Pattern-Guided k-Anonymity

Robert Bredereck[⋆], André Nichterlein, and Rolf Niedermeier

Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Berlin, Germany
`{robert.bredereck,andre.nichterlein,rolf.niedermeier}@tu-berlin.de`

Abstract. We suggest a user-oriented approach to combinatorial data anonymization. A data matrix is called k-anonymous if every row appears at least k times—the goal of the NP-hard k-Anonymity problem then is to make a given matrix k-anonymous by suppressing (blanking out) as few entries as possible. We describe an enhanced k-anonymization problem called Pattern-Guided k-Anonymity where the users can express the differing importance of various data features. We show that Pattern-Guided k-Anonymity remains NP-hard. We provide a fixed-parameter tractability result based on a data-driven parameterization and, based on this, develop an exact ILP-based solution method as well as a simple but very effective greedy heuristic. Experiments on several real-world datasets show that our heuristic easily matches up to the established "Mondrian" algorithm for k-Anonymity in terms of quality of the anonymization and outperforms it in terms of running time.

1 Introduction

Making a matrix k-*anonymous*, that is, each row has to occur at least k times, is a classic model for (combinatorial) data privacy [12, 21].[1] The idea behind is that each row of the matrix represents an individual and the k-fold appearance of the corresponding row avoids that the person or object behind can be identified. To reach this goal, clearly some information loss has to be accepted, that is, some entries of the matrix have to be suppressed (blanked out); in this way, information about certain attributes (represented by the columns of the matrix) is lost. Thus, the natural goal is to minimize this loss of information when transforming an arbitrary data matrix into a k-anonymous one. The corresponding optimization problem k-Anonymity is NP-hard (even in special cases) and hard to approximate [1, 2, 3, 7, 20]. Nevertheless, it played a significant role in many applications, mostly relying on heuristic approaches for making a matrix k-anonymous [6, 13, 21].

[⋆] Supported by the DFG, research project PAWS, NI 369/10.

[1] We omit considerations on the recently very popular model of "differential privacy" [8] which has a more statistical than a combinatorial flavor. It is well-known that there are certain weaknesses of the k-anonymity concept when the anonymized data is used multiple times [5, 12]. Here, we focus on k-anonymity which due to its simplicity and good interpretability continues to be of interest in current applications.

To appear in *Proceedings of the Joint Conference of the 7th International Frontiers of Algorithmics Workshop and the 9th International Conference on Algorithmic Aspects of Information and Management (FAW-AAIM '13)*, Dalian, China, June 2013. © Springer.

It was observed that care has to be taken concerning the "usefulness" (also in terms of expressiveness) of the anonymized data [18, 22]. Indeed, depending on the application that has to work on the k-anonymized data, certain entry suppressions may "hurt" less than others. E.g., considering medical data records, the information about eye color may be less informative than information about the blood pressure. Hence, it would be useful for the later user of the anonymized data to specify information that may help doing the anonymization process in a more sophisticated way. Thus, in recent work [4] we proposed a "pattern-guided" approach to data anonymization, in a way allowing the user to specify which combinations of attributes are less harmful to suppress than others. More specifically, the approach allows "pattern vectors" which may be considered as blueprints for the structure of anonymized rows—each row has to be matched with exactly one of the pattern vectors. The correspondingly proposed optimization problem [4], however, has the clear weakness that each pattern vector can only be used once, disallowing that there are different incarnations of the very same anonymization pattern. We have no justification why this should be so and we see no reason to justify this constraint from the viewpoint of data privacy. This leads us to proposing a modified model whose usefulness for practical data anonymization tasks is supported by experiments on real-world data.

Altogether, with our new model we can improve both on k-Anonymity by letting the data user influence the anonymization process as well as on the previous model [4] by allowing the full flexibility for the data user to influence the anonymization process.

Formal Introduction of the New Model. A *row type* is a maximal set of identical rows of a matrix. Matrices are made k-anonymous by suppressing some of their entries. Formally, *suppressing* an entry $M[i, j]$ of an $n \times m$-matrix M over alphabet Σ with $1 \leq i \leq n$ and $1 \leq j \leq m$ means to simply replace $M[i, j] \in \Sigma$ by the new symbol "$\star$", ending up with a matrix over the alphabet $\Sigma \cup \{\star\}$.

Our central enhancement of the k-Anonymity model lies in the user-specific pattern mask guiding the anonymization process: Every row in the k-anonymous output matrix has to conform to one of the given pattern vectors. A row r in a matrix $M \in \{\Sigma, \star\}^{n\times m}$ *matches* a pattern vector $v \in \{\square, \star\}^m$ if and only if $\forall 1 \leq i \leq m : r[i] = \star \iff v[i] = \star$, that is, r and v have $\star$-symbols at the same positions. With these definitions we can now formally define our central computational problem. The decisive difference to our previous model [4] is that in our new model two non-identical output rows can match the same pattern vector.

Pattern-Guided k-Anonymity
Input: A matrix $M \in \Sigma^{n\times m}$, a pattern mask $P \in \{\square, \star\}^{p\times m}$, and two positive integers k and s.
Question: Can one suppress at most s elements of M in order to get a k-anonymous matrix M' such that each row type of M' can be matched to one pattern vector of P?

Our Results. We show that Pattern-Guided k-Anonymity is NP-complete, even if the input matrix only consists of three columns, there are only two pattern vectors, and $k = 3$. Motivated by this computational intractability result, we develop an exact algorithm that solves Pattern-Guided k-Anonymity in $O(2^{tp}t^6p^5m + nm)$ time for an $n \times m$ input matrix M, p pattern vectors, and the number of different rows in M being t. In other words, this shows that Pattern-Guided k-Anonymity is fixed-parameter tractable for the combined parameter (t, p) and actually can be solved in linear time if t and p take constant values. This result paves the way to a formulation of Pattern-Guided k-Anonymity as an integer linear program for exactly solving moderate-size instances of Pattern-Guided k-Anonymity. Furthermore, our fixed-parameter tractability result also leads to a simple and efficient greedy heuristic whose practical competitiveness is underlined by a set of experiments with real-world data, also favorably comparing with the Mondrian algorithm for k-Anonymity [15].

Due to the lack of space, several details and experimental evaluations are deferred to a full version.

2 Complexity and Algorithms

Natural parameters occurring in the problem definition of Pattern-Guided k-Anonymity are the number n of rows, the number m of columns, the alphabet size $|\Sigma|$, the number p of pattern vectors, the degree of anonymity k, and the cost bound s. In general, the number of rows will arguably be large and, thus, also the cost bound s tends to be large. However, analyzing the adult dataset [10] prepared as described by Machanavajjhala et al. [19], it turns out that some of the other mentioned parameters are small: The dataset has $m = 9$ columns and the alphabet size is 73. Furthermore, it is natural to assume that also the number of pattern vectors is not that large. Indeed, compared to the $n = 32,561$ rows even the number of *all possible* pattern vectors $2^9 = 512$ is smaller. Finally there are applications where k, the degree of anonymity, is small [9]. Summarizing, we can state that fixed-parameter tractability with respect to the parameters $|\Sigma|$, m, or p could be of practical relevance. Unfortunately, by reducing from the 3-Set Cover we can show that Pattern-Guided k-Anonymity is NP-hard in very restricted cases.

Theorem 1. Pattern-Guided k-Anonymity *is NP-complete even for two pattern vectors, three columns, and $k = 3$.*

Proof. We reduce from the NP-hard 3-Set Cover [14]: Given a set family $\mathcal{F} = \{S_1, \dots, S_\alpha\}$ with $|S_i| = 3$ over a universe $U = \{u_1, \dots, u_\beta\}$ and a positive integer h, the task is to decide whether there is a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ of size at most h such that $\bigcup_{S \in \mathcal{F}'} S = U$. In the reduction we need unique entries in the constructed input matrix M. For ease of notation we introduce the $\triangle$-symbol with an unusual semantics. Each occurrence of a $\triangle$-symbol stands for a *different* unique symbol in the alphabet Σ. One could informally state this as "$\triangle \neq \triangle$". We now describe the construction. Let $(\mathcal{F}, U, h)$ be the 3-Set Cover instance. We construct an equivalent instance (M, P, k, s) of Pattern-Guided k-Anonymity as follows: Initialize M and P as empty matrices. Then, for each element $u_i \in U$ add the row $(u_i, \triangle, \triangle)$ twice to the input matrix M. For each set $S_i \in \mathcal{F}$ with $S_i = \{u_a, u_b, u_c\}$ add to M the three rows (u_a, S_i, S_i), (u_b, S_i, S_i), and (u_c, S_i, S_i). Finally set $k = 3$, $s = 4|U| + 3|\mathcal{F}| + 3h$ and add to P the pattern vectors $(\square, \star, \star)$ and $(\star, \square, \square)$.

We show the correctness of the above construction by proving that $(\mathcal{F}, U, h)$ is a yes-instance of 3-Set Cover if and only if $(M, P, 3, s)$ is a yes-instance of Pattern-Guided k-Anonymity.

"$\Rightarrow$:" If $(\mathcal{F}, U, h)$ is a yes-instance of 3-Set Cover, then there exists a set cover $\mathcal{F}'$ of size at most h. We suppress the following elements in M: First, suppress all $\triangle$-entries in M. This gives $4|U|$ suppressions. Then, for each $S_i \in \mathcal{F}'$ suppress all S_i-entries in M. This gives at most $6|\mathcal{F}'|$ suppressions. Finally, for each $S_j \notin \mathcal{F}'$ suppress the first column of all rows containing the entry S_j. This are $3(|\mathcal{F}| - |\mathcal{F}'|)$ suppressions. Let M' denote the matrix with the suppressed elements. Note that M' contains $4|U| + 3|\mathcal{F}| + 3|\mathcal{F}'| \leq s$ suppressed entries. Furthermore, in each row in M' either the first element is suppressed or the last two elements. Hence, each row of M' matches to one of the two pattern vectors of P. Finally, observe that M' is 3-anonymous: The three rows corresponding to the set $S_j \notin \mathcal{F}'$ are identical: the first column is suppressed and the next two columns contain the symbol S_j. Since $\mathcal{F}'$ is a set cover, there exists for each element u_j a set $S_i \in \mathcal{F}'$ such that $u_j \in S_i$. Thus, by construction, the two rows corresponding to the element u_j and the row (u_j, S_i, S_i) in M coincide in M': The first column contains the entry u_j and the other two columns are suppressed. Finally, for each row (u_i, S_j, S_j) in M that corresponds to a set $S_j \in \mathcal{F}'$ the row in M' coincides with the two rows corresponding to the element u_i: Again, the first column contains the entry u_i and the other two columns are suppressed.

"$\Leftarrow$:" If $(M, P, 3, s)$ is a yes-instance of Pattern-Guided k-Anonymity, then there is a 3-anonymous matrix M' that is obtained from M by suppressing at most s elements and each row of M' matches to one of the two pattern vectors in P. Since M and so M' contain $2|U| + 3|\mathcal{F}|$ rows, M' contains at most $s = 4|U| + 3|\mathcal{F}| + 3h$ suppressions and each pattern vector contains a $\star$-symbol, there are at most $2|U| + 3h$ rows in M' containing two suppressions and at least $3|\mathcal{F}| - 3h$ rows containing one suppression. Furthermore, since the $2|U|$ rows in M corresponding to the elements of U contain the unique symbol $\triangle$ in the last two columns, in M these rows are suppressed in the last two columns. Thus, at most $3h$ rows corresponding to sets of $\mathcal{F}$ have two suppressions in M'. Observe that for each set $S_i \in \mathcal{F}$ the entries in the last two columns of the corresponding rows are S_i. There is no other occurrence of this entry in M. Hence, the at least $3|\mathcal{F}| - 3h$ rows in M' with one suppression correspond to $|\mathcal{F}| - h$ sets in $\mathcal{F}$. Thus, the at most $3h$ rows in M' that correspond to sets of $\mathcal{F}$ and contain two suppressions correspond to at most h sets of $\mathcal{F}$. Denote these h sets by $\mathcal{F}'$. We now show that $\mathcal{F}'$ is a set cover for the 3-Set Cover instance. Assume by contradiction that $\mathcal{F}'$ is no set cover and, hence, there is a set $u \in U \setminus (\bigcup_{S \in \mathcal{F}'} S)$. But since M' is 3-anonymous, there has to be a row r

in M' that corresponds to some set S_i such that this row coincides with the two rows r_1^u and r_2^u corresponding to u. Since all rows in M' corresponding to elements of U contain two suppressions in the last two columns, the row r also contains two suppressions in the last two columns. Thus, $S_i \in \mathcal{F}'$. Furthermore, r has to coincide with r_1^u and r_2^u in the first column, that is, r contains as entry in the first column the symbol u. Hence, $u \in S_i$, a contradiction. □

Contrasting Theorem 1, we will show fixed-parameter tractability with respect to the combined parameter $(|\Sigma|, m)$. To this end, we additionally use as parameter the number t of different input rows. Indeed, we show fixed-parameter tractability with respect to the combined parameter (t, p). This implies fixed-parameter tractability with respect to the combined parameter $(|\Sigma|, m)$ as $|\Sigma|^m \geq t$ and $|\Sigma|^m \geq 2^m \geq p$. This results from an adaption of combinatorial algorithms from previous work [4, 5].

Theorem 2. Pattern-Guided k-Anonymity *can be solved in* $O(2^{tp} \cdot t^6 p^5 \cdot m + nm)$ *time where* p *is the number of pattern vectors and* t *is the number of different rows in the input matrix* M.

ILP Formulation. Next, we describe an integer linear program (ILP) formulation for Pattern-Guided k-Anonymity employing ideas behind the fixed-parameter algorithm of Theorem 2. To this end, we need the following notation. We distinguish between the *input* row types of the input matrix M and the *output* row types of the output matrix M'. Note that in the beginning we can compute the input row types of M in $O(nm)$ time using a trie [11], but the output row types are unknown. By the definition of Pattern-Guided k-Anonymity, each output row type R' has to match a pattern vector $v \in P$. We call R' an *instance* of v.

More specifically, our ILP contains the integer variables $x_{i,j}$ denoting the number of rows from type i being assigned into an output row type compatible with pattern vector j. The binary variable $u_{j,l}$ is 0 if instance l of pattern vector j is used in the solution, that is, there is at least one row mapped to it, otherwise it may be set to 1. Furthermore, n_i denotes the number of rows of type i, ω_j denotes the costs of pattern vector j, and k is the required degree of anonymity. Let $\hat{p}_i \leq t$ denote the number of instances of pattern vector i and let $c(i, j, l)$ be 1 if mapping row i to pattern vector j produces pattern vector instance l, otherwise $c(i, j, l) = 0$. With this notation we can state our ILP formulation:

$$\min \sum_{i=1}^{t} \sum_{j=1}^{p} x(i, j) \cdot \omega_j \tag{1}$$

$$\sum_{i=1}^{t} c(i, j, l) \cdot x_{i,j} \leq (1 - u_{j,l}) \cdot n \qquad {\scriptstyle 1 \leq j \leq p \atop 1 \leq l \leq \hat{p}_j} \tag{2}$$

$$\sum_{i=1}^{t} c(i,j,l) \cdot x_{i,j} + k \cdot u_{j,l} \geq k \qquad \substack{1 \leq j \leq p \\ 1 \leq l \leq \hat{p}_j} \tag{3}$$

$$\sum_{j=1}^{p} x_{i,j} = n_i \qquad 1 \leq i \leq t. \tag{4}$$

The goal function (1) ensures that the solution has a minimum number of suppressions. Constraint (2) ensures that the variables $u_{j,l}$ are consistently set with the variables $x_{i,j}$, that is, if there is some positive variable $x_{i,j}$ indicating that the instance l of pattern vector j is used, then $u_{j,l} = 0$. Constraint (3) ensures that every pattern vector instance that is used by the solution contains at least k rows. Constraint (4) ensures that the solution uses as many rows from each row type as available.

We remark that, as Theorem 2, our ILP formulation also yields fixed-parameter tractability with respect to the combined parameter (t, p). This is due to a famous result of Lenstra [16] and the fact that the number of variables in the ILP is bounded by $O(tp)$. Theorem 2, however, provides a direct *combinatorial* algorithm with better worst-case running time bounds. Nevertheless, in the experimental section we decided to use the ILP formulation and not the combinatorial algorithm based on the experience that there are very strong (commercial) ILP solvers that in practice typically perform much better than the worst-case analysis predicts.

Greedy Heuristic. We now provide a greedy heuristic based on the ideas of the fixed-parameter algorithm of Theorem 2. The fixed-parameter algorithm basically does exhaustive search on the assignment of rows to pattern vectors. More precisely, for each row type R and each pattern vector v it tries both possibilities of whether rows of R are assigned to v or not. In contrast, our greedy heuristic will just pick for each input row type R the "cheapest" pattern vector v and then assigns all compatible rows of M to v. This is realized as follows: We consider all pattern vectors one after the other ordered by increasing number of $\star$-symbols. This ensures that we start with the "cheapest" pattern vector. Then we assign as many rows as possible of M to v: We just consider every instance R' of v and if there are more than k rows in M that are compatible with R', then we assign all compatible rows to R'. Once a row is assigned, it will not be reassigned to any other output row type and, hence, the row will be deleted from M. Overall this gives a running time of $O(pnm)$. See Algorithm 1 for the pseudo-code of the greedy heuristic. If at some point of time there are less than k remaining rows in M, then these rows will be fully suppressed. Note that this slightly deviates from our formal definition of Pattern-Guided k-Anonymity. However, since fully suppressed rows do not reveal any data, this potential violation of the k-anonymity requirement does not matter.

Our greedy heuristic clearly does not always provide optimal solutions. Our experiments indicate, however, that it is very fast and that it typically provides solutions close to the optimum and outperforms the Mondrian algorithm [15] in most datasets we tested. While this demonstrates the practicality of Algo-

Algorithm 1 Greedy Heuristic (M, P, k)

```
1: Sort pattern vectors $P$ by cost (increasing order)
2: for each $v \in P$ do
3:     Compute all instances of $v$
4:     for each instance $R'$ of $v$ do
5:         if $\geq k$ rows are compatible with $R'$ then
6:             Assign all compatible rows of $M$ to $R'$
7:             Delete the assigned rows from $M$.
```

rithm 1, the following result shows that from the viewpoint of polynomial-time approximation algorithmics it is weak in the worst case.

Theorem 3. *Algorithm 1 for* PATTERN-GUIDED k-ANONYMITY *runs in* $O(pnm)$ *time and provides a factor* m*-approximation. This approximation bound is asymptotically tight for Algorithm 1.*

Proof. Since the running time is already discussed above, it remains to show the approximation factor. Let s_{heur} be the number of suppressions in a solution provided by Algorithm 1 and s_{opt} be the number of suppressions in an optimal solution. We show that for every instance it holds that $s_{\text{heur}} \leq m \cdot s_{\text{opt}}$. Let M be a matrix and M'_{heur} be the suppressed matrix produced by Algorithm 1 and M'_{opt} be the suppressed matrix corresponding to an optimal solution. First, observe that if any row of M occurs more than k times, then this row does not contain any suppressed entry in M'_{heur}. Hence, for any row in M'_{opt} not containing any suppressed entry it follows that the corresponding row in M'_{heur} also does not contain any suppression. Clearly, each row in M'_{heur} has at most m entries suppressed. Thus, each row in M'_{heur} has at most m times more suppressed entries than the corresponding row in M'_{opt}.

To show that this upper bound is asymptotically tight, consider the following instance. Set $k = m$ and let M be as follows: The matrix M contains k times the row with the symbol 1 in every entry. Furthermore, for each $i \in \{1, \ldots, m\}$ there are $k-1$ rows in M such that all but the i^{th} entry contains the symbol 1. In the i^{th} entry each of the $k-1$ rows contains a uniquely occurring symbol. Finally, the pattern mask simply contains all 2^m possible rows/vectors over the alphabet $\{\square, \star\}$. Algorithm 1 will suppress nothing in the k all-1 rows and will suppress every entry of the remaining rows. This gives $s_{\text{heur}} = (k-1) \cdot m^2 = (m-1) \cdot m^2$ suppressions. However, an optimal solution suppresses in each row exactly one entry: The rows containing in all but the i^{th} entry the symbol 1 are suppressed in the i^{th} entry. Furthermore, to ensure the anonymity requirement, in the submatrix with the k rows containing the symbol 1 in every entry the diagonal is suppressed. Thus, the number of suppressions is equal to the number of rows, that is, $s_{\text{opt}} = k + (k-1)m = m^2$. Hence, $s_{\text{heur}} = (m-1)s_{\text{opt}}$. $\square$

3 Implementation and Experiments

In this section we present the results of our experimental evaluation of the heuristic and the ILP formulation presented in Section 2. We used four datasets for our experimental evaluations; these were taken from the UCI machine learning repository [10]. In this extended abstract we only discuss two of them.

Adult[2]: This was extracted from a dataset of the US Census Bureau Data Extraction System. It consists of 32,561 records over 15 attributes. Since the entries in one attribute are unique for roughly half of the records, we removed this attribute from the dataset. Following Machanavajjhala et al. [19], we prepared also this dataset with nine attributes and call this variant Adult-2.

CMC[3]: This dataset is a subset of the 1987 National Indonesia Contraceptive Prevalence Survey. It contains 1,473 records over 10 attributes.

All our experiments are performed on an Intel Xeon E5-1620 3.6GHz machine with 64GB memory under the Debian GNU/Linux 6.0 operating system. The heuristic is implemented in Haskell. The ILP implementation is using ILOG CPLEX by its C++ API. Both implementations are licensed under GPL Version 3. The source code is available from `http://akt.tu-berlin.de/menue/software/`.

We tested our greedy heuristic in two types of experiments. In the first type we "misused" our greedy heuristic to solve the classical k-Anonymity problem by specifying all possible pattern vectors since we wanted to compare the practical relevance of our greedy heuristic with an existing implementation. We decided to compare with an implementation of the well-known Mondrian [15] algorithm[4] since we could not find a more recent implementation of a k-Anonymity algorithm which is freely available. In the second type of experiments, we analyze the distance of the results provided by our greedy heuristic from an optimal solution (with a minimum number of suppressed entries). The optimal solution is provided by the ILP implementation.

Obvious criteria for the evaluation of the experiments are the number of suppressions and the running time. Furthermore, we use the average size h_{avg} and the maximum size $h_{\max}$ of the output row types as already done by Li et al. [17] and Machanavajjhala et al. [19], as well as the number $\#h$ of output row types. The perhaps most difficult to describe measurement we use is the "usefulness" introduced by Loukides and Shao [18]. Roughly speaking, the *usefulness* is the average tuple diversity of all output row types. Usefulness values lie between zero and the number m of columns. Except for $\#h$, small values indicate better solutions.

[2] `ftp://ftp.ics.uci.edu/pub/machine-learning-databases/adult/`
[3] `ftp://ftp.ics.uci.edu/pub/machine-learning-databases/cmc/`
[4] `http://cs.utdallas.edu/dspl/cgi-bin/toolbox/index.php?go=home`

Table 1. Heuristic vs. Mondrian: Results for the Adult dataset. The usefulness is denoted by u, the running time in seconds by r, $\#h$ denotes the number of output row types, h_{avg} denotes the average size of the output row types, and h_{max} denotes the maximum size of the output row types.

Greedy Heuristic						Mondrian					
k	u	r	$\#h$	h_{avg}	h_{max}	k	u	r	$\#h$	h_{avg}	h_{max}
2	2.062	5.5	14589	2.232	16	2	3.505	2789.4	11136	2.709	61
3	2.290	13.2	9208	3.536	18	3	3.782	1803.5	7306	4.128	61
4	2.470	19.538	6670	4.882	25	4	4.007	1337.860	5432	5.553	61
5	2.615	24.9	5199	6.263	31	5	4.191	1062.0	4325	6.974	61
6	2.738	29.663	4315	7.546	42	6	4.362	885.939	3597	8.385	61
7	2.851	34.126	3669	8.875	53	7	4.498	754.652	3053	9.879	61
8	2.942	37.629	3193	10.198	53	8	4.622	659.184	2663	11.326	61
9	3.026	41.216	2832	11.498	52	9	4.766	588.347	2368	12.737	69
10	3.106	44.8	2559	12.724	56	10	4.875	535.9	2145	14.062	69
25	3.840	79.281	1046	31.129	161	25	6.009	229.248	850	35.485	90
50	4.462	117.0	537	60.635	317	50	6.729	127.4	430	70.144	135
75	4.873	144.536	354	91.980	317	75	7.339	93.621	287	105.094	242
100	5.151	163.582	274	118.836	317	100	7.805	76.005	209	144.316	242

Heuristic vs. Mondrian

For each dataset, we computed k-anonymous datasets with our greedy heuristic and Mondrian for $k \in \{2, 3, \ldots, 10, 25, 50, 75, 100\}$. The running time behavior of the tested algorithms is somewhat unexpected. Whereas Mondrian gets faster with increasing k, our greedy heuristic gets faster with decreasing k. The reason why the greedy heuristic is faster for small values of k is that usually the cheap pattern vectors are used and, hence, the number of remaining input rows decreases soon. On the contrary, when k is large, the cheap pattern vectors cannot be used and, hence, the greedy heuristic tests many pattern vectors before it actually starts with removing rows from the input matrix. Thus, for larger values of k the greedy heuristic comes closer to its worst-case running time of $O(pnm)$ with $p = 2^m$.

Adult and Adult-2. Our greedy heuristic anonymized the Adult dataset in less than three minutes for all tested values of k. For $k = 3$ and $k = 4$ Mondrian took more than half an hour to anonymize the dataset. However, in contrast to all other values of k, Mondrian was slightly faster for $k = 75$ and $k = 100$. Except for h_{max} with $k \geq 25$ all quality measures indicate that our heuristic produces better solutions. The usefulness value of the Mondrian solutions is between 1.5 and 1.7 times the usefulness value of the heuristic for all tested k—this indicates significantly better quality of the results of our heuristic. See Table 1 for details and Figure 1 for an illustration.

The solutions for Adult-2 behave similarly to those for Adult. Our greedy heuristic with a maximum running time of five seconds is significantly faster than

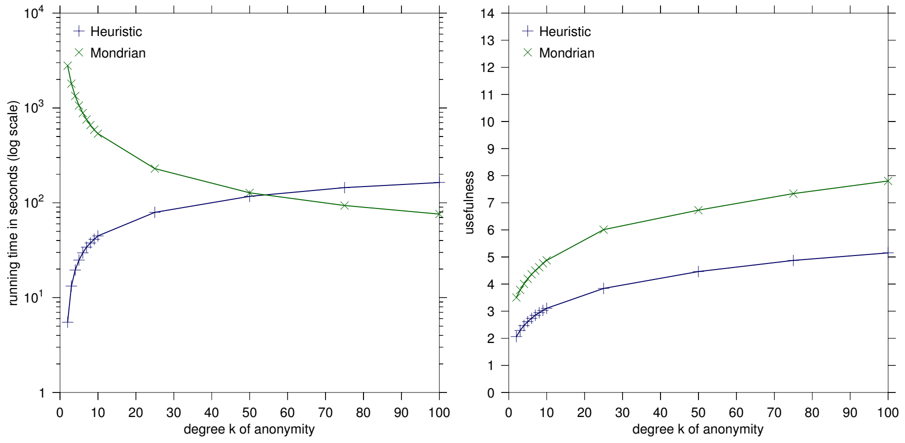

Fig. 1. Comparing running time and usefulness for the Adult dataset.

Mondrian with a maximum running time of 20 minutes (at least 10 times faster for all tested k). However, the usefulness is quite similar for both algorithms. Mondrian beats the heuristic by less than 1% for $k = 50$; the heuristic is slightly better for each other tested k.

CMC. For the CMC dataset, both algorithms were very fast in computing k-anonymous datasets for every tested k. Mondrian took at most 10 seconds and our greedy heuristic took at most 1.2 seconds and was always faster than Mondrian. As for the solution quality, the heuristic can compete with Mondrian. The usefulness of the heuristic results is always slightly better, the Mondrian results have always at least 20% less output row types, and the average output row type size of the heuristic results is always smaller. Only for $k = 5, 6, 7$, and 8, the Mondrian results have a lower maximum size of the output row types.

Conclusions for Classical k-ANONYMITY. We showed that our greedy heuristic is very efficient even for real-world datasets with more than 30,000 records and with $k \leq 100$. Especially for smaller degrees of anonymity $k \leq 10$, Mondrian is at least ten times slower. Altogether, our heuristic outperforms Mondrian in terms of quality of the solution. Hence, we demonstrated that even for the very special case of specifying *all* possible pattern vectors, our heuristic already produces useful solutions that can at least compete with Mondrian's solutions.

Heuristic vs. Exact Solution

In three scenarios with real-world datasets, we showed that our greedy heuristic performs well in terms of solution quality compared with the optimal solution produced by the ILP implementation. The results of the heuristic are typically within 15% away from the optimal solution to the optimum and in fact for many cases they were optimal, although our heuristic is much more efficient than the exact algorithm (the ILP was, on average, more than 1000 times slower). The heuristic results tend to get closer to the optimal number of suppressions with increasing degree of anonymity k.

4 Conclusion

We introduced a promising approach to combinatorial data anonymization by enhancing the basic k-Anonymity problem with user-provided "suppression patterns." It seems feasible to extend our model with weights on the attributes, thus making user influence on the anonymization process even more specific. A natural next step is to extend our model by replacing k-Anonymity by more refined data privacy concepts.

On the experimental side, several issues remain to be attacked. For instance, we use integer linear programming in a fairly straightforward way almost without any tuning tricks (e.g., using the heuristic solution or "standard heuristics" for speeding up integer linear program solving). It also remains to perform tests comparing our heuristic algorithm against methods other than Mondrian (unfortunately, for the others no source code seems freely available).

Acknowledgements. We thank Thomas Köhler and Kolja Stahl for their great support in doing implementations and experiments.

Bibliography

[1] J. Blocki and R. Williams. Resolving the complexity of some data privacy problems. In *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP '10)*, volume 6199 of *LNCS*, pages 393–404. Springer, 2010. (Cited on p. 1)

[2] P. Bonizzoni, G. Della Vedova, and R. Dondi. Anonymizing binary and small tables is hard to approximate. *Journal of Combinatorial Optimization*, 22(1):97–119, 2011. (Cited on p. 1)

[3] P. Bonizzoni, G. Della Vedova, R. Dondi, and Y. Pirola. Parameterized complexity of k-anonymity: hardness and tractability. *Journal of Combinatorial Optimization*, 2011. Available online. (Cited on p. 1)

[4] R. Bredereck, A. Nichterlein, R. Niedermeier, and G. Philip. Pattern-guided data anonymization and clustering. In *Proceedings of the 36th International Symposium on Mathematical Foundations of Computer Science (MFCS '11)*, volume 6907 of *LNCS*, pages 182–193. Springer, 2011. (Cited on pp. 2 and 5)

[5] R. Bredereck, A. Nichterlein, R. Niedermeier, and G. Philip. The effect of homogeneity on the computational complexity of combinatorial data anonymization. *Data Mining and Knowledge Discovery*, 2012. Available online. (Cited on pp. 1 and 5)

[6] A. Campan and T. M. Truta. Data and structural k-anonymity in social networks. In *Proceedings of the 2nd ACM SIGKDD International Workshop on Privacy, Security, and Trust in KDD (PinKDD '08)*, volume 5456 of *LNCS*, pages 33–54. Springer, 2009. (Cited on p. 1)

[7] V. T. Chakaravarthy, V. Pandit, and Y. Sabharwal. On the complexity of the k-anonymization problem. *CoRR*, abs/1004.4729, 2010. (Cited on p. 1)

[8] C. Dwork. A firm foundation for private data analysis. *Communications of the ACM*, 54(1):86–95, 2011. (Cited on p. 1)

[9] P. A. Evans, T. Wareham, and R. Chaytor. Fixed-parameter tractability of anonymizing data by suppressing entries. *Journal of Combinatorial Optimization*, 18(4):362–375, 2009. (Cited on p. 3)

[10] A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`. (Cited on pp. 3 and 8)

[11] E. Fredkin. Trie memory. *Communications of the ACM*, 3(9):490–499, 1960. (Cited on p. 5)

[12] B. C. M. Fung, K. Wang, R. Chen, and P. S. Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys*, 42(4):14:1–14:53, 2010. (Cited on p. 1)

[13] A. Gkoulalas-Divanis, P. Kalnis, and V. S. Verykios. Providing k-anonymity in location based services. *ACM SIGKDD Explorations Newsletter*, 12:3–10, 2010. (Cited on p. 1)

[14] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972. (Cited on p. 3)

[15] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE '06)*, pages 25–25. IEEE, 2006. (Cited on pp. 3, 6, and 8)

[16] H. W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8:538–548, 1983. (Cited on p. 6)

[17] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *Proceedings of the 23rd International Conference on Data Engineering (ICDE '07)*, pages 106–115. IEEE, 2007. (Cited on p. 8)

[18] G. Loukides and J. Shao. Capturing data usefulness and privacy protection in k-anonymisation. In *Proceedings of the 2007 ACM Symposium on Applied Computing*, pages 370–374. ACM, 2007. (Cited on pp. 2 and 8)

[19] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. ℓ-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data*, 1(1), 2007. (Cited on pp. 3 and 8)

[20] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In *Proceedings of the 23rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '04)*, pages 223–228. ACM, 2004. (Cited on p. 1)

[21] G. Navarro-Arribas, V. Torra, A. Erola, and J. Castellà-Roca. User k-anonymity for privacy preserving data mining of query logs. *Information Processing & Management*, 48(3):476–487, 2012. (Cited on p. 1)

[22] V. Rastogi, D. Suciu, and S. Hong. The boundary between privacy and utility in data publishing. In *Proceedings of the 33rd International Conference on Very Large Data Bases*, pages 531–542. VLDB Endowment, 2007. (Cited on p. 2)