applied sciences

MDPI

Article

FSopt_k: Finding the Optimal Anonymization Level for a Social Network Graph

Maryam Kiabod [1], Mohammad Naderi Dehkordi [1,2], Behrang Barekatain [1,2,*] and Kaamran Raahemifar [3,4,5]

1 Faculty of Computer Engineering, Najafabad Branch, Islamic Azad University, Najafabad 85141-43131, Iran
2 Big Data Research Center, Najafabad Branch, Islamic Azad University, Najafabad 85141-43131, Iran
3 Data Science and Artificial Intelligence Program, College of Information Sciences and Technology (IST), Penn State University, State College, PA 16801, USA
4 School of Optometry and Vision Science, Faculty of Science, University of Waterloo, 200 University Ave. W, Waterloo, ON N2L 3G1, Canada
5 Department of Chemical Engineering, Faculty of Engineering, University of Waterloo, 200 University Ave. W, Waterloo, ON N2L 3G1, Canada
* Correspondence: behrang_barekatain@iaun.ac.ir

Abstract: k-degree anonymity is known as one of the best models for anonymizing social network graphs. Although recent works have tried to address the privacy challenges of social network graphs, privacy levels are considered to be independent of the features of the graph degree sequence. In other words, the optimal value of k is not considered for the graph, leading to increasing information loss. Additionally, the graph may not need a high privacy level. In addition, determining the optimal value of k for the graph in advance is a big problem for the data owner. Therefore, in this paper, we present a technique named FSopt_k that is able to find the optimal value of k for each social network graph. This algorithm uses an efficient technique to partition the graph nodes to choose the best k value. It considers the graph structure features to determine the best privacy level. In this way, there will be a balance between privacy and loss in the anonymized graph. Furthermore, information loss will be as low as possible. The evaluation results depict that this algorithm can find the optimal value of k in a short time as well as preserve the graph's utility.

Keywords: privacy preserving; k-degree anonymity; social network; graph; utility

check for updates

Citation: Kiabod, M.; Dehkordi, M.N.; Barekatain, B.; Raahemifar, K. FSopt_k: Finding the Optimal Anonymization Level for a Social Network Graph. *Appl. Sci.* **2023**, *13*, 3770. https://doi.org/10.3390/app13063770

Academic Editors: Kamran Siddique, Ka Lok Man and Rizwan Ali Naqvi

Received: 8 February 2023
Revised: 9 March 2023
Accepted: 14 March 2023
Published: 15 March 2023

1. Introduction

Nowadays, one of the most beloved technologies among people is social networks, in which they interact and share their information. This results in a huge amount of data on social networks, including users' private and sensitive information. As a result, data owners must be cautious about users' information and privacy. Preserving the users' privacy is carried out by hiding their identities in the social network graph. Therefore, the problem of hiding users' identities has received more attraction. The way to hide a user's identity is by anonymizing the social network attributes before publishing them on the network [1]. This technique benefits from astonishing the adversary in order to prevent discovering users' identities and attributes in the social network graph. In 2012, Ferri et al. [2] studied users' concerns in social networks and discovered that up to 90% of users in the social network are concerned about releasing their information to data owners. Additionally, an anonymized 13.5 TB dataset released by Yahoo [3] demonstrates that preserving users' personal privacy in the social network graph is an important issue these days. Therefore, anonymizing the social network graph is necessary before publishing it to analysts. Backstrom et al. [4] showed that naïve anonymization of the graph is not sufficient. They pointed out that an adversary with knowledge about a victim can identify the user's node in the graph. Therefore, there is a need for more advanced methods for the anonymization of the graph.

There are some models for anonymizing social network graphs. One of these models is k-degree anonymity, which protects the graph from attacking the degree of nodes. In this model, it is guaranteed that for each node in the graph, there are at least k-1 nodes with the same degree. Different algorithms were applied to the social network graph to apply this model. All of these algorithms provide the required privacy level (k parameter value in the k-degree anonymity model) in the graph. They also maintain the utility of the graph for analyzing data in the social network graph. However, the value of parameter k must be given to these algorithms as input. The data owner may not be able to determine this value in advance. In fact, the data owner may only be aware of the desired amount of information loss for their social network. In addition, the value of parameter k is determined regardless of the characteristics of the social network graph. As such, a k value may be appropriate for one graph while not good for another graph. In other words, a k value for one graph may have a high level of information loss, but the same k value for another graph may not have much information. Therefore, finding the optimal value of k for a social network graph is an important issue that has not been considered in algorithms so far. To this end, in this paper, we present a method based on the NaFa [5] algorithm that can find the best k value based on the properties of the graph degree sequence and anonymize the graph accordingly. In this method, the maximum amount of k that can be applied to the social network graph so that the social network graph has the desired utility defined by the data owner is determined by the NaFa algorithm.

Since the graph modification technique is widely used to preserve graph privacy, this paper focuses on this technique to anonymize the graph. The purpose of this paper is to introduce a new challenge and propose a solution for it. This challenge is to find the amount of privacy level, or in other words, the amount of graph anonymization, in the k-degree anonymity model according to the characteristics of the graph. In this way, without giving the value of k as an input to the algorithm, as in the previous algorithms, the graph becomes anonymous in such a way that the amount of anonymity applied to the graph is proportional to its characteristics and different from other graphs. In addition, to address this challenge, the algorithm proposed in this paper maintains the utility of the anonymous graph at an acceptable level. The rest of this paper is organized as follows. Section 2 describes some of the related works carried out on graph anonymization. The problem statement is presented in Section 3. Section 4 explains FSopt_k in detail. A case study is provided in Section 5. Section 6 represents the experimental results. Finally, the paper is concluded in Section 7.

2. Related Work

Nowadays, privacy-preserving data publishing is a popular subject. It is employed in both databases and social networks. It benefits from astonishing the adversary in order to avoid him/her discovering the node belonging to a victim in the social network graph. In this section, a brief review of the methods in privacy-preserving graph publishing is represented.

In general, there are three kinds of threats in a social network graph: identity disclosure, attribute disclosure, and link disclosure. Identity disclosure refers to discovering the user's identity that is related to a node in the social network graph. Attribute disclosure occurs when the sensitive attribute of a node is revealed, and link disclosure is related to the exposure of the sensitive relationship between two users in the social network graph. In this paper, identity anonymization is taken into consideration since it is more general than attribute anonymization. If the user's identity is hidden in the social network graph, their attributes will be hidden too.

There are lots of works that have been carried out on identity anonymization in social networks. Generally, these works are divided into three categories: graph modification techniques, generalization techniques, and approaches based on privacy-aware computation [6]. Figure 1 demonstrates a categorization of the privacy-preserving graph publishing techniques. The techniques in each category are discussed in the following paragraphs.

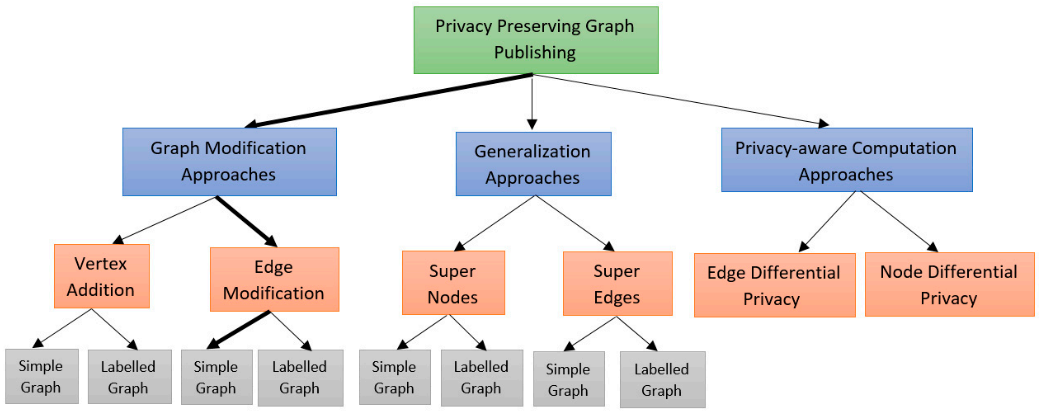

Privacy Preserving Graph Publishing
Graph Modification Approaches
Generalization Approaches
Privacy-aware Computation Approaches
Vertex Addition
Edge Modification
Super Nodes
Super Edges
Edge Differential Privacy
Node Differential Privacy
Simple Graph
Labelled Graph
Simple Graph
Labelled Graph
Simple Graph
Labelled Graph
Simple Graph
Labelled Graph

Figure 1. Categorization of the privacy-preserving graph publishing techniques.

Modification techniques change the graph by adding or deleting some nodes or vertices from the graph. This method adds noise to the nodes or edges of the graph. This technique is employed on weighted and unweighted graphs. In a weighted graph, each edge has a label corresponding to the attributes of the edge. This label can be some criterion to show the strength of the relationship between users. In addition to the edges of the graph, each vertex can have a label too. It can be the users' characteristics, such as his/her age, job, etc. Generalization approaches cluster nodes in the graph to supernodes and edges to super edges. This provides an abstraction of the graph. The anonymous graph has fewer edges and vertices than the original one. Contrary to the increase in anonymity, it decreases the data utility of the graph. Compared with the previous methods, this method has less computational time and can be used on a large network. This method prevents all three types of attacks, i.e., identity disclosure, attribute disclosure, and link disclosure. Finally, privacy-aware computation does not release the graph structure, and the query results are published to the analyzer. One type of privacy-aware computation approach is differential privacy [7]. Differential privacy guarantees that the existence or inexistence of an individual in the dataset does not affect the query results. Therefore, it ensures the user's privacy in the social network graph.

In this paper, the graph modification technique (as shown in Figure 1) is taken into account, which has been a widely used technique recently. The bold line in this figure demonstrates the scope of the current study. This technique is discussed in the following section.

Graph Modification Techniques

Modification techniques are based on the graph structure to achieve anonymity. They modify the structure of the graph so that an adversary cannot discover some information about users in the graph. It is conducted by changing the vertices and the edges of the graph, as shown in Figure 1. One of the popular models in this category is k-anonymity [8,9]. In this model, the probability of identifying an individual in the dataset is constrained to 1/k. This model can be applied to several characteristics of a social network graph. As an example, consider the degree sequence of a graph. In this example, the degree sequence satisfies k-anonymity if there are at least k vertices with the same degree. In 2007, Liu and Terzi used edge addition and deletion to make the graph k-degree anonymous [10]. This method was designed to prevent identity and link disclosure.

Some works on k-anonymity considered k-neighborhood sub-graphs and tried to make them k-anonymous. Zhou [11] made an effort to anonymize the graph in such a way that for each vertex, there are at least k-1 other vertices with the same 1-neighborhood. They used a greedy method to anonymize the labels of vertices. This method adds fake edges to obscure nodes according to their k-neighborhood. The number of vertices in the graph remained unchanged. They also put under consideration attribute disclosure. He et al. [12] considered the local structures of nodes so that after anonymization, the anonymous graph has a structure similar to the original graph. In spite of having a higher privacy level, this algorithm has high complexity. Therefore, it cannot be employed to anonymize large graphs. Tripathy [13] showed that their algorithm does not work when the adversary knows some information about the vertices at a distance of two or higher from a vertex. Therefore, they proposed an algorithm to handle this situation.

Another type of k-anonymity is k-candidate anonymity [14]. A vertex is k-candidate anonymous if there are at least k-1 vertices with the same answer according to a question Q. k-automorphism is an alternative k-anonymity method [15]. A graph has k-automorphism if there are two conditions: the first one is existing k-1 automorphic functions Fa (a = 1, . . . , k-1) in graph G and the second one is that for each vertex vi in graph G, Fa1(vi) $\neq$ Fa2(vi) (1 <= a1 $\neq$ a2 <= k-1). Finding an appropriate automorphic function is an important issue. Therefore, they used three methods to identify this function: graph partitioning, block alignment, and edge copy. Then, they presented a k-automorphic algorithm which is based on these three methods. Contrary to having a high privacy level, this algorithm has greater complexity than the previous works. Therefore, it is not suitable for large networks.

K2-anonymity [16] is another approach that has the purpose of preventing friendship attacks. In a friendship attack, the adversary can detect the victim by checking the degrees of end nodes of the edges in the graph. Therefore, in order to hinder this attack, there should be at least k-1 vertices with the same degrees of the end nodes of the edges incident to a special vertex. They also proposed a heuristic method to anonymize larger networks. K-core attacks [17] violate the user's privacy according to the coreness of the vertices and their degrees in the graph. To deal with this attack, the authors presented a (k, d)-core anonymity technique that anonymizes the vertices based on their k-core characteristics. It applies edge and vertex addition to modify the graph.

On the contrary to the usefulness of the k-anonymity model, it cannot guarantee the privacy of the graph in the case where an attacker knows additional information about the user. Therefore, more complete approaches were presented. One of them is the (k, l)-anonymity model [18]. A graph is (k, l)-anonymous if, for each vertex V in the graph, there are at least k-1 other vertices with at least one common neighbor with that vertex. Stokes et al. [19] discussed some problems of this work. They argued that Feder's algorithm causes risk for users when k $\leq$ l and users can be identified according to their two neighbors. They also propounded a different definition for k-anonymity. According to their definition, a graph is k-anonymous if, for every vertex v1 in the graph, there are at least k other vertices {vi}ki = 1, so that Γ(vi) = Γ(v1) for all i $\in$ [1, k]. They relaxed the definition of (k, l)-anonymity so that a graph is (k, l)-anonymous if for each vertex v $\in$ V, there is a subset of u $\in$ V so that ∣U∣ $\geq$ k and for each u $\in$ U the two vertices u and v have at least one common neighbor.

One problem in the anonymization of social network graphs is the anonymization of hub-like users in the network. The amount of perturbation that needs to be applied to the graph to anonymize these users is high. Therefore, information utility decreases a lot. To overcome this problem, some researchers presented k-subset anonymity that anonymizes just a special subset of vertices so that the number of edges added to the graph is minimized. k-degree-subset-anonymity [20] is a version of this model that anonymizes a subset X $\subseteq$ V. In this work, the number of edges that are added to the graph should be minimized.

Another category of social network graphs is labeled graphs. Das et al. [21] provided a linear model for k-anonymity aimed at maintaining unchanged graph properties as much as possible. In this work, graph properties consist of the shortest path, k-nearest neighbor, and minimum spanning tree. The graph is assumed to be weighted. Kapron et al. [22] focused on the anonymization of the labels in the graph. Its goal is to obtain a graph with an anonymous k-label sequence. The label of each node consists of the tag of the edges incident to it. Additionally, they proved that the problem of k-label sequence anonymity is P for k = 2, but it is NP-hard for k >= 3. Zhou and Pei [23] adapted the notion of k-anonymity and l-diversity to employ a vertex-labeled social network graph. Additionally, they proved that achieving optimal k-anonymity and l-diversity is NP-hard. The same problem was investigated for labels of the edges in the graph [24]. There are some papers that worked on anonymizing the vertex-labeled graph. In this graph, each vertex has an assigned label which contains some information about that vertex.

L-diversity cannot preserve the user′s privacy when the data is skewed [25]. Therefore, t-closeness was proposed to overcome this problem. In t-closeness, it is guaranteed that the distribution of the data values within each group of data is close to the distribution of data in the entire dataset. Contrary to the fact that t-closeness is strong, it cannot be applied explicitly to the graph [26]. Therefore, a method called α-proximity was proposed that preserves the user′s attributes by adding fake edges to the social network graph [26]. Another algorithm was presented by Yuan et al. [27] for k-anonymity and l-diversity on the vertex-labeled graphs. The aim of this method is to change the average path length of the graph as minimally as possible. This algorithm uses fake vertices and edges to anonymize the graph.

Some recent methods benefit from the notion of uncertain graphs. Uncertain graphs differ from other kinds of graphs from the perspective that they have edges with weights in the range of 0 to 1. Each weight in these graphs is given by a function to an edge. In this approach, edge weight is increased or decreased in order to anonymize the network. The first method was presented in 2012 by ref. [28]. In this method, removing an edge was carried out by decreasing the edge weight from 1 to 0, and for adding an edge, the edge weight was increased from 0 to 1. Edge weights are within the range [0, 1]. Therefore, edge weights can be considered edge probabilities. Additionally, they introduced the notion of a (k, ε)-obfuscation graph. A graph is k-obfuscation with respect to a property P if the entropy of the distribution Yp(v) over the vertices of the graph is greater than or equal to log2k. A graph is (k, ε)-obfuscation with respect to P if at least (1 - ε)*n vertices are k-obfuscated. Another method was presented in 2015 by ref. [29]. This method is a generalized view of graph modification based on the notion of the uncertain adjacency matrix of the graph. Another method was provided by the same authors that used maximum variance to obtain a better tradeoff between privacy and data utility [30].

Some works were conducted on accessing private data in social networks. Park et al. [31] proposed a method that assesses private data access for social network security. In this work, the normal private data access pattern is determined in advance. Therefore, any abnormal data access pattern can be identified. It uses Bayesian probability to make normal patterns for private data accesses.

Rousseau et al. [32] also proposed a graph modification technique that changes the edges of the graph by preserving the core of the graph so that the structure of the graph community is preserved. Although this algorithm preserves the core number of vertices after anonymity, by changing the level of the anonymization of the graph, the algorithm must be run from the beginning, which is time-consuming. Siddula et al. [33] provided an overview of privacy techniques and divided existing techniques into two categories: node privacy techniques and edge privacy techniques. Li et al. [34] proposed a user privacy exposure measure to evaluate the privacy of users who contribute to multiple online social networks. In addition, they considered the information loss reduction.

Zhang et al. [35] proposed an algorithm for anonymizing a dynamic directional graph degree sequence that uses the addition of artificial nodes to the graph and, as the node is added, groups it. Then, the nodes that make the least change to the graph structure are connected. In the same year, Siddula et al. [36] considered user-level privacy at the network level rather than the user-level and developed a method that clustered nodes, edges, and attributes by clustering users based on feature similarity. In this method, in addition to the k-anonymity model, the l-diversity model is also considered. Additionally, Kiabod et al. [37] presented a time and memory-saving algorithm which uses a tree structure to anonymize the degree sequence of the graph. Bazgana et al. [38] proved in 2021 that the minimum number of edge rotations to anonymize a social network graph is NP-hard. In the same year, AL-Asbahi [39] proposed an algorithm that anonymizes the social network graph with the least number of edges added to the graph. Singh et al. [40] examined the mutual friends attack model and developed an algorithm that also supports k-degree attacks. Finally, in the same year, Kiabod et al. [41] developed a fast method for modifying social network graphs that ensures the k-degree anonymity model. In this method, an effective method was used to identify and select the appropriate edges for graph editing, which increases the utility of the graph while reducing execution time.

Xiang et al. [42] presented an algorithm for the social network graph k-anonymization problem, which used the tree structure to anonymize the degree sequence of the graph and improve the usefulness of the anonymous graph. In the same year, Ren et al. [43] presented a new technique that reduces the amount of noise applied to the data and reduces the d-neighborhood attack of the graph. In this technique, edges and vertices were anonymized. Table 1 underlines some recent methods in graph modification techniques.

Table 1. The comparison of some recent methods in graph modification techniques.

References	Graph Type	Method	Characteristics
[44]	Simple, undirected	k-degree sequence anonymity	Edge Modification
[32]	Simple, undirected	k-core anonymization	Edge Modification
[36]	Labelled, directed	k-anonymity and l-diversity	Vertex and Edge Modification
[35]	Dynamic, directed	k-in&out-degree anonymity	Vertex and Edge Modification
[37]	Simple, undirected	k-degree sequence anonymity	Edge Modification
[39]	Simple, undirected	k-anonymity and l-diversity	Edge Modification
[40]	Simple, undirected	k-anonymity	Edge Modification
[41]	Simple, undirected	k-degree sequence anonymity	Edge Modification

As shown in Table 1, recently, some works were carried out on k-degree anonymity, and it is still a popular topic. Although these works have achieved good results, they need to be improved. Firstly, the privacy level should be given to the algorithm to be able to anonymize the graph. This level of privacy may not be appropriate for the graph. Applying a high amount of privacy to the graph leads to removing too much useful information from the graph, and its utility decreases. Additionally, considering a small value as the level of privacy leads to the disclosure of sensitive information of users. Therefore, the current paper focuses on achieving an appropriate value of privacy level for the graph with respect to its structural feature. The next section describes it in more detail.

3. Problem Statement

As explained in the introduction, the algorithms presented in the k-degree-anonymity field have the following challenges:

- For all social network graphs, the input k value is determined independently and regardless of the graph structure to be anonymized. This may impose a higher level of privacy than the graph requires, thereby reducing the graph's utility too much. Additionally, the amount of privacy level applied to the graph may not be sufficient and lead to the disclosure of sensitive information of users. For this reason, it is necessary to determine the privacy level of the graph according to the characteristics of that graph and automatically by the algorithm itself, and in this way, a balance between the level of privacy and the utility of the graph is satisfied.
- The value of the k parameter, the anonymization and privacy level, should be specified as the algorithm input, which is often difficult to find the appropriate k value of the graph for the data owner. As explained before, improper selection of k can reduce privacy or increase information loss and reduce graph utility.

Therefore, an algorithm should be presented that has the following properties:

- It should determine the optimal value of the k parameter by the algorithm itself. This way, the data owner's concerns about the appropriate level of graph privacy are decreased.
- In determining the k value, it should consider the features of the graph degree sequence. As discussed before, considering the graph structure in determining the privacy level of the graph can increase the anonymization of sensitive information and decrease information loss. Therefore, the utility of the anonymized graph increases.

For this purpose, in this paper, we present an algorithm that calculates the optimal value of k for a given value of the graph's utility with respect to its degree sequence and then anonymizes the social network based on the level of privacy obtained (i.e., the value of k). In the following section, the steps of this algorithm will be explained and discussed in detail.

4. FSopt_k Algorithm

In this section, the proposed algorithm for solving the problems mentioned in the problem statement section will be presented and explained in detail. For this purpose, we use the following two definitions in this section.

Definition 1. *A graph is anonymous if there are at least k-1 nodes with the same degree for each node in the network [10].*

Definition 2. *The degree of anonymization in the k-anonymity model is defined as the value of k.*

Definition 3. *The optimal value of k is the amount of privacy level that creates the desired value of utility in the graph. The optimal amount of utility should be determined automatically according to the structural characteristics of the graph, and it is different for each graph. Therefore, the optimal value of k is different for each social network graph as well.*

4.1. General Structure of Proposed Method

The proposed algorithm in this paper consists of two steps. In the first step, the algorithm first determines the level of privacy appropriate for the social network graph according to the degree sequence of the graph, and then in the second step, the graph is changed such that the degree of each node is equal to the degree assigned in the first step. The general flowchart of this algorithm is shown in Figure 2.

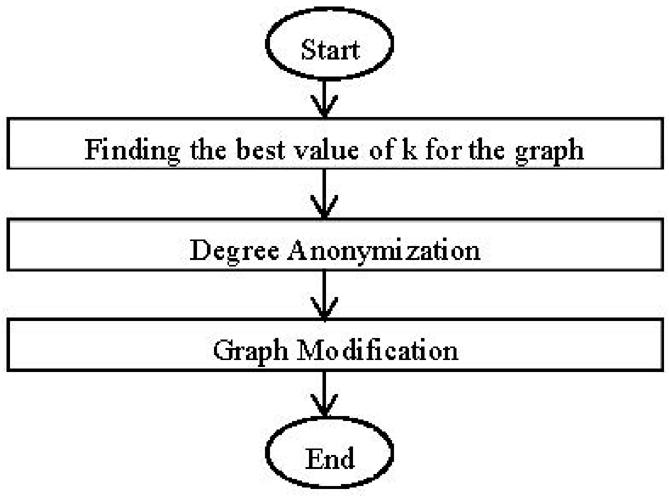


Figure 2. Flowchart of the proposed algorithm.

It should be noted that the graph assumed in this paper is an unordered and unlabeled graph.

4.2. Details of the Proposed Algorithm

In this section, the details of each of the steps in Figure 1 will be explained.

4.2.1. Finding the Optimal Value of k for the Graph

As discussed earlier, choosing the correct value for the k parameter as the privacy level has a great impact on the utility of the anonymous graph. For this reason, an algorithm will be presented in this section that can find the optimal value of k with respect to the degree sequence of the social network. In this paper, we define the optimal value of k as defined by Equation (1).

Definition 4. *Consider the social network graph G. The maximum value of k, which, if applied to graph G, results in the creation of an anonymous graph with desired η, is called the optimal value of k. Desired utility η is calculated by the user-specified coefficient. This factor determines the user's opinion of the percentage of the graph's utility changes. The utility of the graph can be varied for different values of k. This value is calculated by the following formula.*

$$\eta = max(u) - \mathcal{L} \times (\ max(u) - min(u)\ ), \tag{1}$$

where max (u) and min (u) are the maximum and minimum utility of the graph, respectively, and $\mathcal{L}$ is the coefficient of variation of the utility that is determined by the user. The maximum utility is when k = 2, and the minimum utility occurs with the privacy level k = n, where n is the number of graph nodes. The u parameter indicates the utility of the graph. The Formula (14) describes how to obtain it. Then, the optimal value of k can be expressed by the following equation.

$$\text{Optimal}_\text{k}(\text{G}) = \text{Argmin}_\text{k}(\eta - u), \tag{2}$$

where η represents the value of the optimal utility from the user's perspective and u denotes the utility of graph G.

To find the best value of k, first, an aggregate vector of the graph degree sequence is created. In this vector, which is a compact version of the graph degree sequence, each node contains several nodes of the graph degree sequence. This vector is given as input to the NaFa algorithm, and then, the best partitioning for the graph is determined by the NaFa algorithm. Finally, the optimal value of k is determined by this partitioning. Algorithm 1 shows the way of finding the optimal value of k for graph G. Each of the steps of this algorithm will be explained in the following sections. The number written against each step of this algorithm represents the part of the article that describes the details of that step.

Algorithm 1 Finding the optimal value of k

```
Input: graph G
Output: the best value of k
    Create the aggregate vector of G
    Employ NaFa to find the best solution
    Compute the best value of k based on the NaFa solution
    Return k value
```

Creating an Aggregate Vector

The first line of Algorithm 1 creates a compact vector of the graph degree sequence. This compact vector results in faster operation and, thus, lowers the execution time of the algorithm. To this end, we use the method presented in ref. [37]. Each node in this compact vector contains a number of nodes in the same social network graph. If the degree sequence d = {d1, d2, … , dn} is given by the graph, where di is the degree of node vi, to create an aggregate vector, the degree sequence is first ascended so that d1 $\leq$ d2 $\leq$ d3 $\leq \ldots \leq$ dn.

Then, the aggregate vector consists of finding the vertices (n_i, n_j) such that $d_i = d_j$. In this paper, we show the number of vertices with equal degrees in the arranged degree sequence d by *c* which is defined as follows.

$$c_i = \{ \ |\mathrm{U}_{\mathrm{i=1}}^{\mathrm{num}}\mathrm{n_i}| \ \mid \ d_i = d_{i+1} \ \}, \tag{3}$$

where num is the number of consecutive nodes of the same order in the ordered degree sequence of the graph.

Each of the degrees, together with the number of nodes of that degree, forms one element of the aggregate vector. The aggregate vector is defined as follows [37].

$$\mathrm{AV(G)} = \mathrm{U}_{\mathrm{i=1}}^{|\mathrm{c}|}\mathrm{c_i}, \tag{4}$$

where ∣c∣ is the size of the c collection which was constructed by Equation (3). The goal is to find the best partition in this aggregate vector so that we have the least amount of information loss for the highest k value obtained by the algorithm. Since finding the best partitioning in the aggregate vector AV is an NP problem, this paper uses the NaFa algorithm [5] to find it. The steps of this algorithm will be explained in the following sections.

Employing NaFa to Find the Best Solution

After creating the aggregate vector, it can be used to determine the optimal value of k. In order to find the best value of k, all of the cases should be investigated. Therefore, it is an NP problem. Hence, an optimization algorithm is needed to handle this issue. For this purpose, in this paper, the NaFa optimization algorithm [5] is used. This algorithm is an improved version of the firefly algorithm. In NaFA, each firefly is drawn from a predefined neighborhood rather than the entire population to the other brighter fireflies. The rate of firefly i motion in this algorithm is determined by Equation (5).

$$\mathrm{x_{id}(t+1) = x_{id}(t) + \beta\left(x_{jd}(t) - x_{id}(t)\right) + \alpha(t)s_d\epsilon i}, \tag{5}$$

where ϵi is a random number from the interval [−0.5, 0.5], α is the step factor from the interval [0, 1], and s_d is the length scale of each variable described. The β value represents the absorption rate, determined by Equation (6), between two fireflies.

$$\beta = \ \beta_{\min} \ + (\ \beta_0 \ - \beta_{\min} \ )e^{-\gamma r_{ij}^2} \ . \tag{6}$$

In this formula, βmin is the minimum value of β, γ is the light absorption coefficient, and rij is the distance between the two fireflies i and j. The value of α is obtained by Equation (7).

$$\alpha(t+1) = \left(\frac{1}{9000}\right)^{\frac{1}{t}} \alpha(t). \tag{7}$$

In order to use NaFa, position vectors should be initialized. Position vectors in the NaFa algorithm are formed based on the aggregate vector described before. In fact, each element in the position vector contains one element in the aggregate vector. Then, the NaFa algorithm finds the best partition in this vector. The details of each of these steps are described below.

- Initial population creation

The initial population in the NaFa algorithm is the number of x_i position vectors. Suppose Lx is the length vector of the position x_i in this population. The Lx value is equal to ∣AV (G)∣, where AV (G) is the aggregate vector of the graph G and ∣AV (G)∣ shows the length of this vector. For each element in the position vector x, the following relation exists.

$$\forall \mathrm{x_i} \in \mathrm{x},\ \mathrm{x_0} = 1 \ \&\ \mathrm{x_j} \in \{0,\ 1\} \ \big|\ 1 \le \mathrm{j} \le \mathrm{i}. \tag{8}$$

Each element in the position vector x corresponds to one element in the aggregate vector AV, and the value 1 for x_i represents a new group in the partition. The first element in the aggregate vector always starts the first group in the partition, x_0 = 1. Therefore, any group g_k in the *P* partition can be defined as follows.

$$g_k(P) = \{\cup_{i=s_i}^{s_{i+1}-1}(C_i,\ D_i) \in AV(G) \mid x_{si} = 1\ \&\&\forall s_i < j < s_{i+1},\ x_j = 0\}, \tag{9}$$

where *si* is an element index in the position vector x whose value is 1. The elements in the position vector x in the interval (s_i, s_{i+1}) have a value of 0. In this paper, the following formula is used to initialize position vector elements.

$$x_i = rand + norm(noise), \tag{10}$$

where rand is a random number within the range [0, 1]. The numerical noise value is proportional to the value of C_i in the *i*th element of the experimental vector. As mentioned earlier, C_i is the number of degrees i in the graph degree sequence. The reason for using the noise parameter is that the elements of the cumulative vector that have less C_i must be merged with more elements of the cumulative vector to create an anonymous graph. By adding noise to the rand parameter, the element x_i increases proportionally to the C_i value, thereby increasing the probability of merging the lower C_i groups with the other groups. The value of norm (noise) is obtained by the following formula.

$$norm(noise)_i = \frac{c_i - \min(C_i)}{\max(c_i) - \min(c_i)}. \tag{11}$$

Since the NaFa algorithm is a continuous algorithm, these elements are first quantified in the interval [0, 1] and then determined by the formula below the final value of the element x_i in the position vector.

$$x_i = \begin{cases} 1\ if\ x_i \geq 0.5 \\ 0\ if\ x_i \leq 0.5 \end{cases}. \tag{12}$$

Then, for each position vector, the fitness function is calculated, and the best is used to create a new population.

- Fitness function

The fitness function is used to measure the quality of position vectors in a population. The fitness function considered in this paper is the degree of utility of the graph over the optimal utility specified by the data owner. Additionally, since the goal is to find the optimal value of k, the maximum value for which the desired utility in the graph g is obtained is calculated. For this purpose, the fitness function is defined as follows:

$$fitness\ function = |\eta - u| + \frac{1}{k}, \tag{13}$$

where *u*(G) is the utility value obtained for graph G by the FSopt_k algorithm, and η is the desired value of graph utility which is computed based on Equation (1). The k value represents the privacy level of the anonymous degree sequence. The utility of graph *G* is calculated by the following formula.

$$u(G) = 1 - \left(\frac{IL}{\max(d) - \min(d)}\right) * \left(\frac{1}{n}\right), \tag{14}$$

where *IL* is the information loss value of the graph and is defined as Equation (15). max(*d*) and min(*d*) denote the maximum and minimum in the graph degree sequence. The *n* parameter represents the number of graph nodes.

$$IL(P) = \sum_{i=1}^{\omega} SE(g_i), \tag{15}$$

where ω is the number of groups in partition P, and SE is the sum of the errors of group g_i in partition P. The sum of the errors is defined as follows.

$$SE(g_i) = \sum_{j=1}^{n} |d_j - \widetilde{d}|, \tag{16}$$

where n is the number of degrees in the g group and d is the mean of the degrees in the g group.

Definition 5. *Consider the position vector X. The value of k for vector X is the minimum size of the groups contained therein and is defined as follows.*

$$k = \min|g_i|,\ g_i \in X \ \&\ 0 \le i \le |X|, \tag{17}$$

where g_i represents the g' ith group in position vector X and $|g_i|$ denotes the number of nodes in group g_i. Additionally, $|X|$ indicates the number of groups in position vector X.

$$|g_i| = \{\sum_{i=\alpha}^{\gamma} C_i \mid x_\alpha = 1 \ \&\&\ \forall \alpha < j < \gamma : x_j = 0\}, \tag{18}$$

where α and γ are the indexes of the first and last element in the position vector X, respectively. C_i denotes the number of degrees inside the ith element of the cumulative vector.

- Creating new population

In the NaFa algorithm, some noise is added to each $\mathrm{x_i}$ element in the position vector x to create a new population. Therefore, the value of elements in the new position vector may be out of range [0, 1]. In order to address this problem, the new numbers obtained for the position vector elements are normalized by the following formula to fit into the interval [0, 1].

$$norm(x_i) = \frac{x_{i-min}}{max - min}, \tag{19}$$

where max is the largest number and min is the lowest number in the position vector. The word norm represents the normal value of x_i.

Compute the Best Value of k Based on the NaFa Solution

After the best position vector is obtained by the NaFa algorithm, the optimal value of k is calculated and given as an input to the degree sequence anonymization step. Based on the values of elements in the position vector, groups of degrees are formed. Each group is then assigned a final grade, and all grades in that group are converted to that grade. How to assign the final grade to each group is explained in the next section.

4.2.2. Anonymizing the Graph Degree Sequence

After the best partitioning has been determined by the NaFa algorithm, the target degree d′ is assigned to each node in group g, which belongs to partitioning P. This target degree is the final degree of all the nodes in that group.

We used the method provided by Casas-Roma [44] in 2017 to determine the target degree of each group. In this method, the degree of each node is calculated based on the following two probabilities.

$$p(m_j = m_{j1}) = 1 - \left(\frac{m_{j1}}{m_{j1} + m_{j2}}\right), \tag{20}$$

$$p(m_j = m_{j2}) = 1 - \left(\frac{m_{j2}}{m_{j1} + m_{j2}}\right), \tag{21}$$

where m_j is the group's target degree and m_{j1} and m_{j2} are the group's lower and upper averages.

From two degrees, m_{j1} and m_{j2}, the most probable degree is chosen and assigned to group g. In this way, an anonymous degree sequence is created. Then, in the graph editing step, the social network graph is modified based on this anonymity sequence to achieve privacy.

4.2.3. Graph Modification

Three editing, addition, and edge-switching operations are used to edit the graph [44]. In this method, first, by removing or adding the edge between vertices, the graph sequence is modified so that the sum of the degrees inside it is equal to the sum of the degrees inside the anonymous degree sequence. Then, by the switch operation, the edges of the vertices are changed so that the nodes in the graph have the final degree assigned to them. In this operation, an edge is selected for editing that does not alter the graph structure much. For this reason, a score is calculated for each editable edge, and a lower-value edge is selected for removal or addition. The edge score is calculated based on the following formula [44].

$$\mathrm{NC}\{\mathrm{v}_i, \mathrm{v}_j\} = \frac{\left|\Gamma(v_i) \cup \Gamma(v_j)\right| - \left|\Gamma(v_i) \cap \Gamma(v_j)\right|}{2max(deg)}, \tag{22}$$

where NC indicates the neighborhood centrality and $\Gamma(v_i)$ represents the number of node i neighbors.

4.3. Time Complexity Analysis

In this section, the proposed algorithm in this paper is examined in terms of time complexity. Since the FSopt_k and UMGA algorithms differ in the first step of the algorithm, in this section, only the time complexity of this step of the two algorithms is compared.

Remark 1. *Creating a cumulative vector has a time complexity O(n) such that n is the number of vertices of a graph [37].*

Theorem 1. *The time complexity of the computation of the fitness function in the NaFa algorithm is O (∣AV (G)∣, where ∣AV (G)∣ represents the cumulative vector length.*

Proof. As described in before, the position vector length is ∣AV (G)∣, where AV is an aggregate vector. For the calculation of the fitness function value, each of the aggregate vector elements is examined, and the SSE value is calculated for elements greater than or equal to 0.5. In order to calculate SSE for a group, the value of each element in that group is subtracted from the mean value, and the sum of the results is computed. Thus, the time complexity of computing SSE for group $\mathrm{G_i}$ is the number of elements in the *i*th group. Therefore, the time complexity of SSE for all groups is obtained as follows, where ∣G∣ Shows the number of groups.

$$\mathrm{T(n)} = (|\mathrm{G1}| + |\mathrm{G2}| + |\mathrm{G31}| + \ldots + |\mathrm{Gi}|) = \textstyle\sum_{i=1}^{|G|} |G_i| = |AV(G)|. \tag{23}$$

□

Theorem 2. *Finding the best solution by the NaFa algorithm has time complexity O (∣AV (G)∣).*

Proof. The time complexity of the NaFa algorithm is O (Gmax * N * k * f), where Gmax is the maximum number of iterations, N denotes the population size, and f represents the time complexity of the fitness function [5]. Since the values of Gmax, N, and k are constant in our algorithm, the time complexity of this algorithm is O (f). According to Theorem 1, the time complexity of the computation of fitness function value is (O (∣AV (G)∣). Therefore, the time complexity of this function is (O (∣AV (G)∣). □

Theorem 3. *Finding the k value for the best solution found by the NaFa algorithm is O (∣AV (G)∣).*

Proof. To obtain the value of k for a solution, all elements of the position vector must be investigated, and the number of elements in each group must be calculated. Given the number of elements in the position vector is ∣AV (G)∣, the time complexity of obtaining k is also O∣AV (G)∣. □

Theorem 4. *The time complexity of the FSopt_k algorithm is O(nlogn).*

Proof. In this algorithm, the graph sequence is first sorted, with time complexity O(nlogn). Then, this ordered degree sequence is given to the FSopt_k algorithm as input. This algorithm consists of three steps: creating an aggregate vector, finding the best solution by NaFa, and finding the optimal value of k based on the solution obtained. Therefore, the time complexity of this algorithm consists of the time complexity of these three steps. Therefore, the time complexity of the proposed algorithm in this paper is as follows:

$$T(n) = O(nlogn + n + \mid AV(G)\mid + \mid AV(G)\mid) = O(nlog), \tag{24}$$

where n represents the number of graph nodes. □

5. Case Study

An example is given in this section to provide a better view of the proposed algorithm. Suppose the social network graph has the following ordered sequence:

d = {1, 1, 1, 2, 3, 4, 4, 5}.

The goal is to find the optimal value of k for this degree sequence. As described in Section 4, to find the optimal value of k, an aggregate vector of this degree sequence is first created. Then, by the NaFa algorithm, the best k value is obtained for this aggregate vector. Each of these steps will be explained in the following sections.

5.1. Creating an Aggregate Vector

As described in previous section, the aggregate vector is obtained by aggregating the same degrees in one element of this vector. Therefore, the aggregate vector of degree sequence d is obtained as follows.

AV (G) = {(1, 3), (2,1), (3,1), (4, 2), (5,1)}.

For example, element (1, 3) indicates that degree 1 is repeated three times in sequence d. Then, based on the NaFa algorithm, we obtain the optimal value of k as follows:

5.2. Employing NaFa to Find the Best Solution

As described in before, an initial population is formed on the basis of an aggregate vector, and then the NaFa algorithm finds the best partitioning in this vector.

5.2.1. Creating Initial Population

Each position vector in the initial population is as long as the aggregate vector length. Since the aggregate vector has five elements, the position vector has five elements too. Suppose the vector in Figure 3 is a position vector in the initial population. Each ‘1’ in this position vector represents the creation of a new group in node partitioning. Therefore, node partitioning based on the above vector is as follows. Each group in Figure 3 is represented by a specific color.

As this vector represents, the value of k is 4. The next step is to calculate the value of the fitness function for the initial position vectors.

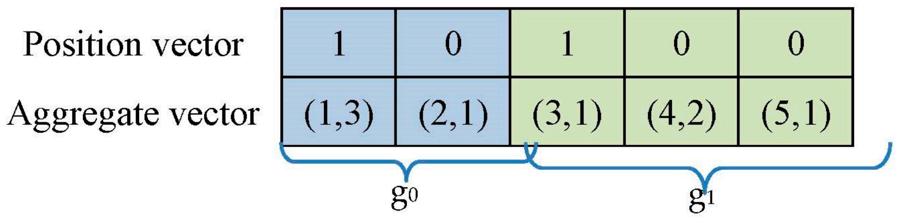


Figure 3. Aggregate vector of position vector.

5.2.2. Fitness Function

Suppose the desired value of graph utility is 0.19. In order to calculate the fitness function, Equation (13) is used, and the cost of the degree sequence anonymization is obtained. The position vectors with lower costs are the best ones. The sum of the errors is used to calculate the anonymization cost in this article. This value is calculated as follows.

$$Avg(g_0) = \frac{1 \times 3 + 2 \times 1}{3 + 1} = 1.25,$$

$$SE(g_0) = \sum_{j=1}^{n} |d_j - \widetilde{d}| = |1 - 1.25| \times 3 + |2 - 1.25| \times 1 = 1.5,$$

$$Avg(g_1) = \frac{3 \times 1 + 4 \times 2 + 5 \times 1}{1 + 2 + 1} = 4,$$

$$SE(g_1) = \sum_{j=1}^{n} |d_j - \widetilde{d}| = |3 - 1| \times 1 + |4 - 4| \times 2 + |5 - 4| \times 1 = 3,$$

$$IL(P) = \sum_{i=1}^{\omega} SE(g_i) = SE(g_0) + SE(g_1) = 1.5 + 3 = 4.5.$$

Now the value of graph utility is calculated as follows.

$$u(G) = 1 - \left(\frac{IL}{\max(d) - \min(d)}\right) * \left(\frac{1}{n}\right) = 1 - \left(\frac{4.5}{5 - 1}\right) * \left(\frac{1}{6}\right) = 0.18.$$

Since the value of k for this vector is 4, the value of the fitness function is obtained as follows.

$$fitness\ function = |\eta - u| + \frac{1}{k} = |0.19 - 0.18| + 0.25 = 0.26.$$

The best position vector is the one with the least fitness function. This vector is used to generate the next generation.

5.2.3. Creating New Population

In the NaFa algorithm, some noise is applied to each element of the position vector to create a new population (Figure 4). Suppose the new elements of the position vector are as follows.

Position vector	1.2	0.12	0.1	0.73	0.21

Figure 4. Position vector.

To obtain the new position vector, the above vector elements are normalized (Figure 5). Therefore, we obtain the following vector.

Position vector	1	0.02	0	0.572	0.1

Figure 5. Normalized position vector.

Since position vector elements can only be 0 or 1, numbers greater than 0.5 are considered 1 and other elements are considered 0. Therefore, we obtain the following vector (Figure 6).

Position vector	1	0	0	1	0

Figure 6. The final position vector.

According to the new position vector, the first three elements in the aggregate vector are in one group, and the next two elements are in another group, as shown in Figure 7.

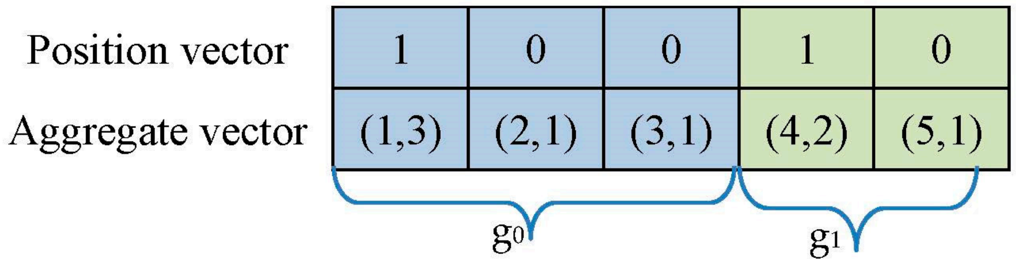


Figure 7. The aggregate vector of position vector.

Therefore, the anonymous degree sequence is as follows.

$$\widetilde{d} = \{\{1,1,1,1\},\{4,4,4\}\}.$$

Given the above anonymous degree sequence, the optimal k value for this degree sequence is k = 3.

6. Experimental Results

FSopt_k is compared with other similar algorithms in this section. To this end, FSopt_k was implemented in Java, and testing took place on a PC with two AMD processors 6134, 2.30 GHz, 24.0 GB RAM, and Windows 10 processors. Because of the use of probability to select the target degree for the nodes, in UMGA and opt k algorithms, each of them was run 10 times. The results presented in Section 6.2. used the igraph package [45] to evaluate FSopt_k. Two aspects of FSopt_k were evaluated; first, the runtime properties of the algorithm, and second, the information loss metric. In the case of some graph properties, such as average path length, transitivity, average clustering coefficient, and edge difference between anonymized and original graphs, information loss examines the difference between the original and anonymous graphs. The datasets used to evaluate FSopt_k, and the used criteria, are listed in the section below.

6.1. Algorithm Runtime Properties

In this section, the proposed algorithm is evaluated from the runtime aspect of the point. As runtime depends on the implementation of the algorithm, and only UMGA code was accessible, FSopt_k was compared to UMGA from the aspect of runtime in this section.

6.1.1. Datasets

Three very large datasets were considered for testing the efficiency of OPT K, as shown in Table 2. Email-Enron [46] is a network which denotes the communication within a dataset of about half a million emails. Network nodes denote email addresses, and each edge represents exchanging emails between two nodes. DBLP [47] is a dataset containing a list of computer science research papers, and Youtube [47] is a video-sharing website that contains friendships within a social network.

Table 2. Attributes of the benchmark datasets.

Characteristics / Data Sets	Nodes	Edges
Email-Enron	36,692	183,831
DBLP	317,080	1,049,866
Youtube	1,134,890	2,987,624

6.1.2. Empirical Results

In order to evaluate the runtime properties of the graph, the optimal value of k obtained by FSopt_k was considered, and the UMGA algorithm was run for it. Then, the runtime was measured. Since UMGA uses an optimal way to partition the nodes, comparing our algorithm with it denotes FSopt_k algorithm performance. We include numerical values in this section to provide a more accurate view of the FSopt_k results. It is worth mentioning that since the first phase of these algorithms, i.e., the degree anonymization phase, is different in these algorithms, we compare only this phase of these algorithms. The results are shown in Table 3. As this table shows, on average, FSopt_k needs 9 min to anonymize the graph's degree sequence. This value is 1 h for UMGA, which is much larger than FSopt_k's runtime. In brief, FSopt_k reduced the runtime needed to anonymize the graph. This results in using the NaFa algorithm to determine the best partitioning of the graph nodes. Contrary to UMGA, NaFa does not build a big portioning graph to partition the nodes. Therefore, the required time for running the FSopt_k algorithm is much less than UMGA.

Table 3. Comparison of FSopt_k and UMGA with respect to the runtime.

Algorithm Data Set	$\mathcal{L}$	FSopt_k (Hours:Minutes:Seconds)	UMGA (Hours:Minutes:Seconds)
Email-Enron	0.05	00:00:09	00:15:36
	0.1	00:00:15	00:25:24
	0.2	00:00:20	00:46:51
DBLP	0.05	00:08:25	1:08:25
	0.1	00:09:42	1:35:28
	0.2	00:09:52	1:58:41
Youtube	0.05	1:13:32	2:39:12
	0.1	1:15:40	3:55:19
	0.2	1:16:21	5:03:06
Mean		00:09:29	1:32:26

6.2. Information Loss Metrics

Information loss metrics evaluate the utility of an anonymous social network graph. In this section, three types of these metrics are used to evaluate the proposed algorithm: the optimal value of k obtained, the metrics for general graph properties, and the metrics for the graph's community structure. The first one concerns the graph's properties, such as the length of the shortest paths, while the second one refers to the community structure of the graph. If the social network graph properties are similar to those of the original graph, the algorithm will be better at retaining the social network graph utility. To test the proposed algorithm in terms of anonymous graph utility, three algorithms are used for k-degree-anonymity, which are similar in characteristics to FSopt_k.

The first one was presented by Chester et al. [48] in 2013, called vertex addition (VA). The next algorithm, KDVEM [49], was presented by MA et al. in 2015, and the third algorithm is UMGA [44], proposed by Casas-Roma in 2017 for k-degree anonymization. In this section, three datasets of the real world are used. Table 4 shows the characteristics of those datasets. The first one is Dolphins [50] which shows the dolphin associations in a community. The second one, netscience [51], is a netscience network of scientists working on network theory and experiments, and the latter is an undirected graph called Power Grid [52].

Table 4. Attributes of the benchmark datasets.

Data Sets	Characteristics	
	Nodes	**Edges**
Dolphins	62	159
Netscience	1589	2742
Polbooks	105	441
Power Grid	4941	6594

6.2.1. Optimal Value of k

In this section, the proposed algorithm is evaluated to determine the optimal value of k for a specified utility value. For this purpose, we compare the proposed algorithm with the UMGA algorithm. As discussed in Section 2, the UMGA algorithm uses a partitioning graph to find the best node partitioning for a value of k. This partitioning graph yields an anonymous degree sequence that is slightly different from the graph degree sequence. In this paper, the value of the desired utility is changed, and the obtained k value by the FSopt_k algorithm is investigated. The higher value for k denotes the algorithm is stronger in anonymizing the degree sequence of the graph. In order to evaluate UMGA, the value of k is increased from 2 to n, where n is the number of nodes in the graph, and then the value of k that resulted in the desired utility was selected. Table 5 represents the evaluation results of the two algorithms on the datasets. As it is seen in this table, FSopt_k resulted in the values of k that are close to the k values by UMGA. However, the k values are lower than UMGA. Our goal in this paper is to reduce the algorithm runtime while preserving other properties of the algorithm. Therefore, it is reasonable to have a little reduction in the graph's utility. However, this amount is not a lot.

Table 5. Evaluation results of FSopt_k and UMGA with respect to the obtained k value and utility.

Algorithms / Data Sets		FSopt_k			UMGA	
	$\mathcal{L}$	**Desired Utility**	**Obtained *k* Value**	**Obtained Utility**	***k* Value**	**Utility**
Dolphins	0.05	0.974633443	10	0.94134	8	0.973607
	0.1	0.94369499	14	0.935483	13	0.938416
	0.2	0.91612901	20	0.9149560	20	0.914956
netscience	0.05	0.997917	10	0.9971	13	0.9979023
	0.1	0.992014	101	0.99005	106	0.9913992
	0.2	0.98426684	219	0.9802	227	0.9844766
Power Grid	0.05	0.998594999	79	0.993636	65	0.9984933
	0.1	0.99334365	248	0.992	224	0.992938
	0.2	0.9869257	520	0.9846	516	0.9861476
Polbooks	0.05	0.9415431	7	0.940935	6	0.95012
	0.1	0.90262082	8	0.89991	7	0.84225
	0.2	0.936106561	11	0.93355	9	0.9512

6.2.2. General Properties of Graph Analysis

In order to evaluate the proposed algorithm in terms of general graph properties, four metrics are evaluated, including average path length, transitivity, average clustering coefficient, and modularity. The more the similarity between the values of these parameters is in the main graph and the anonymous graph, the more the graph's utility can be preserved by the algorithm. These parameters are explained below.

- Average path length (APL) measures the average length of the shortest paths in the social network graph between vertices. This parameter is one of the most important graph analysis criteria and is defined as Equation (25).

$$APL(G) = \frac{1}{n(n-1)} * \sum_{i \neq j} d(v_i, v_j), \tag{25}$$

where d (v_i, v_j) denotes the shortest path length between each of the v_i and v_j nodes, and n indicates the number of graph nodes.

- Transitivity (T) is a further criterion utilized to assess FSopt_k, which measures the relative number of triangles in a graph and is defined as Equation (26).

$$Transitivity = \frac{3 * \Delta}{\wedge}, \tag{26}$$

where Δ refers to the number of triangles, and $\wedge$ denotes the number of paths with length two.

- Average clustering coefficient (ACC) measures the closeness of the neighbors of each of the graph nodes to the full graph. This criterion is defined in Equation (27).

$$ACC(G) = \frac{1}{n} * \sum_{u=1}^{n} \frac{2T(u)}{\deg(u) * (\deg(u) - 1)}, \tag{27}$$

where u shows a node in the graph, n denotes the number of graph nodes, and $deg(u)$ represents the degree of node u.

- Modularity (mod) is one of the criteria used to measure the strength of the network division into modules.

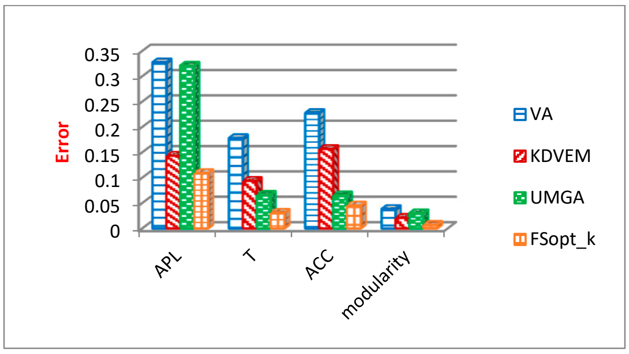
Evaluation Results of General Graph Properties

The results of FSopt_k evaluation from the aspect of general graph properties on the datasets are shown in Figures 8–11. The vertical axis in these figures represents the error value related to the graph's general properties, and the horizontal axis shows the evaluation metrics. The error value is measured by (32) [37] for each metric. The lower the error value is, the better the algorithm performs.

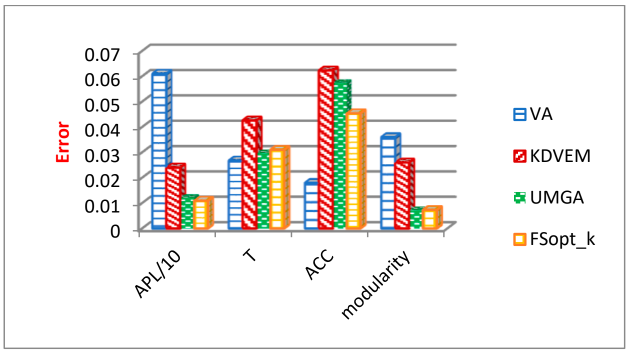
$$\text{Error} = |\text{ATTRIB}_\text{i} - \text{ATTRIB}|, \tag{28}$$

where ATTRIB_i is the value of evaluation criteria evaluated for the optimal k value obtained by FSopt_k. The ATTRIB parameter shows the evaluation criteria.

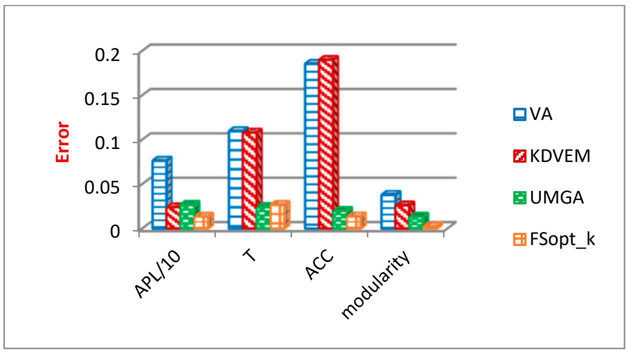
The evaluation results of the algorithms presented in this paper on the Dolphins dataset for different values of L are shown in Figure 8. As the figure shows, the FSopt_k algorithm works better than UMGA in all cases. The reason for this improvement is the use of the coreness and PL criteria when selecting the best edge to remove or add. By increasing the value of $\mathcal{L}$, we have almost similar results on this dataset. For the value of $\mathcal{L}$ = 0.1, the proposed algorithm performs better than other algorithms in most cases. The only criterion in which the accuracy of the algorithm is reduced is the T criterion. In this case, the accuracy of the algorithm is not reduced too significantly. What is clear is that although FSopt_k used better criteria for selecting edges than UMGA in the graph modification process, the accuracy of the algorithm is reduced in some cases. In other words, UMGA uses an optimal partitioning graph to find the best grouping of graph nodes. Therefore, the accuracy of UMGA is very high. However, the proposed algorithm can find the best partitioning of graph nodes using an optimization algorithm with low runtime. Since the purpose of the proposed algorithm in this paper is to reduce the runtime of the anonymization algorithm, it is reasonable to reduce the accuracy of the algorithm. However, as shown in the evaluation results, this decrease in the accuracy of the algorithm is low and does not have much effect on the utility of the anonymous graph.



(**a**)

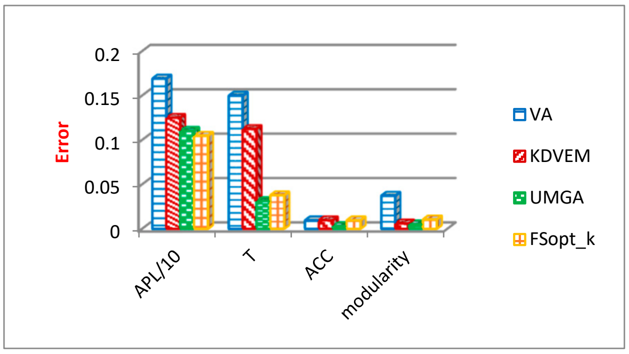


(**b**)

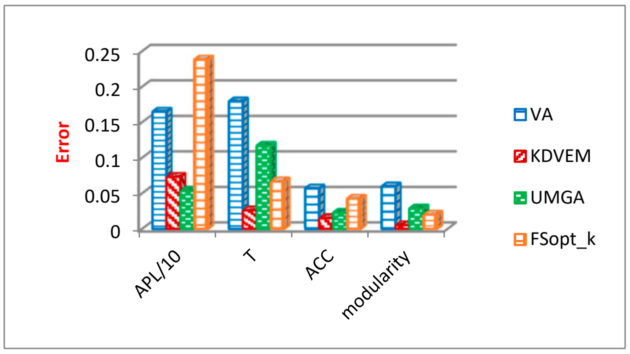


(**c**)

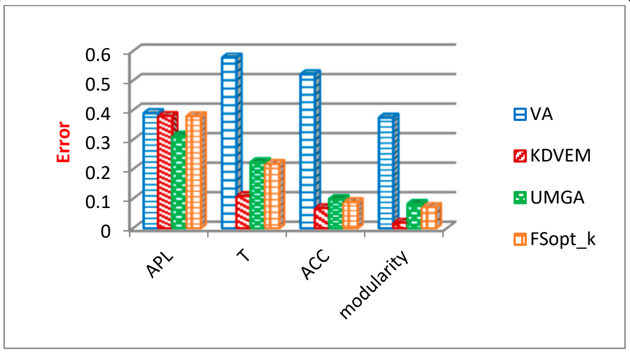
Figure 8. Comparison of error values of the algorithms on Dolphins dataset for (**a**): $\mathcal{L} = 0.05$, (**b**): $\mathcal{L} = 0.1$, and (**c**): $\mathcal{L} = 0.2$.



(**a**)



(**b**)



(**c**)

Figure 9. Comparison of error values of the algorithms on netscience dataset for (**a**): $\mathcal{L} = 0.05$, (**b**): $\mathcal{L} = 0.1$, and (**c**): $\mathcal{L} = 0.2$.

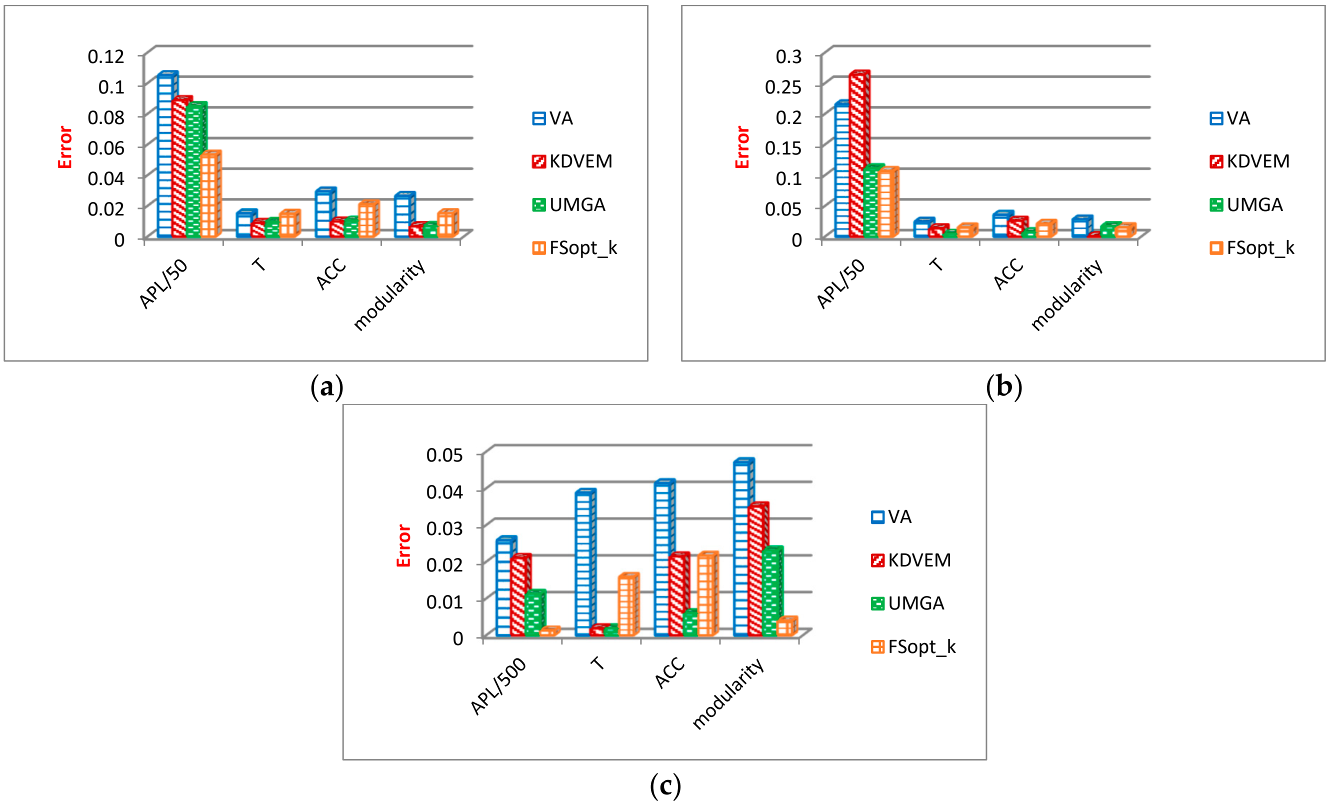


Figure 10. Comparison of error values of the algorithms on power grid dataset for (**a**): $\mathcal{L} = 0.05$, (**b**): $\mathcal{L} = 0.1$, and (**c**): $\mathcal{L} = 0.2$.

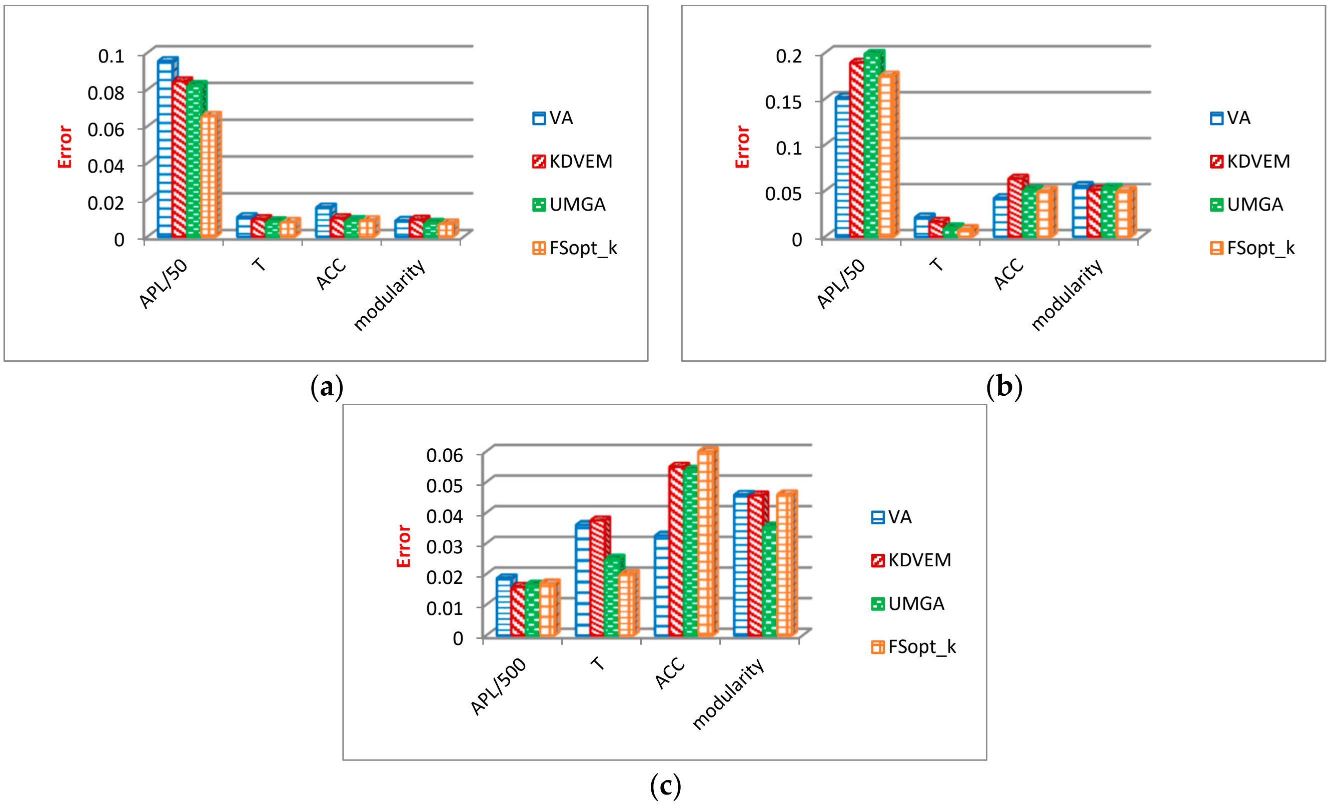


Figure 11. Comparison of error values of the algorithms on polbooks dataset for (**a**): $\mathcal{L} = 0.05$, (**b**): $\mathcal{L} = 0.1$, and (**c**): $\mathcal{L} = 0.2$.

Figure 9 shows the results of the evaluation of the proposed algorithm on the netscience dataset for values of $\mathcal{L}$ = 0.05, $\mathcal{L}$ = 0.1, and $\mathcal{L}$ = 0.2. As can be deduced from this figure, the proposed algorithm has improved in most cases over the existing algorithms. This improvement is more visible for $\mathcal{L}$ = 0.1 and $\mathcal{L}$ = 0.2. As mentioned earlier, a slight decrease in the accuracy of the proposed algorithm is acceptable. In fact, the cost we pay to speed up to anonymize the graph is a reduction in algorithm accuracy. However, in most cases, we see an increase in the accuracy of the algorithm as a result of using PL and coreness criteria in selecting the best edges to remove and add.

The FSopt_k algorithm evaluation results are compared to other algorithms on the power grid dataset for different values of $\mathcal{L}$ in Figure 10. Evaluation results on this dataset show that in the APL and modularity criteria, the accuracy of the proposed algorithm is increased. This increased accuracy is due to the use of coreness and PL criteria in selecting the most appropriate edges in the graph modification process. However, the results of evaluating the algorithms on this dataset show that in the two criteria, ACC and T, the accuracy of the proposed algorithm is reduced.

The results of evaluating the proposed method on the polbooks dataset are shown in Figure 11. This figure shows an improvement in general criteria due to using coreness and PL in finding the best edges to be changed in the graph. The increase in error of $\mathcal{L} = 0.2$ is due to calculating the k value more than what has been obtained in UMGA. In general, the anonymous graph has more than the one obtained by UMGA.

In order to conclude the evaluation results presented in this section, the mean of the errors was obtained for each criterion. The result can be seen in Figure 12. This figure illustrates the improvement of the proposed algorithm over existing algorithms in terms of general graph criteria. This improvement is the result of using appropriate criteria to select the best edges in the graph modification step to be added or removed. In the case of the ACC criterion, the UMGA algorithm performs better than FSopt_k. However, the results of the evaluation of the two algorithms FSopt_k and UMGA are close together. As already explained, the accuracy of the proposed algorithm is reduced. In fact, the UMGA algorithm uses an optimal method to find the best node partitioning for only a value of k, whereas the proposed algorithm obtains the best grouping of graph nodes using an optimization algorithm and then calculates the optimal value of k. Therefore, a slight error in the proposed algorithm is normal.

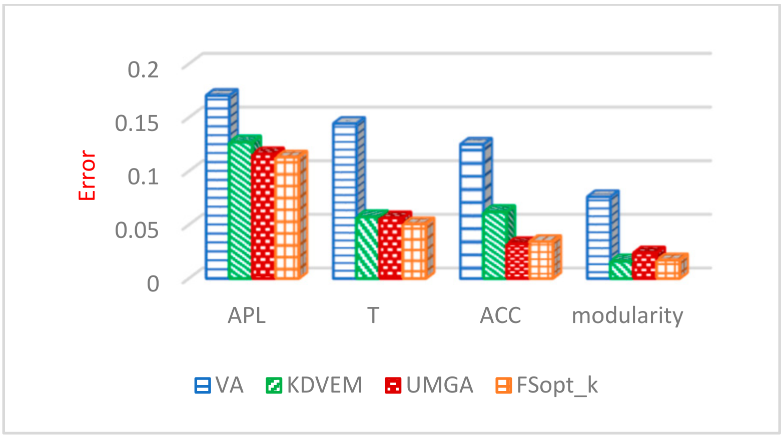


Figure 12. Average of the obtained error values on all datasets.

6.2.3. Analysis of Community Structure Metrics

In order to evaluate the proposed algorithm in this section, the graph community criteria are used, which include the variation of information (VI) [53], normalized mutual information [53] (NMI), the split-join distance [54], the Rand index [55], and the adjusted Rand index [56], to evaluate NaFa4KDA algorithm. When the community structure of the two graphs is alike, the VI and split–join metrics are zero, and the other metrics are one.

The parameters that their best value is one are reversed to provide a better understanding of the evaluation outcomes. Thus, lower values for all metrics in this section indicate better algorithm efficiency.

Given that VA and KDVEM include fake nodes in the social network graph, the number of vector members is different for the two graphs. Therefore, these algorithms are neglected when this condition occurs.

Evaluation Results of Community Structure Metrics

Figure 13 shows the results of the evaluation of the algorithms on the Dolphins dataset for different $\mathcal{L}$ values. As can be seen in this figure, in most cases, the evaluation results show the improvement of the proposed algorithm in terms of structural graph criteria. These results show that in addition to the proposed algorithm being able to find the optimal value of k in a short time, it is able to preserve the structural properties of the social network graph. In other words, in addition to partitioning the graph nodes close to the UMGA algorithm, the use of PL and coreness parameters has further preserved the graph structure after the anonymization operation. However, the evaluation results on this dataset for $\mathcal{L}$ = 0.2 indicate an increase in the error of the proposed algorithm compared to UMGA. We had the same result for the T criterion in this figure. These results show that by applying the FSopt_k algorithm to this dataset for only $\mathcal{L}$ = 0.2, the structure of the anonymous graph is more modified than the UMGA algorithm.

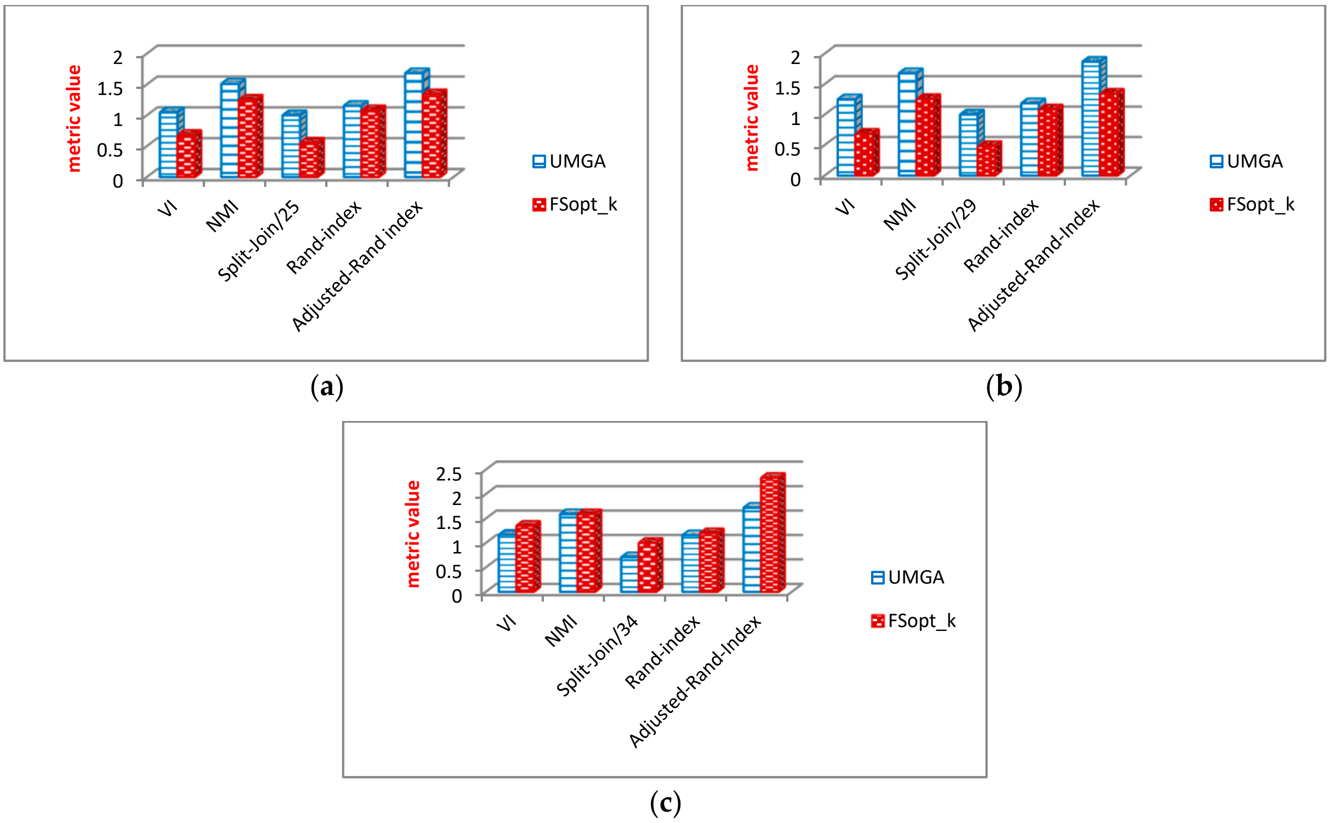


Figure 13. Evaluated values of community structure metrics on Dolphins for (**a**): $\mathcal{L} = 0.05$, (**b**): $\mathcal{L} = 0.1$, and (**c**): $\mathcal{L} = 0.2$.

We obtained similar results for structural graph metrics on the co-authorship dataset. These results are shown in Figure 14. As we infer from these results, the use of the proposed algorithm on this dataset has led to an increase in the utility of the anonymous graph. The results show that the proposed algorithm is able to preserve the properties of the social network graph in addition to finding the optimal value of k at low runtime.

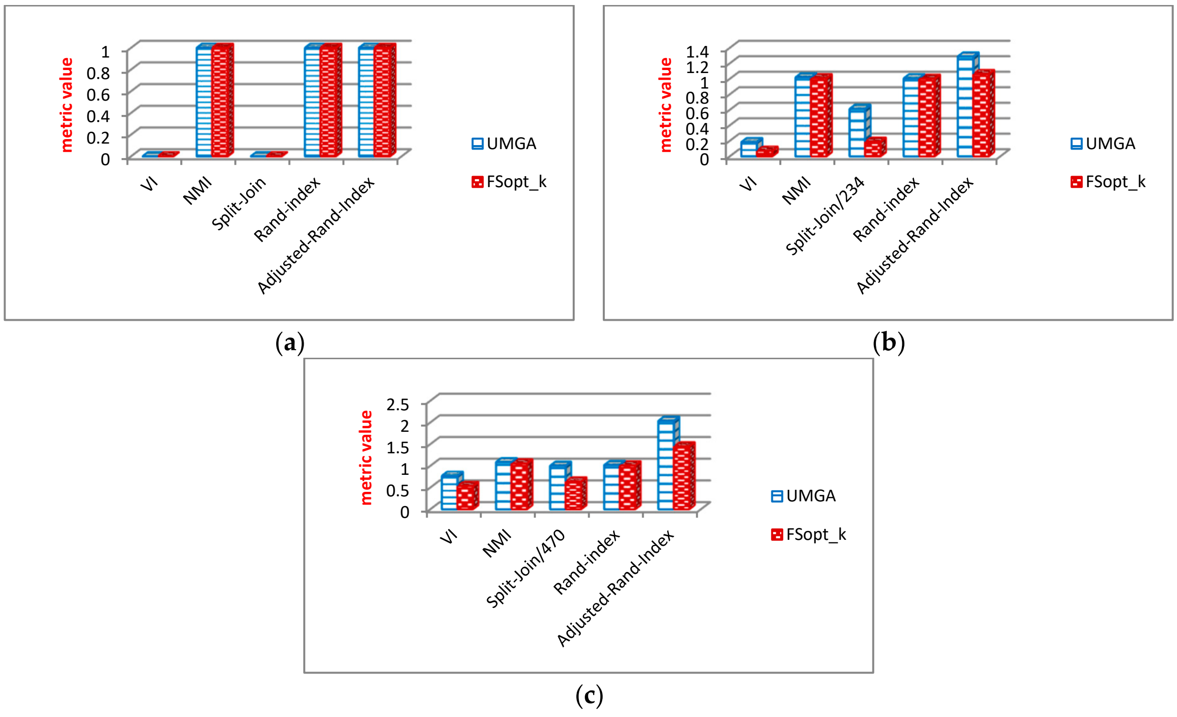


Figure 14. Evaluated values of community structure metrics on netscience for (**a**): $\mathcal{L} = 0.05$, (**b**): $\mathcal{L} = 0.1$, and (**c**): $\mathcal{L} = 0.2$.

The results of the evaluation of FSopt_k on the power grid dataset are shown in Figure 15. As shown in this figure, the error generated in the structural features of the social network graph is less than the UMGA algorithm when applying the FSopt_k algorithm on the graph. Therefore, it can be concluded that the proposed algorithm was effective in reducing the number of changes to the social network graph after anonymization.

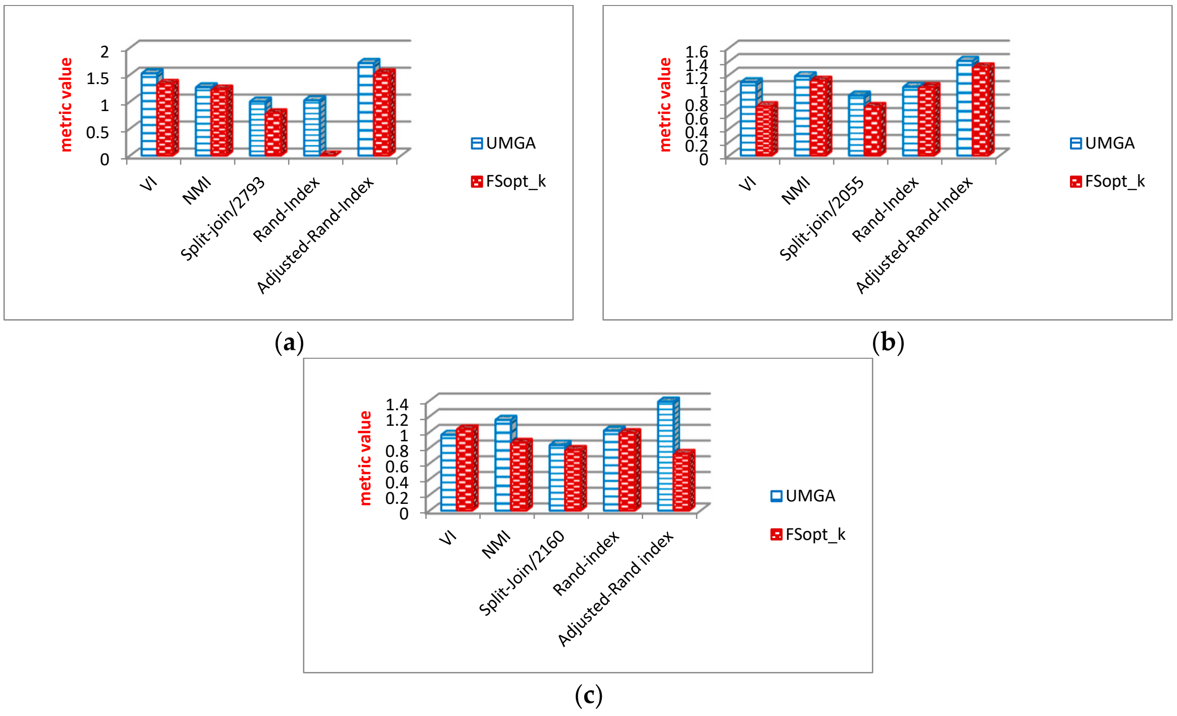
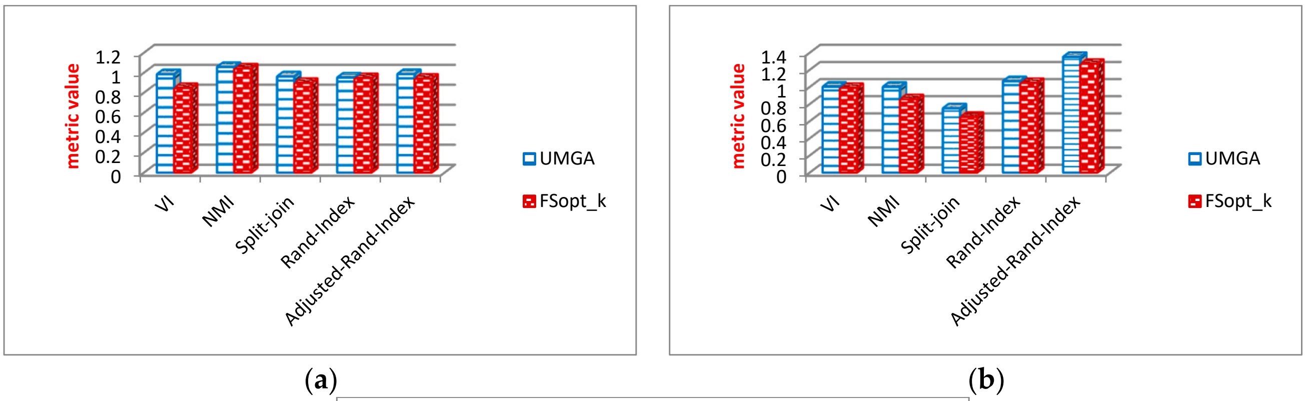


Figure 15. Evaluated values of community structure metrics on power grid for (**a**): $\mathcal{L} = 0.05$, (**b**): $\mathcal{L} = 0.1$, and (**c**): $\mathcal{L} = 0.2$.

Figure 16 denotes the evaluation results of the proposed algorithm on the polbooks dataset. The evaluation results indicate an improvement in anonymous graph utility, which is the result of using more effective criteria to change the graph.



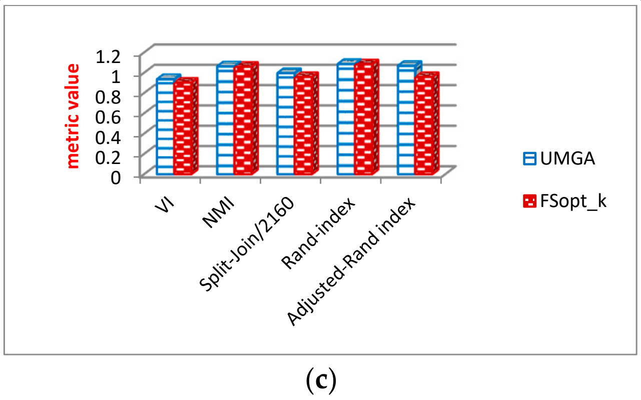


Figure 16. Evaluated values of community structure metrics on polbooks for (**a**): $\mathcal{L} = 0.05$, (**b**): $\mathcal{L} = 0.1$, and (**c**): $\mathcal{L} = 0.2$.

Figure 17 shows the mean values of errors obtained for each criterion on all datasets. As shown in this figure, on average, employing the proposed algorithm on the dataset leads to fewer changes in the social network graph.

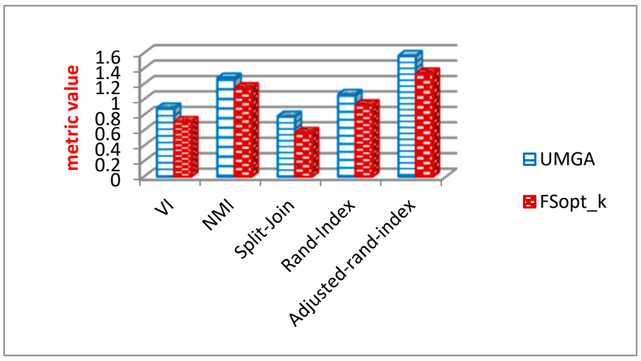


Figure 17. Average of the normalized values for each metric on all datasets.

The test results show that the use of the NaFa optimization algorithm and the application of this algorithm to address the k-degree-anonymity problem leads to a better partitioning of the graph degree sequence. Therefore, the utility of the graph increases in terms of general and structured criteria. In addition, the use of this algorithm has made it possible to determine the level of graph anonymization (i.e., k value) automatically based on the structural characteristics of the graph.

6.3. Comparison of Optimization Algorithms

In this section, the optimization algorithms are compared from the point of view of performance. These algorithms are evaluated from the aspect of the ability to maintain the social network graph properties. At that point, based on the assessment, the best algorithm is chosen to be utilized. For this purpose, each algorithm was executed 10 times, and the average of the assessed values was calculated.

The algorithms compared within the present paper include:

- NaFA [5]: It was introduced in 2017 as an improvement on the Firefly algorithm. The firefly algorithm absorbs fireflies from the entire environment, while the NaFa absorbs fireflies from a special neighborhood.
- PSO [57].
- Bat [58].

6.3.1. Evaluation of Runtime of Optimization Algorithms

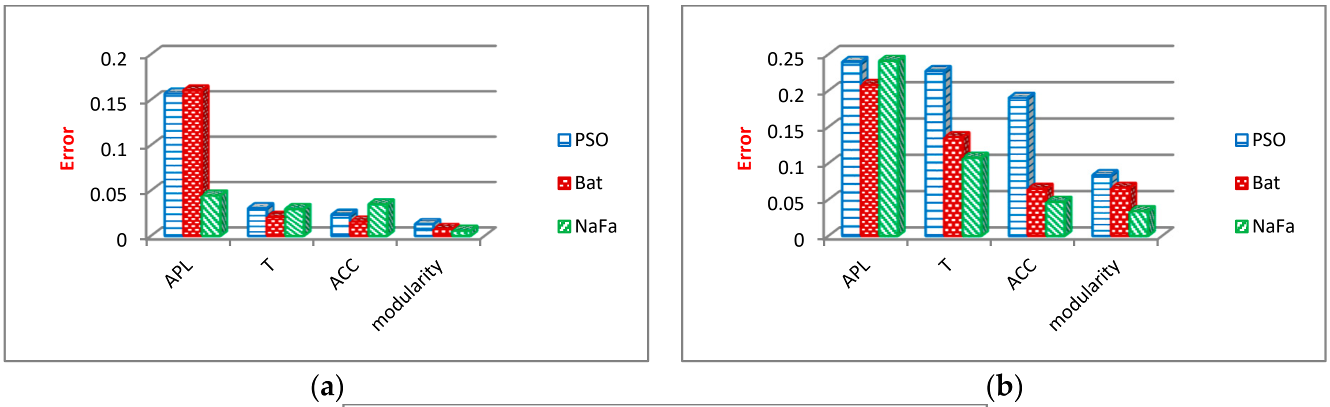
In this section, three popular optimization algorithms are compared in terms of preserving graph properties. Then, the best one is selected as the most appropriate algorithm for our technique. Table 6 shows the evaluation results of the algorithms in terms of k value and utility. The UMGA column shows the optimal value for each of the $\mathcal{L}$ values. The optimization algorithm that produces values closer to the values obtained by UMGA is better suited for use in this paper. The best algorithm in BOLD is specified in each row of this table. As can be seen, in most cases, that the NaFa algorithm performs better than the other two algorithms, and the values obtained by this algorithm are closer to the values obtained by the UMGA.

Table 6. The evaluation results of optimization algorithms with respect to the obtained k value and utility.

Algorithms / Data Set		**UMGA**			**NaFa**		**Bat**		**PSO**	
	$\mathcal{L}$	**Desired Utility**	***k* Value**	**Utility**	**Obtained *k* Value**	**Obtained Utility**	***k* Value**	**Utility**	***k* Value**	**Utility**
Dolphins	0.05	0.9746334	8	0.973607	10	0.94134	12	0.9346	11	0.9395
	0.1	0.9436949	13	0.938416	14	0.935483	9	0.9422	12	0.9332
	0.2	0.9161290	20	0.914956	20	0.9149560	15	0.9227	28	0.9104
netscience	0.05	0.997917	13	0.9979023	10	0.9971	32	0.9862	22	0.9948
	0.1	0.992014	106	0.9913992	101	0.99005	146	0.981	110	0.9861
	0.2	0.9842668	227	0.9844766	219	0.9802	321	0.9772	288	0.9684
Power Grid	0.05	0.9985949	65	0.9984933	79	0.993636	105	0.9892	61	0.9924
	0.1	0.9933436	224	0.992938	248	0.992	301	0.9912	346	0.9916
	0.2	0.9869257	516	0.9861476	520	0.9846	699	0.9821	598	0.9864

6.3.2. Evaluation Results of General Graph Properties

In order to evaluate the optimization algorithms and select the best ones, values of 0.05, 0.1, and 0.2 were considered for the parameter $\mathcal{L}$ and then averaged from the obtained error values. The average evaluation results of the optimization algorithms on the Dolphins dataset are shown in Figure 18a. These results show that the NaFa algorithm performs better than the other algorithms for both APL and modularity criteria, and the error is less. For the T and ACC criteria, the results of the evaluation of the Bat algorithm show less error. Figure 18b denotes the average of the evaluation results of the optimization algorithms on the netscience dataset. In this figure, in general, the amount of error generated in the social network graph features by applying the NaFa algorithm is less than Bat and PSO. We obtained similar results for the power grid dataset. Figure 18c shows the evaluation results of the algorithms on this dataset. As can be seen in this figure, the best algorithm for minimally changing the social network graph is the NaFa algorithm.



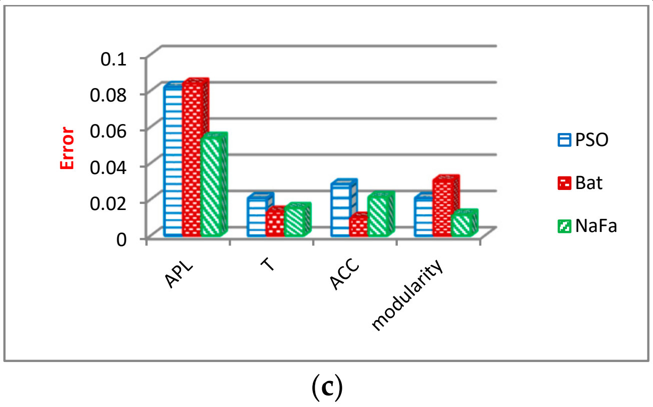


Figure 18. Average of error values of the algorithms for different values of $\mathcal{L}$ on (**a**) Dolphins, (**b**) netscience, and (**c**) power grid.

In order to conclude the evaluation performed on all datasets, the mean values of the errors obtained from the datasets were computed. Figure 19 shows the results. As can be deduced from this figure, on average, the NaFa algorithm performed fewer changes to the social network graph than the other two algorithms.

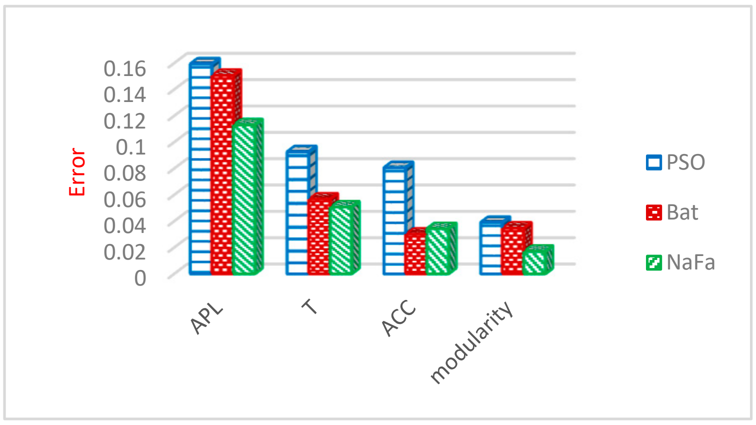
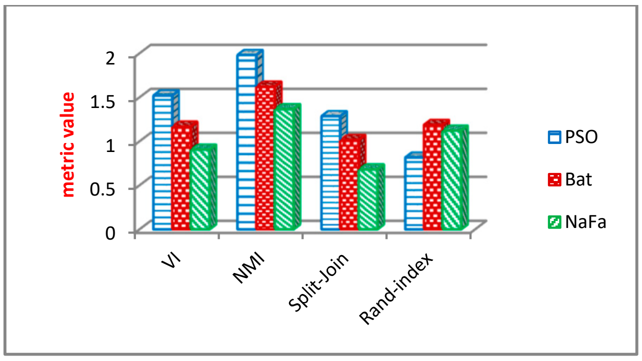


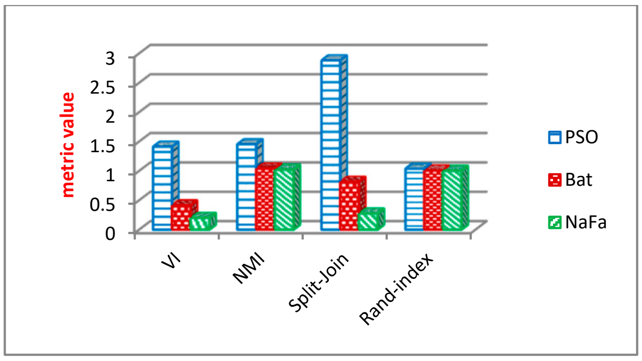
Figure 19. Average of error values on all datasets.

6.3.3. Evaluation Results of Community Structure Metrics

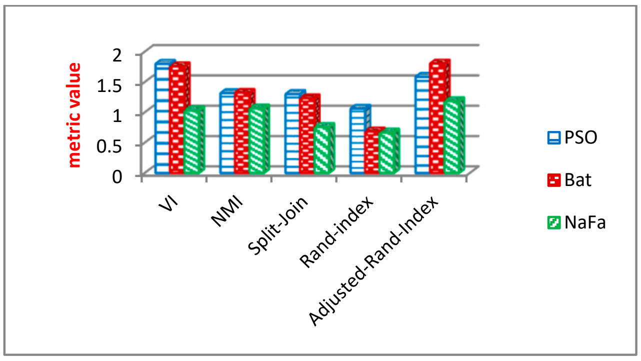
In this section, the optimization algorithms are compared in terms of structural changes in the social network graph. In the following, the evaluation results of these algorithms are investigated on each dataset. The first dataset used is the Dolphins dataset. Figure 20a shows the evaluation results obtained on this dataset. As can be deduced from this figure, in most cases, the NaFa algorithm can better maintain the structural properties of the social network graph compared to the other two algorithms.



(**a**)



(**b**)



(**c**)

Figure 20. Evaluated values of community structure metrics on (**a**) Dolphins, (**b**) netscience, and (**c**) power grid.

Figure 20b shows the results of the evaluation of the optimization algorithms on the netscience dataset. The results of evaluating the optimization algorithms on this dataset show that among the three optimization algorithms, the NaFa algorithm has the least error rate and is, therefore, more suitable for use in the proposed algorithm in this paper. The last dataset used is the power grid dataset. The average of the evaluation results on this dataset can be seen in Figure 20c. As this figure shows, the NaFa algorithm is better able to maintain the utility of the graph in terms of its structural properties than the other two algorithms.

Figure 21 shows the mean of the results on all datasets for the structural criteria. As can be deduced from this figure, on average, the NaFa algorithm performs better than the other two algorithms in terms of maintaining the structural properties of the social network graph.

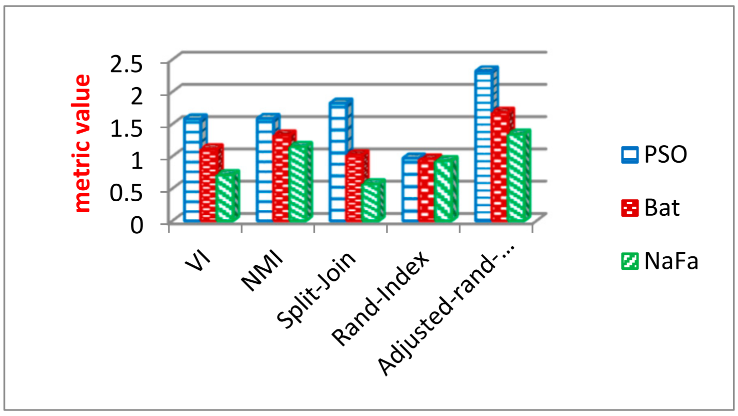


Figure 21. Average of the normalized values for each metric on all datasets.

7. Conclusions

In this paper, an effective algorithm for addressing the optimal k-value challenge for the k-grade anonymization model is presented, which uses an optimization algorithm to find the best partitioning of the graph nodes and results in the least amount of information loss in the graph structure. In this way, the graph structure of the social network is preserved, and the extraction of useful information from the graph is carried out carefully. In addition, the algorithm proposed in this paper can determine the best level of anonymity with respect to the graph structure and based on that, anonymizes the social network graph. Therefore, this algorithm removes the need for the input value k to anonymize the graph. The effectiveness of the proposed method was measured through the runtime, utility metrics, and clustering structure metrics. The results indicate that while this technique reduces the runtime for the graph degree anonymization, it helps in maintaining the graph characteristics close to the original graph as well. In the future, we will consider other models, such as l-diversity, and try to find the best l value for the social network graph.

Author Contributions: Methodology, M.K.; Validation, M.N.D.; Supervision, K.R.; Project administration, B.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Informed Consent Statement: Not applicable.

Data Availability Statement: No new data were created.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Fung, B.C.M.; Wang, K.; Fu, A.W.-C.; Yu, P.S. *Introduction to Privacy-Preserving Data Publishing: Concepts and Techniques*, 1st ed.; Chapman and Hall/CRC: Boca Raton, FL, USA, 2011.
2. Ferri, F.; Grifoni, P.; Guzzo, T. New forms of social and professional digital relationships: The case of Facebook. *Soc. Netw. Anal. Min.* **2012**, *2*, 121–137. [CrossRef]
3. Martin, A.J. Yahoo Dumps 13.5 TB of Users News Interaction Data for Machine Eating. 2016. Available online: https://www.theregister.com/2016/01/14/yahoo_dumps_135tb_of_users_news_interaction_data_for_machine_eating/ (accessed on 1 January 2020).
4. Backstrom, L.; Dwork, C.; Kleinberg, J. Wherefore art thou r3579x? In Proceedings of the 16th International Conference on World Wide Web-WWW'07, Banff, AB, Canada, 8–12 May 2007; Volume 54, p. 181. [CrossRef]
5. Wang, H.; Wang, W.; Zhou, X.; Sun, H.; Zhao, J.; Yu, X. Firefly algorithm with neighborhood attraction. *Inf. Sci.* **2017**, *382–383*, 374–387. [CrossRef]
6. Roma, J.C. *Privacy-Preserving and Data Utility in Graph Mining*; Universitat Autònoma de Barcelona, Departament d'Enginyeria de la Informació i de les Comunicacions: Barcelona, Spain, 2014.
7. Dwork, C. Differential Privacy. In *International Colloquium on Automata, Languages, and Programming*; Springer: Berlin/Heidelberg, Germany, 2006.
8. Samarati, P. Protecting respondents' identities in micro-data release. *IEEE Trans. Knowl. Data Eng.* **2001**, *13*, 1010–1027. [CrossRef]
9. Sweeney, L. K-anonymity: A Model for Protecting Privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* **2002**, *10*, 557–570. [CrossRef]
10. Liu, K.; Terzi, E. Towards identity anonymization on graphs. In Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, Vancouver, BC, Canada, 9–12 June 2008; pp. 93–106. [CrossRef]
11. Zhou, B.; Pei, J. Preserving Privacy in Social Networks Against Neighborhood Attacks. In Proceedings of the 2008 IEEE 24th International Conference on Data Engineering, Cancun, Mexico, 7–12 April 2008; pp. 506–515. [CrossRef]
12. He, X.; Vaidya, J.; Shafiq, B.; Adam, N.; Atluri, V. Preserving privacy in social networks: A structure-aware approach. In Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence, Milan, Italy, 15–18 September 2009; Volume 1, pp. 647–654. [CrossRef]
13. Tripathy, B.K.; Panda, G.K. A new approach to manage security against neigborhood attacks in social networks. In Proceedings of the 2010 International Conference on Advances in Social Network Analysis and Mining, Odense, Denmark, 9–11 August 2010; pp. 264–269. [CrossRef]
14. Hay, M.; Miklau, G.; Jensen, D.; Towsley, D.; Weis, P. Resisting Structural Re-identification in Anonymized Social Networks. *Proc. VLDB Endow.* **2008**, *1*, 102–114. [CrossRef]
15. Zou, L.; Chen, L.; Özsu, M.T. K-Automorphism: A General Framework for Privacy Preserving Network Publication. *Proc. VLDB Endow.* **2009**, *2*, 946–957. [CrossRef]

16. Tai, C.; Yu, P.S.S.; Yang, D.-N.; Chen, M.; Yang, D.-N.; Chen, M. Privacy-preserving social network publication against friendship attacks. In Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining-KDD '11, San Diego, CA, USA, 21–24 August 2011; p. 1262. [CrossRef]
17. Assam, R.; Brysch, M.; Seidl, T. (k, d)-Core Anonymity: Structural Anonymization of Massive Networks. In Proceedings of the 26th International Conference on Scientific and Statistical Database Management, New York, NY, USA, 30 June–2 July 2014; Volume 17, pp. 1–17.
18. Feder, T.; Nabar, S.U.; Terzi, E. Anonymizing Graphs. *CoRR*, abs/0810.5, 1–15, 2008. Available online: http://arxiv.org/abs/0810.5578v1 (accessed on 30 October 2008).
19. Stokes, K.; Torra, V. Reidentification and k-anonymity: A model for disclosure risk in graphs. *Soft Comput.* **2012**, *16*, 1657–1670. [CrossRef]
20. Chester, S.; Gaertner, J.; Stege, U.; Venkatesh, S. Anonymizing subsets of social networks with degree constrained subgraphs. In Proceedings of the 2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, Istanbul, Turkey, 26–29 August 2012; pp. 418–422. [CrossRef]
21. Das s, E.A.A.; Egecioglu, Ö.; Das, S.; Egecioglu, O.; El Abbadi, A. Anonymizing weighted social network graphs. In Proceedings of the Data Engineering (ICDE), 2010 IEEE 26th International Conference on 2010, Long Beach, CA, USA, 1–6 March 2010; pp. 904–907. [CrossRef]
22. Kapron, B.; Srivastava, G.; Venkatesh, S. Social network anonymization via edge addition. In Proceedings of the 2011 International Conference on Advances in Social Networks Analysis and Mining, Kaohsiung, Taiwan, 25–27 July 2011; pp. 155–162. [CrossRef]
23. Zhou, B.; Pei, J. The k-anonymity and l-diversity approaches for privacy preservation in social networks against neighborhood attacks. *Knowl. Inf. Syst.* **2011**, *28*, 47–77. [CrossRef]
24. Chester, S.; Kapron, B.M.; Venkatesh, G.S.S. Complexity of Social Network Anonymization. *Soc. Netw. Anal. Min.* **2013**, *3*, 151–166. [CrossRef]
25. Li, N. T-Closeness: Privacy Beyond k-Anonymity and-Diversity. In Proceedings of the IEEE International Conference on Data Engineering (ICDE), IEEE Computer Society Turkey, Istanbul, Turkey, 15–20 April 2007; pp. 106–115.
26. Chester, S.; Srivastava, G. Social network privacy for attribute disclosure attacks. In Proceedings of the 2011 International Conference on Advances in Social Networks Analysis and Mining, Kaohsiung, Taiwan, 25–27 July 2011; pp. 445–449. [CrossRef]
27. Yuan, M.; Chen, L.; Yu, P.S.; Yu, T. Protecting sensitive labels in social network data anonymization. *IEEE Trans. Knowl. Data Eng.* **2013**, *25*, 633–647. [CrossRef]
28. Boldi, P.; Bonchi, F.; Gionis, A.; Tassa, T. Injecting Uncertainty in Graphs for Identity Obfuscation. *Proc. VLDB Endow.* **2012**, *5*, 1376–1387. [CrossRef]
29. Nguyen, H.H.H.; Imine, A.; Est, L.I.N.; Rusinowitch, M. Anonymizing Social Graphs via Uncertainty Semantics. In Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security (ASIA CCS '15), New York, NY, USA, 14–17 April 2015; pp. 495–506. [CrossRef]
30. Nguyen, H.H.; Imine, A.; Rusinowitch, M. A maximum variance approach for graph anonymization. In *Foundations and Practice of Security. FPS 2014*; Lecture Notes in Computer Science Series; Springer: Cham, Switzerland, 2014; Volume 8930, pp. 49–64. [CrossRef]
31. Park, J.H.; Sung, Y.; Sharma, P.K.; Jeong, Y.-S.; Yi, G. Novel assessment method for accessing private data in social network security services. *J. Supercomput.* **2017**, *73*, 3307–3325. [CrossRef]
32. Rousseau, F.; Casas-Roma, J.; Vazirgiannis, M. Community-preserving anonymization of graphs. *Knowl. Inf. Syst.* **2018**, *54*, 315–343. [CrossRef]
33. Siddula, M.; Li, L.; Li, Y. An Empirical Study on the Privacy Preservation of Online Social Networks. *IEEE Access* **2018**, *6*, 19912–19922. [CrossRef]
34. Li, X.; Yang, Y.; Chen, Y.; Niu, X. A privacy measurement framework for multiple online social networks against social identity linkage. *Appl. Sci.* **2018**, *8*, 1790. [CrossRef]
35. Zhang, X.; Liu, J.; Li, J.; Liu, L. Large-scale Dynamic Social Network Directed Graph K-In&Out-Degree Anonymity Algorithm for Protecting Community Structure. *IEEE Access* **2019**, *99*, 108371–108383. [CrossRef]
36. Siddula, M.; Li, Y.; Cheng, X.; Tian, Z.; Cai, Z. Anonymization in online social networks based on enhanced equi-cardinal clustering. *IEEE Trans. Comput. Soc. Syst.* **2019**, *6*, 809–820. [CrossRef]
37. Kiabod, M.; Dehkordi, M.N.; Barekatain, B. TSRAM: A time-saving k-degree anonymization method in social network. *Expert Syst. Appl.* **2019**, *125*, 378–396. [CrossRef]
38. Bazgana, C.; Cazalsa, P.; Chlebíková, J. Degree-anonymization using edge rotations. *Theor. Comput. Sci.* **2021**, *873*, 1–15. [CrossRef]
39. Al-asbahi, R. Structural Anonymity For Privacy Protection In Social Network. *Int. J. Sci. Res. Publ.* **2021**, *11*, 102–107. [CrossRef]
40. Singh, A.; Singh, M.; Bansal, D.; Sofat, S. Optimised K-anonymisation technique to deal with mutual friends and degree attacks. *Int. J. Inf. Comput. Secur.* **2021**, *14*, 281–299. [CrossRef]
41. Kiabod, M.; Naderi, M.; Barekatain, B. A fast graph modification method for social network anonymization. *Expert Syst. Appl.* **2021**, *180*, 115148. [CrossRef]
42. Ma, N.X.X. TKDA: An Improved Method for K-degree Anonymity in Social Graphs. In Proceedings of the IEEE Symposium on Computers and Communications (ISCC), Rhodes, Greece, 30 June–3 July 2022. [CrossRef]

43. Ren, X.; Jiang, D. A Personalized (*α*,*β*,*l*,*k*)-Anonymity Model of Social Network for Protecting Privacy. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 7187528. [CrossRef]
44. Casas-Roma, J.; Herrera-Joancomartí, J.; Torra, V. k-Degree anonymity and edge selection: Improving data utility in large networks. *Knowl. Inf. Syst.* **2017**, *50*, 447–474. [CrossRef]
45. Csárdi, G.; Nepusz, T. The igraph software package for complex network research. *InterJ. Complex Syst.* **2006**, *1695*, 1–9. [CrossRef]
46. Leskovec, J.; Lang, K.J.; Mahoney, M.W. Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters. *Internet Math.* **2009**, *6*, 29–123. [CrossRef]
47. Yang, J.; Leskovec, J. Defining and Evaluating Network Communities based on Ground-truth. In Proceedings of the 2012 IEEE 12th International Conference on Data Mining, Brussels, Belgium, 10–13 December 2012; pp. 745–754.
48. Chester, S.; Kapron, B.M.; Ramesh, G.; Srivastava, G.; Thomo, A.; Venkatesh, S. Why Waldo befriended the dummy? k-Anonymization of social networks with pseudo-nodes. *Soc. Netw. Anal. Min.* **2013**, *3*, 381–399. [CrossRef]
49. Ma, T.; Zhang, Y.; Cao, J.; Shen, J.; Tang, M.; Tian, Y.; Al-Dhelaan, A.; Al-Rodhaan, M. KDVEM: A k-degree anonymity with vertex and edge modification algorithm. *Computing* **2015**, *97*, 1165–1184. [CrossRef]
50. Lusseau, D.; Schneider, K.; Boisseau, O.J.; Haase, P.; Slooten, E.; Dawson, S.M. The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations: Can geographic isolation explain this unique trait? *Behav. Ecol. Sociobiol.* **2003**, *54*, 396–405. [CrossRef]
51. Newman, M.E.J. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E* **2006**, *74*, 036104. [CrossRef]
52. Watts, D.J.; Strogatz, S.H. Collective dynamics of 'small-world' networks. *Nature* **1998**, *393*, 440–442. [CrossRef] [PubMed]
53. Danon, L.; Díaz-Guilera, A.; Duch, J.; Arenas, A.; Albert, D.; Duch, J. Comparing community structure identification. *J. Stat. Mech. Theory Exp.* **2005**, *09008*, 219–228. [CrossRef]
54. van Dongen, S. *Performance Criteria for Graph Clustering and Markov Cluster Experiments*; Technical Report INS-R0012; National Research Institute for Mathematics and Computer Science: Amsterdam, The Netherlands, 2000.
55. Rand, W.M. Objective Criteria for the Evaluation of Clustering Methods. *J. Am. Stat. Assoc.* **1971**, *66*, 846–850. [CrossRef]
56. Hubert, L.; Arabie, P. Comparing partitions. *J. Classif.* **1985**, *2*, 193–218. [CrossRef]
57. Kennedy, J.; Eberhart, R. Particle Swarm Optimization. In Proceedings of the IEEE International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; pp. 1942–1948.
58. Yang, X. Bat Algorithm: A Novel Approach for Global Engineering Optimization. *Eng. Comput.* **2012**, *29*, 464–483. [CrossRef]