Dissertation

Quasi-Identifier Discovery to Prevent Privacy Violating Inferences in Large High Dimensional Datasets

Erkennung von Quasi-Identifikatoren zum Schutz der Privatsphäre vor Rückschlüssen in hochdimensionalen Datensätzen

Nikolai Jannik Podlesny

Hasso Plattner Institute for Digital Engineering
Internet Technologies and Systems
Prof.-Dr.-Helmert-Straße 2-3
14482 Potsdam, Germany

Supervisors:

Prof. Dr. Christoph Meinel
Dr. Anne Kayem

Hasso Plattner Institute, Potsdam, Germany

Defense: January 17, 2023
Publication: March 15, 2023

This dissertation is submitted in partial fulfilment of the requirements for the degree of „Doctor rerum naturalium” (Dr. rer. nat.)

Abstract

Personal data privacy is considered to be a fundamental right. It forms a part of our highest ethical standards and is anchored in legislation and various best practices from the technical perspective. Yet, protecting against personal data exposure is a challenging problem from the perspective of generating privacy-preserving datasets to support machine learning and data mining operations. The issue is further compounded by the fact that devices such as consumer wearables and sensors track user behaviours on such a fine-grained level, thereby accelerating the formation of multi-attribute and large-scale high-dimensional datasets.

In recent years, increasing news coverage regarding de-anonymisation incidents, including but not limited to the telecommunication, transportation, financial transaction, and healthcare sectors, have resulted in the exposure of sensitive private information. These incidents indicate that releasing privacy-preserving datasets requires serious consideration from the pre-processing perspective. A critical problem that appears in this regard is the time complexity issue in applying syntactic anonymisation methods, such as k-anonymity, l-diversity, or t-closeness to generating privacy-preserving data. Previous studies have shown that this problem is NP-hard.

This thesis focuses on large high-dimensional datasets as an example of a special case of data that is characteristically challenging to anonymise using syntactic methods. In essence, large high-dimensional data contains a proportionately large number of attributes in proportion to the population of attribute values. Applying standard syntactic data anonymisation approaches to generating privacy-preserving data based on such methods results in high information-loss, thereby rendering the data useless for analytics operations or in low privacy due to inferences based on the data when information loss is minimised.

We postulate that this problem can be resolved effectively by searching for and eliminating all the quasi-identifiers present in a high-dimensional dataset. Essentially, we quantify the privacy-preserving data sharing problem as the Find-QID problem. Further, we show that despite the complex nature of absolute privacy, the discovery of QID can be achieved reliably for large datasets. The risk of private data exposure through inferences can be circumvented, and both can be practicably achieved without the need for high-performance computers.

For this purpose, we present, implement, and empirically assess both mathematical and engineering optimisation methods for a deterministic discovery of privacy-violating inferences. This includes a greedy search scheme by efficiently queuing QID candidates based on their tuple characteristics, projecting QIDs on Bayesian inferences, and countering Bayesian network's state-space-explosion with an aggregation strategy taken from multigrid context and vectorised GPU acceleration. Part of this work showcases magnitudes of processing acceleration, particularly in high dimensions. We even achieve near real-time runtime for currently impractical applications. At the same time, we demonstrate how such contributions could be abused to de-anonymise *Kristine A.* and *Cameron R.* in a public Twitter dataset addressing the US Presidential Election 2020.

Finally, this work contributes, implements, and evaluates an extended and generalised version of the novel syntactic anonymisation methodology, attribute compartmentation. Attribute compartmentation promises sanitised datasets without remaining quasi-identifiers while minimising information loss. To prove its functionality in the real world, we partner with digital health experts to conduct a medical use case study. As part of the experiments, we illustrate that attribute compartmentation is suitable for everyday use and, as a positive side effect, even circumvents a common domain issue of base rate neglect.

Acknowledgement

I wish to acknowledge all those who supported me during my research activities and the preparation of this thesis. Special thanks go to Prof. Dr. Christoph Meinel, who provided me with academic encouragement. Further, I would like to thank Dr. Anne Kayem for her continuous support, supervision, her openness, accessibility for any questions and joint work over the past years.

Also, thank belongs to Dr. Florian Niedermann, Dr. Henning Soller, Dr. Asin Tavakoli, Dr. Ulrike Deetjen, Stephen Simpson, Dr. Oliver Bauer, Dr. Lars Schor, Jan Rösner, as well as all members of the Internet-Technologies & Systems research group for their fruitful discussions. Moreover, I appreciate Dr. Sven Jungmann and Dr. Daniel Wigbers as inspirational and amicable contacts. By providing technical, knowledge and data resources, the Hasso Plattner Foundation, the SAP SE, and McKinsey & Company Inc enabled me to validate my concepts with real-world cases on heavy hardware.

Last, my innermost thanks go to Joscha Podlesny, Monika Podlesny and Nina Müller for their encouragement and moral support throughout my academic journey.

Contents

List of Figures

List of Tables

Glossary

AFIB	Atrial fibrillation.
CCPA	US California consumer privacy act.
CGI	Common gateway interface.
CPU	Central processing unit.
CSC	Compressed sparse column.
CSR	Compressed sparse row.
CUDA	Compute unified device architecture.
DBMS	Database management system.
DNS	Domain name system.
DOB	Date of birth.
ECG	Electrocardiogram.
ETL	Extract, transform, load.
FHE	Fully homomorphic encryption.
FLOPS	Floating-point operations per second.
FPT	Fixed parameter tractable.
GB	Gigabyte.
GDPR	General data protection regulation.

GPGPU	General-purpose computation on GPU.
GPU	Graphics processing unit.
GUI	Graphical user interface.
ID	Identifier.
IoT	Internet of things.
MB	Megabyte.
mpmUCC	Maximal partial minimal unique column combination.
mpUCC	Maximal partial unique column combination.
mUCC	Minimal unique column combination.
PB	Petabyte.
PHE	Partially homomorphic encryption.
PHI	Protected health information.
PII	Personally identifiable information.
QID	Quasi-identifier.
RSA	Rivest–Shamir–Adleman public-key cryptosystem.
SQL	Structured query language.
SSN	Social security number.
SWHE	Somewhat homomorphic encryption.
TB	Terabyte.
UCC	Unique column combination.

1 Introduction

The protection of personally identifiable information (PII) is held in high esteem and is incorporated in our highest professional standards. Legal frameworks around the world have been implemented to ensure the privacy of private data, most prominently the EU General Data Protection Regulation [32] or the US California Consumer Privacy Act [75]. These regulations give consumer data privacy a more central and important role in their data collection, processing, and analysis [109]. However, the industry continues to struggle with finding the right balance for compliance with the legislature and consumer satisfaction [151].

Typically, data privacy can be achieved by either asking the user for their consent or by eliminating any personally identifiable information. Doing the latter is known as anonymisation, a technique that is not entirely new as its' roots originate in the previous century [33]. However, syntactic data anonymisation has been acknowledged to be NP-hard and, therefore, complex to solve. At the same time, we observe an increasing amount of digital information. More data records become available through enhancements in technology like cheap and tiny sensors, cumulating a soaring amount of observation points and hence describing attributes. These dimensions can reach, for instance, in the healthcare genetics sector, thousands of attributes breaking down human DNA into its organic molecule components. These data characteristics with a massive amount of rows and columns are also referred to as high-dimensional datasets.

To counter the determined anonymisation complexity and handle the growing data size, there has been a shift to probabilistic, use-case sensitive, semantic data anonymisation methods by the research community [7, 17, 45, 52, 54]. Here, statistic approaches are designed to reduce the risk of private data exposure based on query-specific constraints. In the recent years, however, we have also observed an increasing news footprint of de-anonymisation incidents in various industries and sectors like telecommunication [40], transportation [148, 152], financial transactions [41, 178] and healthcare [10, 39]. Examples include the re-identification of US Governor William Weld's medical information [10], the exposure of tens of thousands of private health records from a large clinical laboratory network that contained patient names, dates of birth, social security numbers, lab results, and diagnostics data [39] and the de-anonymisation of personal search history resulting in a class-action lawsuit against AOL [8].

1.1 Motivation

The possibility of studying large-scale and high-dimensional datasets in terms of how to generate privacy-preserving data offers the tremendous hope of identifying, understanding and deriving objective insights in a way that has not been possible before. Such inferences can unlock new industry sectors [36], offer novel angles for venues in various research fields like education [76] or bring an enormous promise for advancing clinical care for the greater public good as a new era of medicine [77, 159].

Simultaneously, protecting individual privacy must be respected and is crucial but extremely challenging. Applying syntactic data anonymisation seems impractical or consumes uneconomical resources given its NP-hard nature. Suggested semantic approaches due to their probabilistic character may not suit strong anonymity guarantees, and numerous privacy experts question practicability in large-scale environments [77, 109].

Radical new perspectives are desired to balance economic and social reasoning with the risk of private data exposure. To illustrate these, this work relies on digital health examples like fitness, health, and genomics but finally is not bound to a specific use case after all.

1.2 Problem Statement

Releasing privacy-preserving datasets has been shown to be fraught with problems as de-anonymisation incidents are growing [10, 40, 41, 148, 152, 178]. One rationale for this is that inferences in data are underestimated by their risk of private data exposure and often serve as quasi-identifiers for record re-identification attacks [29, 58, 92, 99, 105] – namely *inference-* and *semantic* attacks. Related work and existing syntactic anonymisation mechanisms strive for minimising information-loss. Inference attacks exploit low information loss by drawing a conclusion from observed patterns in the data using auxiliary data to derive private information [184, 185]. Semantic attacks extend the same methodology by exploiting the meaning of information [99, 105, 186].

Modern sensors used, for instance, in consumer wearables, track a variety of observation points. These data records compose highly sparsely populated, multi-attribute, high-dimensional datasets in combination and on a fine-grained level. The search for and elimination of quasi-identifiers (QID) in these datasets is essential for the privacy-preserving data sharing problem. This occurs since a large diversity of information fosters data patterns for inferences. Consequently, a single inference can already lead to the compromise of privacy.

A series of algorithms or methods is needed to realise efficient quasi-identifier discovery for high-dimensional datasets both to enable effective data transformation for privacy, and to verify whether or not a given dataset is privacy preserving.

1.3 Contribution

This work proposes multiple novel methodologies of discovering and eliminating privacy-violating inferences, namely quasi-identifiers in high-dimensional datasets. We will discuss their implementation details, offer optimisation options and demonstrate their efficiency. These new approaches are use case agnostic, highly effective, and proved in a real-world use case. Further, our contributions can be summarised as follows:

- We start by demonstrating the privacy problem is even worse than already perceived by linking the search for quasi-identifiers to the "Unique" problem. Through this hierarchy, we derive that reliable discovery of all quasi-identifiers is a fixed-parameter traceable problem (FPT) and W[2]-complete. FPT allows efficient algorithm for small values of the fixed parameter, but not in large, high dimensions. Determining all quasi-identifiers in a dataset leads to superpolynomial runtime.
- Next, we present new approaches of deterministic discovering all quasi-identifiers that can serve as inferences for de-anonymisation on a large-scale.
- To counter exponential growth, we discuss optimisations for greedy processing, mathematical multigrid solver, and massive-parallelisation through vectorised GPU-acceleration. These options will be algorithmically outlined, their implementation details discussed and compared empirically. With magnitudes of processing acceleration, we will showcase that currently impractical applications can be achieved.

- Furthermore, we demonstrate how such runtime optimisations can be abused to attack an already published dataset, by exemplarily re-identifying two individuals, *Kristine A.* and *Cameron R.*, in a Twitter dataset drawn from the 2020 US Presidential Election.
- Finally, this work extends, generalises, implements and evaluates the deterministic anonymisation methodology, *attribute compartmentation* [135], which promises sanitised datasets without remaining quasi-identifiers while minimising information-loss. To prove its functionality in real-life settings, we partner with digital health experts to conduct a medical use case study showcasing that attribute compartmentation is suitable for everyday use and, as a side effect, even circumvents base rate neglect.

1.4 Thesis Structure

The rest of the work is structured as follows. We summarise, discuss and demarcate related work in Chapter 2. Chapter 3 formalises the complexity of detecting quasi-identifiers as "Find-QID" problem and outlines the similarities to a known data profiling problem. An exact discovery scheme for the "Find-QID" problem is presented in Chapter 4. The same Chapter offers different optimisation techniques, including a greedy approach to discover the attribute combinations serving as quasi-identifiers and a massive-parallelised method through vectorised compute. Chapter 5 addresses the venue of utilising Bayesian inferences to triangulate quasi-identifier (QID) in the variety of QID candidates and multigrid solver as well optimisation methods and massive parallelisation by dint of GPU acceleration. Chapter 6 covers the improved elimination of QIDs, especially a novel approach to *attribute compartmentation.* While each of the previous Chapters has their enclosed assessment section, an overarching empirical comparison is offered in Chapter 7. Finally, a summary, as well as avenues for future work, are provided in Chapter 8.

2

Related Work

Data anonymisation is not an entirely new field and has been studied extensively throughout the last decades. Existing privacy models can be generally classified in two categories of operations: *syntactic*- and *semantic* data anonymisation. In the following, we will detail out each of their limitations on transforming highly sparsely populated, multi-attribute, high-dimensional data to generate privacy-preserving datasets.

2.1 Syntactic Data Anonymisation

The family of techniques associated with syntactic data anonymisation relies on a variety of data transformation methods such as randomisation [82, 104, 147], generalisation [67, 166], suppression [67, 166], and perturbation [104, 147]. While suppression simplified erases data, generalisation typically restructures the content of a dataset by modifying its values according to a pre-defined term replacement taxonomy. In hierarchy-based taxonomies, each value progressively loses uniqueness as one moves upwards in the hierarchy.

The k-anonymity family. One of the first and best known is k-anonymity, where each tuple in the data set is classified with at least $k-1$ similar data records to limit distinguishability. The $k-1$ nearest neighbours are, according to Sweeney, selected based on similar describing attributes and enforced by generalisation and suppression [166]. Generalisation follows the pattern of aggregating data values through a pre-defined hierarchy, for instance, the individual year 2021 into a year range of 2020-2025. Suppression, on the contrary, removes the designated data value completely. The toolkit of generalisation is vulnerable to homogeneity and background knowledge attacks [106]. To counter this, l-diversity additionally considers the granularity of sensitive data representations to ensure a diversity of a factor of l for each quasi-identifier within a given equivalence class (usually a size of k). t-closeness as an extension handles skewness and background knowledge attacks by considering the relative distributions of sensitive values both in individual equivalence classes and in the entire dataset [98].

k-anonymity also serves as a privacy metric known as k-map. A dataset satisfies the k-map constraint if every combination of attribute values for quasi-identifiers appears at least k times [166]. Further, k-map applies the assumption that an attacker has no background knowledge on entities in the given dataset. δ-presence, introduced by Nergiz et al. [122], builds on both k-anonymity and k-map to protect against symmetric attacks that exploits the equivalence classes in a dataset for re-identification. Hidden in the δ-parameter δ-min and δ-max exist. These two parameters cover the information that someone is not present.

Data transformation techniques. The previous anonymisation algorithms and their extensions [11, 112] use generalisation and suppression to support transforming the data [67]. This works well for theoretical showcases but quickly reaches its limits for larger datasets. As Meyerson et al. [112] and Bayardo et al. [11, 72] have demonstrated, syntactic data anonymisation algorithms like k-anonymity [166], l-diversity [106], and t-closeness [98] are NP-hard.

The reason is that the dependent *generalisation* algorithms in themselves are NP-hard due to their iterative and incremental nature. Aggrawal et al. [4] have shown that applying generalisation and suppression to high dimensional data results in high information loss, thereby rendering the data useless for data analytics. This effect is particularly true, as for many describing attributes generalisation's runtime grows exponentially and hence is impractical. Consequently, suppression remains and radically removes attribute values leading to large information-loss. Bearing this complexity in mind, all variants of this algorithm can only use heuristics like k-optimize [11] with compute constrains to achieve better approximations known as suboptimal privacy, not perfect privacy [10].

As a suitable alternative to generalisation, perturbation has been presented [104, 147]. Perturbation corresponds to the actual value's alternation to the closest similar findable value. This includes the effect of introducing an aggregated value or using a close-by value that is adopted in the way that just one value needs modification instead of multiple ones to build clusters. Under these circumstances, finding such a value can take longer due to iteratively rechecking the newly created value(s), which consequently impacts performance negatively.

Optimal k-anonymity has been shown to be an NP-hard problem [10, 112]. Applying generalisation and suppression schemes to high-dimensional data results in a high-information loss due to their algorithmic nature, thereby rendering the data almost useless for data analytics. Wong et al. [187] proposed a method to distribute all tuples non-uniformly the same quasi-identifier (QID) in a partition for achieving k-anonymity (*non-homogeneous generalisation*), which promises to decrease information-loss. Meanwhile, Tassa et al. [168] suggest reducing the information loss that is caused by generalising the database entries with k-concealment. Both contributions, however, deteriorate the NP-hardness in the field of high-dimensional application fields. Additionally, heuristic approximation methods like k-optimize have been introduced [11], offering an accelerated computation runtime with the effect of reduced accuracy, and therefore limited privacy guarantees. An in-depth review, categorisation and guidelines for selecting generalisation algorithm have been published by Fredj et al. [67].

For syntactic data anonymisation, optimal and holistic approaches guaranteeing k-anonymity have been proven to be an NP-hard problem [112]. Consequently, scaling, particularly generalisation and perturbation in high-dimensional, results in an impractical runtime [137, 140] and high-information loss, thereby rendering the data useless for data analytics. All variants of k-anonymity algorithms can only use heuristics to achieve better approximations to perfect privacy, but not perfect privacy [10].

2.2 Semantic Data Anonymisation & Differential Privacy

In a bid to re-define the notion of privacy not merely as a process of syntactically transforming datasets but also to consider both the statistical distributions of data values and the semantic meanings drawn from linking (defining patterns) between data points are summarised as semantic data anonymisation approaches. By removing strong links between the data and an individual, the data veracity is altered. This is typically achieved either by noise injection, permutation, or some statistical shifting [49, 82]. Such algorithms are also known under the umbrella of *differential privacy*, and their statistical methods are highly optimised for pre-defined use cases and bulk data processing [13, 51, 73, 82, 103]. For instance, in differential privacy, this is done by determining at the runtime of a query how many noise injections to add to the resulting dataset to ensure anonymity in each case [49]. Differential privacy also uses the exponential mechanism to release statistical information about a dataset without revealing private details of individual data entries [110]. Furthermore, the Laplace mechanism for perturbation supports statistical shifting in differential privacy by employing controlled random distribution sensitive noise additions [54, 91]. It is worth noting here that the discretised version [73, 103] is known as a matrix mechanism because both sensitive attributes and quasi-identifiers are evaluated on a per-row basis during anonymisation [97]. Since these anonymisation are runtime and use case-specific, the anonymisation processing is postponed to query runtime, thereby increases the risk of data leakage. Examples of such data leaks include the vulnerabilities created by colluding users as depicted by Kifer et al. [86]. Leoni introduces "non-interactive" differential privacy by applying the statistical transformations a priori to user queries [96]. A further issue that differential privacy has come up against is that applying differential privacy to large datasets is computationally infeasible (impractical). That differential privacy is also NP-hard [55]. Experts currently still discuss whether approximate differential privacy algorithms satisfy strong privacy guarantees. Under certain circumstances, a single data record could be linked back to its owner through an arbitrary family of attribute sets [61].

These insights lead to an unresolved problem. The anonymisation of a large dataset as either approximate approaches potentially leaves data inferences that can be used to de-anonymise individuals or their exact counterparts result in exponentially increasing runtime due to their complexity.

In addition to the previous summary of very recent contributions, Dwork et al. [50, 53] and Xiong et al. [189] offer an in-depth survey of the previous work. Also, Dankbar et al. presented a thorough review of current literature on differential privacy. They further highlighted significant general limitations, including the theoretical nature of the privacy parameter constraining the ability to quantify the level of anonymisation that would be guaranteed to patients [37, 38]. Ji et al. offered insights into the interplay between machine learning and differential privacy [83]. Li et al. focus on empirical accuracy performances of algorithms and semantic meanings of differential privacy to explain both its strong guarantees and limitations [100].

For large datasets, semantic data anonymisation approaches, including differential privacy, have been shown to be NP-hard [55]. Applying them to extensive high-dimensional data makes them computationally infeasible (impractical performance-wise) given their runtime and use case-specific nature.

2.3 Homomorphic Encryption

The field of cryptography addressed privacy concerns as well. Homomorphic encryption allows actors to perform basic algebraic calculations on encrypted data without decrypting it first [150]. Fontaine et al. offered an overview of state-of-the-art homomorphic encryption schemes with an in-depth discussion of their parameters, performances and security issues. [65]. Fully homomorphic encryption (FHE) that improves the efficiency of secure multi-party computation has been presented by Gentry et al. [71]. Acar et al. picked up demonstrated performance challenges on fully homomorphic encryption (FHE) for practical usage and offered further details on important FHE pillars in their survey, including well-known Partially Homomorphic Encryption (PHE) and Somewhat Homomorphic Encryption (SWHE) [3]. Damgård et al. proposed a general multi-party computation protocol that is secure against an active adversary corruption and offers a complexity linear to the number of parties [35]. Parmar et al. conducted a case study on various principles and properties of homomorphic algorithms [131] using asymmetric key systems (RSA, ElGamal, Paillier, and more). Tebaa et al. explored the setup of hybrid cluster setups and securely sharing data between on-premise servers and cloud computing providers without the need of decryption intending to preserve privacy [170]. Naehrig et al. exhibited several real-world applications in the medical, financial and advertising field and evaluated their optimised homomorphic encryption implementation [116]. Lehmann recently presented a system, *ScrambleDB*, that utilises a homomorphic encryption scheme for establishing a pseudonymisation-as-a-service through multi-party computation [95]. Other examples of homomorphic encryption scheme include, but are not limited to the sector of medical diagnosis by Carpov et al. [24], or medical processing by Kocabas et al. [87], anomaly detection by Alabdulatif et al. [5], IoT by Song et al. [161], smart grids by Bos et al. [19], and energy market by Garcia et al. [70]. Also, homomorphic encryption has been explored by Barni et al. as possibility for securing private data in biometric systems [9]. Similar thoughts have been deepened by Salem et al. to protect biometrical fingerprinting mechanisms as we have in modern smartphones against private data exposure [156]. Erkin et al. presented a method to securely generate recommendations in recommender systems with multiple values packed in one encryption to protect private data against service providers while preserving the functionality of the recommender engine [60]. Wang et al. published a technical patent on how homomorphic encryption can offer privacy-preserving data aggregation [179]. Yet, Kocabas et al. correctly mentioned the initial challenges of acquiring data access for any homomorphic scheme and the implied performance- and storage-related problems like in the use case of long-term patient ECG-data monitoring [88].

A fully homomorphic scheme offers an interesting perspective of securely sharing and conducting algebraic operations on data without private data exposure risk. Yet, the issue of data acquisition and especially performance problems in applying such cryptographic method to extensive high-dimensional data in very short periods are unresolved. Also, a fully homomorphic scheme currently only supports essential operation on numeric data or does not guarantee the absence of a quasi-identifier upon data publishing.

2.4 Unique Column Combinations

In data profiling, unique column combinations (UCC) are attribute combinations that form a unique identifier for the given dataset (table). Discovering these unique column combinations (UCC) is a fundamental research problem.

Abedjan et al. [2] summarised and formalised in their work the latest advances in the discovery of UCCs. Building on their contribution, Heise et al. [80] presented a scalable discovery of unique column combinations based on parallelisation and the scale-out principle. The same has been done by Feldmann [62]. Han et al. [78] extend on similar thoughts and utilise Hadoop with its MapReduce technique [43] for a distributed computing setup. A comparison of different discovery schemes has been presented by Papenbrock et al. [129]. At the same time, Papenbrock et al. offered a hybrid combination of fast approximation techniques and efficient validation techniques for UCCs [129]. Ruiz et al. recently contributed a patent and summarised various datasets profiling tools, methods, and systems, including efficient UCC discovery [153]. In our work, we will later formalise that the search for quasi-identifiers (QID) is a special case for discovering unique column combinations.

According to Bläsius et al. [16], the search for UCC can be enclosed in a circular dependency to the Hitting-Set problem as a family of W[2]-complete problems [16, 47, 48]. In the worst case, this implies at least a super polynomial runtime, making its applicability to large high-dimensional data currently impractical.

2.5 High-Dimensional Data

Given previous developments in syntactic and semantic data anonymisation, further work has reverted to hybrid approaches that combine elements from the initial syntactic data anonymisation and the semantic data anonymisation approaches and offer abstractions from the raw dataset through aggregations or separations. In attribute compartmentation [137, 140], for instance, privacy is guaranteed by drawing on the concept of maximum partial unique column combinations (mpUCC) from the data profiling domain to guarantee privacy by separating attributes that form quasi-identifiers (mpUCCs). Attribute value combinations that uniquely identify individuals in the dataset are also known as quasi-identifiers (QID). Eliminating those QIDs also prevents the re-identification attack of combining QIDs with auxiliary data to draw conclusions and derive private information [184, 185]. Real-world instances of such private data exposure can be found in cases of re-identification of US Governor William Weld's medical information [10], the exposure of tens of thousands of private health records from a large clinical laboratory network included patient names, dates of birth, social security numbers, lab results, and diagnostics data [39] and the de-anonymisation of private search history resulting in a class-action lawsuit against AOL [8]. But discovering quasi-identifiers is not easy.

High-dimensional data are characterised by their large number of rows and columns. While the increasing number of rows is often not a substantial issue, the large number of columns can quickly lead to state-space explosions for enumeration problems [15, 160]. As derived from all previous subsections, the different disciplinary methods like data profiling and mining, anonymisation processing and differential privacy sooner or later run into NP-hard problems to achieving privacy. The larger the dataset dimensions are, the quicker a computational infeasibility is reached.

A few instances are addressing high-dimensional data for anonymisation in detail. Adaptations based on a Secure Multi-party Computing (SMC) protocol have been proposed by Kohlmayer et al. as a flexible approach on top of k-anonymity, l-diversity and t-closeness as well as heuristic optimisation to anonymise distributed and separated data silos in the medical field [89]. Furthermore, to address scalability challenges of large-scale high-dimensional distributed anonymisation that emerge in the healthcare industry, Mohammed et al. [113] propose *LKC-privacy* to achieve privacy in both centralised and distributed scenarios promising scalability for anonymising large datasets. *LKC-privacy* works on the premise that acquiring background knowledge is nontrivial and therefore limits the length of quasi-identifier tuples to a pre-defined size. While one can argue about this approach's practically, the main concern is that *LKC-privacy* does not ensure the complete absence of privacy-violating identifiers due to QID candidate tuple size limitations. Other works use a MapReduce technique based on the Hadoop distributed file system (HDFS) to boost computation capacity as introduced by Zhang et al. [194]. Yet, the NP-hard nature quickly outperforms the economic scalability possibilities. Handling large numbers of entities describing attributes (hundreds of attributes) in a performance efficient and privacy-preserving manner remains to be addressed.

Similar to Manolis Terrovitis [171] work, there are two reasons why distinguishing between sensitive and non-sensitive attributes is dangerous. First, by observation of de-anonymisation attacks (homogeneity, similarity, background knowledge) we note that sensitive attributes alone are not the only basis for their success. Second, defining an exhaustive set of sensitive and non-sensitive attributes is impractical for high dimensional datasets where user behaviours exhibit unique patterns that increase with the volumes of data collected on the individual.

To alleviate the complexity of the compartmentation problem [137, 140], Podlesny et al. proposed modelling the attribute linkage problem for creating privacy-preserving data silos as a Bayesian network [139, 141]. Training a Bayesian network is NP-hard as Chickering et al. [28] had demonstrated. The same NP-hardness applies to the problems of exact inference learning [12, 121] and approximate inference learning [34]. Yet, recent contributions demonstrate that compressing the graph based on attribute linkage heuristics allows a performance-scalable data processing even on large datasets [141].

Clifton et al. offered a weighted discussion on remaining issues in both syntactic and semantic data anonymisation algorithms, their advantage, belongings and summarised critics [31]. As part of their work, Clifton et al. highlight that the difference in various syntactic and semantic anonymisation origin models is less dramatic than previously thought. Differential privacy often provides the best empirical privacy for a fixed (empirical) utility level, but it might be preferable to adopt syntactic anonymity models for more accurate answers. Yet, both archetypes will suffer issues for large-scale data environments.

Regardless of its origin, data anonymisation has not been successfully applied to large-scale, multi-attribute, high-dimensional datasets in reasonable runtime with economic resources. Each approach suffers significant complexity constraints for large amounts of describing attributes (columns), leading to either massive information-loss or computation needs and therefore runtime or suffers privacy assurances through approximation approaches.

2.6 Quasi-Identifier Discovery

Based on Sweeneys work on the family of k-anonymity techniques [157, 166, 167], Byun et al. addressed the absence of diversity through equivalence classes and their information-loss by transforming the k-anonymity problem to a k-member clustering problem [23]. While Byun et al. approach work for numeric and categorical data with distance and cost functions, it does not guarantee approximation factors. The projection of quasi-identifier similarity for clustering purposes remains data-specific.

Xiao et al. presented anatomy [188], a novel technique that directly releases all quasi-identifiers and sensitive values in two separate tables. This, in combination with grouping activities, shall offer the capturing of correlation and minimise the error of reconstruction. Koot et al. use Kullback-Leibler distance to measure heterogeneity as the probability for each member of a group to quantify anonymity by a score outlining group members risk to be uniquely identified using a quasi-identifier [90]. Zhang et al. explored the scalability opportunities of horizontal scaling in cloud computing settings, combined with a quasi-identifier index-based approach to expedite data querying on large datasets. [193]

LeFevre et al. offered an approach utilising group-by-and-count methods to search for quasi-identifiers [93, 94]. Their experiments show promising results for small datasets, but relying on the minimal form does only reduce complexity on average by half, so $n - 1$. Fung et al. [69] extended the previously mentioned LKC-privacy and introduced an interesting architecture utilising multi-party computation, that works with the assumption of a maximum number of values that an adversary's prior knowledge might have. Yet, experiments are conducted on small n and its practical scalability remains unclear.

A different perspective on transactional, or event-based data is offered by Xu et al. [190, 191], who differentiate between private and non-private items and offer first experiments that are limited to very small datasets.

Narayanan et al. [118] demonstrated statistical de-anonymisation attacks on high-dimensional datasets for re-identifying individuals in the Netflix Prize dataset with tolerance for some mistakes in the adversary's background knowledge. Narayanan et al. [119] elaborated on the PII fallacy of the HIPAAs privacy rule when removing designated attributes against additional personally identifiable information (PII), as the elimination of all quasi-identifiers is not guaranteed. Soria-Comas et al. summarised the issue of re-linkage through quasi-identifiers and discussed data governance aspects addressing user consent, purpose limitation, transparency, individual rights of access, rectification and erasure [162]. Further, Soria-Comas et al. work raised the need for new privacy models designed from scratch with big data requirements in mind like continuous and massive data collected from multiple source systems forming multi-attribute and high-dimensional datasets.

As parallelisation itself is not completely new, Braghin et al. have contributed an optimised quasi-identifier scheme that utilises parallelisation for efficient QID discovery [20]. Given its detailed describing, and promising results, Braghin et al.'s work will serve as a comparison baseline for our experiments.

Previous work has shared insights into the parameterized complexity of the k-anonymity family. In 2011, Bonizzoni et al. showed that k-anonymity is W[1]-hard and admits a fixed parameter algorithm [18]. Around the same time, Dondi et al. presented that l-diversity is W[1]-hard [46]. Also, t-closeness proved to be FPT as well [101]. A precise classification of the quasi-identifier search as prerequisite remains open, while the majority of existing quasi-identifier discovery approaches have been demonstrated during experiments mainly on small datasets.

2.7 Related Fields

Further work on data transformation for anonymity appears in the data mining field, where a variety of work exists on addressing privacy-related constraints in publishing anonymised datasets. Some of this work includes but is not limited to regression models [64], clustering [175], and naive Bayes classification [176]. These methods are strongly focused on data mining tasks in specific application areas with well-defined privacy models and constraints. This is the case particularly when merging various distributed data sets to ensure privacy in each partition [40, 41, 117, 164, 178, 192].

2.8 Demarcation

We note from this discussion that a new perspective on data anonymisation operations is needed to handle the archetype of highly sparsely populated, multi-attribute, high-dimensional data.

Statistical and semantic data anonymisation approaches under the umbrella of differential privacy can offer a precise definition of privacy and a formal framework for determining when privacy is impossible. Yet, these approaches are bound to use-case sensitive settings, limited in their scope of applicability to large datasets and are NP-hard [55]. Previous complexity analysis for selected semantic techniques confirms the same objective [1, 56, 174]. Syntactic data anonymisation has been proven to be NP-hard as well [10, 112], while recent work offers perspectives to circumvent the NP-hardness issue and handle large datasets. Nevertheless, guaranteeing anonymity through the absence of any personally identifiable information remains an issue [140].

The anonymisation problem's NP-hard nature is an essential issue in considering the compliance challenges faced by data privacy and data protection practitioners in the face of recent privacy regulations like the EU General Data Protection

Regulation and the US California Consumer Privacy Act. In essence, the implication is that based on the current state of the art and the growing data dimensions, achieving perfect privacy is not possible, but rather approximately perfect privacy is. Given the contemporary legal interpretation of GDPR, self-contained anonymity postulates guarantees that no family of attribute sets uniquely identifies one original entity [32, 61, 137].

A new perspective on data anonymisation operations is needed to handle highly sparsely populated, multi-attribute, high-dimensional data and guarantee the absence of quasi-identifiers. This work will contribute novel methods and data processing sequences to transform datasets with the same characteristics for the problem of finding quasi-identifiers and data inferences that can violate privacy guarantees.

In the subsequent sections, we derive from the previous partial anonymisation problems and deduce why data anonymisation is an underestimated problem [126] by demonstrating its classification as W[2]-complete problem [16] which is fixed-parameter tractable through its combinatorial nature [84].

3

The Quasi-Identifier Search Problem

The discovery of quasi-identifiers is an integral part of the anonymisation processing to ensure privacy and, for various attacks, vectors aiming at de-anonymising published datasets. Hereunto, such processing aims to find a rare combination of attributes that can identify unique data records.

3.1 Formalising the Nature of Quasi-Identifiers

Unique attribute combinations that we refer to as quasi-identifiers (QID) are multiple attribute values that combine an indirect identifier. Table 3.1 illustrates an example, where the attribute values *cortisol* and *female* are unique in the given data snapshot and therefore identify the second row. Reversing the meaning, with the knowledge of these two characteristics, we can derive that the same person has been diagnosed with *COVID* and received *cortisol* (see Table 3.1).

Table 3.1: Sample health dataset

Gender	**ZIP**	**Drug**	**Disease**
male	10001	remdesivir	COVID
female	64123	cortisol	COVID
male	60617	cortisol	COVID
male	60617	remdesivir	COVID

To formalise such characteristic, we start by defining a feature:

Definition 1. *Feature*
A feature f is a function $f : E \longrightarrow A$ mapping the set of entities $E =$

$\{e_1, \ldots, e_m\}$ to a set A of all possible realisations of an attribute or attribute combination forming new single attributes. Additionally, $F = \{f_1, ..., f_n\}$ denotes a feature set.

Individual features, in fact, can be used as *standalone identifiers*:

Definition 2. *Standalone identifiers*
Let F be a set of features $F = \{f_1, ..., f_n\}$, where each feature is a function $f_i : E \longrightarrow A$ mapping the set of entities $E = \{e_1, ..., e_m\}$ to a set A of realisations of f_i. A feature f_i is called a standalone identifier, if the function f_i is injective, i.e. for all $e_j, e_k \in E :\ f_i(e_j) = f_i(e_k) \Longrightarrow e_j = e_k$.

Most prominent examples of standalone identifiers are user IDs, social security numbers, invoice or document IDs, but also telephone numbers or robust password selections, which by themselves are unique across a given, large enough dataset. The US Health Insurance Portability and Accountability Act does specify a list of attributes that can be used as standalone identifiers in healthcare settings (HIPAA) [124].

Building on this, the combination of features form candidates for quasi-identifier. A collection of any (not selective) attributes (columns) that uniquely identifies at least one entity (row) will be considered as quasi-identifiers (QID). More formally, we specify:

Definition 3. *Quasi-identifier*
Let $F = \{f_1, .., f_n\}$ be a set of all features and $B := \mathcal{P}(F) = \{B_1, .., B_k\}$ its power set, i.e. the set of all possible feature combinations.
A set of selected features $B_i \in B$ is called a quasi-identifier, if B_i identifies at least one entity uniquely and all features $f_j \in B_i$ are not standalone identifiers.

For several reasons, this work, like Manolis Terrovitis et al. [171], does not differentiate between sensitive and non-sensitive attributes. First, as one has seen in previous de-anonymisation attacks (background information, homogeneity, similarity, etc.), sensitive attributes will not be the only success factor in these attacks. Second, for high-dimensional datasets, describing an exhaustive collection of sensitive and non-sensitive attributes is impractical. User behaviour has distinct patterns that grow in proportion to the amount of data collected on the person. For instance, in medical datasets, behaviour patterns like medication adherence and drug intake patterns increase in uniqueness as the data points collected grow to several hundred attributes per patient.

Example 1. Imagine a dataset describing medical records as delineated in Table 3.2. While it is self-explanatory that a patient ID or its social security numbers (SSN) qualify as a standalone identifier, the combination of gender, birthday and ZIP can serve as quasi-identifiers to identify selective patient records uniquely. The restriction to selective patient records is important in this context since most patient records might be identifiable, while that does not have to apply to all given records.

Table 3.2: Sample health dataset with (Q)IDs

Patient ID	**SSN**	**Gender**	**DOB**	**ZIP**	**Drug**	**Disease**
1	433-46-1872	male	2020-11-23	60617	remdesivir	COVID
2	551-31-8713	female	2020-11-23	64123	cortisol	COVID
3	624-20-5414	male	2020-11-23	60617	cortisol	COVID
4	513-72-4512	male	2020-11-23	10001	remdesivir	COVID
..	..	..	..	..	..	..

Similar to this given an example, Sweeney et al. demonstrated based on the US census from 1990 that 87% of the entire US population (216 million out of 248 million) are uniquely identifiable through the combination of their gender, birthday, and ZIP [165]. We, therefore, draw on the definition of a *Feature*, to define *Self-Contained Anonymity* which captures the idea of anonymity of individual records or a dataset, as follows:

Definition 4. *Self-contained Anonymity*
Let E be a set of entities. A snapshot S of E is said to be self-containing anonymous or sanitised, if no family $\mathcal{F} = \{F_1, .., F_m\}$ of feature sets uniquely identifies one original entity or row.

For means of realising such self-contained anonymity, all instances of quasi-identifier must be found and defused or removed. To do so, we define the discovery of quasi-identifiers as follows:

Definition 5. *Find QID-problem*
Consider a dataset D and its features $F = \{f_1, ..., f_n\}$. For every feature combination $B_i \in B = \mathcal{P}(F)$, determine if B_i is a Quasi-identifier.

In practical words, it implies that there remains no combination of attributes in the sanitised data snapshot that can be used to re-identify an individual through cross-linking that potentially exposes private or personally identifiable information (PII). Following the European Commission's opinion on anonymisation techniques [61], this definition therefore also satisfies both our highest ethical standards and current privacy legislation, including EU GDPR [32] and US California Consumer Privacy Act [75].

3.2 QIDs and their Counterpart of mpUCCs

In the field of data profiling, unique column combinations (UCCs) describe a similar behaviour to the previous example in Tables 3.1 and 3.2. Here, an attribute combination serves as unique column combinations (UCC) if these attributes uniquely identify all records in a table. In the privacy context, it is already sufficient to link an individual record instead of all. Therefore our context of *cortisol* and *female* in Table 3.1 is a subset of UCCs, also known as *maximal partial unique column combinations* (mpUCCs). Maximal partiality refers to the condition that a UCC serves the minimal subset of records (per row) and is maximal partial to the entire record population. According to this, the search for quasi-identifiers is equivalent to the discovery of mpUCCs.

It is also worth looking into these close-by research fields as there are often synergies in their research activities. From the context of UCCs, we borrow the findings of the *minimal* form by formally stating:

Definition 6. *Minimal Quasi-Identifier*
A quasi-identifier $B_i \in \mathcal{P}(F)$ is called minimal if there is no combination of features $B_j \subset B_i$ that is also a quasi-identifier.

The formalised existence of minimal quasi-identifiers (mQIDs) implies that any superset of an already QID is a QID itself without the need of verifying.

Example 2. Continuing with Table 3.2 previous example, where the attribute combination of gender, birthday, and ZIP code has been used as a quasi-identifier. Given a working understanding of minimal quasi-identifiers (mQIDs), any additional attribute added to gender, birthday, or ZIP will remain a QID and reveal the same private data. As a result, the patient's SSN (551-31-8713) or disease (COVID) will be revealed by {gender, DOB, ZIP, drug}.

Besides the knowledge of the minimal form with UCCs, data profiling offers a variety of formalism, which we will use in the next section to determine the (parameterised) complexity of the "Find-QID" problem.

3.3 The Complexity of the Quasi-Identifier Search

The previous sections outlined the importance of discovering quasi-identifiers (QID) in a given dataset to avoid private data exposure by all means. These attribute combinations potentially leaking information about individuals can adopt any length and any composition of all available describing attributes.

To determine all QIDs, its search is quite complex as all potential candidates need to be considered. Hence, achieving the creation of those QID candidates, a potential algorithmic solution needs to generate all attribute combinations. Generally, the number of combinations equals:

$$C(n,r) = \binom{n}{r} = \frac{n!}{(r!(n-r)!)} \tag{3.1}$$

where n is the population of attributes and r the subset of n.

Considering that quasi-identifier tuples can have any length with more than one element, r must equal all potential lengths of subsets of attributes. We express this using the following equation:

$$C_2(n) = \sum_{r=1}^{n} \binom{n}{r} = \sum_{r=1}^{n} \frac{n!}{(r!(n-r)!)} = 2^n - 1 \tag{3.2}$$

This implies that for a dataset with n describing attributes, nearly 2^n attribute combinations need to be generated and evaluated to ensure no QID remains. Any solution which neglects candidates from the search room ultimately risks exposing private information.

To quantify the complexity and formally state the *Find-QID* problem, we will introduce the theorem of parameterised complexity and utilise similarities of the *Find-QID* to UCC to state a polynomial reduction.

3.3.1 Theorem of Parameterised Complexity

According to Bläsius et al. [16], if I is an instance of a decision problem and $k \in \mathbb{N}^+$ is a parameter, the pair (I, k) expresses then an instance of the corresponding *parameterised problem*. Further, the running time of an algorithm is calculated, taking into account both the input size $|I|$ and the parameter k. When a given instance can be solved in $O(f(k) \times p(|I|))$, where p is a polynomial and f is an arbitrary computable function, the problem is fixed-parameter tractable (FPT). Consequently, the problem belongs to the complexity class FPT. In addition, the algorithm is said to run in FPT-time.

Let us say there are two parameterised problems, P and P'. An FPT-time algorithm that transfers an instance (I, k) of P to an equivalent instance (I', k') of P', with the parameter k' depending purely on the value of k (rather than $|I|$), is known as a parameterised reduction from P to P'. It's worth noting that reducing a (hypothetical) FPT-algorithm for P' to an FPT-algorithm for P yields an FPT-algorithm for P. As a result, when their parameterised complexity is considered, P is at most as difficult as P', which we denote as $P \leq_{FPT} P'$. We assume that P and P' are FPT-equivalent problems if $P' \leq_{FPT} P$ still holds [16, 47, 48].

The parameterised reduction produces a hierarchy of complexity groups, known as the W-hierarchy, with each class defining a complete problem. In propositional logic, we use Boolean formulas to describe the desired family of problems. Let φ be an example of such a formula. If exactly k variables are set to true in this assignment, it is a satisfying truth assignment for φ with Hamming weight k. If it can be written as repetitive conjunction of disjunctions of literals with $t - 1$ alternations between conjunction and disjunction, the formula φ is t-normalised. Observe that a Boolean formula is 2-normalised if it is in conjunctive Normal Form (CNF) and 3-normalised if it is a conjunction of subformulas in Disjunctive Normal Form (DNF).

The problem Weighted t-Normalised Satisfiability (WtNS) is to decide for a given t-normalised formula φ and a positive integer k whether φ has a weight k satisfying assignment. For any $t \geq 1$, a parameterised problem P is said to be in the complexity class W[t] if $P \leq_{FPT} WtNS$.

The classes $FPT \subseteq W[1] \subseteq W[2] \subseteq W[3]$ form an ascending hierarchy, with all inclusions considered to be proper, but this remains unproven [47]. The lower a problem ranks in the W-hierarchy, the higher we consider the chances of finding an FPT-algorithm to solve it.

3.3.2 Previous Parameterised Complexity of Unique Column Combinations

Sets of attributes (columns) that qualify as identifiers for a fixed given data set are known as unique column combinations (UCCs). Bläsius et al. derived that detecting key candidates like UCCs or functional dependencies in a given data set is, in terms of its parameterised complexity, in general, as hard as finding a Hitting-Set [16]. This holds when the necessary parameters k denote the upper bound for the size of the key candidate, the functional dependency, and the Hitting-Set, respectively. In the following, the problem of determining if a given data set contains a key candidate with a size of no more than k, where

k is appropriately chosen from the positive natural numbers, is referred to as "Unique". Since the Hitting-Set problem and weighted 2-normalised satisfiability are both NP-complete and W[2]-complete, the following chain was discovered to be valid: Hitting-Set, Unique, and weighted 2-normalised satisfiability. Thus, all problems belong to the same equivalence class [16]. The derivation of the chain above from a sequence of reductions defines all problems as belonging to the same parameterised complexity equivalence class.

Since data privacy law aims to protect each individual record, i.e. row of a data set, even column combinations that can only be used to define a single entry must be pre-processed. As a result, we move on to studying the latter by formally presenting them as quasi-identifiers.

3.3.3 Parameterised Complexity of the Quasi-Identifier Search

In the previous subsection, we explained that discovering UCCs is a W[2]-complete problem and that syntactic data anonymisation approaches are NP-hard. A parameterised complexity classification permits classes beyond para-NP, especially for multivariate algorithms.

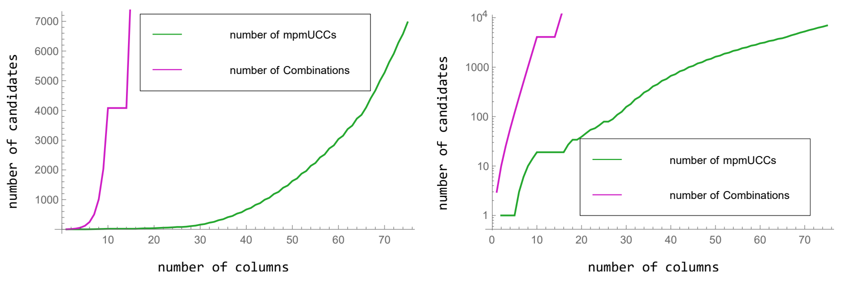
In typical circumstances, the classification is based on the number of input bits, but additional in- or output parameters necessitate a finer scale. Following the parametric method, problems can be reduced to propositional satisfiability (SAT), for which a selection of exceptionally efficient and resilient SAT solutions are available [14, 30, 74, 107, 108, 155].

Making the loop and attempting to construct an attribute of a database that violates privacy assurances, we discover that removing quasi-identifiers satisfies the anonymity criterion (see Definition 4). Contrariwise, the presence of even a single quasi-identifier in a database allows at least one record to be linked to its individual, allowing for the achievement of a shared goal in data profiling. Thus, eliminating quasi-identifiers is both required and sufficient.

To verify if any arbitrary set of features of size r satisfies the QID Definition 5 in a particular context, one can assess all possible attribute combinations of size r in an exact fashion (not probabilistic). That leads to

$$C(n,r) = \binom{n}{r} = \frac{n!}{(r!(n-r)!)} \tag{3.3}$$

tuple candidates where n is the total number of features and r the size of the subset of features with r to be chosen such that $r \in [n] := \{1, ..., n\}$. Because quasi-identifiers can range in size from 1 to n, the total number of feature combination serving as quasi-identifier candidates sums up to:



(a) Linear Scale (b) Log Scale

Fig. 3.1: Candidates and mpmUCCs

$$C_2(n) = \sum_{r=1}^{n} \binom{n}{r} = \sum_{r=1}^{n} \frac{n!}{(r!(n-r)!)} = 2^n - 1. \tag{3.4}$$

As there are around 2^n conceivable column combinations, the number of attribute combinations grows exponentially in n (see Figure 3.1a & 3.1b), necessitating a determination of whether they serve as quasi-identifiers or not.

By arranging the binomial coefficients into Pascal's triangle, such quasi-identifier candidates' combination distribution and symmetry can be depicted. The exponent value corresponds to each stage of the triangle (see Figure 3.2). This Pascal's triangle shows the exact size of tuples for each hierarchy layer of the combination tree, provided the number of levels r. For a four-column data set ($r = 4$), the first layer has one element, the second layer has four, the third layer has six, the fourth layer has four, and the final layer has one element. The real search tree (see Figure 3.2) then delineates this symmetry.

A reduction of layers and combinations, i.e. through their minimal form (see Definition 6) is possible. This, however, also results in exponential growth.

In Section 3.2, we highlighted equivalence of quasi-identifiers to maximal partial unique column combinations (mpUCCs) in the field of data profiling. These, in turn, have minimal forms, known as mpmUCCs. Every attribute combination that serves as an identifier for any entity remains an identifier when additional attributes are introduced to it [129], i.e. the attribute tuple of mpmUCCs is contained in the tuple of mpUCCs:

$$mpmUCC \subseteq mpUCC. \tag{3.5}$$

Further, the following inclusion relation exists:

$$mUCC \subseteq UCC \subseteq mpUCC \tag{3.6}$$

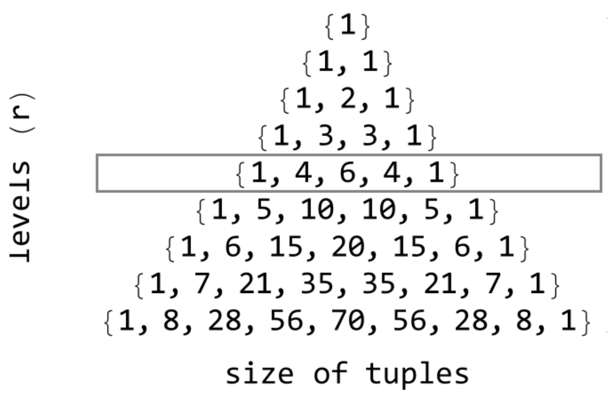


Fig. 3.2: Pascal triangle representation of the tuple sizes

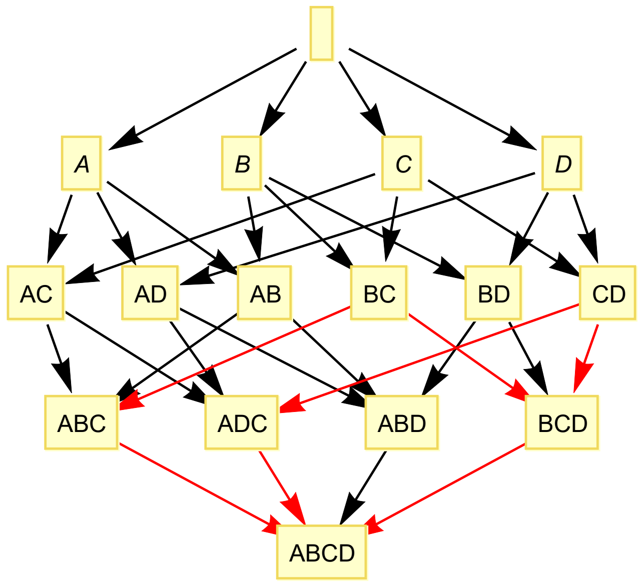


Fig. 3.3: Candidate search tree - red lines highlight obsolete UCCs covered by their minimal sibling (BC and CD)

This is true because every UCC is a mpUCC, but not every mpUCC can qualify as a UCC. It may only be able to identify a portion of the data set's entries uniquely. This concludes with

$$mUCC \subseteq mpmUCC \subseteq mpUCC \tag{3.7}$$

for which applies $UCC \nsubseteq mpmUCC$ and $mpmUCC \nsubseteq UCC$. Finally, detecting and eliminating the minimal forms is sufficient and algorithmically more effi-

Table 3.3: Examples for inclusion affiliations of UCCs and mpUCCs

(a) for $UCC \nsubseteq mpmUCC$

a	b
1	1
2	1
3	1
4	1

(b) for $mpmUCC \nsubseteq UCC$

a	b
1	0
2	1
3	1
4	1

cient, and empirical studies should produce significantly better results than the worst-case analysis performed.

Assume we have an algorithm that explores all potential column combinations with a size no larger than k contained in $\mathcal{P}(F)$ for some fixed parameter k as described previously. Suppose the number of unique entries discovered while scanning the rows is also computed and saved at each step. In that case, there is no substantial contribution to the algorithm's complexity in terms of running time to account for.

Because the relation $UCC \subseteq mpUCC$ holds, a run of this method would also allow us to determine if the issue "Unique" examined by Bläsius et al. in [16] has a negative or positive answer: If and only if there is a mpUCC of size k that uniquely identifies all rows in the dataset, there is a UCC in the dataset of size at most k. From this follows that

$$mpUCC \geq_{FPT} UCC$$

where the decision problem is associated with the object of concern; mpUCCs and UCCs, and respectively, the latter relates to the decision problem "Unique".

As it is known that the "Unique" (UCC) problem can be reduced to the Hitting Set

$$\text{HITTING SET} \leq_{FPT} \text{UNIQUE} \leq_{FPT} FD_{fixed} \leq_{FPT} \text{FD}$$

and Hitting Set, Unique, and FD are W[2]-complete. Further, a mpUCC and therefore quasi-identifiers (QID) are by definition a Functional Dependency (FD). We can therefore derive, that the decision problem mpUCC and therefore "Find-QID" is W[2]-complete as well. As a result, any algorithmic solution that depends on detecting mpUCCs to eliminate the latter may be solved efficiently for small values of the fixed parameter (n), but in large, high dimension it cannot. Yet, this is required to fulfil the anonymity requirement from Definition 4.

Other W[2]-complete problems include deciding if a given graph includes a dominant set of size k or solving the “short multi-tape Turing machine acceptance” problem [81, 111]. Techniques for the *Find-QID* problem that offer a quicker worst-case runtime than superpolynomial would, through reduction, provide a solution to the “Unique” problem of UCC as well.

4

Algorithmic Realisation of the Find-QID Problem

With the knowledge in mind that the “Find-QID” problem is quite complex and at least W[2]-complete in its complexity classification, we shift our attention towards algorithmic realisation and optimisation approaches. To do this, we start by specifying a rudimentary, exact algorithm that will serve as a benchmark for further optimisations. Afterwards, a greedy approach will be introduced, discussed and evaluated.

4.1 The Exact Search Schema

We do not distinguish between sensitive and non-sensitive attributes because classifications of sensitive and non-sensitive attributes are the primary enabler for de-anonymisation attacks based on the anonymised dataset’s semantics. However, we do separate standalone identifiers from attribute combinations, forming a quasi-identifier like behaviour patterns such as medication adherence or drug intake pattern uniqueness increases with available data points since they must be processed differently. Hence, the *QID search* can be split as well into two parts. The first objective is to determine all standalone identifiers (see Definition 2); all remaining attributes are evaluated in a second step whether their combination qualifies as quasi-identifier.

4.1.1 Determining Standalone Identifiers

Following Definition 2, any attribute that can uniquely identify individual data records may serve as a standalone identifier. Removing these standalone identifiers reduces the number of attribute combinations needed to be processed afterwards for the QID search. The treatment of standalone identifiers to achieve anonymity typically occurs as debarment or exclusion from the original dataset. This reduces the information loss, time complexity and ensures data anonymity.

Occasionally, official guidelines of attributes qualifying as standalone identifiers exist, like included in the US Health Insurance Portability and Accountability Act (HIPAA)[1]. Utilising this metadata is a good starting point, yet, the ambition is not to rely on static lists rather determining on the fly whether attributes satisfy the Definition 2.

> The US Health Insurance Portability and Accountability Act explicitly list 18 attributes that are required to be removed. These attributes include but are not limited to email addresses, IP addresses, geographic details, phone or fax numbers, certificate IDs, URLs, biometric identifiers and basically any unique number or characteristics. Additionally, a constraint is mentioned that "no actual knowledge residual information can identify individuals" in the remaining data [66]. While the former is commonly known as a standalone identifier, the latter will be referred to as a quasi-identifier. Examples of such are illustrated in Table 3.2.

What might occur to the reader is the fact that attributes serving as identifiers correlate with high cardinality. Meaning, an attribute like a genome or password hashes, which in general is not an identifier, could be an identifier in a given dataset if it includes a unique value across the dataset and therefore identifies a single entity with some given genome.

Furthermore, we define cardinality and entropy as follows:

Definition 7. *Cardinality*
The cardinality $c \in \mathcal{Q}$ of a column or an attribute is:

$$c = \frac{\text{Number of unique entries}}{\text{Total number of entries}}$$

Definition 8. *Entropy (Kullback-Leibler Divergence)*
Let p and q denote discrete probability distributions. The Kullback-Leibler divergence or relative entropy e of p with respect to q is:

$$e = \sum_i p(i) \cdot \log\left(\frac{p(i)}{q(i)}\right)$$

With those metrics in mind, the first algorithmic step consists of iterating over all individual attributes in linear time and calculating their metric value. An

[1] `http://www.dhcs.ca.gov/dataandstats/data/Pages/ListofHIPAAIdentifiers.aspx`

initial threshold highly depends on data (size) characteristics. Depending on the threshold specified, information-loss will be balanced against the second algorithmic part's runtime for evaluating attribute combinations (detailed in Subsection 4.1.2).

This originates to the point that fewer remaining attributes will generate fewer attribute combination candidates that need to be verified. No bijective linkages from the dataset to the original entities remain by removing these standalone identifiers with high cardinality. Certainly, it is still possible to combine several attributes for re-identification. Therefore, we must identify and remove these attribute combinations in the following section.

> The information-loss during syntactic data anonymisation differs from the application field and industry sector, as these dimensions influence the consistency of the underlying dataset. The more attribute diversity exists, and the more describing records (rows), the more algorithmic opportunities are present to remove data inferences while reducing information-loss. With the opposite extreme, if only a single describing attribute is present, like a phone number, the algorithm is limited to alternate its value resulting in larger information-loss. Industry sectors like (car-) insurance, for instance, suffer larger information-loss than retail on average based on the same data population.

4.1.2 Assessing Quasi-Identifiers

Assessing quasi-identifier is similar to finding candidates for a primary key or (maximal partial) unique column combinations (mpUCC) in the data profiling field. As previously denoted, unique column combinations (UCC) are tuples of columns that serve as identifiers across the entire dataset. However, maximal partial UCC can be understood as identifiers for (at least) one specific row. This means one searches for the UCC for each specific row (maximal partial). Papenbrock et al. offer an assessment of efficient algorithms developed for functional dependency discovery [130]. Yet, those are not directly applicable to QID discovery and require adaptions to the special case of UCCs with a larger complexity.

In that first starting point, the population of attribute combinations serving as QID candidates need to be generated. This can be a list of tuples, or an iterable, as long as all combinations of any tuples length are included (see Algorithm 1). The implementation of generating combinations is available in almost any modern programming language and well explored so far [27, 132].

Algorithm 1: Pseudocode for generating QID candidates

```
1 generate QID candidates (df);
  Input  : df as table of data
  Output: List of QID candidates
2 listOfCandidates = initialise an empty list;
3 listOfColumns = retrieve all columns from df;
4 numberOfColumns = count elements in $listOfColumns$;
5 for for r in range(2, numberOfColumns) do
6     candidatesForR = generate combinations for $listOfColumns$ with
        tuple length $r$;
7     listOfCandidates += append $listOfCandidates$ with $candidatesForR$
8 end
9 return $listOfCandidates$
```

Neglecting any optimisations like candidate sorting, for now, the second step involves assessing each candidate. Therefore, the algorithm loops through the list or iterable and checks if each corresponding candidate fulfils the QID Definition 5. Algorithm 2 details the procedure. In a dataset given for a designated attribute tuple as QID candidate, all attribute values are being grouped, and the appearance of the same attribute values is being counted. If there is at least one instance where this QID candidate uniquely identifies k rows or less, the candidate satisfies our previous QID Definition 5. This corresponds to Sweeneys k-anonymity requirement [167], for $k = 2$ meaning that at all times, there are at least a group of two individuals sharing the same data characteristics (non-bijective).

Utilising known grouping and counting activities for aggregation promise several advantages. First, the concept and algorithmic implementation is proven and easy to understand. Second, various research and optimisation approaches exist, including caching and sorting techniques summarised and demonstrated by Mueller et al. [115] and Plattner et al. [134]. Most prominently, *group by* and *count* is available in most common frameworks, libraries or database technologies, making its usage very practicable without the need for sophisticated or complex implementation efforts.

Implementing this algorithmic approach, Figure 3.1 delineates the candidate growth of the "Find-QID" search. The reader will quickly acknowledge that the processing quickly becomes unfeasible due to its exponential growth. Therefore, optimisation methods will be addressed in the following sections. Improving pro-

Algorithm 2: Evaluating quasi-identifier candidates

```
1  Assess QID candidates ($df$);
   Input  : $df$ as table of data
   Output: $Boolean$ of QID candidates
2  listOfCandidates = generate QID candidates for $df$;
3  listOfQIDs = initialise an empty list;
4  k = define $k$-anonymity constraint (e.g. 2);
5  for for candidateTuple in listOfCandidates) do
6      exposedRows = SELECT COUNT(*) FROM $df$ GROUP BY
        candidateTuple HAVING COUNT(candidateTuple) $< k$;
7      representativeCount = count number of $exposedRows$;
8      if representativeCount $> 0$ then
9          listOfQIDs += remember $candidateTuple$ as QID and append to
            list
10     end
11 end
12 return $listOfQIDs$
```

cessing can happen in various dimensions. While earlier we elaborated on reducing the incoming attribute amount with the side effect of higher information-loss, Section 4.2 offers greedy processing and a smart queuing to minimise practical runtime (not worst-case). Further, Section 4.4 discusses engineering options for parallelised execution.

4.2 Optimisation Method: Greedy Approach of Sorting Candidate Queues

It is being recognised that preventing private data exposure through the existence of quasi-identifiers (QID) is W[2]-complete. We observed exponential nature during the exact processing of all QID candidates in the previous section. To counter the runtime growth, a greedy approach will be presented in the following by sorting the candidate queues efficiently for accelerated processing results.

4.2.1 Minimal Quasi-Identifiers

In Section 3.2, the *minimal* form has been introduced. As of Definition 6, when imaging the Find-QID within a tree structure, the discovery in one branch of the search tree can be stopped as soon as a minimal quasi-identifier is found (see Figure 3.3). This is similar to Papenbrock et al.'s [129] approach to handling maximal partial UCCs. Such processing improves computation time dramatically since all super-sets can be neglected from the candidate queue. Dependent on the nature of the dataset, yet first testing reveals that most mpmUCCs appear in the first third of the search tree, at most in the first half. This implies a reduction of layers and combinations, i.e. by half for their minimal form, is possible. However, this still leads to exponential growth due to the *symmetry of the binomial coefficient*, which results in

$$\frac{2^n}{2} = 2^{n-1} \tag{4.1}$$

combinations to be processed and evaluated. The binomial coefficients' symmetry and combination distribution can be delineated by arranging the binomial coefficients to form Pascal's triangle, where each Pascal's triangle level corresponds to an n value. Figure 3.2 depicts this phenomenon, where the Y-axis answers to r the number of attribute levels and the x-axis to the tuple sizes. So, for the search tree illustration from Figure 3.3 with four attributes {a,b,c,d} the first and last tree-level has one tuple, the second four and the middle six tuples. We still have exponential growth by reducing the layers and number of combinations by just considering the minimal form.

4.2.2 Utilising QID Tuple Characteristics

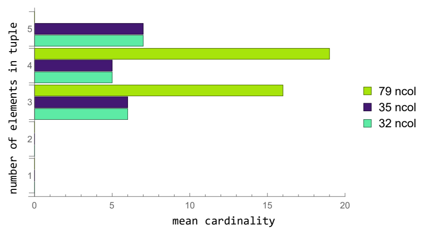
Nonetheless, this example conveyed an important observation as there are structural similarities in the candidate tuples. When utilising the entropy and cardinality metrics generated in Subsection 4.1.1 to determine the standalone identifiers, more interesting observations can be made.

Figure 4.1 delineates different metric angles on found quasi-identifiers. As metrics, cardinality based features like the sum of their cardinality (see Figure 4.1d), the mean cardinality (see Figure 4.1a), its mean value (see Figure 4.1c) or the number of elements per tuple (see Figure 4.1b) are generated. Given the observed distribution of tuple sizes regarding their elements expressed, one may observe: The more tuples exist, the more we can filter given a static threshold.

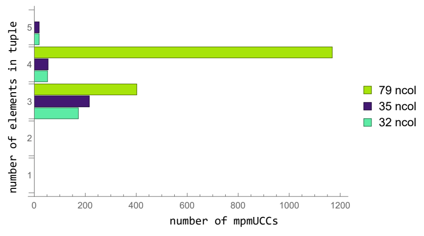
This knowledge can be used to implement a greedy approach by sorting the candidate queue based on their likelihood of success. If no combination candidates are left for assessment after filtering

- while the tuple length under examination is incomplete concerning the rearranging of the binomial coefficients, or
- while not all tree branches are covered by the already found minimal quasi-identifier,

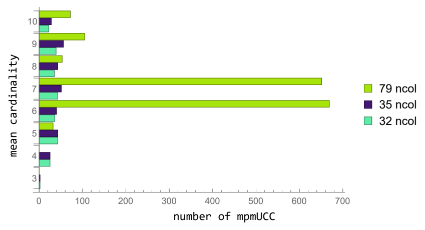
one may decrease these thresholds successively. Having found a minimal quasi-identifier (or mpmUCC), it is crucial to review its direct neighbours. Only if



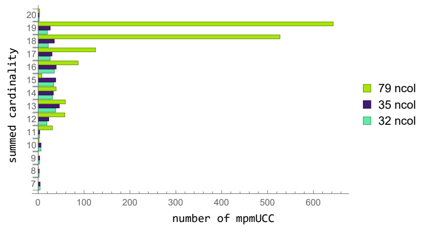
(a) mean cardinality vs elements in tuple



(b) number mpmUCCs vs elements in tuple



(c) number mpmUCCs vs mean cardinality



(d) number mpmUCCs vs summed cardinality

Fig. 4.1: Characteristics of mpmUCCs serving as optimised starting point

neither the parent nor sibling neighbour is a (minimal) identifier can exit the inspection for this branch. After each increment of the tuple size, the QID candidate queue is iteratively sorted, and super-sets of already identified QIDs are removed from the queue.

A pseudocode structure for this procedure is depicted in Algorithm 3. Here, we notice the procedure is exact and not heuristic and performs very well against all existing approaches. We will show in Figure 4.2 that almost 98.5% of all attribute combinations can be trustworthy skipped due to the sorted QID candidate queue. Through this incremental and iterative procedure, the worst-case runtime remains. Still, the effective runtime drops significantly, allowing us to process easily datasets of two-digit entities describing attributes in a few seconds on common hardware.

To reduce the set of combinations a priori, the clustering of similar columns based on their relative entropy or cardinality by k-means clustering has been discussed earlier [79]. By evaluating cluster representatives, the intention is to project results on other cluster members. However, this does not effectively consider all existing representative combinations regarding possible tuple lengths and, therefore, does not identify even half of the existing quasi-identifiers. Also, we kindly denote that creating overlapping groups of columns, where the overlapping columns are those with a high entropy or cardinality, does not overcome this issue because combinations of columns are not being represented in

Algorithm 3: Prioritisation of quasi-identifier candidates based on their metrics

```
Prioritise quasi-identifier candidate
  ($combination, quasiIdentifiers, thresholds$);
Input  : Array $quasiIdentifiers$ as list of already identified quasi
         identifiers,
         Set combination as a tuple of attributes forming a candidate,
         Dictionary $thresholds$ defining the thresholds for selection
Output: $True$ if the quasi-identifier candidate shall be evaluated,
         $False$ if the quasi-identifier can be skipped and neglected
// check if combination is a superset of already identified quasi-identifier;
for quasiIdentifier in quasiIdentifiers do
    if set(combination).issuperset(quasiIdentifier) then
        return False;
    end
end
// filter based on configured thresholds;
if MeasureSummedCardinality(combination) <
  thresholds['minCardinalityToBeConsidered'] then
    return $False$;
end
if MeasureMeanCardinality(combination) <
  thresholds['overallMinMeanCardinality'] then
    return $False$;
end
if MeasureMeanCardinality(combination) <
  thresholds['minMeanCardinality'] & length(combination) >
  thresholds['minLengthForMinMeanCardinality'] then
    return $False$;
end
return $True$;
```

the groups. In sum, we kindly dissuade against any kind of cluster-based result projection rather expand on the thought to use data characteristic similarities for efficient queue sorting.

Optimisation through a greedy approach does promise runtime accelerations. In the following Section 4.4 we will evaluate engineering tweaks to parallelise execution and improve runtime.

4.3 Optimisation Method: Parallelisation of the Find-QID

Optimisations can be of structural nature as offered in the previous Section 4.2 through a greedy approach, wherefore the QID tuple characteristics are being exploited to sort the candidate queue by their likelihood of success. As an alter-

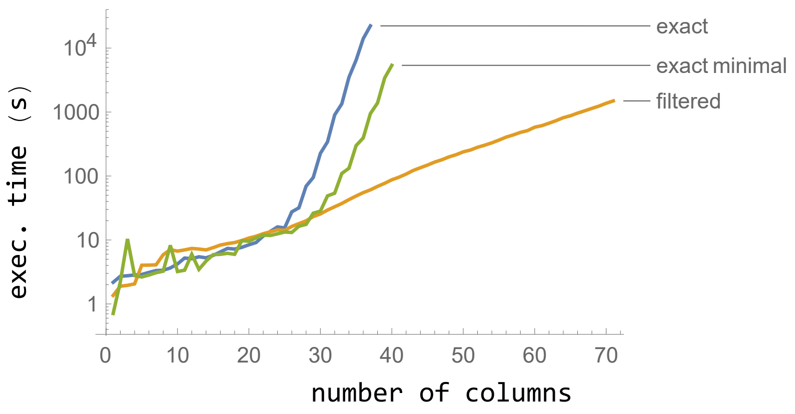


Fig. 4.2: Greedy optimisation comparison for the Find-QID problem

native, engineering opportunities can also accelerate data processing. A common scheme for this purpose is parallelisation, where multiple tasks are computed simultaneously in a multi-core environment.

In the "Find-QID" problem, the computation heavy piece is essentially the iteration across all candidates and their assessment of anonymity satisfaction given the sheer amount of combinations. To verify the anonymity for a candidate, the same algorithmic steps are applied as presented in Section 4.1.2. The incremental nature of all computation activities within the layer (same candidate tuple length) can be parallelised as their calculation does not depend upon each other. When pointing the reader's attention towards Figure 3.3 for a second time, the levels of attribute combination length are clearly represented. Starting at the top, the first tuple length is none, then one, two, three, and finally four-element tuples. As there are no dependencies within a given layer, just in-between parent and siblings, such nature can be utilised for parallelised task execution. In particular, each QID candidate within the same level can be parallelised in the best case considering chunking to reduce process spawn efforts. Incrementing levels (L) should be done sequentially brings two advantages: First, it avoids the necessity of checking potentially superfluous candidates which siblings might have already satisfied the QID requirements. Second, processing candidates with the length of the same tuple maximises cache hits for the underlying dataset. Yet, the compute-heavy tasks within a specific level can be parallelised without any risk of dependencies. And after each increment of L, the pre-selection will be applied again to filter out supersets of minimal QIDs and significantly reduce candidates in the next iteration.

Algorithm 2 depicts the original pseudocode, wherein line 5, the combinatoric most challenging iteration, is specified. Parallelising this loop execution by in-

Algorithm 4: Evaluating quasi-identifier candidates

```
1  Assess QID candidates ($df$);
   Input  : $df$ as table of data
   Output: $Boolean$ of QID candidates
2  listOfCandidates = generate QID candidates for $df$;
3  candidatesQueue = create queue and fill with listOfCandidates;
4  listOfQIDs = initialise an empty shared list;
5  k = define $k$-anonymity constraint (e.g. 2);
6  for for process in pool) do
7      candidateTuple = pop candidate from candidatesQueue queue
         exposedRows = SELECT COUNT(*) FROM $df$ GROUP BY
         candidateTuple HAVING COUNT(candidateTuple) $< k$;
8      representativeCount = count number of $exposedRows$;
9      if $representativeCount > 0$ then
10         listOfQIDs += append $candidateTuple$ to $listOfQIDs$
11     end
12 end
13 return $listOfQIDs$
```

troducing some queue logic and multiprocessing as specified in Algorithm 4 promises quicker results and pretends a shorter runtime. Multiprocessing comes with the cost of resource management. In the CPU context, this implies process management and spawning, forking a process, its context and memory to n-times for n-tasks [134]. In the process context hence, a common scheme is to re-use processes in a process pool. Additionally, sufficient chunking of process tasks can be implemented, offering processing of batched tasks within an individual process instance. As efficient multiprocessing is a well-explored field, the reader is kindly referred to the latest research [6, 63].

The verification of satisfying the QID Definition 5 remains by utilising a *GROUP BY* and *COUNT* (see Section 4.1.2) as aggregation technique. This way, one can rely on established and well explored caching and sorting optimisations discussed by Mueller et al. [115] and Plattner et al. [134].

With the parallelised assessment of QID candidates, the effects are visible through the flattening in the curve growth for the *number of columns* below $n < 28$ (see Figure 4.2). This asymptote can be stretched and tightened based on the available cores to execute tasks in parallel. It is fair to assume that the parallelisation mimics an acceleration of runtime yet is highly limited to the performing cluster's available hardware specifications and therefore simply delays the expected exponential runtime growth to some degree. Indeed, the distribution of workload links to a different field, namely *system scaling.*

There are two key concepts for scaling a system, **vertically** and **horizontally**. For vertical scaling, effectively, more hardware is combined on the same motherboard bus (scale-up). This has technical limitations but can currently be scaled to several hundreds TB of RAM for instances with lightning-fast data transfers [133, 134] where the motherboard bus becomes the bottleneck. On the other side, horizontal scaling goes back to some MapReduce approach. The workload is broken down and distributed to several working nodes, and their partial results back combined, assembled and consequently returned. This is also known as a split-apply-combine strategy [182] and offers almost infinite scaling capabilities (scale-out) but implies relatively slow network I/O and significant communication overhead.

The parallelised assessment of QID candidates offers temporary relief of the NP-hard "Find-QID" problem. Yet, scaling CPU capacity quickly reaches its limits. In modern enterprise environments, a few hundred CPU cores are available, as we will mimic in our experiments (see Appendix B). The utilisation of such capacities goes along with economic aspects, which do extremely differ on a use case basis. Hence, purely scaling hardware is not a desired or generally feasible option. As parallelisation itself is not entirely new, Braghin et al. have published a parallel QID search scheme which will be used as a baseline for comparison in the following sections [20].

4.4 Optimisation Method: GPU Accelerated Find-QID through Vectorisation

One perspective on addressing combinatorial problems is to parallelise their execution, as previously discussed. In the domain of data anonymisation, such parallelisation has been essentially restricted to the number of CPU cores available in the processing node. By combining hyper-threading and algorithmic multithreading, various threads can be processed simultaneously, resulting in a performance boost. Particularly for privacy settings, Nayahi et al. [120] explored options of horizontal scaling to scale-out calculations with a MapReduce principle of *divide and conquer* large process tasks among many processing nodes [42]. However, when more nodes are added, the network I/O and infrastructure requirements increase, limiting the system's realistic scalability.

4.4.1 Trends & Concept for GPU Compute

Observing the latest developments with GPU acceleration, a similar methodology can be applied for the "Find-QID" problem. Traditionally, graphics processing units (GPU) promise high computations per memory access, high compute density, high throughput, high latency tolerance combined with deep pipelines (>100 stages). Yet, GPU memory capacities are still limited, unlike CPU architectures with large L1-L3 caches (see Illustration 4.4). A consistent decrease in main memory cost can be observed for CPUs while at the same time its capacity growths.

Individual CPU cores are fast and savvier than the instruction set of a single GPU core. The immense parallelism and the sheer number of GPU cores more than make up for the single-core clock speed differential that limits instruction sets. Yet, the same decreasing pricing evolution can be observed for GPUs as well, where Figure 4.3 delineates the rapid price decline per transistor over time.

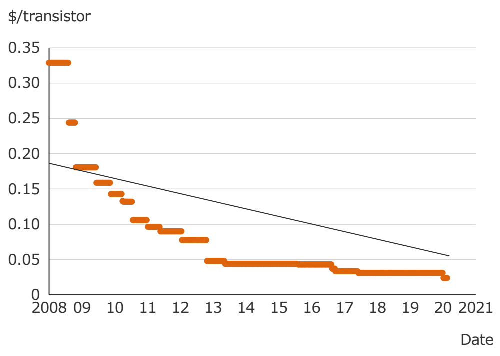


Fig. 4.3: Price trends for GPU hardware by USD/transistor

These GPU trends combined promise a huge compute potential for a fraction of a price with similar CPU capacities. Instead of individual scalar-based compute like with CPUs, GPUs execute vector- or even tensor-based calculations. Figure 4.5 illustrates these compute differences. Yet, these adjustments in architecture require different data processing as well. A traditional L1-L3 cache hierarchy illustrated in Figure 4.4 becomes obsolete, as GPU cores all operate on the same memory simultaneously, just on different data subsets. There are, however, limitations currently on the applicability of GPU usage for computing besides traditional video rendering. GPU chips promise massive-parallelisation, but currently, their data I/O is extremely restricted on the bus capacity. Shifting datasets exceeding main memory capacity will not effectively utilise a GPU. Hence, tasks, where most efforts are on iterative compute on the same or similar data, are desired. Besides the video rendering, the hashing and brute force scenario like for crypto mining has been proven [44, 169] but also enumeration problems like the "Find-QID" one fulfils these criteria.

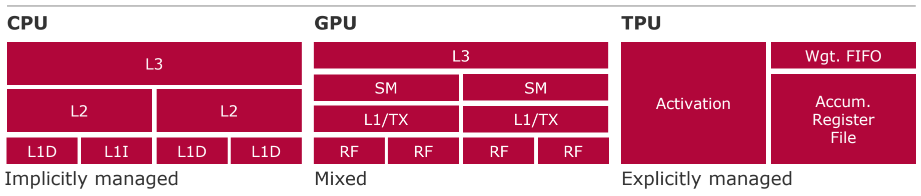


Fig. 4.4: GPU hardware architecture

As one can not simply execute a CPU-like code base on a GPU, several phases of pre-processing are necessary. These include:

- (pre-)fetching the designated dataset into main memory
- converting the dataset into a suitable format, for instances Apache Arrow to accelerate unified memory management between CPU L1-L3 and GPU memory and minimised data transformation steps
- applying some compressed or mapping technique like dictionary encoding of attribute values to have a numeric representation
- generating combinations of attribute value tuples and compose vectors

For the next step, one can leverage existing CUDA frameworks to massively perform the vectorised calculation of the probabilistic appearances after loading the pre-processed dataset in GPU memory. Section 4.4.2 will detail out the implementation steps.

4.4.2 Vectorising the Find-QID

For the designated "Find-QID" problem, we noticed a significant growth in candidates and processing time. This increase worsens for data sets with high-dimensions. Essentially, two computation intense pieces can be summarised as the iteration across all candidates given the sheer number of combinations and their assessment of anonymity satisfaction requiring cross-checking of all attribute values groups. Both instances can be vectorised.

For the latter verification of QID candidates, the same algorithmic steps are applied as presented in Section 4.1.2. Yet, their implementation slightly differs. Figure 4.5 illustrates the methodology of vectorisation. Instead of individual scalar-based compute, where single operations are executed, vector-based calculation calculates multiple (non-overlapping) scalar simultaneously. Originally,

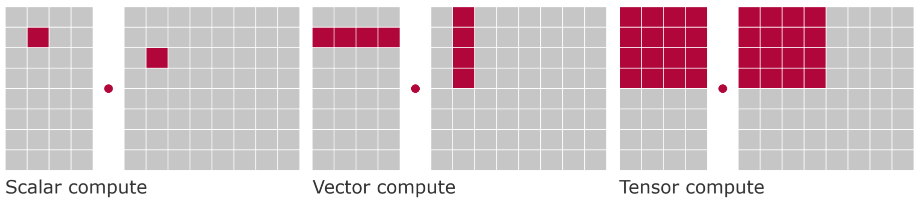


Fig. 4.5: Compute primitive differences between CPU (scalar), GPU (vector) and Tensor (tensor)

we used SQL alike operations of *GROUP BY* and *COUNT* to compute aggregations of attribute values accordingly to the QID candidate tuples and count their appearances. The desired equivalent class k is found by grouping the attribute combination candidate and counting the matched rows. We can use out-of-the-box libraries like the open data science framework cuDF[2] because this basic function of grouping and counting is supported in virtually every universe. cuDF is a RAPIDS Nvidia initiative that enables GPU acceleration out of the box. By chunking the entire dataset, the grouping activity is rendered as a vector operation [123, 128, 158] rather than a scalar one, potentially even on multi-GPU environments. Each CUDA core is allocated a data subset to operate on, and their partial results are combined afterwards. With the unified memory schema data, shifting the data on the motherboard bus between CPU and GPU becomes the actual bottleneck. The same performance gains are depicted in Figure 4.6 and will be further detailed in the next Section 4.5.

For the other computation heavy task of enumeration processing, we discussed in the previous Section 4.3 opportunities to parallelise the workload in a multi-core environment. Task parallelisation, chunking, caching and sorting all promise an uplift in compute time. Until now, however, such parallelisation has been effectively limited to CPU cores in the field of data anonymisation. The incremental nature of all computation activities within layers of the same candidate tuple length offers a natural staircase for parallelisation as their calculation does not depend upon each other. These candidates within the same step of a tuple length with size L can also be computed in parallel on a GPU chip. Compute and memory are shared and distributed across blocks inside each chunk, with many threads running independently and concurrently (see Figure 4.5).

The implication of both vectorisations will be empirically studied and discussed in the next Section 4.5.

[2] `https://github.com/rapidsai/cudf`

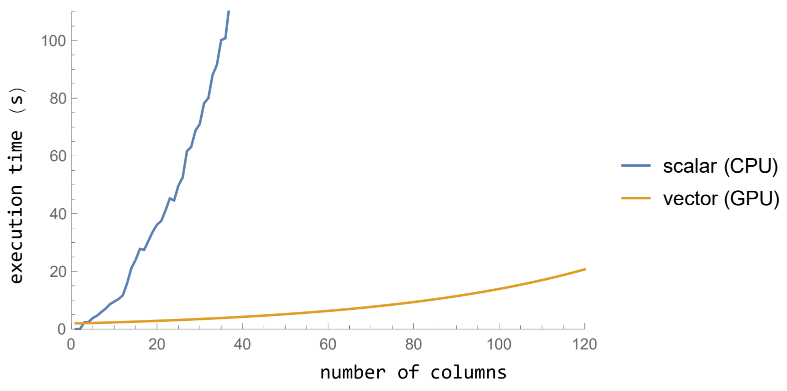


Fig. 4.6: Compute difference between scalar and vector calculations

4.5 Assessment and Limitations

Previous optimisation approaches, both algorithmic and engineering, promise runtime uplifts for the "Find-QID" problem. In the following section, an empirical study shall offer more in-depth insights into the practicability and scalability of those approaches.

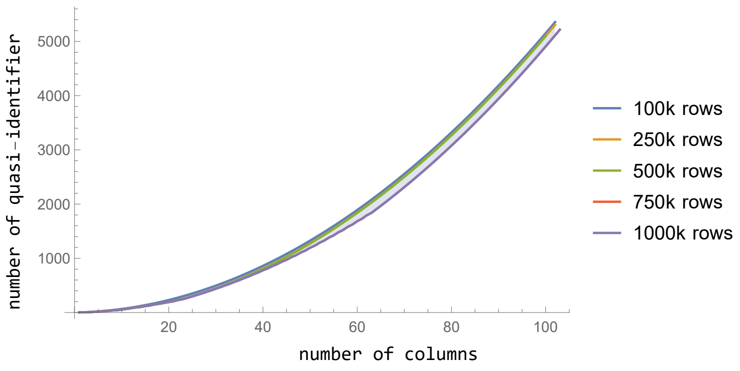
4.5.1 The Dataset

A semi-synthetic dataset was compiled to allow for a reproducible assessment and the disclosure of raw data samples for comparison. Various sources have been concatenated, including official government websites, public statistical data and datasets as part of previous publications to assemble a semi-synthetic health dataset. A complete list of all attributes is publicly available on github.com [136], while Appendix A depicts details about the used attributes and their data types. Naturally, this dataset includes various quasi-identifiers (QIDs), which amount increases over the number of available describing attributes (see Figure 4.7a).

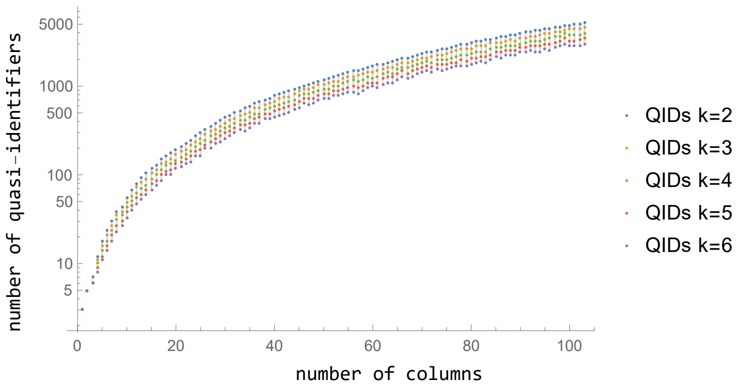
The variability of our k-anonymity requirement does not significantly alter the QID amount growth in their high-dimensions. Figure 4.7b illustrates this evolution.

4.5.2 Experimental Hardware

The experiments are conducted on a GPU-accelerated high-performance compute cluster, housing 160 CPU cores (E5-2698 v4), 760GB RAM, and 10x Tesla V100. Each Nvidia Tesla GPU is equipped with 5120 CUDA cores and offers a combined Tensor performance of 1120 TFlops. The execution environment for GPU-related experiments will be restricted to one dedicated CPU core and a



(a) QID growth over available rows



(b) QID growth over k-size

Fig. 4.7: Dataset characteristics for the experiment

single, dedicated Tesla V100 GPU. For CPU-related experiments, the compute cluster's runtime environment is restricted to 10x dedicated cores unless specified otherwise. Further hardware specifications will be available in Appendix B.

4.5.3 Performance Comparison of the Greedy Approach

As previously adumbrated, Braghin et al.'s work will serve as a baseline for the QID search scheme of our "Find-QID" problem [20]. In the first step, we will compare the pure original exact search schema from Subsection 4.1.2. It will not entirely surprise the reader that the latter depicts worse runtime than the Braghin et al. baseline as delineated in Figure 4.8. Also, utilising the minimal QID knowledge from Subsection 4.2.1 offers only smaller improvements. When using the QID tuple characteristics for greedy processing by rearranging the QID candidate queue by their likelihood of QID satisfaction as presented in Subsection 4.2.2, Figure 4.2 shows a promising runtime improvement. We acknowledge that worst-case runtime remains W[2]-complete, but the expected runtime decreases significantly, especially for multi-attribute settings.

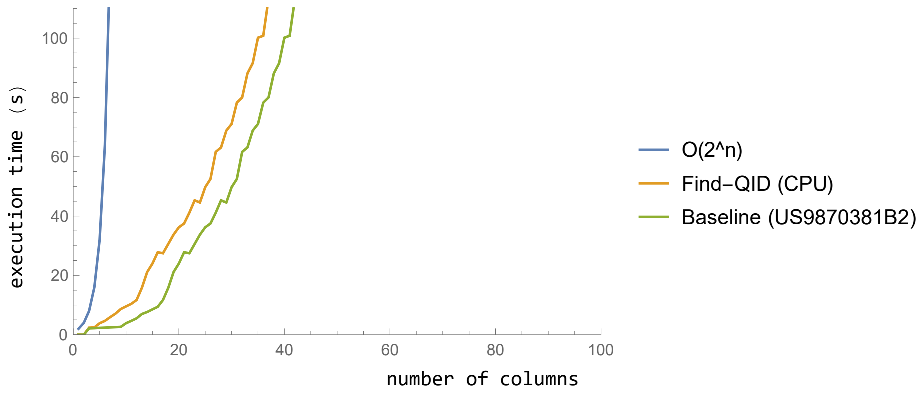


Fig. 4.8: Comparison Find-QID (CPU) vs baseline

4.5.4 Performance Comparison of the Parallelised Approach

To benchmark the efficiency uplift of different architectures, Figure 4.10 depicts the increase in execution time as the number of columns grow. The more describing attributes are available and require processing, the higher its execution time. Utilising parallelisation, the exponential nature of growth can be delayed in both architectures. By increasing available CPU cores, the runtime can be smoothed a bit (see Figure 4.9). However, because of the GPU's enormous parallelisation capabilities, the boost in runtime may be sustained even for large-scale applications (see Figure 4.10). Transfer to GPUs might be detrimental in some cases, especially in smaller dimensions, due to the initial overhead of memory shifting across the motherboard bus.

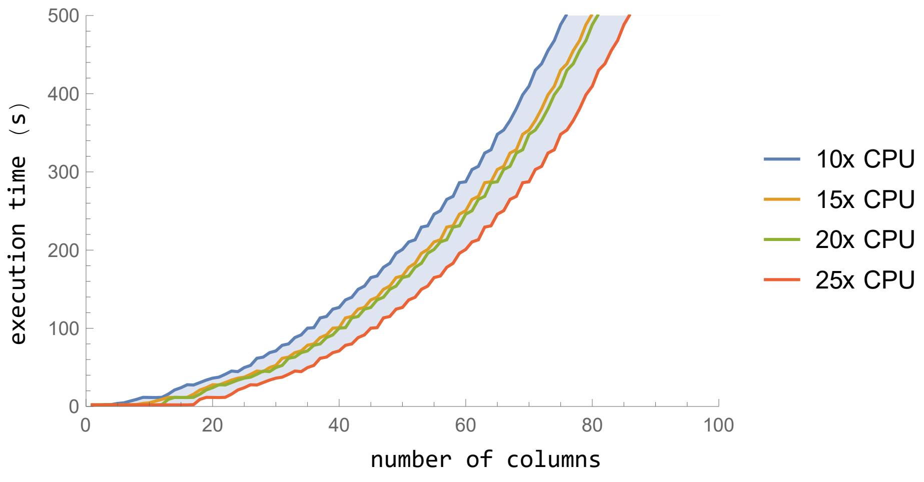


Fig. 4.9: Comparison of scaling CPU cores

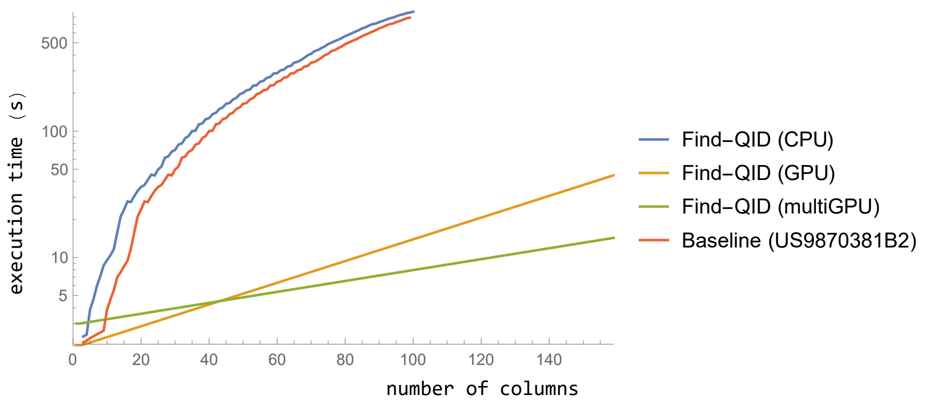


Fig. 4.10: Comparison of different QID search schema against baseline

Varying Number of Records. The size of the rows has been changed from 100k to 1M as part of the experiment. The runtime impact was negligible on both the CPU and GPU since the combinatorial complexity did not change (see Figure 4.7a). However, the necessary memory allocation was strongly impacted by the number of records, and GPU RAM is nowadays restricted to 16GB - 32GB without swapping to traditional main memory.

Varying Number of Attributes. Unlike the rows, the number of attributes had a significant impact on the runtime. As the number of attributes and columns increases, so do the combination possibilities and, as a result, the processed tuples.

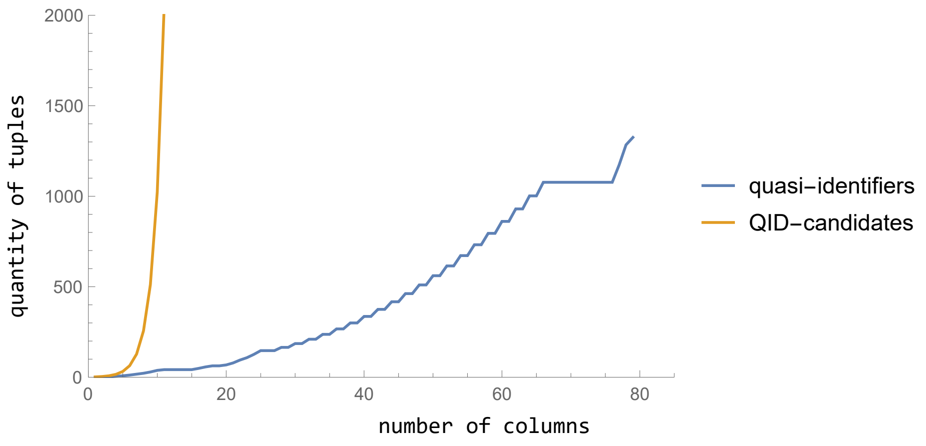


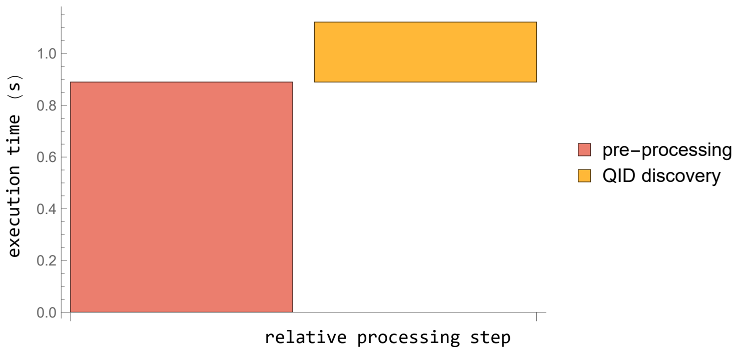
Fig. 4.11: GPU QIDs vs combinatoric growth

Figure 4.11 depicts the increase in the number of theoretical attribute combination candidates that required processing in comparison to the actual QIDs discovered. The exponential character of the growth can be noticed by the reader.

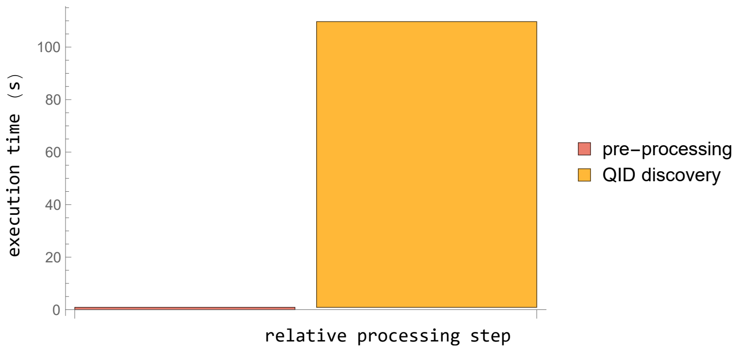
Varying Cluster Sizes. The equivalence class with a cluster size of k follows k-anonymity in the way that k is the least number of identical data records for the same attribute value. This means, when $k = 2$ applies, at least two distinct rows should have the same subset of attribute values. By adjusting this setting, the balance between information loss and equivalence class can be weighted. While the k serves as a privacy metric, the equivalence class simultaneously enforces the minimal information loss and therefore serves as a balance weight option. This phenomenon is depicted in Figure 4.7b. The larger the equivalence class k, the higher the obstacle for attribute combination to qualify as quasi-identifier as more rows are needed to qualify, and therefore the fewer QIDs in the same dataset.

GPU-Based Greedy QID Discovery. While the worst-case runtime remains the same, making wise choices on the starting points of a greedy QID discovery can have a significant impact. This holds as one can eliminate the search of a branch after the discovery of the minimal QID. When the first selection of a QID candidate is already the QID itself, no more processing of this branch is needed (see Section 3.2). To find the optimal entrance points for such greedy QID discovery, Figure 4.12 delineates the execution time and ratio of both *pre-processoring* and *QID discovery* activities against the increase in the number of columns. The pre-processing remains almost the same for increasing data dimensions, while the QID discovery grows with large numbers of attributes (100+) and rows (1M). This ratio illustrates the overhead in shifting the workload to GPU and outlines the surplus massive-parallelisation via GPU can contribute to the "Find-QID" problem for large-scale and high-dimensional datasets. Simultaneously, Figure 4.12 also delineates the time needed to validate $2^{100} = 1.2 * 10^{30}$ attribute combinations as QID candidates.

In sum, this section offered multiple options to realise the discovery of quasi-identifiers. Suggested optimisation techniques promise runtime performance uplifts from a conceptual perspective, and given previous experiments, we confirmed the same practically without the risk of compromising privacy guarantees, unlike probabilistic methods do. To better understand the risk of data inferences, the following Section 5 will address the triage of QID candidates in high-dimensional settings and their risk in compromising individuals.



(a) Execution time components for L=2



(b) Execution time components for L=200

Fig. 4.12: Breakdown of GPU pre-processing vs QID candidate discovery runtime

5

Enhanced Detection of Privacy Violating Inferences

Unique attribute value combinations proved to be a risk to the anonymity promise from Definition 4. In the shape of a quasi-identifier specified in Definition 5, those attribute combinations can be used to de-anonymise individuals and therefore need to be discovered and removed. Another perspective on privacy guarantees is inferences in datasets, which can also be used to re-identify individuals. Privacy violating inferences combined with auxiliary data are utilities for an attacker to draw conclusions and derive private information [184, 185].

In the following sections, we will demonstrate how to determine probabilistic linkages between attribute values in a use case agnostic method by using Bayesian inferences. Further, optimisation approaches will be presented, compared and empirically studied.

5.1 Bayesian Inferences to Determine Candidates

Bayesian networks are a type of probabilistic (directed acyclic) graphical model that is based on Bayes' theorem [57]. Commonly, Bayesian networks are used to develop models using existing data sources or domain experts, and they use directed acyclic graphs to describe a set of random variables, including their conditional dependencies.

The Bayes theorem can also be defined as the interdependency of two independent events A and X, where A happens knowing that X has already occurred. This is equivalent to $P(A|X)$, which is the conditional probability or dependency that A occurs when X has previously occurred. The graph is usually represented by an adjacency matrix or an adjacency list in most libraries. With a Boolean value in a two-dimensional array, adjacency matrices express whether a combination of two nodes is nearby in the graph, suggesting that there is a directed edge between those nodes (see Table 5.1).

From a theoretical standpoint, constructing a Bayesian network necessitates the storage of $O(n^2)$ edges and double values implementing the assigned probability with n answering to the number of nodes. In adjacency lists, on the contrary, a node requires n times the number of edges e ($n \cdot e$ instead of n^2) for the same representation of a Bayesian network. Thence, adjacency lists are more efficient in storage for sparse graphs than adjacency matrices (i.e. ones with few edges). Depending on the graph and operation characteristics, however, one or the other is more suitable. This can apply for instances insofar that adjacency lists are preferable for returning a node's neighbours, whereas adjacency matrices make testing if nodes are adjacent easier.

Neither Naive Bayes classifiers nor Bayesian neural networks are to be confused with Bayesian networks. A Naive Bayes method assumes a set of conditionally independent attributes, allowing Bayes' theorem to be applied to probabilities. In contrast, the same assumption does not apply to Bayesian networks, which need all dependencies to be modelled a priori. As a consequence, a Naive Bayes model is to be considered as a special case of a Bayesian network. Although nodes and edges in Bayesian networks appear to be similar to those in Bayesian neural networks, nodes and edges in Bayesian networks have intrinsic meaning. A node, for example, relates to a state, but an edge deduces a conditional probability. The same holds not always true for Bayesian neural networks.

In the context of Bayesian networks, the concept of Markov chains or networks is frequently employed, along with four properties indicating whether the state space and time are discrete or continuous. Markov networks are made up of undirected arcs that may or may not be cyclic. Yet, an acyclic directed arc constitutes a factorisation of some joint probability distribution in Bayesian networks. Often, the terminology of Markov chains or networks is used in the context of Bayesian networks combined with four characteristics corresponding to whether its state space and time are either discrete or continuous. While Markov networks are composed of undirected arcs, which may be cyclic, Bayesian networks have directed arcs that are acyclic, representing a factorisation of some joint probability distribution. Powell et al. offer an illustrative visualisation to support the differentiation of those concepts [149].

5.1.1 The Concept

Using Bayesian networks to conduct anonymisation promises a number of benefits. First, by augmenting the model with newly obtained datasets, continuous, partial, use case agnostic, and low-cost update of the underlying model is achievable. Furthermore, by evaluating the conditional probabilities within the network, data exposure problems can be discovered.

Various risks can be avoided by using the rounding error and intervening on attribute value inferences through conditional probability. This way, adjusting the conditional probability will exacerbate homogeneity- and background knowledge attacks and simultaneously scramble inferential attacks. Because only meta connections (probabilistic cases) are stored, rather than the actual and duplicated ones, improved scalability in terms of storing space can also be accomplished. This has the drawback of being computation-heavy and, therefore, time-consuming. The NP-hardness of the exact [28] or even approximate [34] inference learning may be a bottleneck because nodes in Bayesian networks generally represent one single state, and a high-dimensional data set with many attributes holding many possible states results in a massive increase of nodes.

We are used to looking at relational data and traditional table representations, yet the Bayesian network depicts occurrences of events. As a result, relational data must be understood as if it were an event. Rather than representing each data record in a common table representation, as illustrated in the prior example from Chapter 3 depicted in Table 3.1, we can construct the likelihood of each data attribute value in relation to the others.

This can be expressed as a network, with each node representing a potential attribute value and the edges typify the probability of occurring in relation to the connected nodes. A network like this is shown in Figure 5.1, which was constructed from the preceding example Table 3.1.

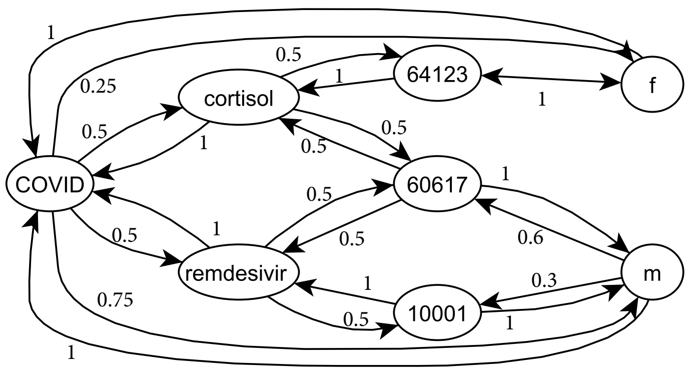


Fig. 5.1: Net representation of relational data with weights

Two nodes are drawn with edges to surrounding attribute values for both genders from the same table, similar to their row-based relationship (ZIP, drug, disease). Non-existing attribute value combinations with a conditional probability of 0, such as $female$ and 10001, are not associated with a network edge for the sake of simplicity.

An adjacency matrix, as shown in Table 5.1, can be produced in accordance with the network topology, reflecting all edges between node pairs and their conditional probability. The technical representation of the stated network with all node combinations of size 2 is an adjacency matrix like this. Data exposure can now be identified using conditional probability or, more precisely, attribute value inferences. Only half of the matrix must be filled for bidirectional edges. However, directed edges, such as the ones in our example, necessitate the use of the entire matrix.

Table 5.1: Adjacency matrix representation

	m	**f**	**10001**	**64123**	**60617**	**Cortisol**	**Remdesivir**	**COVID**
m	1	0	0.3	0	0.6	0.3	0.6	1
f	0	1	0	1	0	1	0	1
10001	1	0	1	0	0	0	1	1
64123	0	1	0	1	0	1	0	1
60617	1	0	0	0	1	0.5	0.5	1
Cortisol	0.5	0.5	0	0.5	0.5	1	0	1
Remdesivir	1	0	0.5	0	0.5	0	1	1
COVID	0.75	0.25	0.25	0.25	0.5	0.5	0.5	1

5.1.2 Implementation Details

It is possible to create the underlying adjacency matrix in either an accurate or an approximate way. Tsamardinos et al. introduced the approximation approach "hill-climbing" to efficiently construct a structure learning algorithm [173], despite the fact that precise Bayesian model learning is NP-hard. Because approximate approaches have been thoroughly investigated, the focus in the following will be on optimising exact learning techniques. Only an exact approach can provide sufficient anonymity. The most efficient and precise adjacency matrix generation we've found so far is depicted following.

The Algorithm 5 computes all column permutations with a size of $n = 2$ tuples:

$$P(n, r) = \frac{n!}{(n-r)!} \tag{5.1}$$

as seen in line 5. A native method for generating permutations with a specific length is available in most computer languages. This collection of column permutations can be examined in parallel and chunked to reduce inefficient process spawning and process management overhead (see line 7). The parallel execution results are first collected as a list, then concatenated as bulk to reduce overhead once again (see lines 8 and 10). The Algorithm 6 specifies how each adjacency matrix element is processed. Through a group-by statement, all current attribute value combinations for each attribute permutation from the original dataset are determined, such as $female$ for gender and 64123 for ZIP (line 4).

The actual probabilistic calculation is as simple as counting the number of times an attribute appears in relation to the total number of attribute values (column size) in the evaluated column tuple (line 7). As a result, this gives us the conditional probability for each attribute value tuple in the adjacency matrix we're building.

Next, both axes are being pre-populated with all relevant attribute values. The matching conditional probability that has been determined earlier is being injected at the intersection to create a matrix. The column tuple describes each of these probabilities as an edge value, with the first element marking the edge's beginning and the latter marking the edge's end. For assembling an equivalent matrix, we pre-populate both axes with all describing attribute values and inject at the intersection the corresponding conditional probability we have calculated earlier. Each of these probabilities serves as an edge value directed by the column tuple setting, where the first element marks the edge's start and the latter marks the edge's end (see Table 5.1).

The exact determination of conditional probabilities is demonstrated in the following example:

Example 3. Deriving from Table 3.1, 12 attribute permutations exist that need to be analysed: (ZIP, drug), (gender, ZIP), (gender, drug), (drug, disease), (ZIP, disease), (gender, disease) and their inverse tuples. For each of these tuples, the likelihood of each value combination occurrences is being calculated: $(M, 60617) = 0.66$, $(M, 10001) = 0.33$, $(F, 64123) = 1$. To put it another way, how likely is it that the ZIP code will be 60617 if the gender is M?

When the original dataset is refurbished in such a way that one gains the possibility to quickly generate and persist adjacency matrices like the sample one in Table 5.1. These matrices, in turn, serve as a model for Bayesian networks (see Figure 5.1). In the following, we will elaborate on exact optimisation approaches.

Algorithm 5: Calculate probabilistic linkages

```
1  Calculate probabilistic linkages ($table, settings$);
   Input  : Table $table$ containing the dataset
            Object $settings$ with setting values
   Output: Array $result$ including all attribute value tuples and their
            probability
2  result = initialise a table with columns ["value1","value2","likelihood"]
3  partialTables = initialise an empty list
4  // Prepare the dataset for parallelise its execution;
5  columnCombination = create a list of permutations of table's columns with
    length of 2
6  create a process pool:
7      split columnCombination in chunks and execute
        build_probability_for_column for each chunk in the process pool
8          append the outcome from each process to the partialTables
9  close the process pool
10 result = concatenate the partialTables
```

Algorithm 6: Calculating probabilities for attribute value tuples

```
1  calculating the probability for a column ($columnTuple, table$);
   Input  : Array $columnTuple$ as list of two column names,
            Table $table$ containing the dataset
   Output: Array $result$ including all attribute value tuples and their
            probability
2  // Transform table to adjacency format through GROUP BY method;
3  group = initialise a table with columns ["value1","value2","likelihood"]
4  group = group table by columnTuple and count group appearance
5  columnSize = count the total column length
6  // Calculate their conditional probability;
7  group['likelihood'] = divide each group size by columnSize
8  return group
```

5.1.3 Identification of Data Exposure & Ensuring Anonymity

The inferences stated in an adjacency matrix characterise the probabilistic linkage of attribute representatives across the dataset. A Bayesian network can use such adjacency as an underlying model. Aside from learning and sampling from this model, the same inferences can be used to identify data exposure risks.

Utilising these inferences for privacy risk triage, further metrics are needed. One can be the *summed cycle inference* (scf), which is constructed by iterating over the created adjacency matrix and summing up all inferences for each row (cycle). Therefore, we formally define:

$$scf = \sum_{i=1}^{m} cf_i \tag{5.2}$$

where m answers to the number of columns (length of the adjacency matrix) and t to the qualifying non-null value amount of the selective row. Similarly, the *mean cycle inference* (mcf) describes row wise:

$$mcf = \frac{\sum_{i=1}^{m} cf_i}{t} \tag{5.3}$$

Both measures provide information on the probability of the respective attribute combinations occurrences. **The higher the summed and mean cycle inference, the greater the risk of data exposure**, one can deduce. As an example, one can consider the following:

Example 4. The adjacency matrix in Table 5.1 addresses high inferences between $female \rightarrow$ 64123, $female \rightarrow$ *cortisol* and $female \rightarrow COVID.$ based on the data in Table 3.1. This conclusion is marked by the conditional probability of 1, which means that in 100% of the instances, the attribute values are consistent. The weighted summed and mean inference is, in essence, the highest in the matrix. When comparing such an injection function to the original example, the reader will immediately recognise the row's uniqueness.

The mean of the accumulated cycle inference and the associated deviation as a threshold for detecting data exposure in an adjacency matrix proved to be effective. Experiments in the following Section 5.4 will highlight: The greater the threshold, the more likely we are to disregard privacy risks. As a result, the lower the threshold, the more records fall into the analysis, which will increase the runtime and potentially information loss.

The selection of such a threshold, on the other hand, can be done greedily. As more combinations become possible, the metric values naturally drop as the dimensions increase. Figure 5.2 depicts the declining cycle inference trend, with gaps in the curve caused by standalone identifiers characterised by their high cardinality. In the following Section 5.4, similar patterns will be explored further as part of the experiments.

After successfully identifying a risky inference, one can adjust the appropriate conditional probability in a way that the mean summed cycle inference approaches average values again, effectively eliminating any data exposure.

Definition 9. *Anonymity Assurances*
Given an involved column c_i, let $t = \{c_1, ..., c_n\}$ describe a cycle. To quantify

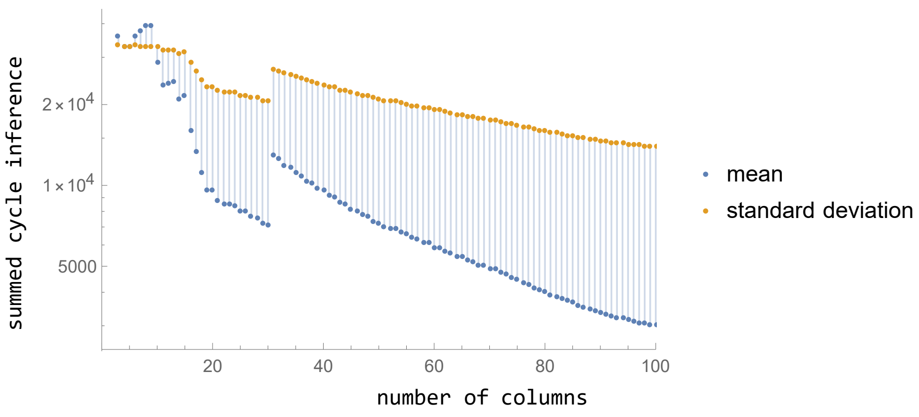


Fig. 5.2: Evolution of summed cycle inference over increasing data attributes

the data exposure risk of any cycle t, let $\mu(t)$ be a suitable measure. Further, the set of cycles with length n containing the column c is denoted by $T_n(c)$. Any cycle t that deviates strongly from the statistics of $T_n(c)$, $c \in t$ is likely to be a quasi-identifier and thus poses a potential anonymity risk. Therefore applies:

$$|E\,[\mu(T_n(c))] - \mu(t)| \leq \theta\,\sigma(\mu(T_n(c))), \tag{5.4}$$

with θ answering to a threshold parameter, E the expected value and σ the standard deviation.

Anonymity is ensured only when no unique column combinations remain, and hence no individual record can be traced back to its original data owner.

Following data profiling research, a unique column (combination) that fulfils the requirements of a maximal partial (minimal) unique column combinations (mpmUCC) can be determined to encompass all quasi-identifiers [138]. This holds true for quasi-identifiers of every length and combination in a data set without the need of separating sensitive and non-sensitive data. Next, the collection of high-risk cycle inferences can be compared against the mpmUCCs to see whether there is a delta in the overlap. To counter the risk of data exposure, the candidates shall be altered to aggravate the re-identification of the data owner. The punctual deviation of selected conditional probabilities, as well as the overall summed cycle inference, may be used to achieve the necessary aggravation. Because this adjustment is a simple subtraction procedure, its algorithmic realisation should be easy to implement. Finally, there is no way to recreate any linkable unique column combinations using a Bayes model. The outcome may be called anonymous if it is not possible to re-identify individuals using the composed data.

Example 5. In the previous example of the adjacency matrix in Table 5.1, high inferences between $female \rightarrow 64123$, $female \rightarrow cortisol$ and $female \rightarrow COVID$ are identified. A quick sight on the original Table 3.1 confirms the unique column combination. Data exposure risks can be aggravated by lowering the relevant inferences towards the mean cycle inference in a greedy manner. When converging to the average conditional probability, the sample group of, for instance, $female \rightarrow 64123$ increases and consequently the risk of exposure.

5.1.4 Projection on Data Streams

The demand for data anonymisation technologies in data streams has increased in recent years. Data inputs from various sensors are increasingly being used to track the number of characteristics. Belt speed, acceleration, and rotations are examples of these in manufacturing. Sensors have also been used in healthcare to measure patients' vital signs such as heart rate and oxygen saturation. Most notably, fitness sensors as consumer hardware have become practical for regular usage, resulting in large data flows, including often highly sensitive information (movement patterns, health or vital status).

While structured data had to be handled in the past to give privacy-aware replicas of a static data repository, new domains such as mobile health apps now necessitate proper processing on continuous data flows. However, there is a considerable difference between static and stream content, which has implications for the anonymisation process. With a static environment, the dataset may be easily viewed and analysed as a comprehensive snapshot of as it is, however in stream processing, only a small sample of the entire stream is available and is being processed at the same time. This snapshot could be a single data record or a (micro-) batch of data.

To avoid the linking of distributed quasi-identifiers among distinct separated yet related events, the previous data stream events must be considered when anonymising this standalone sample. Extending the previous example from Table 3.1, this causality may be demonstrated:

Example 6. The reader is referred to the previous data from Table 3.1 with four initial patient records, including ZIP code, patient gender, disease diagnosis and medication prescription. With the addition of two more records, {F, 10002, hydroxychloroquine, COVID} and {F, 10001, remdesivir, COVID}, it is difficult to assess their private data exposure risk without the previous context. With knowledge of the full existing Table 5.2, it is evident that the second record stands out due to its new ZIP code and a new medicine prescription. On the contrary, whereas the first added row blends in from an anonymisation perspective.

Table 5.2: Extended sample health dataset through stream data snapshot

Gender	**ZIP**	**Drug**	**Disease**
M	10001	remdesivir	COVID
F	64123	cortisol	COVID
M	60617	cortisol	COVID
M	60617	remdesivir	COVID
F	10001	remdesivir	COVID
F	10002	hydroxychloroquine	COVID

When assessing privacy risks, the transparency of having references is crucial. We acknowledge that re-evaluating the entire table on a wide scale with state-of-the-art for each snapshot is impossible. The reader may observe a balance between batching incoming information and updating the reference table against which to compare. The larger a batch, the less incremental and iterative data stream processing is required to be. A huge batch, on the other hand, signifies a high level of information loss due to the lack of granularity. One of the experimental objectives for the empirical study in Section 5.4 will be to achieve this equilibrium. An increase in the processing speed of typical syntactic data anonymisation algorithms is required to achieve near real-time anonymisation of a continuous data stream. Solving this computational problem also eliminates existing anonymisation limitations in data stream processing.

There are now two possibilities for practically discovering quasi-identifier using a Bayes theorem. The first is to use a greedy mathematical solver, and the second is to rely on massive-parallelisation. Both will be presented in the following, discussed, and finally empirical studied.

5.2 Optimisation Methods: Vertex (Node) Aggregation

The concept of utilising Bayesian networks to discover inferences, potentially exposing private data, is promising. Yet, simultaneously it suffers NP-hardness of the exact [28], and even approximate [34] inference calculation as previously outlined. Since our motivation is to process large and high-dimensional datasets, we looked into reducing the given complexity.

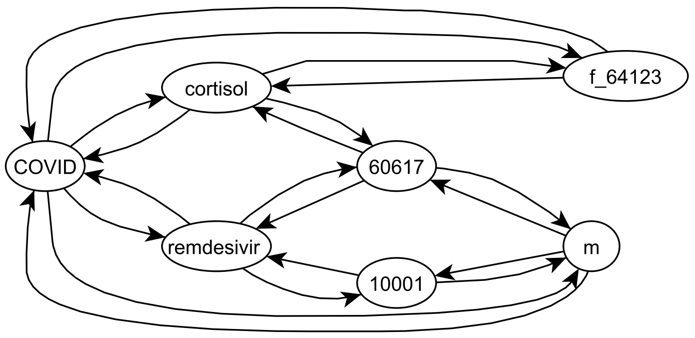
Multigrid solver, which incarnates the concept of coarsening the state space, may be transferable to this problem space. The problem space of complexity Bayesian network calculations can be addressed with a multigrid solution, which captures the concept of coarsening the growing state space. Multigrid methods are used in the field of numerical analysis and answer to a family of effective algorithms for approximating answers to equation systems formed from partial differential equations discretisation [21, 163]. In the case of a symmetric adjacency matrix using unidirectional arcs instead of directed arcs, multigrid can be applied to address the vertex growth.

For this purpose, our multigrid solver approach operates in an exact manner by aggregating nodes of the Bayesian network representation that are in all probability. The conditional probability for each edge, or the inferences in the adjacency matrix, determines this probability. The aggregate is solely dependent on the conditional probability along the edges that emerge across all attribute values, and it only reflects the nodes of disjunctive attribute subsets. A new node is formed as a result of this aggregation, which replaces existing nodes with new edges.

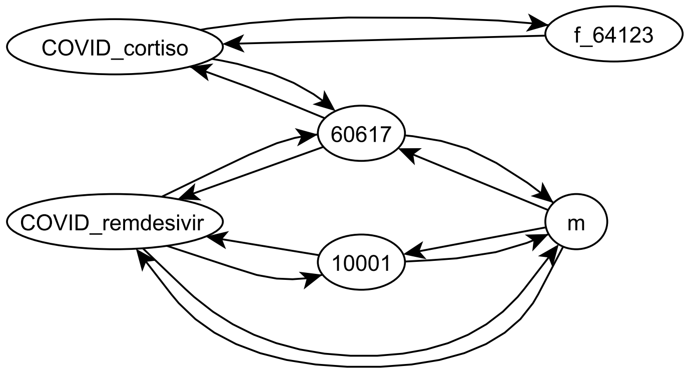
When being in all probability, the vertex of an exact training approach can be aggregated to accelerate computation time. This can be done by defining a threshold for the likelihood of appearances. Simultaneously it implies a higher information loss given the shift from exact determination towards approximated probabilistic linkages. Our first experiments encourage a threshold of >95% for the conditional probability (see Section 5.4). To ensure the anonymity requirement (see Definition 4), such a threshold should be determined in a greedy manner. Decreasing the threshold significantly, like in the case of 95%, increases the number of nodes under aggregation, but it may result in the loss of 5To achieve an equilibrated result between computation time and information loss, one can greedily adjust the threshold as the characteristics of the dataset may alter.

In this work, however, the focus shall remain on leveraging the exact multigrid solver approach. The strategy to aggregating nodes is illustrated in the following example:

Example 7. Continuing with the previous example from Figure 3.1, in case of $f \rightarrow 12160$ being in all probability, $f\_\_12160$ is derived as a new state. At the same time, the original states and transitions are being conflated. This transformation is being depicted in Figure 5.3a where the two nodes f and 12160 are condensed. In analogue $Ibuprofin \rightarrow Flu$ being in all probability, the new node $Ibuprofin\_Flu$ is being added (see Figure 5.3b).



(a) Combining exact dependencies: f + 12160



(b) Combining high probabilistic dependencies: drug + disease

Fig. 5.3: Complex reduction through manifolds and multigrid

The exploding complexity of large, high-dimensional datasets can be captured in this way. Because fewer node combinations and, thus, edges must be evaluated, the runtime for sampling and analysing drops dramatically as the number of nodes is reduced in exponential space. First experiments have demonstrated promising results.

As a reference and to understand the potential of the multigrid approaches, the reader is kindly referred to the security research field. In this case, multigrid techniques offer a solution to the Poisson-Equation for elliptic curves in $O(n)$ rather than $O(n^{2.5})$ [68, 177].

Manifold learning, in addition to multigrid, can be leveraged to aggregate those Bayesian network nodes. Manifolds as a topological space have their origins in mathematical physics, where they have been used to decompose complex geometric patterns into simpler topological properties by approximating Euclidean space around the observation point.

Likewise, the number of nodes can be efficiently aggregated to reduce processing complexity by treating the dimensionality reduction problem as a classical problem in Riemannian geometry [25, 102]. Figure 5.3 illustrates the runtime gains. Several optimisations, such as Wang et al.'s adaptive manifold learning [180], are employed in various geometric research domains and could be used as a foundation for future state aggregation in Bayesian networks.

The algebraic multigrid (AMG) solver library, for instance, was created by Luke Olson and Jacob Schroder [127], and it implements a "multilevel approach for solving large-scale linear problems with optimal or near-optimal efficiency."

AMG, unlike geometric multigrid, requires little to no geometric knowledge of the underlying problem and generates a coarser grids sequence directly from the input matrix. For problems discretised on unstructured meshes and irregular grids, this aspect is important.

Those algebraic solvers serve as a basis to combine and subsume nodes and corresponding edges within our network to reduce its complexity, especially during sampling.

Algebraic solver to aggregate vertex promise to reduce nodes, runtime and therefore complexity, as we have learned. As another optimisation methodology, parallelisation offers various improvements in other data processing fields. Hence, we will outline an algorithmic breakdown and implementation of massive parallelisation through GPU accelerated Bayesian inference detection in the following.

5.3 Optimisation Methods: Vectorised GPU Acceleration

Parallelising the execution of combinatorial problems is a different way of approaching the problem. The quantity of CPU cores usable in the processing environment has effectively limited parallelisation so far. With hyper-threading enabled, multiple simultaneous computing threads can be processed. Yet, their performance is still limited to the CPU capacity. Nayahi et al. examined scaling-out approaches using a horizontal scaling technique following the MapReduce concept of dividing and conquering huge process workloads across many servers [42, 120]. However, when more processing nodes are added, the network I/O and infrastructure needs increase, which in turn limits the system's realistic scalability.

5.3.1 Prerequisite for Massive Parallelisation

A similar concept can be employed here as well, based on recent improvements in GPU acceleration. Graphics processing units (GPUs) have traditionally offered high compute density, high computations per memory access, deep pipelines, fast throughput, and low latency tolerance. However, because they lack a huge L1-L3 cache like CPU design, their memory capacity is still restricted. CPU cores are, in general, faster and smarter than individual GPU core instruction sets, but the sheer quantity of GPU cores and the tremendous amount of parallelism they provide more than compensate for the single-core clock speed disparity and limited instruction sets (see Illustration 5.4).

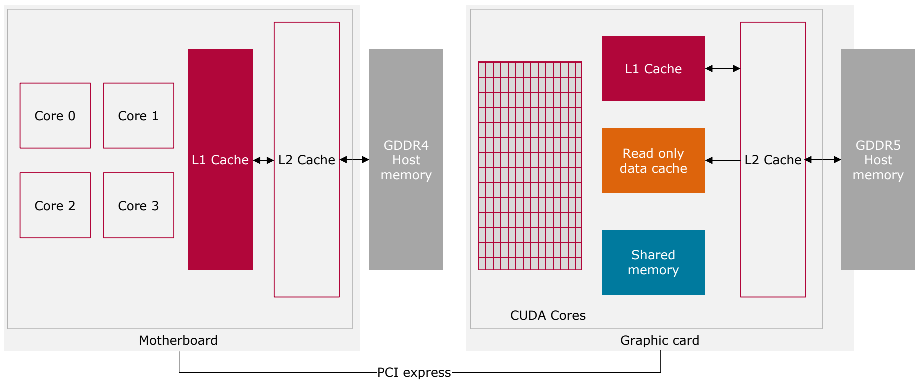


Fig. 5.4: Motherboard architecture with periphery GPU hardware

Established libraries and algorithms are not immediately transferable to run on GPUs as they require a few additional processing steps. First, the designated dataset needs to be (pre-)fetched and preferable converted in a unified memory format like Apache Arrow. This step enables unified memory management between CPU L1-L3 and GPU memory to reduce data transformation efforts when shifting data between the different hardware components. Additionally, by applying some dictionary encoding or similar encoding of attribute values to a numeric representation, the dataset is reduced in size and becomes handy. As the final step, the attribute value combinations need to be generated and composed in a vectorised format to unfold the full processing capacity of GPUs.

After completing these steps, the pre-processed dataset is available for GPU acceleration, and existing CUDA frameworks can be leveraged to execute the vectorised computation of the probabilistic appearances massively. The following Section 5.3.2 will deep dive on the exact implementation steps.

Two choices remain for the discovery of quasi-identifier and data inferences using the Bayesian network model. The first one is to use some mathematical solver following a greedy methodology. The second method is to utilise massive-parallelisation. While the former has been discussed in depth in [139, 141], we will focus on the latter of massive-parallelisation in the next sections and provide thorough benchmarking results, runtime, and resource allocation.

5.3.2 Vectorised QID Search Scheme

Most of the current implementations of computing Bayesian inferences, as outlined by Podlesny et al. [139, 141], is currently purely CPU based and therefore scalar oriented. On the CPU instruction set, single operations are then translated and executed. As outlined earlier, the GPU instruction set is more limited and optimised for vector-based calculations. Hence, to fully exploit GPU capacities, this paradigm needs to be algorithmically adjusted.

To take use of this computing opportunity, the difficulty is to convert workload from a scalar to a vector-based calculation. The objective at hand in Bayesian inferences is to compute the likelihood of occurrences as an interdependency between events A and X, where A occurs knowing that X has already occurred: $P(A|X)$. These events, as detailed in Section 5.1, technically respond to each attribute value in the chosen dataset. One counts how many times A and X appear together in a record over the total appearances of A to generate $P(AvertX)$, the inter-dependency. Simple mathematical functions such as counting and grouping have previously been vectorised and made accessible in open source libraries such as RAPIDS'[3] cuDF[4] as a NVIDIA initiative. CuDF provides an interface similar to the known python pandas[5], but it already runs on vectorised CUDA[6]. CUDA is a parallel computing API for Nvidia GPU devices that includes features such as scattered reads, shared and unified memory management, and integer and bit-wise operations that are fully supported. CuDF, on top of these functionalities, provides a well-known interface, which we use for Algorithm 7 to generate Bayesian inferences. Pre-fetching the original dataset, converting it to Apache Arrow format for unified memory management, and creating attribute value combinations to assemble the vectors were all covered in Section 5.2. In the pseudo-code, the same process is reflected (see Algorithm 7).

[3] https://rapids.ai/
[4] https://github.com/rapidsai/cudf
[5] https://pandas.pydata.org/
[6] https://developer.nvidia.com/cuda-zone

Algorithm 7: Vectorised computation of Bayesian inferences

```
1  Calculate probabilistic linkages ($self, table$);
   Input  : Object $self$ including the CUDA runtime environment
            Table $table$ as Apache Arrow table containing the dataset
   Output: Array $result$ including all attribute value tuples and their
            probability
2  // Prepare the data table;
3  columnNames = extract attribute names as columns from $table$
4  candidates = generate all permutations of $columnNames$ as set
5  tableEncoded = apply dictionary encoding of the $table$ to replace varchar
     with integer
6  result = initialise result a table with columns
     ["fromValue","toValue","likelihood"]
7  // Parallelise execution;
8  for $candidate$ in $candidates$:
9      fromAttribute = get first set element of $candidate$
10     toAttribute = get last set element of $candidate$
11     countVector = group $table$ by candidate set and count attribute value
         pairs
12     weightedCountVector = divide attribute value occurrences in
         $countVector$ by their individual occurrences
13     append $results$ with $weightedCountVector$ where the $fromAttribute$
         value answers to the $fromValue$ and $toAttribute$ value to $toValue$
         and its weightedCount as $likelihood$
14 return result
```

The group by and count operations are disassembled and delivered to each GPU core and GPU RAM as device instructions and variables (see CUDA docs[7]).

```
cudaMemcpy(destination, source, bitwidth,
    cudaMemcpyHostToDevice);
```

Each CUDA core calculates the algorithms on its designated data chunk on shared memory, then returns the computed results to the CPU and main memory for assembly (see CUDA docs[8]).

```
cudaMemcpy(destination, source, bitwidth,
    cudaMemcpyDeviceToHost);
```

[7] https://docs.nvidia.com/cuda/cuda-runtime-api/group__CUDART__MEMORY.html

[8] https://docs.nvidia.com/cuda/cuda-runtime-api/group__CUDART__TYPES.html

As a result, the vector computation can be executed with 5120 operations on a single Tesla V100 or 51200 operations for 10x Tesla V100s at the same time.

To ensure that the algorithm is sound, we compare the expected number of QIDs with the reference implementation on CPUs, as well as all CPU-based detected QIDs with the GPU implementation. Given that this is not a probabilistic implementation, it's no surprise that both sets of results are identical. It is noteworthy to grant the fact that the optimisation approach to aggregate the state-space through multigrid and manifolds outlined in Section 5.2 generates more false positives and may result in a somewhat higher information loss in exchange for reducing runtime complexity. An empirical study will examine both runtime uplifts and type I / II errors in greater depth in the following Section.

5.4 Empirical Analysis and Discussion

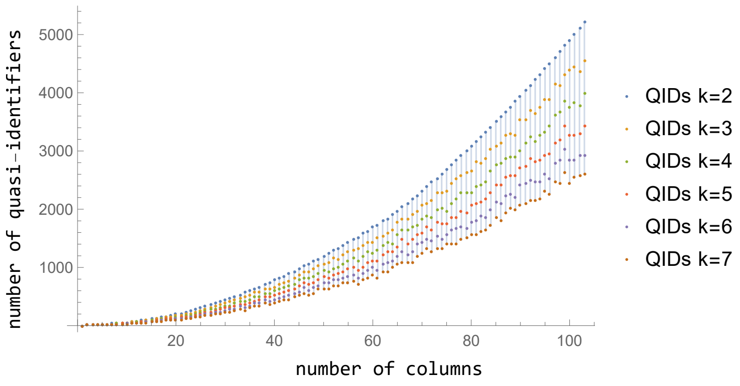
To prevent private data exposure, the reliable discovery of quasi-identifiers or any other unique attribute value combination is essential to initiate countermeasures. The same applies to any venue of data inference that can be used to uniquely identify data records. Yet, building Bayesian networks on high-dimensional data quickly results in a state-space explosion. Both algebraic and engineering optimising promise astonishing improvement to counter the exponential compute growth during the search of data inferences. To quantify these contributions, we will empirically study their effects in the following.

Hardware. Our examination runs on a GPU-accelerated high-performance compute cluster, housing 160 CPU cores (E5-2698 v4), 760GB RAM, and 10x Tesla V100 with 5120 CUDA cores each and a combined Tensor performance of 1120 TFlops. The execution environment for GPU related experiments will be restricted to one dedicated CPU core and a single, dedicated Tesla V100 GPU[9]. For CPU related experiments, the runtime environment on the compute cluster is restricted to 10x dedicated cores.

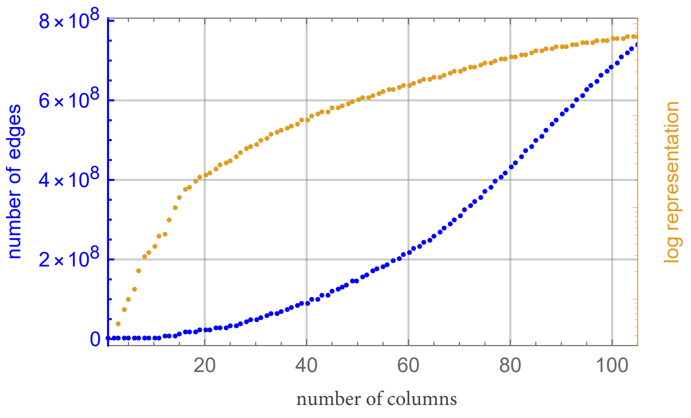
Data sets. To allow a reproducible assessment and the disclosure of raw data samples for comparison, a semi-synthetic dataset was compiled. Similar to the evaluation dataset in Chapter 4, various sources have been concatenated, including official government websites, public statistical data and datasets as part of previous publications to assemble a semi-synthetic health dataset. A complete list of all attributes is publicly available on Github [136]. Naturally, this dataset includes data inferences and various quasi-identifiers (QIDs). Correspondingly,

[9] `https://images.nvidia.com/content/technologies/volta/pdf/volta-v100-datasheet-update-us-1165301-r5.pdf`

the QID amount increases over the number of available describing attributes (see Figure 5.9a), as well as the number of edges (see Figure 5.5b).



(a) QID growth over k



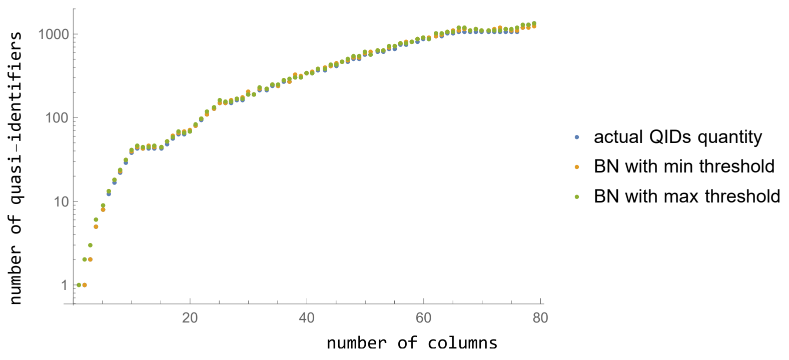
(b) edges over ncols

Fig. 5.5: Data characteristics for Bayesian experiments

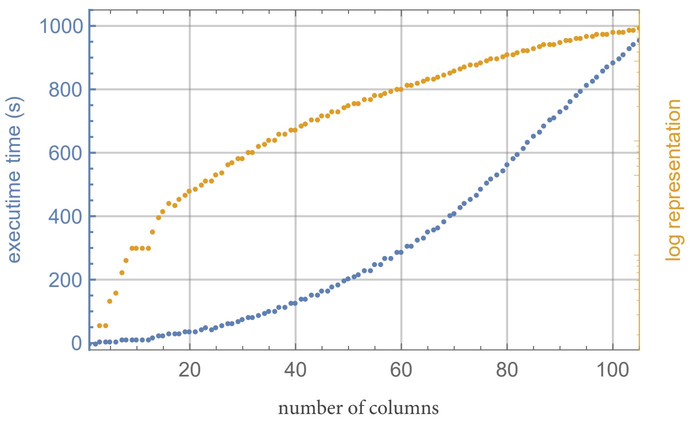
5.4.1 Inference Detection in Bayesian Networks

To preserve privacy, all inferences and, therefore, quasi-identifiers must be discovered. As a baseline, we employ the "Find-QID" search scheme from Chapter 4 to determine all available quasi-identifiers as reference. Regardless of applied optimisation, Figure 5.6a delineates the precision of the inference detection based on the cycle metrics introduced in Section 5.1.3.

Different thresholds in the summed or mean cycle inference metric minimise compute efforts to determine QIDs. The closer the green and orange dots are



(a) Precision eval quasi identifier over columns



(b) Runtime as time over the number of columns

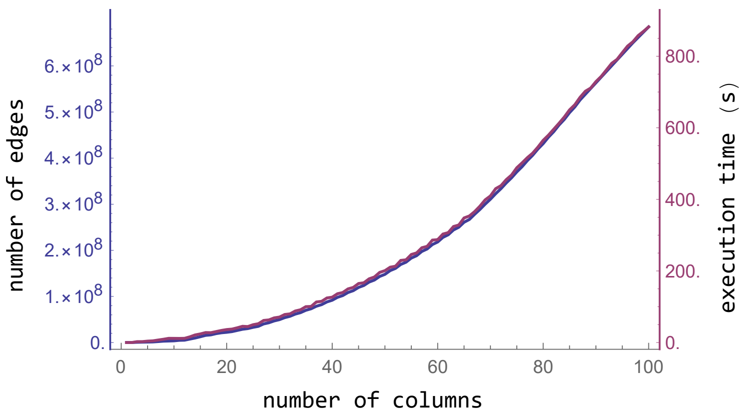
Fig. 5.6: Bayesian experiments results on precision and runtime

to the blue ones representing the actual number of QIDs, the less overhead in additional compute to be completed. In every instance, all QIDs have been determined with the 100% true positive rate and 0% false negative. Yet, false positives can differ based on the applied threshold, which again highly depends on the underlying data characteristics.

Figure 5.6b depicts the runtime without any optimisation applied, which supports the need for runtime improvements, especially for high-dimensional datasets. In the following empirical study, the surplus through optimisation approaches of mathematical nature and GPU acceleration will be quantitatively assessed to verify their performance gain promises of the underlying enumeration complexity.

5.4.2 Performance Comparison of Vertex Aggregations

The pure and original creation of a Bayesian network and its inference discovery, particularly in high-dimensional datasets, is extremely challenging and does not scale well without further action. Aggregating vertex through combining nodes as attribute values being in all probability offers a mathematical improvement. Multigrid solver has been well explored, but as their usage for state-space generalisation in Bayesian networks is light, we will outline their improvements.



(a) Number of columns over time edges

Fig. 5.7: Statistical evolution of Bayesian's vertex aggregation

Figure 5.7a delineates the increasing runtime over growing columns in the dataset. Once plotted against the number of edges, once against the permutation under assessment for inferences. It becomes evident that the runtime is directly linked to edges and permutations.

Aggregating states reduces the complexity of a Bayesian network, and therefore, the compute needed to create or sample from it. Figure 5.8 compares the original, traditional network against the optimised one with aggregated vertex. Especially for larger n of describing attributes, an improved runtime of 20% for $n = 100$ can be observed.

5.4.3 Performance Comparison of the Parallelised Approach

The most prominent performance difference between running the native scalar-based Bayesian inference implementation, the vector-based GPU acceleration and the state aggregation optimisation is the uplift in execution time. Figure 5.9b delineates again the runtime in seconds over the increasing number of attributes. This time including massive-parallelisation through vectorised GPU

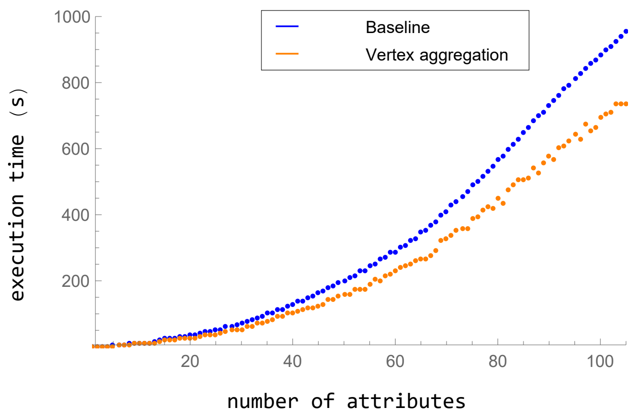
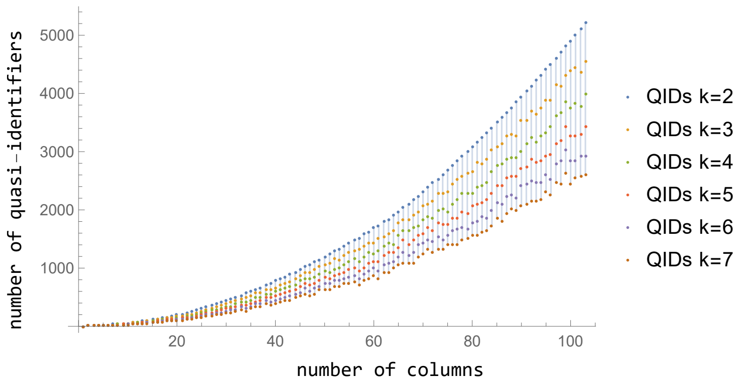
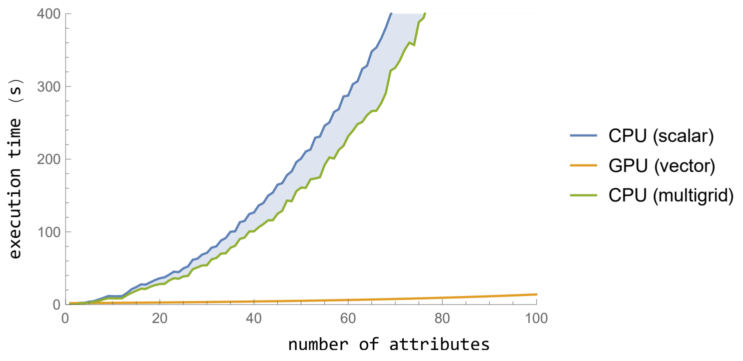


Fig. 5.8: Comparison of runtime growth over increasing columns

compute. Multigrid reduces the state-space explosion as part of the Bayesian network creation through aggregations by some factors. Yet, a vector-based implementation outperforms both previous solutions by magnitudes.



(a) Increase of QIDs split by size k of k-anonymity requirement



(b) Time complexity by optimisation approach

Fig. 5.9: Runtime comparison for parallelised Bayesian inference calculations

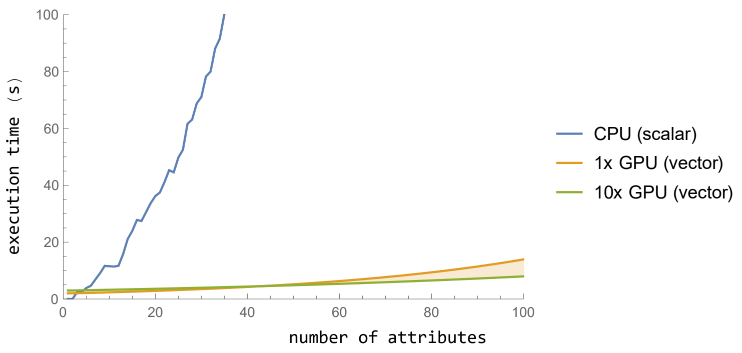
Varying Number of Records. The dimension of records (rows) does not really significantly impact the performance of already large datasets. This is not the case with smaller datasets. Additional rows usually represent alternative compositions of the same attribute values or increase their tuple count rather than introducing new attribute values. As a result, the binomial coefficient and its temporal complexity stay the same. This is expected to be different for much smaller datasets.

Varying Number of Attributes. As the number of columns grows, so does the number of descriptive attributes and the number of attribute combinations that can be used as QID candidates. Consequently, with an exponentially growing number of candidates, the processing time increase as well. This is shown in Figure 5.9b. This trajectory has been slowed by the degree of massive-parallelisation in the GPU implementation. For very large n currently out-of-scope, however, we do expect an increase as well. With any dataset, both semi-synthetic and real-world, we have not reached this theoretical point yet.

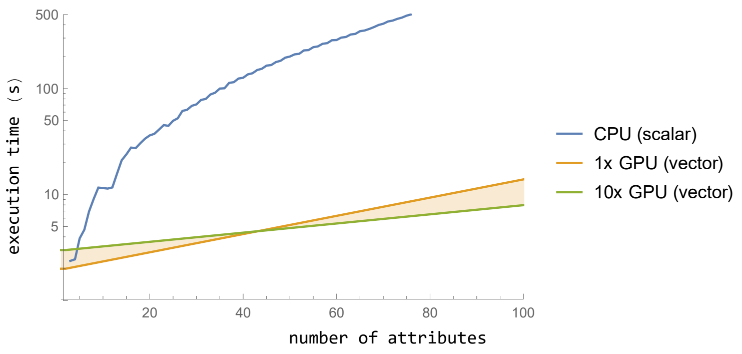
Varying Cluster Sizes. Recognising the potential of simultaneously conducting calculations on 5120 CUDA cores per hardware component, the question of scaling limitations also emerges with GPU resources. Currently, not all libraries support multi-GPU setups, yet their implementation is still worth it.

With the ability to run calculations on 5120 CUDA cores per hardware component, the subject of scaling limits also arises with GPU resources. Although not all libraries currently provide multi-GPU arrangements, they are nevertheless desirable. Figure 5.10a delineates the original scalar based implementation running on common CPUs against a vectorised algorithm on one and ten GPU nodes. Only a small variation in the vectorised approaches is evident due to the axis size. Converting the same to a log scale Figure 5.10b illustration highlights more details. A slight overhead, in the beginning, is visible that represents the divide and conquer preprocessing. After the workload is distributed across multiple nodes by the CPU and a potential overhead in bus I/O between main memory and GPU memories is bridged, the break-even point of the scaling becomes transparent. The anticipated execution time for n attributes bigger than our experiments, in particular, appears to be highly promising.

Resource Utilisation For monitoring the utilisation of the GPU nodes during the experiments, every second a data points is measured and further aggregated to 5-second intervals for visibility reasons. Figure 5.11a depicts compute consumption over time, whereas Figure 5.11b summarises the memory usage. Both figures highlight that there are still enough resources available, indicating that



(a) Time complexity with different compute resources



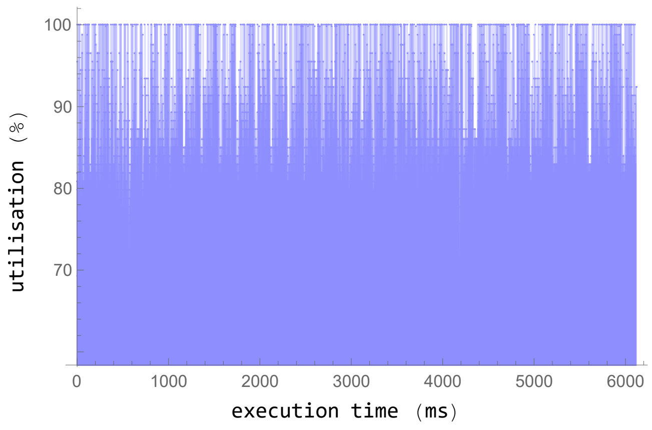
(b) Time complexity with different compute resources on log scale

Fig. 5.10: Runtime comparison by resources for Bayesian inference calculations

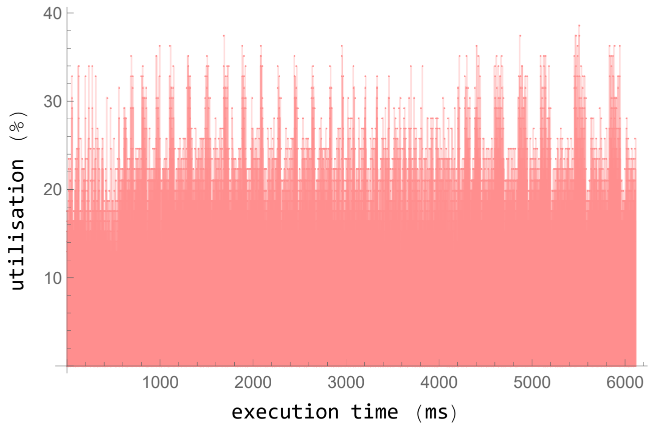
there are various chances to expand data sizes. Further optimisation modifications are left for future work. Simultaneously, we like to draw attention to the 20% underutilised compute capacity and 70% spare memory space. In fact, in all of the conducted experiments, not a single GPU node exceeded the whole available capacity of the system.

In sum, these experiments and its observation highlight that we can determine data inferences and quasi-identifiers that can expose private data. The search and its NP-hard complexity can be improved by mathematical optimising by 20% on 2^n for $n = 100$, and introducing the latest technological enhancement in the shape of vectorised GPU compute by 99.9% to a few seconds almost near-real time. For larger n, which are currently impractical to perform, these optimisations still offer a magnitude of runtime improvement.

Therefore, being able to discover inferences and quasi-identifier in a blink of an eye, we will pivot our attention towards eliminating such privacy-violating attribute combinations in the following chapter.



(a) Compute utilisation



(b) Memory utilisation

Fig. 5.11: GPU resource utilisation during experiments

6

Improving the Quasi-Identifier Elimination

To achieve the privacy guarantees advertised in previous Definition 4, we have thoroughly completed the search for quasi-identifiers and privacy-violating inferences in the previous chapters. Knowing what attribute values can be used in re-identification attacks is the first step to prohibiting private data exposure. In the second step, these attribute values need to be alienated or removed in a way that their risk of exposure is averted. We could remove all involved data records, yet this leads to a high information-loss as well. The removal can range from fine-grain individual values to excluding the entire attribute (or column). A proper balance of removal and the information-loss are desired, as the elimination of attribute (values) can usually reduce the corresponding dataset's meaningfulness. Related work has explored various algorithms for such alienation of data values, either by some kind of aggregation through syntactic data anonymisation techniques or through randomisation with semantic data anonymisation approaches (see Chapter 2).

In the following, we extend and generalise an algorithm that does not alienate the data values themselves, rather their relations of forming privacy-violating tuples in a syntactic manner row-wise.

6.1 Syntactic Data Anonymisation Methods

The family of syntactic data anonymisation algorithms build on data transformation techniques that somehow generalises the underlying data values through some generalisation [67, 166], suppression [67, 166], or perturbation [104, 147]. In essence, these approaches try to achieve some k-anonymity schema, where each tuple in the data set is classified with at least $k-1$ similar data records to limit distinguishability. The $k-1$ nearest neighbours are, according to Sweeney, selected based on similar describing attributes and enforced by functions that manipulate the designated attribute values composing a quasi-identifier [166].

As we will detail in the following section, the extended and generalised *attribute compartmentation* aims at the same goal of achieving k-anonymity but rather than modifying the attribute values instead of the possible association of attributes being shaped. This targets the objective to prevent the formation of attributes forming quasi-identifiers through any foreign key pair or functional dependency.

The following section will describe the *attribute compartmentation* concept in-depth, exemplary showcase its functionality, and demonstrate its performance against established syntactic data anonymisation algorithms.

6.2 Attribute Compartmentation

As a novel syntactic data anonymisation technique, attribute compartmentation has been initially introduced for a specific health data setting [135]. By extending and generalising this methodology, the aim is to break the data value relations forming quasi-identifiers (QID) instead of alienating or appending the attribute values themselves. This can be realised row-wise, per data record, instead of generically for all rows of a describing attribute.

To do so, after identifying the quasi-identifiers that violate the definition of anonymity 3.1, we classify the identifiers that adhere to anonymity. Existing research focuses on sophisticated statistical approaches which are highly use-case sensitive [181]. Standard use-case agnostic data transformations measure such as *local suppression*, *global generalisation*, and *perturbation* [166] can be used to complement attribute *compartmentation*. Formally, we define:

Definition 10. *Compartmentation as an admissible family of feature sets*
Let $Q = \{Q_1, .., Q_n\}$ *be a set of quasi-identifiers and denote with* $F \subset Q$ *a feature set (a set of quasi-identifiers). A family* $\mathcal{F} = \{F_1, .., F_m\}$ *of feature sets,* $F_i \subset Q, 1 \leq i \leq m$*, is called admissible, if:*
$\forall\ F_i, F_j \in \mathcal{F}, i \neq j : \bar{Q} = F_i \cup F_j$ *is k-anonymous set of features for given data* ($k < 1$)

Basically, *compartmentation* separates attributes forming quasi-identifiers in different pools. Without the ability to rejoin the original dataset, all additional attributes from the original dataset are duplicated and included in the split compartments. Because partitions are intuitively disjoint (disjoint and distinct) and compartments are intended to overlap, it's helpful to separate *compartmentation* from partitioning. Finding the best compartment with the fewest partitions possible is part of *compartmentation*. This holds as every supplemental compartment adds to the original dataset's partial redundancy. To counter redundancy, the following process is desired: All column combinations (mpmUCCs) will be projected to a graph (see Figure 6.1a). This graph now represents all attribute tuples that should be combined. Next, the graph representation will be inverted to describe all desired attribute combinations like delineated in Figure 6.1b. The outcome of discovering maximal cliques in this network will reveal to us all potential and best-compressed compartments that can split the original dataset into parts while adhering to the previously established anonymisation criteria (see Figure 6.1c). After separating the original dataset into compartments based on the needs of each row, the separations can be preserved as separate datasets or merged using a FULL OUTER JOIN.

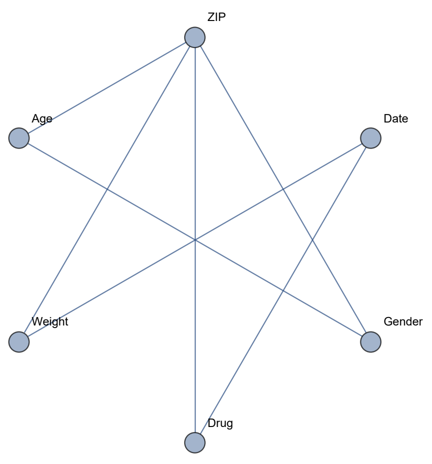
6.3 Experiments

To evaluate *attribute compartmentation*, we study its impact on data anonymisation using several common precision metrics that we briefly outline in the evaluation setup, followed by a discussion of our results and an in-depth use case study.

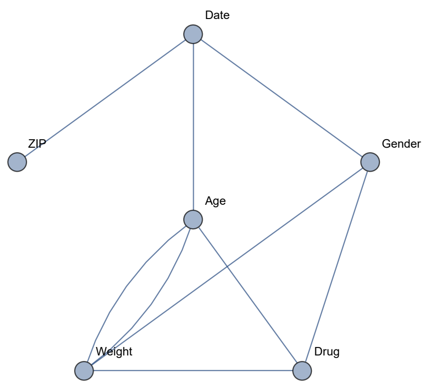
For the comparison, we selected established anonymisation techniques like global generalisation [67, 166], local suppression [67, 166], and perturbation [104, 147] that are of syntactic and not semantic, heuristic nature similar to the novel approach. A summary of global generalisation and local suppression is available in Section 2 and more details are available in Latanya Sweeney [166], N. Li et al. [98], and Ashwin Machanavajjhala et al. [106] work.

Setup & Metrics. The data quality is determined based on three factors illustrated in Table 6.1. Unique values are awarded by one point, duplicates, less than one point depending on the number of duplicates. Falsified values, which include any alternation from the original value, are penalised by some function measuring their distance from the original value.

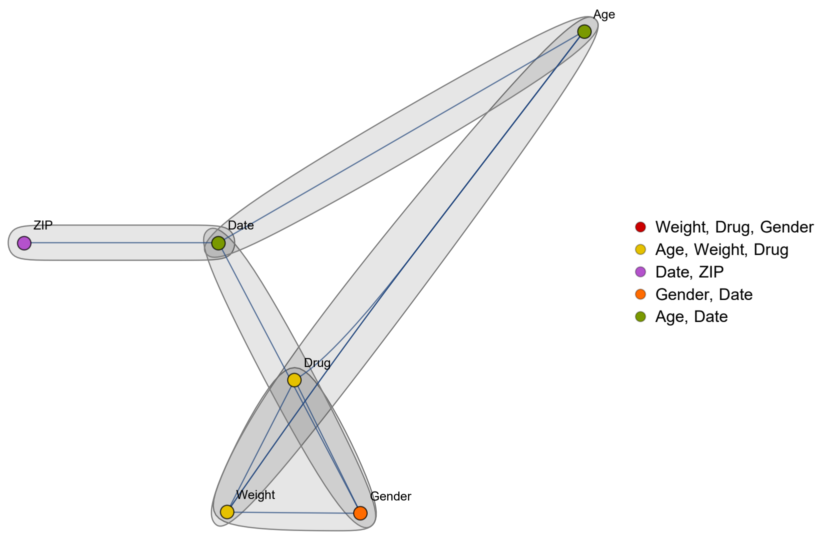
Hardware. The experiments were conducted on a machine equipped with 16x Intel Xeon E5-2697 v3 @ 2.60GHz and 32GB RAM using an enriched dataset with 109 attributes and 1M rows. We have taken real-world data from multiple sources and enriched those with fake profile data in close adjustment with real-world data distributions to ensure a fair evaluation set that can be un-



(a) Graph containing quasi-identifiers



(b) Complemented graph illustrating all allowed attribute combinations



(c) Max cliques in the completed graph representing compartments with the lowest redundancy

Fig. 6.1: Steps for creating attribute compartments

anonymously published for reasons of confirmability and traceability without endangering single entities through their personally identifiable information.

Dataset. As an experimental dataset, we re-used the previous semi-synthetic dataset to facilitate a repeatable evaluation and provided raw data for a side-by-side comparison. It has been created from various sources, including government websites, statistical data sets, and datasets that have already been made available to synthesise a semi-theoretical health dataset. An exhaustive list of all-inclusive list of attributes can be found on Github [136]. In contrast, a more

Table 6.1: Weighting of values for the data score

Condition (attribute value)	**Score**	**Comment**
unique	+1 point	rewarding diversity of values
duplicated	$+\tanh(1/x)$ points	x = number of duplicates (x)
falsified	$-\frac{\text{distance}}{\text{original value}}$ points	only for numeric values

condensed list depicting the used attributes can be found in Appendix A. This dataset contains quasi-identifiers (QIDs), which grow as one adds more describing data attributes (see Figure 6.2).

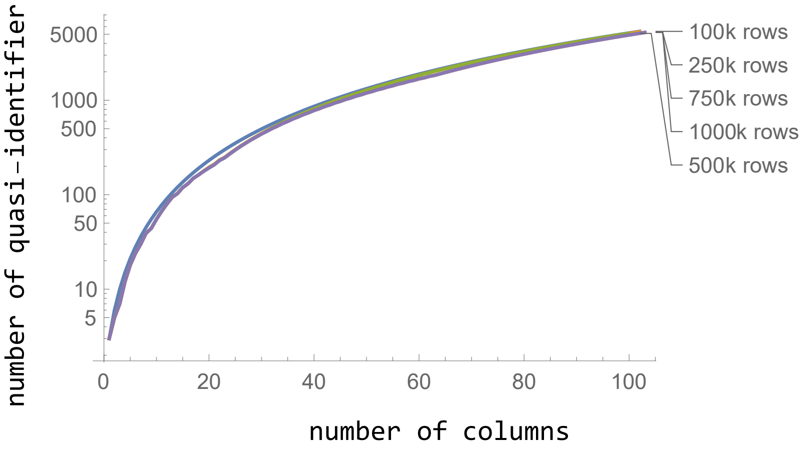


Fig. 6.2: Number of hidden quasi-identifiers in the dataset

Evaluation. Figure 6.5 depicts the number of values that must be changed for each transformation to ensure anonymity, divided by the number of columns. The percentage of modifications made in terms of changing the original dataset can be calculated using the *distortion rate* depicted in Figure 6.6. However, because *attribute compartmentation* does not change individual values, the default distortion rate remains 0. Also, established means to measure information-loss will give attribute compartmentation an unfair advantage as these rely on some means to calculate alienation of attribute values (ie., from an exact age to an age range). In fact, the information-loss metrics would certify an almost perfect ratio, which is not the case because attribute compartmentation does suffer information-loss, not in the attribute value but their relationships. Hence, we will adapt established metrics by awarding and penalising duplication to some extent (following tanh curve that can be customised per dataset with individual characteristics). The more individual values that must be transformed, the longer the technique will usually take and the lower the data score can be. Figure 6.3 illustrates this concept. The more quickly a higher data score is achieved, the less data quality loss can be achieved.

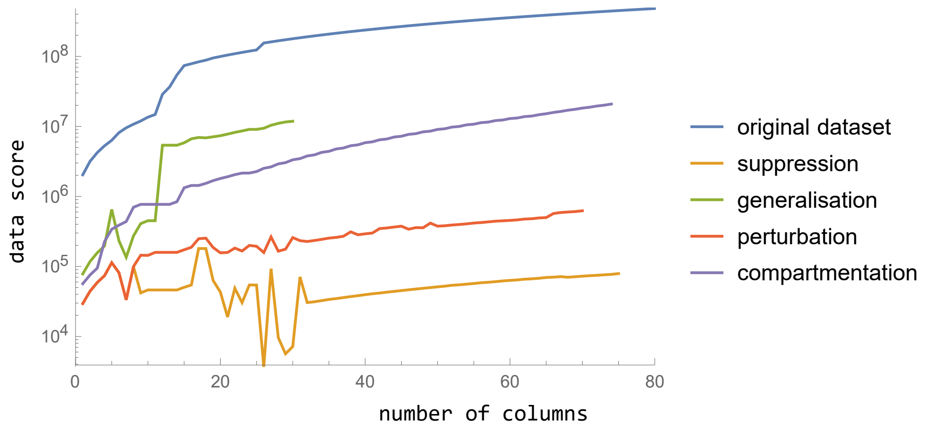


Fig. 6.3: Data score evolution comparing various syntactic anonymisation techniques

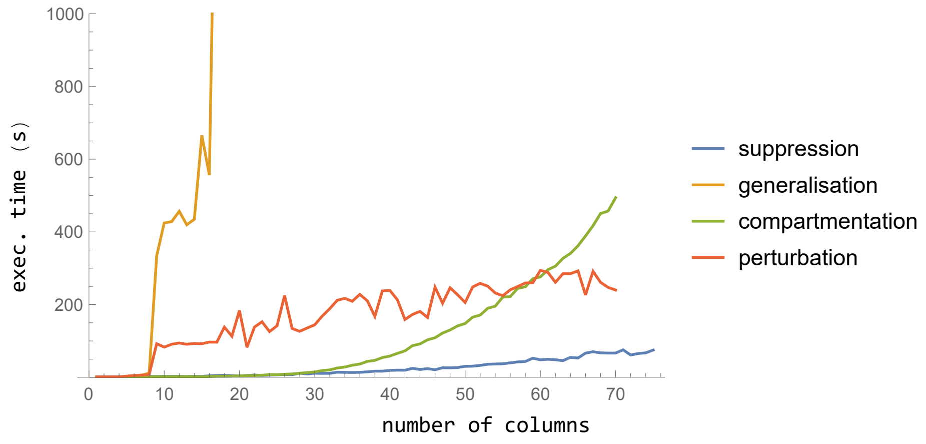


Fig. 6.4: Execution time growth comparing various syntactic anonymisation techniques

Finally, Figure 6.4 shows the temporal complexity of each individual therapy. Because of its repetitive design, generalisation describes exponential growth. Figure 6.7a illustrates the uncompressed data sizes of the sanitised datasets, with compartmentation increasing significantly faster than the other designated methods. This is dependent on the number of compartments required for effectively segregating attributes in quasi-identifiers in order to prevent their use as

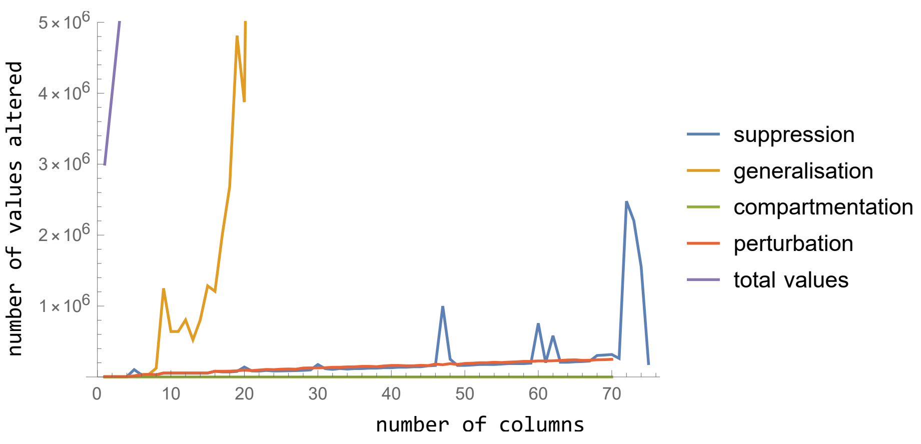


Fig. 6.5: Data value alteration growth comparing various syntactic anonymisation techniques

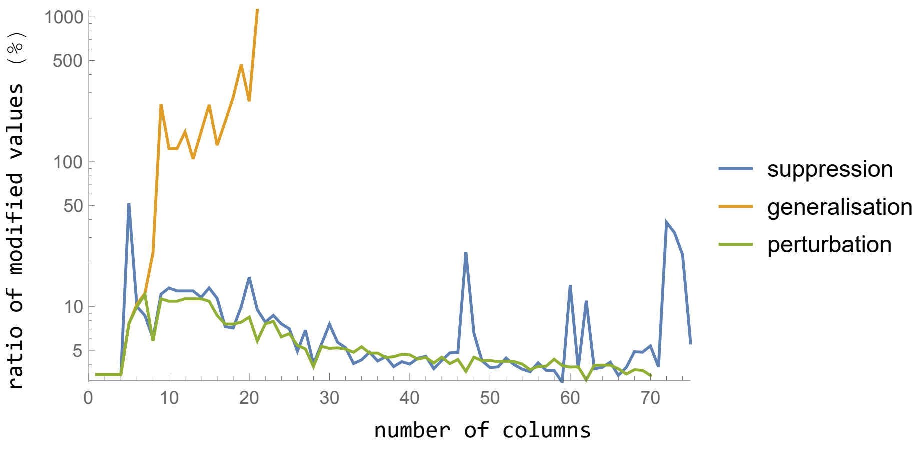
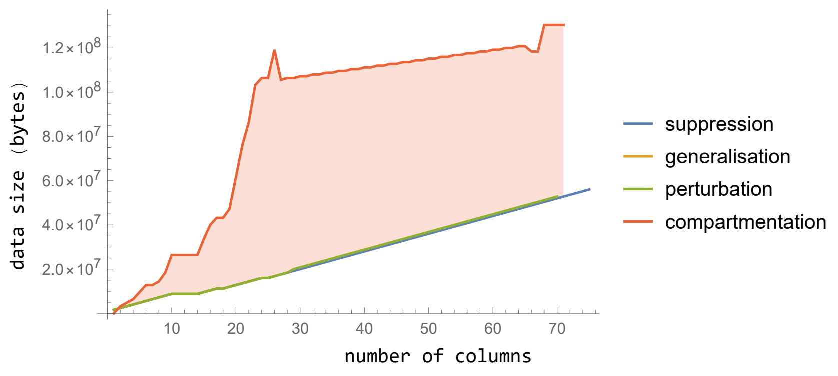
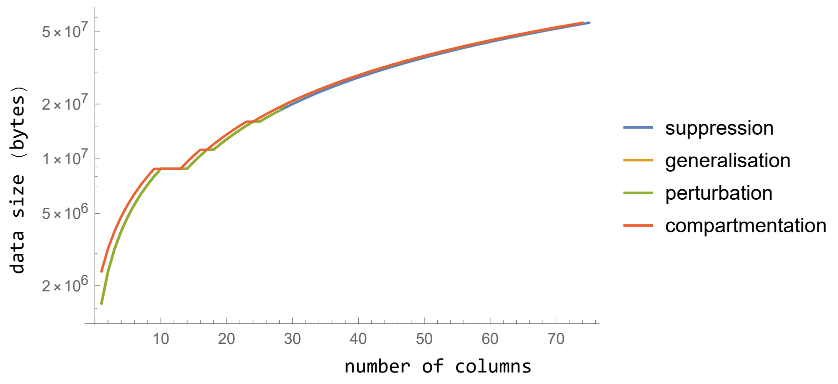


Fig. 6.6: Distortion rate evolution comparing various syntactic anonymisation techniques

an identifier. According to John W Moon and Leo Moser, there are $3^{n/3}$ maximum cliques for every n-vertex graph [114]. This can be explained through the growing possibilities of assembling (maximal) cliques as compartments in such a manner that their number does not have to grow. The Bron–Kerbosch method was developed by Coen Bron and Joep Kerbosch [22] to discover maximal cliques in $O(3n/3)$ [172]. As a result, the time complexity of *compartmentation* can be likewise equated to $O(3n/3) = O(1.4422n)$. Furthermore, it has been observed that the practical time complexity is quicker [26, 59]. On the execution runtime for attribute compartmentation, Figure 6.9 depicts the increase of dimensions (number of rows and columns). In the case of an in-memory application that uses dictionary encoding, such as SAP HANA, a redundancy of factor 20 or 30 can be obliviously overlooked, as shown in Figure 6.7b.



(a) Data size (uncompressed)



(b) Data size (compressed)

Fig. 6.7: Anonymised dataset characteristics

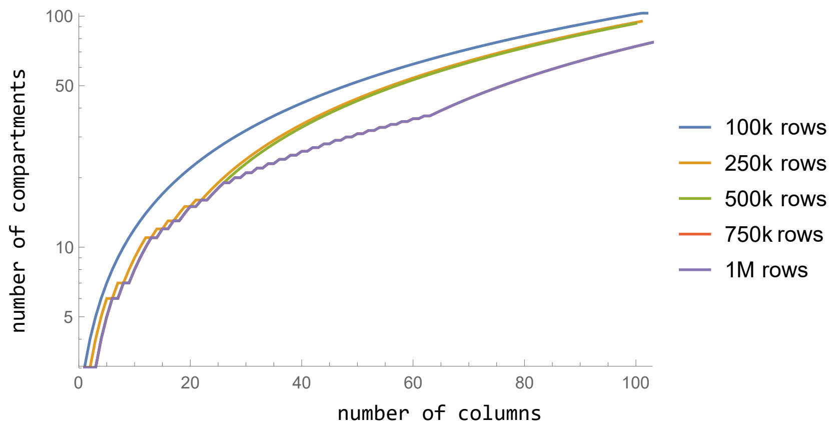


Fig. 6.8: Compartments over ncols and nrows

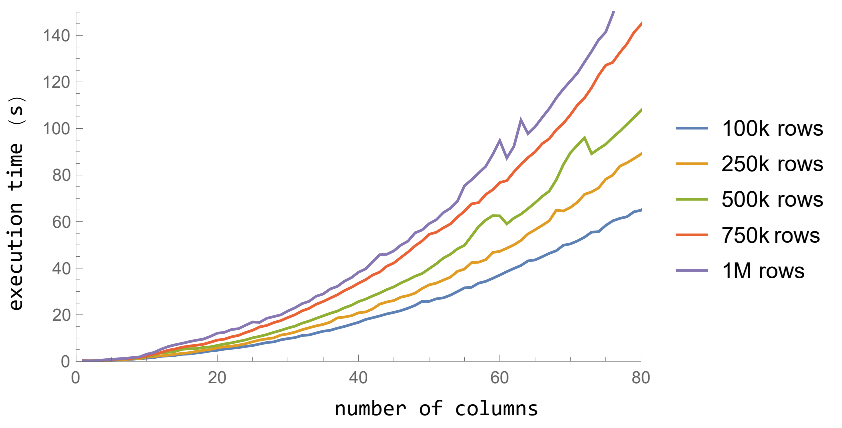


Fig. 6.9: Runtime complexity of compartmentation over ncols and nrows

Local suppression noticeably prevails in terms of time complexity, whereas compartmentation still surpasses generalisation (see Figure 6.4). However, it should be highlighted that suppression causes significant information loss, which shall be prevented. Linear growth is given by perturbation and suppression, whereas an exponential increase applies to generalisation and compartmentation. This is unsurprising given that suppression, perturbation, and compartmentation techniques do not require re-evaluation. Yet, due to its recursive nature, generalisation results in a considerable increase in execution time. The overall quantity of modified attribute values grows exponentially with the size of columns in Figure 6.5, whereas the number of modified data for generalisation, suppression, and perturbation grows linearly with the number of columns per modified value. Compartmentation does not affect attribute values; thus, it is not shown. Although the linkage between some of the attributes is abolished during compartmentation, the sanitised dataset retains the majority of its cardinality, resulting in a significant improvement in the data score (see Table 6.1). A high data score implies desired data quality depicted in Figure 6.3. The evolution of the data quality metric for each method portrays a similar picture in high-dimensions, yet on a different magnitude level. For a smaller number of columns, noise in the data affects especially suppression and slightly perturbation. However, we cannot compare the findings over the entire length of available columns due to the exponential increase in computing time. Generalisation quickly exceeds manageable run time, wherefore its data scores for large n can only be adumbrated. The data sizes of most syntactic data anonymisation procedures are similar after their transformations (see Figure 6.7a), but compartmentation stands out. This originates from the deliberate introduction of column redundancy in various compartments. The same redundancy can be effectively reduced by implementing dictionary encoding (see Figure 6.7b). In fact, initial testing suggests a redundancy factor of 30 compartments for 70 supplied attributes, implying that one actual data value can have up to 30 indices, which has no bearing on the compressed data size. The number of compartments does not expand exponentially because, as more quasi-identifiers are introduced, other combinations of maximal cliques become feasible, avoiding the need to increase the number of existing cliques in proportion (see Figure 6.8).

6.4 Real World Implication of Anonymisation on Digital Health Disease Triage

While the former benchmark serves as a generic and use case-independent evaluation of the introduced algorithms, this section will provide an exemplary exercise of orchestrated transformation approaches for a predefined real-world use case given the dataset and benchmark setup from Section 6.3.

We learned from a multinational pharmaceutical and life sciences company that a typical research use case is to find drug-to-drug, gene-to-drug, disease-to-disease, or drug-to-disease relationships. This can be typically achieved through a regression approach or by using the Bayes Theorem. Following Hayden Wimmer and Loreen Powell's approach to investigate the effects of k-anonymity on common machine learning algorithms [183], we use the same methodology to investigate the effects of different treatments and especially their composition to this use case.

Table 6.2: Combining anonymisation algorithms (c answers to the column's cardinality threshold)

Approach	**Data type condition**	**Composition A**	**Composition B**	**Composition C**
Perturbation	numeric	$c < 50$	$c < 75$	$c < 90$
Suppression	categoric	$90 < c < 100$	$80 < c < 100$	$95 < c < 100$
Generalisation	numeric	$50 < c < 100$	$75 < c < 100$	$90 < c < 100$
Compart-mentation	categoric	$c < 90$	$c < 80$	$c < 95$

The composition of transformations and their application thresholds was created by interpreting the benchmark results of Section 6.2 in combination with a weighted brute force approach where the time complexity is represented through an exponential interval. At the same time, the decision criterion is the achieved data score of the sanitised dataset. Multiple different treatment compositions have been evaluated (see Table 6.2) and finally decided on the favoured one depicted in Table 6.3.

Table 6.3: Optimal composition of syntactic data anonymisation techniques

Numerical	**Coverage**[10]	**Treatment**
yes	>15	generalisation
yes	<50	perturbation
no	<50	suppression
*	>=50	compartmentation

Using this setup of thresholds and transformations, a sanitised dataset is created. Figure 6.6 illustrates this data transformation as a partial selection of the available feature set. For comparison, the same logistic regression function is applied to the original dataset as well as a sanitised dataset that has only been converted using one of the data transformation methods. The transformation time for anonymising the underlying dataset with 73 attributes based on the transformation composition configuration takes roughly *168s* resulting in a data quality score of 15983644.28 and a size of 52800080 bytes (uncompressed).

As a hypothesis, the following analysis provides influencing factors for *DOID:3393*, namely “coronary artery disease”, where plaque conglomerates along the inner walls of an artery reducing the blood supply to cardiac muscles [125]. As part of the feature selection, we determine centimetres (height), age, blood type, kilograms (weight), as well as several single-nucleotide polymorphisms (SNPs) markers and the patient’s drug intake as interesting features for the logistic regression. The attribute coefficients are given as weights in Table 6.4 to influence the likelihood of coronary artery disease. Given the initial information, the reader will quickly notice that the patients’ age, weight, and height are significant predictors of DOID:3393. A link between blood type, drug usage, and coronary artery disease can also be discovered.

While only depending on perturbation or suppression for anonymisation, the coefficients shift toward one feature, which is not surprising given these methods’ nature. Compartmentation keeps most of the features. However, re-weights them slightly differently. The composition of weights performs the best with deviations

[10] The coverage corresponds to the percentage of rows needed to be transformed for the present quasi-identifier.

Table 6.4: Logistic regression coefficients as scaled weights for the given attributes as features

Attribute	**Original coefficients**	**Composition coefficients**	**Compartmentation coefficients**	**Perturbation coefficients**	**Suppression coefficients**
Age	100.00	100	32.99	0	0.05
Centimeters	49.44	37.48	100	100	6.8
drug_0	63.38	0	4.3	0	100
BloodType	33.96	8.82	45.06	0	0.05
Kilograms	50.53	62.29	24.25	0	0
snp_0	0	0	0	0	0
drug_1	0	0	0	0	0
drug_2	0	0	6.4	0	0

of 10% to 20%. Table 6.5 delineates the type I and type II errors for predicting coronary artery disease.

Table 6.5: Type I and type II errors for predicting coronary artery disease

	False Positives	**False Negatives**
Original	0.05	0.01
Composition A	0.08	0.021
Compartmentation	0.11	0.025
Perturbation	0.23	0.15
Suppression	0.21	0.14
Generalisation	0.10	0.072

This proves the previous results of the use-case agnostic evaluation that compartmentation introduces much better data quality. Also, in composition with existing approaches, the best possible anonymisation results can be achieved.

Besides publishing the data basis, we also make the anonymised dataset publicly available, engaging the community to review the results [136]. Since there remain no unique tuples from the original dataset, homogeneity and background knowledge attacks are significantly exacerbated.

Table 6.6: Data transformation sample for disease and SNP being combined a quasi-identifier

(a) Before processing

City	**Disease**	**SNP**	**..**
..	..	..	
Bad Sülze	DOID:12361	rs104894270	..
Bad Sülze	DOID:12361	*	..
Bad Sülze	DOID:1024	rs104894503	..
..	..	..	

(b) After processing

City	**Disease**	**SNP**	**..**
..	..	..	
Bad Sülze	DOID:12361	n/a	..
Bad Sülze	n/a	rs104894503	..
Bad Sülze	n/a	rs104894270	..
Bad Sülze	DOID:1024	n/a	..
..	..	..	

7

Overarching Evaluations

As each chapter's evaluation focuses on comparing the new contribution and its specific comparison to related work, the overarching evaluation provides an overview and delineate the consortium of contributions among shared metric. As some of the contributions originate in different processing steps or methodology, not all angles can be compared.

7.1 Experimental Hardware

The experiments performed in the following utilised a large compute cluster, which is equipped with the latest Intel Xeon E5-2698 v4 (160 CPU cores), 760GB RAM, and ten Nvidia Tesla V100. Each of these GPU units has 5120 CUDA cores and a combined Tensor performance of 1120 TFlops. The execution environment is limited to a single dedicated Tesla V100 and ten CPU cores unless classified otherwise. Hardware requirements, including the CPU cache hierarchy, are detailed in Appendix B.

7.2 The Dataset

For algorithmic measurements, a reproducible semi-synthetic dataset has been assembled. Similar to the evaluation dataset in Sections 4.5, 5.4 and 6.3 various sources have been explored and concatenated. These data repositories including official government websites, public statistical data and datasets as part of previous publications to extract raw data and mimic their attribute distribution. A full list of all attributes is accessible on the Github website. [136]. Given its nature, the publicly available dataset includes inferences and various quasi-identifiers (QIDs), which increase over the number of available describing attributes highlighted in Figure 7.1.

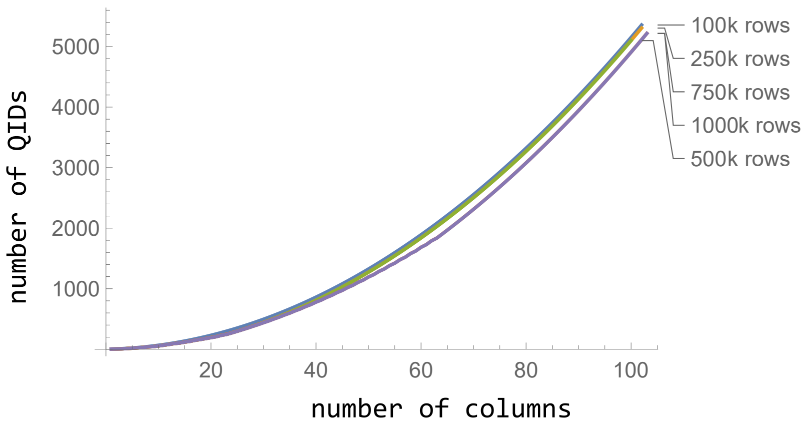


Fig. 7.1: QID quantity evolution over growing describing attributes

7.3 Experiments

For the final and overarching evaluation, there are three main objectives to monitor. First, the novel approaches shall demonstrate their soundness to discover all quasi-identifiers. Second, involved data processing activities on a larger scale shall be practical with reasonable efforts. Third, after data anonymisation, no unique attribute patterns that allow a unique re-identification shall remain.

7.3.1 Anonymity Constraint

The precision of the novel approaches will be measured by the number of discovered quasi-identifiers and compared against the total number of QIDs. Each remaining unique attribute combination that serves as QID may lead to a successful re-identification of its data owner, and therefore answers to the risk of private data exposure.

Figure 7.2 depicts the total number of QIDs that are actually present in the dataset as formalised in Chapter 3. Using the approaches from Chapter 4 summarised as mp(m)UCC guarantees to find all of these actual QIDs. The Bayes approach from Chapter 5 achieves similar results dependent on the designated threshold, yet they carry the burden of additional false positives as visualised in Figure 7.2.

However, the important observation that can be derived is that all new approaches guarantee the correct identification of QIDs in the sample dataset and therefore reliably level the next steps for their necessary removal or fracturing.

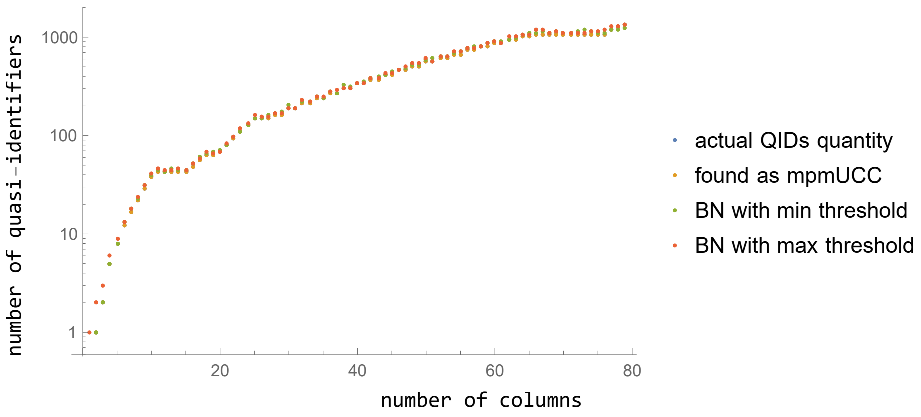


Fig. 7.2: Comparing anonymity constraint as algorithmic precision

7.3.2 Algorithmic Complexity & Runtime

The next observation angle addresses the runtime implications of algorithmic complexity. As part of the detailed comparison within each chapter, the reader gained advanced insights into the novel methodology against its specific baselines. Figure 7.3 offers a summarised perspective for various introduced contributions. While the exact mpUCC search scales similar to the baseline by Braghin et al. [20], the vectorised optimisations clearly outperform previous work by magnitudes also in high-dimensions.

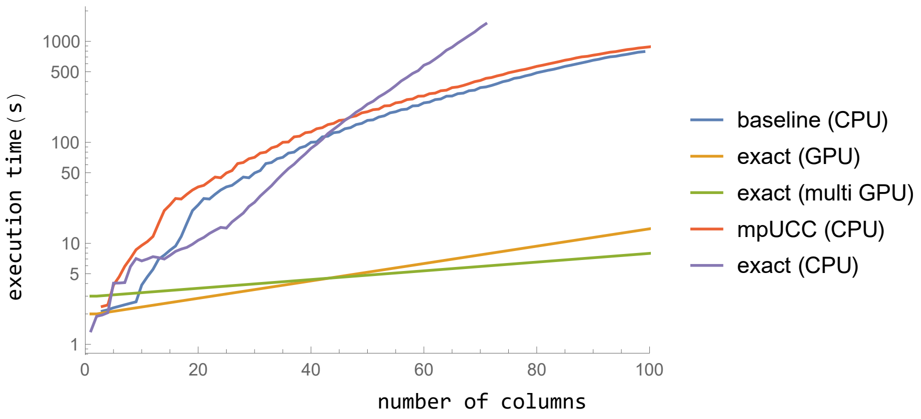


Fig. 7.3: Run time comparison for QID search

Changing the angle from pure quasi-identifier candidate processing to the anonymisation compute step removes the same attribute combinations identified as QIDs. Figure 7.4 delineates a comparison between established syntactic data anonymisation techniques and the new ones. For a fair comparison, all algorithms ran on exactly the same hardware resources. Both *attribute compartmentation* and *Bayesian network* achieve good results when comparing to well-proven generalisation. Given the nature of the Bayesian network's state-space explosion, its runtime decreases for a large column amount. Yet, reminding the reader on Bayesian GPU acceleration contribution that significantly accelerates model training, particularly in high-dimensions (see Section 5.4).

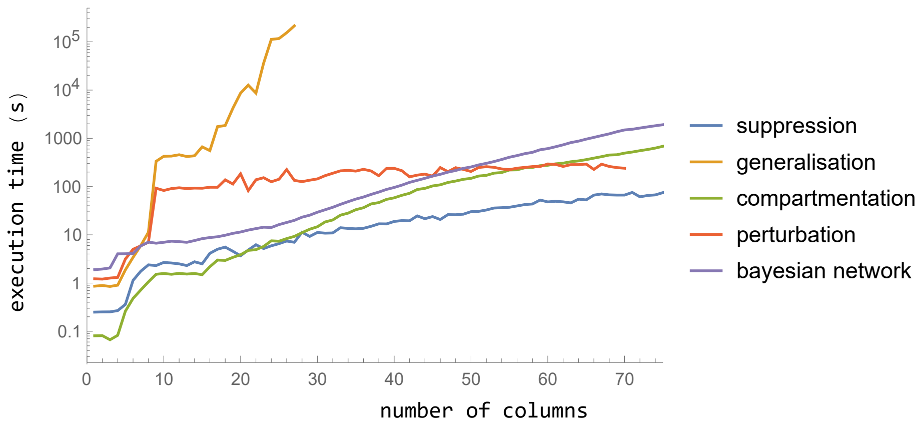


Fig. 7.4: Run time comparison of designated syntactic data anonymisation techniques

From an algorithmic complexity, estimating the worst-case runtime essentially confirms the original NP-hard nature of the "Find-QID" problem (see Chapter 3). This can be expressed in the following manner:

Definition 11. *Big O Notation:*
Let $f, g : \Omega \longrightarrow \mathbb{R}$ be functions on a subset $\Omega \subset \mathbb{R}$ of real numbers. Asymptotically, we write $f(x) = O(g(x))$ as $x \to \infty$, if there are constants $c > 0$ and $x_0 \in \Omega$, such that $|f(x)| \leq c \cdot |g(x)|$ for all $x \geq x_0$. This indicates, that f grows slower than g and g bounds f from above for all sufficiently large x.

For instance, $O(n)$ represents an algorithmic approach with a linear complexity, while $T(n) = O(n^2)$ denotes a quadratic one.

For the given approaches, the exact discovery of QIDs takes nearly $O(2^n)$ which is indicated in Figure 7.3 also. As previously discussed, reducing the search to the minimal QIDs as mpmUCCs, a complexity of roughly $O(2^n/2) = O(2^{n-1})$ can be assumed. For optimisation approaches, the same worst-case remains, yet the actual compute runtime is significantly accelerated through the contributions described in Chapter 4 and Chapter 5.

With the identification of attribute-value combinations serving as QIDs, the anonymisation approaches for mitigation become interesting. As the best in class mechanism for time complexity remains suppression. Along with the linear runtime, suppression also introduces the highest possible information-loss through removing the designated information. The algorithmic opposite would be a generalisation, which suffers an exponential runtime due to the iterative and incremental processing nature of aggregating and re-evaluating new group sizes for compliance. As a consequence of the additional compute effort, the lowest information-loss can be achieved (see Figure 7.4). The novel attribute compartmentation mitigates same compute complexity to the well-known problem of searching for max cliques in a graph structure. Moon et al. showed that for every n-vertex graph $3^{n/3}$ maximal cliques exist [114]. The Bron–Kerbosch algorithm introduced by [22] can be used to find the maximal cliques in $O(3^{n/3})$ [172], wherefore the time complexity of *compartmentation* is $O(3^{n/3}) \approx O(1.4422^n)$ as delineated in Figure 7.4. Still, the time complexity has been reported to be faster in practice [26, 59].

In sum, the runtime results indicate quite impressive compute speedups. These acceleration combine to reduce data processing efforts from original hours and minutes to a few seconds compute in total as depicted in Figure 7.3. Running anonymisation queries in short time windows while guaranteeing the absence of QID and therefore making re-identification impractical is one of the targeted key objectives.

7.3.3 Data Quality (Information Loss)

Information loss is one of the most important criteria for anonymisation techniques next to their privacy guarantees and runtime. The more an approach alters the origin dataset, the higher the information loss can be expected. That information-loss can be measured and monitored either through some data quality scores or through the number of attribute-value alternations. Often, the latter of attribute value alternations correlates to the runtime since the more often an algorithm has to adjust designated attributes values, the slower its runtime is. Particular with iterative and incremental characteristics of algorithms like in the case of generalisation, attribute value alterations are visible.

To calculate the data score itself, we orient ourselves towards common procedures and create a data score based on three factors. The first indicator is the number of unique values – the more, the better. Since duplicate values are better than none, each duplicated data value is scored with

$$\tanh(1/x) = \frac{\sinh(1/x)}{\cosh(1/x)} \tag{7.1}$$

where x is the number of duplicates per unique value. A falsified (not aggregated) value is penalised by considering the distance to the original one according to

$$1 - \frac{\text{distance}}{\text{original value}} \tag{7.2}$$

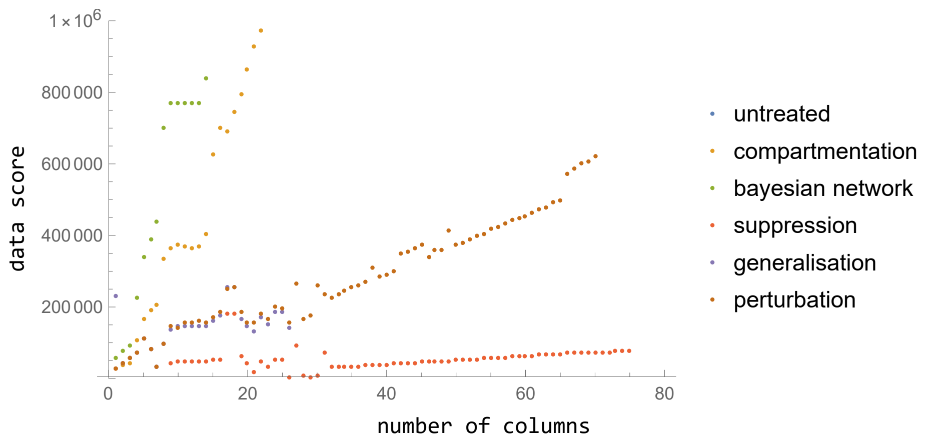


Fig. 7.5: Data score comparison

Figure 7.5 demonstrates this as data score over the increasing number of columns and describing attributes. In this chart, the quicker an evolution reaches higher data scores, the better the approach is suitable. Suppression represents a pretty radical approach and therefore suffers the most information loss represented through low score values. While Figure 7.5 offers a comparison to established syntactic data anonymisation methods, Figure 7.6 delineates a selection of the novel methodologies against the best case of the original, untreated dataset.

Guaranteeing privacy, as we know, is a balance to losing information, yet this ratio differs quite intensively between available algorithms. Suppression as a methodology is more radical and therefore suffers higher information-loss while compartmentation maintains more information value (see Figure 7.5). Consequently, the drawbacks rest in data redundancy given the described nature from Chapter 6.

However, the search for quasi-identifier is not only an integral part of data anonymisation but also for de-anonymisation or re-identification attacks. The next section will showcase the usage of described optimisations and approaches

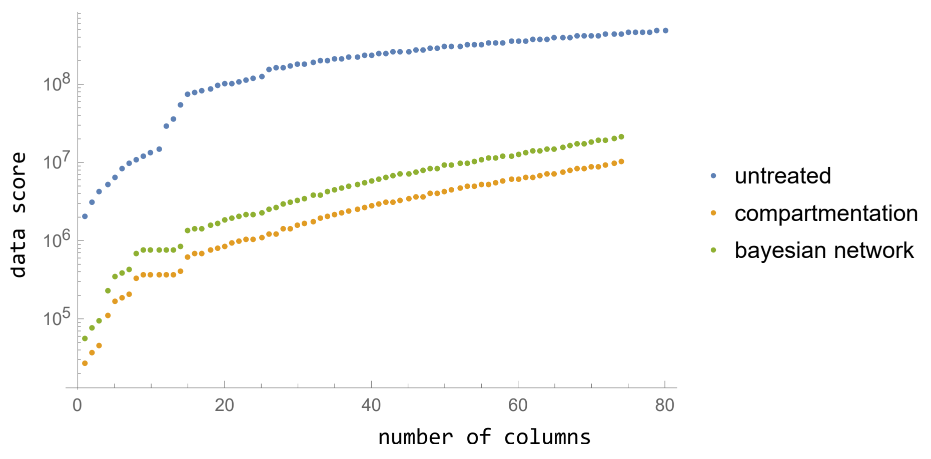


Fig. 7.6: Data score comparison of novel approaches

to re-identify individuals in a social media dataset about the US Presidential Election 2020 tweets to demonstrate our contributions' efficiency.

7.4 Re-Identification of Individuals Tweets in US Presidential Election 2020 Dataset

The previous evaluation results indicate a novel opportunity to ensure privacy while minimising information-loss. Yet, the same methodology can be flipped and used to attack existing, published datasets to re-identify individuals if not processed correctly in the first place. Particularly against probabilistic approaches, the sheer compute power can be used to easily brute-force de-anonymisation by try-and-error linkage guessing. Two examples should illustrate the successful utilisation of the accelerated QID discovery as a basis for de-anonymisation.

Figure 5.10a shows how the latest contributions make it possible to process 2^{100} attribute combinations for QID discovery at the touch of a finger. As a first sample, we looked at a publicly available social media dataset with more than 20 million entries about the 2020 US Presidential Election [154]. The hashtags "#USAelection" or "#NovemberElection", as well as at least one of the following parties keywords, were used as selection criteria:

- **Democratic Party:** @DNC OR @TheDemocrats OR Biden OR @JoeBiden OR "Our best days still lie ahead" OR "No Malarkey!"
- **Republican Party:** #MAGA2020 OR @GOP OR Trump OR @POTUS OR @realDonaldTrump OR Pence OR @Mike_Pence OR @VP OR "Keep America Great"

Table 7.1: US Presidential Election 2020 tweet dataset characteristics [154]

Attribute	**Description**
Created-At	Exact creation time of the tweet
From-User-Id	Unique ID of the user that sent the tweet
To-User-Id	Unique ID of the user that tweet sent to
Language	Language of tweets that are coded in ISO 639-1
Retweet-Count	number of retweets
PartyName	The Label showing which party the tweeting is about. [Democrats] or [Republicans] if the tweet contains any keyword
Id	Unique ID of the tweet
Score	The sentiment score of the tweets. A positive (negative) score means positive (negative) emotion.
Scoring String	Nominal attribute with all words taking part in the scoring
Negativity	The sum of negative components
Positivity	The sum of positive components

- **Green Party:** @GreenPartyUS OR @TheGreenParty OR “Howie Hawkins” OR @HowieHawkins OR “Angela Walker” OR @AngelaNWalker
- **Libertarian Party:** @LPNational OR “Jo Jorgensen” OR @Jorgensen4POTUS OR “Spike Cohen” OR @RealSpikeCohen

This dataset consists of 11 describing attributes that specify roughly 20 million data records (see Table 7.1). As we have learned, the search for quasi-identifiers can be applied to re-identify individuals as well, wherefore we use the exact search algorithm from Chapter 4 in this instance. The full algorithmic execution takes approximately 2.315s on the same experimental hardware cluster described earlier. As a result, we can re-identify individuals based on unique attribute combinations like their posting day, time, keywords and retweet count.

The following record is one of those outliers discovered in the available dataset (see Table 7.2). Using the unique attribute value tuples that form a quasi-identifier, one can link those to a single Twitter account that identifies *Kristine A.* from southern California.

We have removed the full last name for privacy reasons. Another demonstrating reference is *Cameron R.* from Mount Vernon, NY. Cameron completed Boston College as an undergrad in 2019 with a major in Computer Science and a minor

Table 7.2: Quasi-identifiers in Kristine's tweet data

Attribute	**Value**
Created-At	9/15/20 7:05 AM
From-User-Id	929777987892928514
To-User-Id	-1
Language	en
Retweet-Count	2578.0
PartyName	Democrats
Id	1305719270572085249
Score	-0.6410256410256411
Scoring String	destroy (-0.64)
Negativity	0.6410256410256411
Positivity	0

in Art History, as his profile states. He frequently uses his Twitter's Android app or the mobile website version as it appears.

There are more re-identification instances, but we keep it to the sample as it already proves the power of the contributed quasi-identifier search and incorporated optimisations.

Table 7.3: Quasi-identifiers in Cameron's tweet data

Attribute	**Value**
Created-At	7/1/20 7:44 PM
From-User-Id	950208325244870656
To-User-Id	-1
Language	en
Retweet-Count	2347.0
PartyName	Republicans
Id	1278368973134999552
Score	0.025641025641025644
Scoring String	matter (0.03)
Negativity	0.0
Positivity	0.025641025641025644

7.5 Discussion

This work formalises the “Find-QID” problem and discusses methods to resolve the same data anonymisation challenge for high-dimensional datasets. Simultaneously, a variety of optimisation approaches have been introduced and individually evaluated against established algorithms. As part of the overarching evaluation, we compared the different archetypes of finding quasi-identifiers and eliminating them in a second step. Each of the methods has different weights on runtime complexity and information-loss. Yet, all contributions guarantee the absence of QIDs and, therefore, can offer anonymity constraints in high-dimensions, which have been impractical before.

For quasi-identifier discovery, the runtime has been reduced from minutes and hours to a few seconds without compromising privacy. The same methodologies might not accelerate execution time on small datasets due to their nature but scale very well, as shown in the previous experiments.

While all analysed anonymisation approaches have their individual advantages and disadvantages, they can be differentiated in terms of time complexity, storage needs and data quality. Based on the previous findings, optimal privacy treatment methods can be chosen based on the number of affected rows within each quasi-identifier tuple. Suppression performs well on data with minor amounts of outliers where values cannot be generalised effectively. In contrast, compartmentation suffers higher information-loss than generalisation yet outperforms the same on its time complexity. Perturbation and generalisation only work for numeric values and not categorical attributes, while compartmentation and suppression are flexible regarding the data type. Perturbation falsifies the data values, and its execution time behaves similarly to suppression, but with a significant offset, albeit producing a slightly better data score. Unlike compartmentation and generalisation, both perturbation and suppression achieved data scores that constitute a dramatic loss of data quality. While a generalisation scheme’s runtime increases impractically fast with the number of attributes, compartmentation presents an excellent trade-off between time complexity and data quality.

To get there, quasi-identifiers have to be discovered reliably and efficiently. We introduced, analysed and discussed different techniques and optimisation schemes to evaluate QID candidates and ensure their detection. Benchmarked against the latest contribution patented by Braghin et al. [20], the exact search scheme performs worse and the mp(m)UCC equivalent in respect to runtime. Finally, the GPU accelerated approach outperforms state-of-the-art by magnitudes and enables a whole new privacy attack vector on previously published datasets as demonstrated in Section 7.4.

In sum, the “Find-QID” problem remains an NP-hard, W[2]-complete problem that is fixed-parameter tractable. Presented and discussed optimisation schemes alleviate the runtime symptoms for the current understanding of high-dimensions. Yet, as data sizes are continuously rising, the problem might surface in the future again.

8 Conclusions

In this work, we addressed the problem of publishing highly sparsely populated, multi-attribute, high-dimensional data in a privacy-preserving manner. As data-gathering techniques evolve, more and more describing attributes become available, forming datasets with many rows and columns. Data anonymisation processing to avoid privacy-violating inferences in such datasets is complicated and selected anonymisation methods are NP-hard.

As part of our work, we formalised the complexity of discovering quasi-identifiers (QID) as “Find-QID” problem due to the immense diversity of information. A sequence of reductions to the Hitting-Set-Problem indicates that the complete absence of QIDs and inferences is W[2]-complete or worse. This classification fortifies previous testimonies that achieving data anonymisation in certain settings like genome sequencing due to their high-dimensions is currently impractical. To solve this, we will summarise our contributions in the following.

8.1 Summarised Contributions

With the newly W[2]-complete classification of the “Find-QID” problem and its labelling as fixed-parameter tractable (FPT), we know that the first step of privacy-preserving data sharing leads to a superpolynomial runtime. FPT allows efficient algorithm for small values of the fixed parameter, but not in large, high dimensions. For high-dimensional datasets, we showed that this runtime quickly becomes impractical to process due to the high combinatorial count of QID candidates under verification. To compensate for such compute issue, we discussed the method of reducing the search space to minimal quasi-identifiers that bisects the candidate’s amount. Additionally, we introduced greedy sorting of the candidate queue based on tuple characteristics to accelerate processing by quicker finding quasi-identifiers in the search tree and therefore eliminating the entire tree branch. For both approaches, experiments confirmed their usability and theoretical improvements. The most significant advancement yet is achieved by vectorising the search algorithm and utilising massive-parallelisation through GPU hardware which outperforms previous optimisations by magnitudes. Despite this complexity, we showed that the discovery of QID can be achieved reliably for large data.

For privacy-preserving data sharing, one risk relies upon the presence of data inferences. Attackers can use data inferences combined with auxiliary data to draw conclusions and derive private information for de-anonymisation and re-identification attacks. Quasi-identifiers (QID) can serve as such inferences as well. To find data inferences reliably, we presented a model of utilising Bayesian inferences. Yet, generating Bayesian networks and sampling them is NP-hard. Even approximate approaches require much computation as Bayesian networks generate a new node for each unique attribute value, resulting in a state-space explosion for high-dimensional datasets. For this purpose, we proposed multiple novel methodologies of discovering and eliminating privacy-violating inferences in high-dimensional datasets. To aggregate such a state-space explosion, we have presented and evaluated the aggregation strategy from multigrid context to aggregate nodes or inferences being in all probability, reducing the Bayesian network in size. As a second venue, we have contributed a vectorised algorithm to calculate Bayesian inferences. While experiments confirm the acceleration due to the mathematical aggregation of nodes, the vectorisation and its massive parallelisation through GPU hardware outperform previous optimisations. In fact, inferences can be practicably achieved even for large, high-dimensional datasets without the need for supercomputers. As the evaluation and experiments showcased, even near real-time runtime for currently impractical applications can be achieved.

At the same time, we demonstrated how such runtime optimisation could be abused to attack an already published dataset. To illustrate this potential, we exemplary re-identified two individuals, *Kristine A.* and *Cameron R.*, in a Twitter dataset published on the US Presidential Election 2020.

Finally, this work extended, generalised, implemented, and evaluated the deterministic anonymisation methodology, *attribute compartmentation.* While established syntactic data anonymisation approaches alter the data value itself, *attribute compartmentation* promises sanitised datasets without remaining quasi-identifiers while minimising information-loss. This is being realised by breaking the linkage in attribute combinations forming quasi-identifiers in overlapping partitions without the possibility of directly re-joining. Ultimately, the attribute values remain untouched and the information-loss is minimised. The conducted experiments proved its functionality in real-life settings. We further showcased a medical use case study jointly with digital health experts to demonstrate that attribute compartmentation is suitable for everyday use and, as a side effect, even thwarts base rate neglect.

8.2 Open Problems

Reflecting on the past contributions of this work, three venues for future activities can be derived.

One approach that promised significant compute uplift has been vectorised or tensorised GPU-calculations. As we have briefly experimented with a multi-GPU setup, an in-depth benchmarking of certain hardware limitations would be interesting, especially as shifting data across the motherboard bus and VRAM for a large dataset is still limited. Horizontal scaling of GPUs remains restricted to the motherboard bus I/O. Therefore, a first venue can quantify these restrictions and elaborate on initiatives similar to utilising dictionary encoding for L1-L2 CPU cache optimisations to maximise direct GPU memory capacity. Appreciating that the experiments in this work have not nearly utilised available GPU memory by 50%, we expect the trajectory for large memory will quickly fortify.

A second venue for future activities may address extending practical insights on real-world scenarios similar to the previous digital health experiments we have provided. Kessler et al. [85] have contributed to implementing research from the privacy community in enterprise systems. Yet, there is still a significant gap from theory to practice resulting in an actual struggle to realise privacy standards for organisations. A detailed, quantitative survey on industry perspectives and implementation hurtles may be desired to start bridging the gap and overcome these implementation issues.

The third and final open opportunity is the perspective of additional side effects due to data anonymisation. We discussed if syntactic data anonymisation may foster base rate neglects in a digital health setting as part of this work. Expanding these insights on other application fields and further quantifying anonymisation effects on analytic queries would be interesting, especially towards causal inference and other fallacies.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318. ACM, 2016.

[2] Ziawasch Abedjan and Felix Naumann. Advancing the discovery of unique column combinations. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1565–1570, 2011.

[3] Abbas Acar, Hidayet Aksu, A Selcuk Uluagac, and Mauro Conti. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Computing Surveys (CSUR)*, 51(4):1–35, 2018.

[4] Charu C. Aggarwal. On k-anonymity and the curse of dimensionality. In *Proceedings of the 31st International Conference on Very Large Data Bases*, VLDB '05, pages 901–909. VLDB Endowment, 2005. ISBN 1-59593-154-6.

[5] Abdulatif Alabdulatif, Heshan Kumarage, Ibrahim Khalil, and Xun Yi. Privacy-preserving anomaly detection in cloud with lightweight homomorphic encryption. *Journal of Computer and System Sciences*, 90:28–45, 2017.

[6] Tyler Allen, Xizhou Feng, and Rong Ge. Slate: Enabling workload-aware efficient multiprocessing for modern gpgpus. In *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 252–261. IEEE, 2019.

[7] Boaz Barak, Kamalika Chaudhuri, Cynthia Dwork, Satyen Kale, Frank McSherry, and Kunal Talwar. Privacy, accuracy, and consistency too: a holistic solution to contingency table release. In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 273–282, 2007.

[8] Michael Barbaro and Tom Zeller. A face is exposed for aol searcher no. 4417749, Aug 2006. URL `http://www.nytimes.com/2006/08/09/technology/09aol.html`.

[9] Mauro Barni, Tiziano Bianchi, Dario Catalano, Mario Di Raimondo, Ruggero Donida Labati, Pierluigi Failla, Dario Fiore, Riccardo Lazzeretti, Vincenzo Piuri, Alessandro Piva, et al. A privacy-compliant fingerprint recognition system based on homomorphic encryption and fingercode templates. In *2010 Fourth IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, pages 1–7. IEEE, 2010.

[10] Daniel Barth-Jones. The're-identification'of governor william weld's medical information: a critical re-examination of health data identification risks and privacy protections, then and now. *Then and Now (July 2012)*, 2012.

[11] Roberto J Bayardo and Rakesh Agrawal. Data privacy through optimal k-anonymization. In *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*, pages 217–228. IEEE, 2005.

[12] Irad Ben-Gal. Bayesian networks. *Encyclopedia of statistics in quality and reliability*, 2007.

[13] Raghav Bhaskar, Srivatsan Laxman, Adam Smith, and Abhradeep Thakurta. Discovering frequent patterns in sensitive data. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 503–512. ACM, 2010.

[14] Armin Biere, Alessandro Cimatti, Edmund M Clarke, Ofer Strichman, Yunshan Zhu, et al. Bounded model checking. *Advances in computers*, 58 (11):117–148, 2003.

[15] Johann Birnick, Thomas Bläsius, Tobias Friedrich, Felix Naumann, Thorsten Papenbrock, and Martin Schirneck. Hitting set enumeration with partial information for unique column combination discovery. *Proceedings of the VLDB Endowment*, 13(11):2270 – 2283, 2020.

[16] Thomas Bläsius, Tobias Friedrich, and Martin Schirneck. The Parameterized Complexity of Dependency Detection in Relational Databases. In Jiong Guo and Danny Hermelin, editors, *11th International Symposium on Parameterized and Exact Computation (IPEC 2016)*, volume 63 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 6:1–6:13, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. ISBN 978-3-95977-023-1. doi: 10.4230/LIPIcs.IPEC.2016.6. URL `http://drops.dagstuhl.de/opus/volltexte/2017/6920`.

[17] Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical privacy: the sulq framework. In *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 128–138, 2005.

[18] Paola Bonizzoni, Gianluca Della Vedova, Riccardo Dondi, and Yuri Pirola. Parameterized complexity of k-anonymity: hardness and tractability. *Journal of combinatorial optimization*, 26:19–43, 2013.

[19] Joppe W Bos, Wouter Castryck, Ilia Iliashenko, and Frederik Vercauteren. Privacy-friendly forecasting for the smart grid using homomorphic encryption and the group method of data handling. In *International Conference on Cryptology in Africa*, pages 184–201. Springer, 2017.

[20] Stefano Braghin, Aris Gkoulalas-Divanis, and Michael Wurst. Detecting quasi-identifiers in datasets, January 16 2018. US Patent 9,870,381.

[21] William L Briggs, Van Emden Henson, and Steve F McCormick. *A multigrid tutorial*. SIAM, 2000.

[22] Coen Bron and Joep Kerbosch. Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577, 1973.

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

[24] Sergiu Carpov, Thanh Hai Nguyen, Renaud Sirdey, Gianpiero Constantino, and Fabio Martinelli. Practical privacy-preserving medical diagnosis using homomorphic encryption. In *2016 IEEE 9th International Conference on Cloud Computing (CLOUD)*, pages 593–599. IEEE, 2016.

[25] Jack Carr. *Applications of centre manifold theory*, volume 35. Springer Science & Business Media, 2012.

[26] Frédéric Cazals and Chinmay Karande. A note on the problem of reporting maximal cliques. *Theoretical Computer Science*, 407(1-3):564–568, 2008.

[27] Phillip J Chase. Algorithm 382: combinations of m out of n objects [g6]. *Communications of the ACM*, 13(6):368, 1970.

[28] David M Chickering, Dan Geiger, David Heckerman, et al. Learning bayesian networks is np-hard. Technical report, Technical Report MSR-TR-94-17, Microsoft Research, 1994.

[29] Young-Chul Chung, Ya-Ri Lee, Jung-Sook Kim, and Ho-Kyun Park. A study on personal health information de-identification status for big data. *Advanced Science and Technology Letters*, 136:54–58, 2016.

[30] Edmund Clarke, Armin Biere, Richard Raimi, and Yunshan Zhu. Bounded model checking using satisfiability solving. *Formal methods in system design*, 19(1):7–34, 2001.

[31] Chris Clifton and Tamir Tassa. On syntactic anonymity and differential privacy. In *2013 IEEE 29th International Conference on Data Engineering Workshops (ICDEW)*, pages 88–93. IEEE, 2013.

[32] European Commission. General data protection regulation, Aug 2019. URL `https://ec.europa.eu/info/law/law-topic/data-protection/reform/what-does-general-data-protection-regulation-gdpr-govern_en`.

[33] United States. Privacy Protection Study Commission. *Personal Privacy in an Information Society: The Report of the Privacy Protection Study Commission*, volume 2. The Commission, 1977.

[34] Paul Dagum and Michael Luby. Approximating probabilistic inference in bayesian belief networks is np-hard. *Artificial intelligence*, 60(1):141–153, 1993.

[35] Ivan Damgård, Valerio Pastro, Nigel Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In *Annual Cryptology Conference*, pages 643–662. Springer, 2012.

[36] HEER Daniel and Jannik Podlesny. Process for the user-related answering of customer inquiries in data networks, July 24 2018. US Patent 10,033,705.

[37] Fida Kamal Dankar and Khaled El Emam. The application of differential privacy to health data. In *Proceedings of the 2012 Joint EDBT/ICDT Workshops*, pages 158–166, 2012.

[38] Fida Kamal Dankar and Khaled El Emam. Practicing differential privacy in health care: A review. *Trans. Data Priv.*, 6(1):35–67, 2013.

[39] Jessica Davis. Health data, medical documents exposed by labcorp website error, Jan 2020. URL `https://healthitsecurity.com/news/health-data-medical-documents-exposed-by-labcorp-website-error`.

[40] Yves-Alexandre De Montjoye, César A Hidalgo, Michel Verleysen, and Vincent D Blondel. Unique in the crowd: The privacy bounds of human mobility. *Scientific reports*, 3:1376, 2013.

[41] Yves-Alexandre De Montjoye, Laura Radaelli, Vivek Kumar Singh, et al. Unique in the shopping mall: On the reidentifiability of credit card metadata. *Science*, 347(6221):536–539, 2015.

[42] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. 2004.

[43] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[44] Jega Anish Dev. Bitcoin mining acceleration and performance quantification. In *2014 IEEE 27th Canadian conference on electrical and computer engineering (CCECE)*, pages 1–6. IEEE, 2014.

[45] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-*

SIGART symposium on Principles of database systems, pages 202–210, 2003.

[46] Riccardo Dondi, Giancarlo Mauri, and Italo Zoppis. The l-diversity problem: Tractability and approximability. *Theoretical Computer Science*, 511: 159–171, 2013.

[47] Rodney G Downey and Michael R Fellows. *Fundamentals of parameterized complexity*, volume 4. Springer, 2013.

[48] Rodney G Downey and Michael Ralph Fellows. *Parameterized complexity*. Springer Science & Business Media, 2012.

[49] Cynthia Dwork. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*, pages 1–19. Springer, 2008.

[50] Cynthia Dwork. The differential privacy frontier. In *Theory of Cryptography Conference*, pages 496–502. Springer, 2009.

[51] Cynthia Dwork. Differential privacy. In *Encyclopedia of Cryptography and Security*, pages 338–340. Springer, 2011.

[52] Cynthia Dwork and Kobbi Nissim. Privacy-preserving datamining on vertically partitioned databases. In *Annual International Cryptology Conference*, pages 528–544. Springer, 2004.

[53] Cynthia Dwork and Adam Smith. Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality*, 1(2), 2010.

[54] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*, pages 265–284. Springer, 2006.

[55] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil Vadhan. On the complexity of differentially private data release: Efficient algorithms and hardness results. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 381–390, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-506-2. doi: 10.1145/1536414.1536467. URL `http://doi.acm.org/10.1145/1536414.1536467`.

[56] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[57] Bradley Efron. Bayes' theorem in the 21st century. *Science*, 340(6137): 1177–1178, 2013.

[58] Khaled El Emam. Methods for the de-identification of electronic health records for genomic research. *Genome medicine*, 3(4):25, 2011.

[59] David Eppstein, Maarten Löffler, and Darren Strash. Listing all maximal cliques in large sparse real-world graphs. *Journal of Experimental Algorithmics (JEA)*, 18:3–1, 2013.

[60] Zekeriya Erkin, Thijs Veugen, Tomas Toft, and Reginald L Lagendijk. Generating private recommendations efficiently using homomorphic encryption and data packing. *IEEE transactions on information forensics and security*, 7(3):1053–1066, 2012.

[61] European Commission. Opinion 05/2014 on anonymisation techniques, Apr 2014. URL `https://www.pdpjournals.com/docs/88197.pdf`.

[62] Benjamin Feldmann. Distributed unique column combinations discovery.

[63] Donald F Ferguson, Leonidas Georgiadis, and Christos N Nikolaou. Workload manager for achieving transaction class response time goals in a multiprocessing system, April 2 1996. US Patent 5,504,894.

[64] Stephen E Fienberg and Jiashun Jin. Privacy-preserving data sharing in high dimensional regression and classification settings. *Journal of Privacy and Confidentiality*, 4(1):10, 2012.

[65] Caroline Fontaine and Fabien Galand. A survey of homomorphic encryption for nonspecialists. *EURASIP Journal on Information Security*, 2007: 1–10, 2007.

[66] Office for Civil Rights. Methods for de-identification of phi, Nov 2015. URL `https://www.hhs.gov/hipaa/for-professionals/privacy/special-topics/de-identification/index.html`.

[67] Feten Ben Fredj, Nadira Lammari, and Isabelle Comyn-Wattiau. Abstracting anonymization techniques: A prerequisite for selecting a generalization algorithm. *Procedia Computer Science*, 60:206–215, 2015.

[68] Scott R Fulton, Paul E Ciesielski, and Wayne H Schubert. Multigrid methods for elliptic problems: A review. *Monthly Weather Review*, 114 (5):943–959, 1986.

[69] Benjamin CM Fung, Thomas Trojer, Patrick CK Hung, Li Xiong, Khalil Al-Hussaeni, and Rachida Dssouli. Service-oriented architecture for high-dimensional private data mashup. *IEEE Transactions on Services Computing*, 5(3):373–386, 2011.

[70] Flavio D Garcia and Bart Jacobs. Privacy-friendly energy-metering via homomorphic encryption. In *International Workshop on Security and Trust Management*, pages 226–238. Springer, 2010.

[71] Craig Gentry and Dan Boneh. *A fully homomorphic encryption scheme*, volume 20. Stanford university Stanford, 2009.

[72] 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, pages 758–769. VLDB Endowment, 2007.

[73] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. Universally utility-maximizing privacy mechanisms. *SIAM Journal on Computing*, 41 (6):1673–1693, 2012.

[74] Carla P Gomes, Henry Kautz, Ashish Sabharwal, and Bart Selman. Satisfiability solvers. *Foundations of Artificial Intelligence*, 3:89–134, 2008.

[75] US California Government. Us california consumer privacy act. URL `https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=201720180AB375`.

[76] Megan M Griffin and Trisha D Steinbrecher. Large-scale datasets in special education research. In *International Review of Research in Developmental Disabilities*, volume 45, pages 155–183. Elsevier, 2013.

[77] A Gutmann, J Wagner, Y Ali, AL Allen, JD Arras, BF Atkinson, NA Farahany, AG Garza, C Grady, SL Hauser, et al. Privacy and progress in whole genome sequencing. *Presidential Committee for the Study of Bioethical*, (2012), 2012.

[78] Shupeng Han, Xiangrui Cai, Chao Wang, Haiwei Zhang, and Yanlong Wen. Discovery of unique column combinations with hadoop. In *Asia-Pacific Web Conference*, pages 533–541. Springer, 2014.

[79] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[80] Arvid Heise, Jorge-Arnulfo Quiané-Ruiz, Ziawasch Abedjan, Anja Jentzsch, and Felix Naumann. Scalable discovery of unique column combinations. *Proceedings of the VLDB Endowment*, 7(4):301–312, 2013.

[81] Oscar H Ibarra. Reversal-bounded multicounter machines and their decision problems. *Journal of the ACM (JACM)*, 25(1):116–133, 1978.

[82] Md Zahidul Islam and Ljiljana Brankovic. Privacy preserving data mining: A noise addition framework using a novel clustering technique. *Knowledge-Based Systems*, 24(8):1214–1223, 2011.

[83] Zhanglong Ji, Zachary C Lipton, and Charles Elkan. Differential privacy and machine learning: a survey and review. *arXiv preprint arXiv:1412.7584*, 2014.

[84] Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[85] Stephan Kessler, Jens Hoff, and Johann-Christoph Freytag. Sap hana goes private: from privacy research to privacy aware enterprise analytics. *Proceedings of the VLDB Endowment*, 12(12):1998–2009, 2019.

[86] Daniel Kifer and Ashwin Machanavajjhala. No free lunch in data privacy. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, SIGMOD '11, pages 193–204, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0661-4. doi: 10.1145/1989323.1989345. URL `http://doi.acm.org/10.1145/1989323.1989345`.

[87] Ovunc Kocabas and Tolga Soyata. Towards privacy-preserving medical cloud computing using homomorphic encryption. In *Virtual and Mobile Healthcare: Breakthroughs in Research and Practice*, pages 93–125. IGI Global, 2020.

[88] Ovunc Kocabas, Tolga Soyata, Jean-Philippe Couderc, Mehmet Aktas, Jean Xia, and Michael Huang. Assessment of cloud-based health monitoring using homomorphic encryption. In *2013 IEEE 31st International Conference on Computer Design (ICCD)*, pages 443–446. IEEE, 2013.

[89] Florian Kohlmayer, Fabian Prasser, Claudia Eckert, and Klaus A Kuhn. A flexible approach to distributed data anonymization. *Journal of biomedical informatics*, 50:62–76, 2014.

[90] Matthijs R Koot, Michel Mandjes, Guido van't Noordende, Cees de Laat, et al. Efficient probabilistic estimation of quasi-identifier uniqueness. *Proceedings of NWO ICT. Open*, 2011.

[91] Fragkiskos Koufogiannis, Shuo Han, and George J Pappas. Optimality of the laplace mechanism in differential privacy. *arXiv preprint arXiv:1504.00065*, 2015.

[92] Clete A Kushida, Deborah A Nichols, Rik Jadrnicek, Ric Miller, James K Walsh, and Kara Griffin. Strategies for de-identification and anonymization of electronic health record data for use in multicenter research studies. *Medical care*, 50:S82–S101, 2012.

[93] Kristen LeFevre, David J DeWitt, and Raghu Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 49–60. ACM, 2005.

[94] Kristen LeFevre, David J. DeWitt, and Raghu Ramakrishnan. Mondrian multidimensional k-anonymity. In *Proceedings of the 22Nd International Conference on Data Engineering*, ICDE '06, pages 25–, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2570-9. doi: 10.1109/ICDE.2006.101. URL `http://dx.doi.org/10.1109/ICDE.2006.101`.

[95] Anja Lehmann. Scrambledb: Oblivious (chameleon) pseudonymization-as-a-service. *Proceedings on Privacy Enhancing Technologies*, 2019(3): 289–309, 2019.

[96] David Leoni. Non-interactive differential privacy: a survey. In *Proceedings of the First International Workshop on Open Data*, pages 40–52. ACM,

2012.

[97] Chao Li, Gerome Miklau, Michael Hay, Andrew McGregor, and Vibhor Rastogi. The matrix mechanism: optimizing linear counting queries under differential privacy. *The VLDB Journal*, 24(6):757–781, 2015. ISSN 0949-877X. doi: 10.1007/s00778-015-0398-x. URL `http://dx.doi.org/10.1007/s00778-015-0398-x`.

[98] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 106–115, April 2007. doi: 10.1109/ICDE.2007.367856.

[99] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 106–115, 2007.

[100] Ninghui Li, Min Lyu, Dong Su, and Weining Yang. Differential privacy: From theory to practice. *Synthesis Lectures on Information Security, Privacy, & Trust*, 8(4):1–138, 2016.

[101] Hongyu Liang and Hao Yuan. On the complexity of t-closeness anonymization and related problems. In *Database Systems for Advanced Applications: 18th International Conference, DASFAA 2013, Wuhan, China, April 22-25, 2013. Proceedings, Part I 18*, pages 331–345. Springer, 2013.

[102] Tong Lin and Hongbin Zha. Riemannian manifold learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):796–809, 2008.

[103] Fang Liu. Generalized gaussian mechanism for differential privacy. *arXiv preprint arXiv:1602.06028*, 2016.

[104] Kun Liu, Hillol Kargupta, and Jessica Ryan. Random projection-based multiplicative data perturbation for privacy preserving distributed data mining. *IEEE Transactions on knowledge and Data Engineering*, 18(1): 92–106, 2006.

[105] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1):3?es, March 2007. ISSN 1556-4681. doi: 10.1145/1217299.1217302. URL `https://doi.org/10.1145/1217299.1217302`.

[106] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3, 2007.

[107] Sharad Malik and Lintao Zhang. Boolean satisfiability from theoretical hardness to practical success. *Communications of the ACM*, 52(8):76–82, 2009.

[108] Joao Marques-Silva, Inês Lynce, and Sharad Malik. Conflict-driven clause learning sat solvers. In *Handbook of satisfiability*, pages 131–153. ios Press, 2009.

[109] Rohan Massey. *How The GDPR Will Impact Life Sciences And Health Care*. Law360 - Expert Analysis, February 2017.

[110] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on*, pages 94–103. IEEE, 2007.

[111] Albert R Meyer and Larry J Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *SWAT (FOCS)*, pages 125–129, 1972.

[112] Adam Meyerson and Ryan Williams. On the complexity of optimal k-anonymity. In *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 223–228. ACM, 2004.

[113] Noman Mohammed, Benjamin Fung, Patrick CK Hung, and Cheuk-Kwong Lee. Centralized and distributed anonymization for high-dimensional healthcare data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 4(4):18, 2010.

[114] John W Moon and Leo Moser. On cliques in graphs. *Israel journal of Mathematics*, 3(1):23–28, 1965.

[115] Ingo Müller, Peter Sanders, Arnaud Lacurie, Wolfgang Lehner, and Franz Färber. Cache-efficient aggregation: Hashing is sorting. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1123–1136, 2015.

[116] Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, pages 113–124, 2011.

[117] Arvind Narayanan and Vitaly Shmatikov. How to break anonymity of the netflix prize dataset. *CoRR*, abs/cs/0610105, 2006. URL `http://arxiv.org/abs/cs/0610105`.

[118] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 111–125. IEEE, 2008.

[119] Arvind Narayanan and Vitaly Shmatikov. Myths and fallacies of" personally identifiable information". *Communications of the ACM*, 53(6):24–26, 2010.

[120] J Jesu Vedha Nayahi and V Kavitha. Privacy and utility preserving data clustering for data anonymization and distribution on hadoop. *Future Generation Computer Systems*, 74:393–408, 2017.

[121] Richard E Neapolitan. *Probabilistic reasoning in expert systems: theory and algorithms*. CreateSpace Independent Publishing Platform, 2012.

[122] Mehmet Ercan Nergiz, Maurizio Atzori, and Chris Clifton. Hiding the presence of individuals from shared databases. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 665–676, 2007.

[123] John Nickolls and William J Dally. The gpu computing era. *IEEE micro*, 30(2):56–69, 2010.

[124] California Department of Health Care Services. List of hipaa identifiers, 2017. URL `http://www.dhcs.ca.gov/dataandstats/data/Pages/ListofHIPAAIdentifiers.aspx`.

[125] U.S. National Library of Medicine. Coronary artery disease, 2016. URL `https://medlineplus.gov/coronaryarterydisease.html`.

[126] Paul Ohm. Broken promises of privacy: Responding to the surprising failure of anonymization. *UCLA l. Rev.*, 57:1701, 2009.

[127] L. N. Olson and J. B. Schroder. PyAMG: Algebraic multigrid solvers in Python v4.0, 2018. URL `https://github.com/pyamg/pyamg`. Release 4.0.

[128] John D Owens, Mike Houston, David Luebke, Simon Green, John E Stone, and James C Phillips. Gpu computing. *Proceedings of the IEEE*, 96(5): 879–899, 2008.

[129] Thorsten Papenbrock and Felix Naumann. A hybrid approach for efficient unique column combination discovery. *Proc. der Fachtagung Business, Technologie und Web (BTW). GI, Bonn, Deutschland (accepted) Google Scholar*, 2017.

[130] Thorsten Papenbrock, Jens Ehrlich, Jannik Marten, Tommy Neubert, Jan-Peer Rudolph, Martin Schönberg, Jakob Zwiener, and Felix Naumann. Functional dependency discovery: An experimental evaluation of seven algorithms. *Proceedings of the VLDB Endowment*, 8(10):1082–1093, 2015.

[131] Payal V Parmar, Shraddha B Padhar, Shafika N Patel, Niyatee I Bhatt, and Rutvij H Jhaveri. Survey of various homomorphic encryption algorithms and schemes. *International Journal of Computer Applications*, 91 (8), 2014.

[132] William H Payne and Frederick M Ives. Combination generators. *ACM Transactions on Mathematical Software (TOMS)*, 5(2):163–172, 1979.

[133] Hasso Plattner. A common database approach for oltp and olap using an in-memory column database. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pages 1–2, 2009.

[134] Hasso Plattner and Alexander Zeier. *In-memory data management: technology and applications*. Springer Science & Business Media, 2012.

[135] Nikolai J. Podlesny. A Hybrid Approach for Anonymizing High-Dimensional Health Data. Master's thesis, Hasso-Plattner-Institute, University of Potsdam, Germany, 2017.

[136] Nikolai J. Podlesny. Synthetic genome data, 2020. URL `https://github.com/jaSunny/synthetic_genome_data`.

[137] Nikolai J Podlesny, Anne VDM Kayem, Stephan von Schorlemer, and Matthias Uflacker. Minimising information loss on anonymised high dimensional data with greedy in-memory processing. In *International Conference on Database and Expert Systems Applications*, pages 85–100. Springer, 2018.

[138] Nikolai J. Podlesny, Anne V.D.M. Kayem, Stephan von Schorlemer, and Matthias Uflacker. Minimising information loss on anonymized high dimensional data with greedy in-memory processing. In *International Conference on Database and Expert Systems Applications*. Springer, 2018.

[139] Nikolai J Podlesny, Anne VDM Kayem, and Christoph Meinel. Identifying data exposure across high-dimensional health data silos through bayesian networks optimised by multigrid and manifold. In *2019 IEEE 17th Intl Conf on Dependable, Autonomic and Secure Computing (DASC)*. IEEE, 2019.

[140] Nikolai J Podlesny, Anne VDM Kayem, and Christoph Meinel. Attribute compartmentation and greedy ucc discovery for high-dimensional data anonymization. In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy*, pages 109–119. ACM, 2019.

[141] Nikolai J Podlesny, Anne VDM Kayem, and Christoph Meinel. Towards identifying de-anonymisation risks in distributed health data silos. In *International Conference on Database and Expert Systems Applications*, pages 33–43. Springer, 2019.

[142] Nikolai J Podlesny, Anne VDM Kayem, Christoph Meinel, and Sven Jungmann. How data anonymisation techniques influence disease triage in digital health: A study on base rate neglect. In *Proceedings of the 9th International Conference on Digital Public Health*, pages 55–62, 2019.

[143] Nikolai J Podlesny, Anne VDM Kayem, and Christoph Meinel. A parallel quasi-identifier discovery scheme for dependable data anonymisation. In *Transactions on Large-Scale Data-and Knowledge-Centered Systems XLIV*. Springer, 2021.

[144] Nikolai J Podlesny, Anne VDM Kayem, and Christoph Meinel. Gpu accelerated bayesian inference for qid discovery in high-dimensional data. In *International conference on advanced information networking and applications (AINA)*. Springer, 2021.

[145] Nikolai J Podlesny, Anne VDM Kayem, and Christoph Meinel. A review of scaling genome sequencing data anonymisation. In *International Workshop on the Theories and Intricacies of Information Security problems*. Springer, 2021.

[146] Nikolai J. Podlesny, Anne V.D.M. Kayem, and Christoph Meinel. Cok: A survey of privacy challenges in relation to data meshes. In *International Conference on Database and Expert Systems Applications*. Springer, 2022.

[147] Huseyin Polat and Wenliang Du. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pages 625–628. IEEE, 2003.

[148] Jules Polonetsky, Omer Tene, and Kelsey Finch. Shades of gray: Seeing the full spectrum of practical data de-intentification. *Santa Clara L. Rev.*, 56:593, 2016.

[149] Victor Powell and Lewis Lehe. Markov chains, 2014. URL `http://www.dhcs.ca.gov/dataandstats/data/Pages/ListofHIPAAIdentifiers.aspx`.

[150] Ronald L Rivest, Len Adleman, Michael L Dertouzos, et al. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11): 169–180, 1978.

[151] Tony Romm. U.s. government issues stunning rebuke, historic $ billion fine against facebook for repeated privacy violations, Jul 2019. URL `https://www.washingtonpost.com/technology/2019/07/24/us-government-issues-stunning-rebuke-historic-billion-fine-against-facebook-repeated-privacy-violations/`.

[152] Ira S Rubinstein and Woodrow Hartzog. Anonymization and risk. *Wash. L. Rev.*, 91:703, 2016.

[153] Jorge Arnulfo Quiané Ruiz, Felix Naumann, and Ziawasch Abedjan. Datasets profiling tools, methods, and systems, June 11 2019. US Patent 10,318,388.

[154] IBRAHIM SABUNCU. Usa nov.2020 election 20 mil. tweets (with sentiment and party name labels) dataset, 2020. URL `https://dx.doi.org/10.21227/25te-j338`.

[155] Karem A Sakallah et al. Anatomy and empirical evaluation of modern sat solvers. *Bulletin of the EATCS*, (103):96–121, 2011.

[156] Milad Salem, Shayan Taheri, and Jiann-Shiun Yuan. Utilizing transfer learning and homomorphic encryption in a privacy preserving and secure biometric recognition system. *Computers*, 8(1):3, 2019.

[157] Pierangela Samarati and Latanya Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, Technical report, SRI International, 1998.

[158] Jason Sanders and Edward Kandrot. *CUDA by example: an introduction to general-purpose GPU programming*. Addison-Wesley Professional, 2010.

[159] Eric Schadt and Sastry Chilukuri. The role of big data in medicine, Nov 2015.

[160] Marcel-Paul Schützenberger. On an enumeration problem. *J. Combinatorial Theory*, 4:219–221, 1968.

[161] Wei-Tao Song, Bin Hu, and Xiu-Feng Zhao. Privacy protection of iot based on fully homomorphic encryption. *Wireless Communications and Mobile Computing*, 2018, 2018.

[162] Jordi Soria-Comas and Josep Domingo-Ferrer. Big data privacy: challenges to privacy principles and models. *Data Science and Engineering*, 1 (1):21–28, 2016.

[163] Klaus Stüben. An introduction to algebraic multigrid. *Multigrid*, pages 413–532, 2001.

[164] Jessica Su, Ansh Shukla, Sharad Goel, and Arvind Narayanan. De-anonymizing web browsing data with social networks. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1261–1269. International World Wide Web Conferences Steering Committee, 2017.

[165] Latanya Sweeney. Simple demographics often identify people uniquely. *Health (San Francisco)*, 671(2000):1–34, 2000.

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

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

[168] Tamir Tassa, Arnon Mazza, and Aristides Gionis. k-concealment: An alternative model of k-type anonymity. *Trans. Data Privacy*, 5(1):189–222, 2012.

[169] Michael Bedford Taylor. The evolution of bitcoin hardware. *Computer*, 50(9):58–66, 2017.

[170] Maha Tebaa, Saïd El Hajji, and Abdellatif El Ghazi. Homomorphic encryption applied to the cloud computing security. In *Proceedings of the World Congress on Engineering*, volume 1, pages 4–6, 2012.

[171] Manolis Terrovitis, Nikos Mamoulis, and Panos Kalnis. Privacy-preserving anonymization of set-valued data. *Proceedings of the VLDB Endowment*, 1(1):115–125, 2008.

[172] Etsuji Tomita, Akira Tanaka, and Haruhisa Takahashi. The worst-case time complexity for generating all maximal cliques and computational experiments. *Theoretical Computer Science*, 363(1):28–42, 2006.

[173] Ioannis Tsamardinos, Laura E Brown, and Constantin F Aliferis. The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning*, 65(1):31–78, 2006.

[174] Salil Vadhan. The complexity of differential privacy. In *Tutorials on the Foundations of Cryptography*, pages 347–450. Springer, 2017.

[175] Jaideep Vaidya and Chris Clifton. Privacy-preserving k-means clustering over vertically partitioned data. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 206–215. ACM, 2003.

[176] Jaideep Vaidya, Murat Kantarcıoğlu, and Chris Clifton. Privacy-preserving naive bayes classification. *The VLDB Journal—The International Journal on Very Large Data Bases*, 17(4):879–898, 2008.

[177] Petr Vaněk, Jan Mandel, and Marian Brezina. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. *Computing*, 56(3):179–196, 1996.

[178] P.J. Vessenes and R.B. Seidensticker. System and method for analyzing transactions in a distributed ledger, March 29 2016. URL `https://www.google.com/patents/US9298806`. US Patent 9,298,806.

[179] Jiahe Helen Wang, Qiang Huang, and David Jao. Privacy-preserving data aggregation using homomorphic encryption, December 21 2010. US Patent 7,856,100.

[180] Jing Wang, Zhenyue Zhang, and Hongyuan Zha. Adaptive manifold learning. In *Advances in neural information processing systems*, pages 1473–1480, 2005.

[181] Larry Wasserman and Shuheng Zhou. A statistical framework for differential privacy. *Journal of the American Statistical Association*, 105(489): 375–389, 2010.

[182] Hadley Wickham et al. The split-apply-combine strategy for data analysis. *Journal of Statistical Software*, 40(1):1–29, 2011.

[183] Hayden Wimmer and Loreen Powell. A comparison of the effects of k-anonymity on machine learning algorithms. In *Proceedings of the Confer-*

ence for Information Systems Applied Research ISSN, volume 2167, page 1508, 2014.

[184] Raymond Chi-Wing Wong, Ada Wai-Chee Fu, Ke Wang, and Jian Pei. Minimality attack in privacy preserving data publishing. In *Proceedings of the 33rd International Conference on Very Large Data Bases*, VLDB ?07, page 543?554. VLDB Endowment, 2007. ISBN 9781595936493.

[185] Raymond Chi-Wing Wong, Ada Wai-Chee Fu, Ke Wang, and Jian Pei. Anonymization-based attacks in privacy-preserving data publishing. *ACM Trans. Database Syst.*, 34(2), July 2009. ISSN 0362-5915. doi: 10.1145/1538909.1538910. URL `https://doi.org/10.1145/1538909.1538910`.

[186] Raymond Chi-Wing Wong, Ada Wai-Chee Fu, Ke Wang, Philip S. Yu, and Jian Pei. Can the utility of anonymized data be used for privacy breaches? *ACM Trans. Knowl. Discov. Data*, 5(3), August 2011. ISSN 1556-4681. doi: 10.1145/1993077.1993080. URL `https://doi.org/10.1145/1993077.1993080`.

[187] Wai Kit Wong, Nikos Mamoulis, and David Wai Lok Cheung. Non-homogeneous generalization in privacy preserving data publishing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 747–758. ACM, 2010.

[188] Xiaokui Xiao and Yufei Tao. Anatomy: Simple and effective privacy preservation. In *Proceedings of the 32nd international conference on Very large data bases*, pages 139–150. VLDB Endowment, 2006.

[189] Ping Xiong, Tian-Qing Zhu, and Xiao-Feng Wang. A survey on differential privacy and applications. *Jisuanji Xuebao/Chinese Journal of Computers*, 37(1):101–122, 2014.

[190] Yabo Xu, Benjamin CM Fung, Ke Wang, Ada WC Fu, and Jian Pei. Publishing sensitive transactions for itemset utility. In *2008 Eighth IEEE International Conference on Data Mining*, pages 1109–1114. IEEE, 2008.

[191] Yabo Xu, Ke Wang, Ada Wai-Chee Fu, and Philip S Yu. Anonymizing transaction databases for publication. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 767–775, 2008.

[192] Baichuan Zhang, Vachik Dave, Noman Mohammed, and Mohammad Al Hasan. Feature selection for classification under anonymity constraint. *arXiv preprint arXiv:1512.07158*, 2015.

[193] Xuyun Zhang, Chang Liu, Surya Nepal, and Jinjun Chen. An efficient quasi-identifier index based approach for privacy preservation over incremental data sets on cloud. *Journal of Computer and System Sciences*, 79 (5):542–555, 2013.

[194] Xuyun Zhang, Laurence T Yang, Chang Liu, and Jinjun Chen. A scalable two-phase top-down specialization approach for data anonymization using mapreduce on cloud. *IEEE Transactions on Parallel and Distributed Systems*, 25(2):363–373, 2014.

A

Dataset Characteristics

Table A.1: Exemplary list of the 5 distinct countries by count

Country	**Amount**
DE	850000
NO	50000
FR	50000
IT	50000
Other	19

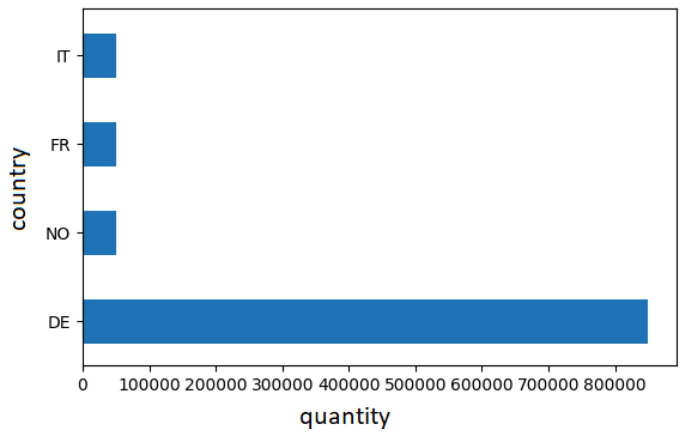


Fig. A.1: Data characteristics: Distribution of countries

Table A.2: Exemplary list of the 9 distinct blood types by count

BloodType	**Amount**
O+	365101
A+	263196
B+	225406
AB+	50263
O-	43200
A-	34516
B-	13881
AB-	4437
Other	19

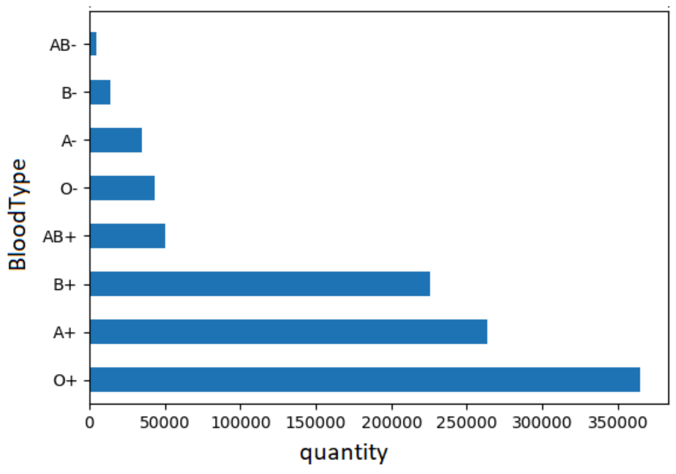


Fig. A.2: Data characteristics: Distribution of bloodtypes

Table A.3: Sampled list of the 83 distinct heights by count

Height	**Amount**
...	...
154	3523
165	27051
158	3598
177	18480
184	17448
171	34850
183	3475
161	19465
156	3616
...	...

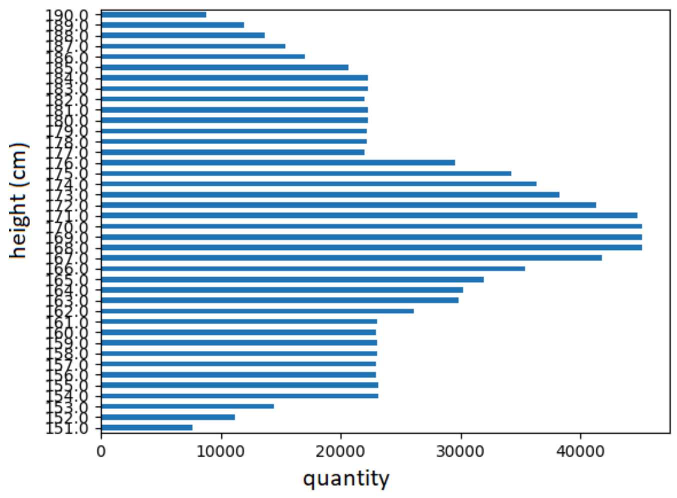


Fig. A.3: Data characteristics: Distribution of heights

Table A.4: Sampled list of the 1487 distinct weights by count

Weight	Amount
...	...
102.8	268
73.9	297
64.1	303
116.4	12
79.0	308
65.0	292
94.1	317
105.7	866
78.9	278
54.2	823
76.5	1626
...	...

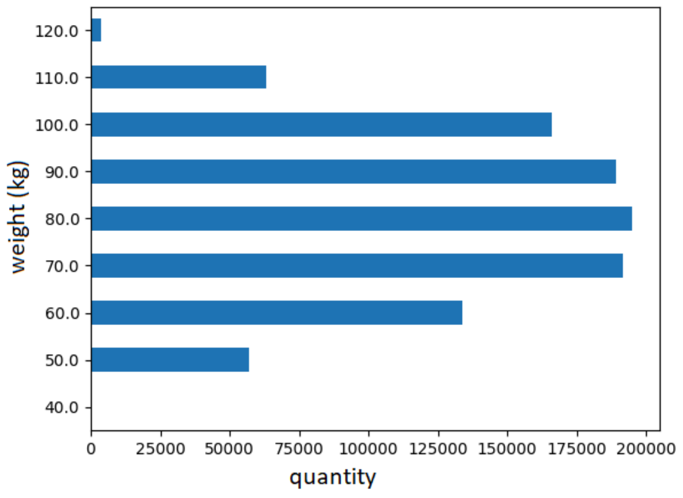


Fig. A.4: Data characteristics: Distribution of weights

Table A.5: Sampled list of the 14184 distinct available firstnames by count

Firstname	**Amount**
...	...
Robert	3755
Emin	58
Sabrien	5
Gayle	32
Deidre	2
Lincoln	201
Aletta	4
Davita	1
Maeve	56
Ishak	9
Andy	68
Erdal	4
Lodewijk	8
Thorben	3
Martine	384
...	...

Table A.6: Sampled list of the 31409 distinct available surnames by count

Surname	**Amount**
...	...
Koval	2
Main	7
Ølmheim	1
McNatt	4
Hindman	2
Wynyard	11
Røkenes	8
de Best	6
Mucha	2
Bischof	9
Laboissonnière	209
Knepp	1
Vervuurt	5
Domhof	7
Beyer	726
Troup	9
de Schipper	11
Terwilliger	2
Baert	4
...	...

Table A.7: Sampled list of the 23560 distinct available cities by count

City	**Amount**
...	...
Podresca	3
Scomigo	3
Sant'Elpidio Morico	5
Beano	5
Barbata	7
Attenkirchen	103
Munningen	69
San Giorgio Scarampi	3
Emmerting	59
Bühlertal	59
Pergine Valdarno	6
Vestenanova	2
Mörlen	50
Neulehe	64
Bidingen	50
Penzing	47
Capece Bax	4
Veppo	1
Fratta	4
...	...

Table A.8: Sampled list of the 156 distinct states by count

State	Amount
...	...
Como	665
Languedoc-Roussillon	1460
Ascoli Piceno	388
Picardie	896
Bretagne	1011
Biella	333
Trento	1170
Verona	765
Guadeloupe	714
Saarland	4801
Aosta	367
Genova	668
Piacenza	369
Hessen	22552
Nordrhein-Westfalen	69915
Crotone	125
Cremona	447
Gorizia	164
Limousin	540
Campobasso	312
Udine	690
Ragusa	99
Brindisi	115
Caserta	624
...	...

Table A.9: Sampled list of the 1031 distinct companies by count

Company	**Amount**
...	...
Block Distributors	955
Sunny Real Estate Investments	943
Buena Vista Garden Maintenance	936
Noodle Kidoodle	910
Sportswest	1006
The Network Chef	975
Liberty Wealth Planners	942
American Appliance	914
Wickes Furniture	987
The Flying Hippo	937
Paul Harris	1001
Earthworks Yard Maintenance	993
AJ Bayless	981
Liberty Wealth Planner	986
Endicott Johnson	935
Warner Brothers Studio Store	998
Matrix Architectural Service	974
...	...

Table A.10: Sampled list of the 113 distinct diseases by count

Disease	**Amount**
...	...
DOID:3121	6150
DOID:635	4315
DOID:2994	7073
DOID:12995	2987
DOID:10941	3615
DOID:175	6525
DOID:10534	11147
DOID:11239	28
DOID:1612	4309
DOID:11476	4366
DOID:13241	8092
DOID:0050156	6459
DOID:12930	7917
DOID:11949	1502
DOID:2596	6817
DOID:3083	13093
DOID:2174	2866
DOID:4606	5656
DOID:1192	812
DOID:1312	6419
DOID:4989	14179
...	...

Table A.11: Sampled list of the 1802 distinct SNPs by count

SNP	**Amount**
...	...
rs104894725	1323
rs10512437	23
rs104886099	195
rs104894664	434
rs104894488	563
rs104886415	385
rs104894757	342
rs10505128	26
rs10519774	3
rs104893692	482
rs104893683	516
rs104886112	224
rs10455872	649
rs104895085	197
rs104895472	313
rs104894810	254
rs104886235	256
rs10490924	12770
rs104886381	367
rs104893924	1973
rs104886300	311
rs104894229	6887
rs104893775	4139
rs10499271	101
...	...

B

Experimental Hardware Specifications

```
# cat /proc/cpuinfo
processor       : 159
vendor_id       : GenuineIntel
cpu family      : 6
model           : 85
model name      : Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz
stepping        : 4
microcode       : 0x2006a08
cpu MHz         : 1296.240
cache size      : 28160 KB
physical id     : 3
siblings        : 40
core id         : 26
cpu cores       : 20
apicid          : 245
initial apicid  : 245
fpu             : yes
fpu_exception   : yes
cpuid level     : 22
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic
    sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr
     sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm
    constant_tsc art arch_perfmon pebs bts rep_good nopl
    xtopology nonstop_tsc aperfmperf eagerfpu pni pclmulqdq
    dtes64 monitor ds_cpl vmx smx est tm2 ssse3 sdbg fma cx16
     xtpr pdcm pcid dca sse4_1 sse4_2 x2apic movbe popcnt
    tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm
    3dnowprefetch epb cat_l3 cdp_l3 invpcid_single intel_ppin
     intel_pt ssbd mba ibrs ibpb stibp tpr_shadow vnmi
    flexpriority ept vpid fsgsbase tsc_adjust bmi1 hle avx2
    smep bmi2 erms invpcid rtm cqm mpx rdt_a avx512f avx512dq
     rdseed adx smap clflushopt clwb avx512cd avx512bw
    avx512vl xsaveopt xsavec xgetbv1 cqm_llc cqm_occup_llc
    cqm_mbm_total cqm_mbm_local dtherm ida arat pln pts pku
    ospke md_clear spec_ctrl intel_stibp flush_l1d
```

```
# cat /proc/meminfo
MemTotal:        790944916 kB
MemFree:         317023780 kB
MemAvailable:    571409388 kB
Buffers:              5364 kB
Cached:          234214360 kB
SwapCached:          18072 kB
Active:          251551836 kB
Inactive:        167219424 kB
Active(anon):    183220292 kB
Inactive(anon):   1650304 kB
Active(file):    68331544 kB
Inactive(file): 165569120 kB
Unevictable:          8220 kB
Mlocked:              8244 kB
SwapTotal:       33411064 kB
SwapFree:        33352184 kB
Dirty:                2316 kB
Writeback:               0 kB
AnonPages:       184536632 kB
Mapped:           3919356 kB
Shmem:             312700 kB
Slab:            28574828 kB
SReclaimable:    21803960 kB
SUnreclaim:       6770868 kB
KernelStack:       310496 kB
PageTables:        869416 kB
CommitLimit:     428883520 kB
Committed_AS:    206315436 kB
VmallocTotal:    34359738367 kB
VmallocUsed:      4005912 kB
VmallocChunk:    33752440564 kB
Percpu:           1072128 kB
HardwareCorrupted:     0 kB
AnonHugePages:   156755968 kB
Hugepagesize:         2048 kB
DirectMap4k:     14077896 kB
DirectMap2M:     416495616 kB
DirectMap1G:     375390208 kB
```

```
# lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                160
On-line CPU(s) list:   0-159
Thread(s) per core:    2
Core(s) per socket:    20
Socket(s):             4
NUMA node(s):          4
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 85
Model name:            Intel(R) Xeon(R) Gold 6148 CPU @ 2.40
    GHz
Stepping:              4
CPU MHz:               1491.943
CPU max MHz:           3700.0000
CPU min MHz:           1000.0000
BogoMIPS:              4807.22
Virtualization:        VT-x
L1d cache:             32K
L1i cache:             32K
L2 cache:              1024K
L3 cache:              28160K
```

```
# hwinfo --short
cpu:
     Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz, 2400 MHz
graphics card:
         nVidia 3D controller
         Matrox VGA compatible controller
storage:
         Intel Lewisburg SSATA Controller [AHCI mode]
         Intel Lewisburg SATA Controller [AHCI mode]
         Dell PERC H730P Adapter
         Marvell 88SE9230 PCIe SATA 6Gb/s Controller
         QLogic Fibre Channel
network:
         QLogic Ethernet controller
         Dell iDRAC Virtual NIC USB Device
network interface:
eth0                 Ethernet network interface
lo                   Loopback network interface
disk:
/dev/sda             DELL PERC H730P Adp
/dev/sdb             DELLBOSS VD
/dev/sdc             Linux Virtual Floppy
```

(removed duplicate entries)

```
# cat /proc/version
Linux version 3.10.0-1127.19.1.el7.x86_64 (
    mockbuild@kbuilder.bsys.centos.org) (gcc version 4.8.5
    20150623 (Red Hat 4.8.5-39) (GCC) ) #1 SMP Tue Aug 25
    17:23:54 UTC 2020
```

```
# nvidia-smi
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 455.45.01    Driver Version: 455.45.01    CUDA Version: 11.1     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla V100-PCIE...  Off  | 00000000:9B:00.0 Off |                    0 |
| N/A   31C    P0    26W / 250W |      0MiB / 32510MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
# sudo lshw -C display
 *-display
       description: VGA compatible controller
       product: Integrated Matrox G200eW3 Graphics
           Controller
       vendor: Matrox Electronics Systems Ltd.
       physical id: 0
       bus info: pci@0000:03:00.0
       version: 04
       width: 32 bits
       clock: 66MHz
       capabilities: vga_controller bus_master cap_list rom
       configuration: driver=mgag200 latency=64 maxlatency
           =32 mingnt=16
       resources: irq:16 memory:91000000-91ffffff memory
           :92808000-9280bfff memory:92000000-927fffff
  *-display
       description: 3D controller
       product: GV100GL [Tesla V100 PCIe 32GB]
       vendor: NVIDIA Corporation
       physical id: 0
       bus info: pci@0000:25:00.0
       version: a1
       width: 64 bits
       clock: 33MHz
       capabilities: bus_master cap_list
       configuration: driver=nvidia latency=0
       resources: iomemory:38200-381ff iomemory:38280-3827f
           irq:723 memory:9e000000-9effffff memory
           :382000000000-3827ffffffff memory
           :382800000000-382801ffffff
```

(removed duplicate entries)

C

List of Publications

The work that addresses selected research activities have been published in the following conferences and journals ordered by their release date.

- Nikolai J Podlesny, Anne VDM Kayem, Stephan von Schorlemer, and Matthias Uflacker. Minimising information loss on anonymised high dimensional data with greedy in-memory processing. In *International Conference on Database and Expert Systems Applications*, pages 85–100. Springer, 2018
- Nikolai J Podlesny, Anne VDM Kayem, and Christoph Meinel. Attribute compartmentation and greedy ucc discovery for high-dimensional data anonymization. In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy*, pages 109–119. ACM, 2019
- Nikolai J Podlesny, Anne VDM Kayem, and Christoph Meinel. Towards identifying de-anonymisation risks in distributed health data silos. In *International Conference on Database and Expert Systems Applications*, pages 33–43. Springer, 2019
- Nikolai J Podlesny, Anne VDM Kayem, and Christoph Meinel. Identifying data exposure across high-dimensional health data silos through bayesian networks optimised by multigrid and manifold. In *2019 IEEE 17th Intl Conf on Dependable, Autonomic and Secure Computing (DASC)*. IEEE, 2019
- Nikolai J Podlesny, Anne VDM Kayem, Christoph Meinel, and Sven Jungmann. How data anonymisation techniques influence disease triage in digital health: A study on base rate neglect. In *Proceedings of the 9th International Conference on Digital Public Health*, pages 55–62, 2019
- Nikolai J Podlesny, Anne VDM Kayem, and Christoph Meinel. Gpu accelerated bayesian inference for qid discovery in high-dimensional data. In

International conference on advanced information networking and applications (AINA). Springer, 2021

- Nikolai J Podlesny, Anne VDM Kayem, and Christoph Meinel. A review of scaling genome sequencing data anonymisation. In *International Workshop on the Theories and Intricacies of Information Security problems*. Springer, 2021
- Nikolai J Podlesny, Anne VDM Kayem, and Christoph Meinel. A parallel quasi-identifier discovery scheme for dependable data anonymisation. In *Transactions on Large-Scale Data-and Knowledge-Centered Systems XLIV*. Springer, 2021
- Nikolai J. Podlesny, Anne V.D.M. Kayem, and Christoph Meinel. Cok: A survey of privacy challenges in relation to data meshes. In *International Conference on Database and Expert Systems Applications*. Springer, 2022

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except as declared in the Preface and specified in the text. It is not substantially the same as any that I have submitted, or am concurrently submitting, for a degree or diploma or other qualification at the University of Potsdam or any other University or similar institution except as declared in the Preface and specified in the text.

Throughout the years selected contributions of the research activities have been made available to the research community and submitted to conferences and journals. These publications are listed in Appendix C.

Nikolai Jannik Podlesny
Potsdam, 02 July 2021