anonymized datasets.
- **Anonymization:** Data should be anonymized before being published. Failure of good anonymization could lead to leakage of information or a linking attack could be performed by adversaries.
- **Real Time Sensing:** The software should accept input data in real time and anonymize the dataset with the existing datasets to find the minimal anonymized datasets and respond to end users in real time.
- **Privacy:** Private data should be anonymized such that adversaries should not be able to link or re-identify data.
- **Data Utility:** The data utility should be maintained upon applying the anonymization process. Users with limited privacy levels will have more information loss compared with those users that have full authorization. But the utility of data should be high by using effective anonymization algorithms and accurate information loss measures.

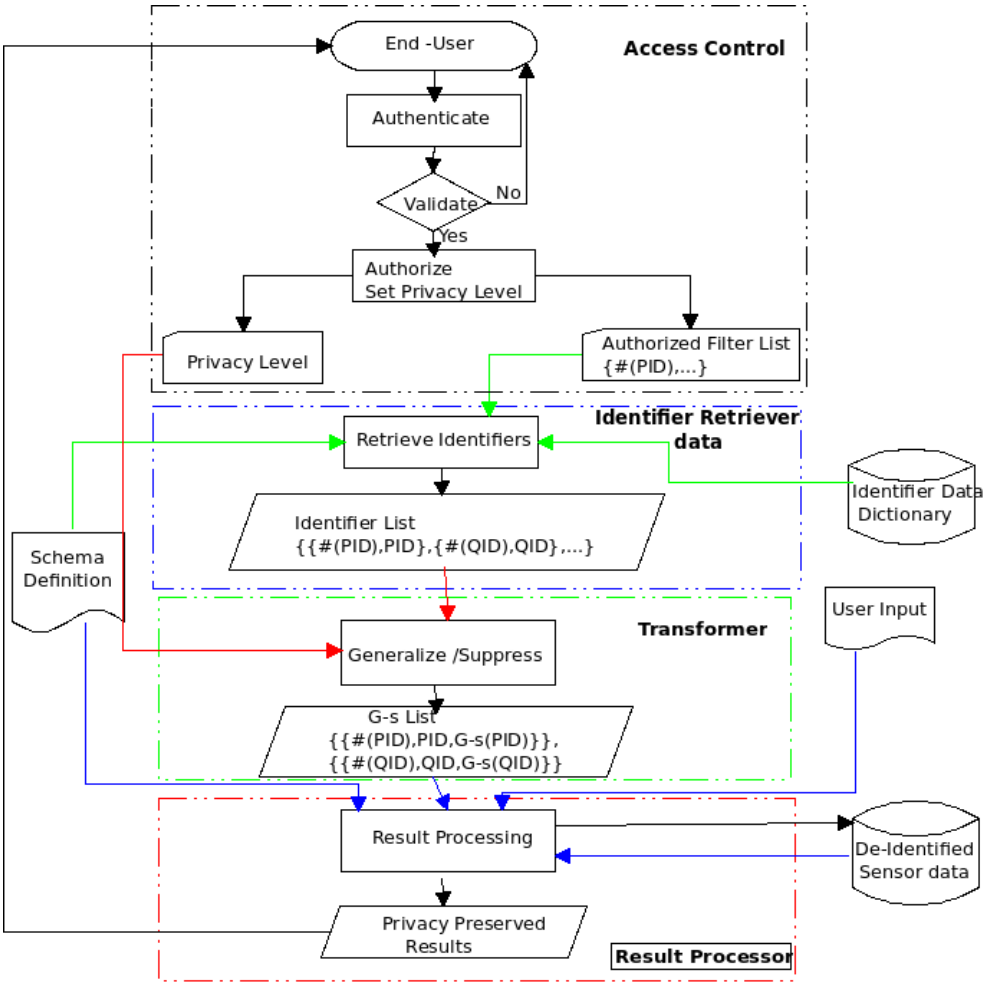

FIGURE 4.1: Software anonymization flow diagram [4]

Fig 4.1 depicts the flow diagram of anonymization process. The process starts by validating end users with already registered users of the system, and forwards those users that are authenticated. Each of these users is assigned a privacy level according to their type of security level. To determine which of the attributes in the original dataset are identifiers, non-identifiers or sensitive attributes, a separate schema definition is in position to make it easily configurable to the non fixed requirement of the anonymization requirement. This can be configured by the administrator based on the anonymization process needed. Thus, using the privacy level and the schema definition we can anonymize the original dataset to meet the requirements of the k-anonymity algortihm. Finally, the output of this process will be different with respect to the privacy level and schema definition.

4.2 System Requirement

4.2.1 Functional and non-Functional Requirement

ID	System Feature	Functional Requirements Details
1	Authentication	**Description:** The system shall provide an interface to let the users type their unique credentials. **Stimulus/Response Sequence:** User to enter user name and password. • Each of the users shall be assigned privacy levels. Users with privacy level value of 1 are those with high security level that can see the original data without perturbation. Those users with limited privileges have higher privacy level numbers.
2	Real time input dataset	The system shall accept input in real time from Hadoop/HBase.
3	Real time output	**Description:** The system shall provide appropriate feedback to the user in real time. **Stimulus/Response Sequence:** User to enter command to perform an immediate task. • The system shall provide feedback for a user request for an instant from the command being issued by the user.
4	Anonymization	**Description:** The system shall anonymize datasets. **Stimulus/Response Sequence:** Anonymizing datasets based on the privacy level of end user. • The system shall anonymize input dataset into a minimal anonymized dataset.
5	Data utility	**Description:** How much data is changed after anonymizing original data. **Stimulus/Response Sequence:** Minimizing the information loss. • Data which has been anonymized should keep the integrity and consistency of the original data. • It should be useful for researchers and analysts.

6	System Management	**Description:** The system shall provide an interface to let users type their unique credentials. **Stimulus/Response Sequence:** User to enter username and password. • The system shall allow only administrators to delete existing users. • For a user already logged in, the system shall request the users password again after the account has been idle for x number of minutes (configurable within user preferences). • The system shall allow only administrator users to create or modify users.

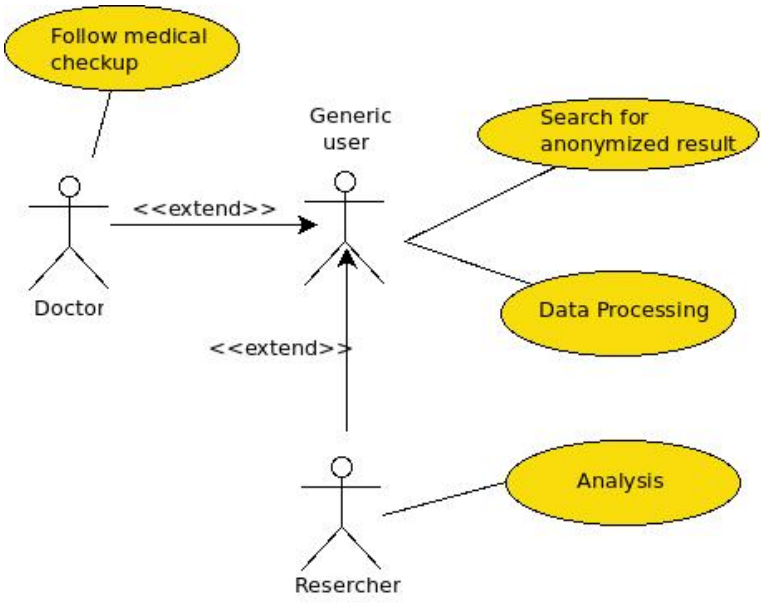

FIGURE 4.2: Use case model

Fig 4.2 shows the sample of use case model for researchers and doctors. These two users have a common purpose in searching stored data and processing them for their specific needs. For each of the two different actors we could have different privacy levels that are classified under them. For example, a nurse is grouped under the doctor category where it has privileges lower than the doctor. Thus, the result for the doctor group could be different for each of those actors under that category. The same case applies to the researcher case. There could

be in- house researchers (researchers of data owner organization) and out-of-house researchers (researchers from other organizations). In-house researchers have privacy levels more than those of out house researchers.

Fig 4.3 depicts the use case scenario of a doctor or researcher.

- Users log-on on to the system, by entering their credentials.
- Smart Home Application checks if the user is authenticated and the result is sent to the client browser with information, authorization, privacy levels.
- Based on his/her privacy level, the user requests to query some information from the database. The results query is the anonymized output of the original dataset which is de-personalized based on the privacy level of the user who is requesting the service.

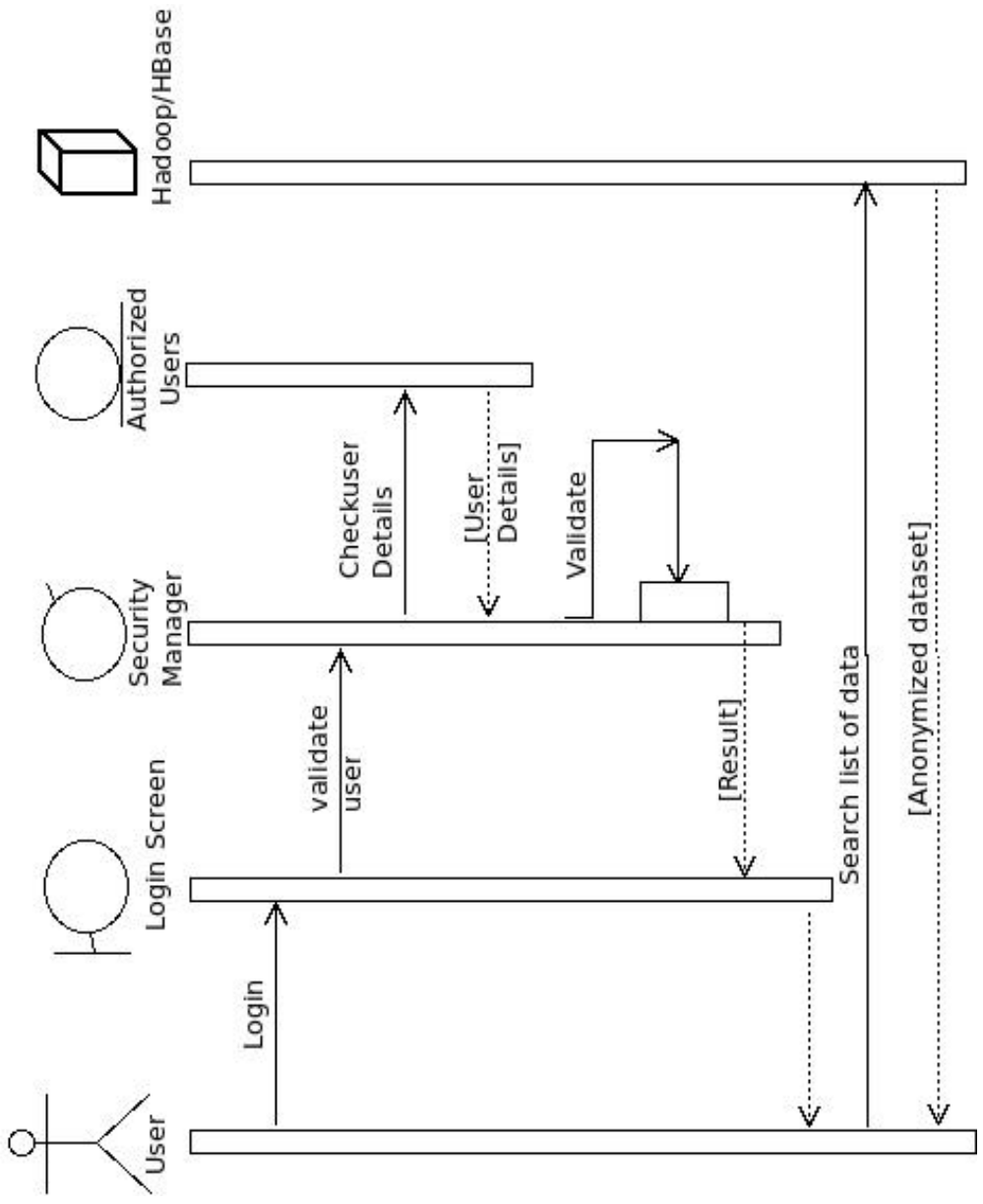

FIGURE 4.3: Usec case scenario

5

Framework for Evaluating Privacy Preserving

The aim of having a framework for evaluating privacy preserving data mining techniques is to preserve the privacy of users while extracting information from a dataset. Research has indicated that there is no single privacy preserving algorithm which performs all the evaluation criteria of privacy preserving techniques. Different algorithms require different metrics to quantify their privacy preserving techniques.

According to [18] the main goals of privacy preserving in data mining (PPDM) algorithm are:

- After applying PPDM algorithms sensible information must be hidden.
- Have to be resistant to the various data mining techniques.
- It should not compromise the access and the use of non sensitive data.
- It should not have an exponential computational complexity.

Privacy preserving data mining techniques can be evaluated by:

1 **Privacy level**: Indicates how much information can still be inferred from the anonymized information.

2 **Hiding failure** : Indicates information that are not hidden even after applying the anonymization algorithm.

3 **Data quality**: How does data change after the application of PPDM algorithm.

4 **Complexity**: Ability of privacy preserving algorithm to execute with good performance in terms of all the resources implied by the algorithm.

For each of these evaluation criteria there exist metrics to quantify their degree of information loss. The following are metrics of evaluation criteria:

5.0.2 Metrics for Quantifying Privacy level

The measurement of quantifying data privacy is based on the degree of uncertainty, i.e how much information can be inferred based on the original dataset. The higher the amount of uncertainty achieved by the privacy preserving data mining algorithm, the better the data privacy is protected by PPDM algorithm. According to [18] PPDM algorithms can be classified into two main categories: Heuristic-based approaches and cryptography-based approaches. Heuristic based approaches are: additive noise, multiplicative noise, K-anonymization and statistical disclosure.

The objective of anonymization is to protect against the disclosure of personal data and keep its utility before being made open to the public. The anonymized data has to be useful for further analysis or research by third parties. K-Anonymization technique is a heuristic technique we have adopted to make the anonymization of data sets. It was introduced by Samarati and Sweeney in [5] [27]. A dataset is anonymous with respect to quasi-identifier attributes if there exists at least k-1 same tuples in the dataset having exact values of quasi-identifier attributes. So for every dataset T, k-anonymization is performed to produce a new dataset $T*$ that guarantees the k-anonymity property for sensitive attributes by generalization and suppression on the set of quasi-identifiers. The probability of finding the individual form set of k will be $1/k$. which is the degree of uncertainty.

5.1 Metrics for quantifying Hiding Failure

Even though privacy preserving techniques have applied to a dataset, there can be sometimes failure to not anonymize the whole dataset. Those dataset which can still be discovered shows the hidden failures. K-anonymity is designed to achieve zero hiding failure. Thus, all patterns potentially sensitive are hidden. But the more information is hidden the more non-sensitive information will be missed. So algorithms have to balance between the need to make dataset private and knowledge to discover from anonymized datasets. Oliveira and Zaiane [30] calculate hiding failure as the ratio of the sanitized dataset with original dataset:

$$HF = \frac{\#R_p(D')}{\#R(_pD)}$$

where $\#RP(D)$ is the number of restrictive patterns of the original data and $\#RP(D')$ is for the perturbed data.

5.2 Metrics for quantifying data quality

Privacy preserving algorithms usually modify datasets through insertion of fake informations or by generalizing and suppressing the original values. It is obvious to see that the more the data is changed or perturbed the less useful it becomes for data analysis or researchers. So quantifying the data quality after performing the anonymization is the most important part of the metrics. There exists several data quality metrics for quantifying the utility of data after anonymization. However there is no specific metric that can be used widely by the research community. To evaluate the quality of data after the privacy preserving methodology is processed it is important to assess both the quality of the data mining results and the quality of the data resulting from the PPDM process. The resulting data after performing the privacy preserving technique can be the evaluating measurement and the same with the data mining results which evaluates the change in the information that is extracted from the database after the privacy preserving process is applied.

The result of data quality metric usually depends on the context in which the data is used [31]. Data quality not only depends on the privacy preserving data mining algorithm but also on the structure of the database. The following parameters are among the most important data quality measurements.

- **Accuracy:** measures the relation of original data and anonymized data.

- **Completeness:** measures how much the anonymized data is changed from the original data.
- **Consistency:** measures the relationships that must hold among data items in a database.

Accuracy measures the information loss after privacy preserving techniques are applied to the original dataset. The information loss resulting from the operation can be measured by how much the generalized dataset approximates the original dataset. The less information that is lost, the better the data quality is. This depends on the different kinds of privacy preserving data mining algorithms.

To measure the utility of numerical attributes, lets consider a table T with quasi-identifier $(A_1, ..., A_n)$. If a tuple $t = (x_1, ..., x_n)$ is generalized to a tuple $t' = ([y_1-, z_1], ..., [y_n-, z_{n1}])$ such that $y_i \leq x_i \leq z_i (1 \leq i \leq n)$. Then normalized certainity penalty (NCP) of tuple t on attribute A_i is defined by:

$$NCP_{A_i}(t) = \frac{z_i - y_i}{|A_i|} \; where \mid A_i \mid = max_{t \epsilon T} t.A_i - min_{t \epsilon T} t.A_i \text{ [32]}$$

This matematical formulation measures how much tuple t is generalized on attribute A_i in terms of the generalized interval size [32, 33]. Tuples with interval size smaller have the higher accuracy when they are queried. The sum of all the inveral size on all the quasi-identifier attributes of the generalized tuples measures the certainity penalty. The total certainity penalty of the generalized dataset is the sum of certainity penalty of all the tuples.

Utility measures for categorical attributes follows the hierarchial tree structure as shown in our previous example of occupation. These trees specify the attribute values with different granularity. Suppose a tuple t has value v on categorical attribute A_i, v is generalized to a set of values v_1,...,v_m in the hierarchy tree. To measure the generalization quantitatively :

$$NCP_{A_i}(t) = \frac{\mid ancestor(v_1 ... v_m) \mid}{A_i} \text{ [33]}$$

In categorical, NCP measures how much tuple t is generalized on A_i in terms of the number of distinct values the generalized value covers [33].

For both numerical and categorical attributes, the weighted normalized certainity penality is given by :

$$NCP(t) = \sum_{i=1}^{n} (w_i . NCP_{A_i}(t)), where \sum_{i=1}^{n} w_i = 1 \text{ [33]}$$

To find the weighted normalized certainity penalty for the whole table T it is defined by:

$$NCP(T) = \sum_{t \epsilon T} NCP(t)$$

From these two matematical formulas we can conclude that NCP(t) is the weighted sum of the normalized certainity penalty on all attributes. But to find the certainity penalty on the whole table we take the sum of the penalty on all the tuples.

Based on the NCP metric a new metric is discovered which is called Global Certainity Penalty (GCP). This metric measures the information loss in the whole perturbed dataset. Suppose P is the set of all equivalence classes in the published anonymized dataset. GCP is defined by :

$$GCP(P) = \frac{\sum_{G \epsilon P} \mid G \mid .NCP(G)}{d.N} \text{ [34]}$$

Where G denotes the equivalence class, N is the number of total records in the original dataset and d is the dimensionality of quasi-identifier. This measures the information loss among the tables with varing dimensionality and cardinality. The value of GCP is between 0 and 1, where 0 signifies that there is no information loss and 1 corresponds to total information loss or the number of equivalence classes is only one covering all the records of the original dataset.

Classification metric is also another information loss measurement introduced by Iyneger [17] to optimize a k-anonymous dataset for training a classfier. Classification metric measures the information loss by adding the individual penalities for each tuple in the table normalized by the total number of rows in N. It is matematically give by :

$$CM(T*) = \frac{\sum_{all\ rows} penalty(row\ r)}{N}$$

The value of r is 1 if it's penalized or if it is suppresed. Otherwise, the penalty value of row r is 0.

Bayardo and Agrawal [35] proposed another metric called the discernibility metric (DM). The discernibility metric assigns penality to each tuple based on how many tuples in the transformed dataset are indistinguishable from it. In a global recoding scheme, suppressed entries are assigned a cost equal to the size of the dataset. The cost of anonymization is the sum of all the anonymization cost in the tuples:

$$C_{DM} = \sum_{e \epsilon E} \mid x \mid^2 + \mid S \mid\mid D \mid \text{ [7]}$$

Where E is the set of all equivalence classes, S is the set of suppressed entries and D is the original dataset.

For example Let t be a tuple from the original table T, and let $GT*(t)$ be the set of tuples in an anonymized table T indistinguishable from t or the set of tuples in T equivalent to the anonymized value of t. Then discernibility metric (DM) is defined as follows:

$$DM(T*) = \sum |GT*(t)|$$

Tuples which are suppressed tend to lose more information, so it's better to avoid the practice of tuple suppression. For example if the tuple t is suppressed, the size of $GT*(t)$ is the same size as the size of $T*$.

6

Design and Implementation

6.1 The Mondrian algorithm Explanation

Mondrian is a multidimensional k-anonymity algorithm that anonymizes data through recursively partitioning using the quasi-identifier attribute dimensions with a median-partition methodology. This model is very fast, scalable [32] and it produces better results than most other recoding models [16].

The Mondrian algorithm as shown in fig 6.1 uses strict partitioning and relaxed partitioning methods instead of domain generalization or value generalization methods. These methodologies enable Mondrian to have a higher data utility result with lower runtime cost than the older implementations of k-anonymity. A greedy algorithm approach is used to partition the original dataset into an equivalence class less with size than $2k - 1$ by recursively partitioning on the median of the quasi-identifier attribute with the largest normalized range in the equivalence class.

A dataset contains a number of rows where each row is represented by a tuple T. The dataset is made up of several attributes, which are composed of identifiers, quasi-identifiers and non-identifiers. Our main concern here is the attributes of

Anonymize($partition$)
if (no allowable multidimensional cut for $partition$)
return $\phi : partition \rightarrow summary$
else
$dim \leftarrow$ choose_dimension()
$fs \leftarrow$ frequency_set($partition$, dim)
$splitVal \leftarrow$ find_median(fs)
$lhs \leftarrow \{t \in partition : t.dim \leq splitVal\}$
$rhs \leftarrow \{t \in partition : t.dim > splitVal\}$
return Anonymize(rhs) $\cup$ Anonymize(lhs)

FIGURE 6.1: Mondrian Algorithm

quasi-identifiers. If we have d quasi-identifiers, then we need a d dimension representation of Mondrian.

Fig 6.2 depicts the flow diagram of the Mondrian algorithm, where it starts with input of some dataset made up of set of tuples where each tuple represents a patient record. Using the quasi-identifier attributes the algorithm classifies the tuples into equivalence classes. Assume a table T consist of a multi set of tuples. An equivalence class for T with respect to the quasi-identifier attributes $Q_1...Q_d$ is the set of all tuples in T containing identical values $(x_1, ..., x_d)$ for (Q_1 ...Q_d). Since there could be many equivalence classes in a table, they will be represented as list of equivalence classes. Each of these equivalence classes in the list, needs to be partitioned based on the dimension with the largest normalized range attribute. These dimensions could be multi-dimensional or single-dimension. The input indicated by the name "choose dimension" signifies which quasi-identifier to be used in order to perform the partitioning. This means if we have d quasi-identifiers we have to perform partitioning based on all the d quasi-identifiers. If the result of the partitioning is still bigger than the value of the $2k - 1$ value, we keep partitioning the equivalence class until it becomes less than the size of $2k - 1$. There are two types of partitioning algorithms in Mondrian, the relaxed and the strict partitioning. The Strict partitioning uses dimension to partition the equivalence class on the median. Each value of dimension less than the median is partitioned into the left tree and for values greater than the median to the right of the tree. Where as relaxed partitioning uses dimension to partition the equivalence class in the median, but the division is into left, center and right partition. Values which are equal to the median will be allocated in the center node.

Strict Multidimensional Partitioning [7] defines a set of non-overlapping multidimensional regions that cover $DQ_1...DQ_d$. ϕ maps each tuple $(x_1, ..., x_d) \epsilon DQ_1...DQ_d$

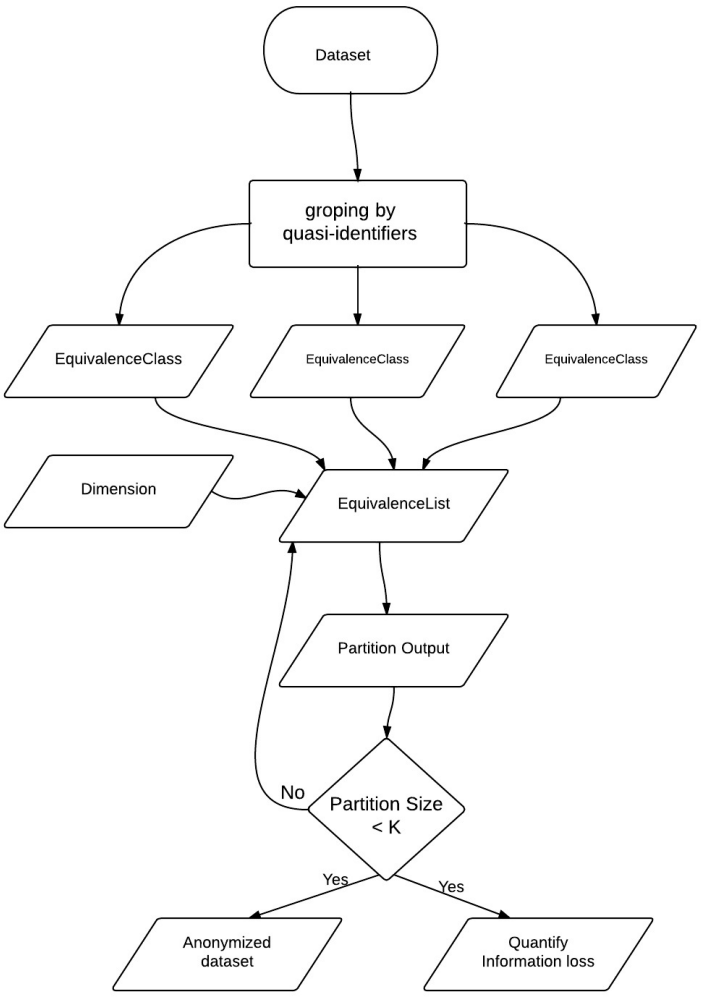

FIGURE 6.2: Mondrian algorithm flow diagram implemenation

to a summary statistic for the region in which it is contained, where ϕ is a function.

Relaxed Multidimensional Partitioning [7] for relation T defines a set of (potentially overlapping) distinct multidimensional regions that cover $DQ_1...DQ_d$. Local recoding function ϕ * maps each tuple $(x_1, ..., x_d)\epsilon\, T$ to a summary statistic for one of the regions in which it is contained. This relaxation offers an extra degree of flexibility. For example, consider generating a 3-anonymization of Patients, and suppose Zip code is the single quasi-identifier attribute ranging from 53710 to 53712 . Using the strict model, we would need to recode the Zip code

value in each tuple to $[53710 - 53712]$. Under the relaxed model, this recoding can be performed on a tuple-by-tuple basis, mapping some occurrences of 53711 to $[53710 - 53711]$ and some other occurrences to $[53711 - 53712]$.

Mondrian algorithm uses the same approach as the KD-Tree approach to plot the d dimension of quasi-identifiers and find the range of the points and generalization. A KD-Tree is a data tree similar to Binary Space Partitioning Tree in funcitionality. A KD-Tree partitions the data to generate an evenly balanced tree, with each leaf partition containing an approximately equivalent number of objects.

The following dataset shows the partitioning process of Mondrian algorithm:

Name	Age	Sex	Zip Code
Bob	23	M	11000
Ken	27	M	13000
Linda	65	F	25000
Alice	65	F	25000
Peter	35	M	59000
Sam	59	M	12000
Jane	61	F	54000
Mandy	70	F	30000
Jane	62	F	54000
Moore	79	F	30000

TABLE 6.1: A sample dataset with ten tuples

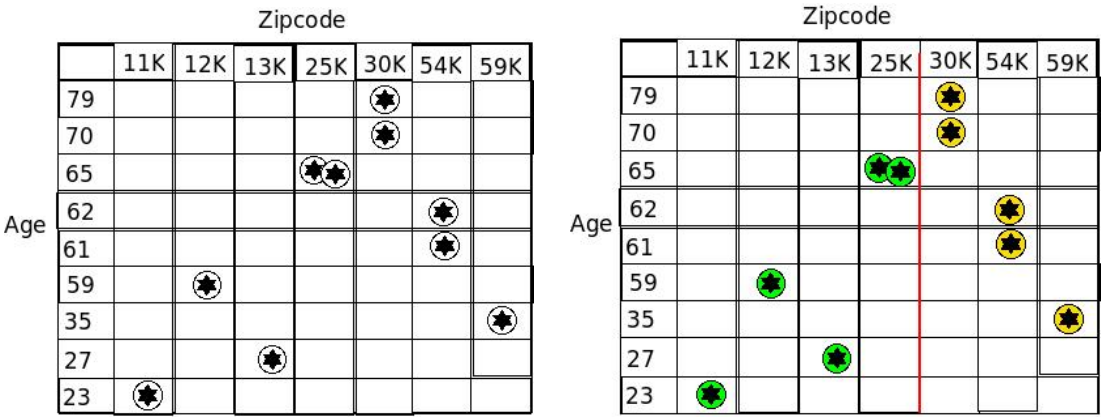

FIGURE 6.3: Steps 1 and 2

The first Figure in 6.3 shows the plotting of the original dataset into the 2-dimensional quasi-identifier dataset (Age and zip code). The process starts by

partitioning in the y-axis because the normalized range is bigger than the x-axis, resulting in two regions. Since the number of points in each of the regions are more than $2k - 1$ partioning need to be performed again upon the two newly formed tables. *(*For this example a point is considered as a tuple).

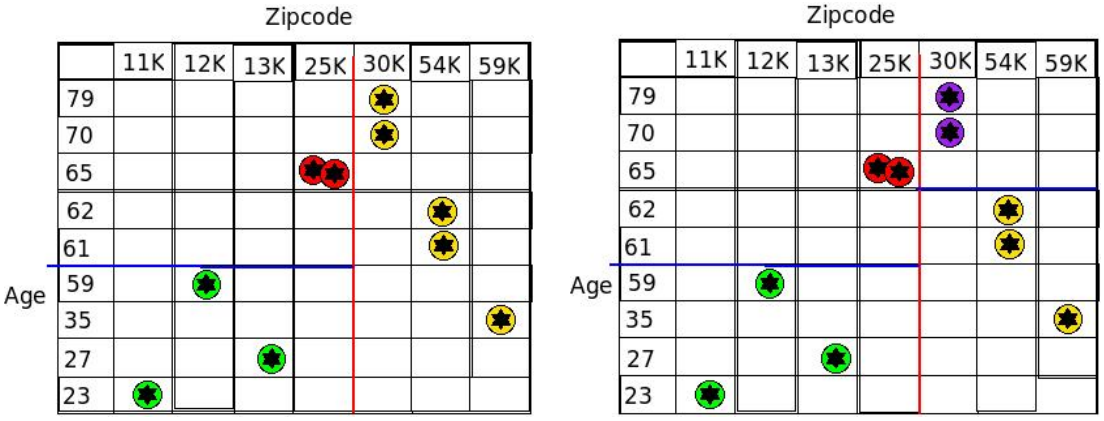

FIGURE 6.4: Steps 3 and 4

6.2 Mondrian Anonymization Models

The process started on Fig 6.3 continues until the number of points is less than $2k-1$, where k is a postive constant number. The second Figure in 6.4 shows the final minimal representation of the Mondrian recursive partitioning algorithm resulting in smaller regions.

All anonymization algorithms use generalization and suppression to transform the original datasets into anonymous datasets. The previous algorithms from the background section show that these anonymization techniques use the domain generalization and value generalization. Whereas the Mondrian algorithm uses two different approaches of generalization and suppression. These two different approaches are Global recoding and Local recoding. The Mondrian algorithm implemented with global recoding tends to maintain the consistency of the dataset, whereas the Mondrian algorithm implemented with local recoding give higher utilities. The Mondrian algorithm uses the top down model approach. This method first takes the whole dataset as an equivalence class or group. The data utility of the table is minimal since all tuples are generalized to the same equivalence class or group. It then recursively partitions each group into two. If each subset contains at least k tuples, the utility is improved because the degree of generlization is minimal. The algorithm stops when further partitioning leads to the violation of the k-anonymity.

6.2.1 Global Recoding

Datasets which are anonymized by global recoding have all the attributes generalized or suppressed equally to all the entries. If different tuples have the same value of an attribute, this value will be mapped to the same generalized value, for example if a tuple has an age attribute with value "50" then it can only map to the same interval [40-50].

Global recoding can be categorized as single-dimensional recoding or multi-dimensional recoding. Single-dimensional global recoding is defined by a function f_i: $D_{X_i} \rightarrow D$ for each attribute X_i of the quasi-identifier. Anonymization value can be obtained by applying the function f_i to the values of the quasi-identifiers X_i in each tuple of the original dataset. Whereas multidimensional global recoding is defined by a single function $f : D_{X_1} * ... * D_{X_n} \rightarrow D$, the anonymized value is obtained by applying the function f to the vector of quasi-identifiers in each tuple of the original dataset.

6.2.2 Local Recoding

Unlike global recoding, local recoding suppresses attributes on a per-cell basis. This will result in a dataset with less data suppressed than the global recoding. The data space is partitioned into different regions and all records in the same region are mapped to the same generalized group. Since the bounds on the solution quality are big enough it's not easy to prove the generalization given the optimal solution. The global recoding has a state space of 2^k, whereas, local recoding has state space of 2^{nk}.

A global recoding achieves anonymity by mapping the domains of the quasi-identifier attributes to generalized or altered values [36]. Whereas, local recoding models map (non-distinct) individual data items to generalized values. The advantage of global recoding is that the anonymized table will have a homogeneous set of values while the disadvantage is that it completely over-generalize the orignal table and results in a large amount of information loss in the process. Whereas, local recoding gives much better results in real practice and it has much more utility than the global recoding methodology.

In Safer@Home though, both privacy and utility is important, Since we are looking for data to be processed by third parties like doctor, nurses and researchers, giving invalid or more generalized values could lead to data which doesn't represent the real picture of our customers data. For this reason it is designed

and implemented the local recoding methodology to develop my Mondrian algorithm.

To illustrate the main difference between these methods lets take a simple example.

Considering Table 6.2, Attributes Age and Zip Code are quasi-identifiers. Table 6.3 and Table 6.4 are 3 anonymous tables generated by the two methods of Mondrian algrorithm global recoding and local recoding respectively. The Age attribute in the global recoding has one big range [54-62], whereas in local recoding it has two groups [54-60] and [60-62]. Global recoding has a full range in the domain and it is the most generalized value, while local recoding has more specific information about Age and Zip Code.

6.3 Class Diagram

Fig 6.5 show the class diagram of anonymization and information loss. The basic building class of this has been adopted from [37]. I have extended those classes to include anonymization models for both the Mondrian and Optimal K-anonymity.

According the specification given in Chapter 5 the class diagram shows that, new anonymization algorithms can be easily pluged in and extend to impelement efficient anonymizaiton algorithms which may come in future.

Age	Zip Code	Disease
54	4041	Flu
55	4042	Flue
60	4043	Diabetic
60	4044	AIDS
62	4045	HIV
62	4046	influenza

TABLE 6.2: Example: A dataset with 10 tuples

Age	Zip Code	Disease
[54-62]	[4041-4043]	Flu
[54-62]	[4041-4043]	Flue
[54-62]	[4041-4043]	Diabetic
[54-62]	[4041-4043]	AIDS
[54-62]	[4041-4043]	HIV
[54-62]	[4041-4043]	influenza

TABLE 6.3: global recoding

Age	Zip Code	Disease
[54-60]	[4041-4043]	Flu
[54-60]	[4041-4043]	Flue
[54-60]	[4041-4043]	Diabetic
[60-62]	[4044-4046]	AIDS
[60-62]	[4044-4046]	HIV
[60-62]	[4044-4046]	Influenza

TABLE 6.4: local recoding

- **AbstractAlgorithm**: This class is the fundamental class which defines the quasi-identifiers, ranges of attributes and calculate information loss.
- **Tuple**:is a class which represents a row data and recursively builds the whole dataset.
- **EquivalenceClass**: Each tuple is grouped based on the quasi-identifiers range into equivalenceClasses.
- **EquivalenceList**: represents the equivalences classes in lists. Each dataset is represented in one equilence list.
- **Mondrian-Anonymization**: This class implements the Mondrian multidimensional algorithm. It uses the local recoding method to anonymize the dataset.
- **Mondrian-InformationLoss**: is a class which quantifies the information loss of anonymized datasets.
- **Optimal-Anonymization**: a class which implements the anonymizaiton of Optimal K-anonymity algorithm.

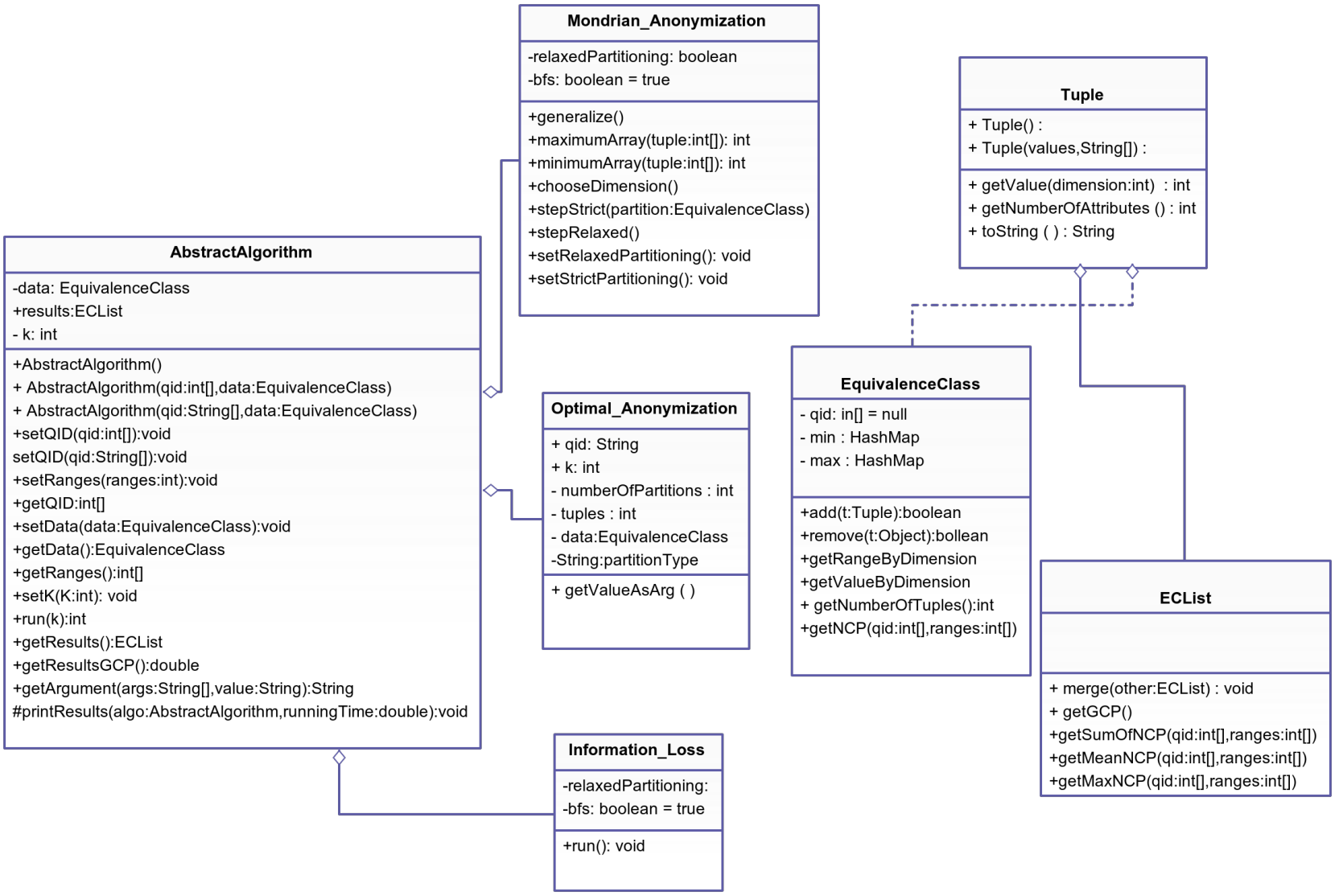

FIGURE 6.5: Mondrian class diagram implementation

6.4 Parallelizing Mondrian k-anonymity algorithm

Mondrian works in a top down manner and the main idea in parallelizing it is to split each dataset in to two equally sized datasets (when I say a dataset I mean a set -or a List- of tuples and not the whole dataset). This means that at first step the input of the algorithm is one huge set containing every tuple and the output is two equally sized partitions, say $t0$ and $t1$. At the second step $t0$ is partitioned again in two other partitions (say $t00$ and $t01$) and $t1$ is also partitioned as well (say to paritions $t10$ and $t11$). This creates a tree-like structure of recursions, until each created parition contains less than $2k - 1$ tuples.

My proposal here is to model each level of this "tree" as a separate map reduce job. This means that at the aforementioned second step, we should create one map reduce job with two tasks dealing with $t0$ and $t1$ simultaneously (which will then create $t00, t01, t10, t11$). This approach is much better than creating one MapReduce job for each input set (like 1 MapReduce job for $t0$ and another one for $t1$) because input partitions are created very fast (exponentially actually) and the performance of our application could be compromised. Fig 6.6 illustrates the implementation of Mondrian algorithm using the MapReduce framework.

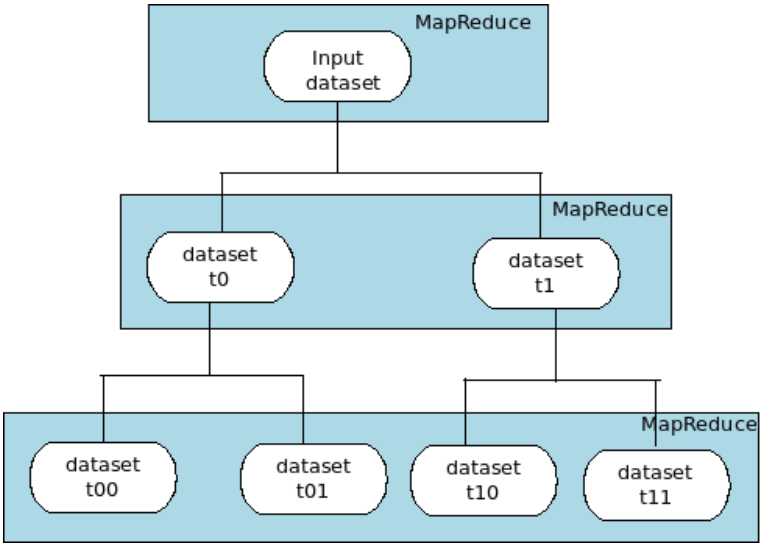

FIGURE 6.6: Parallelizing Mondrian using MapReduce framework

To parallelize the Mondrian k-anonymity algorithm using the MapReduce frame work, lets consider the different steps of the algorithm into different modules.

6.4.1 Sorting

The first module is to sort the datasets based on the quasi-identifier sets.

Map Step: The input dataset is stored on HDFS in the format of $< key, value >$ pairs, each of which represents a record in the data set. The key is combination of the quasi-identifiers and the value is content of the tuple. The data set is split and broadcasted to all mappers. The pseudo code of map function is shown in Algorithm 1. We can set the input parameters like quasi-identifiers, privacy level (K) to the job before the map function invoked.

Input: (key_1: quasi-identifiers; $value_1$: text of a record)
Output: key_2: a string representing a cell, $value_2$: the value in current dimension
Parse the string value ;
Set string outKey and outValue as null;
key ← set of quasi-identifiers;
value ← value in current dimension;
outKey ← sorted key based on quasi-identifiers;
outValue ← data[currentDimension];
output(outKey, outValue);

Algorithm 1: Map pseudo code

Reduce Step: The input of the reduce function is the result obtained from the map function as output. Since the objective of this MapReduce programme is to sort the input data, the Reduce function does only operate as a combiner.

Input: (key_2: a string representing a cell, $value_2$: the value in current dimension)
Output: key_3:text, $value_3$: the value in current dimension
outKey ← sorted key based on quasi-identifiers;
outValue ← data[currentDimension];
output(outKey, outValue);

Algorithm 2: Reduce pseudo code

Algorithm 2 shows that the reduce programe combines the input from the mapper based on their key and gives a sorted dataset as an output.

6.4.2 Recursive MapReduce

As shown in fig 6.6 each dataset is partitioned recursively into two equal sized datasets. This iteration goes until each of the tables size is less than or equal to 2k. The choice of the partition is based on the median of the quasi-identifier with the biggest range. During the Map phase of this recursion, input datasets have a privacy level as input parameter to check if the required level of privacy is met. Inputs are read recursivley and sort them based on their key and hand it to the Reduce phase. In the Reduce module, range of the quasi-identifiers is calculated inorder to find the median partitioner.

Input: (key_1: quasi-identifiers; $value_1$: text of a record; privacy level k)
Output: key_2: a string representing a cell, $value_2$: the value in current dimension
Parse the string value ;
Set string outKey and outValue as null;
key ← set of quasi-identifiers;
value ← value in current dimension;
outKey ← sorted key based on quasi-identifiers;
outValue ← data[currentDimension];
output(outKey, outValue);

Algorithm 3: Recursive map psedo code

Algorithm 3 shows the regular Map algorithm with input from the previous sorted dataset.

In Hadoop each MapReduce programme perform a partitioning process. Partitioning is the process of partitioning the key space to determine which reducer instance will process which intermediate keys and value. All keys which are the same goes to the same reducer.

As show in Fig 6.1 the Mondrian algorithm partitiones the dataset using the median of the attribute with the largest range. The explicit partioner of the Hadoop MapReduce frame works has to be changed to a custom partioner and set the median as a partitioner.

Input: (key_3: quasi-identifiers; $value_3$: text of a record; privacy level k)
Output: key_4: a string representing a cell, $value_4$: the value in current dimension
dimension = chooseDimension() ;
splitVal = findMedian(dimension) ;
ltable = (t ϵ partition : t.dim $\leq$ splitVal) ;
rtable = (t ϵ partition : t.dim $\geq$ splitVal) ;
outKey $\leftarrow$ key of table;
outValue $\leftarrow$ value of table (left/right);
output(outKey,outValue) ;

Algorithm 4: Custom partitioner

The ouput of algorithm 4 is two tables (those with quasi-identifier attribute less than the median value to the left table and the others to right table).

Input: (key_5: quasi-identifiers; $value_5$: text of a record; privacy level k)
Output: (key_6: a string representing a cell, $value_6$: the value in current dimension)
if *dataset size* $\leq 2K-1$ **then**
    Initialize ïňĆoat numbers max=Float.MIN VALUE, min=Float.MAX VALUE and split=0 to record the maximum, minimum ;
    while *(value.hasNext())* **do**
        get value.next named tuple ;
        if *(tuple > max)* **then**
            max =tuple ;
        end
        if *(tuple < min)* **then**
            min =tuple ;
        end
        outKey $\leftarrow$ sorted key based on quasi-identifiers;
        outValue $\leftarrow$ data[currentDimension];
        `replace the selected numerical quasi-identifier by [min-max] value`
    end
    output(outKey, outValue) ;
end
else
    Parse the string value ;
    Set string outKey and outValue as null;
    key $\leftarrow$ set of quasi-identifiers;
    value $\leftarrow$ value in current dimension;
    outKey $\leftarrow$ sorted key based on quasi-identifiers;
    outValue $\leftarrow$ data[currentDimension];
    output(outKey,outValue);
end

Algorithm 5: Recursive Map

In Algorithm 3 we first check the input dataset if it satisfies the k-anonymity requirment of $2k-1$ size. For those dataset with size less than $2k-1$ we start to anonymize the datasets by taking the maximum and minimum value of the attribute and change the explicit value by min-max value. Otherwise, the partitioning of the table countinues.

7

Results and Evaluation

In this section, I will evaluate both the anonymization and information loss models. I have used two datasets to show how we can achieve the required perturbed data and quantify the information loss based on the metrics discussed in Chapter 5. Unlike the existing papers of K-anonymity, this experiment does not compare the different algorithms of K-anonymity. Papers [7, 32, 34] show this comparison, and they have concluded that Mondrian multidimensional recoding method results are better in both utility and information loss. Thus, all results presented in this thesis paper are a result of the Mondrian multidimensional algorithm.

Throughout the experiment, 2-dimensional attributes are taken as a set of quasi-identifiers. All algorithms are implemented in Java and the experiments were run on a Dell Insipiron 15R with $3.8GB$ of memory and Intel Core^{T}M i5 CPU M 460 @ 2.53GHz x 4.

The first dataset is the famous Adult dataset from the UC Irvine machine learning repository [17]. The second dataset is a sample dataset that I have generated for demonstrating how the anonymization process perturbs the original dataset to a $2K - 1$ anonymous dataset. In this thesis report I have included only these ten tuple sample records results for demonstration purposes. Because the size file of the anonymized data of UC Irvine machine learning repository is too big to

be included in this thesis report, I haven't included it here. The anonymization of all UC Irvine machine learning repository can be found on the CD/DVD given with the Master thesis paper.

7.0.3 Anonymization

Input

Name	Age	Sex	Zip Code	Disease
Bob	23	M	11000	Pneumonia
Ken	27	M	13000	Dyspepsia
Linda	65	F	25000	Gastritis
Alice	65	F	25000	Flu
Peter	35	M	59000	Dyspepsia
Sam	59	M	12000	Pneumonia
Jane	61	F	54000	Flu
Mandy	70	F	30000	Bronchitis
Jane	62	F	54000	Flu
Moore	79	F	30000	Bronchitis
Kjetil	30	M	12000	Flu
Stephen	54	F	13000	Bronchitis

TABLE 7.1: Random sample dataset

Table 7.1, shows the original sample dataset with quasi-identifiers (Age, Gender and Zip Code), with $K = 2$, the Mondrian multidimensional algorithm partitions these attributes into the minimum cut to allow $2k - 1$ tuples in each equivalence class. Each equivalence class is represented by the range from minimum and maximum values. Attributes with unique identifiers such as Name (if we consider it to be a unique identifier) will suppressed to a value $*$ in each of the tuples. Table 7.2 shows the local recoding of 2-anonymization for Table 7.1, some tuples overlap over the range of others, for example tuple $t5, t6$ and $t7$ have a range of $[59 - 65]$ and tuples from $t8, t9$ and $t10$ have the range of $[62 - 79]$.

7.0.4 Utility Measure

The Adults dataset from the UC Irvine Machine Learning Repository is mostly used to compare the different kinds of K-anonymity algorithms. The dataset contains anonymous census data of 32562 individuals from the 1990 US census. The following shows the fields represented in the set.

- **Age:** numeric

Name	Age	Sex	Zip Code	Disease
*	[23-27]	M	[11000-25000]	Pneumonia
*	[23-27]	M	[11000-25000]	Dyspepsia
*	[35-61]	F	[30000-59000]	Gastritis
*	[35-61]	F	[30000-59000]	Flu
*	[35-61]	M	[30000-59000]	Dyspepsia
*	[59-65]	M	[11000-25000]	Pneumonia
*	[59-65]	F	[11000-25000]	Flu
*	[59-65]	F	[11000-25000]	Bronchitis
*	[62-79]	F	[30000-59000]	Flu
*	[62-79]	F	[30000-59000]	Bronchitis
*	[62-79]	M	[30000-59000]	Flu
*	[62-79]	F	[30000-59000]	Bronchitis

TABLE 7.2: Result of anonymizing 7.1

- **Employment type:** Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.
- **Final sampling weight:** numeric
- **Education:** Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.
- **Years of education:** continuous
- **Marital status:** Married-civ-spouse, Divorced, Never-married, Separated, Widowed,Married-spouse-absent, Married- AF-spouse
- **Occupation:** Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty,Handlers-cleaners, Machine-op-inspect, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed- Forces.
- **Relationship:** Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried
- **Sex:** Female, Male
- **Capital gain:** continuous
- **Capital loss:** continuous
- **Hours worked per week:** continuous.
- **Native country:** United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI- etc), India, Japan, Greece, South, China,

Cuba, Iran,Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal,Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia,Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador,Trinidad and Tobago, Peru, Hong, Holland- Netherlands

Attribute	Description	Type
Age	Age of respondent	Continuous
Final Sampling weight	sample weight	Continuous

- 39, State-gov, 77516, Bachelors, 13, Never-married, Adm-clerical, Not-in-family, White, Male, 2174, 0, 40, United-States, <=50K
- 50, Self-emp-not-inc, 83311, Bachelors, 13, Married-civ-spouse, Exec-managerial, Husband, White, Male, 0, 0, 13, United-States, <=50K
- 38, Private, 215646, HS-grad, 9, Divorced, Handlers-cleaners, Not-in-family, White, Male, 0, 0, 40, United-States, <=50K
- 53, Private, 234721, 11th, 7, Married-civ-spouse, Handlers-cleaners, Husband, Black, Male, 0, 0, 40, United-States, <=50K

I assume the Age and Final sampling weight are the set of quasi-identifiers with numerical values. The anonymization process for the categorical values can be approached in a similar way. In the categorical types, each attribute values could be represented by hierarchical levels where each level could be an integer value.

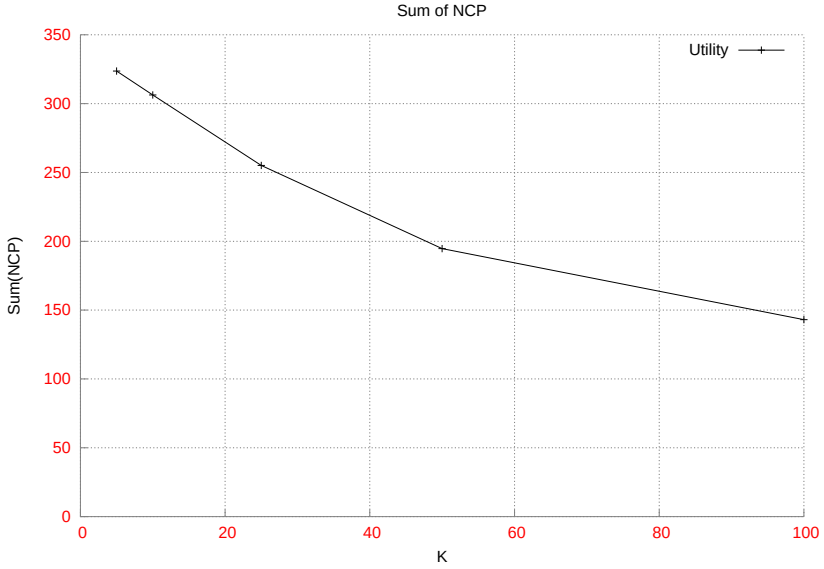

FIGURE 7.1: Normalized certainty penalty with respect to K (privacy Level), on datasets with distribution (N=32000)

The normalized certainty penalty measures the utitlity of anonymized dataset. Fig 7.1 depicts the summation of normalized range of equivalence classes with high privacy (low value of k) have the higher normalized certainity penality than those with low privacy. Eventhough, the normalized range of each equivalence classes in high privacy is small, the number of tuples in each equivalence group are high that their summation is larger than than the normalized range of equivalence classes in lower privacy (high value of K).

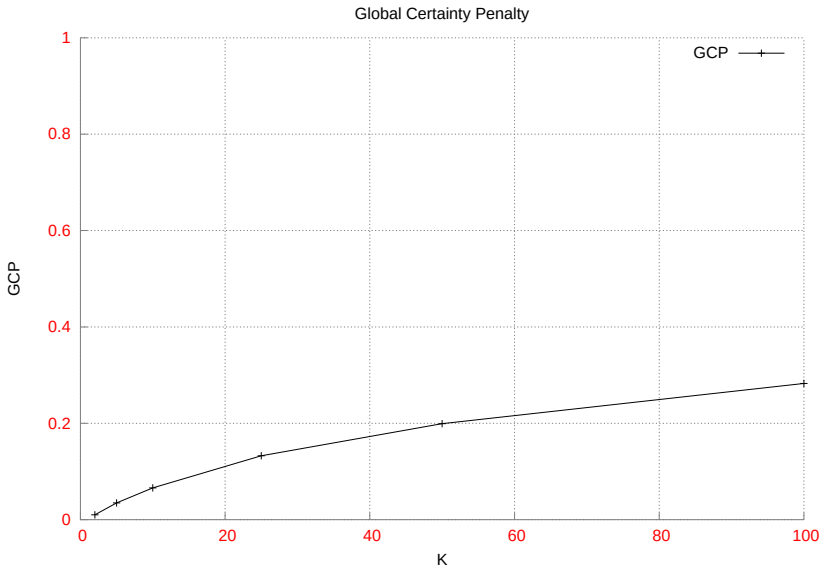

FIGURE 7.2: Global Certainty penalty with respect to K (privacy Level), on datsets with distribution (N = 32000) and 2 quasi-identifier attributes

Global Certainty Penalty (GCP) measures the information loss over the entire anonymized table as a value from 0 to 1. A value of 0 represents zero information loss, and a value of 1 represents total information loss. Figure 7.2 shows the GCP of our anonymized UC Irvine machine learning dataset with different privacy levels (k). The result shows that, as the privacy level increases, so does the GCP. At a privacy level of 25, there is around 15% information loss, and as the privacy level increases to 100, the information loss doubles to 30%. This growth in information loss is due to the fact that there are less equivalence classes as k increases, leading to more generalized attributes. Users such as doctors have a very low value of k, and they get information which is not perturbed with 100% data utility.

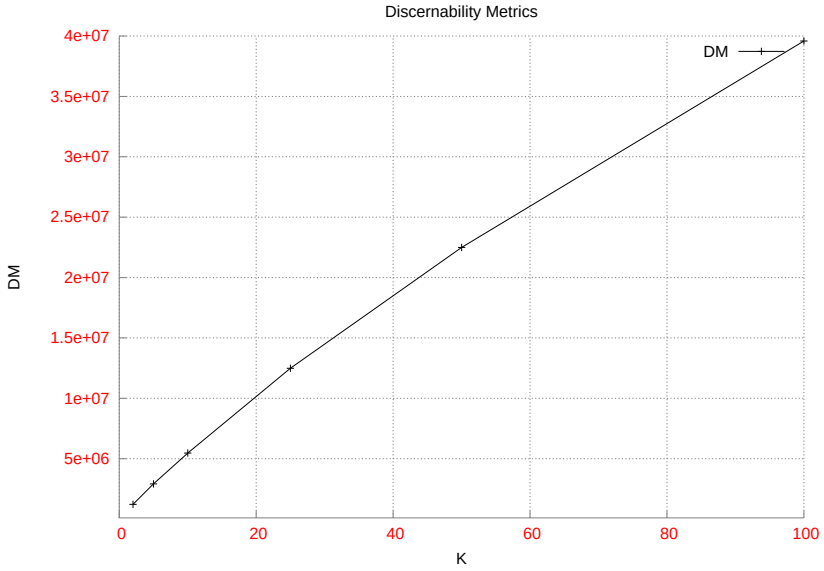

FIGURE 7.3: Discernability penalty with respect to K (privacy Level), on datsets with distribution (N = 32000) and 2 quasi-identifier attributes.

Discernability metrics measures the cardinality of the equivalence class. For a low value of K the cardinalty of equivalence is too small. If the privacy level is high (high value of K) the discernability metric increases sharply which increase the cardinalty of an equivalence class. Equivalence classes with large cardinalty tend to group datasets in large range leading to large information loss. Fig 7.3 presents the discernability penalty of 32000 records.

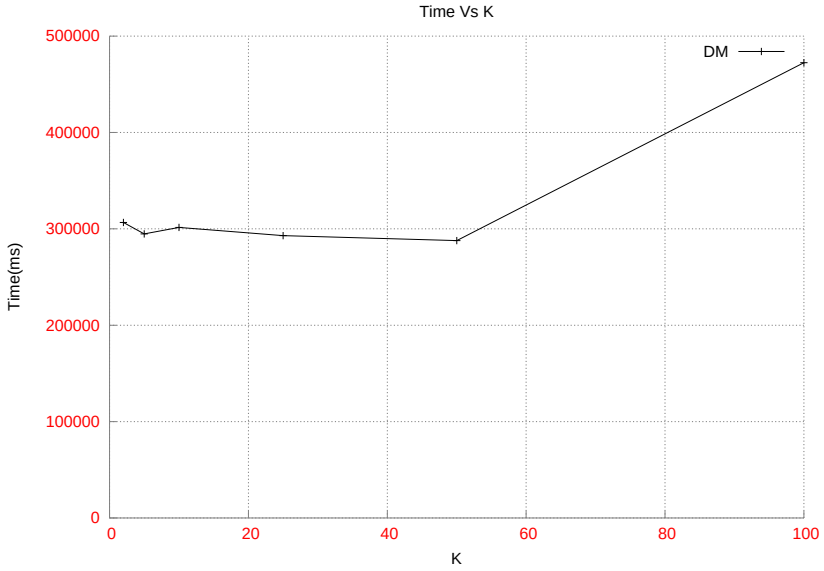

FIGURE 7.4: Running time on datsets with distribution (N = 32000) and 2 quasi-identifier attributes.

Fig 7.4 shows the time it takes to complete the anonymization of 32000 records with 2 quasi-identifiers in the UC Irvine data was almost steadily constant until k=50. For larger value of k, the running time starts to increase sharply. Because the number of equivalence class are small in size and the ranges are getting bigger. It takes more time to process each of these large equivalence classes than those equivalence classes with large intervals.

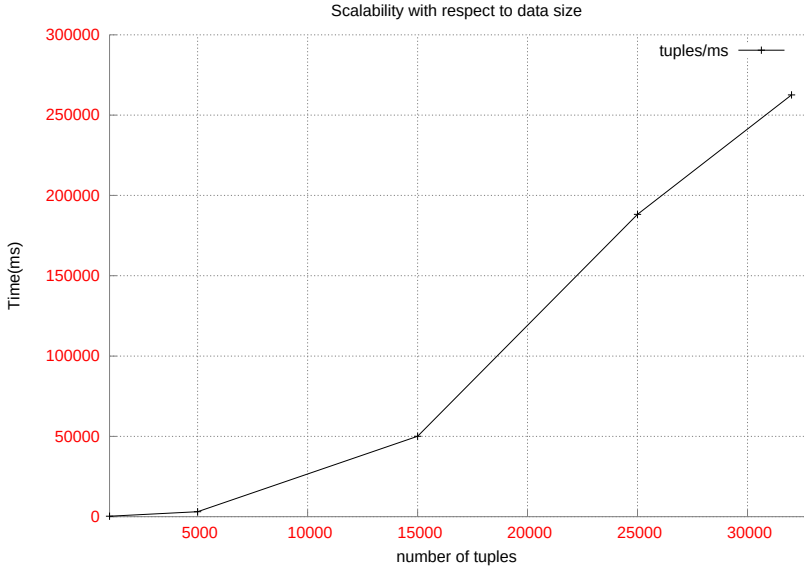

FIGURE 7.5: Scalability with respect to database size,on synthetic data sets with uniform distribution (dimensionality=2, k=10).

Fig 7.5 depicts that the scalability of the system with the data size to be anonymized. This experiment shows a fixed privacy level (K) 10 and with two quasi-identifiers. The result shows that the time it takes to anonymize a dataset of 5000 tuples or less tuples are almost similar, but as the dataset gets bigger the time it takes to complete the anonymization increases steadily. Thus, our anoynimization input dataset will have a strong effect on the performance of anonymization.

8

Conclusions

8.1 Conclusion

Concerns over the privacy of personal information has grown in the last few years. In order to overcome these problems, it is essential that a solution is devised. This solution must use efficient and effective data anonymization techniques. K-anonymization is one of the most extensively used techniques for data anonymization. K-anonymization works by making each tuple in a data set identical to at least k-1 other tuples.

In order to build software related to anonymization of data sets, the software must be thoroughly checked to make sure it can stand up against possibly security related attacks. To do this, the first step is the creation of a requirement specification. The Mondrian algorithm is a multidimensional partitioning algorithm – one kind of K-anonymity algorithm – is the second step, which I have implemented and discussed throughout this thesis paper. Finally, the implementation of Mondrian in a distributed way is presented as an algorithm.

To ensure that the required level of privacy has been achieved by an anonymization implementation and to evaluate anonymized data sets, an evaluation framework has also been discussed. Based on these evaluation metrics, the privacy

and information loss in a data set are quantified. The results proved that as the privacy level of users is low, the anonymization results in more information loss than those with high privacy level. The running time of anonymization also depends on the size of data and cardinality of the datasets.

In conclusion, the Mondrian anonymity technique is capable of anonymizing data to a reasonable level of privacy while still retaining the data utility. For a researcher with a privacy level of k=25, the loss of information is as low as 15% and for those with high privacy such as a doctor the loss of information is as low as 0%. This proves to us that the Mondrian algorithm can provide us privacy and utility.

8.2 Future Work

Although this thesis presented anonymization techniques and a framework for quantifying the information loss, there are several conditions that could be studied in the future.

It is important to study the risk of re-identifing individual if adversaries have some knowledge of the original dataset. Risk evaluation could give an insight into how much information adversaries could dig out. This could help us to prepare against such attacks and help us to choose the safest anonymization parameters such as the set of quasi-identifiers, and the privacy level of users.

The anonymization process is limited by the assumptions I made on the type of quasi-identifier (numeric) and selection of two quasi-identifier attributes. This could be expanded to include the anonymization of categorical attributes and support for more than two quasi-identifiers.

Last but not least, It would be interesting to prepare a graphical user interface for the process of anonymization, utility evaluation, and risk evaluation through the use of bar charts and histograms.

Bibliography

[1] Ricky Ho. How hadoop map/reduce works. URL http://architects.dzone.com/articles/how-hadoop-mapreduce-works.

[2] Anco Hundepool and Leon Willenborg. Argus, software packages for statistical disclosure control. pages 341–345, 1998. doi: 10.1007/978-3-662-01131-7_45. URL http://dx.doi.org/10.1007/978-3-662-01131-7_45.

[3] Issa R Jonker E Amyot D El Emam K, Dankar F. A globally optimal k-anonymity method for the de-identification of health data. In *A globally optimal k-anonymity method for the de-identification of health data.*, WPES '06, pages 370–682. Journal of the American Medical Informatics Association 16: 670âĂŞ682, 2009. URL http://www.ncbi.nlm.nih.gov/pubmed/19567795.

[4] Antorweep. Secuirty and privacy for big data storage and analysis of smart home sensor data, 2013.

[5] L. Sweeney and Latanya Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10:2002, 2002.

[6] Tomasz Wiktor Wlodarczyk. Smart system to support safer independent living and social interaction for elderly at home, 2013.

[7] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *Data Engineering, 2006. ICDE 06. Proceedings of the 22nd International Conference on*, pages 25–25, 2006. doi: 10.1109/ICDE.2006.101.

[8] Wu Li, Yann-Hang Lee, Wei-Tek Tsai, Jingjing Xu, Young-Sung Son, Jun-Hee Park, and Kyung-Duk Moon. Service-oriented smart home applications: composition, code generation, deployment, and execution. *Serv.*

Oriented Comput. Appl., 6(1):65–79, March 2012. ISSN 1863-2386. doi: 10.1007/s11761-011-0086-7. URL http://dx.doi.org/10.1007/s11761-011-0086-7.

[9] R.S. Yamaguchi, K. Hirota, K. Hamada, K. Takahashi, K. Matsuzaki, J. Sakuma, and Y. Shirai. Applicability of existing anonymization methods to large location history data in urban travel. In *Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on*, pages 997–1004, 2012. doi: 10.1109/ICSMC.2012.6377859.

[10] Bin Zhou, Jian Pei, and WoShun Luk. A brief survey on anonymization techniques for privacy preserving publishing of social network data. *SIGKDD Explor. Newsl.*, 10(2):12–22, December 2008. ISSN 1931-0145. doi: 10.1145/1540276.1540279. URL http://doi.acm.org/10.1145/1540276.1540279.

[11] Xuan Ding and Wei Wang. Enabling dynamic analysis of anonymized social network data. In *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2012 International Conference on*, pages 21–26, 2012. doi: 10.1109/CyberC.2012.13.

[12] Charles Safran, Meryl Bloomrosen, W. Edward Hammond, Steven Labkoff, Suzanne Markel-Fox, Paul C. Tang, and Don E. Detmer. Toward a national framework for the secondary use of health data: An american medical informatics association white paper. *Journal of the American Medical Informatics Association*, 14(1):1 – 9, 2007. ISSN 1067-5027. doi: 10.1197/jamia.M2273. URL http://www.sciencedirect.com/science/article/pii/S106750270600212X.

[13] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, pages 111–125, 2008. doi: 10.1109/SP.2008.33.

[14] Nate Anderson. Aol releases search data on 500,000 users (updated). August 2007. URL http://arstechnica.com/uncategorized/2006/08/7433/.

[15] Yohannes Kifle Russom. Secure data colleciton, 2012.

[16] Benjamin C. M. Fung, Ke Wang, Lingyu Wang, and Patrick C. K. Hung. Privacy-preserving data publishing for cluster analysis. *Data and Knowledge Engineering*, 2009.

[17] Vijay S. Iyengar. Transforming data to satisfy privacy constraints. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 279–288, New York, NY, USA, 2002. ACM. ISBN 1-58113-567-X. doi: 10.1145/775047.775089. URL http://doi.acm.org/10.1145/775047.775089.

[18] Elisa Bertino, Dan Lin, and Wei Jiang. A survey of quantification of privacy preserving data mining algorithms.

[19] Xiaokui Xiao, Guozhang Wang, and Johannes Gehrke. Interactive anonymization of sensitive data. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, SIGMOD '09, pages 1051–1054, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-551-2. doi: 10.1145/1559845.1559979. URL http://doi.acm.org/10.1145/1559845.1559979.

[20] LATANYA SWEENEY. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002. doi: 10.1142/S0218488502001648. URL http://www.worldscientific.com/doi/abs/10.1142/S0218488502001648.

[21] Philippe Golle. Revisiting the uniqueness of simple demographics in the us population. In *Proceedings of the 5th ACM workshop on Privacy in electronic society*, WPES '06, pages 77–80, New York, NY, USA, 2006. ACM. ISBN 1-59593-556-8. doi: 10.1145/1179601.1179615. URL http://doi.acm.org/10.1145/1179601.1179615.

[22] Traian Marius Truta and Bindu Vinay. Privacy protection: p-sensitive k-anonymity property. In *In Proc. of 22nd IEEE IntâĂŹl Conf. on Data Engineering Workshops*, page 94. IEEE Computer Society, 2006.

[23] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1), March 2007. ISSN 1556-4681. doi: 10.1145/1217299.1217302. URL http://doi.acm.org/10.1145/1217299.1217302.

[24] Tom White. *Hadoop: The Definitive Guide*. O'Reilly Media, third edition edition, 5 2012. ISBN 9781449311520. URL http://amazon.com/o/ASIN/1449311520/.

[25] Dhruba Borthakur. Hdfs architecture guide, 2013. URL http://hadoop.apache.org/docs/stable/hdfs_design.html.

[26] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, January 2008. ISSN 0001-0782. doi: 10.1145/1327452.1327492. URL http://doi.acm.org/10.1145/1327452.1327492.

[27] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Trans. on Knowl. and Data Eng.*, 13(6):1010–1027, November 2001. ISSN 1041-4347. doi: 10.1109/69.971193. URL http://dx.doi.org/10.1109/69.971193.

[28] Geoffrey I. Webb. Opus: An efficient admissible algorithm for unordered search, 1995.

[29] Christos Kalloniatis, Evangelia Kavakli, and Stefanos Gritzalis. Addressing privacy requirements in system design: the pris method. *Requir. Eng.*, 13(3):241–255, August 2008. ISSN 0947-3602. doi: 10.1007/s00766-008-0067-3. URL http://dx.doi.org/10.1007/s00766-008-0067-3.

[30] Stanley R.M. Oliveira and Osmar R. ZaÃŕane. Privacy preserving frequent itemset mining, 2002.

[31] Giri Kumar Tayi and Donald P. Ballou. Examining data quality. *Commun. ACM*, 41(2):54–57, February 1998. ISSN 0001-0782. doi: 10.1145/269012.269021. URL http://doi.acm.org/10.1145/269012.269021.

[32] Jian Xu, Wei Wang, Jian Pei, Xiaoyuan Wang, Baile Shi, and Ada Wai-Chee Fu. Utility-based anonymization using local recoding. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '06, pages 785–790, New York, NY, USA, 2006. ACM. ISBN 1-59593-339-5. doi: 10.1145/1150402.1150504. URL http://doi.acm.org/10.1145/1150402.1150504.

[33] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining: Models and algorithms, 2000.

[34] Gabriel Ghinita, Panagiotis Karras, Panos Kalnis, and Nikos Mamoulis. Fast data anonymization with low information loss. In *Proceedings of the 33rd international conference on Very large data bases*, VLDB '07, pages 758–769. VLDB Endowment, 2007. ISBN 978-1-59593-649-3. URL http://dl.acm.org/citation.cfm?id=1325851.1325938.

[35] R.J. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *Data Engineering, 2005. ICDE 2005. Proceedings. 21st*

International Conference on, pages 217–228, 2005. doi: 10.1109/ICDE.2005.42.

[36] Leon CRJ Willenborg and Ton De Waal. *Elements of statistical disclosure control*, volume 155. Springer Verlag, 2001.

[37] giagiannis. *K-anonymity*, volume 155. Springer Verlag, Nov 23, 2011.