COCOA: A Synthetic Data Generator for Testing Anonymization Techniques

Vanessa Ayala-Rivera, A. Omar Portillo-Dominguez,
Liam Murphy, and Christina Thorpe

Lero@UCD, School of Computer Science, University College Dublin, Dublin, Ireland
vanessa.ayala-rivera@ucdconnect.ie,
{andres.portillodominguez,liam.murphy,christina.thorpe}@ucd.ie

Abstract. Conducting extensive testing of anonymization techniques is critical to assess their robustness and identify the scenarios where they are most suitable. However, the access to real microdata is highly restricted and the one that is publicly-available is usually anonymized or aggregated; hence, reducing its value for testing purposes. In this paper, we present a framework (COCOA) for the generation of realistic synthetic microdata that allows to define multi-attribute relationships in order to preserve the functional dependencies of the data. We prove how COCOA is useful to strengthen the testing of anonymization techniques by broadening the number and diversity of the test scenarios. Results also show how COCOA is practical to generate large datasets.

1 Introduction

The increasing availability of microdata has attracted the interest of organizations to collect and share this data for mining purposes. However, current legislation requires personal data to be protected from inappropriate use of disclosure. Anonymization techniques help to disseminate data in a safe manner while preserving enough utility for its reuse. For this reason, a plethora of methods for anonymizing microdata exists in the literature. Each of these techniques claims a particular superiority over the others (e.g., improving data utility or computational resources). However, their performance can vary when they are tested with different datasets [13]. This fact makes difficult for data controllers to generalize the conclusions of the performance evaluation to decide which algorithm is best suited for their requirements.

All performance evaluations are limited in one way or the other (due to time/effort/cost constraints). A common limitation is the number and the diversity of the datasets used, as the process to obtain good quality datasets can be burden and time consuming. For example, the access to real microdata is highly restricted to protect the privacy of individuals. Agencies grant access only for certain period and once this expires, the provided files must be destroyed [2, 4, 10] which does not always allow for reproducibility of experimental results.

The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-319-45381-1_13

Consequently, researchers often use real datasets that are publicly-available or synthetic data generated in an ad-hoc manner [20]. Both solutions are to some extend biased as they are constrained by an specific data distribution: either a real one which consists of a single scenario (e.g., demographics of a single country) or an ideal synthetic data which may never be found in the real world. Moreover, the applicability of most real datasets is often limited for testing anonymization techniques as datasets available in research are usually aggregated and pre-anonymized to protect the privacy of people (which is exactly the use case of privacy-preserving methods). Thus this data lacks of enough diversity to simulate attacking scenarios or to evaluate the robustness of the proposed techniques (e.g., data sparsity and outliers). For these reasons, synthetic data is a valuable resource for conducting testing in multiple areas, as it allows to manipulate the data to meet specific characteristics that are not found in the real data but still need to be considered for testing (hypothetical future scenarios). To be adequate substitutes for real data, the functional dependencies of the data need to be preserved.

To tackle these issues, our research work has centered on developing techniques to create realistic synthetic datasets applicable to the privacy domain. The aim has been to help researchers and practitioners (hereinafter referred as users) to improve the testing of anonymization techniques by decreasing the effort and expertise needed to create useful datasets in order to be able to perform more robust testing. As a first step in that direction, in our previous work [12] we described the process followed to mimic the distribution of the aggregated statistics from the 2011 Irish Census. The work proposed in this paper builds on top of that work by presenting a framework (COCOA) that facilitates the creation of realistic datasets in the privacy domain. Internally, COCOA leverages on a set of supported domains to automatically generate datasets of different characteristics. The contributions of this paper are the following:

1. A framework (COCOA) to generate realistic synthetic datasets (at record-level) that allows to define multi-attribute relationships in order to preserve the functional dependencies of the data.
2. A comprehensive practical evaluation of COCOA, consisting of a prototype and a set of experiments to assess it in terms of the benefits it brings to the testing of anonymization techniques as well as its costs.
3. Three sets of datasets (publicly available [1]), each one composed of 72 different datasets (based on real data and offering diverse data distributions and sizes) which can be useful for the research community to perform more comprehensive validations in the data privacy domain.

2 Related Work

Numerous approaches have been proposed to generate synthetic data. Some of those works are general-purpose synthetic data generators [17,24]. That is, they do not target a specific application area. However, this generality may reduce the accuracy in the results when specific features are required. Depending on the domain, it is important to preserve certain characteristics in the generated datasets to make it realistic and thus, suitable for their intended use. In the data privacy community, synthetic data generation is used as a strategy for disseminating data while ensuring confidentiality. The work in this research area focuses on generating synthetic populations that reproduce certain statistical properties of the original data. This kind of approaches was firstly introduced in [26]. Afterward, many techniques and tools have been proposed for generating synthetic datasets and populations [7, 8, 23]. Similarly, other research efforts have focused on evaluating the privacy of synthetic methods and avoid the risk of disclosure [18, 22]. In our work, we focus on generating synthetic microdata, not as a technique of data protection, but as a mechanism to improve the testing of anonymization techniques. Furthermore, our solution focuses on categorical data due to its relevance (e.g., a valuable asset of data mining) and because methods for numerical data have been well studied in the literature.

3 COCOA: A Synthetic Data Generator

Here we provide the context of our solution, discuss its internal workings, and describe the generated datasets.

3.1 Overview

The goal of this research work has been to develop a framework for synthetically generating datasets (COCOA) that can be used to diversify the set of characteristics available in microdata. This strategy would help researchers to improve the testing of anonymization techniques. In Fig. 1, we depict the conceptual view of our solution. It can be seen how COCOA follows an iterative process (see Section 3.2) to build a set of datasets based on the *information base* provided by the user. The information base is composed of all the input parameters required by the chosen dataset domain (e.g., dataset size).

The key element of COCOA is its *domain base*, which encapsulates the expert knowledge of the supported business domains (e.g., census, healthcare, finance). This element allows COCOA to be easily extensible and capable of incorporating multiple business cases (even for the

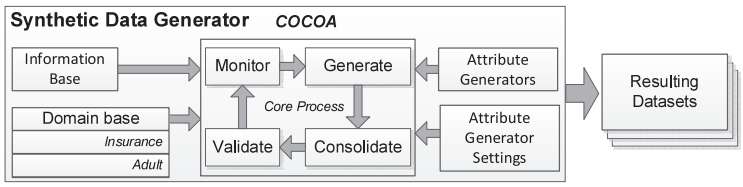

Fig. 1. COCOA - Conceptual View.

same domain e.g., Irish census, USA census), which might be suitable to different test scenarios. In this context, a *domain* defines the rules and constraints required to generate a dataset that preserves the functional dependencies of a business case. Each domain is characterized by a name, a set of attributes and their corresponding attribute generators.

To generate data, the domains make use of the available set of *attribute generators*. These elements are supporting logic which offer miscellaneous strategies to generate the values of attributes. For example, a generator might focus on diversifying the data in an attribute by fitting it into a data distribution (e.g., normal distribution). In this example, several generators can be combined to offer different data distributions (as the appropriate distribution might vary depending on the usage scenario). In case an attribute generator requires any particular settings to work properly (e.g., the default values for its applicable parameters), this information (e.g., mean and variance for a normal distribution) can also be captured by the framework (as an *attribute generator setting*).

3.2 Core Process

From a workflow perspective, COCOA has a core process (depicted in Fig. 2a). It requires three user inputs related to desired characteristics of the resulting dataset: the domain (selected among the ones available in the framework), the size, and the dataset name. An additional optional user input is the set of specific parameters that a domain might need to initialize a dataset. As an initial step, the process sets all the configured input parameters and generates an empty dataset for the chosen domain. Next the loop specified in the monitor, generate, consolidate and validate phases is performed: First, the size of the new dataset (number of tuples) is monitored. This check is done in order to determine when the dataset has reached the target size and finish the process. Next, the applicable generators to create the attributes' data are retrieved from the knowledge database (as they will depend on the domain). Then, their relationships are identified in order to sort the generators and execute them in the correct order, so that the proper chain of functional dependencies is executed (see Section 3.4). Once there are new values for all the attributes, a new

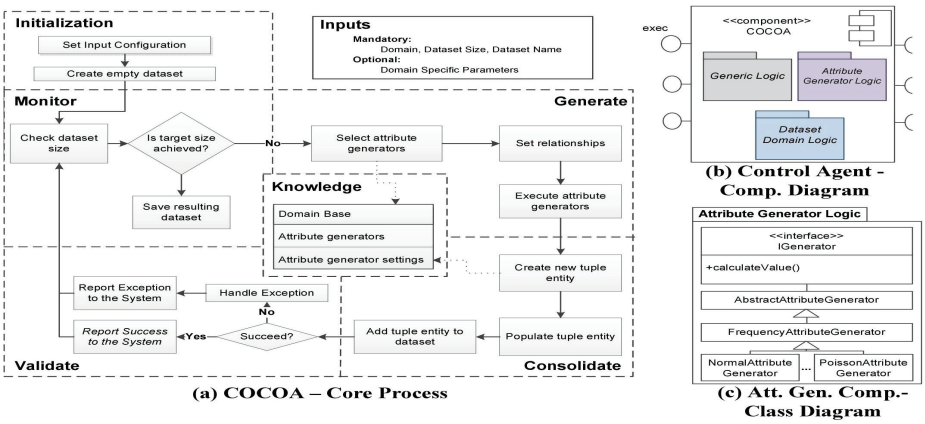

Fig. 2. COCOA Architecture

entity object is created to represent the new tuple. Next, its attributes are populated with the new values and the tuple entity is added to the dataset. Moreover, any exceptions are internally handled and reported. This core process continues iteratively until the new dataset has been fully generated. As a final step, the dataset is saved to disk.

3.3 Architecture

COCOA is complemented by the architecture presented in the component diagram [9] of Fig. 2b. COCOA is composed of three main components: The generic component contains the control logic and all supporting functionality which is independent of the supported domains and attribute generators (e.g., the monitor and validate phases of the core process). Regarding the logic that interfaces with the domains, it needs to be customized per domain (by defining a tuple entity and a dataset generator). Similar case with the supported attribute generators. Therefore, these two logics are encapsulated in their respective components to minimize the required code changes. To complement this design strategy, the components are only accessed through interfaces. This is exemplified in Fig. 2c, which presents the high-level structure of the attribute generator component. It contains a main interface *IGenerator* to expose all required actions and an abstract class for all the common functionality. This hierarchy can then be extended to support specific attribute generators.

3.4 Attribute Generators

As discussed in Section 3.1, attribute generators are supporting logic which offer miscellaneous strategies to generate the values of attributes. Among the alternative choices to develop attribute generators for COCOA, we have initially concentrated on implementing the following three:

Distribution-based generator. This type of generator produces independent data, so it is applicable for attributes whose values do not depend on others. In this case, the strategy used to diversify the data is to mimic a probability distribution. This generator can also be a good fit for data that comes from a previously consolidated real data source (like most of the datasets currently available). This is because, the objective in those cases is commonly to diversify the frequencies of the existing values without modifying the actual values or the cardinality of the attribute. A distribution-based generator requires two parameters: a distribution (which will be used to re-distribute the frequencies of the values) and a sorting strategy (which will be used to order the candidate values before applying the distribution). For the initial version of COCOA, 11 commonly-used data distributions are supported: normal, beta, chi, chi square, exponential (exp), gamma, geometric, logarithmic (log), poisson, t-student (Tstu), and uniform (uni) [29]. An additional supported distribution is the "original" one. When it is used, the generator only mirrors the given input distribution. This is useful for generating new datasets (of different sizes) for existing domains (e.g., adult and german credit) without modifying the original distribution present in the real data. Regarding the sorting strategies, COCOA currently supports three: alpha-numeric, reverse alpha-numeric, and no-sort. The usage of alpha-numeric and reverse alpha-numeric allows a user not only to logically sort the categorical values, but also helps to further diversify the tested behaviors by mixing the supported sorts and distributions. For instance, the usage of an alpha-numeric sorting and an exponential distribution can generate a J-Shaped distribution, while a reverse J-Shaped distribution can be easily generated by switching to the reverse alpha-numeric sorting. Finally, the no-sort strategy respects the original order of the data (following the same line of thought as the original distribution previously discussed).

Attribute-based generator. This type of generator produces dependent data. It is applicable to cases when there are functional dependencies that need to be preserved between attributes (so that the generated data can be realistic) as well as cases when an attribute needs to be derived from others. For instance, consider a dataset with information about individuals owned by an insurance company. The aim is to derive a class attribute that identifies groups with a higher risk of having an accident. One way to derive this class (e.g., low, medium, high) would be to use the values from attributes such as age, occupation, and hobbies. The parameters required by this type of generator might vary, but normally they will take most of their required input information from other attributes. As discussed

in Section 3.2, COCOA executes the generators in such an order that an attribute-based generator has available its required information (i.e., the attributes it depends on) before its execution.

Distribution and attribute-based generator. This type of generator also produces dependent data. This is because it is a hybrid of the previous two generators. It is useful to capture more complex relationships where the generation of a value is influenced not only by a frequency distribution, but also by the value of one or more attributes. For instance, the place where an activity is practiced does not only depend on the performed activity but also on a certain distribution. For example, based on historical information, soccer is mostly practiced on outside fields, and in less degree, in other places such as an indoor facility or beach. Another example is the salary of a person, which is influenced by multiple factors such as her occupation and years of work experience. This kind of relationships can be easily captured in COCOA by this type of generator.

3.5 Supported Domains

The following sections describe the domains currently supported by COCOA. The datasets generated for all the domains are publicly available [1].

Irish census. This domain, initially presented on [12], is composed of 9 attributes belonging to the Irish Census 2011. Appendix A lists the attributes and their generators. It is worth noticing how different types of generators were used to capture the relationships among the data.

Insurance domain. To further exemplify the capabilities of COCOA and its different attribute generators, we have designed the insurance domain. It represents information of interest to an insurance company carrying out a risk assessment on potential clients. This domain is based on real information obtained from the Payscale USA website [6]. The list of attributes, and the type of generator used for generating their data, are shown in appendix A. It is worth highlighting the extensive usage of different attribute generators to retain the realistic characteristics of the generated data. For example, the gender assigned to a tuple depends on a person's occupation (e.g., nursing is overwhelmingly female).

Adult domain. This domain is based on the Adult census dataset from the UCI Machine Learning Repository [21], which has become one of the most widely-used benchmarks in the privacy domain. This domain is composed of 9 socio-economic demographic attributes (e.g., occupation, education, and salary class). To mimic the values of this dataset, this domain exclusively leverages on distribution-based data generators.

German credit domain. This domain is based on the German credit dataset from the UCI Machine Learning Repository [21]. This domain is

composed of 20 credit-related attributes (e.g., employment, purpose of credit, credit class). To mimic the values of this dataset, this domain exclusively leverages on distribution-based data generators.

4 Experimental Evaluation

Here we present the experiments performed to assess the benefits and costs of using COCOA. Firstly, we evaluated how well COCOA achieved its objective of generating diverse datasets within a domain. Secondly, we assessed how useful the resulting datasets are to strengthen the testing of an anonymization algorithm. Thirdly, we evaluated COCOA's costs.

4.1 Experimental Setup

Here we present the developed prototype, the test environment and the parameters that defined the evaluated experimental configurations. We also describe the evaluation criteria used in our experiments.

Prototype: Our current prototype supports the domains discussed in Section 3.5, and the three types of attribute generators discussed in Section 3.4. To simplify the configuration of the distribution-based attributes, as well as to exemplify the benefits of using attribute generator settings, representative default values were configured for the parameters applicable to each distribution (e.g., mean and standard deviation for the normal distribution). From a technical perspective, we built our prototype on top of the Benerator tool [3]. This solution was chosen because it is open source and developed in Java, characteristics which facilitated its integration with other used libraries (e.g., the OpenForecast library [5], which is used for all statistical calculations). Developing the prototype in Java also makes our solution highly portable, as there are Java Virtual Machines (JVM) available for most contemporary operating systems.

Environment: All experiments were performed in an isolated test environment using a machine with an Intel Core i7-4702HQ CPU at 2.20Ghz, 8 GB of RAM and 450 GB of HD; running 64-bit Windows 8.1 Professional Edition, and Hotspot JVM 1.7.0_67 with a 1 GB heap.

Configurations: As evaluation data, we used the insurance, adult and german credit domains. This was done with the aim of diversifying the evaluated domains and test the generality of the benefits and costs of using COCOA. For the adult and german datasets, the richest attribute (in terms of diversity of categorical values) was used as quasi-identifier (QID) for the anonymization and for the data generation: occupation for adult, and purpose for german credit. In contrast, as the insurance dataset offers a broader set of interesting categorical attributes, four of them were

selected: occupation (occ), place of work, activity (act) and place of activity. For each domain, 72 different datasets were created. This was achieved by varying the frequencies of the QIDs (by using the 12 distributions discussed in Section 3.4) and generating 6 different dataset sizes (5K, 10K, 20K, 30K, 50K and 100K). As anonymization settings, we selected a popular greedy multidimensional algorithm called Mondrian [20] (from the UTD Anonymization Toolbox [11]). It partitions the domain space recursively into regions that contain at least k records (k-anonymity [27, 28]). We tested different levels of privacy, varying the k-values $\in$ [2..100].

Evaluation Criteria: To evaluate the diversity among the generated datasets, we used the Principal Components Analysis (PCA), which is a technique commonly used in the literature [14–16] to assess the (dis)similarity among benchmarks (such as our datasets). PCA is a multivariate statistical procedure that decreases a X dimensional space into a lower dimensional uncorrelated space (hence simplifying the analysis). Moreover, it generates a positive or negative weight associated with each metric. These weights transform the original higher dimension space into Y principal components. In our case, the chosen constituent metrics were the average and standard deviations of the frequencies of all the attributes in a domain. This strategy is similar to the one used in [14]. To measure the utility remaining in the data after anonymization, we used Generalized information loss (GenILoss) [19], a widely-used general-purpose metric that captures the penalty incurred when generalizing a specific attribute. In terms of costs, our main metrics were execution time, CPU (%) and memory (MB) utilizations. Garbage collection (GC) was also monitored as it is an important performance concern in Java [25].

4.2 Experimental results

Here we discuss our experimental results in terms of the diversity of the generated datasets, the benefits of using diverse datasets in the testing of anonymization algorithms, and the costs of using COCOA. Due to space constraint, we only present the most relevant results.

Dataset diversity. To assess the diversity of the datasets generated by COCOA, we calculated the PCA for all the datasets (grouped by domain and size). PCA computes the principal components (i.e., PC1, PC2, etc.) and identifies them in order of significance (i.e., PC1 is the most determinative component). Typically, most of the total variance is explained by the first few principal components. In our case, PC1 and PC2 accounted for at least 80% of the total variance in all the cases, so our analysis centered on them. Our hypothesis was that the usage of the different types of attribute generators (see Section 3.4) should lead to

variances in the data regardless the domain and size. This was confirmed by the results of this analysis. Although there were some (expected) differences in the degree of variance that COCOA achieved across the tested domains, in all cases a fair diversity of characteristics was achieved. Similar behavior was exhibited with all the sizes of the generated datasets. The main difference when comparing them was that the "scales" of the differences varied (precisely due to the size differences). This behavior is visually illustrated in Figs. 3a and 3b, which shows how the insurance datasets differ in a two-dimensional space (PC1 vs. PC2) for the 5K and 100K sizes. Intuitively, the farther the distance that separates the datasets within a domain size, the more different they are with respect to the metrics. Thus, as the datasets are well dispersed in both figures, the datasets differ independently of the size. For further reference, appendix B presents the constituent metrics used for PCA.

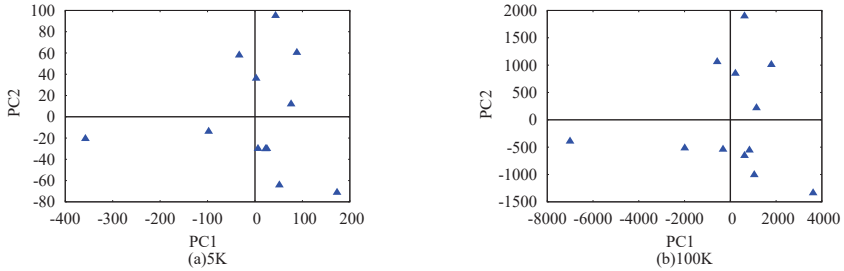

Fig. 3. PC1 vs PC2 for Insurance dataset with sizes (a)5K and (b)10K

Anonymization. We performed two sets of anonymizations. Firstly, to exemplify the usefulness of testing with multiple different datasets, five datasets of size 5K for each domain were anonymized. Secondly, to exemplify the benefits of testing with different sizes (i.e., scalability), we used a single variant of the insurance dataset (ActPoissonOccNormal) with 6 different sizes. It is important to remark that our intention was not to exhaustively investigate the data utility of the Mondrian algorithm (nor the effectiveness of k-anonymity), but to document how the resulting datasets are useful to broaden the testing of anonymization algorithms.

Our first analysis focused on assessing the data utility. Figs. 4a and 4b offer a high-level view of the GenILoss obtained, per domain, at each tested k-value. Fig. 4a shows the results obtained by using the original versions of the domains, while Fig. 4b shows the results achieved by using all the different generated datasets per domain. It can be clearly noticed how diverse values can be obtained by using a broad range of datasets (depicted by the standard deviations in Fig. 4b). This test strategy can then help to derive more generic conclusions from the results.

To further exemplify the risks of evaluating with single datasets, Figs. 5a, 5b, and 5c presents the GenILoss obtained for each anonymized dataset version, per domain, using a 5 k-value. It can be seen how the GenIloss values considerably fluctuate among the dataset variants. This shows the variability of results that can be obtained when different datasets are considered. This further motivates the usage of benchmarks that provide an adequate coverage of test scenarios.

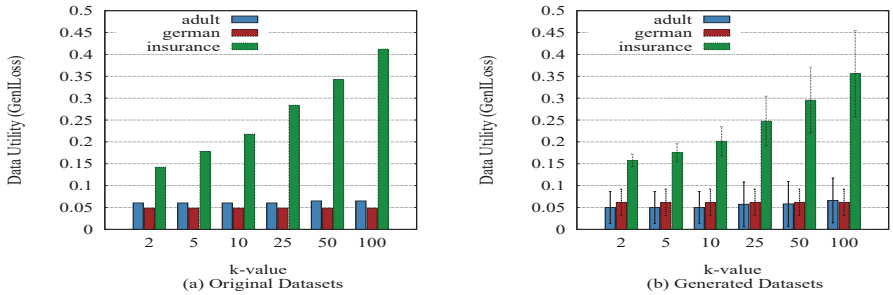

Fig. 4. Effectiveness of (a)Original and (b)Generated datasets of 5K size

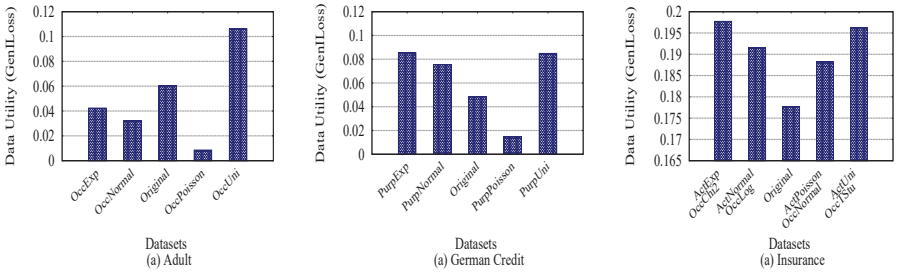

Fig. 5. Effectiveness of anonymized datasets (a)Adult, (b)German and (c)Insurance

Normally, the most important aspect of a scalability testing is to assess the costs of using an anonymization technique. These results are shown in Figs. 6a, 6b, and 6c which depict the execution time, average CPU, and memory utilizations, respectively. It can be noticed how Mondrian experiences a relatively exponential growth in terms of execution time, while requiring a low amount of resources (as both CPU and memory do not considerable grow with respect to the dataset size). Finally, the analysis of GC behavior showed that its performance cost was only significant for the 100K versions of the datasets. In those cases, the time spent performing MajorGC (MaGC) was 5% of the execution time (meaning that the processes can benefit from additional memory).

Costs. Figs. 7a, 7b, and 7c depict the execution time, average CPU, and memory utilization of the data generation process (per dataset size). It can be seen how COCOA experienced a relatively linear growth in all metrics. Furthermore, COCOA proved to be lightweight in terms of CPU

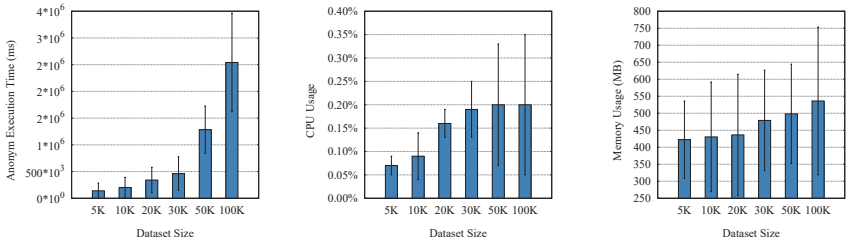

Fig. 6. Efficiency of Anonymization w.r.t. (a)Execution time, (b)CPU and (c)Memory

and execution time. For instance, the generation of the largest datasets (i.e., 100K) took an average of 4 sec (with a standard deviation of 1.5 sec). Similarly, the CPU never exceeded 10% (meaning that there was a considerable amount of idle resources to support larger dataset sizes). In terms of memory, COCOA only used approximately 35% of the available memory. It also never triggered a MaGC, which was another indicator that the memory settings were always appropriate for COCOA. Costs were also analyzed per domain. Although comparable across domains, the biggest costs were experienced by the german credit. This was because it contains the largest number of attributes. A second factor influencing the execution time was the complexity of the attribute generators. In this sense, the distribution-based ones tend to be less expensive (in terms of resources) than the other two types.

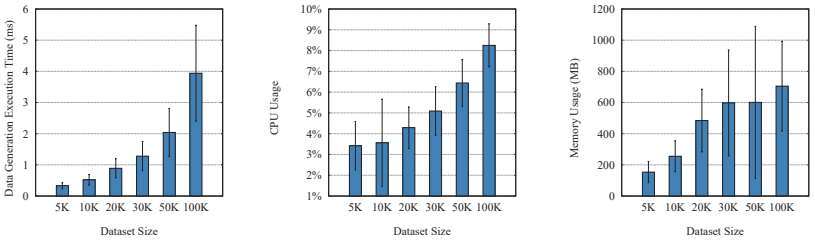

Fig. 7. Efficiency of COCOA w.r.t. (a)Execution time, (b)CPU and (c)Memory

5 Conclusions And Future Work

This paper presented COCOA, a framework for the generation of realistic synthetic microdata that can facilitate the testing of anonymization techniques. Given the characteristics of the desired data, COCOA can effectively create multi-dimensional datasets. Our experiments demonstrated the importance of using a comprehensive set of diverse testing datasets, and how COCOA can help to strengthen the testing. Finally, we showed that COCOA is lightweight in terms of computational resources, which makes it practical for real-world usage. As future work, we plan to expand the domains supported by COCOA, investigate other correlation types in the dataset, integrate other features for testing anonymization techniques, and release COCOA as a publicly-available tool.

Acknowledgments

This work was supported, in part, by Science Foundation Ireland grant 10/CE/I1855 to Lero - the Irish Software Research Centre (www.lero.ie)

References

1. COCOA Datasets, https://github.com/ucd-pel/COCOA/
2. CSO. Access to Microdata, http://www.cso.ie/en/aboutus/dissemination/accesstomicrodatarulespoliciesandprocedures/accesstomicrodata/
3. Data Benerator Tool, http://databene.org/databene-benerator
4. Eurostat. Access to Microdata, http://ec.europa.eu/eurostat/web/microdata
5. OpenForecast Library, http://www.stevengould.org/software/openforecast/index.shtml
6. Payscale USA, http://www.payscale.com/research/US/
7. simPop: Simulation of Synthetic Populations for Survey Data Considering Auxiliary Information, https://cran.r-project.org/web/packages/simPop
8. synthpop: Generating Synthetic Versions of Sensitive Microdata for Statistical Disclosure Control, https://cran.r-project.org/web/packages/synthpop
9. UML basics: The component diagram, https://www.ibm.com/developerworks/rational/library/dec04/bell/
10. US Census. Restricted-Use Microdata, http://www.census.gov/research/data/restricted_use_microdata.html
11. UTD Anonymization ToolBox, http://cs.utdallas.edu/dspl/cgi-bin/toolbox/
12. Ayala-Rivera, V., McDonagh, P., Cerqueus, T., Murphy, L.: Synthetic data generation using benerator tool. arXiv preprint arXiv:1311.3312 (2013)
13. Ayala-Rivera, V., McDonagh, P., Cerqueus, T., Murphy, L.: A systematic comparison and evaluation of k-anonymization algorithms for practitioners. Transactions on Data Privacy 7(3), 337–370 (2014)
14. Blackburn, S.M., Garner, R., Hoffmann, C., Khang, A.M., McKinley, K.S., Bentzur, R., Diwan, A., Feinberg, D., Frampton, D., Guyer, S.Z., et al.: The dacapo benchmarks: Java benchmarking development and analysis. In: ACM Sigplan Notices. vol. 41, pp. 169–190. ACM (2006)
15. Chow, K., Wright, A., Lai, K.: Characterization of java workloads by principal components analysis and indirect branches. In: Proceedings of the Workshop on Workload Characterization (WWC-1998), held in conjunction with the 31st Annual ACM/IEEE International Symposium on Microarchitecture (MICRO-31). pp. 11–19 (1998)
16. Eeckhout, L., Georges, A., De Bosschere, K.: How java programs interact with virtual machines at the microarchitectural level. In: ACM SIGPLAN Notices. vol. 38, pp. 169–186. ACM (2003)
17. Hoag, J.E., Thompson, C.W.: A parallel general-purpose synthetic data generator. ACM SIGMOD Record 36(1), 19–24 (2007)
18. Hu, J., Reiter, J.P., Wang, Q.: Disclosure risk evaluation for fully synthetic categorical data. In: Privacy in Statistical Databases. pp. 185–199. Springer (2014)
19. Iyengar, V.S.: Transforming data to satisfy privacy constraints. In: Int. Conf. on Knowledge Discovery and Data Mining. pp. 279–288 (2002)
20. LeFevre, K., DeWitt, D., Ramakrishnan, R.: Mondrian Multidimensional K-Anonymity. In: Int. Conf. Data Eng. p. 25 (2006)

21. Lichman M.: UCI Machine Learning Repository (2013)
22. Machanavajjhala, A., Kifer, D., Abowd, J., Gehrke, J., Vilhuber, L.: Privacy: Theory meets Practice on the Map. In: Int. Conf. Data Eng. pp. 277–286 (2008)
23. Mateo-Sanz, J.M., Martínez-Ballesté, A., Domingo-Ferrer, J.: Fast generation of accurate synthetic microdata. In: Privacy in Statistical Databases. pp. 298–306. Springer (2004)
24. Pedersen, K.H., Torp, K., Wind, R.: Simple and realistic data generation. In: Proceedings of the 32nd International Conference on Very Large Data Bases. Association for Computing Machinery (2006)
25. Portillo-Dominguez, A.O., Perry, P., Magoni, D., Wang, M., Murphy, J.: Trini: an adaptive load balancing strategy based on garbage collection for clustered java systems. Software: Practice and Experience (2016)
26. Rubin, D.B.: Discussion of statistical disclosure limitation. J. Off. Stat. 9(2), 461–468 (1993)
27. Samarati, P.: Protecting respondents identities in microdata release. Trans. on Knowledge and Data Engineering 13(6), 1010–1027 (2001)
28. Sweeney, L.: k-Anonymity: A model for protecting privacy. Int. J. Uncertain. Fuzziness Knowl.-Based Syst 10(05), 557–570 (Oct 2002)
29. Walck, C.: Handbook on statistical distributions for experimentalists (2007)

Appendix A Structure of the Irish Census and Insurance Domains

Tables 1 and 2, respectively, list the attributes and the type of generators used for producing data for the Irish census and insurance domains (discussed in Section 3.5).

Table 1. Irish census domain structure.

attribute name	**distribution -based**	**distribution/ attribute-based**	**attribute -based**
given name		x (age,native country)	
family name		x (native country)	
age	x		
gender	x		
marital status		x (age)	
education level		x (age)	
county		x(age)	
economic status		x (age)	
industral group		x (gender)	
field of study area		x (gender)	
field of study		x (field of study area)	
native country	x		

Table 2. Insurance domain structure.

Attribute name	**Frequency -based**	**Frequency/ attribute-based**	**Attribute-based**
gender		x (occupation)	
age	x		
age range			x (age)
occupation	x		
place of work		x (occupation)	
activity	x		
place of activity		x(activity)	
years of experience		x (age)	
salary		x (occupation,YoE)	
risk of accident			x (age, occ, placeOfWork, activity, placeOfActivity)
salary class			x (salary)
risk of accident class			x (riskOfAccident)

Appendix B PCA Constituent Metrics

Table 3 lists the constituent metrics used to perform the PCA analysis of the datasets generated by COCOA (discussed in Section 4.2).

Table 3. Constituent Metrics for PC Analysis per domain.

German domain Metrics	**Adult domain Metrics**	**Insurance domain Metrics**
otherDebtors-Stdev.Freq.	country-Avg.Freq.	placeOfActivity-Stdev.Freq.
foreignWorker-Stdev.Freq.	country-Stdev.Freq.	activity-Avg.Freq.
presentResidence-Stdev.Freq.	sex-Avg.Freq.	riskOfAccidentClass-Stdev.Freq.
housing-Avg.Freq.	race-Stdev.Freq.	riskOfAccidentClass-Avg.Freq.
otherInstallmentPlans-Avg.Freq.	education-Avg.Freq.	salaryCatClass-Avg.Freq.
housing-Stdev.Freq.	marital-Avg.Freq.	occupation-Avg.Freq.
presentEmployment-Stdev.Freq.	education-Stdev.Freq.	placeOfActivity-Avg.Freq.
creditHistory-Stdev.Freq.	income-Stdev.Freq.	salaryClass-Avg.Freq.
presentEmployment-Avg.Freq.	marital-Stdev.Freq.	occupation-Stdev.Freq.
ageRange-Avg.Freq.	occupation-Avg.Freq.	semAge-Avg.Freq.
ageRange-Stdev.Freq.	sex-Stdev.Freq.	age-Avg.Freq.
duration-Stdev.Freq.	typeEmployer-Stdev.Freq.	age-Stdev.Freq.
statusCheckingAccount-Stdev.Freq.	typeEmployer-Avg.Freq.	placeOfWork-Stdev.Freq.
duration-Avg.Freq.	occupation-Stdev.Freq.	gender-Stdev.Freq.
savings-Stdev.Freq.	income-Avg.Freq.	activity-Stdev.Freq.
creditHistory-Avg.Freq.	race-Avg.Freq.	salaryCatClass-Stdev.Freq.
savings-Avg.Freq.	age-Avg.Freq.	placeOfWork-Avg.Freq.
property-Stdev.Freq.	age-Stdev.Freq.	gender-Avg.Freq.
creditAmount-Stdev.Freq.		semAge-Stdev.Freq.
property-Avg.Freq.		salaryClass-Stdev.Freq.
presentResidence-Avg.Freq.		
age-Avg.Freq.		
installmentRate-Avg.Freq.		
statusCheckingAccount-Avg.Freq.		
statusAndSex-Stdev.Freq.		
statusAndSex-Avg.Freq.		
job-Stdev.Freq.		
age-Stdev.Freq.		
otherInstallmentPlans-Stdev.Freq.		
telephone-Stdev.Freq.		
costMatrix-Stdev.Freq.		
numberExistingCredits-Stdev.Freq.		
numberLiablePeople-Stdev.Freq.		
telephone-Avg.Freq.		
numberLiablePeople-Avg.Freq.		
creditAmount-Avg.Freq.		
numberExistingCredits-Avg.Freq.		
foreignWorker-Avg.Freq.		
job-Avg.Freq.		
costMatrix-Avg.Freq.		
otherDebtors-Avg.Freq.		
installmentRate-Stdev.Freq.		
purpose-Avg.Freq.		
purpose-Stdev.Freq.		