Scalable Algorithms for Facility Accessibility Improvement Problems on Networks

Nguyen T. Thach
School of Computing
University of Nebraska-Lincoln
Lincoln, NE, USA
nate.thach@huskers.unl.edu

Chenhao Wang
Advanced Institute of Natural Sciences
Beijing Normal University
Zhuhai, Guangdong, China
chenhwang@bnu.edu.cn

Hau Chan
School of Computing
University of Nebraska-Lincoln
Lincoln, NE, USA
hchan3@unl.edu

***Abstract*—Facility location problems on networks often deal with locating facilities (e.g., parks, schools, or health facilities) to serve a set of clients (e.g., local citizens). Many existing facility location studies assume that it is always possible to (re-)locate predetermined facilities. However, this assumption is not realistic for facilities that have already been built, where relocating them would be too expensive or impossible. Recognizing this challenge, existing literature proposed the facility accessibility improvement problems (FAIPs) on networks that aim to add a given number of new edges (e.g., constructing new roadways or bridges) to improve the accessibility of the clients to the facilities under the total and maximum accessibility cost objectives. Yet, all existing approaches for FAIPs fail to scale to even moderate-sized problem instances and provide no solution quality guarantees. In this paper, we address these shortcomings by developing scalable (approximation) algorithms and heuristics for FAIPs under the two cost objectives. We then consider the strategic aspects of FAIPs in which clients' locations are not visible publicly. We design scalable truthful algorithms and heuristics to address FAIPs under the two aforementioned cost objectives while incentivizing clients to report their locations truthfully. We conduct experiments on synthetic and real-world networks to demonstrate the effectiveness and efficiency of all proposed methods against various baselines.**

***Index Terms*—network, approximation algorithms, heuristics**

I. Introduction

Facility location problems on networks [1]–[6] are classical problems that have been studied in economics, operations research, and computer science communities for decades. They have numerous applications for modeling and solving various real-world optimization and resource allocation problems, e.g., in urban planning [7]–[12] and preference aggregation [13]–[16]. A typical facility location problem on a network deals with locating a set of facilities (e.g., parks, schools, or libraries) to serve a set of clients (e.g., local citizens) within an infrastructural network to optimize a given cost objective based on the client accessibility costs or distances to the facility locations. These problems have been applied to locating healthcare facilities [17]–[19], locating landfills [20]–[23], clustering [24], [25], and network routing [26]–[28].

Yet, many existing studies in facility location assume that it is always possible to (re-)locate predetermined facilities to account for, e.g., changes in the client population or cost objective. However, this assumption is not realistic for situations in which facilities (e.g., hospitals, landfills, or schools) have already been built, where relocating them would be too expensive and requires considering space, time, legal regulations, costs, and existing facility locations.

Recognizing this challenge, existing facility location optimization literature (e.g., see [29], [30]) proposed the *facility accessibility improvement problems (FAIPs)* on networks. Such problems aim to structurally modify the underlying networks to improve the accessibility of the clients to the facilities under specific cost objectives by adding a given number of edges to the networks. For instance, considering the underlying network as a transportation network, structural modification can be viewed as constructing new roadways or bridges (i.e., adding new network links) to increase the accessibility of various populations to a key facility (e.g., schools and hospitals) [29], [30]. In a communication network, structural modification can be viewed as creating new links between network nodes to increase the communicability of groups of nodes to a router/server/tower [31]–[34]. Analogous interpretations also apply to other contexts such as pipelines, layouts of manufacturing plants, etc [29], [30].

However, the standard FAIPs are strongly NP-hard under the previously considered cost objectives. Moreover, all existing approaches for FAIPs fail to scale to even moderate-sized problem instances (see Section V-D for our experimental evaluations), and provide no solution quality guarantees. For example, in our experiments with real networks, the method of [30] takes quite a long time to deal with just 200 nodes. Therefore, our primary goal is to address these shortcomings by developing scalable and effective algorithms for FAIPs.

A. Our Contributions

In this paper, our goal is to develop effective and scalable algorithms to better inform decision-making for improving the accessibility of the clients to the facilities by manipulating the network infrastructure.

We begin our study by considering the most standard version of facility accessibility improvement problems (FAIPs) on networks [29], [30]. In an FAIP, there is a single facility s pre-located on the network G and n clients.[1] Each client is

[1]The setting with multiple k facilities can be reduced to the single facility setting by creating a dummy node and k edges connecting it directly to each of the k facilities [29].

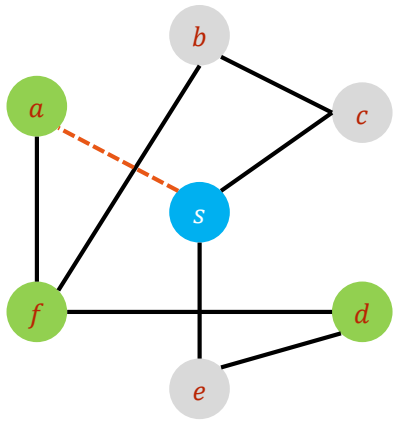

Fig. 1: Toy example of an FAIP wherein the facility s and the clients ($n = 3$) are marked in blue and green, respectively. The red dashed edge is to be added.

positioned or located in one of the network nodes. The FAIP seeks to find k new edges to add to the network such that the given accessibility cost objective of the clients to the facility is minimized. For instance, in a transportation network, adding a few edges can be viewed as increasing the accessibility of some subset of individuals to a key facility [29], [30]. In a communication network, adding a few links can be viewed as creating new links between different network nodes to increase the communicability of some subset of nodes to a source node [31]. Figure 1 shows a toy example of the FAIP in which the addition of an edge from the facility s to a client a helps improve the accessibility i.e., reduce the distance to s for both clients a and f.

Given the FAIPs, we focus on the total accessibility cost (TAC) and maximum accessibility cost (MAC) objectives which are defined to be the sum and the maximum distances of the clients to the facility, respectively. When there is a strict budget constraint on the number of edges to be added, if there are a few outliers (for G having small diameter and moderate to high density), TAC is preferred over MAC because this will help prevent adding edges (connected from facility; see the problem characterization in Lemma 1 for details) to these outliers, which does not have much effect in reducing costs for other clients. Otherwise, if either there are no outliers (for G with similar said characteristics) or G has a large diameter and low density, then MAC is the better choice. Therefore, both objectives can be useful under different settings, and hence have been studied in existing works [29], [30] However, the methods of these works are not scalable (see our experiments in Section V-D) and do not provide quality guarantees. In response, we make the following contributions.

1) We provide a characterization of the structure of optimal solutions of FAIPs under the TAC and MAC objectives. This characterization is key to designing effective methods for these NP-hard problems.
2) Based on the characterization, we design scalable approximation algorithms and heuristics for TAC and MAC objectives.
   - For minimizing the MAC objective, we develop a farthest-first traversal algorithm, which iteratively selects a client farthest from the facility and then adds an edge connecting it to the facility. This algorithm provides $(2 - 1/OPT)$-approximation and runs in linear time where OPT is the optimal objective value.
   - For minimizing the TAC objective, we provide two constant factor approximation algorithms that are based on linear programming and local search [35], [36]. We achieve this by showing that if there is an α-approximation algorithm for the k-median problem with penalties, then there is an α-approximation algorithm for our problem.
   - To deal with large problem instances, we develop a simple scalable meta-heuristic to create edges that connect the facility to the nodes based on their importance, measured by some centrality criteria (e.g., degree, betweenness, and local clustering coefficient). The heuristic runs much faster than our approximation algorithms, as we show in our experiments. Its complexity is polynomial and depends on the complexity of computing node centralities.

Strategic Aspects of FAIPs. In some situations, the clients' locations may not be visible publicly to the algorithm designer. For instance, their locations can be their physical addresses or preferences of the facility's location on the infrastructure network or virtual addresses on the communication network. In order to improve the accessibility costs (i.e., TAC or MAC) of the clients, the designer needs to design an algorithm (or a mechanism) to elicit clients' locations and output k new edges. However, as noted in the facility location and mechanism design literature [37]–[45] in the last few decades, clients can strategically misreport locations to benefit themselves (i.e., manipulating algorithms' solutions to obtain closer distances to the facility). Therefore, we initiate the design of truthful (or strategyproof) algorithms for FAIPs that are robust to manipulation and incentivize the clients to report true locations. For the strategic aspects of FAIPs under the TAC and MAC objectives, we make the following contributions.

1) We show that it is not possible to design truthful algorithms with bounded approximation ratios in general for TAC and MAC objectives.
2) Because the developed algorithms and heuristics are not truthful, we propose globalization and randomization approaches to design provably truthful variants of these algorithms and heuristics for TAC and MAC objectives.

For all of the developed algorithms and heuristics, we conduct rigorous experiments on synthetic and real-world networks to measure the effectiveness of the proposed methods in reducing TAC and MAC objectives over several baselines and the scalability/efficiency (i.e., running time) of the proposed methods over various problem instances and network sizes.

B. Related Works

We now review prior studies addressing problems similar to ours and discuss their limitations.

Facility Accessibility Improvement Problems. The general FAIPs under the TAC and MAC objectives are first considered by [29], [30]. In their setting, a facility has been located on a network. There is a set of clients with demands that need to be served by the facility. The problem seeks to add a fixed number of edges to the networks to minimize the cost of serving the clients subject to the client's demands. We consider the most standard version of the FAIPs in which the demands of the clients are either 0 or 1. For both objectives, the FAIPs are NP-hard to solve. While their approaches apply to the standard version, their approaches (i.e., based on exhaustive heuristics [29], [30] and integer programming [30]) do not provide any approximation guarantee (or characterize optimal solutions), scale to small-size problem instances, and are ineffective (refer to our experiments).

Optimizing Node Eccentricity. The considered FAIP is also related to the problem of adding edges to the network to optimize node eccentricity subject to some objectives. Node eccentricity is defined to be the largest shortest distance of a given node to all other nodes in a network. For optimizing node eccentricity with respect to the largest shortest distance by adding edges to networks, [46] provide an LP-based $(1+\epsilon)$-approximation algorithm when adding $O(k \log |V|)$ edges. They show that any α-approximation algorithm for the diameter minimization problem can be seen as a 2α-approximation for the node eccentricity minimization problem. Finally, [31] develop a $(2+1/OPT)$-approximation algorithm, where OPT is the optimal solution value, using a similar approximation scheme as in [47] that adds shortcut edges to minimize a node eccentricity.

For optimizing node eccentricity with respect to the average shortest distance to all nodes instead of the largest shortest distance, [36] show that any α-approximation algorithm for the k-median with penalties problem gives an α-approximation to the optimization problem (with average shortest distance).

These works consider clients occupying the whole network and do not consider optimizing for TAC and MAC. Therefore, their approaches may not provide effective solutions for our FAIPs directly. For completeness, we implemented all these approaches in our experiments except the LP-based of [46] because it requires more edges to be added (i.e., $> k$).

Mechanism Design for Facility Location Problems. The study of the strategic aspects of FAIPs is within the algorithmic mechanism design [48]. More specifically, the considered setting is related to the mechanism design for facility location problems [37]–[41] in which one needs to determine the placement of a facility to minimize the distances of the clients to the facility. In our setting, the facility location has been determined, and we need to add edges to the network to minimize the distances to the clients. Thus, existing truthful algorithms (e.g., see [14], [37], [40]) no longer work directly for our problems.

C. Outline

In Section II, we present the necessary notations and the considered FAIPs. In Sections III and IV, we study the FAIPs and strategic aspects of FAIPs, respectively, for optimizing TAC and MAC. In Section V, we present our experimental results. We refer readers to the appendix for the missing plots in the experiments.

II. Preliminaries: Notations and Problems

In the following, we present necessary notations and the facility accessibility problems (FAIPs) on networks, including the strategic aspects of FAIPs.

A. Notations

A network $G = (V, E)$ consists of a node (or vertex) set V and an edge set E. An edge between two nodes, say $u, v \in V$, is denoted by $(u, v) \in E$. For simplicity, we consider undirected, unweighted, simple, and connected networks. Given network G, for any nodes $u, v \in V$, we define $d^G(u, v)$ to be the length of the shortest path of u and v in G.

We let $N = \{1, ..., n\}$ to be the set of n clients where each client $i \in N$ has location $x_i \in V$. For a node $v \in V$ and N, we define the maximum accessibility cost (MAC) objective of v to be $a^m_N(G, v) = \max_{i \in N} d^G(x_i, v)$ and the total accessibility cost (TAC) objective of v to be $a^t_N(G, v) = \sum_{i \in N} d^G(x_i, v)$, which measure the largest and total shortest distances of v over N in G, respectively.

B. Facility Accessibility Improvement Problems

Given MAC and TAC objectives, our main goal is to optimize these objectives for a given facility by adding a given number of edges in the network. Before presenting the FAIPs, we provide the following definitions.

Let $\overline{E} = V^2 \backslash E$ be the set of edges that are not in G. For any $E' \subseteq \overline{E}$, let $G + E' = (V, E \cup E')$ be the resultant network after adding E' to E. Let $s \in V$ be the pre-located facility's location. We consider the following FAIPs under the TAC and MAC objectives.

FAIP 1: Minimizing MAC. Given network $G = (V, E)$, facility's location $s \in V$, a set N of clients, and integer k, find $E' \subseteq \overline{E}$ of k edges so that $E' \in \arg \min_{A \subseteq \overline{E}: |A| = k} a^m_N(G + A, s)$.

FAIP 2: Minimizing TAC. Given $G = (V, E)$, $s \in V$, a set N of clients, and k, find $E' \subseteq \overline{E}$ of k edges such that $E' \in \arg \min_{A \subseteq \overline{E}: |A| = k} a^t_N(G + A, s)$.

For simplicity, we refer to the FAIPs as minimizing MAC or TAC for fixed input G, s, N, and k and do not explicitly state them when the context is clear.

In Section III, we will introduce approximation algorithms and heuristics to address FAIPs. An algorithm is α-*approximation* (or has *approximation ratio* α) for $\alpha \geq 1$, if for all instances, the objective value (i.e., MAC/TAC) induced by the algorithm is no greater than α times the optimal value.

C. Strategic Aspects of FAIPs

As discussed in the introduction, the clients' locations (e.g., physical/virtual addresses or clients' preferences) might not be publicly available. To optimize for MAC or TAC, we must rely on the clients to reveal their locations. However, as noted in the facility location and mechanism design literature [37]–[45], clients can strategically misreport locations to benefit themselves (i.e., obtain closer distances to the facility). Therefore, we consider designing truthful (or strategyproof) algorithms that are robust to manipulation and incentivize the clients to report true locations. Below, we formulate the strategic aspects of the FAIPs.

Recall that $N = \{1, ..., n\}$ is the set of clients. Each client $i \in N$ has a location $x_i \in V$ of a node of network G, and the collection $\mathbf{x} = (x_1, \ldots, x_n)$ is referred to as the location profile of clients. The location profile of clients is private. Each client strategically reports its location based on its cost. The *cost* of client $i \in N$ is the shortest distance to facility s in the resultant network after adding the k edges. That is, $cost(f(\mathbf{x}), x_i) = d^{G+f(\mathbf{x})}(x_i, s)$, where $f : V^n \to \overline{E}^k$ that maps a (reported) location profile $\mathbf{x}$ to k edges in the complementary edge set is a (deterministic) algorithm (or mechanism in mechanism design).

Notice that given f and its dependence on location profile $\mathbf{x}$, each client may have the incentive to misreport a different location to change the outcome of f. This situation is often not desirable and can create worse outcomes for all other clients. Thus, we seek to design an algorithm (or mechanism) f that is *truthful* or *strategyproof* in which no client can benefit from misreporting a false location, regardless of the strategies of other clients. That is, for all $\mathbf{x} \in V^n$, $i \in N$ and all $x'_i \in V$, we have $cost(f(\mathbf{x}), x_i) \leq cost(f(x', \mathbf{x}_{-i}), x_i)$, where $\mathbf{x}_{-i}$ is the profile of the locations of all clients in $N \backslash \{i\}$.

In Section IV, we show that the algorithms and heuristics in Section III are not truthful. We then design truthful algorithms and heuristics for optimizing MAC and TAC (i.e., FAIPs 1-2).

III. Scalable Algorithms for FAIPs

We study the FAIPs under the objectives of minimizing MAC in Section III-A and minimizing TAC in Section III-B. The following result says that there exists an optimal solution that only adds edges incident to the facility for both objectives. This result holds for the case when each node has a client (i.e., $N = V$) [36], [46]. We generalize it non-trivially for arbitrary N with any locations in V.

Lemma 1. *For every instance under either MAC or TAC objective, there is an optimal solution in which all of the k added edges are incident to the facility.*

Proof. Let $E^* \in \overline{E}^k$ be an optimal solution in which an edge $(u, v) \in E^*$ is not incident to the facility s. We show that only one direction is used for all shortest paths from the nodes in N to s that use this added edge. If the shortest path from $x \in V$ to s in $G + E^*$ passes through u first and then v, then it implies $d^{G+E^*}(u, s) > d^{G+E^*}(v, s)$, as otherwise this path would not pass through v. Similarly, if the shortest path from another node $y \in V$ to s in $G + E^*$ passes through v first and then u, it implies $d^{G+E^*}(u, s) < d^{G+E^*}(v, s)$, which yields a contradiction. Thus, only one direction in edge (u, v) is used.

If the direction used in (u, v) is from u to v, then we can replace (u, v) with (u, s) in the solution E^*, by which no node in N will increase the distance to s. Also, if the direction used is from v to u, then we can replace the edge (u, v) with (v, s). This establishes the lemma. □

A. Minimizing MAC

The FAIP of minimizing MAC is NP-hard even for the special case $N = V$ [46]. We present the following new *Farthest-first traversal* algorithm (Algorithm 1), which iteratively selects a client farthest from the facility and then adds an edge connecting it to the facility. This general idea comes from Gonzalez's algorithm [49] for clustering problems, where the first point is selected arbitrarily, and each successive point is as far as possible from the set of previously-selected points. The main difference is that our algorithm selects nodes in N only and completely ignores those nodes which are not in N.

Algorithm 1 Farthest-First Traversal (FFT)

Input: Network $G = (V, E)$, facility $s \in V$, integer k, set N of clients

Output: Network G' with k edges added

1: Set $D(v) \leftarrow d^G(v, s)$ for each $v \in V$ and $i \leftarrow 0$
2: **while** $i < k$ **do**
3: $\quad v_{i+1} \in \arg\max_{i \in N} D(x_i)$
4: $\quad$ Set $D(v) \leftarrow \min\{D(v),\, d^G(v, v_{i+1})\}$, $\forall v \in N$
5: $\quad i \leftarrow i + 1$
6: **end while**
7: $E' \leftarrow \{(s, v_1), (s, v_2), \ldots, (s, v_k)\}$ the set of k added edges
8: Return $G' \leftarrow G(V, E \cup E')$

Theorem 2. *Algorithm 1 is $(2 - \frac{1}{OPT})$-approximation for minimizing MAC, where OPT is the optimal MAC. This algorithm takes k steps and uses $O(kn)$ distance computations.*

Proof. Let $E^* = \{(v_1^*, s), \ldots, (v_k^*, s)\}$ be an optimal solution in which all of the k added edges are incident to the facility s. We can divide all nodes in V into $k + 1$ clusters $(V_s, V_1, V_2, \ldots, V_k)$, where $V_s \subseteq V$ is the set of nodes that do not use any edge in E^*, and V_i (for all $i \in [k]$) is the set of nodes that use edge $(v_i^*, s) \in E'$. Let $E' = \{(v_1, s), \ldots, (v_k, s)\}$ be the set of added edges, and $V(E') = \{v_1, \ldots, v_k\}$.

Consider an arbitrary cluster V_i. If there exists a client $v_{i'} \in V(E')$ such that $v_{i'} \in V_i$, then every client $x \in V_i$ has a distance at most $2(OPT - 1)$ from $v_{i'}$, because $d^G(x, v_{i'}) \leq d^G(x, v_i^*) + d^G(v_i^*, v_{i'}) \leq 2(OPT - 1)$ (note that $d^G(x, v_i^*) + 1 = d^{G+E^*}(x, s) \leq OPT$). Hence, the cost of client $x \in V_i$ is $d^{G+E'}(x, s) = d^G(x, v_{i'}) + 1 \leq 2OPT - 1$.

It remains to consider the case when $V(E') \cap V_i = \varnothing$, i.e., no edges in E' are incident to V_i. We discuss two subcases.

- Case 1. There exists some client $v_{s'} \in V(E') \cap V_s$. By the procedure of Algorithm 1, it implies that, for any client $x \in V_i$, we have $d^G(x, V(E')) \leq d^G(v_{s'}, s) \leq OPT$, and thus the cost of x is at most $d^G(x, V(E')) + 1 \leq OPT + 1$.
- Case 2. There exists a cluster V_j such that $|V_j \cap V(E')| \geq 2$. Assume w.l.o.g. that $v_1, v_2 \in V_j \cap V(E')$. By the procedure of Algorithm 1, it implies that, for any client $x \in V_i$, we have $d^G(x, V(E')) \leq d^G(v_1, v_2) \leq d^G(v_1, v_{j^*}) + d^G(v_2, v_{j^*}) \leq 2(OPT - 1)$, and thus the cost of x is at most $d^G(x, V(E')) + 1 \leq 2OPT - 1$.

Therefore, the approximation ratio of the Algorithm is $\frac{2OPT-1}{OPT} = 2 - \frac{1}{OPT}$. □

The Farthest-first traversal algorithm is proven to have an approximation ratio of $2 + \frac{1}{OPT}$ in [31] for minimizing the maximum distance to the facility from all nodes (which corresponds to a special case in our problem when $N = V$). Our result also improves the ratio of farthest-first traversal to $2 - \frac{1}{OPT}$ for this special case, and the analysis is tight. For comparison, we remark that for a related problem, Bounded-Cardinality Minimum-Diameter Edge Addition, which asks to find a set E' of k edges to be added to graph G in such a way that the *diameter* of $G + E'$ is minimized, farthest-first traversal has $(2 + \frac{2}{OPT})$-approximation [47].

B. Minimizing TAC

The FAIP of minimizing TAC is NP-hard even for the special case $N = V$ [36]. We derive two constant approximation algorithms by non-trivially reducing our problem to the k-median problem with penalties [35], [36].

In the k-median problem with penalties, we are given a set of facilities F and a set of clients C. Each client wants to get some service at one of the facilities. There is a cost/distance $c_{ij} \geq 0$ that specifies the cost/distance of every pair of facilities and clients $i, j \in C \cup F$. Each client $j \in C$ has a penalty $p_j \geq 0$ for not getting the service. For each client, we can either provide the service and pay the cost to its cheapest/nearest facility or pay the penalty for not serving the client. The goal is to select k facilities to open to minimize the total payment by serving and not serving the clients.

We note that, for the case of $N = V$, [36] reduce our problem to the k-median problem with penalties. Below, we reduce our problem for arbitrary N with any locations to the k-median problem with penalties.

Lemma 3. *If there is an α-approximation algorithm for the k-median problem with penalties, then there is an α-approximation algorithm for minimizing TAC.*

Proof. Consider an instance of our problem where we are giving a network $G = (V, E)$, a client set N, the fixed facility location $s \in V$, and an integer k.

We construct an instance of k-median problem with penalties. Let $F = V$ and $C = N$. For each $j \in C$, let $p_j = d^G(j, s)$ be the distance to the fixed facility in our problem. For each $i \in F, j \in C$, let $c_{ij} = d^G(i, j) + \mathbf{1}[i, j \neq s]$ be the shortest distance between i and j in G, and plus one when neither i nor j is s. The goal is to open k facilities to minimize the total payment for serving/not serving the clients.

Suppose we have an optimal solution of facilities F^* of size k. Let C^{S*} and C^{NS*} be the sets of clients that are served by some facilities in F^* and that are not served by any of the facilities in F^*, respectively. Assume w.l.o.g. that each facility in F^* strictly improves some clients' costs; otherwise we could just remove the additional facilities and we will retain the same cost. It follows that

$$\begin{aligned} & \text{total payment} \\ = & \sum_{j \in C^{S*}} \min_{i \in F^*} c_{ij} + \sum_{j \in C^{NS*}} p_j \\ = & \sum_{j \in C^{S*}} \min_{i \in F^*} \left(d^G(j, i) + \mathbf{1}[i, j \neq s] \right) + \sum_{j \in C^{NS*}} d^G(j, s) \\ = & \sum_{j \in N} d^{G+E'}(j, s), \end{aligned}$$

where E' is the set of edges from s to each node in F^*. Indeed, E' is an optimal solution because we only need to find k endpoints to connect to the fixed facility in our problem (Lemma 1) and this corresponds to the k facilities in the optimal solution. Clearly, if we have an α-approximation algorithm for the k-median problem with penalties, we can use it to get an α-approximation solution for our problem. □

There are two lines of approximation algorithms for k-median problems with penalties: one is LP-based approaches, and the other is local search approaches. Charikar et al. [35] provide an LP-based 4-approximation algorithm in polynomial time, using Lagrangian relaxation and primal-dual techniques.

Meyerson and Tagiku [36] provide a local search $(3 + \frac{2}{q})$-approximation algorithm, where q is the swap size. The general idea of the algorithm is to improve the quality of the solution by iteratively modifying the solution in the previous iterations. More formally, the algorithm starts with an arbitrary solution set of k edges and, at each iteration, checks to see if it can obtain a lower cost (by a ratio of sufficiently large $\delta \in [0, 1]$) by swapping/replacing q edges in the solution set to those q edges that are not in the solution set.

Lemma 3 implies that these approximations naturally apply to our problem for minimizing TAC.

Corollary 4. *For minimizing TAC, there is an LP-based 4-approximation algorithm and a local search $(3 + \frac{2}{q})$-approximation algorithm with swap size q.*

C. Heuristic Algorithms

Experimentally, we show that both of the aforementioned algorithms for minimizing TAC are not scalable to large instances. Therefore, we consider simple and fast meta-heuristic algorithms (which can also be applied to MAC). They may not have any theoretical guarantees on the performance but are intuitive and easy to understand.

Algorithm 2 Highest k Importance (k-Im)

Input: Network $G = (V, E)$, facility $s \in V$, integer k, set N of clients

Output: Network G' with k edges added

1: Let $N(s)$ be the neighbors of s
2: Let $C = N \backslash (\{s\} \cup N(s))$ be the set of clients' locations except s and $N(s)$.
3: For each $v \in C$, compute `importance(`v`)`
4: Let $v_1, v_2, \ldots, v_k$ be the top highest k `importance` nodes
5: $E' \leftarrow \{(s, v_1), (s, v_2), \ldots, (s, v_k)\}$ the set of k added edges
6: Return $G' \leftarrow G(V, E \cup E')$

By Lemma 1, there is an optimal solution that selects some nodes and connects them to the facility. We thus consider meta-heuristic algorithms which select the nodes based on their *importance*, measured by some centrality criteria, e.g., degree, betweenness, local clustering coefficient, etc. For example, if the centrality criterion is the degree, then the nodes with a larger degree are more important than others, and thus have a high priority to be selected for adding edges. See Section V-A for more details on our considered centrality measures.

In our meta-heuristic, shown in Algorithm 2, we score each client based on a certain `importance` function and then select the top k clients with the highest scores. The k edges are then formed between the top k clients and the facility.

The running time of Algorithm 2 depends on the time for computing the different centrality measures, e.g., the degrees of the clients can be computed in $O(n^2)$. Note that in Algorithm 2, only clients are considered to be selected for adding edges. Alternatively, we can consider all the nodes in the network, and more details can be found in the next section.

IV. Scalable Algorithms for Strategic FAIPs

In the above considered optimization problems, the locations of the clients in the problems are given as inputs. However, in a strategic environment, these locations are not readily visible publicly. Therefore, to optimize MAC or TAC, the clients are required to report their own locations. As noted in previous facility location and mechanism design literature (see e.g., [37]–[45]), each client is strategic and has an incentive to misreport their location if it could bring benefits to themselve. Thus, in this section, we study the strategic aspects of FAIPs. We refer readers to Section II for definitions and strategic FAIPs.

Unfortunately, all the algorithms mentioned before are not truthful. For Algorithm 1, consider an instance in which client $i \in N$ is the farthest client from the facility, and j is the second farthest client. Suppose $k = 1$. There is a path (x_i, u, v, x_j), where $u \in V$ is the farthest node from the facility among all nodes. If client j reports their location truthfully, then the algorithm will add an edge (x_i, s), and j incurs a cost of 4. However, if j misreports the location as u, then the algorithm will add an edge (u, s), and the client would incur a smaller cost of 3. For Algorithm 2, if a client has small importance, but one of their non-client neighbor nodes has very large importance, then clearly this client has the incentive to misreport their location as this neighbor node and thus gains. For k-median-based algorithms, similar arguments work for showing the non-truthfulness.

On the other hand, we can show that it is not possible to design a truthful algorithm with bounded approximation ratios for both objectives. Fotakis and Tzamos [39] prove that for any $k \geq 3$, there is no deterministic anonymous algorithm with bounded approximation ratios for the k-facility location problems. We can modify their constructions and arguments to our strategic FAIPs of minimizing MAC and TAC. Therefore, we resort to designing algorithms that may have no bounded approximation ratios.

Nevertheless, there are two general and useful approaches for us to design truthful algorithms [40]: (1) ignoring the reports from the clients (called global approaches) and (2) randomization.[2] We will use these approaches to design provably truthful variants of our algorithms and heuristics in Section III.

A. Global Approaches

The global approaches simply ignore all the reports from the clients, or equivalently, assume that every node of the network is occupied by a client. The global version of Algorithm 1 iteratively selects a node (not necessarily to be a client location) farthest from the facility and then adds an edge connecting it to the facility. The global version of Algorithm 2 defines the importance of all nodes and then chooses the top k nodes for adding edges. These global versions are naturally truthful because the output of the algorithms relies only on the network structure, and the reports from clients cannot change that.

Proposition 5. *The global versions of Algorithm 1 and 2 are truthful.*

We can also globalize the algorithms for the k-median problem with penalties, by assuming that the clients are everywhere, i.e., $C = F$. Such global versions are also truthful algorithms for the TAC objective. However, these algorithms (the local search and linear programming) are not efficient.

B. Randomization

Randomization is a powerful tool to achieve truthfulness in mechanism design, which can be used to circumvents the obstacles in deterministic algorithms via a probabilistic way. Here, we attempt to randomize Algorithms 1 and 2. Algorithm 3 presents a general randomization framework, which samples a subset of clients of size k from N, each selected with

[2] A randomized algorithm (or mechanism) is a function f that maps a location profile of clients $\mathbf{x} \in V^n$ to a probability distributions over $\overline{E}^k$. The cost of client $i \in N$ is the expected distance to the facility, i.e., $cost(f(\mathbf{x}), x_i) = \mathbb{E}_{E' \sim f(\mathbf{x})} d^{G+E'}(x_i, s)$.

Algorithm 3 Randomized Framework (RF)

Input: Network $G = (V, E)$, facility $s \in V$, integer k, client location profile $\mathbf{x}$, probabilities p_i for $i \in N$

Output: Network G' with k edges added

1: Sample k clients $v_1, \ldots, v_k$ from N such that each client $i \in N$ is selected with probability p_i
2: $E' \leftarrow \{(s, v_1), (s, v_2), \ldots, (s, v_k)\}$ the set of k added edges
3: Return $G' \leftarrow G(V, E \cup E')$

probability p_i. Afterward, the k edges are constructed from the facility to these k clients.

Not all definitions of p_i are truthful: When p_i is proportional to the node importance as in Algorithm 2, Algorithm 3 is not truthful. For example, consider a network $G = (V, E)$, where $v \in V$ is a node with a very large degree of L^2 but without any client, and $u \in V$ is a neighbor of v with a degree of 1 and client i lies on u. Suppose that $w \in V$ is a node with degree L and there is some client lying on w. Under the *degree* centrality measure, when telling the truth, client i has a small probability of being selected and thus has a large cost. When misreporting their location to v, since v has a large probability of being chosen, client i incurs a cost of only 1. Thus, misreporting the location to a node nearby with a large degree will improve i.e., lower the cost of a client.

However, we are able to identify p_i that is truthful when the probability is uniform, i.e., $p_i = \frac{1}{n}$. We prove that this is also true when the algorithm samples without replacement.

Lemma 6. *Algorithm 3 with uniform probabilities is truthful and runs in time $O(kn)$, whenever the sampling is with or without replacement.*

Proof. We sample k clients with uniform probability and with replacement. If a client $i \in N$ misreports their location from x_i to x'_i, the probability of selecting other clients is the same. Thus, misreporting will only increase their expected cost. □

Further, if p_i is proportional to i's relative distance from the facility and the sampling is with replacement (though this might not perform well because the same client can be selected again), Algorithm 3 is truthful.

Lemma 7. *Algorithms 3 with p_i defined as the (normalized) distance of client i to the facility, $p_i = \frac{d^G(x_i,s)}{\sum_{j \in N} d^G(x_j,s)}$, is truthful when sampling with replacement, in time $O(kn)$.*

Proof. The algorithm chooses k clients with probabilities proportional to the distances from the facility, with replacement. Suppose a client $i \in N$ misreports their location from x_i to x'_i. For simplicity, denote $d^G(x_i, s)$ as d, and $d^G(x'_i, s)$ as d'. If $d'_i \leq d_i$, then the probability of selecting this client will decrease, while the probability of selecting other clients will increase. Obviously, the expected cost cannot decrease.

If $d'_i > d_i$, then the probability of selecting client i will increase from $p_i = 1 - \left(1 - \frac{d_i}{d_i + \sum_{j \neq i} d_j}\right)^k$ to $p'_i = 1 - \left(1 - \frac{d'_i}{d'_i + \sum_{j \neq i} d_j}\right)^k$, while the probability of selecting other clients will decrease. The cost when telling the truth is

$$cost_i = (1 - p_i)\mathbb{E}[cost|\ i \text{ is not selected}],$$

and the cost when misreporting is

$$cost'_i = p'_i(d'_i - d_i) + (1 - p'_i)\mathbb{E}[cost|\ i \text{ is not selected}],$$

where $\mathbb{E}[cost|\ i \text{ is not selected}]$ is the expected cost of client i given that i is not selected. After computations, we can verify that $cost_i \leq cost'_i$. Therefore, client i cannot decrease the cost by misreporting, and thus the algorithm is truthful. □

We conclude that not every randomization scheme makes Algorithm 3 truthful. Randomizing based on the clients' distances relative to the facility or randomizing uniformly is truthful. Randomizing based on node importance is not.

V. Experiments

In this section, we evaluate our proposed algorithms with respect to the (1) effectiveness in minimizing TAC and MAC on both real-world and synthetic networks and (2) scalability (i.e., running time) on real-world networks.

Recall that in our problems for minimizing MAC and TAC, the inputs consist of network $G = (V, E)$, facility $s \in V$, set of clients N, and integer k. Below, we describe their configurations for our experiments and discuss our proposed methods/benchmarks. All source code and data are publicly available at our GitHub repository. All reported results ran on a Linux 64-bit machine, NVIDIA Tesla K40 GPU with 10GB memory.

Data. We use synthetic and real-world networks:

- For synthetic networks, we use the Erdős–Rényi random network model [50] with $|V| = 200$ nodes and edge-connecting probability $p \in \{0.1, 0.2, 0.3\}$ for small random networks, and $|V| = 3000$ nodes and $p \in \{0.008, 0.05, 0.1\}$ for large random networks, five networks for each set indexed by $i = 1, \ldots, 5$. The p's are selected to capture sparse to dense networks (i.e., to restrict the degree and radius of the facility).
- For real-world networks, we use 3 social (*Karate*, *Dolphins*, and *Jazz*) [51], 9 AS *Oregon A-I*, and 5 *P2P-Gnutella A-E* networks [52], ranging from 34 up to 13,947 nodes and 78 up to 39,994 edges. See Table I for more details on network characteristics.

Setup. Given a network, we generate s and N uniformly random while varying the size of $N = \{1, ..., n\}$ that is proportional to the number of nodes in the network. Note that for simplicity we restrict each client to occupy exactly one unique node and hence $n \leq |V|$. We consider adding different numbers of k edges. More specifically, unless specified explicitly, for each network, we generate problem instances with 10 different facilities. For each pair of the network and a generated facility, we generate 10 random sets of node locations of various (rounded up) sizes $n \in \{0.01, 0.1, 0.25, 0.5, 0.75, 0.90\} \times |V|$.

TABLE I: Characteristics of the real-world networks in experiments. Note that D1, D2, and D4 are originally not connected, so we extract the largest connected component for each (original network specifications shown in parentheses).

Dataset	$nodes$	$edges$	$density$
Karate (K)	34	78	0.1390
Dolphins (D)	62	159	0.0841
Jazz (J)	198	2,742	0.1406
Oregon-A (A0)	633	1,086	0.0054
Oregon-B (A1)	1,503	2,810	0.0024
Oregon-C (A2)	2,504	4,723	0.0015
Oregon-D (A3)	2,854	4,932	0.0012
Oregon-E (A4)	3,995	7,710	0.0009
Oregon-F (A5)	5,296	10,097	0.0007
Oregon-G (A6)	7,352	15,665	0.0005
Oregon-H (A7)	10,860	23,409	0.0004
Oregon-I (A8)	13,947	30,584	0.0003
P2P-GnutellaA (D1)	6,299 (6,301)	20,776 (20,777)	0.0010
P2P-GnutellaB (D2)	8,104 (8,114)	26,008 (26,013)	0.0008
P2P-GnutellaC (D3)	8,717	31,525	0.0008
P2P-GnutellaD (D4)	8,842 (8,846)	31,837 (31,839)	0.0008
P2P-GnutellaE (D5)	10,876	39,994	0.0007

We report the relative percentage decrease of TAC and MAC as the number of added edges k increases from 1 to n, i.e., $100|a_N(G,s) - a_N(G+E',s)|/a_N(G,s)$, where E' denotes the solution returned from the algorithms using k added edges. For scalability studies, we run the algorithms on real networks of different sizes (with respect to $|V|$) and report the wall-clock time in seconds.

A. Proposed Methods

We refer to the Farthest-First Traversal Algorithm (Algorithm 1) and its global version as FFT and FFT$_V$, respectively; Highest k Importance (Algorithm 2) and its global version as k-Im and k-Im$_V$; randomized algorithms (Algorithm 3) with p_i proportional to i's relative distance from the facility and p_i uniform probabilities as RF-Prop and RF-Uniform (we consider RF-Uniform sampling with and without replacement); Meyerson and Tagiku's local search algorithm [36] as LS; and Charikar's LP-based algorithm [35] as LP. We only consider LS (with parameters $q = 1$ and $\delta = 0.95$) and LP on small social networks and the synthetic networks with $|V| = 200$, as they are not scalable for large networks (e.g., they take several hours to run on networks of sizes as small as 200 nodes in our scalability experiment in Section V-D).

For k-Im and k-Im$_V$, we consider the following centrality measures: 'random' (R): randomly sort nodes (averaged over 10 runs); 'high-degree' (HD) and 'low-degree' (LD): highest and lowest degree nodes first, respectively; 'betweenness' (B) [53]; 'pagerank' (P) [54]; 'eigenvector' (E) [55]; 'local clustering coefficient' (C) [56]; 'closeness' (L) [57]; and 'eccentricity' (EC) [58]. Additionally, if multiple nodes have the same importance measure, we sort them lexicographically based on their inherent numerical labels. In practice, it is more computationally efficient to calculate the node importance for the whole network i.e., running k-Im$_V$, then simply extract the nodes occupied by the clients from the result to form the solution for k-Im. We compute this in one shot using MATLAB R2022a's built-in `centrality` function, which measures all types of centrality except local clustering coefficient, which we use an external package. We report the best centrality value. For the random-based methods, we record the performance by averaging the outputs from the resultant G' over 10 runs.

Benchmarks. We consider the first heuristic from [29] and Algorithms 5 and 6 from [30] (referred to as Berman) for FAIPs under the two cost objectives, which aim to optimize more general problems than ours (see our related work). We run them on small networks ($|V| \leq 200$) similar to LS and LP due to its poor scalability. We also consider adding k edges that connect two nodes with the highest scores according to the following criteria without using our characterization as stated in Lemma 1: (1) 'random': connect k random pairs of unconnected nodes (averaged over 10 runs); k pairs between nodes with (2) 'high-high': highest degrees; (3) 'low-low': lowest degrees; (4) 'high-low': highest difference in degrees; (5) 'netgel': k pairs between nodes $u, v, (u,v) \notin E$ with highest product $\mathbf{e_{u1}}\mathbf{e_{v1}}$ where $\mathbf{e_{u1}}$ is the u^{th} element of the principal eigenvector $\mathbf{e_1}$ of the current network's adjacency matrix [59]. We denote these benchmarks as Edge-Score. We run all benchmarks iteratively to consider the added edges.

B. Experiments on Synthetic Networks

In this subsection, we evaluate the effectiveness of the proposed methods on Erdős–Rényi (ER) network model [50] with $|V| = 200$ nodes and $p \in \{0.1, 0.2, 0.3\}$ for small ER networks, and $|V| = 3000$ nodes and $p \in \{0.008, 0.05, 0.1\}$ for large ER networks.

Effectiveness Observations. Figure 2 shows the effectiveness of the proposed methods and benchmarks with respect to the relative percentage decrease of MAC and TAC (y-axis) over different numbers of added edges (i.e., k) when $p = 0.1$ at instance $i = 3$. We observe that

1) All of our proposed algorithms and heuristics significantly outperform the baselines [29], [30] from Berman and benchmarks across all k's and n's;
2) For minimizing TAC, for small numbers of k, LP performs best particularly when n is small; otherwise FFT is the overall best. When n is large, the four methods LP, FFT, k-Im, and RF-Uniform without replacement perform equally well and are the tied best;
3) For minimizing MAC, LP and FFT generally perform best or second-best across all k's and n's. When n increases, the global methods FFT$_V$ and k-Im$_V$ start improving substantially;
4) All of the (truthful) algorithms (i.e., FFT$_V$, k-Im$_V$, RF-Prop, and RF-Uniform) that focus on optimizing the whole networks perform poorly when k is large. However, when the number of clients n increases, the (truthful) algorithm performance starts to increase.

Although LP performs best in general, it does not scale well for large n's. FFT, the second best, is thus a good choice when both effectiveness and scalability are concerned (which we demonstrate below in the scalability experiments). Interestingly, even though both methods were specifically designed for

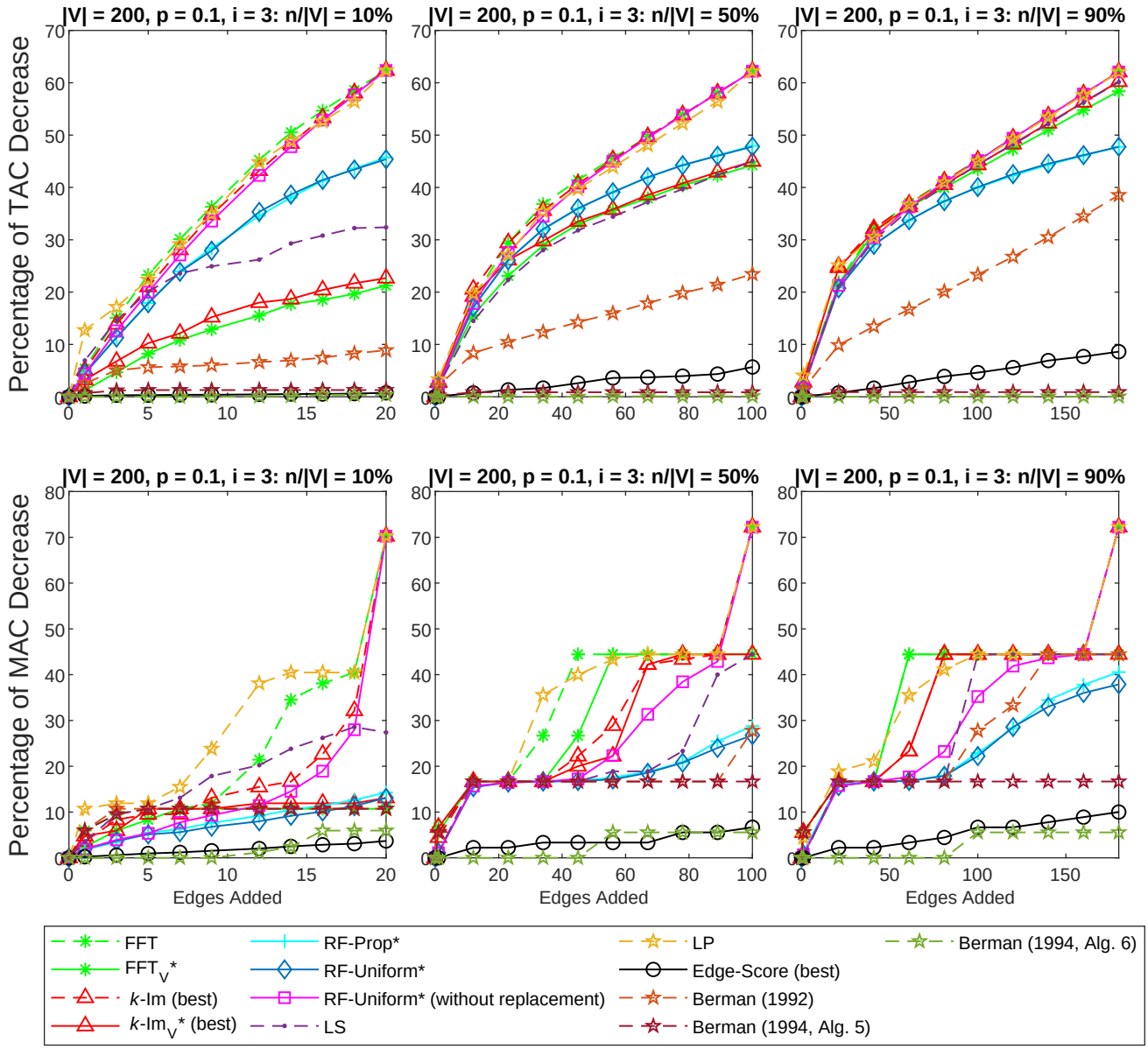

Fig. 2: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the third instance ($i = 3$) of the synthetic network with $|V| = 200$ nodes and $p = 0.1$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$. The (small) error bars are removed for readability. Truthful algorithms are denoted using an asterisk. Results from k-Im, k-Im$_V$, and Edge-Score are condensed (only best showed). All other considered settings (see Appendix A) exhibit similar behavior.

different objectives, with LP being introduced for minimizing TAC and FFT for minimizing MAC, they perform well for both. Regarding Observation 4, as n increases, it is natural to see the increased performance of these algorithms (as we are getting closer to optimizing for the whole network). Notice that the degraded performance of the randomized algorithms with replacement is due to the selection of duplicated nodes.

Truthfulness. Recall that deterministic methods over N are not truthful and deterministic methods over V, in general, do not perform as well. Our proposed randomized algorithm (i.e., RF-Uniform without replacement) seems to provide a good trade-off between performance and truthfulness across all k.

All other omitted plots for the considered settings exhibit similar behavior and observations (see Appendix A).

C. Experiments on Real-world Networks

In this subsection, we evaluate the effectiveness on real-world networks and remark on settings where truthfulness is

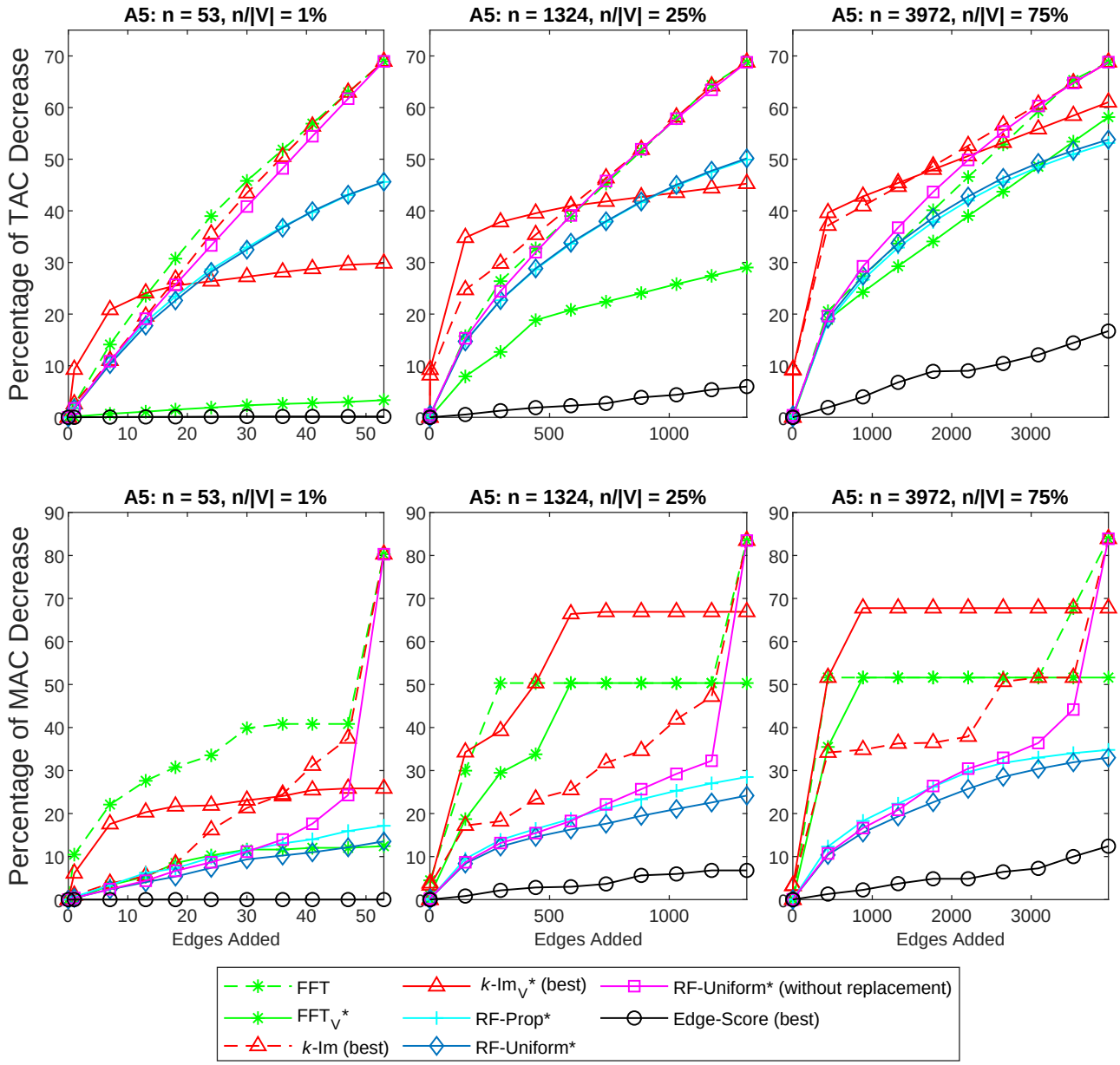

Fig. 3: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *Oregon-F* with different client sizes, (left) $n = 0.01|V|$, (center) $n = 0.25|V|$, and (right) $n = 0.75|V|$. The (small) error bars are removed for readability. Truthful algorithms are denoted using an asterisk. Results from k-Im, k-Im$_V$, and Edge-Score are condensed (only best showed). Other settings on the remaining *Oregon* networks (see Appendix B) exhibit similar behavior.

concerned.

Effectiveness Observations. Figure 3 shows the effectiveness of the proposed methods and benchmarks with respect to relative percentage decrease of MAC and TAC (y-axis) over different numbers of added edges (i.e., k) for *Oregon-F* network. We observe that (including *Oregon-A-I* networks)

1) Again, all of our proposed methods consistently yield much better effectiveness than the considered benchmarks;
2) For minimizing TAC, k-Im$_V$ performs best up to small or moderate k, then the proposed methods on N (FFT, k-Im, and RF-Uniform without replacement) perform best for the remaining k; as n gets larger, k-Im$_V$'s performance at large k approaches the said methods on N;
3) For minimizing MAC, when n is not too small, k-Im$_V$ performs best (or second-best) for up to (very) large k,

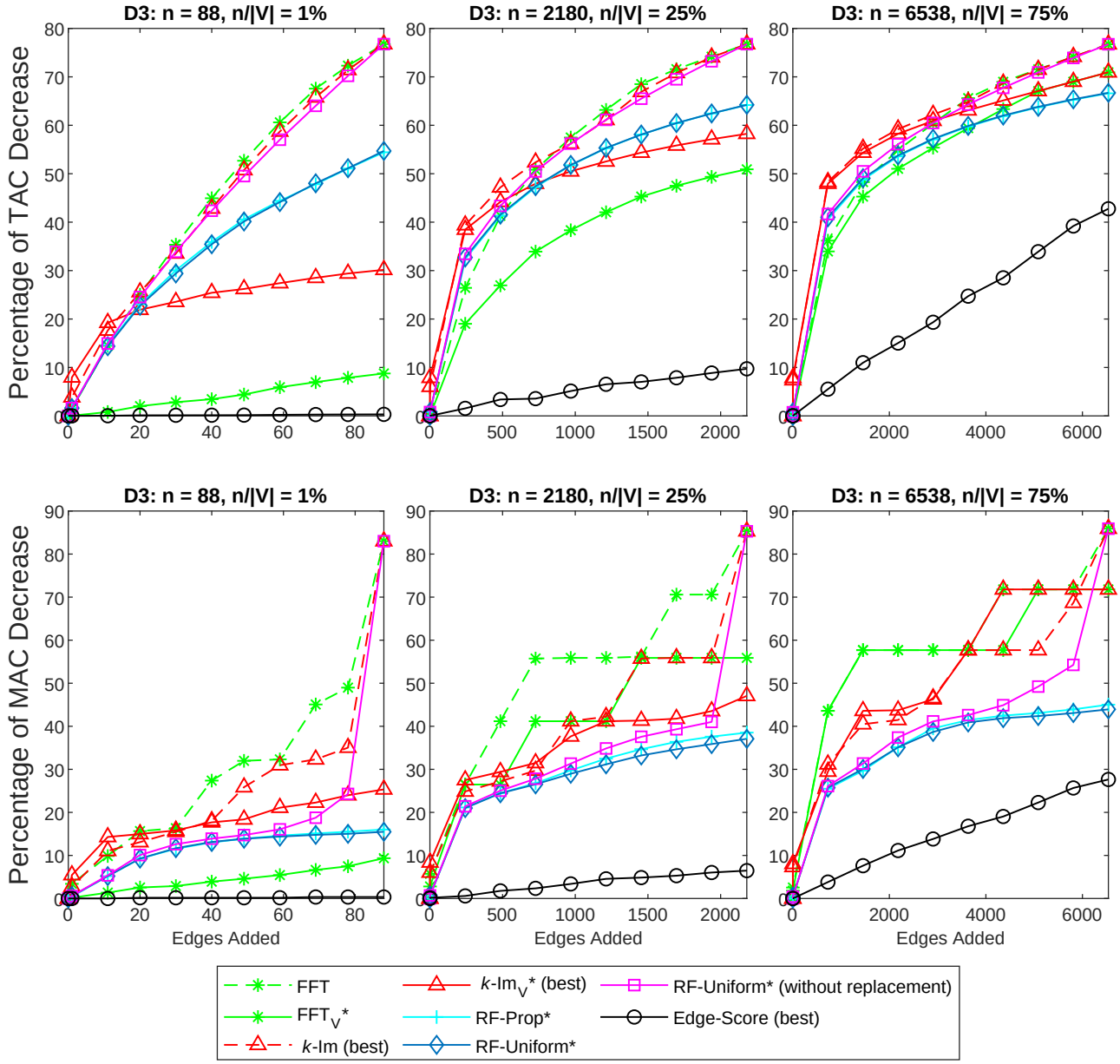

Fig. 4: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *P2P-GnutellaC* with different client sizes, (left) $n = 0.01|V|$, (center) $n = 0.25|V|$, and (right) $n = 0.75|V|$. The (small) error bars are removed for readability. Truthful algorithms are denoted using an asterisk. Results from k-Im, k-Im$_V$, and Edge-Score are condensed (only best showed). Other settings on the remaining *P2P* networks (see Appendix B) exhibit similar behavior.

then the methods on N outperform it, with FFT being the overall best followed closely by k-Im and RF-Uniform without replacement, respectively.

While other smaller real-world networks obtain similar observations as the ER networks in Section V-B, the *P2P* networks (as in Figure 4 for *P2P-GnutellaC*) exhibit different observations (similar for other *P2P* networks):

1) Our methods still outperform the benchmarks by a large margin for any combination of n and k, however
2) For minimizing TAC, when k is (very) small, k-Im$_V$ is the most effective; then for larger k's either FFT or k-Im is effective; as n increases, FFT and k-Im together with RF-Uniform without replacement converge and are the tied best, while the performance of k-Im$_V$ approaches these three methods on N;
3) For minimizing MAC, FFT performs best at every k

across all n's not too small (for very small n k-Im$_V$ is the best for small k); the methods on V (e.g., FFT$_V$ or k-Im$_V$) perform better as n increases (sometimes even better than the methods on N) and outperform the randomized one without replacement for most k's.

Truthfulness Observations. Given that the proposed global methods perform reasonably well on the real-world networks for both MAC and TAC for a wide range of k and the randomized algorithm without replacement performs well for other k, it would be reasonable to combine both of these methods into a unified truthful algorithm that provides a good trade-off between performance and truthfulness across all k.

D. Scalability of Proposed Methods

Figure 5 shows the running time (in seconds) of the proposed methods with respect to the number of nodes. For these plots, we sort the real-world networks from smallest to largest based on their numbers of nodes and measure the running time of the algorithms on the networks. In particular, we use the networks *Oregon A-I*, sorted by $|V|$ to evaluate the scalability of the proposed methods. For completeness (despite to their poor effectiveness as demonstrated earlier), we also run and include Berman's on small datasets of *Karate*, *Dolphins*, and *Jazz*. In the plots, each row associates with a size of client set (n) and each column corresponds to a number of added edges (k). Note that both axes use logarithmic scales with base 10.

We observe that the running times of LS and LP grow exponentially and hence are infeasible for large networks. In particular, LP took nearly 3 hours for a set of only 600 clients (*Oregon-A*), and while LS scaled slightly better, it still required 6-7 hours (which exceeds the y-axis' limit in the plots) to work with 5000 clients (*Oregon-F*). For other proposed algorithms, they all scaled linearly or even sublinearly with respect to the number of nodes and we did see an increase in running time for the proposed methods (especially FFT and FFT$_V$) when either the number of added edges or the number of clients increased.

VI. Conclusions

We consider the standard FAIPs and their strategic aspects under the TAC and MAC objectives. We first provide a characterization of the structure of the optimal solutions for the two objectives. Based on the characterization, we design scalable approximation algorithms and heuristics to address the FAIPs efficiently. Our experiments demonstrate the effectiveness and efficiency of the proposed methods.

Our studies on FAIPs can lead to several possible future directions. An immediate direction is to extend the considered version of FAIPs to account for costs and an overall budget constraint for adding edges. It would also be interesting to consider weighted or directed networks in FAIPs and the corresponding optimization problems. Finally, investigating the applicability and generalizability of the proposed methods to other general settings of FAIPs is also an important direction.

References

[1] R. Z. Farahani, M. Hekmatfar, B. Fahimnia, and N. Kazemzadeh, "Hierarchical facility location problem: Models, classifications, techniques, and applications," *Computers & Industrial Engineering*, vol. 68, pp. 104–117, 2014.

[2] R. Z. Farahani and M. Hekmatfar, *Facility location: concepts, models, algorithms and case studies*. Springer Science & Business Media, 2009.

[3] R. Z. Farahani, M. SteadieSeifi, and N. Asgari, "Multiple criteria facility location problems: A survey," *Applied Mathematical Modelling*, vol. 34, no. 7, pp. 1689–1709, 2010.

[4] Z. Drezner and H. W. Hamacher, *Facility location: applications and theory*. Springer Science & Business Media, 2004.

[5] D. B. Shmoys, E. Tardos, and K. Aardal, "Approximation algorithms for facility location problems (extended abstract)," in *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, 1997, pp. 265–274.

[6] A. Boloori Arabani and R. Z. Farahani, "Facility location dynamics: An overview of classifications and applications," *Computers & Industrial Engineering*, vol. 62, no. 1, pp. 408–420, 2012.

[7] M. S. Daskin, W. J. Hopp, and B. Medina, "Forecast horizons and dynamic facility location planning," *Annals of Operations Research*, vol. 40, no. 1, pp. 125–151, 1992.

[8] C. H. Aikens, "Facility location models for distribution planning," *European Journal of Operational Research*, vol. 22, no. 3, pp. 263–279, 1985.

[9] T. A. Arentze, A. W. J. Borgers, and H. J. P. Timmermans, "The integration of expert knowledge in decision support systems for facility location planning," *Computers, Environment and Urban Systems*, vol. 19, no. 4, pp. 227–247, 1995.

[10] G. Duranton and W. C. Strange, *Handbook of regional and urban economics: applied urban economics*. Elsevier, 1986, vol. 3.

[11] G. DeVerteuil, "Reconsidering the legacy of urban public facility location theory in human geography," *Progress in Human Geography*, vol. 24, no. 1, pp. 47–69, 2000.

[12] J. Allen, M. Browne, and T. Cherrett, "Investigating relationships between road freight transport, facility location, logistics management and urban form," *Journal of Transport Geography*, vol. 24, pp. 45–57, 2012.

[13] D. Black, "On the rationale of group decision-making," *Journal of Political Economy*, vol. 56, no. 1, pp. 23–34, 1948.

[14] H. Moulin, "On strategy-proofness and single peakedness," *Public Choice*, vol. 35, no. 4, pp. 437–455, 1980.

[15] C. Domshlak, R. I. Brafman, and S. E. Shimony, "Preference-based configuration of web page content," in *Proceedings of the 17th International Joint Conference on Artificial Intelligence - Volume 2*, 2001, pp. 1451–1456.

[16] K. L. Wagstaff, M. desJardins, and E. Eaton, "Modelling and learning user preferences over sets," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 22, no. 3, pp. 237–268, 09 2010.

[17] A. Ahmadi-Javid, P. Seyedi, and S. S. Syam, "A survey of healthcare facility location," *Computers & Operations Research*, vol. 79, pp. 223–263, 2017.

[18] D. Celik Turkoglu and M. Erol Genevois, "A comparative survey of service facility location problems," *Annals of Operations Research*, vol. 292, no. 1, pp. 399–468, 2020.

[19] Y. Liu, Y. Yuan, J. Shen, and W. Gao, "Emergency response facility location in transportation networks: A literature review," *Journal of Traffic and Transportation Engineering (English Edition)*, vol. 8, no. 2, pp. 153–169, 2021.

[20] E. Erkut, A. Karagiannidis, G. Perkoulidis, and S. A. Tjandra, "A multicriteria facility location model for municipal solid waste management in north greece," *European Journal of Operational Research*, vol. 187, no. 3, pp. 1402–1421, 2008.

[21] V. Yadav, A. K. Bhurjee, S. Karmakar, and A. K. Dikshit, "A facility location model for municipal solid waste management system under uncertain environment," *Science of The Total Environment*, vol. 603-604, pp. 760–771, 2017.

[22] S. K. Das, S. K. Roy, and G. W. Weber, "Heuristic approaches for solid transportation-p-facility location problem," *Central European Journal of Operations Research*, vol. 28, no. 3, pp. 939–961, 2020.

[23] O. J. Adeleke and D. O. Olukanni, "Facility location problems: Models, techniques, and applications in waste management," *Recycling*, vol. 5, no. 2, pp. 10–0, 2020.

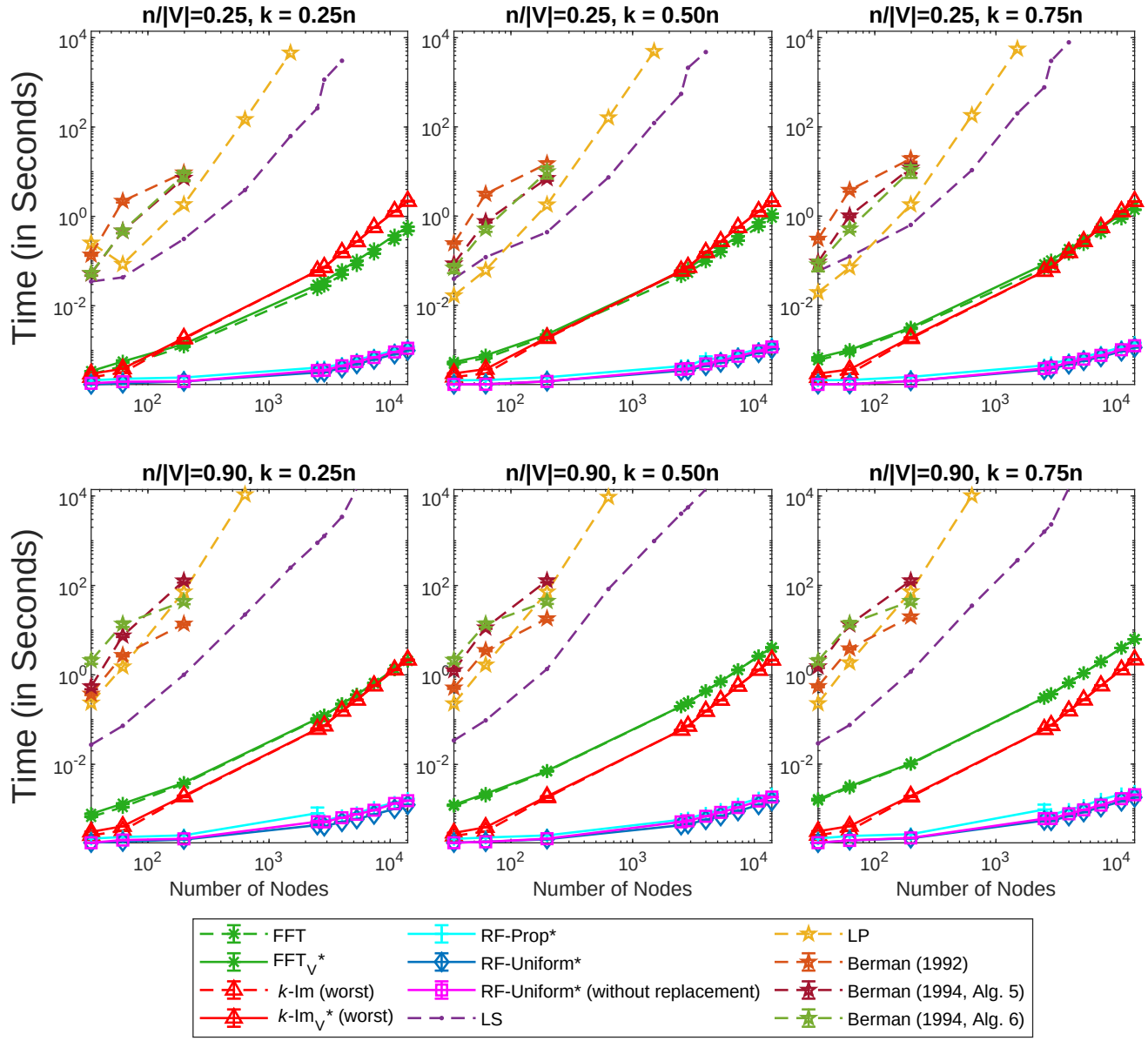

Fig. 5: Scalability of the considered algorithms using $|V|$ as network size when $n = 0.25|V|$ (top row) and $n = 0.90|V|$ (bottom row) with (left) $0.25n$, (center) $0.50n$, and (right) $0.75n$ added edges. Results from k-Im and k-Im$_V$ are condensed (only the centrality with the longest running time showed i.e., betweenness centrality for both).

[24] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009, vol. 2.

[25] J. Kleinberg and E. Tardos, *Algorithm Design*. Addison-Wesley Longman Publishing Co., Inc., 2005.

[26] J. A. Mesa and T. Brian Boffey, "A review of extensive facility location in networks," *European Journal of Operational Research*, vol. 95, no. 3, pp. 592–603, 1996.

[27] A. Balakrishnan, J. E. Ward, and R. T. Wong, "Integrated facility location and vehicle routing models: Recent work and future prospects," *American Journal of Mathematical and Management Sciences*, vol. 7, no. 1-2, pp. 35–61, 01 1987.

[28] A. Klose and A. Drexl, "Facility location models for distribution system design," *European Journal of Operational Research*, vol. 162, no. 1, pp. 4–29, 2005.

[29] O. Berman, D. I. Ingco, and A. R. Odoni, "Improving the location of minisum facilities through network modification," *Annals of Operations Research*, vol. 40, pp. 1–16, 1992.

[30] O. Berman, D. I. Ingco, and A. Odoni, "Improving the location of minimax facilities through network modification," *Networks*, vol. 24, no. 1, pp. 31–41, 1994.

[31] S. Perumal, P. Basu, and Z. Guan, "Minimizing eccentricity in composite networks via constrained edge additions," in *MILCOM 2013-2013 IEEE Military Communications Conference*. IEEE, 2013, pp. 1894–1899.

[32] T. M. Cover and J. A. Thomas, Eds., *Elements of Information Theory*. Wiley, 2006.

[33] J. R. BELL, "Subgroup centrality measures," *Network Science*, vol. 2, no. 2, pp. 277–297, 2014.

[34] M. G. Everett and S. P. Borgatti, "The centrality of groups and classes," *J. Math. Sociol.*, vol. 23, no. 3, pp. 181–201, 01 1999.
[35] M. Charikar, S. Khuller, D. M. Mount, and G. Narasimhan, "Algorithms for facility location problems with outliers," in *SODA*, vol. 1, 2001, pp. 642–651.
[36] A. Meyerson and B. Tagiku, "Minimizing average shortest path distances via shortcut edge addition," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques: 12th International Workshop, APPROX 2009, and 13th International Workshop, RANDOM 2009, Berkeley, CA, USA, August 21-23, 2009. Proceedings*. Springer, 2009, pp. 272–285.
[37] H. Chan, A. Filos-Ratsikas, B. Li, M. Li, and C. Wang, "Mechanism design for facility location problems: A survey," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, Z.-H. Zhou, Ed., 2021, pp. 4356–4365.
[38] E. Dokow, M. Feldman, R. Meir, and I. Nehama, "Mechanism design on discrete lines and cycles," in *Proceedings of the 13th ACM Conference on Electronic Commerce (ACM-EC)*, 2012, pp. 423–440.
[39] D. Fotakis and C. Tzamos, "On the power of deterministic mechanisms for facility location games," *ACM Transactions on Economics and Computation (TEAC)*, vol. 2, no. 4, pp. 1–37, 2014.
[40] A. D. Procaccia and M. Tennenholtz, "Approximate mechanism design without money," *ACM Transactions on Economics and Computation (TEAC)*, vol. 1, no. 4, pp. 1–26, 2013.
[41] J. Schummer and R. V. Vohra, "Strategy-proof location on a network," *Journal of Economic Theory*, vol. 104, no. 2, pp. 405–428, 2002.
[42] F. Gul, "A nobel prize for game theorists: The contributions of harsanyi, nash and selten," *Journal of Economic Perspectives*, vol. 11, no. 3, pp. 159–174, September 1997.
[43] D. Garg, Y. Narahari, and S. Gujar, "Foundations of mechanism design: A tutorial part 1-key concepts and classical results," *Sadhana*, vol. 33, pp. 83–130, 04 2008.
[44] ——, "Foundations of mechanism design: A tutorial part 2-advanced concepts and results," *Sadhana*, vol. 33, no. 2, pp. 131–174, 2008.
[45] Prize Committee of the Royal Swedish Academy of Sciences, "Mechanism design theory," https://www.nobelprize.org/uploads/2018/06/advanced-economicsciences2007.pdf.
[46] E. D. Demaine and M. Zadimoghaddam, "Minimizing the diameter of a network using shortcut edges," in *Algorithm Theory-SWAT 2010: 12th Scandinavian Symposium and Workshops on Algorithm Theory, Bergen, Norway, June 21-23, 2010. Proceedings 12*. Springer, 2010, pp. 420–431.
[47] D. Bilò, L. Gualà, and G. Proietti, "Improved approximability and non-approximability results for graph diameter decreasing problems," *Theoretical Computer Science*, vol. 417, pp. 12–22, 2012.
[48] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, Eds., *Algorithmic Game Theory*. Cambridge University Press, 2011.
[49] T. F. Gonzalez, "Clustering to minimize the maximum intercluster distance," *Theoretical computer science*, vol. 38, pp. 293–306, 1985.
[50] P. Erdos, A. Rényi *et al.*, "On the evolution of random graphs," *Publication of the Mathematical Institute of the Hungarian Academy of Sciences*, vol. 5, no. 1, pp. 17–60, 1960.
[51] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *AAAI*, 2015. [Online]. Available: https://networkrepository.com
[52] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, Jun. 2014.
[53] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, pp. 35–41, 1977.
[54] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Tech. Rep., 1999.
[55] P. Bonacich, "Factoring and weighting approaches to status scores and clique identification," *Journal of mathematical sociology*, vol. 2, no. 1, pp. 113–120, 1972.
[56] D. J. Watts and S. H. Strogatz, "Collective dynamics of "small-world" networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
[57] A. Bavelas, "Communication patterns in task-oriented groups," *The journal of the acoustical society of America*, vol. 22, no. 6, pp. 725–730, 1950.
[58] P. Hage and F. Harary, "Eccentricity and centrality in networks," *Social networks*, vol. 17, no. 1, pp. 57–63, 1995.
[59] H. Tong, B. A. Prakash, T. Eliassi-Rad, M. Faloutsos, and C. Faloutsos, "Gelling, and melting, large graphs by edge manipulation," in *Proceedings of the 21st ACM international conference on Information and knowledge management*, 2012, pp. 245–254.

APPENDIX

A. *Complete Experimental Results on Effectiveness for Synthetic Networks*

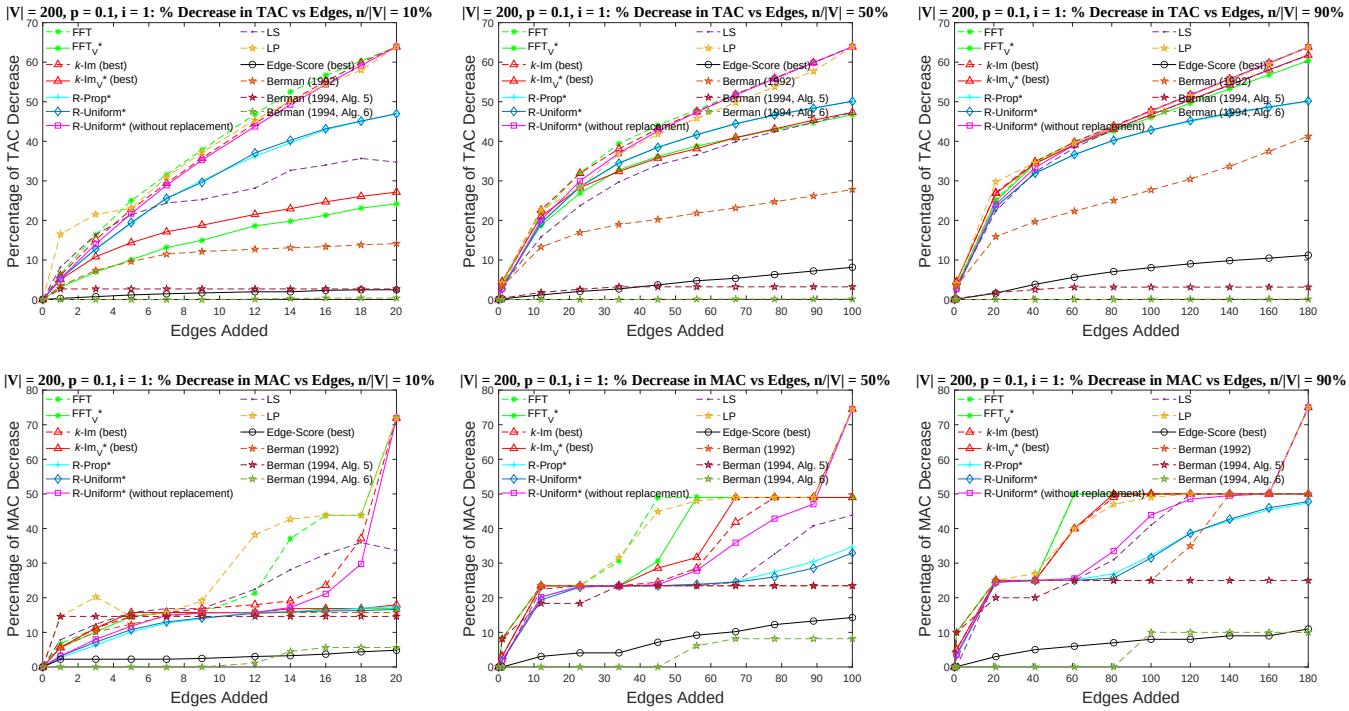

Fig. 6: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the first instance ($i = 1$) of the synthetic network with $|V| = 200$ nodes and $p = 0.1$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

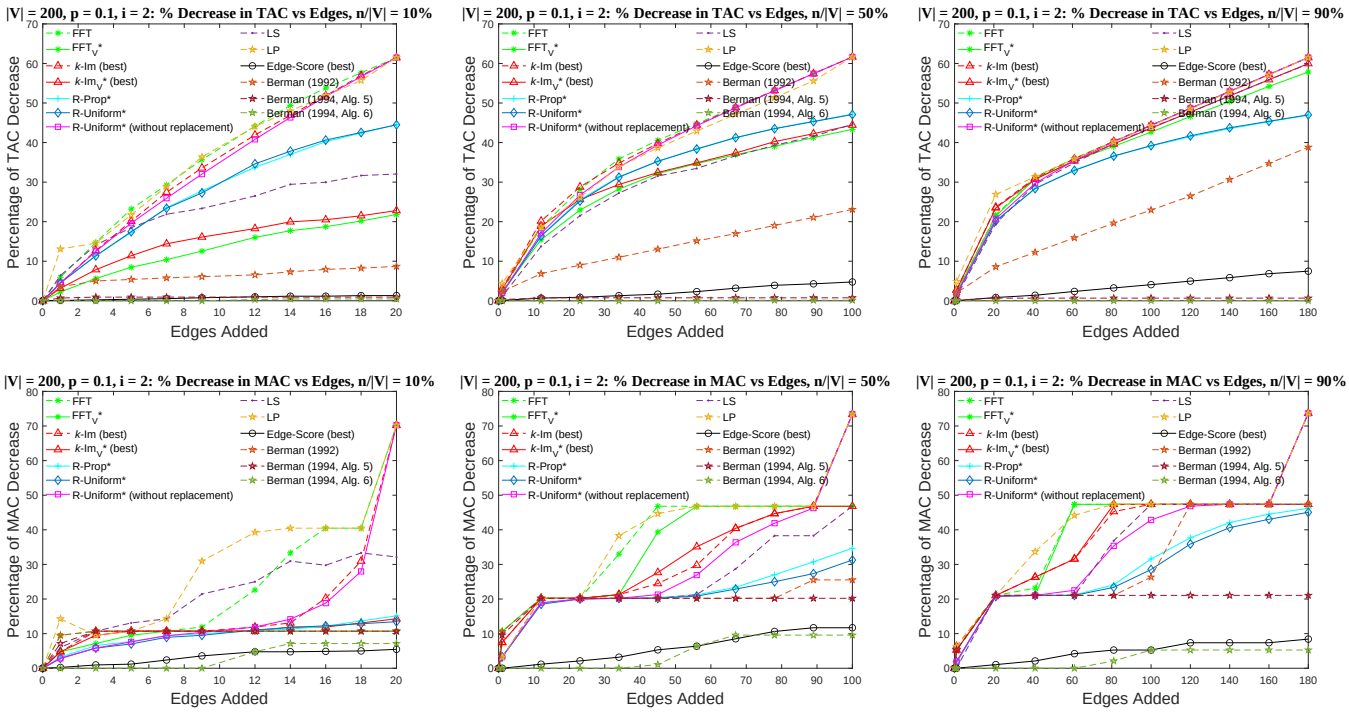

Fig. 7: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the second instance ($i = 2$) of the synthetic network with $|V| = 200$ nodes and $p = 0.1$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

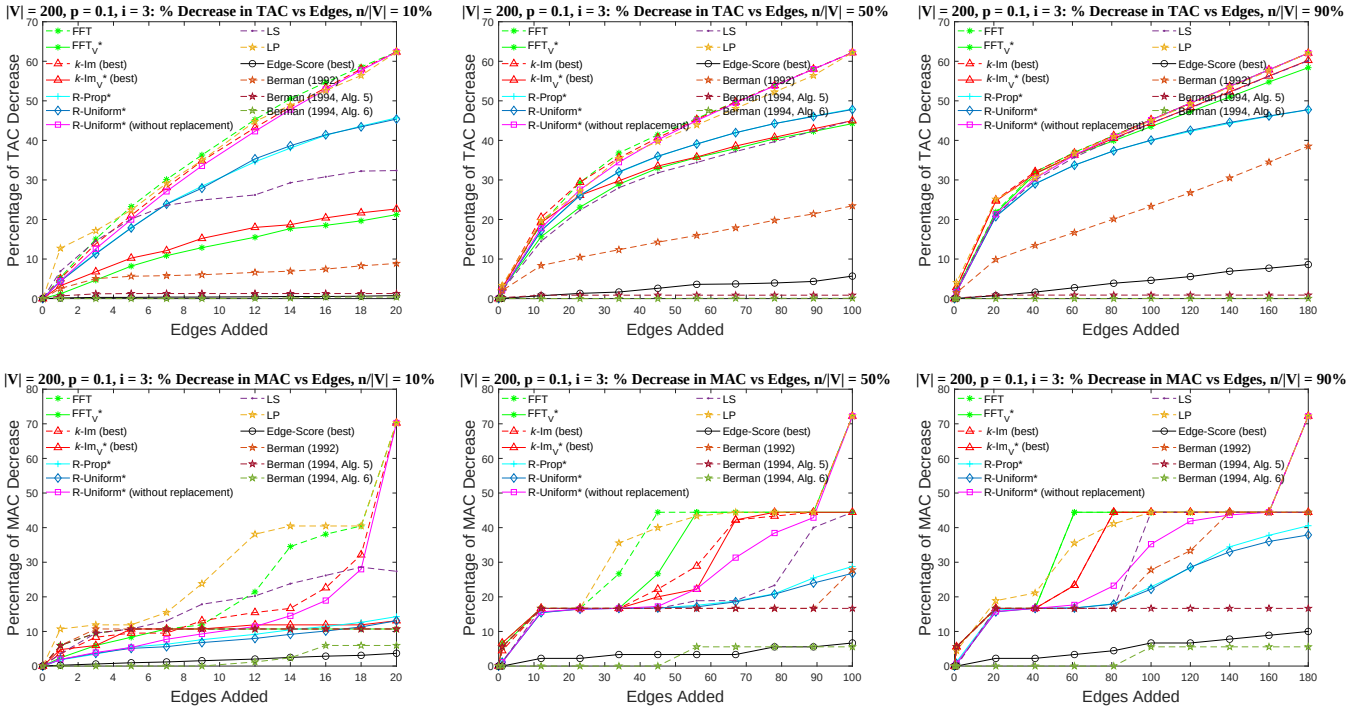

Fig. 8: (Same as Figure 2) Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the third instance ($i = 3$) of the synthetic network with $|V| = 200$ nodes and $p = 0.1$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

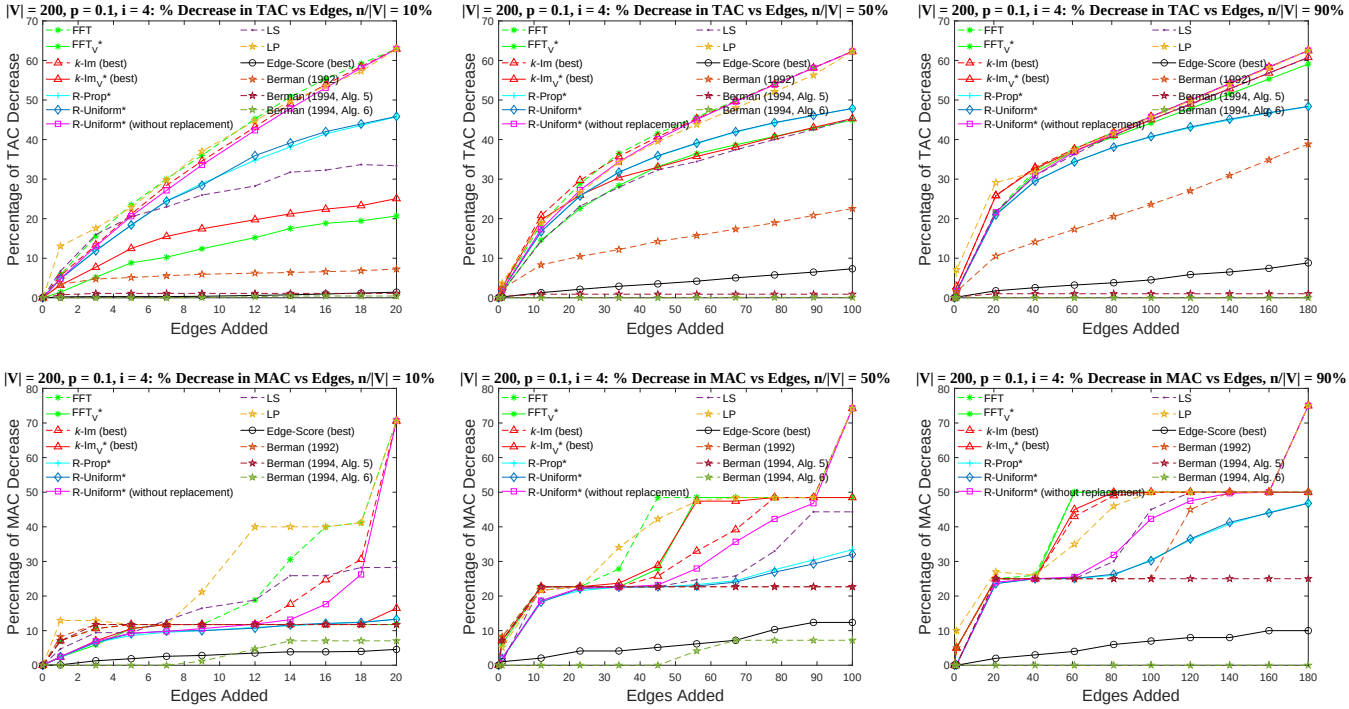

Fig. 9: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the fourth instance ($i = 4$) of the synthetic network with $|V| = 200$ nodes and $p = 0.1$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

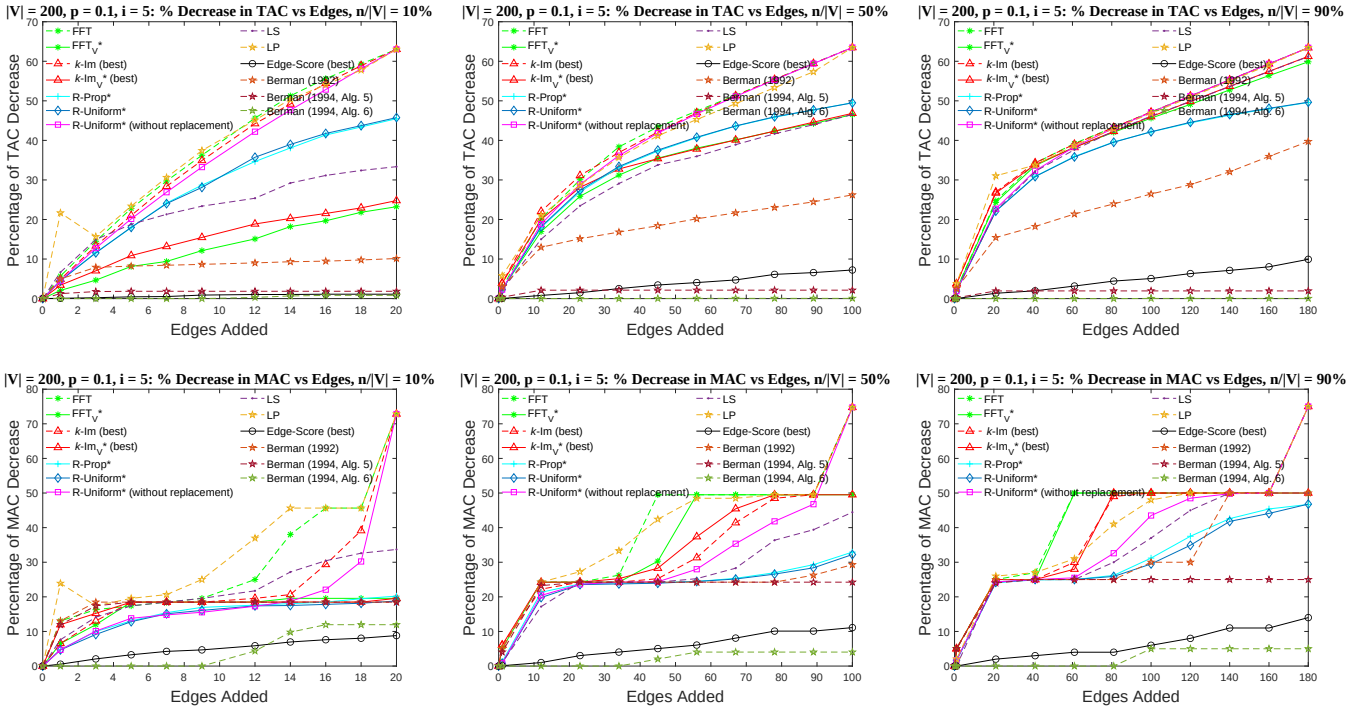

Fig. 10: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the fifth instance ($i = 5$) of the synthetic network with $|V| = 200$ nodes and $p = 0.1$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

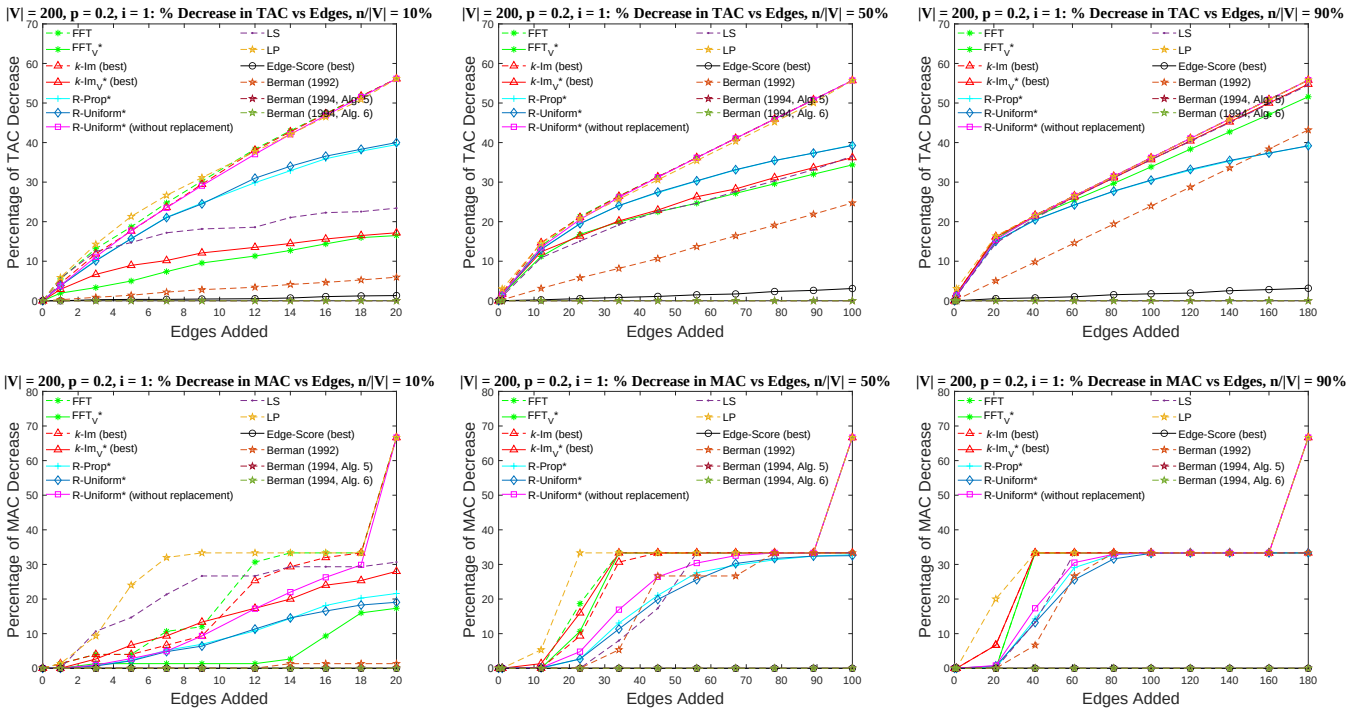

Fig. 11: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the first instance ($i = 1$) of the synthetic network with $|V| = 200$ nodes and $p = 0.2$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

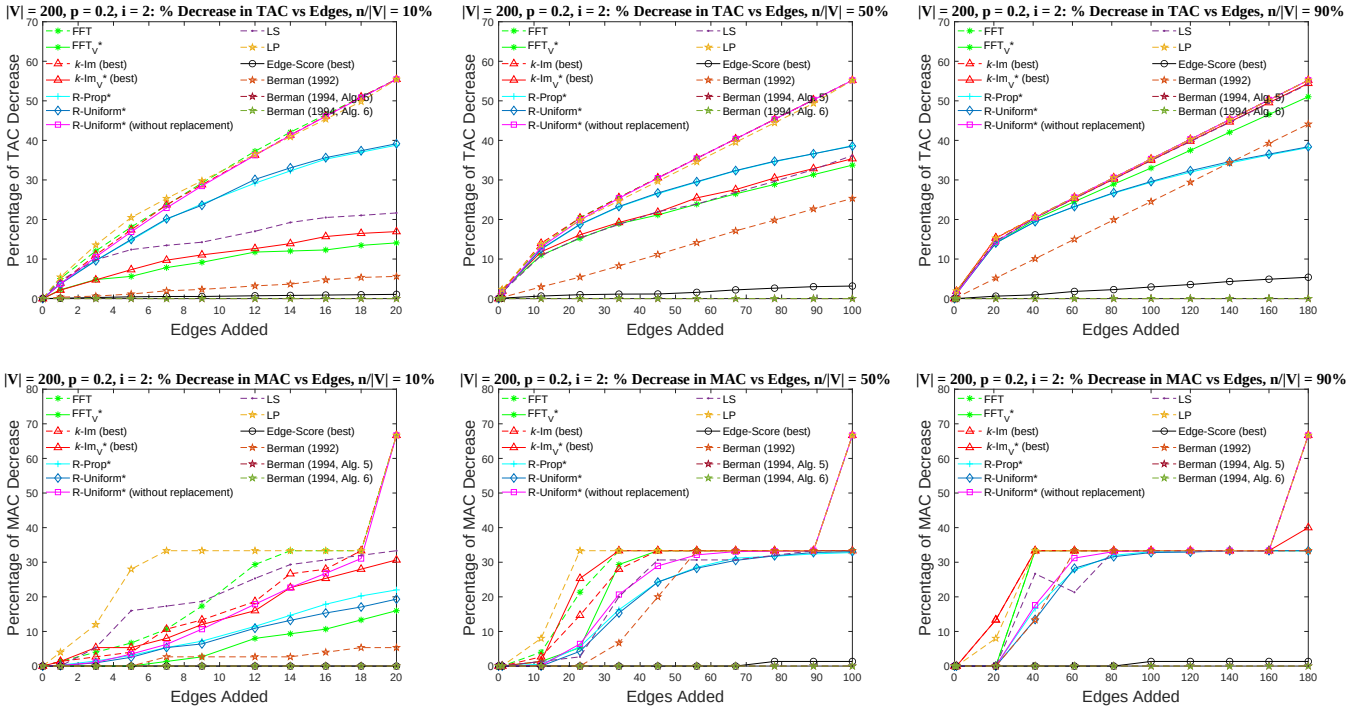

Fig. 12: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the second instance ($i = 2$) of the synthetic network with $|V| = 200$ nodes and $p = 0.2$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

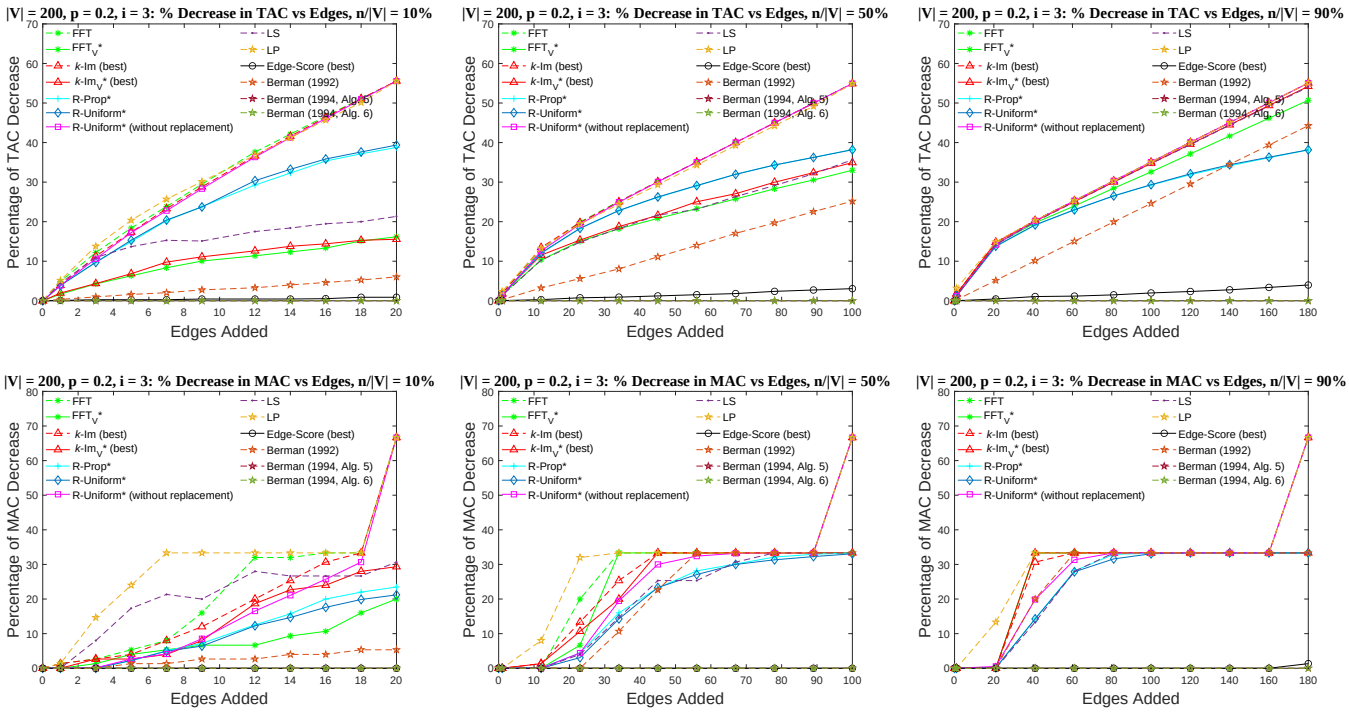

Fig. 13: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the third instance ($i = 3$) of the synthetic network with $|V| = 200$ nodes and $p = 0.2$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

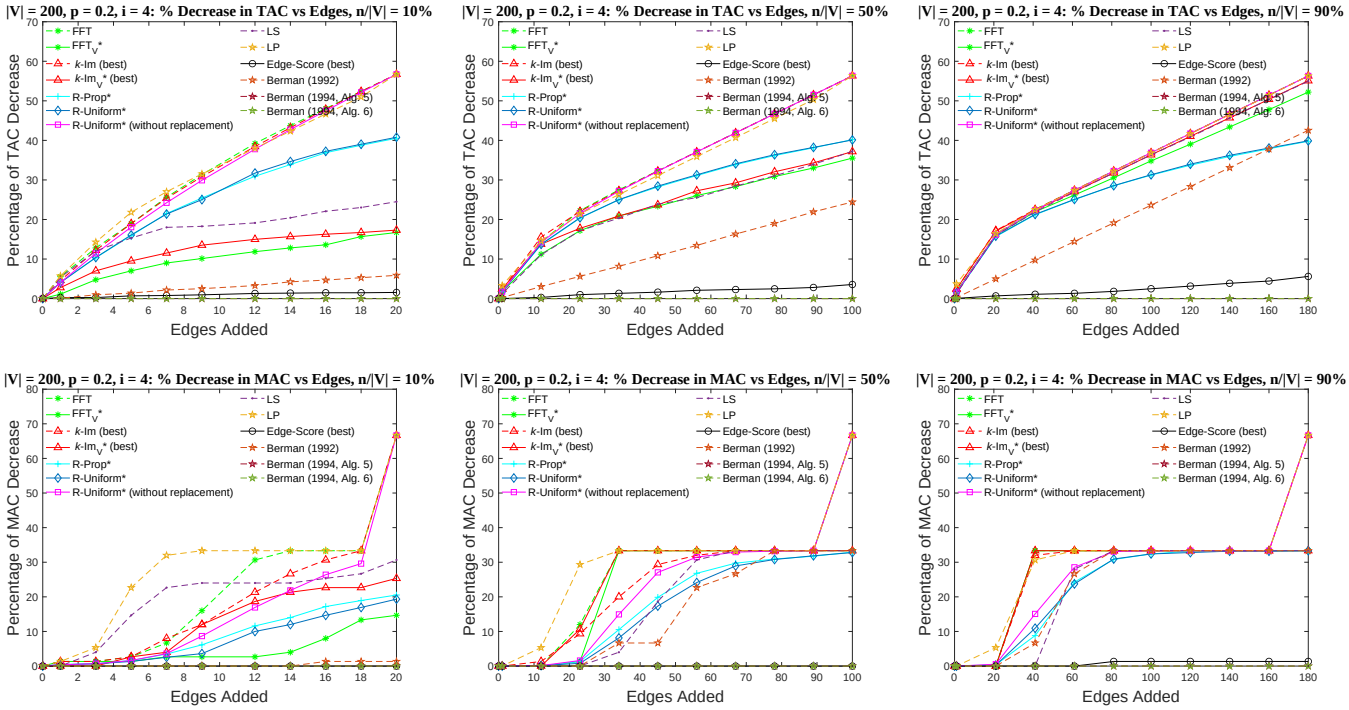

Fig. 14: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the fourth instance ($i = 4$) of the synthetic network with $|V| = 200$ nodes and $p = 0.2$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

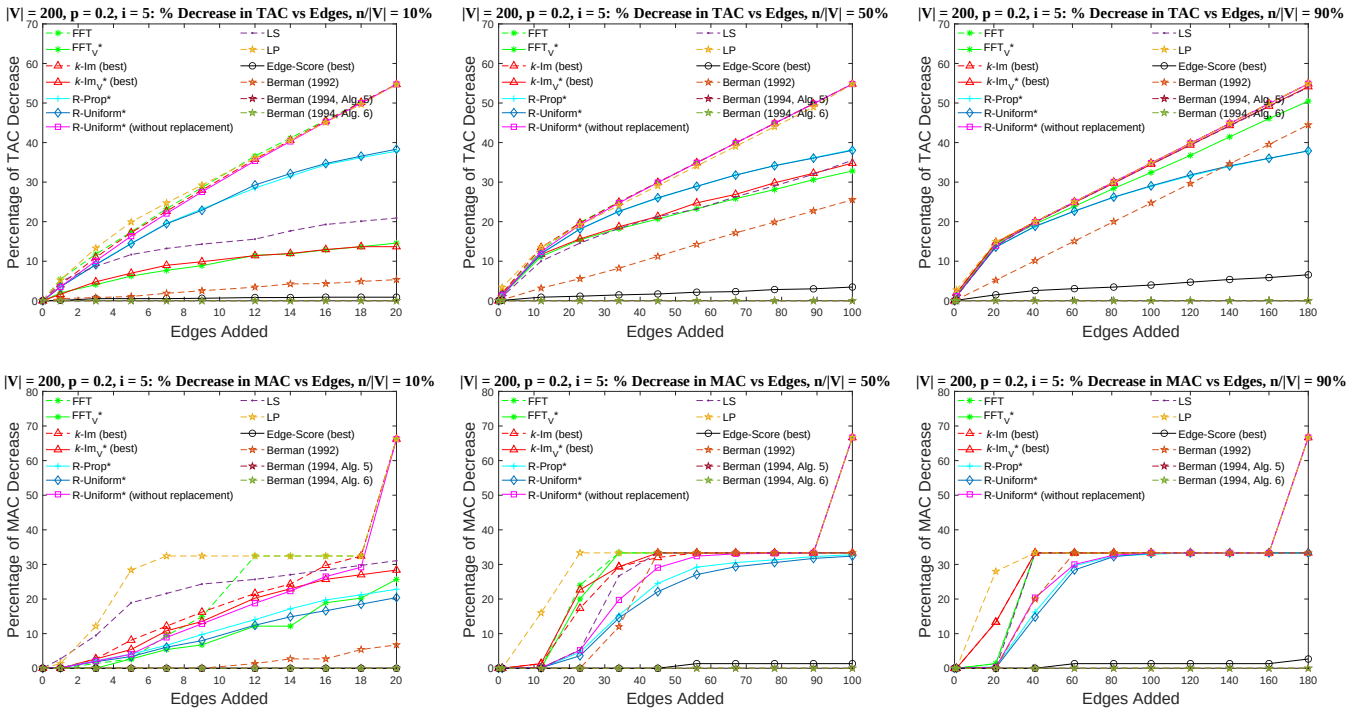

Fig. 15: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the fifth instance ($i = 5$) of the synthetic network with $|V| = 200$ nodes and $p = 0.2$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

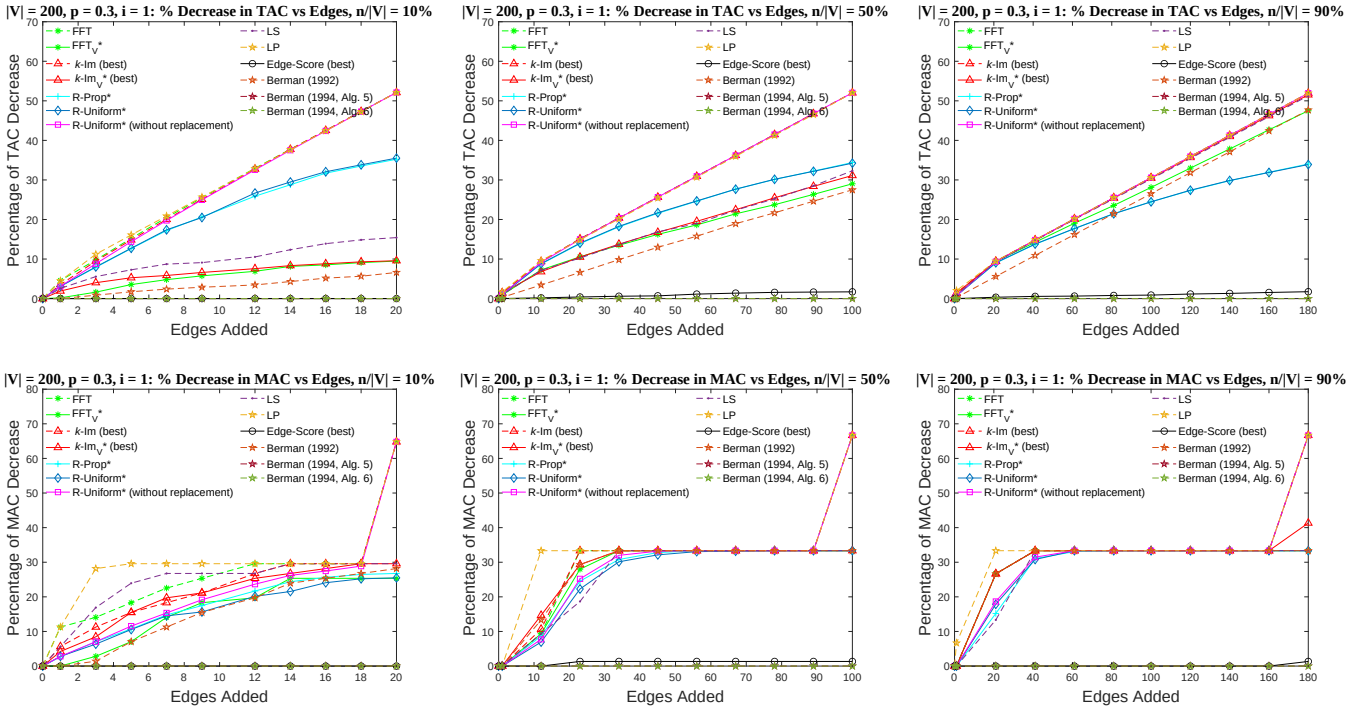

Fig. 16: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the first instance ($i = 1$) of the synthetic network with $|V| = 200$ nodes and $p = 0.3$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

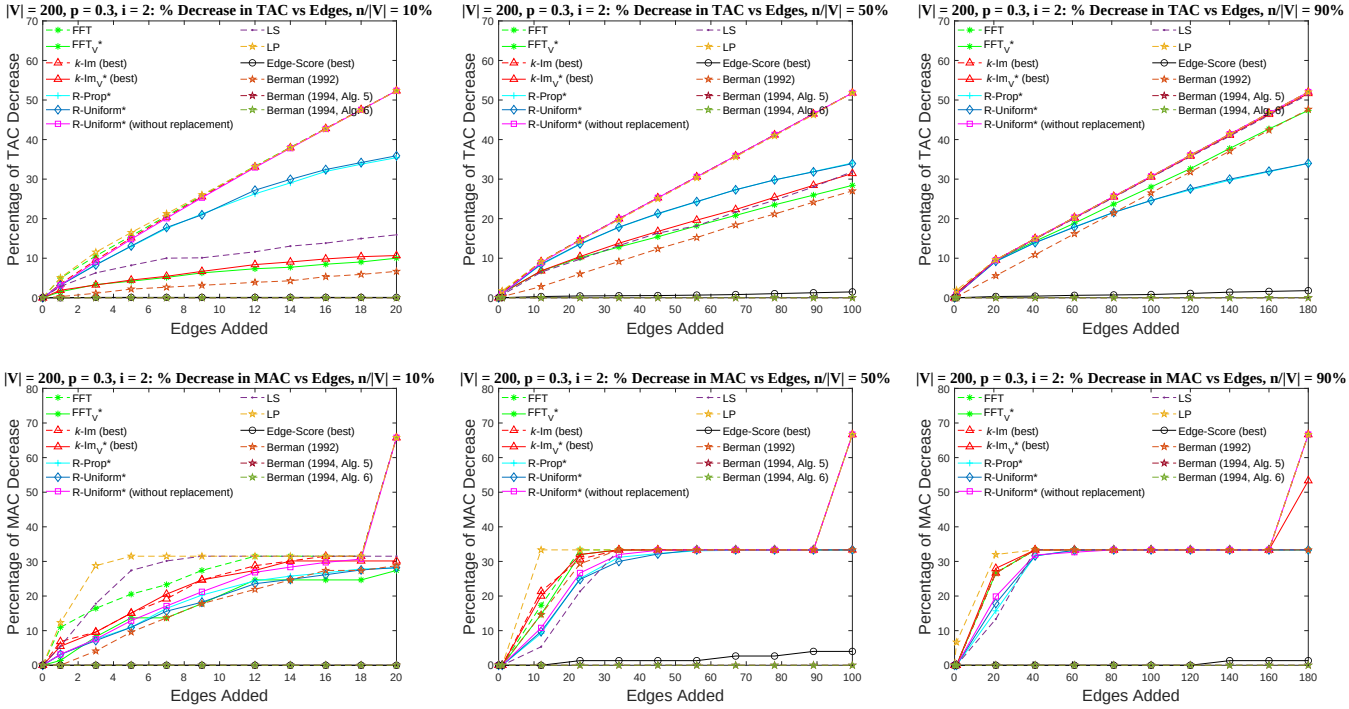

Fig. 17: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the second instance ($i = 2$) of the synthetic network with $|V| = 200$ nodes and $p = 0.3$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

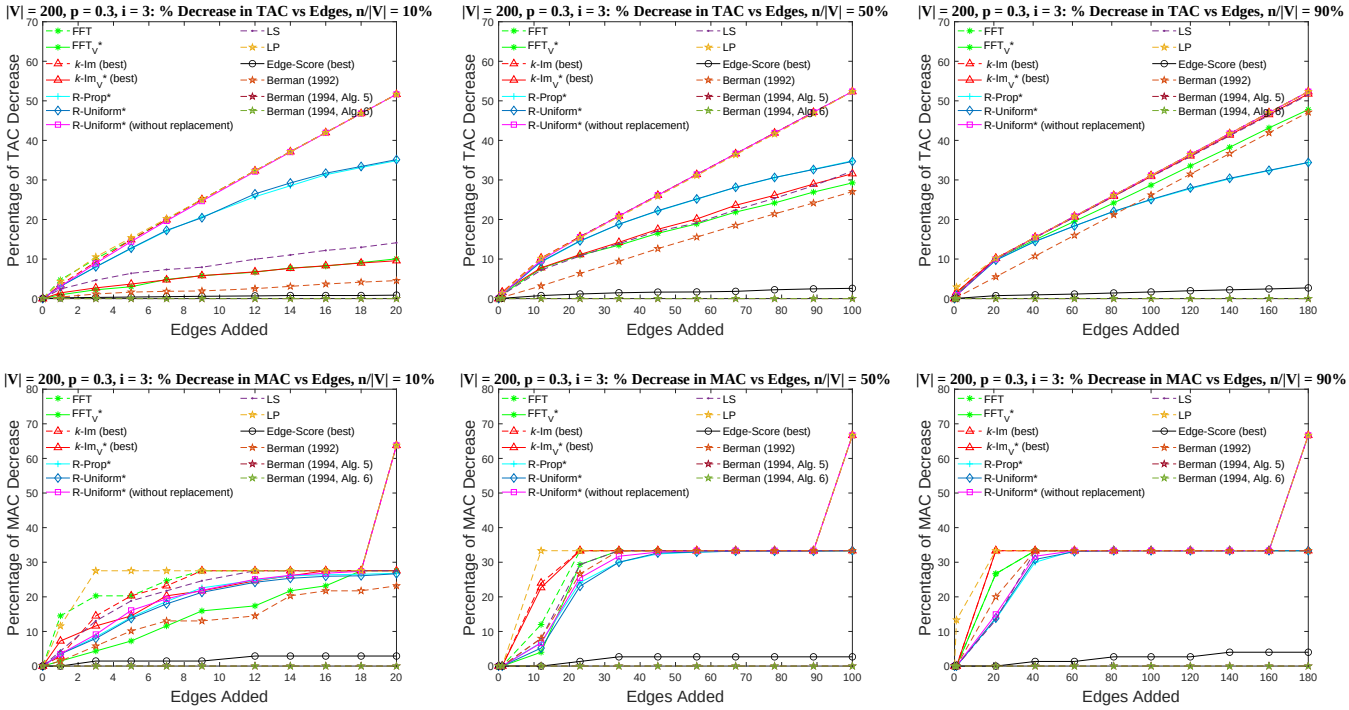

Fig. 18: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the third instance ($i = 3$) of the synthetic network with $|V| = 200$ nodes and $p = 0.3$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

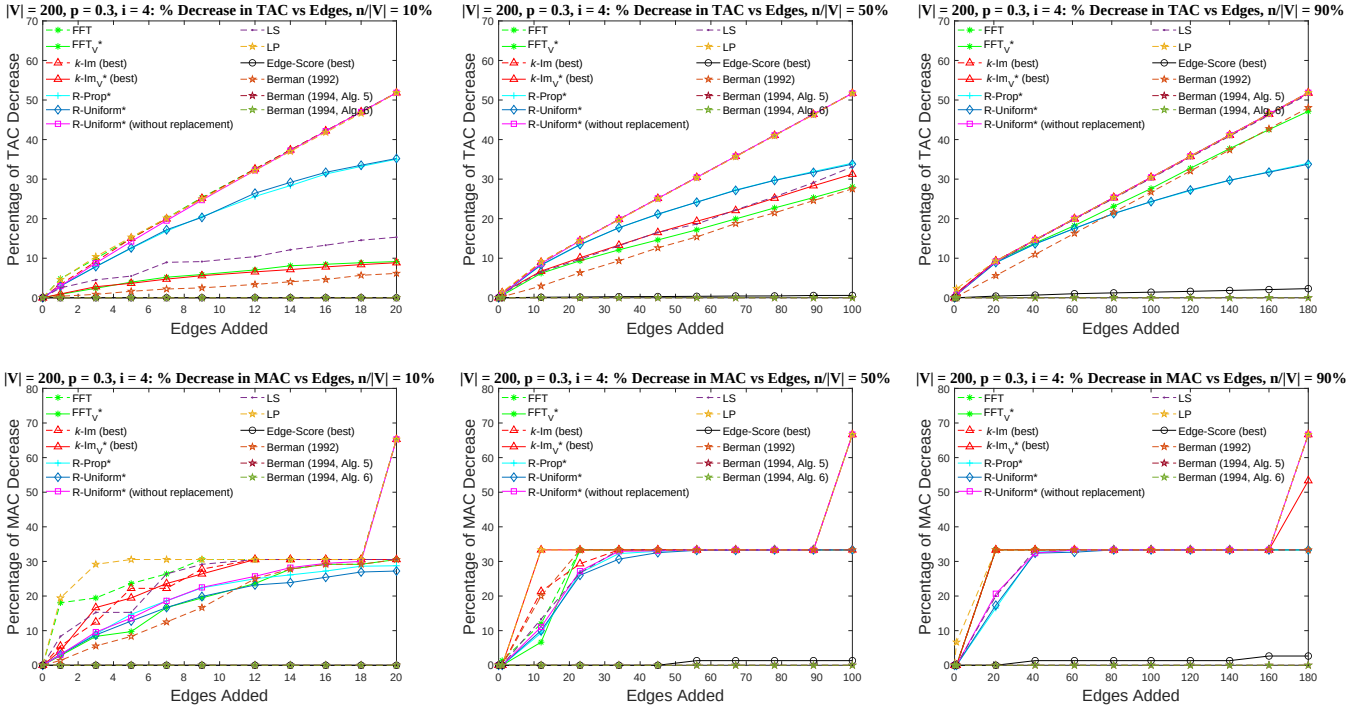

Fig. 19: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the fourth instance ($i = 4$) of the synthetic network with $|V| = 200$ nodes and $p = 0.3$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

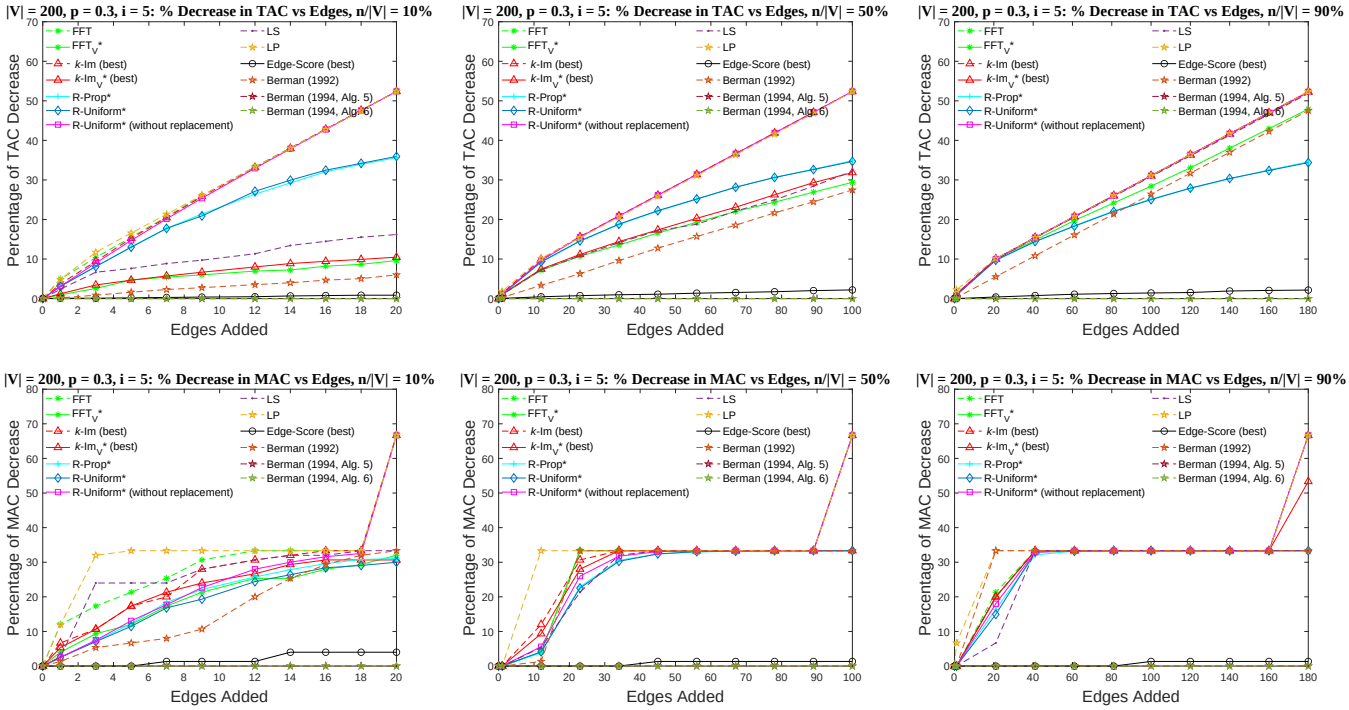

Fig. 20: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the fifth instance ($i = 5$) of the synthetic network with $|V| = 200$ nodes and $p = 0.3$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.9|V|$.

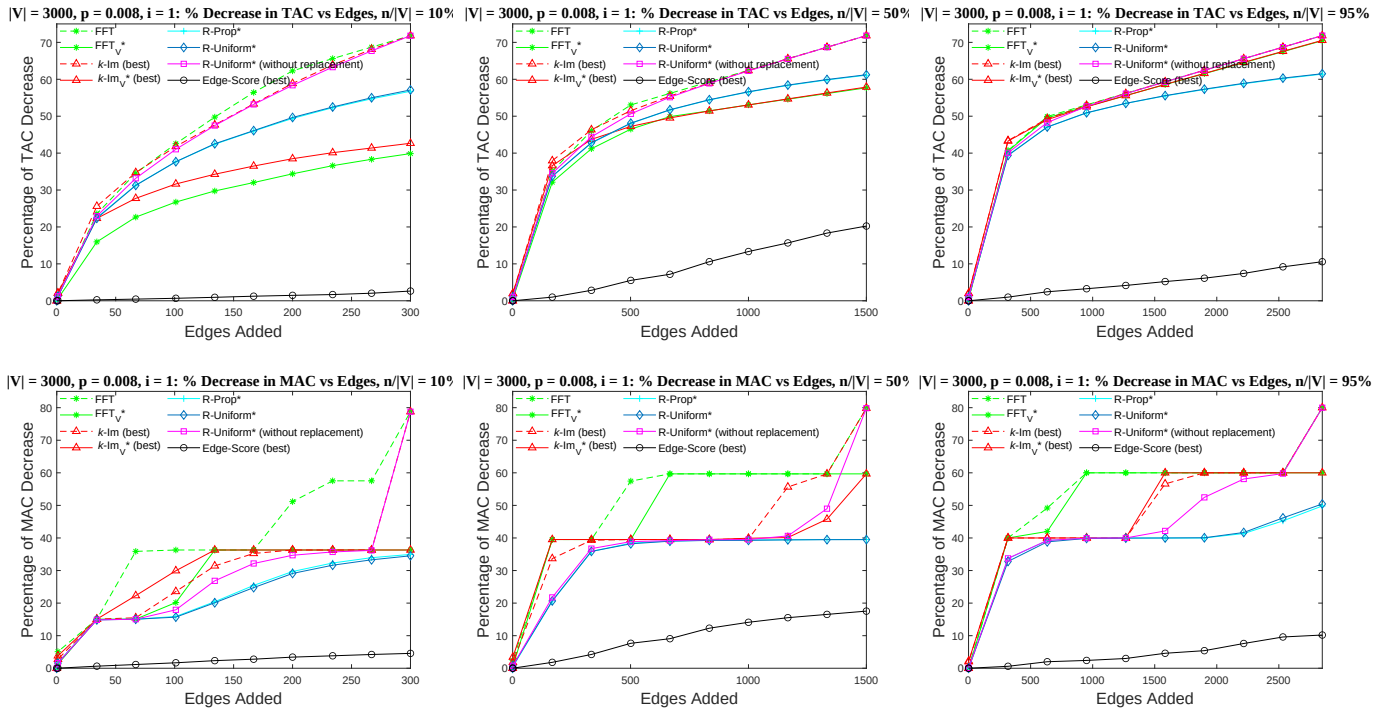

Fig. 21: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the first instance ($i = 1$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.008$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

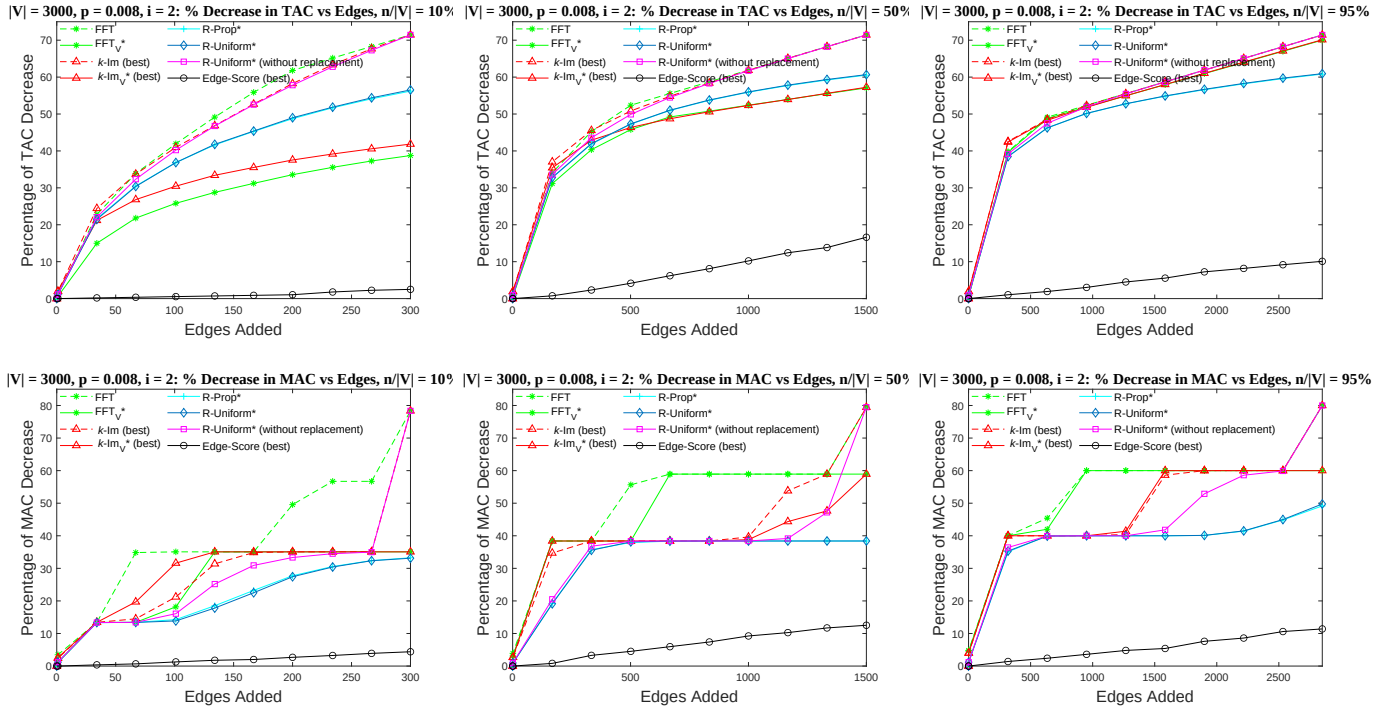

Fig. 22: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the second instance ($i = 2$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.008$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

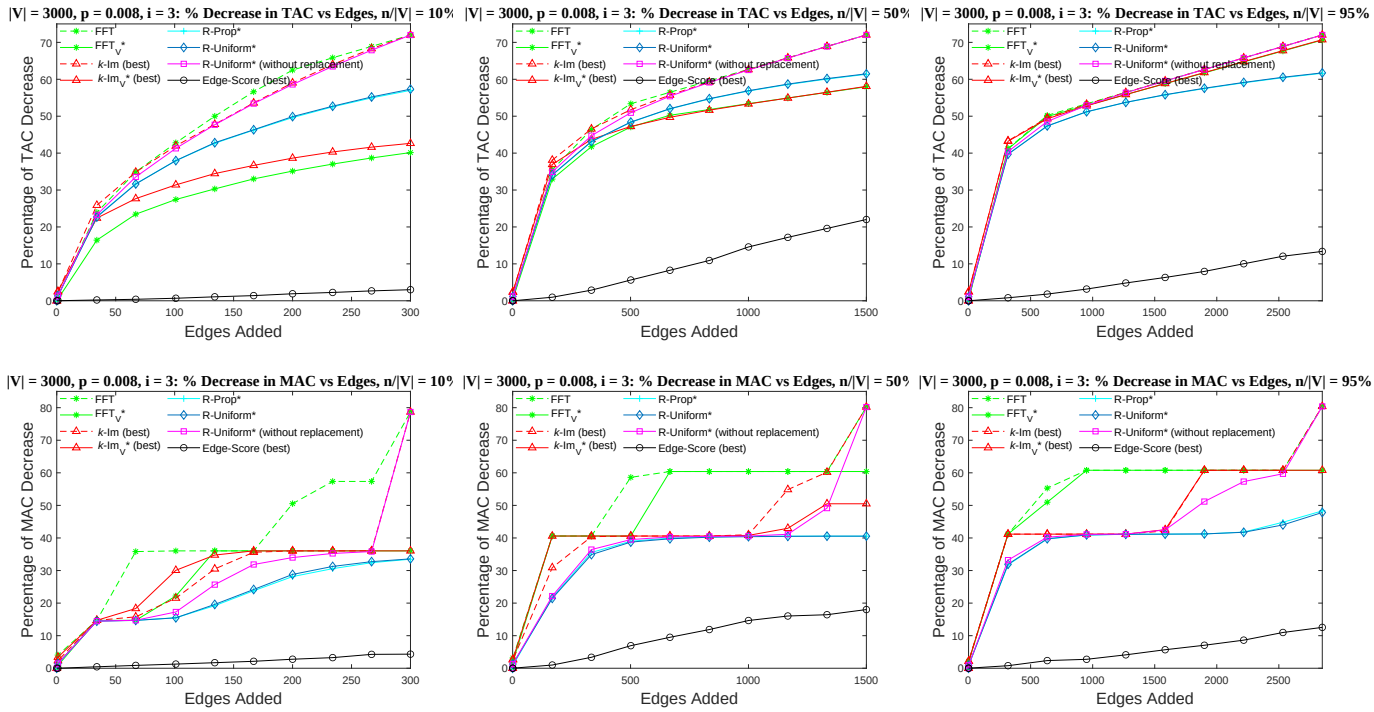

Fig. 23: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the third instance ($i = 3$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.008$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

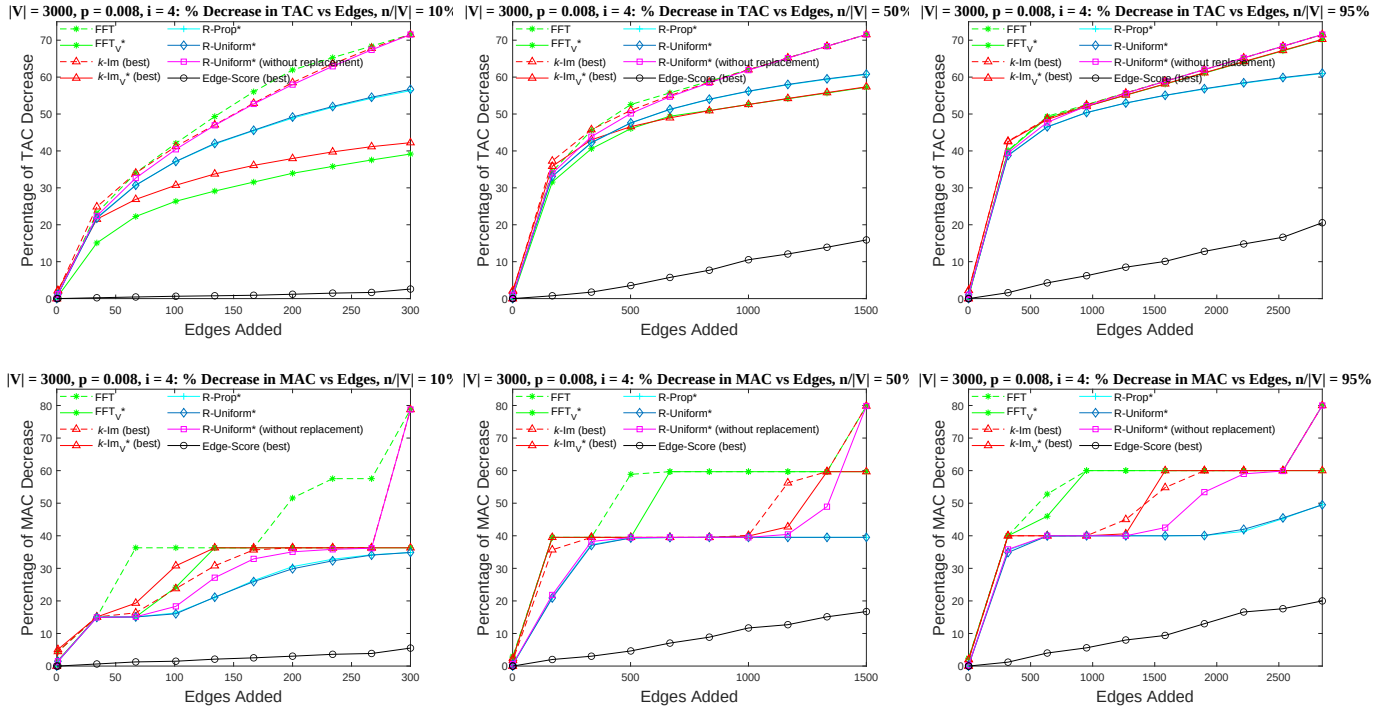

Fig. 24: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the fourth instance ($i = 4$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.008$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

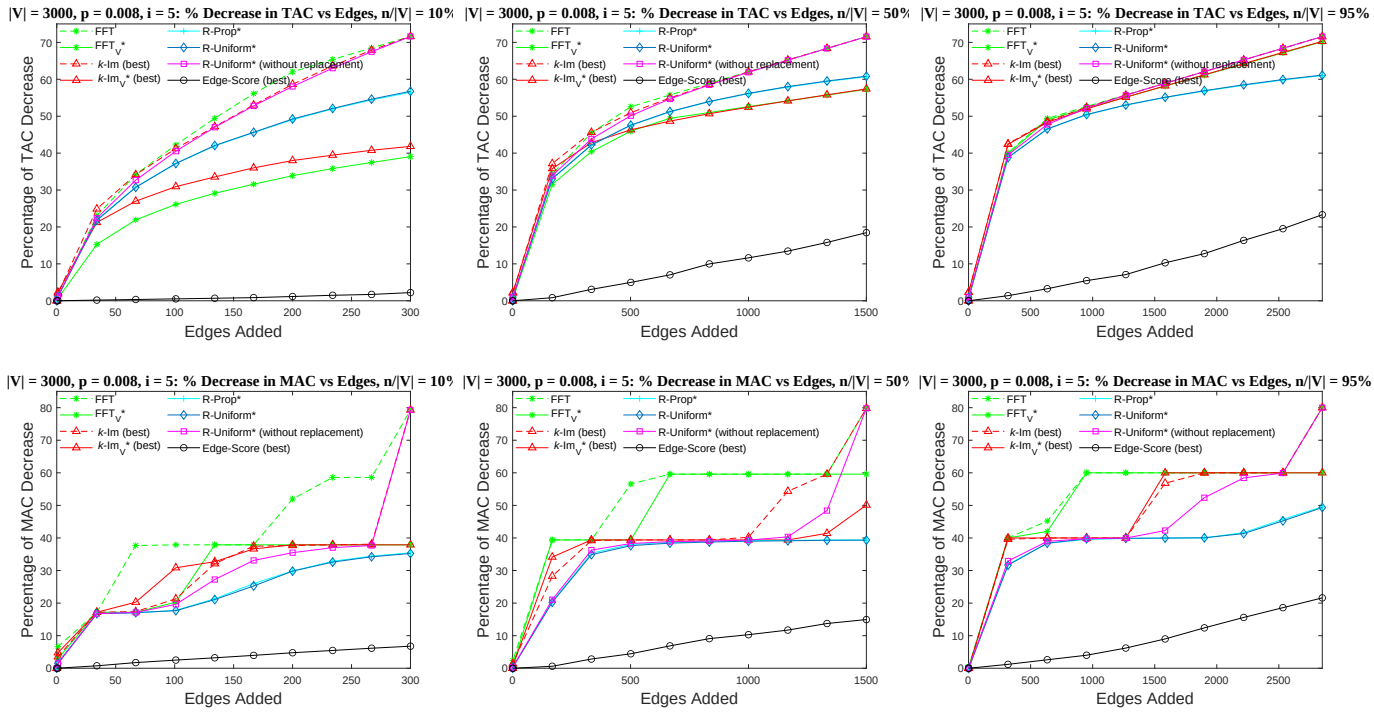

Fig. 25: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the fifth instance ($i = 5$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.008$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

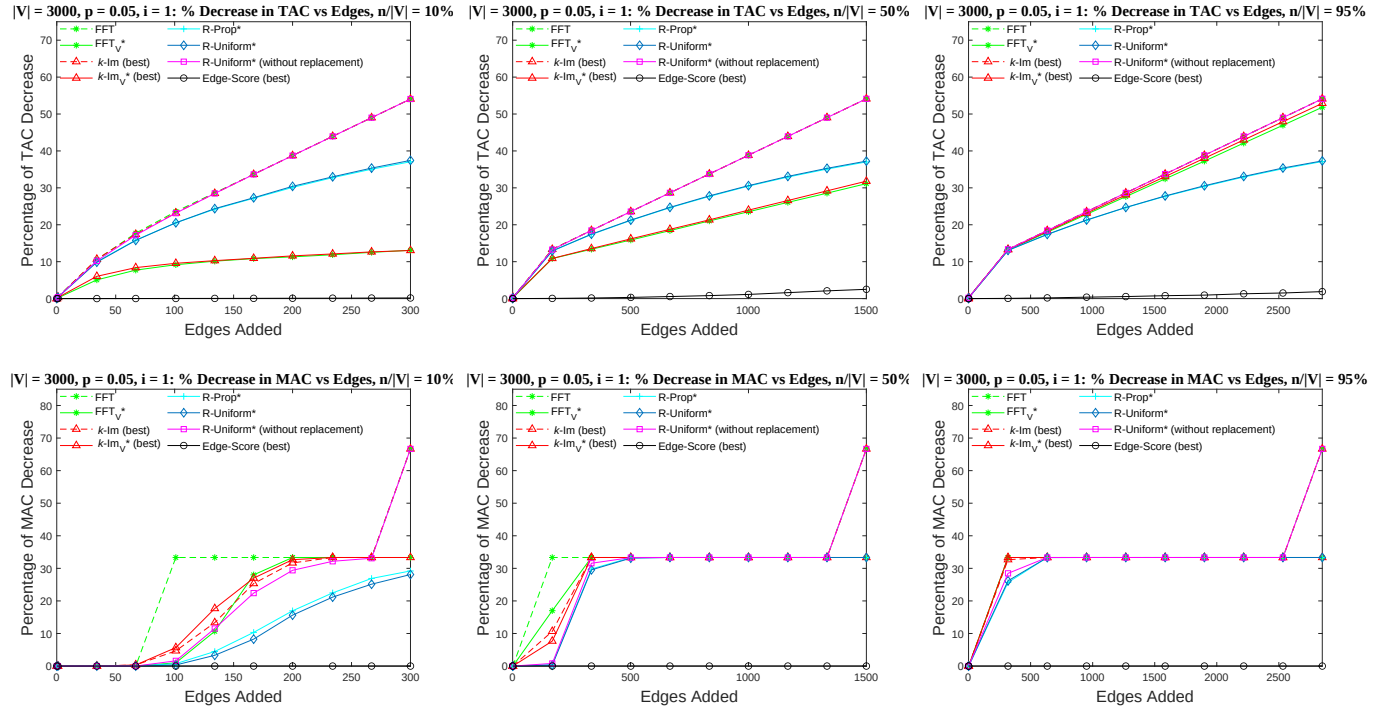

Fig. 26: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the first instance ($i = 1$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.05$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

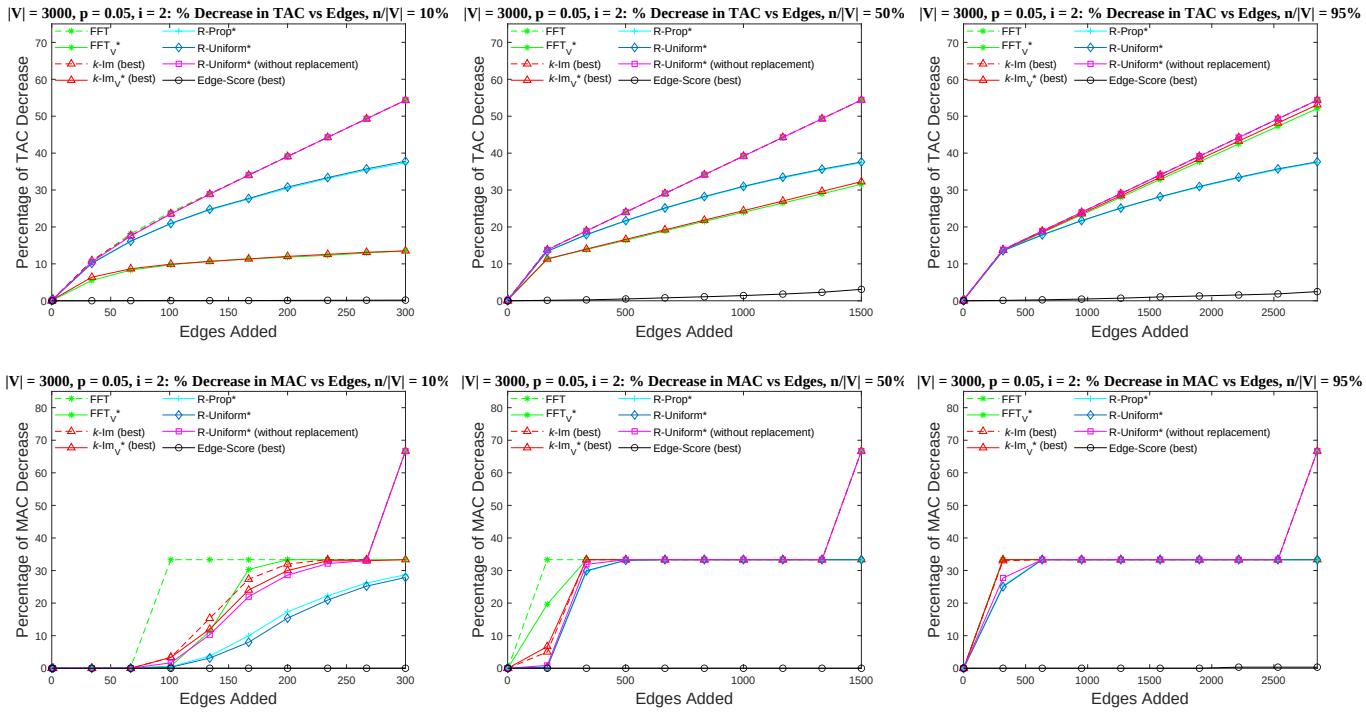

Fig. 27: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the second instance ($i = 2$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.05$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

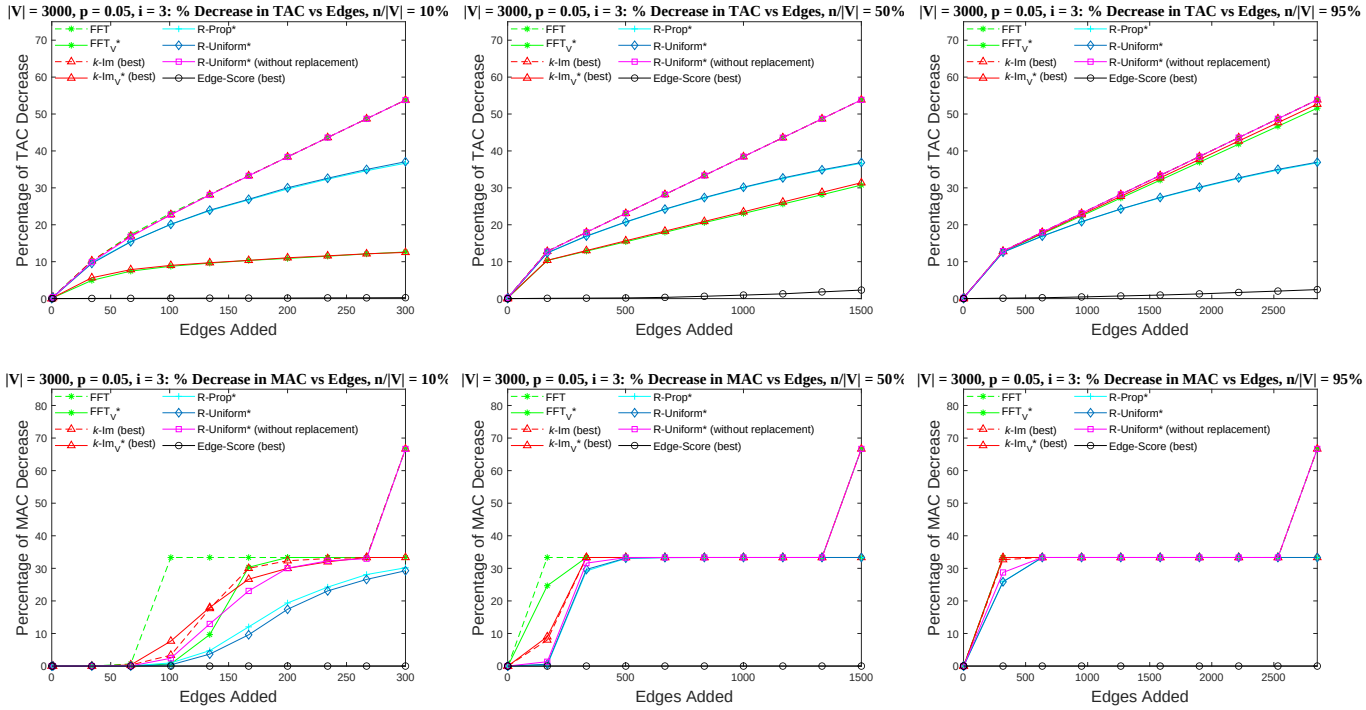

Fig. 28: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the third instance ($i = 3$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.05$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

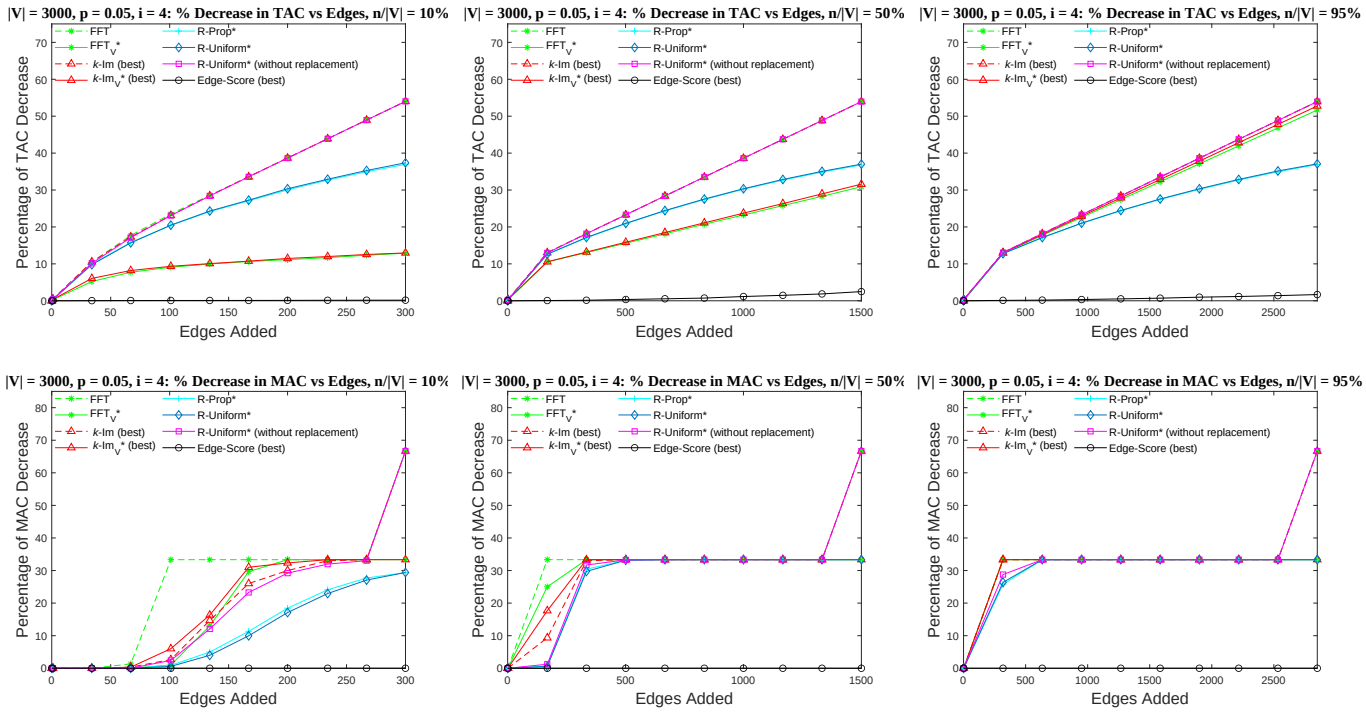

Fig. 29: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the fourth instance ($i = 4$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.05$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

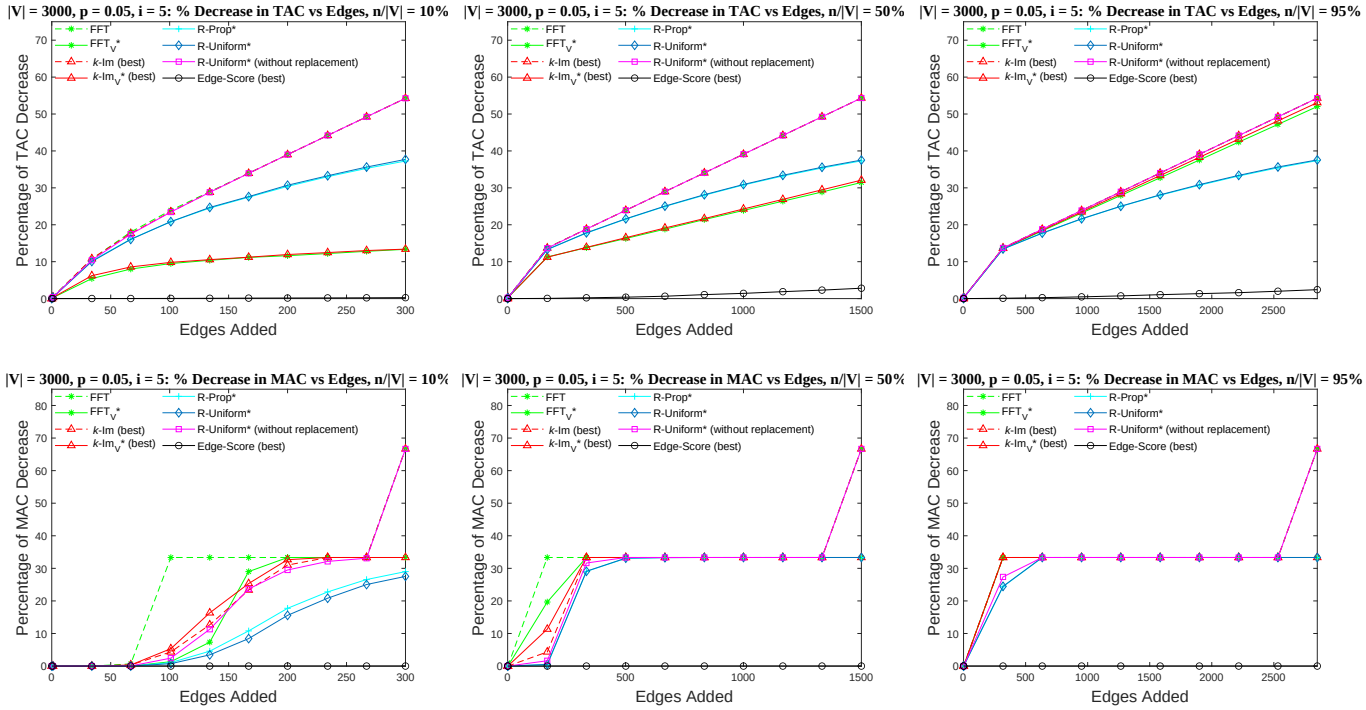

Fig. 30: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the fifth instance ($i = 5$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.05$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

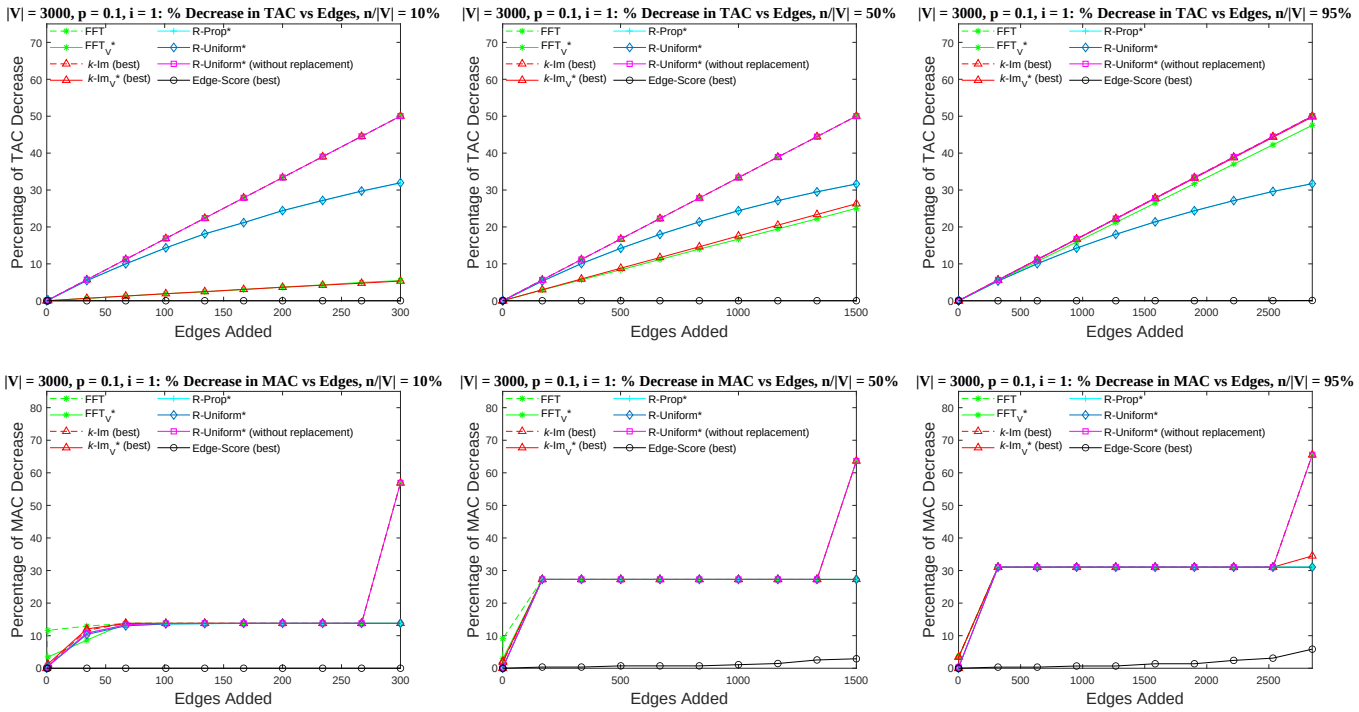

Fig. 31: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the first instance ($i = 1$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.1$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

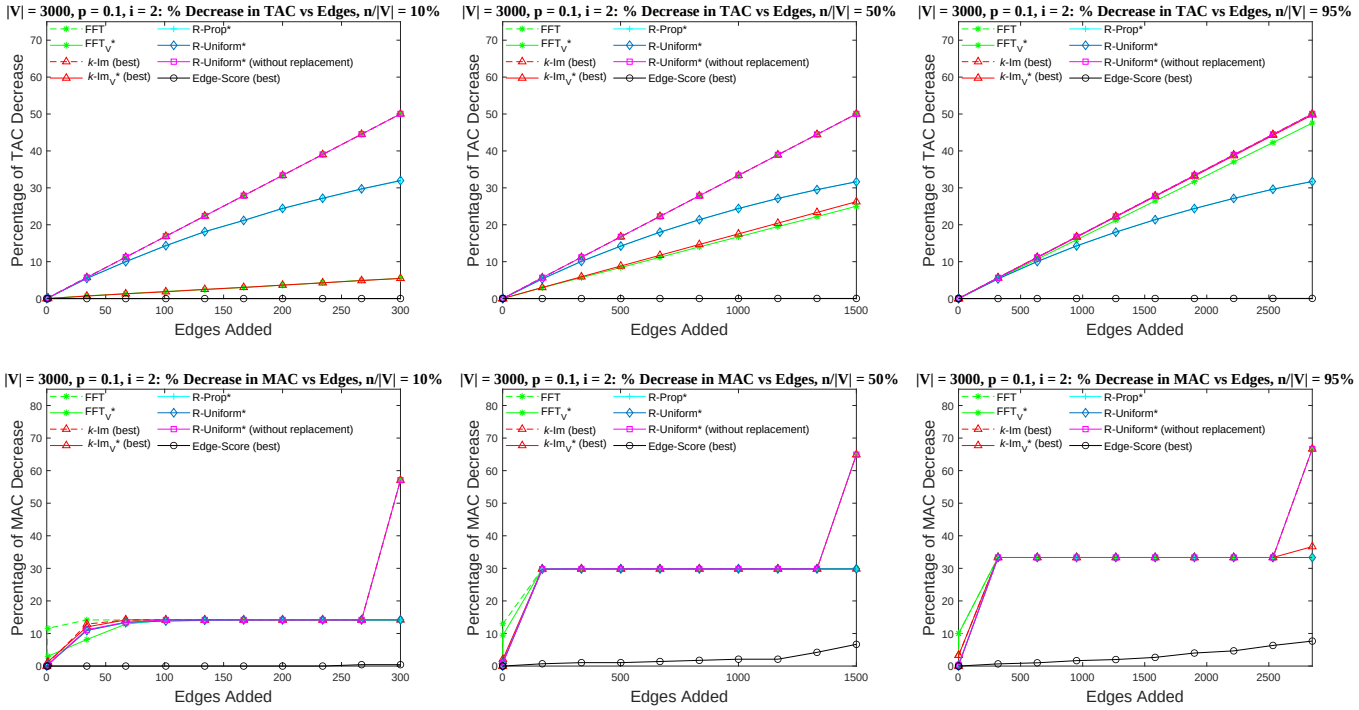

Fig. 32: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the second instance ($i = 2$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.1$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

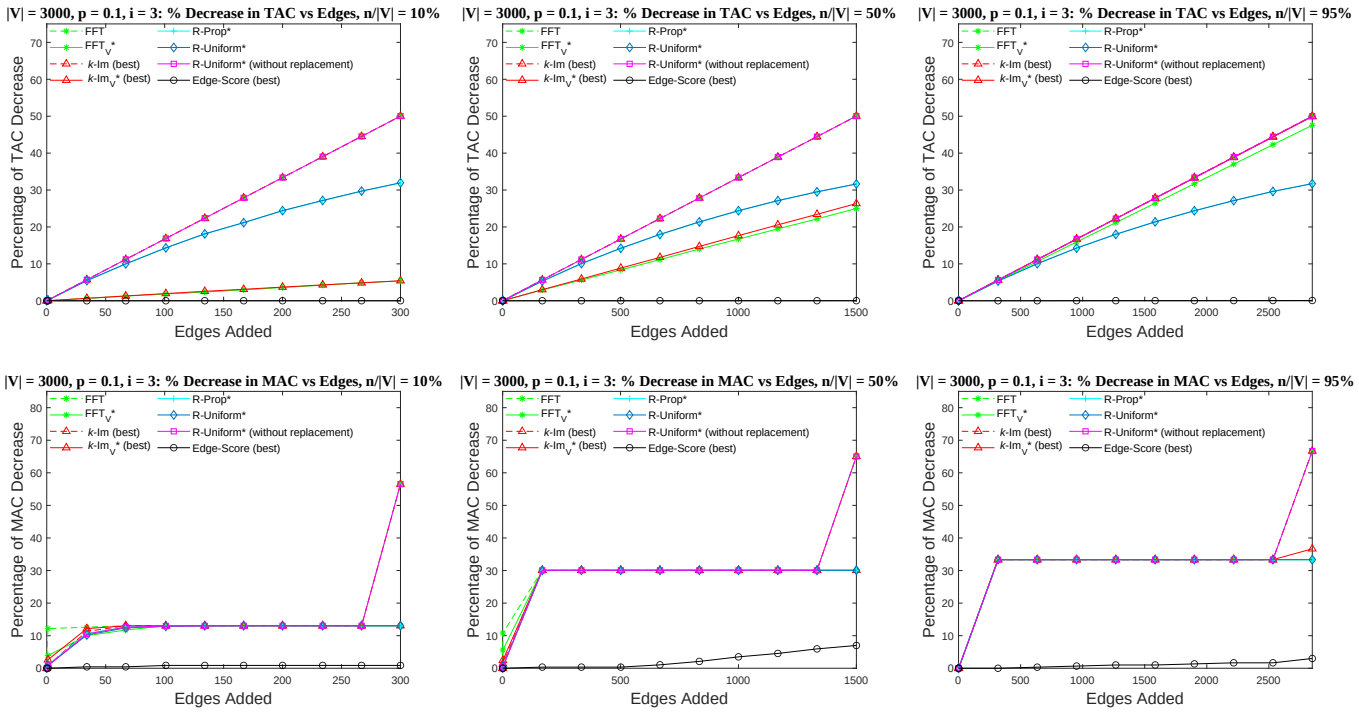

Fig. 33: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the third instance ($i = 3$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.1$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

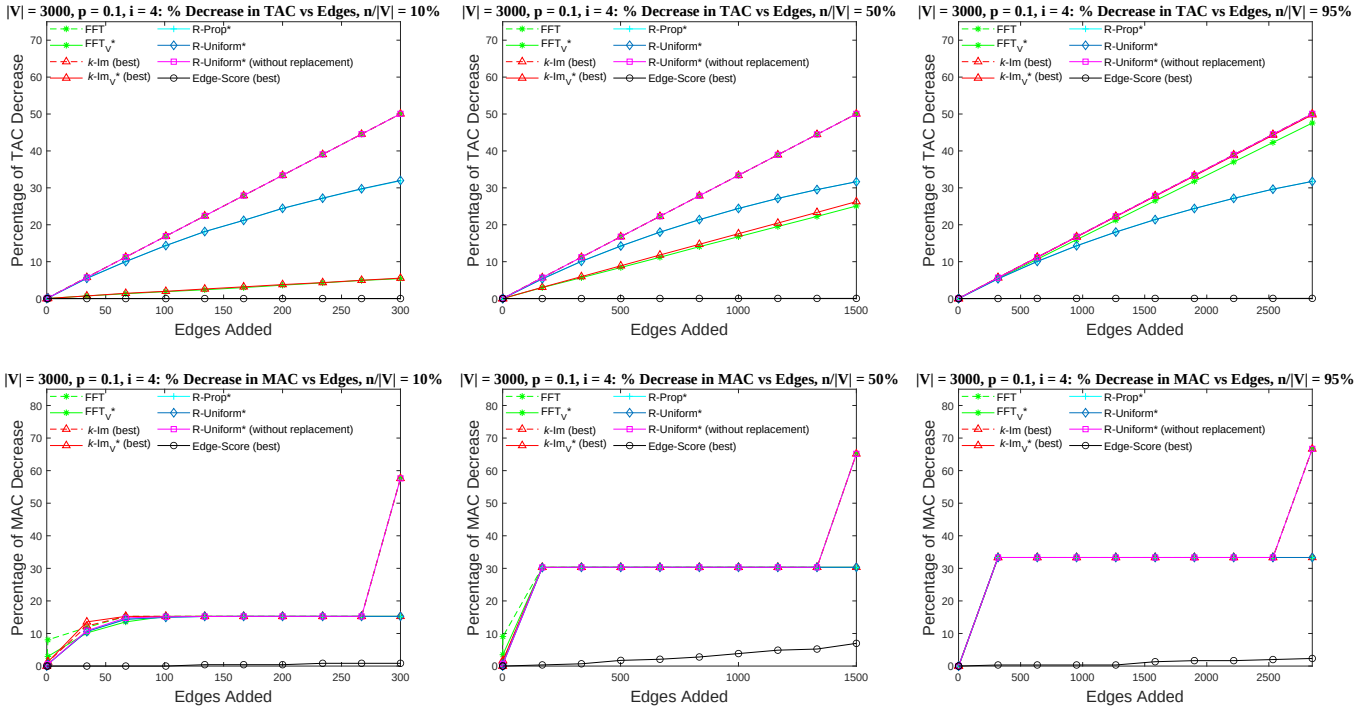

Fig. 34: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the fourth instance ($i = 4$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.1$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

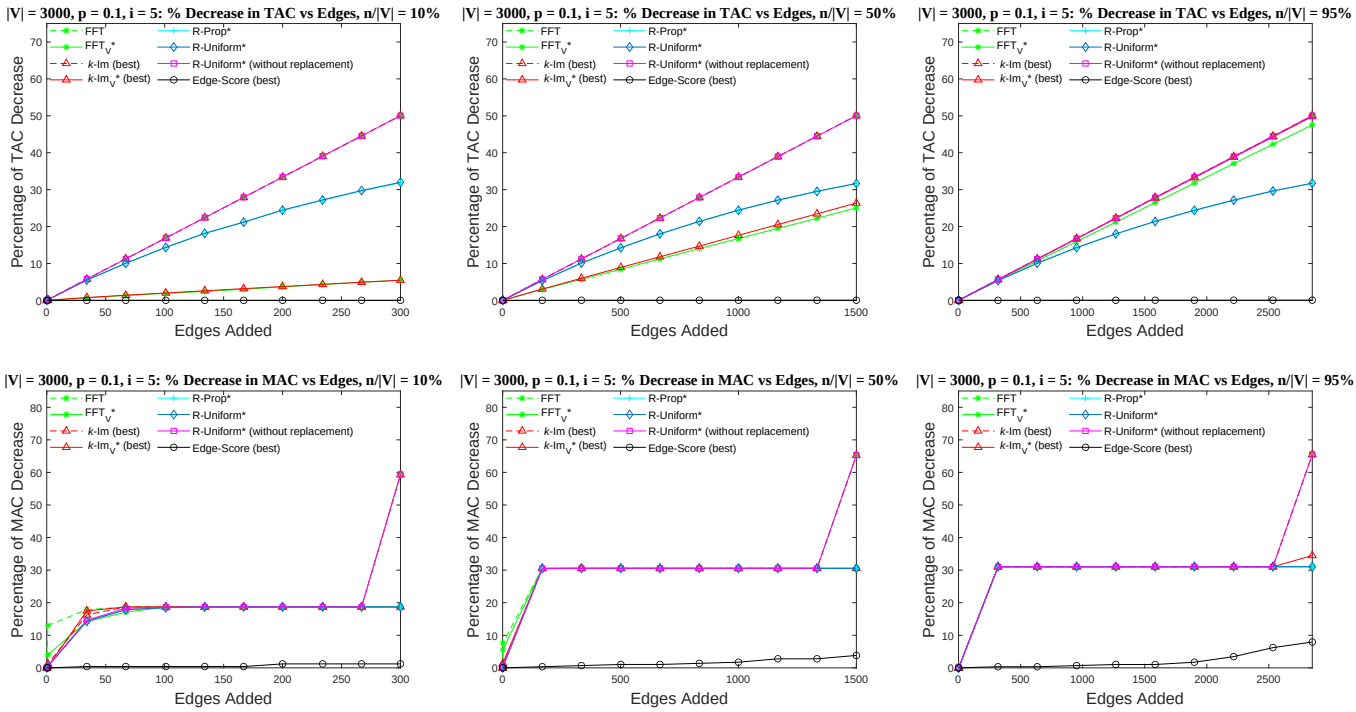

Fig. 35: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on the fifth instance ($i = 5$) of the synthetic network with $|V| = 3000$ nodes and $p = 0.1$, for different client sizes: (left) $0.1|V|$, (center) $0.5|V|$, and (right) $0.95|V|$.

B. Complete Experimental Results on Effectiveness for Real-world Networks

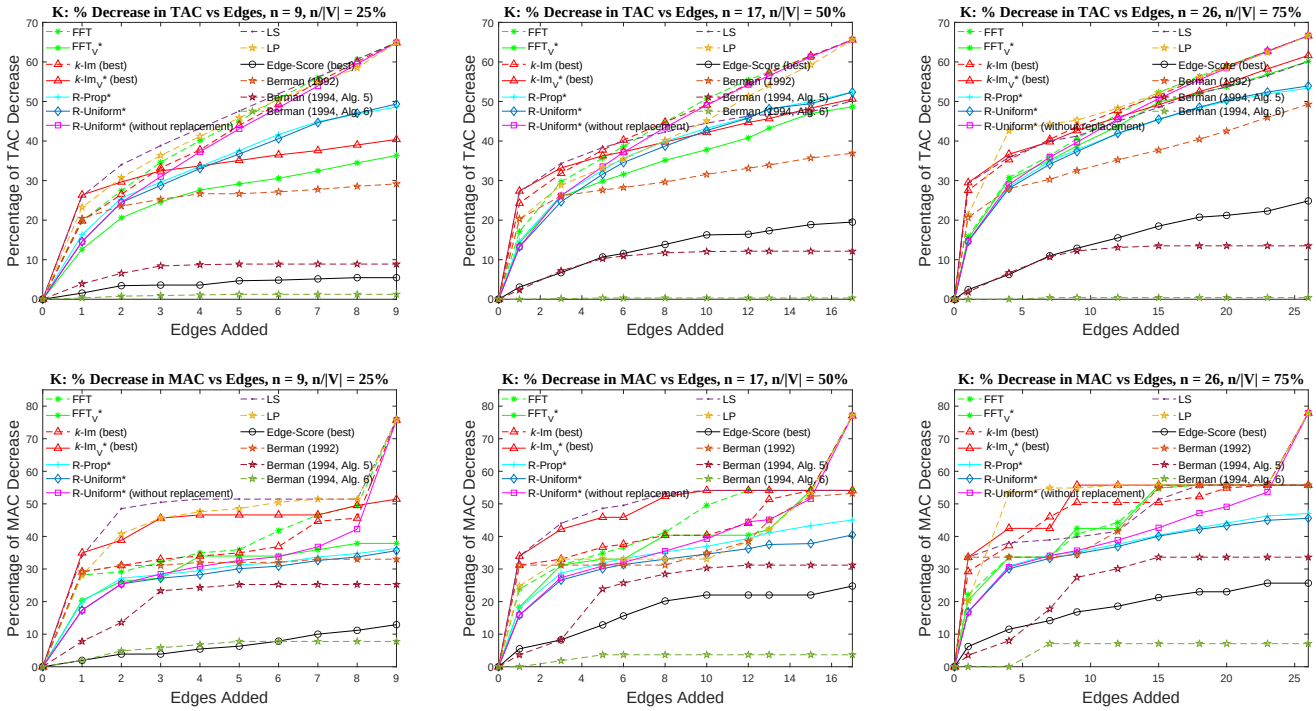

Fig. 36: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *Karate* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

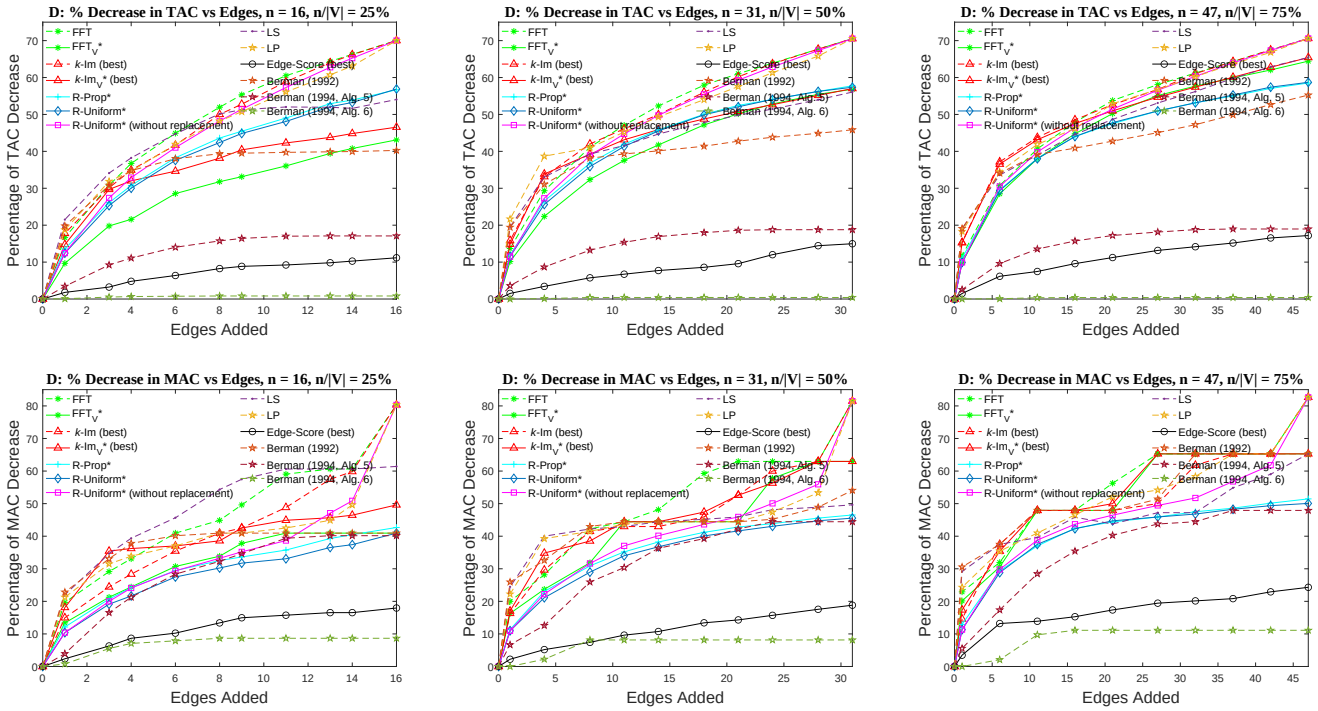

Fig. 37: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *Dolphins* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

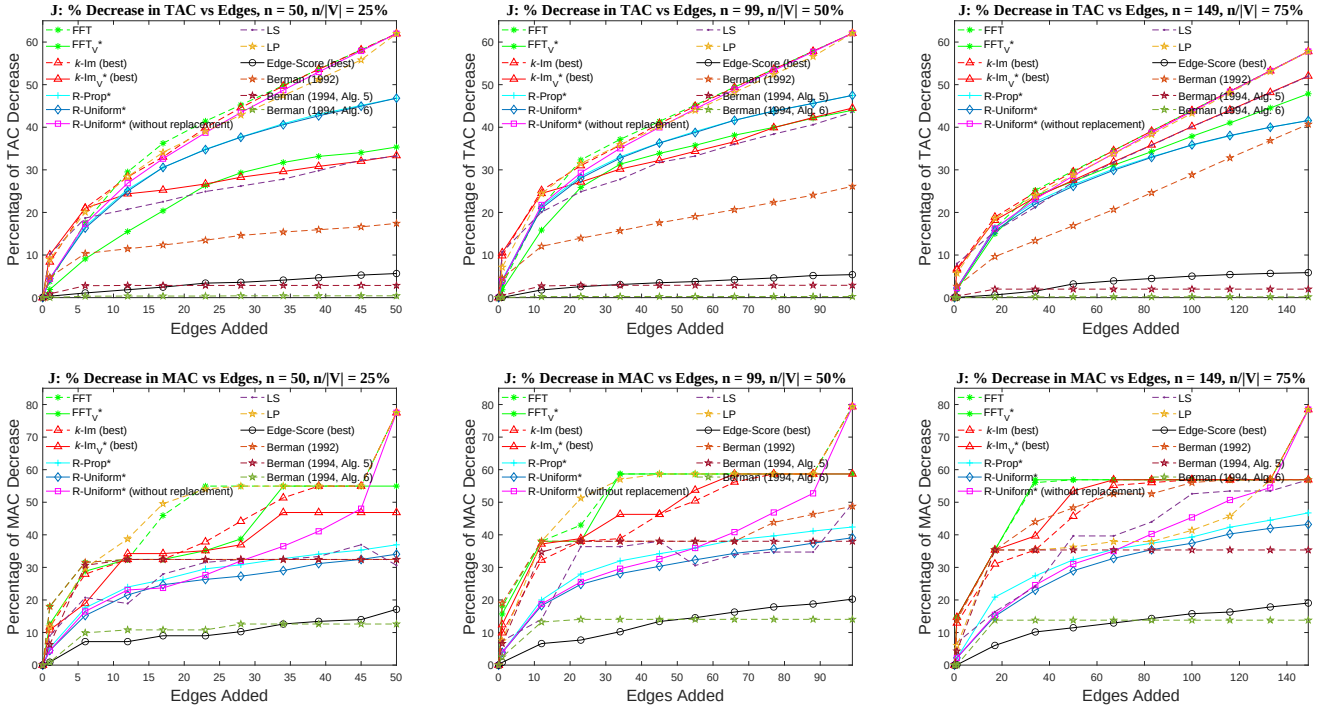

Fig. 38: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *Jazz* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

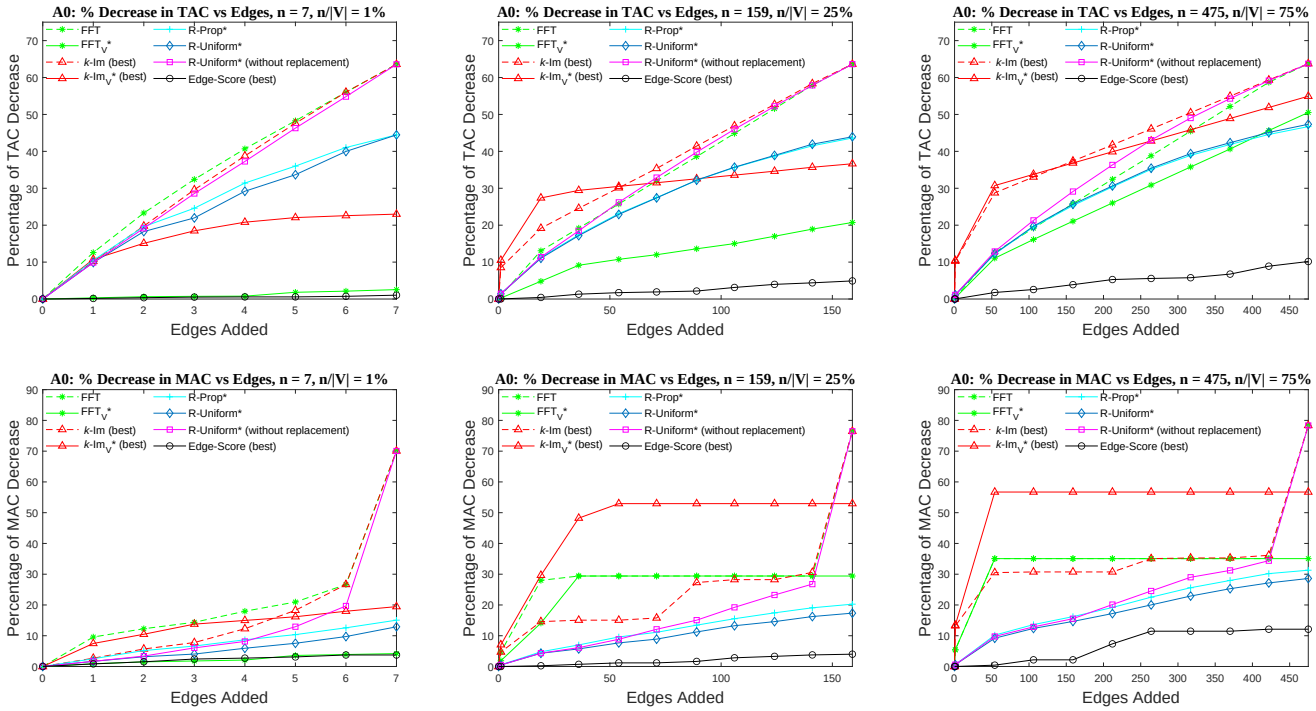

Fig. 39: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *Oregon-A* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

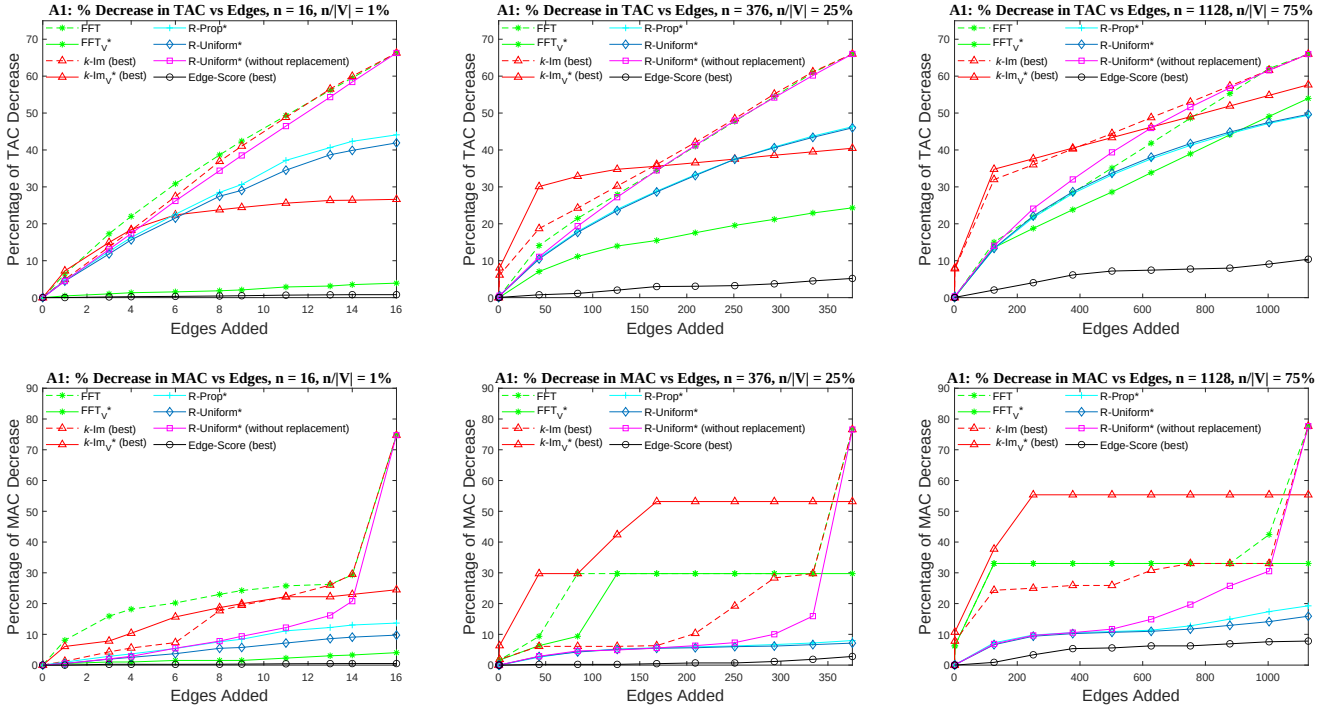

Fig. 40: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *Oregon-B* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

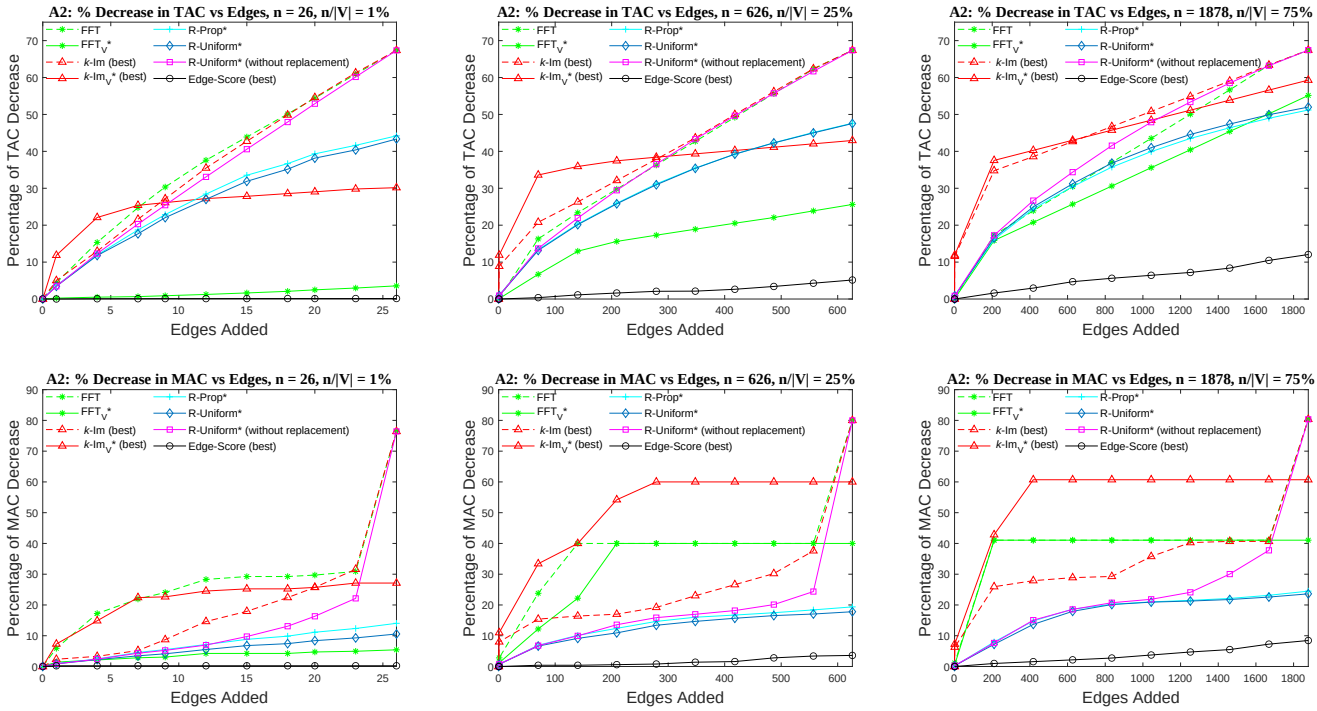

Fig. 41: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *Oregon-C* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

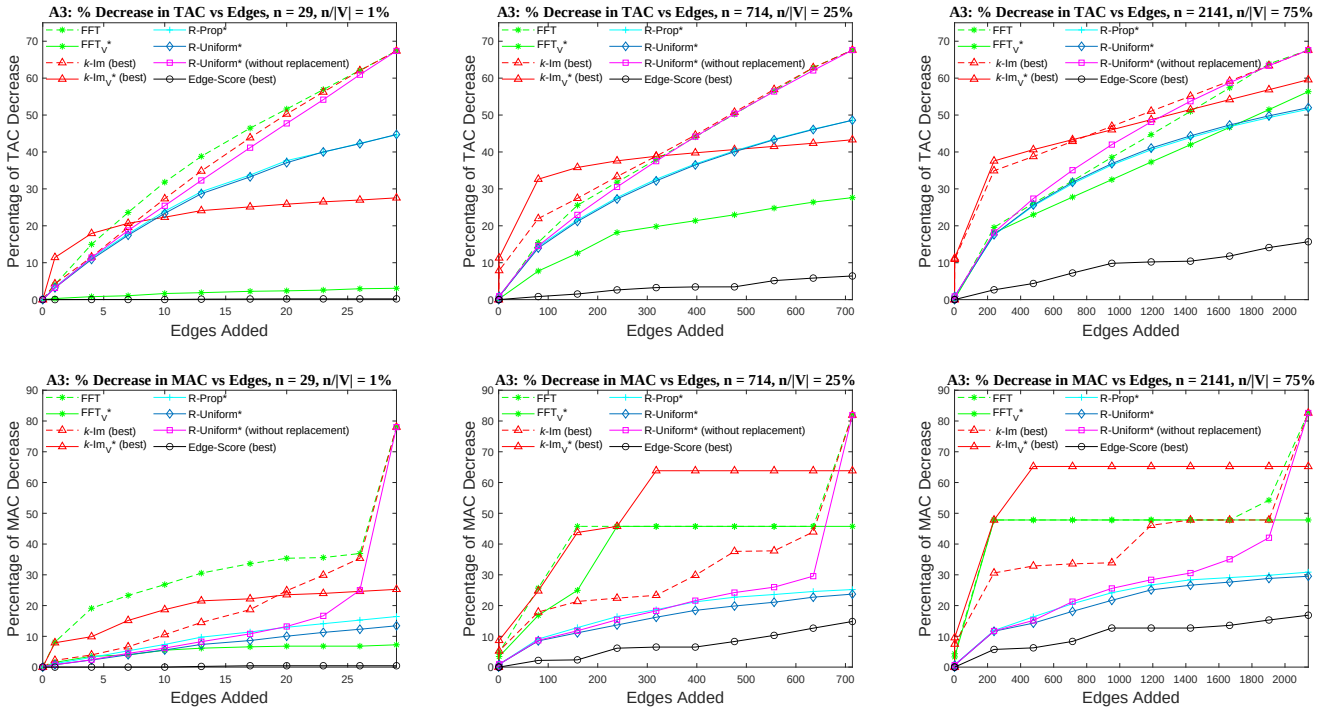

Fig. 42: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *Oregon-D* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

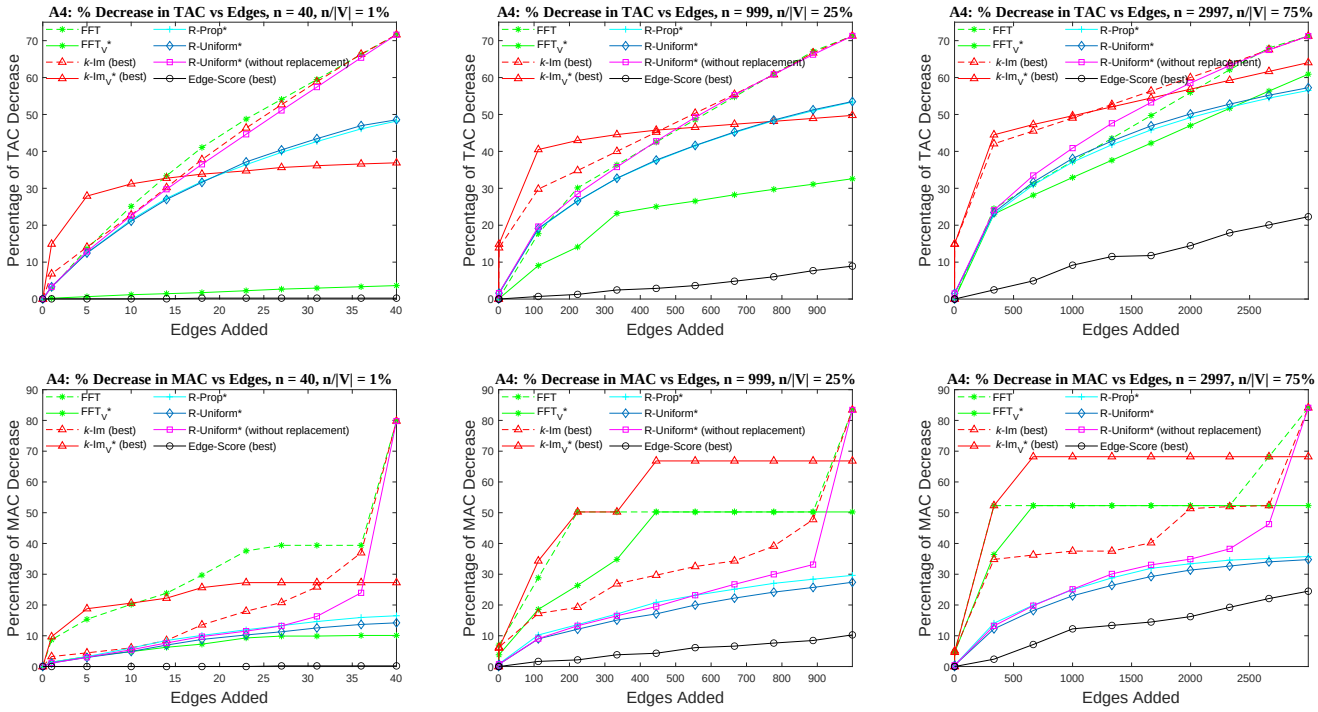

Fig. 43: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *Oregon-E* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

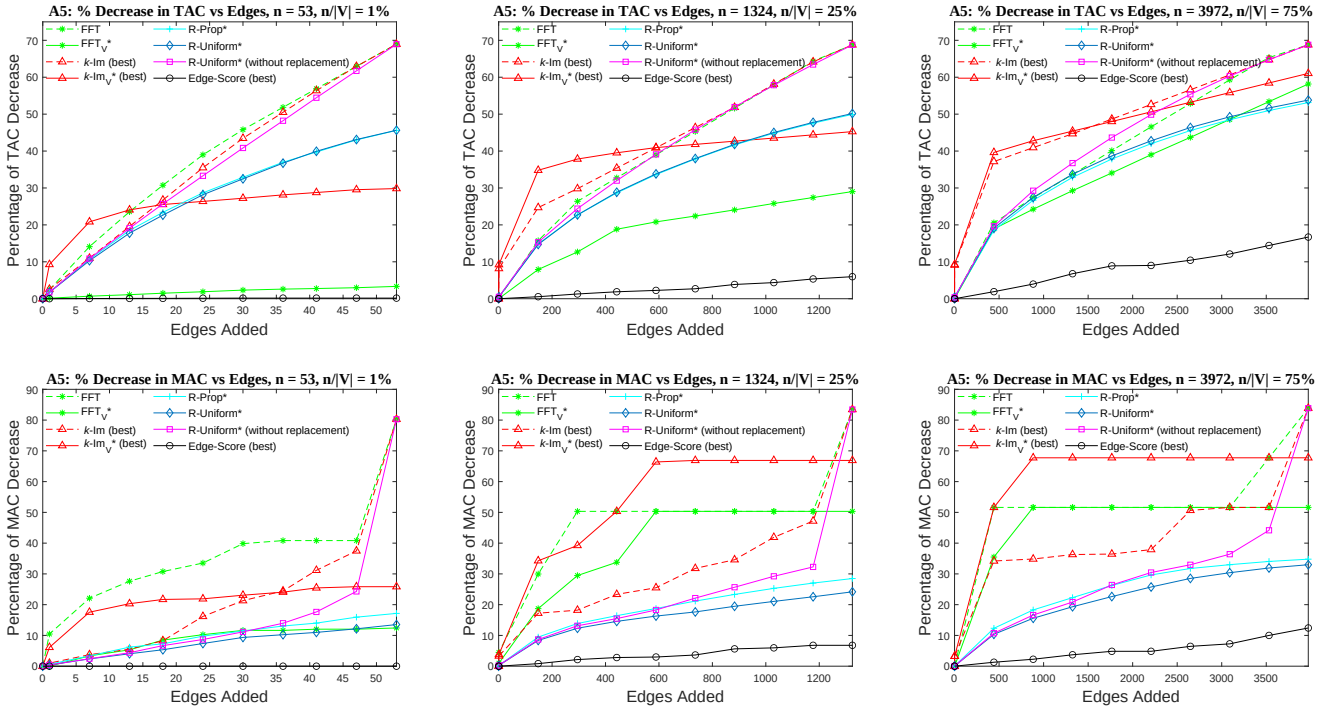

Fig. 44: (Same as Figure 3) Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *Oregon-F* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

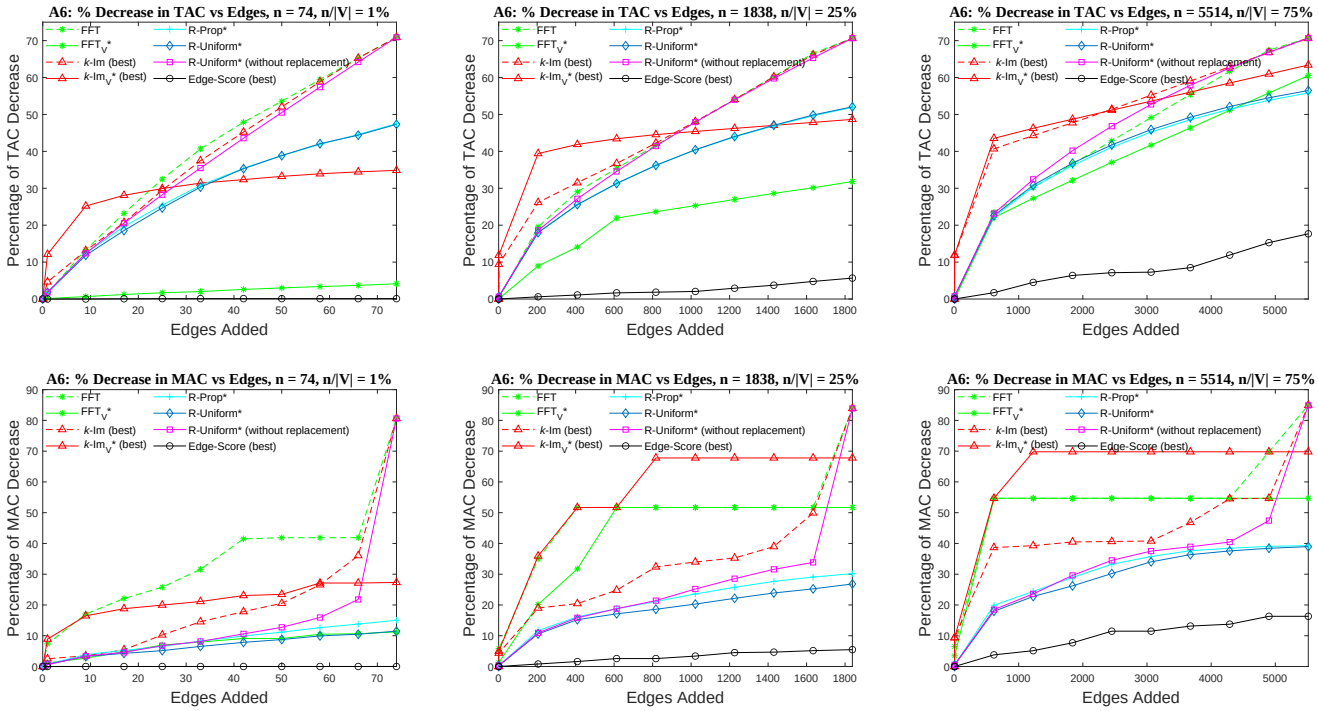

Fig. 45: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *Oregon-G* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

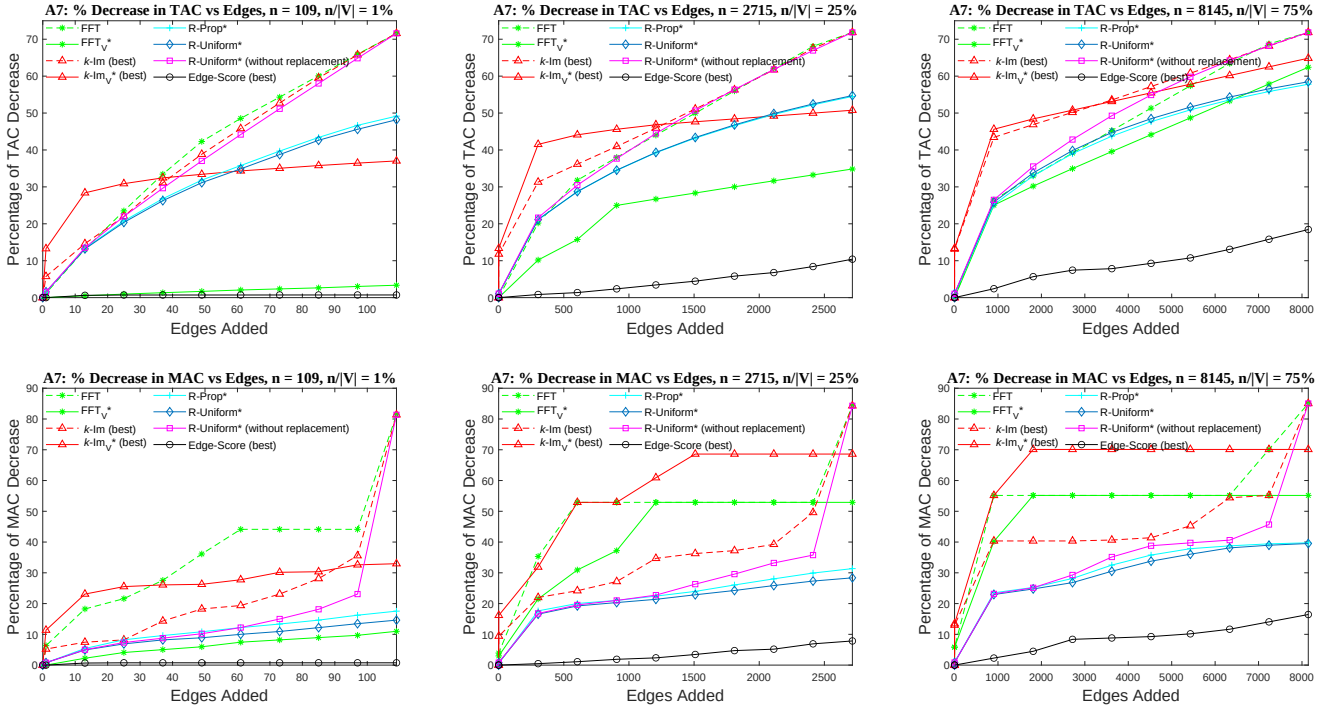

Fig. 46: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *Oregon-H* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

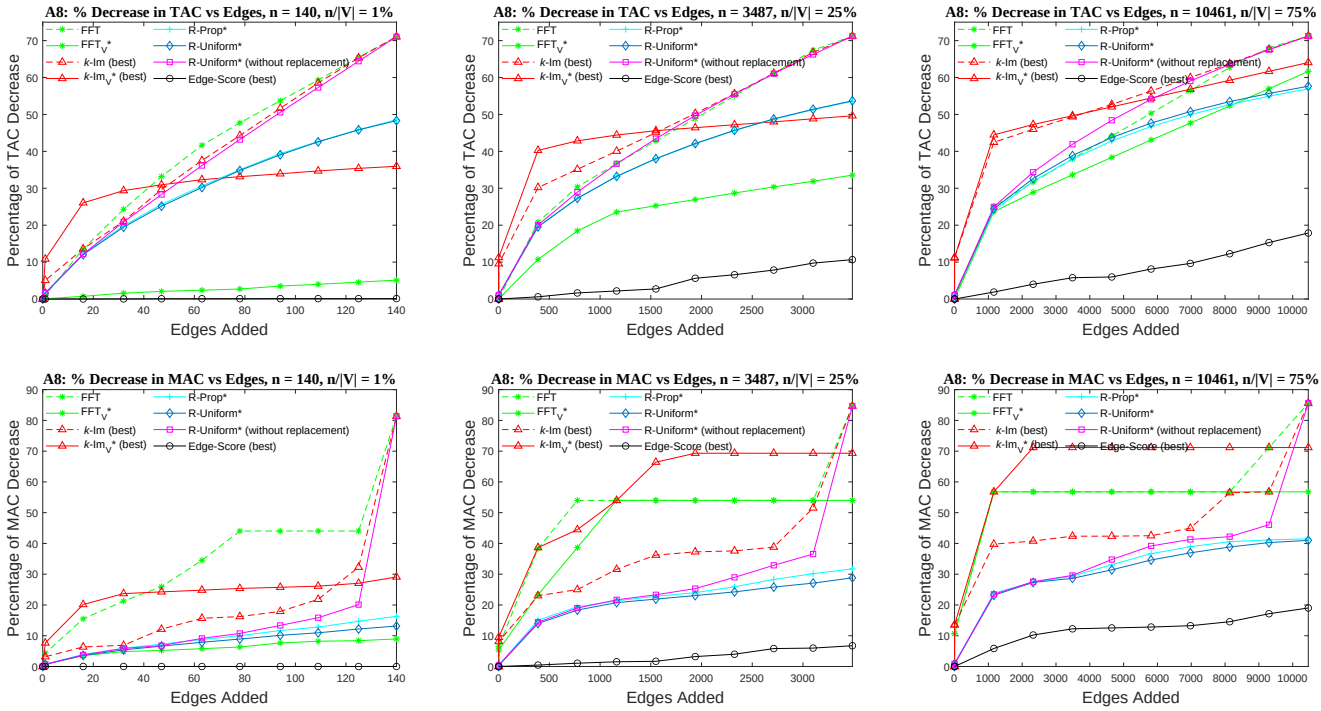

Fig. 47: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *Oregon-I* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

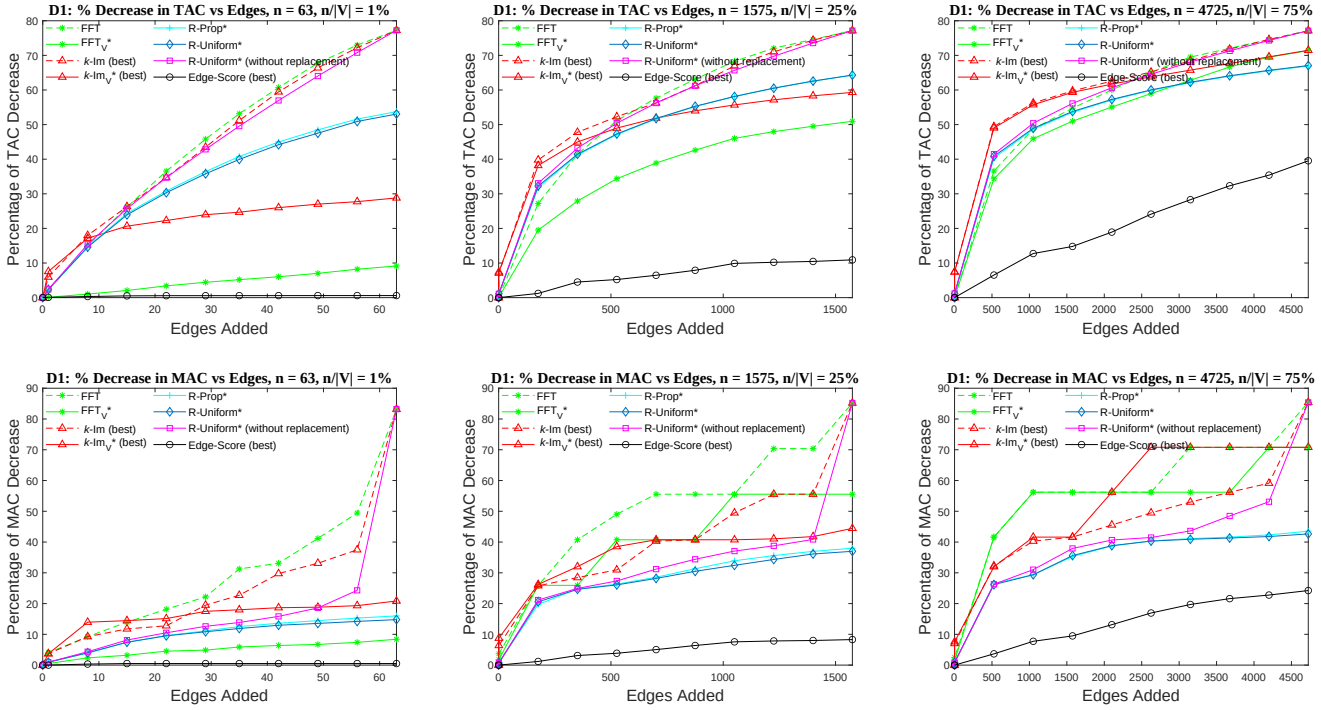

Fig. 48: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *P2P-GnutellaA* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

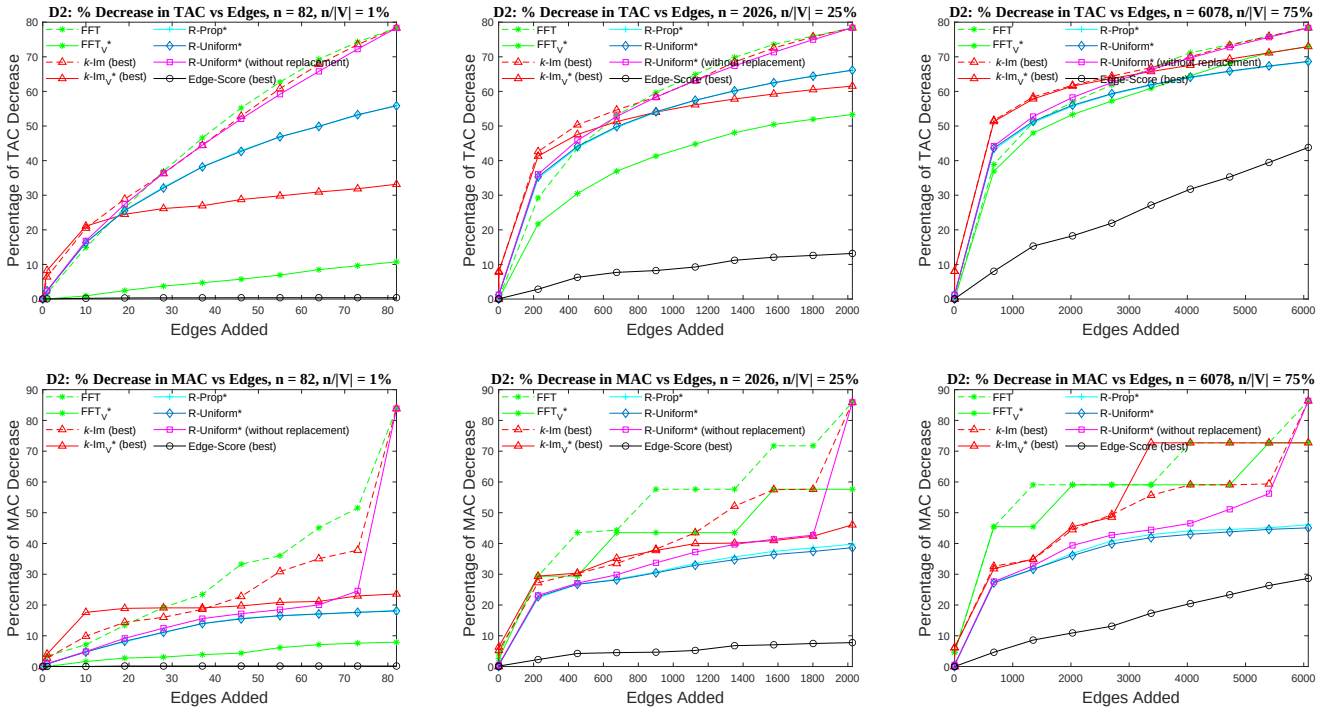

Fig. 49: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *P2P-GnutellaB* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

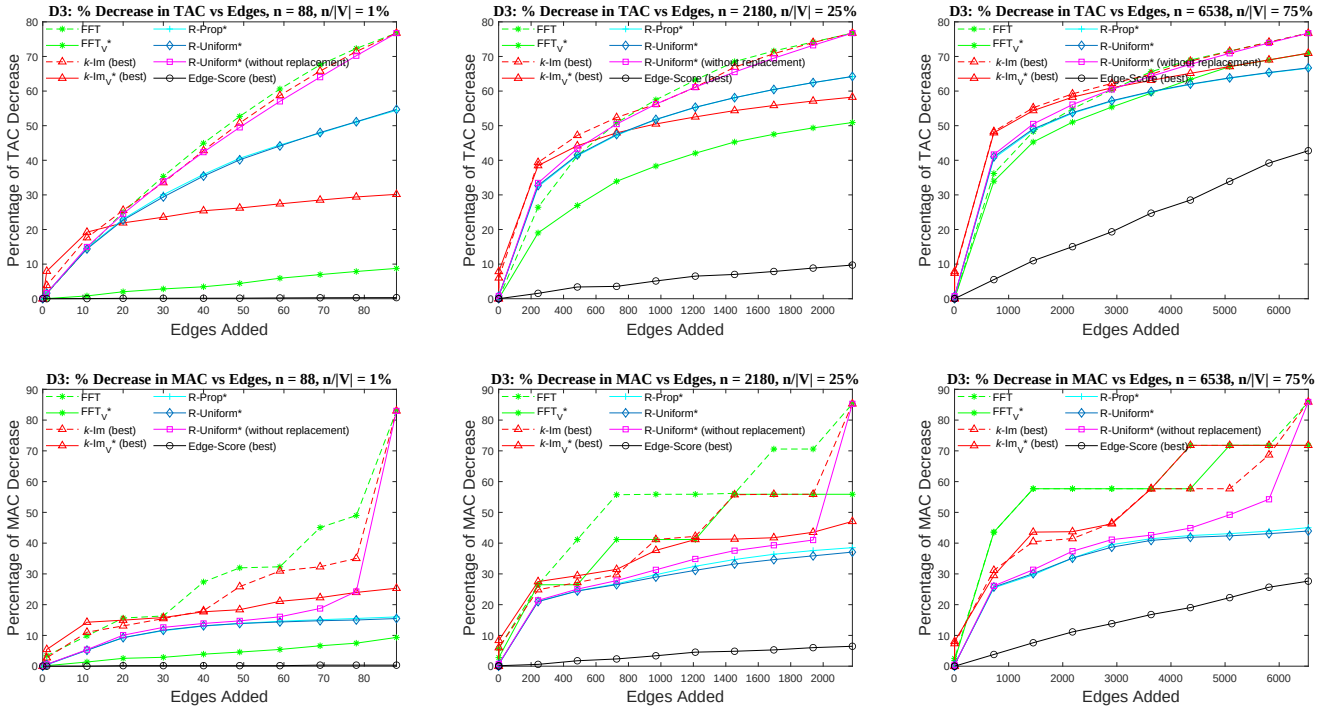

Fig. 50: (Same as Figure 4) Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *P2P-GnutellaC* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

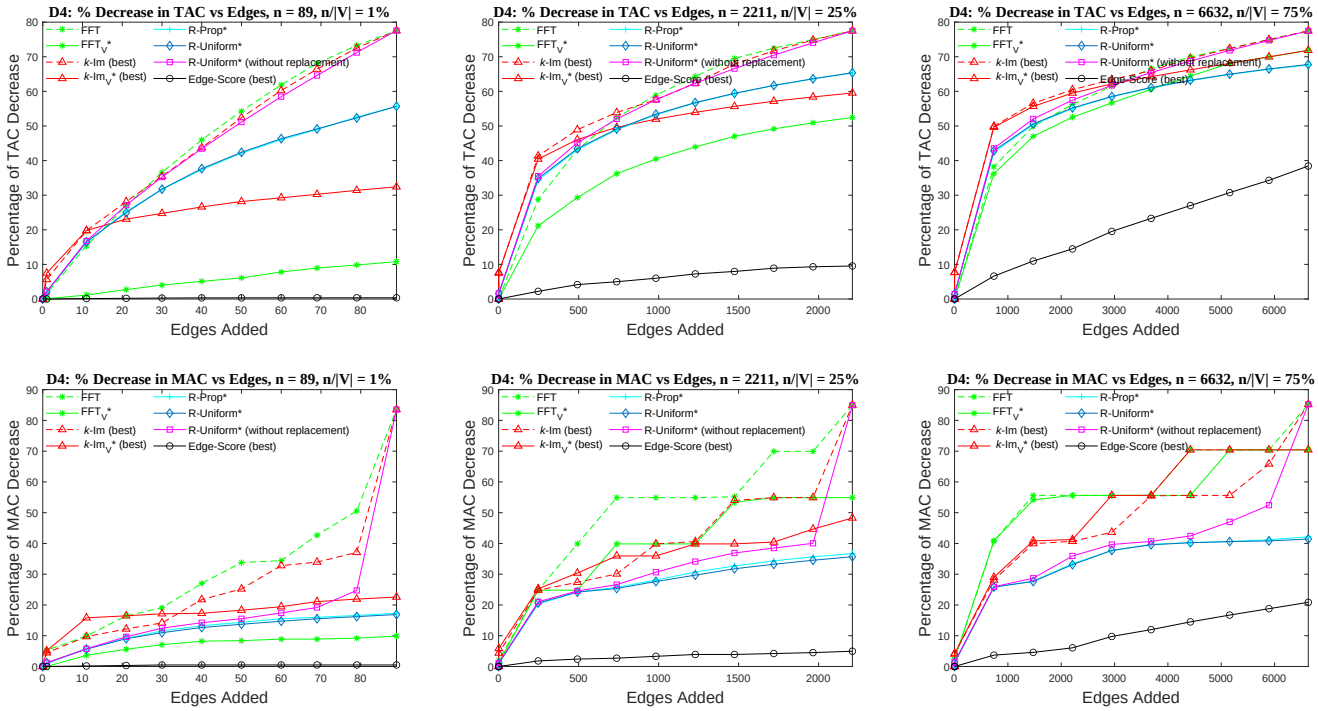

Fig. 51: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *P2P-GnutellaD* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.

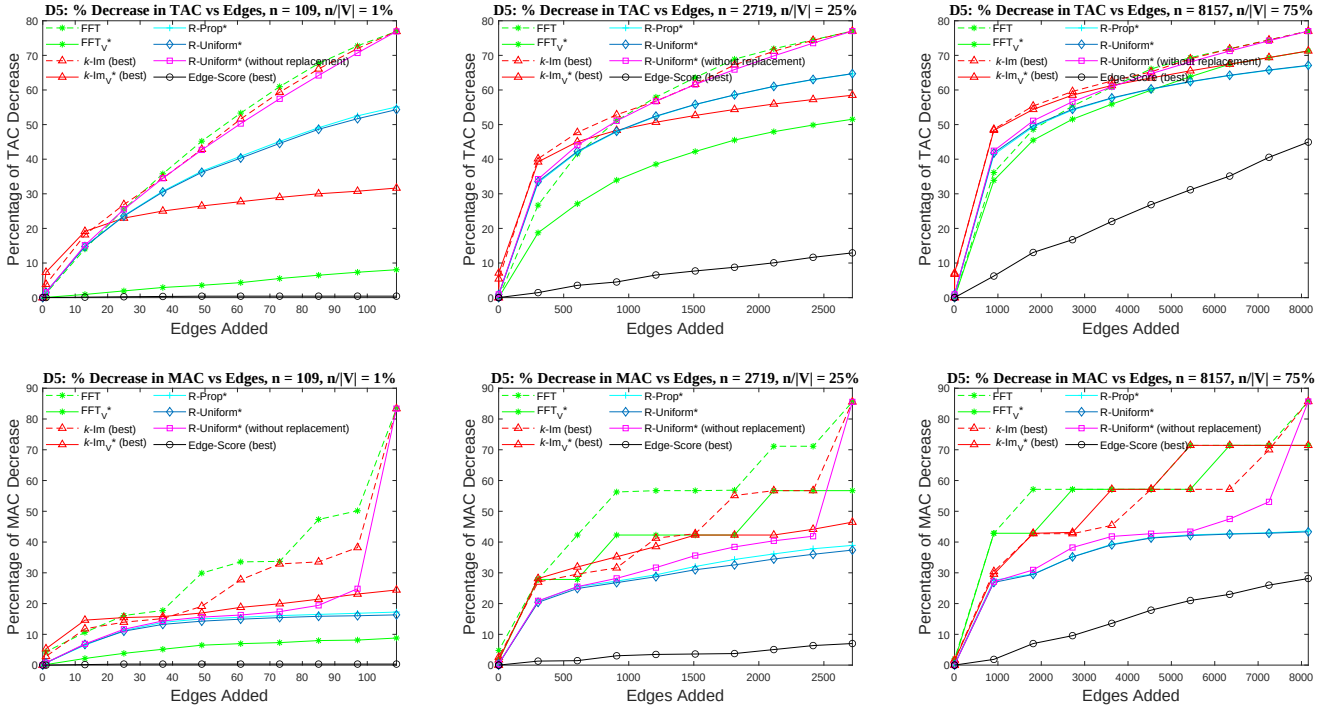

Fig. 52: Percentage of (top) TAC and (bottom) MAC decrease (higher is better) vs. k added edges for proposed algorithms on real-world network *P2P-GnutellaE* with different client sizes, (left) $n = 0.25|V|$, (center) $n = 0.50|V|$, and (right) $n = 0.75|V|$.