A Simulated Annealing Approach to Social Network Anonymization

E. Denisa Arsene[1], Rachel G. de Jong[2][0000−0002−2680−2816], Frank W. Takes[2][0000−0001−5468−1030], and Anna L.D. Latour[1][0000−0002−5802−8271]

[1] Department of Software Technology, Delft University of Technology
[2] Leiden Institute of Advanced Computer Science (LIACS), Leiden University

Abstract. Social networks often contain sensitive information. Sharing such networks potentially puts the privacy of individuals in the network at risk. Leaving out unique identifiers, i.e., pseudonymization, is insufficient to ensure node anonymity, as the network structure surrounding a node may disclose its identity. The goal of network anonymization is to modify the structure of a given complex network to minimize the number of nodes with a unique neighborhood structure. In the budgeted version of this problem, where a given number of edge modifications is allowed, existing heuristic approaches offer speed but obtain only a limited increase in anonymity. We propose a new simulated annealing approach for network anonymization, SANA, which gradually removes edges from the original network structure to optimize anonymity. Experimental results on real-world social network datasets show that the proposed algorithm outperforms state-of-the-art methods by anonymizing, on average, over 17 times more nodes. Compared to existing approaches, it does so with equal or better data utility after anonymization. The results presented in this work further pave the way for enabling safe and privacy-aware sharing of social network data by researchers and practitioners.

Keywords: social networks, simulated annealing, privacy, anonymity

1 Introduction

Ensuring privacy when sharing network data is an important task in social network analysis research. This may concern networks of users on a social media platform [12], participants of a real-world acquaintanceship network gathered for social scientific research purposes [15], or even inhabitants of a country [1].

In this paper, we focus on a particularly pressing privacy-related concern, namely *structural privacy* [13], where an individual may be re-identified based on structural features such as their number of connections. An attacker with partial knowledge of the network could re-identify individuals by matching structural patterns, even in a dataset in which unique identifiers have been removed.

In order to mitigate identification risk in networks, *network anonymization* techniques alter the structure of a given input graph (by, e.g., removing, adding or rewiring some edges), making individual nodes indistinguishable from one another, according to a particular anonymity measure.

A central principle in these techniques is *k-anonymity* [17], which requires that each node is indistinguishable from at least $k-1$ others under some structural measure. In this work, we focus on k-anonymity under the (n, m)-*anonymity* measure, which assumes an attacker knows the degree (n) and number of incident triangles (m) of a node. A node is then k-anonymous if there are at least $k-1$ other nodes with the same *signature*, i.e., the same (n, m) value. This measure is elegantly positioned between *degree*, a relatively simple measure [19], and *ego network isomorphism* [5], a very strict attacker scenario. For a detailed comparison of these measures, see [9]. In the context of network anonymization, it is worth noting that the number of edge deletions affects the so-called *data utility*, which can be measured in two ways. First, in terms of how well topological network properties, such as clustering coefficient and giant component size, are preserved after anonymization. Second, utility can be evaluated based on whether performance in downstream network analysis tasks, such as community detection or identifying central nodes, is retained on the anonymized data.

The focus of this work is on anonymization through *edge deletion*. Existing approaches either provide high-quality anonymization, but require a long running time [2], or they provide fast algorithms, but require many edge deletions to substantially improve anonymity [8]. We address this trade-off by proposing a new method, Simulated Annealing for Network Anonymization (SANA), a metaheuristic that escapes local optima while remaining computationally efficient. An empirical evaluation on real-world social networks shows that SANA achieves higher anonymity in the budgeted variant of the anonymization problem than heuristic and machine learning approaches. While this comes at the cost of a modest increase in running time compared to existing approaches, we achieve similar data utility results, thus showcasing the suitability of our approach for efficient and utility-preserving network anonymization.

The remainder of this paper is organized as follows. We discuss related work in Sect. 2 and introduce key concepts in Sect. 3. We then present SANA in Sect. 4, then evaluate it empirically in Sect. 5, and conclude the paper in Sect. 6.

2 Related Work

Several anonymization algorithms that focus on edge deletion have been introduced in the literature. They aim to anonymize the network, minimizing the number of alterations, to preserve data utility as much as possible.

A naive approach is the *edge sampling* (ES) algorithm, which removes edges uniformly at random [13]. Each edge has an equal probability of being deleted, making it an obvious baseline to compare network anonymization algorithms to.

In contrast, the *unique affected* (UA) method guides the deletion process heuristically [8]. It assigns a weight to each edge, based on the number of uniquely identifiable nodes that would be affected by the removal of that edge. It prioritizes the deletion of edges whose deletion maximizes the number of affected uniquely identifiable nodes. This and other heuristic algorithms are fast, but only manage to anonymize a small fraction of nodes in the budgeted setting [8].

A different approach, based on logistic regression (LR), predicts if removing an edge will decrease uniqueness, prioritizing the removal of edges that are expected to decrease overall uniqueness the most [18]. Predictive features include the degrees of an edge's endpoints, their uniqueness, and the number of incident triangles. Finally, genetic algorithms achieve higher anonymity than fast heuristic methods, but require thorough hyperparameter tuning and are slow [2].

Our main contribution is SANA: *Simulated Annealing for Network Anonymization*, which positions between the heuristic and genetic algorithms. SANA requires less running time than genetic algorithms and has fewer hyperparameters to tune, while offering better anonymization than existing heuristic algorithms.

3 Preliminaries

We introduce key concepts, definitions and notation, illustrated with an example.

3.1 Graphs and local structure

We model a social network as an undirected, unweighted, self-loop-free graph $G := (V, E)$. Here, each node $v \in V$ represents a person and each edge $\{v, w\} \in E \subseteq V \times V$ represents a relationship between two people.

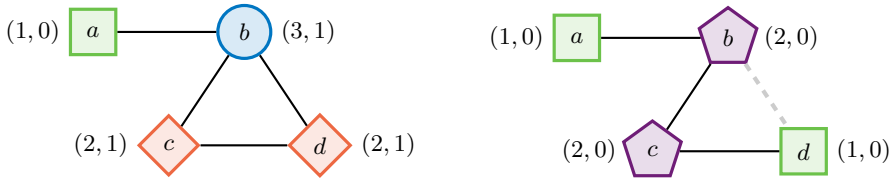
The density of a graph equals the fraction of possible edges that exist in the graph: $density(G) = \frac{2 \times |E|}{|V| \times (|V|-1)}$. We let $\mathsf{deg}(v)$ denote the degree of $v \in V$ and $N(v)$ the set of nodes connected to v. We use $T(v)$ for the number of triangles incident to v, i.e., the number of distinct 3-cycles that include v. We capture the tendency of a node to be part of triangles by its clustering coefficient: $c(v) = \frac{2 \times T(v)}{\mathsf{deg}(v) \times (\mathsf{deg}(v)-1)}$. The average clustering coefficient of a network is given by $\mathrm{ACC} = {}^1\!/\!_{|V|} \sum_{v \in V} c(v)$. We let $\mathsf{dist}(v, w)$ (*distance*) denote the length of the shortest path between nodes $v, w \in V$. If there is no path between them, we let $\mathsf{dist}(v, w) = \infty$. This is the case when two nodes are in different *components*. Most nodes of a network are in the *largest connected component* (LCC). The average path length of a graph, $\mathrm{APL}(G)$, equals the average over all finite distances.

Nodes in social networks tend to form *communities*, which can be found with community detection algorithms [11]. We characterize the importance of nodes in a network using centrality, and here focus on *betweenness centrality* [3], which for a given node v equals the fraction of shortest paths that visit node v.

3.2 Attacks, anonymity and uniqueness

We assume an *attack model* in which an attacker has limited information about a person's network structure. Specifically, we assume that an attacker knows the *degree* of a node $v \in V$ (n), and the *number of incident triangles* of v (m).

We use $\sigma_{(n,m)}(v) := (\mathsf{deg}(v), T(v))$ to denote the (n, m)-*signature* [4] of node $v \in V$. We say that two nodes $v, w \in V$ are *equivalent* (denoted by $v \cong_{(n,m)} w$) if $\sigma_{(n,m)}(v) = \sigma_{(n,m)}(w)$. We denote the set of nodes equivalent to $v \in V$ as



(a) Nodes a and b have unique signatures: $|EC(a)| = |EC(b)| = 1 < 2$.

(b) Removing edge (b, d) makes the network 2-anonymous.

Fig. 1: 2-Anonymizing a small network. Nodes are annotated with their (n, m)-signatures. Node shapes and colors represent membership of equivalence classes.

$EC(v)$ and let $EC_{(i,j)} := \{v \in V \mid \sigma_{(n,m)}(v) = (i, j)\}$ denote the *equivalence class* of nodes with (n, m)-signature (i, j). The network in Fig. 1a partitions the nodes into three equivalence classes: $EC_{(1,0)} = \{a\}$ (nodes with degree 1 and 0 incident triangles), $EC_{(3,1)} = \{b\}$, and $EC_{(2,1)} = \{c, d\}$ ($c \cong_{(n,m)} d$).

For $k \in \mathbb{N}^+$, we say that a *node* $v \in V$ is *k-anonymous* if $|EC(v)| \geq k$. Similarly, we say that a *graph* $G := (V, E)$ is *k-anonymous* if $|EC(v)| \geq k$ for each $v \in V$. We measure the extent to which a graph is k-anonymous using the *uniqueness* measure, i.e., the fraction of nodes that are not k-anonymous:

$$U_k(G) := \frac{|\{v \in V : EC(v) < k\}|}{|V|}$$

In Fig. 1a, nodes a and b are each in an equivalence class of cardinality 1, and c and d in a class of cardinality 2. Hence, for $k = 2$, we have uniqueness $U_2 = 2/4$.

3.3 Network anonymization

The focus of this work is on anonymization through *edge deletion*. We assume a given *budget* $\beta \in \mathbb{N}^+$ (with $\beta \leq |E|$) of edges that we can delete from the input graph G, such that the *residual graph* G' minimizes the graph uniqueness. Hence, the key problem that we study is as follows:

Definition 1 ((n, m) k-Anonymization problem). *Given an undirected, unweighted, self-loop-free, non-empty graph $G := (V, E)$ and $\beta, k \in \mathbb{N}^+$ ($\beta \leq |E|$, $k \leq |V|$), find $E' \subseteq E$ s.t. $|E \setminus E'| \leq \beta$ and $G' := (V, E')$ minimizes $U_k(G')$.*

We focus on the most-studied case of $k = 2$ with budget $\beta = \lfloor 0.05 \times |E| \rceil$ [8,13].

Recall the example in Fig. 1. By removing edge $\{b, d\}$, we partition the nodes into two equivalence classes, neither with cardinality 1. The resulting network is 2-anonymous (Fig. 1b). Hence, the network in Fig. 1a can be fully anonymized with an anonymization budget of $\beta = 1$.

3.4 Data utility

Our ultimate goal is to make the anonymized network available for scientific studies. Hence, we desire that anonymized graphs retain key properties and

have high *data utility*. In our experiments in Sect. 5, we therefore measure how much certain network properties change as a consequence of anonymization.

Following the literature [8], we study the change in *number of edges*, *average clustering coefficient* (ACC) and *average path length* (APL). We also measure how the number of nodes in the *largest connected component* (LCC) changes and how the network's community composition is affected, in terms of normalized mutual information [11] of the original and anonymized network's community division. Finally, we assess the overlap of the top-100 most *central* nodes (using betweenness centrality) between the original and anonymized networks.

4 Approach

In this section, we present SANA: Simulated Annealing for Network Anonymization. We provide motivation, pseudocode, and a description of the algorithm.

4.1 Motivation

Simulated Annealing (SA) is a probabilistic optimization method inspired by the physical process of annealing in metallurgy, where controlled cooling reduces structural defects [10]. An SA algorithm explores the solution space by making a small modification to the current solution in each iteration. If the modified solution represents an improvement w.r.t. the optimization criterion, it is accepted as the new current solution (exploitation). To escape local optima, SA accepts modified solutions that represent a worse solution than the current one with a small probability (exploration). A decreasing 'temperature' parameter tunes that probability, gradually prioritizing exploitation over exploration.

Encouraged by the observation that SA has been successfully applied to a related *k-anonymity* problem [16], we propose to apply SA to solving the (n, m) k-Anonymization problem introduced in Definition 1, using two possible graph modifications: 1) removing an edge $\{u, v\}$ from the network (if the budget constraint allows), or 2) adding it back in. SA can only be applied to problems with small jumps in the cost function between iterations [6]. Our objective function is the network uniqueness score U_k, which may not change much, even if many equivalence classes are affected by an edge modification. In preliminary experiments, we indeed found that applying the above two operations rarely produces a large jump in uniqueness score. Specifically, we found that a single edge modification anonymizes at most 0.75% of nodes. Hence, we find that SA is a suitable candidate method for solving the network anonymization problem.

4.2 The SANA Algorithm

Algorithm 1 presents the pseudocode for SANA, following best practices as outlined by Franzke & Kosko [7].

Main loop. In each iteration of SANA we generate a new candidate solution (Lines 5, 6) by first selecting an edge from the original network, removing it if

Algorithm 1 Simulated Annealing for Network Anonymization (SANA)

Input: graph $G_{\text{in}} := (V_{\text{in}}, E_{\text{in}})$, budget β, and anonymity parameter $k \in \mathbb{N}^+$, initial temperature T_0, iteration limit I, cooling rate α, patience threshold p
Output: graph $G^* := (V^*, E^*)$ and corresponding uniqueness u^*

1: $G_{\text{cur}} \leftarrow G_{\text{in}}$, $G'_{\text{cur}} \leftarrow G_{\text{in}}$, $G^* \leftarrow G_{\text{in}}$, $u^* \leftarrow U_k(G_{\text{in}})$, $\Delta t_{impr} \leftarrow 0$ ▷ Initialize
2: **for** $t = 1$ to I **do**
3: $\quad \Delta t_{impr} \leftarrow \Delta t_{impr} + 1$
4: $\quad \{v, w\} \sim \mathcal{U}(E_{\text{in}})$ ▷ Sample edge uniformly at random
5: $\quad$ **if** $\{v, w\} \in E_{\text{cur}}$ and $|E \setminus E_{\text{cur}}| < \beta$ **then** $E'_{\text{cur}} \leftarrow E_{\text{cur}} \setminus \{\{v, w\}\}$
6: $\quad$ **else if** $\{v, w\} \notin E_{\text{cur}}$ **then** $E'_{\text{cur}} \leftarrow E_{\text{cur}} \cup \{\{v, w\}\}$ ▷ Generate new candidate
7: $\quad$ **end if**
8: $\quad \Delta u \leftarrow U_k(G'_{\text{cur}}) - U_k(G_{\text{cur}})$ ▷ Compute uniqueness improvement
9: $\quad$ **if** $\Delta u < 0$ **then** ▷ Accept better candidate
10: $\quad\quad G_{\text{cur}} \leftarrow G'_{\text{cur}}$
11: $\quad\quad$ **if** $U_k(G'_{\text{cur}}) < u^*$ **then** $G^* \leftarrow G'_{\text{cur}}$, $u^* \leftarrow U_k(G'_{\text{cur}})$, $\Delta t_{impr} \leftarrow 0$ **end if**
12: $\quad$ **else** ▷ Probabilistically accept worse candidate
13: $\quad\quad \vartheta \sim \mathcal{U}(0, 1)$ ▷ Sample probabilistic threshold from uniform distribution
14: $\quad\quad \eta \sim \mathcal{N}(0, \sigma_n)$ ▷ Sample noise from normal distribution
15: $\quad\quad$ **if** $\vartheta < \exp(-(\Delta u + \eta)/T)$ **then** $G_{\text{cur}} \leftarrow G'_{\text{cur}}$ **end if**
16: $\quad$ **end if**
17: $\quad T \leftarrow T_0 \times \alpha^t$ ▷ Update temperature
18: $\quad$ **if** $u^* = 0$ or $\Delta t_{impr} \geq p$ **then break end if**
19: **end for**
20: **return** (G^*, u^*) ▷ Return best solution found

budget β is not depleted, or adding it back in if it was removed in a previous iteration. We accept the new solution if it improves upon the current solution and probabilistically accept it otherwise (Lines 9–16). At the end of each iteration, we lower the temperature T geometrically, using cooling rate α and iteration counter t, to gradually reduce exploration (Line 17). We terminate if a solution with uniqueness 0 has been found, no improvement has been found for p iterations (Line 18), or if the maximum number of iterations I is reached (Line 2).

Evaluation of the objective function. Evaluating if a new candidate solution is better than the current one (Line 9) requires computing the uniqueness, $U_k(G)$, as described in Sect. 3.2. This requires the computationally expensive computation of the equivalence classes. To decrease the number of computations, we use an *incremental evaluation strategy* where we recompute equivalence class membership only for the nodes that are affected, i.e., for which the (n, m)-signature changes, by deleting or adding the chosen edge. We compute the set of nodes that are affected by adding or deleting an edge as: $\text{Aff}(\{v, w\}) := \{v, w\} \cup (N(v) \cap N(w))$ [8], where $N(v) := \{u \in V \mid \mathsf{dist}(v, u) = 1\}$.

Accepting Candidates. We always accept candidates that decrease uniqueness w.r.t. the current network (Line 9). If such a candidate also improves upon the best-found solution so far, $G^* = (V, E^*)$, we also update the best solution in Line 11. Otherwise, if the solution increases uniqueness, we accept the candi-

Table 1: Characteristics of the networks used in our experiments, including their density, average clustering coefficient, average path length, fraction of nodes in the largest connected component, and initial uniqueness.

Network	$\lvert V\rvert$	$\lvert E\rvert$	Density	ACC	APL	Frac. LCC	U_2
`Copnet SMS` [15]	568	697	0.004	0.139	7.324	0.804	0.026
`FB food pages` [14]	620	2 102	0.011	0.330	5.088	1.000	0.191
`Copnet FB` [15]	800	6 429	0.020	0.315	2.980	1.000	0.472
`CollegeMsg.` [12]	1 899	13 838	0.008	0.109	3.055	0.996	0.239
`Ca-GrQc` [12]	5 242	14 496	0.001	0.530	6.048	0.793	0.055
`Hamsterster` [14]	2 426	16 630	0.006	0.537	3.588	0.824	0.248
`FB ego` [12]	4 039	88 234	0.011	0.606	3.692	1.000	0.587

date probabilistically, based on $\exp\left(-(\Delta u + \eta)/T\right)$, where η is zero-mean Gaussian noise with a standard deviation of σ_n (Line 12).

5 Experiments

We describe our experimental evaluation of SANA's performance on different network datasets, comparing its performance to that of other algorithms in terms of anonymization, data utility and running time. To ensure comparability with prior work, we report on the data utility measures described in the literature [2].

5.1 Experimental Setup

Networks. We evaluate SANA's performance on real-world networks from the network analysis literature, including networks of various sizes, densities, and initial uniqueness. Please refer to Table 1 for details. For our experiments, we preprocessed all networks to make them simple, undirected, unweighted, self-loop free. We used a budget β of 5% of the number of edges in each network.

Baselines. We compare SANA's performance against three baseline methods from the network anonymization literature described in Sect. 2: a uniformly at random edge sampling method (ES) [13], a heuristic edge-deletion method (UA) [8], and one based on Logistic Regression (LR) [18].

Hyperparameters. To determine the values for the hyperparameters, we performed preliminary experiments on the real-world networks. Based on this, we set $T_0 = 0.1$ and $\alpha = 0.75$. We let the iteration budget depend on the edge deletion budget as follows: $I := 100 \times |E|$, except for `FB ego`, where we use $I := 50 \times |E|$. To allow the algorithm to terminate early if no improvement has been seen for a while, we set $p := \min\left(\lfloor p_f \times I \rceil, 8000\right)$. Here, $p_f := 0.3$ for all networks except `Copnet SMS` and `FB ego`, where we use $p_f := 1.0$ and $p_f := 0.2$,

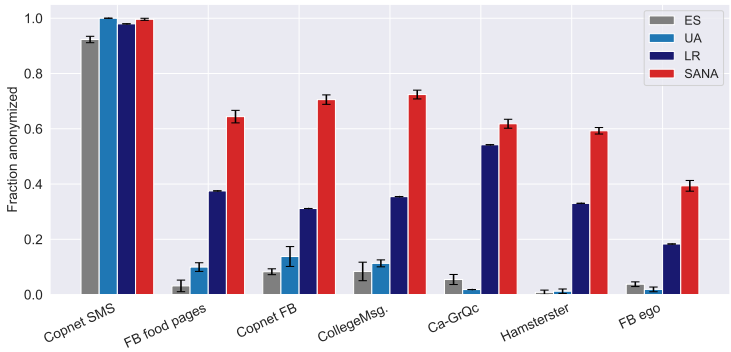


Fig. 2: Relative change in uniqueness (vertical axis) achieved by the algorithms on our test networks (horizontal axis). Higher values indicate more anonymization.

respectively. We include a noise parameter η sampled from a normal distribution with a standard deviation $\sigma_n = 0.0001$. This value is an order of magnitude smaller than Δu. In our preliminary experiments, we found this to achieve better anonymity than when omitting the noise. For LR, ES, and UA we set an iteration budget of $I := \lfloor \beta / r_g \rceil$ and remove r_g (recompute gap) edges per iteration.

Software and Hardware. We implemented all methods in Python 3.12.3 and use igraph 0.11.9 for efficient graph operations. For LR, we use code accompanying the work of Xie [18] with the default settings.[3] We integrate our SANA implementation into the same notebook environment to allow direct comparison under identical setup and evaluation conditions. The source code for our implementation of SANA is publicly available.[4] For the other baseline algorithms ES and UA, we use the C++ ANONET framework.[5]

We used a machine with an Intel i7-7500U CPU (10 cores, 1.80 GHz) and 16 GB RAM. We ran algorithm on a single thread, without parallelization.

5.2 Results

We ran experiments to evaluate the performance of SANA in terms of anonymization, data utility, and running time. To account for the nondeterminism of the algorithms, we average over 5 runs and report ± one standard deviation.

Anonymization. Fig. 2 compares the ability of ES, UA, LR and SANA to anonymize the input network. Due to the differences in initial uniqueness re-

[3] `https://github.com/christine99x/networkAnonymization` (commit `c0cf381` on 21 August 2023).

[4] `https://github.com/arsenedenisa/Simulated-Annealing-for-Network-Anonymization` (commit `56a8cdf` on 22 June 2025).

[5] `https://github.com/RacheldeJong/ANONET` (commit `8089c17` on 16 August 2024).

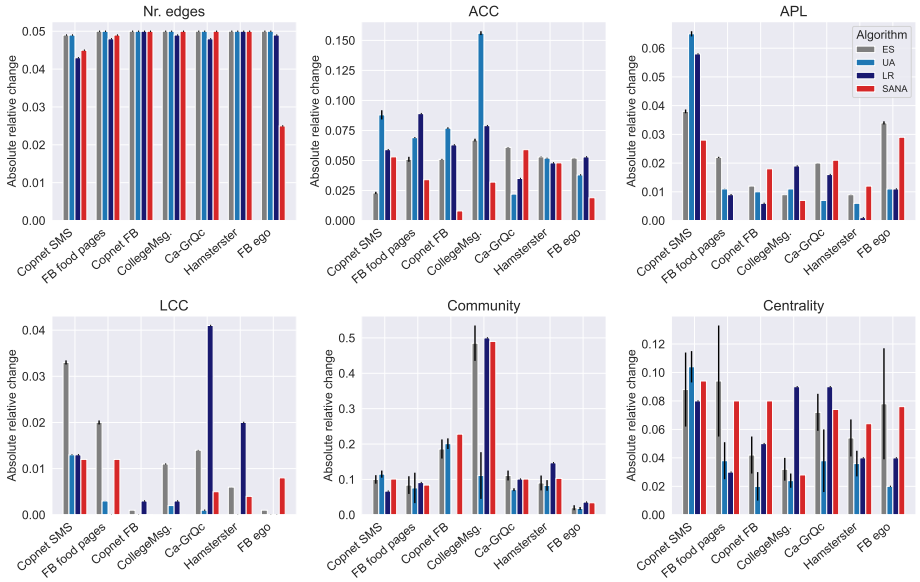


Fig. 3: Absolute values of relative changes in network properties (vertical axes) for the networks from Table 1 (horizontal axes), achieved by the algorithms.

ported in Table 1, we normalize the results by the fraction of initially unique nodes and report the relative improvement. Each value indicates the fraction of initially unique nodes that become anonymous, adjusted for any nodes that turned unique after anonymization. A value of 0 indicates no improvement in anonymity, while a value of 1.0 means all unique nodes became anonymous, resulting in a uniqueness of 0 and all nodes being k-anonymous.

SANA anonymizes on average 18.5 and 17.1 times as many nodes as ES and UA, respectively. Compared to LR, SANA anonymizes 1.8 times as many nodes on average. For the smallest network in our dataset, `Copnet SMS`, UA slightly outperforms SANA, which fails to anonymize the last unique node. SANA performs substantially better than the other algorithms on all other networks.

Utility. Figure 3 shows our data utility preservation results (see Sect. 3.4). The number of edges deleted by each algorithm equals the budgets, except for `Copnet SMS`, for which some runs terminate early as all nodes are anonymous. For average clustering coefficient (ACC) and average path length (APL), all algorithms achieve small differences, with SANA achieving a difference in ACC and APL of at most 5% and 2.5%, respectively.

In general, SANA achieves small (around 1%) changes in LCC, while some algorithms cause much larger changes. On the community and centrality measures, none of the algorithms consistently yield small changes in our experiment, which

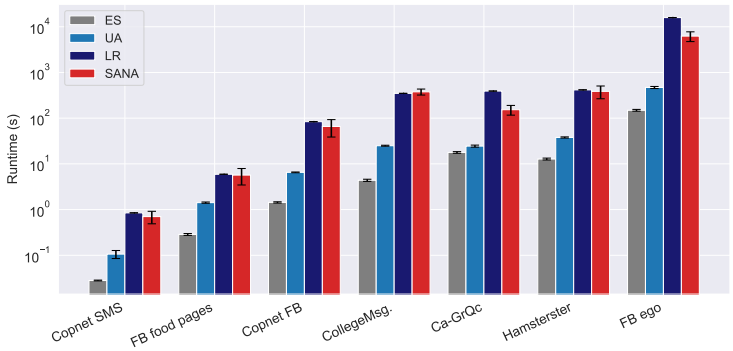


Fig. 4: Running times (vertical axis, logarithmic) of algorithms on the networks (horizontal axis). Error bars represent the standard deviation over 5 runs.

is consistent with findings from the literature [8]. We find that SANA's overall performance on data utility preservation is comparable to the other algorithms.

Running time. Figure 4 shows how the four algorithms compare in terms of running time. SANA requires a higher running time than the baseline algorithms ES and UA, while it is faster than LR on all datasets. While the running time increases with increasing network size for all algorithms, SANA's hyperparameters offer great flexibility in balancing running time and solution quality.

6 Conclusion and Future Work

Our main contribution is SANA: *Simulated Annealing for Network Anonymization*. This algorithm addresses the problem of social network anonymization, where the goal is to reduce the risk of node re-identification, by making nodes anonymous, i.e., sufficiently indistinguishable, under a given anonymity measure. This way, even sensitive social network data can be released publicly for reuse by other researchers and practitioners.

We empirically compared SANA to three algorithms from the literature. Our results demonstrate SANA's superiority w.r.t. the achieved level of anonymity. Moreover, we find that SANA performs on par with other algorithms in terms of preserving data utility, and is only slightly slower in terms of running time.

Further directions of research include a multi-objective optimization setting in which data utility is explicitly optimized by the algorithm. Since social networks tend to evolve over time, real-time on-the-fly anonymity re-optimization algorithms present another important and non-trivial avenue of future research.

7 Acknowledgements

We thank Andrei Ioniţă, Jakub Matyja, Mike J.J.S. Erkemeij, and Emke de Groot for their constructive feedback and discussions. We also thank the anonymous reviewers for their valuable comments.

References

1. Bokányi, E., Heemskerk, E.M., Takes, F.W.: The anatomy of a population-scale social network. Scientific Reports **13**(1), 9209 (2023)
2. Bonello, S., de Jong, R.G., Bäck, T.H.W., Takes, F.W.: Utility-aware social network anonymization using genetic algorithms. CoRR **abs/2504.05183** (2025)
3. Brandes, U.: A faster algorithm for betweenness centrality. Journal of mathematical sociology **25**(2), 163–177 (2001)
4. Burkhart, M., Schatzmann, D., Trammell, B., Boschi, E., Plattner, B.: The role of network trace anonymization under attack. Comput. Commun. Rev. **40**(1), 5–11 (2010)
5. Cheng, J., Fu, A.W., Liu, J.: K-isomorphism: privacy preserving network publication against structural attacks. In: SIGMOD Conference, pp. 459–470. ACM (2010)
6. Davidson, R., Harel, D.: Drawing graphs nicely using simulated annealing. ACM Trans. Graph. **15**(4), 301–331 (1996)
7. Franzke, B., Kosko, B.: Using noise to speed up markov chain monte carlo estimation. In: INNS Conference on Big Data, *Procedia Computer Science*, vol. 53, pp. 113–120. Elsevier (2015)
8. de Jong, R.G., van der Loo, M.P.J., Takes, F.W.: The anonymization problem in social networks. CoRR **abs/2409.16163** (2024)
9. de Jong, R.G., van der Loo, M.P.J., Takes, F.W.: A systematic comparison of measures for k-anonymity in networks. CoRR **abs/2407.02290** (2024)
10. Kirkpatrick, S., Jr., D.G., Vecchi, M.P.: Optimization by simmulated annealing. Sci. **220**(4598), 671–680 (1983)
11. Lancichinetti, A., Fortunato, S.: Consensus clustering in complex networks. CoRR **abs/1203.6093** (2012)
12. Leskovec, J., Krevl, A.: SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data (2014). Accessed: 2023-11-22
13. Romanini, D., Lehmann, S., Kivelä, M.: Privacy and uniqueness of neighborhoods in social networks. CoRR **abs/2009.09973** (2020)
14. Rossi, R.A., Ahmed, N.K.: The network data repository with interactive graph analytics and visualization. In: AAAI (2015). URL https://networkrepository.com
15. Sapiezynski, P., Stopczynski, A., Lassen, D.D., Lehmann, S.: Interaction data from the copenhagen networks study. Scientific Data **6**(1), 315 (2019)
16. Winkler, W.E.: Using simulated annealing for k-anonymity. Tech. rep., U.S. Census Bureau (2002)
17. Wu, W., Xiao, Y., Wang, W., He, Z., Wang, Z.: k-symmetry model for identity anonymization in social networks. In: EDBT, *ACM International Conference Proceeding Series*, vol. 426, pp. 111–122. ACM (2010)
18. Xie, X.: Anonymization algorithms for privacy-sensitive networks (2023). URL https://theses.liacs.nl/2838. Accessed: Feb. 13, 2025
19. Zhou, B., Pei, J.: Preserving privacy in social networks against neighborhood attacks. In: ICDE, pp. 506–515. IEEE Computer Society (2008)