Supplementary Materials for
Hiding Individuals and Communities in a Social Network

Marcin Waniek[1,2], Tomasz P. Michalak[1,3*], Michael J. Wooldridge[3], Talal Rahwan[2*]

[1]Institute of Informatics, University of Warsaw, Warsaw, Poland.

[2]Department of Computer Science, Khalifa University of Science and Technology, Abu Dhabi, UAE.

[3]Department of Computer Science, University of Oxford, Oxford, UK.

[*]Corresponding authors. E-mail: tpm@mimuw.edu.pl, trahwan@masdar.ac.ae

This document is structured as follows:

Supplementary Notes

Definitions

Basic Notation: Let $G = (V, E) \in \mathbb{G}$ denote a network, where $V = \{v_1, \ldots, v_n\}$ is the set of n nodes and $E \subseteq V \times V$ is the set of edges, and let $\mathbb{G}(V)$ denote the set of all possible networks whose set of nodes is V. A path is a sequence of distinct nodes, $\langle v_l, \ldots, v_k \rangle$, such that every two consecutive nodes are connected by an edge. The length of a path is considered to be the number of edges in that path. For any pair of nodes, v_i, v_j in G, the set of all shortest paths between them is denoted by $sp_G(v_i, v_j)$, and the distance between them is denoted by $d_G(v_i, v_j)$, where distance is defined as the length of a shortest path between the two. In case of an *undirected* network G we do not discern between edges (v_i, v_j) and (v_j, v_i); otherwise the network is said to be *directed*. Furthermore, G is said to be *connected* (*strongly connected* for directed networks) if there exists a path between every pair of nodes in G.

We denote by $N_G^{pred}(v_i)$ the set of *predecessors* of v_i in G, that is, $N_G^{pred}(v_i) = \{v_j \in V : (v_j, v_i) \in E\}$. On the other hand, we denote by $N_G^{succ}(v_i)$ the set of *successors* of v_i in G, i.e., $N_G^{succ}(v_i) = \{v_j \in V : (v_i, v_j) \in E\}$. Finally, we denote by $N_G(v_i)$ the set of neighbors of v_i in G, i.e., $N_G(v_i) = N_G^{pred}(v_i) \cup N_G^{succ}(v_i)$. For the case of undirected graph, we will assume that $N_G(v_i) = N_G^{pred}(v_i) = N_G^{succ}(v_i)$.

To make the notation more readable, we will often denote two arbitrary nodes by v and w, instead of v_i and v_j. Moreover, we will often omit the network itself from the notation whenever it is clear from the context, e.g., by writing $d(v, w)$ instead of $d_G(v, w)$; this applies not only to the notation presented thus far, but to all notation.

We consider a *community structure*, $CS = \{C_1, \ldots, C_k\}$, to be a partition of the set of nodes into disjoint and exhaustive subsets, or communities.[1] Formally, it satisfies the following three conditions: $\forall_{C_i \in CS} C_i \subseteq V$, $\bigcup_{C_i \in CS} C_i = V$, and $\forall_{C_i, C_j \in CS} C_i \cap C_j = \emptyset$.

Centrality Measures: Formally, a centrality measure [14] is a function $c : \mathbb{G}(V) \times V \to \mathbb{R}$. The *degree* centrality [29] is denoted by c_{degr}, the *closeness* centrality [3] is denoted by c_{clos}, and the *betweeness* centrality [1, 13] is denoted by c_{betw}. Specifically, given a node $v_i \in V$ and an undirected network, we have:

$$c_{degr}(G, v_i) = \frac{|N_G(v_i)|}{n-1}$$

$$c_{clos}(G, v_i) = \frac{n-1}{\sum_{v_j \in V} d_G(v_i, v_j)}$$

$$c_{betw}(G, v_i) = \frac{2}{(n-1)(n-2)} \sum_{v_j, v_k \in V \setminus \{v_i\}} \frac{|\{p \in sp_G(v_j, v_k) : v_i \in p\}|}{|sp_G(v_j, v_k)|}$$

[1] Some works have considered overlapping community structures [33]. However, as common in the literature, we restrict our attention to disjoint communities.

On the other hand, given a directed network, we have:

$$c_{degr}(G, v_i) = \frac{|N_G^{pred}(v_i)|+|N_G^{succ}(v_i)|}{2(n-1)}$$

$$c_{clos}(G, v_i) = \frac{1}{n-1} \sum_{v_j \in V} \frac{1}{d_G(v_i, v_j)}$$

$$c_{betw}(G, v_i) = \frac{1}{(n-1)(n-2)} \sum_{v_j, v_k \in V \setminus \{v_i\}} \frac{|\{p \in sp_G(v_j, v_k) : v_i \in p\}|}{|sp_G(v_j, v_k)|} + \frac{|\{p \in sp_G(v_k, v_j) : v_i \in p\}|}{|sp_G(v_k, v_j)|}$$

Models of Influence: The propagation of influence through the network is often modeled as follows: when a certain node is sufficiently influenced by its neighbor(s), it becomes "active", in which case it starts to influence any "inactive" neighbor(s) it may have, and so on. Of course, to initiate this propagation process, a set of nodes needs to be activated right from the start; this set is called the *seed set*. Assuming that time moves in discrete rounds, we denote by $I(t) \subseteq V$ the set of nodes that are active at round t, implying that $I(1)$ is the seed set. The way influence propagates from the seed set to the remaining nodes depends on the influence model under consideration. Here, the two main models of influence are:

- *Independent Cascade* [17]: In this model, every pair of nodes is assigned an activation probability, $p : V \times V \to [0, 1]$. Then, in every round, $t > 1$, every node $v \in V$ that became active in round $t - 1$ activates every inactive successor, $w \in N^{succ}(v) \setminus I(t - 1)$, with probability $p(v, w)$. The process ends when there are no new active nodes, i.e., when $I(t) = I(t - 1)$.
- *Linear Threshold* [20]: In this model, every node $v \in V$ is assigned a *threshold value* t_v which is sampled (according to some probability distribution) from the set $\{0, \ldots, |N^{pred}(v)|\}$. Then, in every round, $t > 1$, every inactive node v becomes active, i.e., becomes a member of $I(t)$, if: $|I(t - 1) \cap N^{pred}(v)| \geq t_v$. The process ends when $I(t) = I(t - 1)$.

In either model, the influence of a node, v, on another, w, is denoted by $inf_G(v, w)$ and defined as *the probability that w gets activated given the seed set* $\{v\}$ (we make the common assumption that $inf_G(v, v) = 0$ for all $v \in V$). The influence of v over the entire network G is then: $inf_G(v) = \sum_{w \in V} inf_G(v, w)$.

First Objective (Disguising a Node): Roughly speaking, given an "*evader*", $v^\dagger \in V$, and a limited budget, b, specifying the maximum number of edges that are allowed to be added or removed, our goal is to first rewire the network so as to minimize the centrality of $v^\dagger$, and then to further rewire the network so as to "recover" the influence of $v^\dagger$ (in an attempt to compensate for any influence that $v^\dagger$ might have lost during the centrality-minimization phase). We

consider two variants of the influence-recovery problem; the first focuses on the influence of $v^\dagger$ over every single node, whereas the second focuses on the influence of $v^\dagger$ over the network as a whole. In both cases, only the addition of edges is considered, since the removal of edges can only decrease the influence of $v^\dagger$. Next, we formally define the aforementioned problems.

Definition 1 (Disguising Centrality) *This problem is defined by a tuple, $(G, v^\dagger, b, c, \hat{R}, \hat{A})$, where $G = (V, E) \in \mathbb{G}$ is a network, $v^\dagger \in V$ is the evader (whose centrality is to be minimized), $b \in \mathbb{N}$ is a budget specifying the maximum number of edges that can be added or removed, $c : \mathbb{G} \times V \to \mathbb{R}$ is a centrality measure, $\hat{R} \subseteq E$ is a set of edges whose removal is forbidden, $\hat{A} \subseteq (V \times V) \setminus E$ is a set of edges whose addition is forbidden. The goal is then to identify two sets of edges, $R^* \subseteq (E \setminus \hat{R})$ and $A^* \subseteq (V \times V) \setminus (E \cup \hat{A})$, such that: $|A^*|+|R^*| \leq b$ and $G^* = (V, (E \cup A^*) \setminus R^*)$ is connected (strongly connected if G is directed) and G^* is in:*

$$\underset{G' \in \left\{\left(V,(E\cup A)\setminus R\right): R\subseteq(E\setminus\hat{R}), A\subseteq(V\times V)\setminus(E\cup\hat{A})\right\}}{\arg\min} c(G', v^\dagger).$$

Definition 2 (Individual Influence recovery) *This problem is defined by a tuple, $(G, v^\dagger, inf, \hat{A}, f)$, where $G = (V, E) \in \mathbb{G}$ is a network, $v^\dagger \in V$ is the evader (whose influence is to be recovered), $inf : \mathbb{G} \times V \to \mathbb{R}$ is an influence measure, $\hat{A} \subseteq (V \times V) \setminus E$ is a set of edges whose addition is forbidden, and $f : V \to \mathbb{R}$ specifies the influences to be recovered (i.e., for every $v_i \in V$ we want the influence of $v^\dagger$ over v_i to be at least $f(v_i)$). The goal is then to identify a set of edges, A^*, that is in:*

$$\underset{A\subseteq(V\times V)\setminus(E\cup\hat{A}):\forall_{v_i\in V}\, inf_{(V,E\cup A)}(v^\dagger,v_i)\geq f(v_i)}{\arg\min} |A|.$$

Definition 3 (Global Influence recovery) *This problem is defined by a tuple, $(G, v^\dagger, inf, \hat{A}, \phi)$, where $G = (V, E) \in \mathbb{G}$ is a network, $v^\dagger \in V$ is the evader (whose influence is to be recovered), $inf : \mathbb{G} \times V \to \mathbb{R}$ is an influence measure, $\hat{A} \subseteq (V \times V) \setminus E$ is a set of edges whose addition is forbidden, and $\phi \in \mathbb{R}$ is the total influence to be recovered. The goal is then to identify a set of edges, A^*, that is in:*

$$\underset{A\subseteq(V\times V)\setminus(E\cup\hat{A}):inf_{(V,E\cup A)}(v^\dagger)\geq\phi}{\arg\min} |A|.$$

Second Objective (Disguising a Community): Roughly speaking, given a group of "*evaders*", $V^\dagger$, who wish to hide from community detection algorithms, and given a limited budget, b, specifying the maximum number of edges that are allowed to be added or removed, our goal is to rewire the network so as to hide $V^\dagger$. To this end, we propose a measure of concealment, μ, defined for every group of evaders, $V^\dagger \subseteq V$, and every community structure, CS, as follows:[2]

$$\mu(V^\dagger, CS) = \alpha\mu'(V^\dagger, CS) + (1 - \alpha)\mu''(V^\dagger, CS),$$

[2]When $V^\dagger$ is described as a "community" that wishes to hide from community detection algorithms, the term "community" is used here in its broader sense, and is not meant to imply that $V^\dagger$ is a member of CS. In fact, when measuring how well $V^\dagger$ is hidden in CS, it may well be the case that the members of $V^\dagger$ are spread out across multiple communities in CS.

where $\alpha \in [0, 1]$ and:

$$\mu'(V^\dagger, CS) = \frac{|\{C_i \in CS : C_i \cap V^\dagger \neq \emptyset\}| - 1}{\max(|CS| - 1, 1) \max_{C_i \in CS}(|C_i \cap V^\dagger|)}$$

$$\mu''(V^\dagger, CS) = \sum_{C_i \in CS} \frac{|C_i \setminus V^\dagger|}{\max(n - |V^\dagger|, 1)}.$$

Note that $\mu(V^\dagger, CS) \in [0, 1]$ for all $V^\dagger$ and CS, with greater values indicating greater levels of concealment of $V^\dagger$ in CS. Having presented our concealment measure, we are now ready to formally introduce our problem.

Definition 4 (Disguising a Community) *This problem is defined by a tuple, $(G, V^\dagger, alg, b)$, where $G = (V, E)$ is a network, $V^\dagger \subseteq V$ is the group of evaders, alg is a community-detection algorithm, and $b \in \mathbb{N}$ is a budget specifying the maximum number of edges that can be added or removed. The goal is then to find a set of edges to be added, $A^* \subseteq (V \times V) \setminus E$, and another to be removed, $R^* \subseteq E$, such that $|A^*| + |R^*| \leq b$ and $G^* = (V, (E \cup A^*) \setminus R^*)$ is in:*

$$\underset{G' \in \left\{\left(V, (E \cup A) \setminus R\right) \,:\, A \subseteq (V \times V) \setminus E,\ R \subseteq E,\ |A| + |R| \leq b\right\}}{\arg\max} \mu(V^\dagger, alg(G')),$$

where $alg(G)$ is the community structure returned by the algorithm alg given the network G.

Note that the above optimization problem requires $V^\dagger$ to know the exact community-detection algorithm that the seeker is using. Since such knowledge is hardly available, we avoid this requirement, and instead aim to develop a general-purpose heuristic, designed for no particular community-detection algorithm.

Supplementary Materials

Theoretical Results

From the computational point of view, disguising the degree centrality of $v^\dagger$ is easy, since the only way to decrease this centrality is to remove edges connecting $v^\dagger$ to its neighbor(s). Next, we study the problems of disguising closeness centrality and betweenness centrality, followed by the problem of influence recovery under the Independent-Cascade model and under the Linear-Threshold model.

Theorem 1 *Disguising closeness centrality is NP-complete.*

Proof. The decision version of the optimization problem is the following: given a network $G = (V, E)$, an evader $v^\dagger$, two sets $\hat{R} \subseteq E$, $\hat{A} \subseteq (V \times V) \setminus E$, a budget $b \in \mathbb{N}$ and a value $x \in \mathbb{R}$, does there exist two sets $R^* \subseteq (E \setminus \hat{R})$ and $A^* \subseteq (V \times V) \setminus (E \cup \hat{A})$ such that $|A^*|+|R^*| \leq b$, and the network $(V, (E \cup A^*) \setminus R^*)$ is connected (strongly connected if G is directed) and $c_{clos}((V, (E \cup A^*) \setminus R^*), v^\dagger) \leq x$?

This problem is in NP, as given a solution, i.e., two sets A^* and R^*, we can verify whether $c_{clos}((V, (E \cup A^*) \setminus R^*), v^\dagger) \leq x$ in polynomial time; this only requires computing the closeness centrality of node $v^\dagger$ in network $(V, (E \cup A^*) \setminus R^*)$.

We will now show that the decision version is NP-hard. To this end, let us denote by $q \in \mathbb{R}$ the smallest possible closeness centrality of $v^\dagger$ in any (strongly) connected network whose set of nodes is V. One can see that $q = 2/n$ in the case of undirected networks, and $q = (\sum_{i=1}^{n-1} \frac{1}{i})/(n-1)$ in the case of directed network; this happens if and only if:

- the network is a path of which $v^\dagger$ is an end (when dealing with undirected networks); or
- the network is a directed cycle (when dealing with directed networks).

Let us denote such a network by Q; the closeness centrality of $v^\dagger$ in Q is then q. With this in mind, the proof involves a reduction from the *Hamiltonian cycle* problem (i.e., the problem of determining whether there exists a cycle that visits each node exactly once) to the decision problem of determining whether it is possible to reduce the closeness centrality of $v^\dagger$ to a value smaller than, or equal to, q.

To this end, given some arbitrary network, $G' = (V', E')$, be it directed or undirected, let us modify G' so as to obtain a new network, $G = (V, E)$, as illustrated in Figures 1 and 2.

Formally, we do so by choosing some arbitrary node, $w \in V'$, and then setting:

$$V = V' \cup \{v^\dagger, v_1, v_2\}, \quad E = E' \cup \{(v^\dagger, w), (v_1, v_2)\} \cup \{(v, v_1) : v \in V', v \in N_{G'}(w)\},$$

in the case of undirected networks, or setting:

$$V = V' \cup \{v^\dagger, v_1\}, \quad E = E' \cup \{(v^\dagger, w), (v_1, v^\dagger)\} \cup \{(v, v_1) : v \in V', v \in N^{pred}_{G'}(w)\},$$

in the case of directed networks.

We will now show that the Hamiltonian cycle problem in G' is equivalent to the following decision problem: Given network G and budget $b = |E'|-|V'|+|N_{G'}^{pred}(w)|$, where $\hat{A} = \hat{R} = \emptyset$, determine whether it is possible to reduce the closeness centrality of $v^\dagger$ to a value $\leq q$, by removing at most b edges from G. Throughout the remainder of the proof, the edges and nodes in G that were in G' will be referred to as "*original*".

Firstly, we will show that if G' has a Hamiltonian cycle then it is possible to obtain Q by removing $|E'|-|V'|+|N_{G'}^{pred}(w)|$ edges from G. To this end, fix a Hamiltonian cycle of G', then:

- remove from G all original edges that are not in the Hamiltonian cycle; there are exactly $|E'|-|V'|$ such edges;
- in the Hamiltonian cycle, there are exactly two edges of which w is an end; remove any of those edges in the undirected network, or the one pointing to w in the directed network; let us denote the removed edge as (v', w);
- remove all edges from all predecessors of w to v_1, with the exception of (v', v_1); there are exactly $|N_{G'}^{pred}(w)|-1$ such edges.

In so doing, we have obtained the network Q by removing a total of $|E'|-|V'|+|N_{G'}^{pred}(w)|$ edges from G (see Figures 1 and 2).

Secondly, we show that if it is possible to obtain Q by removing $|E'|-|V'|+|N_{G'}^{pred}(w)|$ edges from G, then there exists a Hamiltonian cycle in G'. We will first deal with the undirected case, before moving on to the directed case.

In the undirected case, observe that nodes $v^\dagger$ and v_2 each have a degree of 1 in G, since their only neighbors are w and v_1, respectively. Now since Q is connected, and since we obtained Q by only removing (rather than adding) edges from G, the nodes $v^\dagger$ and v_2 must each have a degree of 1 in Q. Consequently, they must be the two ends of Q. This, in turn, implies that v_1 must have exactly two neighbors in Q; we know that one of them is v_2, let us call the other v'. This, as well as the fact that $v^\dagger$ is only connected to w, implies that the segment of Q between w and v' contains all original nodes from G' and only original edges from G' (recall that we did not add any edges between original nodes). Finally, by adding to that segment the original edge between v' and w, we obtain a Hamiltonian cycle in G'.

As for the directed case, we observe that node $v^\dagger$ has only one successor in Q, namely w, and only one predecessor in Q, namely v_1. We also know that v_1 has only one predecessor in Q; let us call that predecessor v'. These facts imply that the segment of Q between w and v' contains all original nodes from G' and only original edges from G' (again, recall that we did not add any edges between original nodes). By adding to that segment the original edge between v' and w, we obtain a Hamiltonian cycle in G'.

We have shown that a Hamiltonian cycle in G' exists if and only if it is possible to reduce the closeness centrality of $v^\dagger$ to q by removing exactly $|E'|-|V'|+|N_{G'}^{pred}(w)|$ edges from G, which concludes the proof. □

Theorem 2 *Disguising betweenness centrality is NP-complete.*

Proof. The decision version of the optimization problem is the following: given a network $G = (V, E)$, an evader $v^\dagger$, two sets $\hat{R} \subseteq E$, $\hat{A} \subseteq (V \times V) \setminus E$, a budget $b \in \mathbb{N}$ and a value $x \in \mathbb{R}$, does there exist two sets $R^* \subseteq (E \setminus \hat{R})$ and $A^* \subseteq (V \times V) \setminus (E \cup \hat{A})$ such that $|A^*|+|R^*| \leq b$, and the network $(V, (E \cup A^*) \setminus R^*)$ is connected (strongly connected if G is directed) and $c_{betw}((V, (E \cup A^*) \setminus R^*), v^\dagger) \leq x$?

This problem is in NP, as given a solution, i.e., two sets A^* and R^*, we can verify whether $c_{betw}((V, (E \cup A^*) \setminus R^*), v^\dagger) \leq x$ in polynomial time; this only requires computing the betweenness centrality of node $v^\dagger$ in network $(V, (E \cup A^*) \setminus R^*)$.

We will now show that the decision version is NP-hard. To this end, we propose a reduction from the NP-complete *Set cover* problem. The decision version of this problem is defined by a universe $U = \{u_1, \ldots, u_l\}$ and a collection of sets $S = \{S_1, \ldots, S_m\}$ such that $\forall_j S_j \subset U$, where the goal is to determine whether there exist $k \leq m$ elements of S the union of which equals U.

First, let us create a network G as shown in Figures 3 and 4. More specifically, we create one node for every $S_j \in S$, one node for every $u_i \in U$, and three additional nodes, $v^\dagger$, v_0 and v_1. Next, we add (either undirected or directed) edges as follows. We add the edges $(v^\dagger, v_0)$ and $(v_1, v^\dagger)$, and for every node $u_i \in S_j$ we add the edges (S_j, u_i) and (u_i, v_1). In case of a directed network, we also add the edges (u_i, S_j) and (v_1, u_i) for every $u_i \in S_j$, as well as the edge (v_0, v_1).

Now, consider the problem of disguising the betweenness centrality of $v^\dagger$ in G given $\hat{R} = E$ and $\hat{A} = (V \times V) \setminus \{(S_1, v_0), \ldots, (S_m, v_0)\}$. Note that $v^\dagger$ "controls" (i.e., lies on) every shortest path to v_0, and does not control any shortest path between any other pair of nodes. As such, to minimize the betweenness centrality of $v^\dagger$, we need to create alternative shortest paths to v_0; this should be done by adding (some of) the edges in $\{(S_1, v_0), \ldots, (S_m, v_0)\}$, since no other edge can be added, and no edge can be removed (following the definitions of $\hat{R}$ and $\hat{A}$). To be more precise, we can add at most b edges $\{(S_1, v_0), \ldots, (S_m, v_0)\}$, since we cannot exceed the budget. After this process, the betweenness centrality of $v^\dagger$ may drop to as little as $q = \frac{2}{(n-1)(n-2)}$ in the undirected case, or as little as $q = \frac{1}{(n-1)(n-2)}$ in the directed case; this happens when $v^\dagger$ no longer controls any of the shortest paths to v_0 except for the one from v_1 to v_0. Note that adding an edge (S_j, v_0) creates a new shortest path from every nodes $u_i \in S_j$ to v_0. This implies that the betweenness centrality of $v^\dagger$ can be reduced to q if and only if there exists at most b elements of S the union of which equals U.

We have just reduced the decision version of the Set Cover problem given k to the following decision problem: Given network G and budget $b = k$, where $\hat{R} = E$ and $\hat{A} = (V \times V) \setminus \{(S_1, v_0), \ldots, (S_m, v_0)\}$, determine whether it is possible to reduce the closeness centrality of $v^\dagger$ to some value $\leq q$, by removing at most b edges from G. □

Theorem 3 *Both the global and the individual influence recovery problems are NP-hard under the Independent Cascade model.*

Proof. We show a reduction from the NP-complete *Set cover* problem, defined by a universe $U = \{u_1, \ldots, u_l\}$ and a collection of sets $S = \{S_1, \ldots, S_m\}$ such that $S_1 \cup \ldots \cup S_m = U$ and $\forall_j S_j \subseteq U$, and the goal is to determine whether there exist $k \leq m$ elements of S the union of which equals U.

To this end, let us create a network G as illustrated in Figures 5 and 6. In more detail, we start by creating one node for every $S_j \in S$, one node for every $u_i \in U$, and one additional node $v^\dagger$. After that, for every $S_j \in S$ and every $u_i \in S_j$, we add the edge (S_j, u_i) (either directed or undirected). In the directed case we additionally add an edge $(u_i, v^\dagger)$ for every $u_i \in U$.

Consider the influence recovery problem in G under the Independent Cascade model, where:

- $\hat{A} = (V \times V) \setminus \{(v^\dagger, S_1), \ldots, (v^\dagger, S_m)\}$;
- $p : V \times V \to [0, 1]$ such that $\forall_{S_j \in S}\ p(v^\dagger, S_j) = 1$ and $\forall_{S_j \in S} \forall_{u_i \in S_j}\ p(S_j, u_i) = 1$, and $p(v, w) = 0$ for every other pair of nodes;
- for *individual* influence recovery, $\forall_{u_i \in U} f(u_i) = 1$ and $f(v) = 0$ for every other node;
- for *global* influence recovery, $\phi = k + l$.

The goal is then to identify the smallest subset of edges to be added to the network, $A \subseteq \{(v^\dagger, S_1), \ldots, (v^\dagger, S_m)\}$, such that either $inf_{(V, E \cup A)}(v^\dagger) \geq \phi$ in the *global* variant of the problem, or $\forall_{v_i \in V} inf_{(V, E \cup A)}(v^\dagger, v_i) \geq f(v_i)$ in the *individual* variant of the problem.

Recall that the influence of $v^\dagger$ is measured by setting the seed set as $\{v^\dagger\}$ and calculating the probability that other nodes get activated. Also recall that under the Independent Cascade model an active node, v, activates any of its predecessors, w, with probability $p(v, w)$. Importantly, with the p function defined as above, adding an edge $(v^\dagger, S_j)$ for some $S_i \in S$ makes the influence of $v^\dagger$ on every $u_i \in S_j$ equal to 1. Furthermore, the above definitions of ϕ and f imply that our goal (in both the individual and the global variants of the problem) is achieved if and only if the influence of $v^\dagger$ on *every node* $u_i \in U$ equals 1. Consequently, our goal is achieved if and only if we add to G a set of edges, $A \subseteq \{(v^\dagger, S_1), \ldots, (v^\dagger, S_m)\}$, such that:

$$\bigcup_{(v^\dagger, S_j) \in A} S_j = U.$$

Since we are interested in finding the smallest such subset, a solution to the above instance of the influence recovery problem gives us a solution to the Set Cover problem. □

Theorem 4 *Both the global and the individual influence recovery problems are NP-hard under the Linear Threshold model.*

Proof. We show a reduction from the NP-complete *Set cover* problem, defined by a universe $U = \{u_1, \ldots, u_l\}$ and a collection of sets $S = \{S_1, \ldots, S_m\}$ such that $S_1 \cup \ldots \cup S_m = U$ and

$\forall_j S_j \subseteq U$, and the goal is to determine whether there exist $k \leq m$ elements of S the union of which equals U.

For the directed case, we create a network G as illustrated earlier in Figure 6. As for the undirected case, we create G as illustrated in Figure 7. In more detail, for every $S_j \in S$, we create two nodes, namely S_j and T_j, as well as l additional nodes, namely $s_{j,1}, \ldots, s_{j,l}$. We also create one node for every $u_i \in U$, and finally add the evader, $v^\dagger$. As for the edges, for every $S_j \in S$ and every $u_i \in S_j$, we add the edge (T_j, u_i). Furthermore, for every node $s_{j,i}$, we add the edges $(S_j, s_{j,i})$ and $(s_{j,i}, T_j)$.

Now consider the influence recovery problem in G under the Linear Threshold model, where:

- $\hat{A} = (V \times V) \setminus \{(v^\dagger, S_1), \ldots, (v^\dagger, S_m)\}$;
- $t_v = l$ for every node $v \in \{T_1, \ldots, T_m\}$ and $t_v = 1$ for every other node;
- for *individual* influence recovery, $\forall_{u_i \in U} f(u_i) = 1$ and $f(v) = 0$ for every other node;
- for *global* influence recovery, $\phi = k + l$ for the directed case, and $\phi = k(l+2) + l$ for the undirected case.

The goal is then to identify the smallest subset of edges to be added to the network, $A \subseteq \{(v^\dagger, S_1), \ldots, (v^\dagger, S_m)\}$, such that either $inf_{(V,E \cup A)}(v^\dagger) \geq \phi$ in the *global* variant of the problem, or $\forall_{v_i \in V} inf_{(V,E \cup A)}(v^\dagger, v_i) \geq f(v_i)$ in the *individual* variant of the problem.

Recall that the influence of $v^\dagger$ is measured by setting the seed set as $\{v^\dagger\}$ and calculating the probability that other nodes get activated. Also recall that under the Linear Threshold model a node, v, gets activated if the number of its active predecessors exceeds t_v. Note that, with t_v defined as above, adding an edge $(v^\dagger, S_j)$ in the undirected case leads to the activation of nodes $s_{i,j}$ and T_i, which in turn leads to the activation of every $u_i \in S_j$ (see Figure 7). Likewise, in the directed case, adding $(v^\dagger, S_j)$ leads to the activation of every $u_i \in S_j$ (see Figure 6). To put it differently, when adding $(v^\dagger, S_j)$, the influence of $v^\dagger$ on every $u_i \in S_j$ equals 1. Importantly, the above definitions of ϕ and f imply that our goal (in both the individual and the global variants of the problem) is achieved if and only if the influence of $v^\dagger$ on *every node* $u_i \in U$ equals 1. Those observations imply that our goal is achieved if and only if we add to G a set of edges, $A \subseteq \{(v^\dagger, S_1), \ldots, (v^\dagger, S_m)\}$, such that:

$$\bigcup_{(v^\dagger, S_j) \in A} S_j = U.$$

Since we are interested in finding the smallest such subset, a solution to the above instance of the influence recovery problem gives us a solution to the Set Cover problem. □

Configuring the ROAM Heuristic

As mentioned in the main article, the ROAM heuristic involves choosing v_0 (the neighbor of $v^\dagger$ whom the heuristic will disconnect from $v^\dagger$), and choosing the $b-1$ neighbors of $v^\dagger$ whom the heuristic will connect to v_0. We conducted a number of experiments to determine whether it is more beneficial to choose v_0 as the neighbor of $v^\dagger$ with the *least* connections or the *most* connections. Likewise, we wanted to determine whether it is more beneficial to choose the $b-1$ neighbors of $v^\dagger$ (who will be connected to v_0) as the ones with the *least* connections or the *most* connections. Figure 8 compares the different settings given 50 randomly generated scale-free networks consisting of 100 nodes each, where 3 edges are added with each step of the generation process (for more details, see [2]); we chose scale-free networks as they have been shown to resemble real-life networks in many way, especially in terms of degree distribution. As for the evader, it is chosen to be the node with the lowest sum of centrality rankings (based on Degree, Closeness, and Betweenness), where ties are broken uniformly at random. As for the Independent Cascade model, we set the activation probability to be $p(v,w) = 0.15$ for every pair of nodes, $v, w \in V$. As for the Linear Threshold model, for every node, $v \in V$, the threshold value, t_v, is sampled uniformly at random from the set $\{0, \ldots, |N^{pred}(v)|\}$. For both models, the influence values are approximated using the Monte-Carlo method. In the figure, we write ROAM-x-y(b), where x can either be "*max*" or "*min*" (indicating that v_0 is the neighbor with the *most* connections or the *least* connections, respectively) and y can either be "*max*" or "*min*" (indicating that the $b-1$ neighbors are chosen to be the ones with the *most* connections or the *least* connections, respectively), whereas b represents the budget (which is set to 3 in this experiment). Since the results are averaged over 50 random networks, the shaded areas in the figure represent the 95% confidence intervals. For each network, the ROAM heuristic is executed multiple, consecutive times; the x-axis in each subfigure represents the number of executions. As can be seen, while there is no setting that dominates the others, the best overall performance seems to be achieved by ROAM-max-min(3). Based on this, in all subsequent experiments on ROAM, we choose v_0 as the neighbor of $v^\dagger$ with the *most* connections, and we connect v_0 to the $b-1$ neighbors of $v^\dagger$ with the *least* connections.

Centrality and Influence Results of ROAM

In the main article, due to space constraints, we only presented our experimental results with ROAM on three networks. In contrast, this section presents our results on 24 different networks, including the three considered in the main article. Note that the majority of the networks considered here are undirected, with the exception of the anonymized fragmetns of Twitter and Google+, as well as the directed versions of the randomly-generated networks. Before presenting our experimental results, we will first briefly explain each network.

Starting with the anonymized fragments of social-media networks, these are all taken from SNAP—the Stanford Network Analysis Platform [22].[3] More specifically, the details of these anonymized fragments are outlined below:

- Facebook: the small fragment consists of 61 nodes and 272 edges; the medium one consists of 333 nodes and 2523 edges; the large one consists of 786 nodes and 14027 edges;
- Twitter: the small fragment consists of 201 nodes and 2503 edges; the medium one consists of 247 nodes and 8041 edges; the large one consists of 235 nodes and 15957 edges;
- Google+: the small fragment consists of 108 nodes and 2884 edges; the medium one contains 215 nodes and 7132 edges; the large one consists of 338 nodes and 12341 edges.

We also experimented with ROAM given three different terrorist networks, the details of which are outlined below:

- the WTC 9/11 terrorist network [21];
- the 2002 Bali attack network [19];
- the 2004 Madrid train bombings network [19].

Finally, we considered three widely-used models of random networks, as detailed below:

- *Scale-free* networks, generated using the Barabasi-Albert model [2]. We denote any such network by $ScaleFree(x, y)$, where x is the number of nodes, and y is the number of links added with each node. For directed networks, we write $dScaleFree(x, y)$, and set the added links to be directed from each new node to the existing ones.
- *Small-world* networks, generated using the Watts-Strogatz model [32]. We denote any such network by $SmallWorld(x, y, z)$, where x is the number of nodes, y is the average degree, and z is the rewiring probability. For directed networks, we write $dSmallWorld(x, y, z)$, and take y to be the average out-degree.

[3] `http://snap.stanford.edu/data/#socnets`

- *Random graphs* generated using the Erdos-Renyi model [12]. We write $RandomGraph(x, y)$, with x being the number of nodes, and y being the expected average degree. In contrast, when the network is directed, we write $dRandomGraph(x, y)$, and take y to be the expected average out-degree.

Each experiment consists of a network, a budget, an evader, and an influence model. We experiment with a budget of 1, 2, 3, and 4. The evader is assumed to be the one with the lowest sum of centrality rankings (based on Degree, Closeness, and Betweenness), where ties are broken uniformly at random. Whenever the *Independent Cascade* model is used, an activation probability of 0.15 is assumed on each link. Whenever the *Linear Threshold* model is used, a uniform distribution of thresholds is assumed. For both models, the influence values are approximated using the Monte-Carlo method.

Figures 9 to 16 depict the evader's ranking after each execution of ROAM. As can be seen, the heuristic is able to decrease the evader's ranking according to the three centrality measures (degree, closeness, and betweenness) with varying levels of success, depending on the network at hand, and the budget being spent on rewiring the network. Overall, the heuristic seems to be generally capable of reducing the centrality of the evader given different types of networks, and given different centrality measures.

Next, we evaluate ROAM in terms of recovering (some of) the influence that the evader, $v^\dagger$, lost during the evasion process. More precisely, we evaluate the effectiveness of *Step 2*, whose main purpose is to recover the influence that $v^\dagger$ lost during *Step 1* of the heuristic. To this end, it suffices to calculate the influence of $v^\dagger$ given different budgets. To see why this is the case, recall that the budget, b, is spent as follows: one modification is spent in *Step 1*, while the remaining $b - 1$ modifications are spent in *Step 2*. This basically means that, by setting $b = 1$, we effectively disable *Step 2*. Conversely, by increasing b, we increase the impact of *Step 2*, and thus we expect the evader's influence to increase accordingly. To verify this, we plotted the evader's relative influence value (compared to his or her original influence value before executing ROAM); this value was measured according to two alternative models of influence, namely Independent Cascade and Linear Threshold (see Figures 17 to 24). As expected, when $b = 1$, the evader's influence decreases steadily, since *Step 2* is disabled. Conversely, when $b > 1$, *Step 2* is activated, and some of the evader's lost influence is recovered as a result. Better still, in some of our experiments, when $b > 3$ the influence of $v^\dagger$ actually exceeds its original value (i.e., its value before executing ROAM altogether). This means that ROAM is not only able to hide the evader, but may even boost the evader's influence, depending on the network and the budget at hand.

Similar trends were observed when testing ROAM given other directed and undirected networks (see Section) and given other centrality measures taken from a commercial tool for social network analysis, namely UCINET [7] (see Section), all of which further demonstrate the universality of our findings across different networks and centrality measures.

Concealment Results of DICE

In the main article, we only presented part of our main experimental results with DICE. In contrast, this section presents all of them. More specifically, the results presented here are for all the networks that were described earlier in Section .

For each network, we experiment with seven community-detection algorithms implemented in the *igraph* [9] package of the R language (version $1.0.1$), namely: Eigenvector [23], Betweenness [24], Walktrap [25], Louvain [4], Greedy [8], Infomap [28] and Spinglass [26]. Every experiment consists of a network and a community-detection algorithm. The experiment starts by running the algorithm to obtain a community structure, CS. After that, the group of evaders, i.e., $V^\dagger$, is chosen to be the element in CS whose size is the median of the sizes of all communities in CS (ties are broken uniformly at random). Although $V^\dagger$ does not necessarily have to be an element of CS, we choose it this way in order to study the worst case scenario in which $V^\dagger$ is initially exposed completely by the algorithm. We proceed with the experiment only if $2 < |V^\dagger| < n-2$, to avoid extreme cases in which the $V^\dagger$ happens to be either extremely small or extremely large.

The experiment proceeds in rounds, each involving the execution of DICE followed by the execution of the community-detection algorithm, to measure how well $V^\dagger$ is hidden in the new outcome of the algorithm (to this end we use μ with $\alpha = 0.5$). We set the number of rounds to be $|V^\dagger|$. In each round, we disconnect d links from within $V^\dagger$ (chosen uniformly at random), and then connect $b-d$ members of $V^\dagger$ to $b-d$ non-members of $V^\dagger$ (again chosen uniformly at random). Due to this randomness in our implementation, DICE may yield different results in different executions. Thus, we repeat each experiment multiple times, with the shaded areas representing the 95% confidence interval.

We tested DICE given different networks and different community-detection algorithms; see Figures 25 to 31. As can be seen, DICE is able to hide the group of evaders, $V^\dagger$, with varying levels of success, depending on the evaders' budget and the seeker's community-detection algorithm. Surprisingly, the parameter d did not seem to have a significant impact on performance. The same observation was made when running DICE on a wider variety of networks (see Section) and when testing DICE against a commercial tool for social network analysis, namely UCINET [7] (see Section). This observation implies that the members of $V^\dagger$ can choose at their discretion how much of the budget is spent on removing internal links, and how much is spent on adding external links, without worrying about how this may affect their disguise. For example, the members of $V^\dagger$ might be interested in hiding as much as possible, while removing as few internal links as possible. However, since the addition of an external link is not entirely under the control of $V^\dagger$ (as it requires the consent of a non-member), the number of newly-added external links may be insufficient for providing a satisfactory level of concealment, in which case the members can compensate for this by sacrificing more internal links, i.e., by increasing the parameter d.

Figure 32 illustrates the average value of the concealment measure μ with $\alpha = 0.5$, taken over all of our experiments of DICE where $b = 4$ and $d \in \{0, 2, 4\}$ (the subfigures on the

righthand side are for directed networks, whereas those on the lefthand side are for undirected ones). For every subfigure, each row represents a community-detection algorithm, each column represents a network, and the intensity of the color in each cell represents the average value of μ, taken over 50 simulations, either by generating a new random network in each simulation, or by re-running the simulation over and over on the same real-life network (note that our implementation of DICE is non-deterministic, and may yield different results on the same network). The missing cells correspond to the cases where $V^{\dagger}$ happened to be either extremely small or extremely large (recall that we only consider cases where $2 < |V^{\dagger}| < n-2$). As can be seen, the Infomap algorithm [28] seems to be on average the most difficult to fool when it comes to the undirected networks. In contrast, when it comes to directed networks, Infomap appears to be slightly easier to fool compared to the only other alternative for directed networks, namely the Betweenness algorithm [24] (see the right column of Figure 32). The same observations were made when testing DICE on other directed and undirected networks; see Section .

Experiments on Running ROAM and DICE Simultaneously

In this section, we study how our ROAM heuristic (which hides a single evader) and our DICE heuristic (which hides a group of evaders) influence each other. Each of our experiments consists of a community-detection algorithm and a network. The experiment starts by running the algorithm to obtain a community structure, CS. After that, the group of evaders, i.e., $V^\dagger$, is chosen to be the element in CS whose size is the median of the sizes of all communities in CS (ties are broken uniformly at random). Although $V^\dagger$ does not necessarily have to be an element of CS, we choose it this way in order to study the worst case scenario in which $V^\dagger$ is initially exposed completely by the algorithm. We proceed with the experiment only if $2 < |V^\dagger| < n-2$, to avoid extreme cases in which the $V^\dagger$ happens to be either extremely small or extremely large. The member of $V^\dagger$ with the lowest sum of centrality rankings (based on Degree, Closeness, and Betweenness) is then chosen as the evader, $v^\dagger$, who is interpreted as the leader of $V^\dagger$ and who will be running the ROAM heuristic.

The experiment proceeds in rounds, each involving the execution of ROAM (to hide $v^\dagger$) followed by the execution of DICE (to hide $V^\dagger$). At the end of each round:

(a). We measure the effectiveness of ROAM in terms of decreasing the centrality of $v^\dagger$, and in terms of recovering the influence of $v^\dagger$ that was lost during the centrality-minimization stage;

(b). We measure the effectiveness of DICE in terms of hiding $V^\dagger$. To this end, we first run the chosen community detection algorithm to obtain a community structure, CS, and then we compute $\mu(V^\dagger, CS)$, where μ is our concealment measure with $\alpha = 0.5$ (see Section for more details).

We set the number of rounds to be $\lceil |V^\dagger|/b \rceil$, where b is the budget of the DICE heuristic. In each round, we disconnect d links from within $V^\dagger$ (chosen uniformly at random), and then connect $b-d$ members of $V^\dagger$ to $b-d$ non-members of $V^\dagger$ (again chosen uniformly at random). Due to this randomness in our implementation, DICE may yield different results in different executions. Therefore, we repeat each experiment multiple times, and report the 95% confidence interval.

Figures 33 to 46 present the results of our experiments with ROAM$(b) : b = 3$ where b is the budget in each round, and with DICE$(b, d) : b = 4; d = 2$ where b is the budget of a single round, and d is the number of removed intra-community edges. Each result is taken as an average over 50 simulations, with the shaded areas representing the 95% confidence intervals. Next, we comment on the effectiveness of our heuristics in terms of reducing the centrality of $v^\dagger$, recovering the influence of $v^\dagger$, and increasing the concealment of $V^\dagger$.

In terms of reducing the centrality of $v^\dagger$, the ROAM heuristics is generally successful on most of the undirected networks considered in our experiments. In contrast, it seems that directed networks are more difficult to handle compared to the undirected ones. In fact, with directed networks, there are a few cases where running ROAM and DICE simultaneously caused an increase rather than a decrease in the centrality of $v^\dagger$.

In terms of the influence of $v^\dagger$, the results are quite mixed. In particular, there are cases where the drop in influence is substantial, there are cases where the drop is minor, and there are cases where the influence actually increases. This suggests that hiding the community may come at the expense of significantly reducing the influence of its leader.

Finally, in terms of increasing the concealment of $V^\dagger$, running our heuristics in parallel does not appear to have a marked impact on the effectiveness of concealing $V^\dagger$. Furthermore, as shown in Figure 46, the Infomap algorithm [28] is on average the most difficult to fool given undirected networks (this is the same observation that was made when running DICE without ROAM). Note that the missing cells in Figure 46 correspond to the cases where $V^\dagger$ happened to be either extremely small or extremely large (recall that we only consider cases where $2 < |V^\dagger| < n - 2$).

Robustness of ROAM in the Presence of Noise

In this section, we consider a practical implementation concern, which is that of noise. So far, in all of our experiments we assumed that the evader knows all the connections of each of his/her neighbors. In practice, however, this may not necessarily be the case. For example, if a typical user of Facebook were to check his/her list of friends, he/she will probably find some who keep their own lists of friends private, i.e., not visible to the user. The lack of such information is a form of noise that could be detrimental to the performance of ROAM.

To study the impact that such noise may have on our heuristic, we ran experiments in which a certain percentage of the nodes (chosen uniformly at random) keep their connections private, i.e., not visible to the evader, $v^\dagger$. When faced with such a situation, ROAM chooses v_0 as the neighbor with the most connections, taking into consideration only the neighbors whose connections are visible to $v^\dagger$. Similarly, when ROAM chooses the $b-1$ neighbors with the least connections (to introduce them to v_0), this choice is made starting with the ones whose connections are visible to $v^\dagger$. If there are fewer than $b-1$ such neighbors, then ROAM randomly chooses the remaining ones out of the neighbors whose connections are private. Aside from this modification, the heuristic proceeds as usual. To avoid any potential confusion between the two versions of ROAM, we call this latter version "nROAM", where the "n" stands for "noise". When presenting the experimental results, we will write "nROAM(b, p)", where b denotes the budget in each execution, and p denotes the percentage of randomly-chosen nodes whose connections are kept private.

Figures 47 to 56 present the results of our experiments with noise. As expected, this lack of information affects the performance of ROAM, but the impact is often negligible. More specifically, in terms of influence recovery, the drop in performance seems negligible in about 70% of our experiments. Similarly, in terms of centrality minimization, the drop in performance appears to be negligible in about 80% our experiments. These results hold even the majority of the evader's neighbors keep their connections private.

Testing ROAM Against a Commercial Forensic Tool

We now evaluate ROAM against the following centrality measures implemented in a widely used commercial forensic tool, namely UCINET [7]:

- 2-*Local* centrality [6]—a node's centrality sums the normalized degrees of its neighbors.
- *Average Reciprocal Distance (ARD)* centrality—a node's centrality is the average reciprocal distance to all other nodes in the network.
- *Beta* centrality [5]—also known as the *Bonacich power index*, whereby a node's power is influenced by the power of its neighbors (here, we set the β parameter to its default value).
- *Eigenvector* centrality [5]—whereby the centrality values of nodes are the values from the eigenvector corresponding to the largest eigenvalue of the network's adjacency matrix.
- 2-*Step Reach* centrality—a node's centrality is the number of nodes that can be reached in at most two steps from that node.
- *Aggregate* centrality—a node's centrality is taken as the average reciprocal of the node's position in the ranking produced by each of the other centralities, i.e., the above five centralities, as well as degree, closeness and betweenness.

As with all other sections, the evader is chosen to be the node with the lowest sum of centrality rankings (based on Degree, Closeness, and Betweenness), where ties are broken uniformly at random. As for the random networks, each result is taken as the average over 50 simulations, with the shaded areas representing the 95% confidence intervals. As for the directed networks, we do not report results for Eigenvector centrality, since eigenvectors can be only computed for symmetric matrices. We also report both the in-degree and the out-degree values for each of the following centralities: 2-Local, ARD, Beta and 2-Step Reach. Note that we only report the centrality, not the influence, of the evader. This is because the influence results are exactly the same as those found in Section (note that ROAM performs the same steps, regardless of the centrality measure being used).

Figures 59 to 72 depict the evader's ranking after each execution of ROAM. As can be seen, the heuristic is able to decrease the evader's ranking with varying levels of success, depending on the network at hand, and the budget being spent on rewiring the network. Overall, the results in this section are very similar to those obtained when testing ROAM against other networks and other centrality measures (see Sections and), all of which further demonstrate the universality of our findings across different networks and centrality measures.

Testing DICE Against a Commercial Forensic Tool

In this section, we evaluate DICE against a commercial forensic tool, namely UCINET [7], which is widely used for social network analysis. Note that UCINET's command-line interface provides three community detection algorithms:

- *Factions*, which uses Tabu Search [16];
- *Newman* community detection algorithm [24];
- *Split*, which partitions the nodes into exactly two communities.

As far as the Newman algorithm is concerned, we already considered it in all of our previous DICE experiments; this is the same as the Betweenness algorithm from the *igraph* package. As for the Split algorithm, it can only return community structures with exactly two communities, which is very restrictive in the context of our article. This leaves us with only the Factions algorithm, which is the focus of our next experiments. When running the Factions algorithm, one has to specify the exact number of communities that it returns. As such, we run this algorithm multiple times, with the number of communities ranging from 2 to 8. Out of those community structures, we choose the one that maximizes the normalized modularity [11]. Once we have chose the community structure, CS, we choose the group of evaders, i.e., $V^\dagger$, as the element in CS whose size is the median of the sizes of all communities in CS (ties are broken uniformly at random). The experiment then proceeds in rounds, each involving the execution of DICE followed by the execution of the community-detection algorithm, to measure how well $V^\dagger$ is hidden in the new outcome of the algorithm (to this end we use μ with $\alpha = 0.5$). We set the number of rounds to be $\lceil |V^\dagger|/b \rceil$.

Due to computational complexity of the Factions algorithm, we were only able to obtain results for small networks, consisting of no more tens of nodes each; see Figure 73. As can be seen, DICE is able to hide the group of evaders with ease. Based on these results, the Factions algorithm seems on average easier to fool than many of the other community detection algorithms considered in earlier sections.

Experiments on Running ROAM by Multiple Individuals

So far, in all of our experiments we assumed that there exists only a single evader that is using ROAM to decrease his or her centrality. In this section, we study the case in which multiple such evaders exist, each trying to decrease his or her centrality without any coordination with the remaining evaders. That is, we run experiments in which multiple evaders are each running our ROAM heuristic at the same time.

Each experiment starts by choosing an evader $v^\dagger$; this is done just like in all of our previous experiments with ROAM, i.e., $v^\dagger$ is chosen to be the node with the lowest sum of centrality rankings (based on Degree, Closeness, and Betweenness), where ties are broken uniformly at random. After that, three additional evaders are chosen to be the three nodes with the lowest sums of centrality rankings out of all the nodes that are exactly k steps away from $v^\dagger$ (again, ties are broken uniformly at random). In each round, the evaders run the ROAM heuristic in turn; the order in which they proceed is determined according to their sum of centrality rankings, from smallest to largest, implying that $v^\dagger$ is the first evader to run the heuristic. We write ROAM(b)+k to refer to the experiment in which ROAM is executed repeatedly, where b is the budget of each execution of ROAM, and k is the distance between $v^\dagger$ and the remaining evaders. As a baseline, we also run an experiment in which $v^\dagger$ is running ROAM alone; we refer to this experiment by writing ROAM(b).

Figures 74 to 85 depict the resulting centrality and influence of $v^\dagger$. For the random networks, the figures depict an average taken over 50 simulations, with the shaded areas representing the 95% confidence intervals. Overall, the closer the additional evaders are to $v^\dagger$, the greater their impact on the centrality and influence of $v^\dagger$. In many cases, when additional evaders are running the ROAM heuristic, the process of lowering the centrality of $v^\dagger$ becomes less effective. Interestingly, in many of our experiments, the actions of the additional evaders actually increase, rather than decrease, the effectiveness of recovering the influence of $v^\dagger$.

Experiments with Other Types of Networks

In the previous sections, we evaluated ROAM and DICE on three kinds of networks: (i) social networks, (ii) terrorist networks, and (iii) random networks. In this section, we evaluate ROAM and DICE on other kinds of networks:

- Undirected co-membership network of CEOs [15]. This network consists of 26 nodes and 284 edges. Here, two CEOs are connected with an edge if and only if they are members of the same club.

- Undirected co-membership network of southern women [10]. This network consists of 18 nodes and 139 edges. Here, two women are connected with an edge if and only if they participated in the same social event.

- Undirected co-location network of students at the St Lucia campus of University of Queensland [31]. This network is very large, consisting of 20,308 nodes and 664,146 edges. To obtain a network of tractable size, we perform our experiments on the co-location network of students visiting the 60th most popular location in the dataset. In this network, two students are connected with an edge if and only if they visited this location at the same time. We ended up with a network consisting of 302 nodes and 1,149 edges.

- Undirected email communication network of *University at Rovira i Virgili* (URV) studied in [18], which contains 1,669 users including faculty, researchers, technicians, managers, administrators, and graduate students. Since this network is not connected, we run our experiments on the giant component therein, consisting of 1,133 nodes and 5,451 edges.

- Directed network of transactions between *Bitcoin* users [27]. The Bitcoin network is massive, with 881,678 nodes and 1,607,976 edges. To obtain a network of tractable size, we performed our experiments on the largest connected component of network of users who performed transaction in May 2010. We ended up with a network consisting of 753 nodes and 859 edges.

- Directed network of countries trading manufactures of metal in 1994 [30]. This network consists of 80 nodes and 998 edges.

As for ROAM, every experiment consists of a network, a budget, an evader, and an influence model. More specifically, we experiment with a budget of 1, 2, 3, and 4. The evader, $v^\dagger$, is assumed to be the one with the lowest sum of centrality rankings (based on Degree, Closeness, and Betweenness), where ties are broken uniformly at random. Whenever the *Independent Cascade* model is used, an activation probability of 0.15 is assumed on each link. On the other hand, whenever the *Linear Threshold* model is used, a uniform distribution of thresholds is assumed (see Section for more details). For both models, the influence values are approximated using the Monte-Carlo method. In each of these experiments, the ROAM heuristic is executed multiple, consecutive times, to see how this affects the centrality and influence of the evader.

Figures 86 and 87 depict the evader's ranking after each execution of ROAM. As can be seen, both in directed and undirected networks, the heuristic is able to decrease the evader's ranking according to the three centrality measures (degree, closeness, and betweenness) with varying levels of success, depending on the network at hand, and the budget being spent on rewiring the network. Note that these results are consistent with our findings from previous sections.

Next, we evaluate ROAM in terms of recovering (some of) the influence that the evader, $v^\dagger$, lost during the evasion process. More precisely, we evaluate the effectiveness of *Step 2*, whose main purpose is to recover the influence that $v^\dagger$ lost during *Step 1* of the heuristic. To this end, it suffices to calculate the influence of $v^\dagger$ given different budgets. To see why this is the case, recall that the budget, b, is spent as follows: one modification is spent in *Step 1*, while the remaining $b-1$ modifications are spent in *Step 2*. This basically means that, by setting $b=1$, we effectively disable *Step 2*. Conversely, by increasing b, we increase the impact of *Step 2*, and thus we expect the evader's influence to increase accordingly. To verify this, we plotted the evader's relative influence value (compared to his or her original influence value before executing ROAM); this value was measured according to two alternative models of influence, namely Independent Cascade and Linear Threshold (see Figures 88 and 89). As expected, when $b=1$, the evader's influence decreases steadily, since *Step 2* is disabled. Conversely, when $b>1$, *Step 2* is activated, and some of the evader's lost influence is recovered as a result. Better still, in some of our experiments, when $b>3$ the influence of $v^\dagger$ actually exceeds its original value (i.e., its value before executing ROAM altogether). This means that ROAM is not only able to hide the evader, but may even boost the evader's influence, depending on the network and the budget at hand. Again, these results are consistent with our findings from other sections.

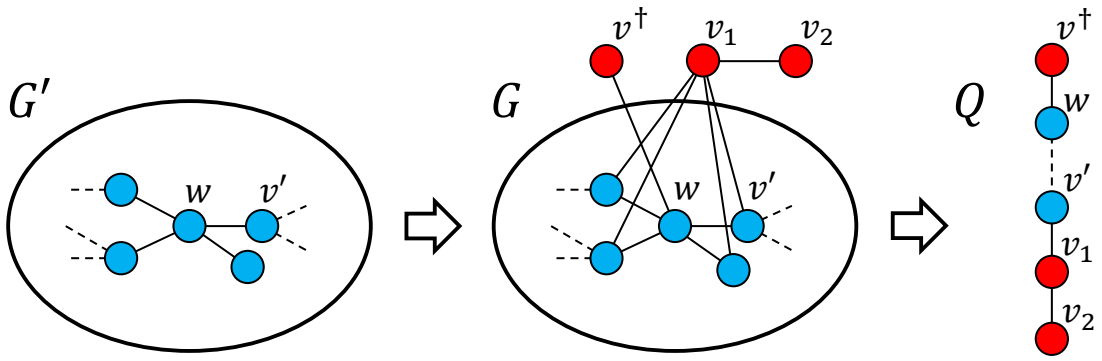
As for DICE, every experiment consists of a network and a community-detection algorithm. The experiment starts by running the algorithm to obtain a community structure, CS. After that, the group of evaders, i.e., $V^\dagger$, is chosen to be the element in CS whose size is the median of the sizes of all communities in CS (ties are broken uniformly at random). Although $V^\dagger$ does not necessarily have to be an element of CS, we choose it this way in order to study the worst case scenario in which $V^\dagger$ is initially exposed completely by the algorithm. We proceed with the experiment only if $2<|V^\dagger|<n-2$, to avoid extreme cases in which the $V^\dagger$ happens to be either extremely small or extremely large. The experiment then proceeds in rounds, each involving the execution of DICE followed by the execution of the community-detection algorithm, to measure how well $V^\dagger$ is hidden in the new outcome of the algorithm (to this end we use μ with $\alpha=0.5$). We set the number of rounds to be $|V^\dagger|$. In each round, we disconnect d links from within $V^\dagger$ (chosen uniformly at random), and then connect $b-d$ members of $V^\dagger$ to $b-d$ non-members of $V^\dagger$ (again chosen uniformly at random). Due to this randomness in our implementation, DICE may yield different results in different executions. Therefore, we repeat each experiment multiple times, with the shaded areas representing the 95% confidence interval.

Figures 90 and 91 depict the results of our experiments with DICE given different networks and different community-detection algorithms. As the figure shows, DICE is able to hide the

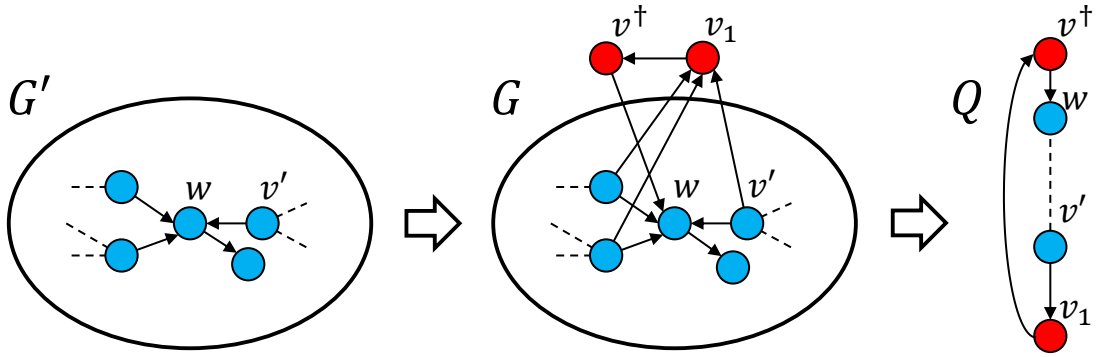
group of evaders, $V^{\dagger}$, with varying levels of success, depending on the evaders' budget and the seeker's community-detection algorithm. Surprisingly, both on directed and undirected networks, the parameter d did not seem to have a significant impact on performance. The same observation was made when running DICE on other networks (see Section) and when testing DICE against a commercial tool for social network analysis, namely UCINET [7] (see Section). This observation implies that the members of $V^{\dagger}$ can choose at their discretion how much of the budget is spent on removing internal links, and how much is spent on adding external links, without worrying about how this may affect their disguise.

Figure 92 illustrates the average value of the concealment measure μ with $\alpha = 0.5$, taken over all of our experiments of DICE where $b = 4$ and $d \in \{0, 2, 4\}$. In particular, each row represents a community-detection algorithm, each column represents a network, and the intensity of the color in each cell represents the average value of μ, taken over 50 simulations, either by generating a new random network in each simulation, or by re-running the simulation over and over on the same real-life network (recall that our implementation of DICE is non-deterministic, and may yield different results on the same network). The missing cells corresponds to the cases where $V^{\dagger}$ happened to be either extremely small or extremely large (recall that we only consider cases where $2 < |V^{\dagger}| < n - 2$). Given undirected networks, the Infomap [28] algorithm seems to be the most difficult to fool. These results suggest that Infomap should be the algorithm of choice from the seeker's perspective. In contrast, given directed networks, Infomap is in fact the easiest algorithm to fool, suggesting that the seeker's algorithm of choice should be the Betweenness algorithm [24] when dealing with directed networks. Again, these results are consistent with our findings from previous sections.

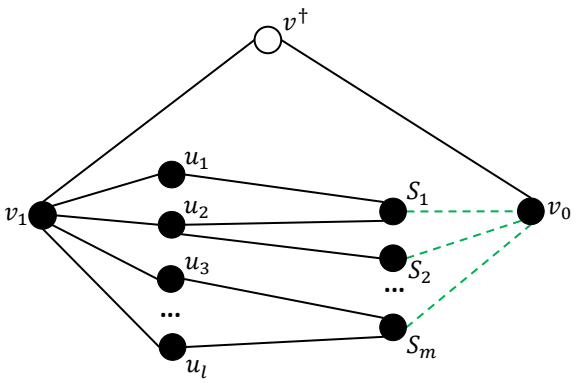
Supplementary Figures



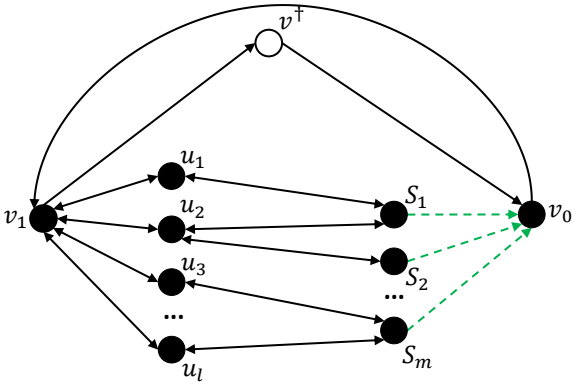
Supplementary Figure 1: The main steps of reducing the Hamiltonian cycle problem to the problem of determining whether the closeness centrality of $v^\dagger$ can be reduced to a value $\leq q$ in an **undirected network**.



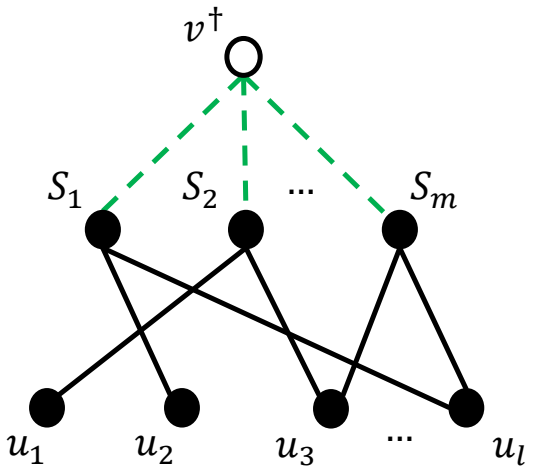
Supplementary Figure 2: The main steps of reducing the Hamiltonian cycle problem to the problem of determining whether the closeness centrality of $v^\dagger$ can be reduced to a value $\leq q$ in a **directed network**.



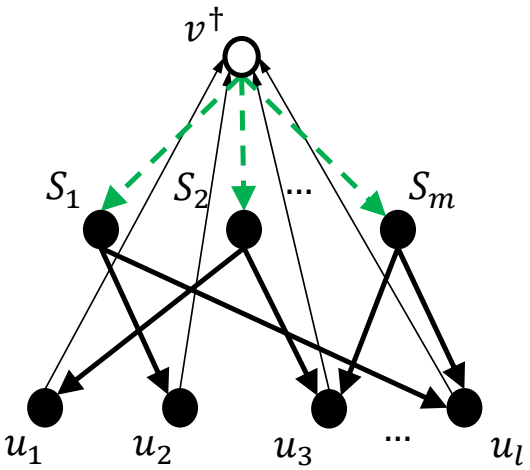
Supplementary Figure 3: Undirected network used to reduce the *Set cover* problem to our problem of disguising the betweenness centrality of $v^\dagger$. To solve both problems, we consider adding (some of) the dashed edges.



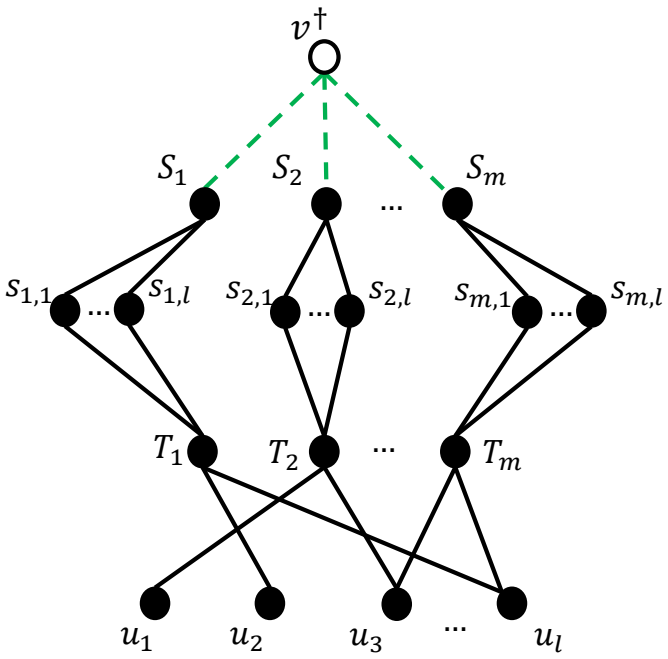
Supplementary Figure 4: A directed network used to reduce the *Set cover* problem to our problem of disguising the betweenness centrality of $v^\dagger$. To solve both problems, we consider adding (some of) the dashed edges.



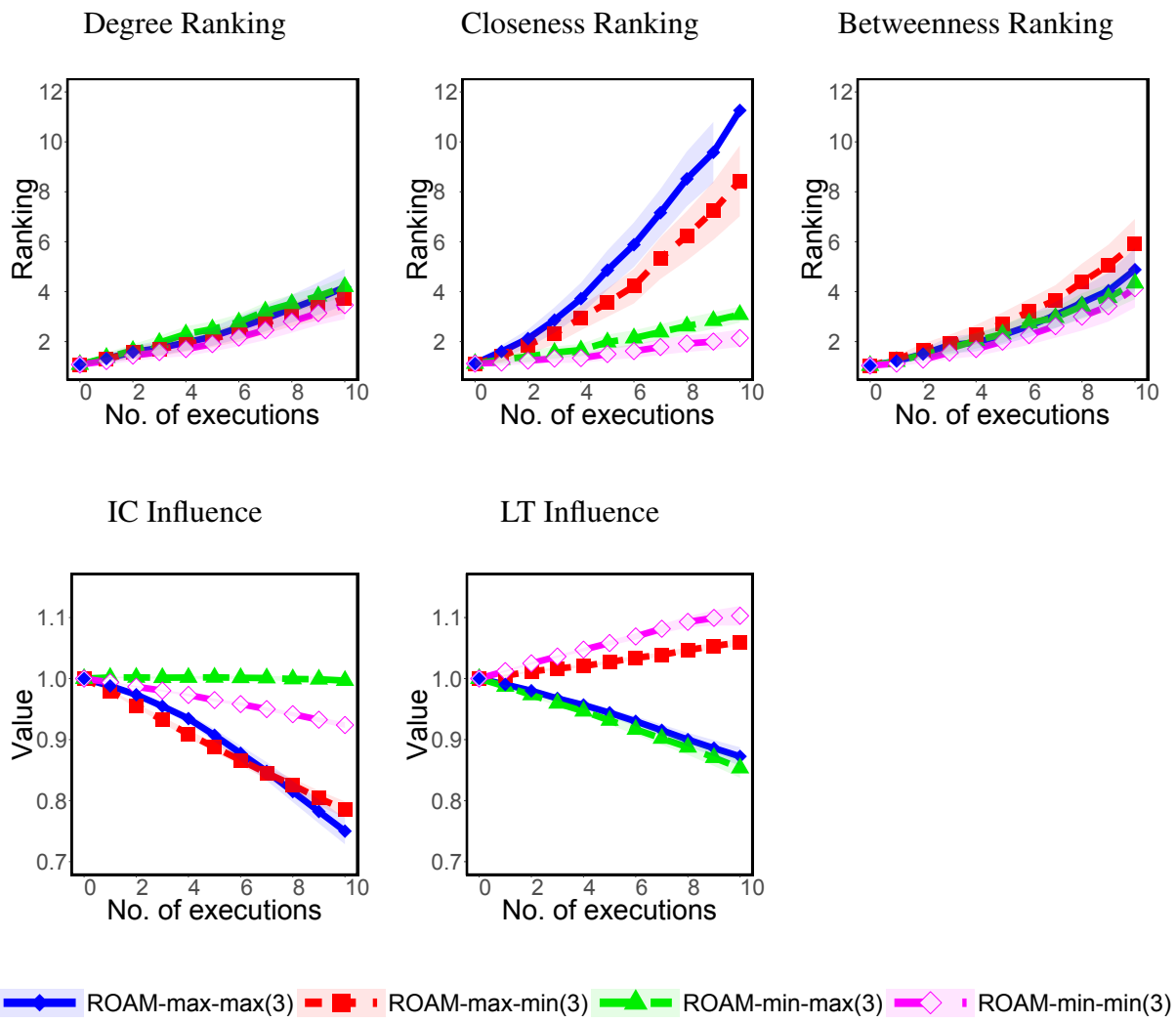
Supplementary Figure 5: Undirected network used to reduce the *Set cover* problem to our influence recovery problem. To solve both problems, we consider adding (some of) the dashed edges.



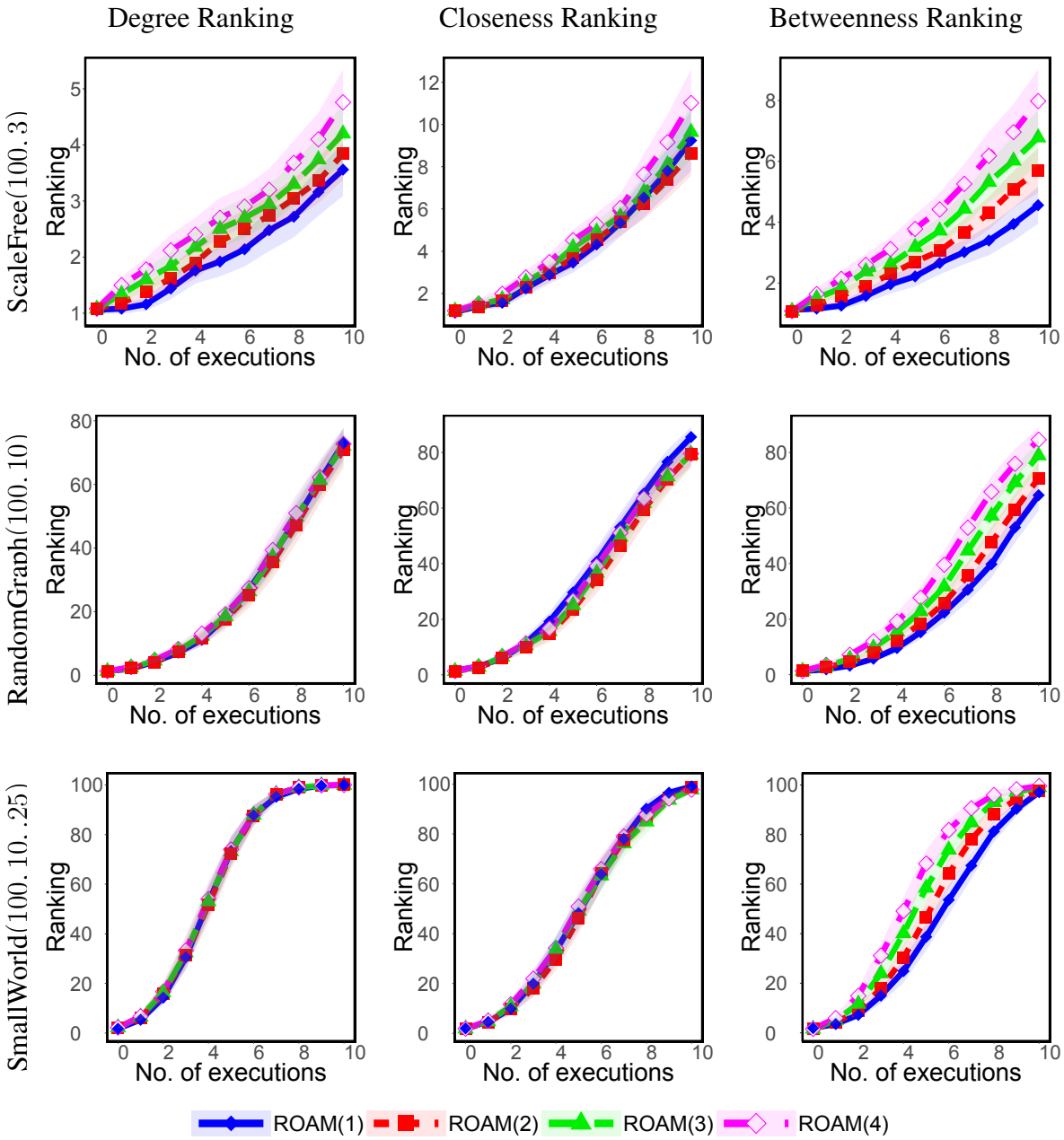
Supplementary Figure 6: A directed network used to reduce the *Set cover* problem to our influence recovery problem. To solve both problems, we consider adding (some of) the dashed edges.



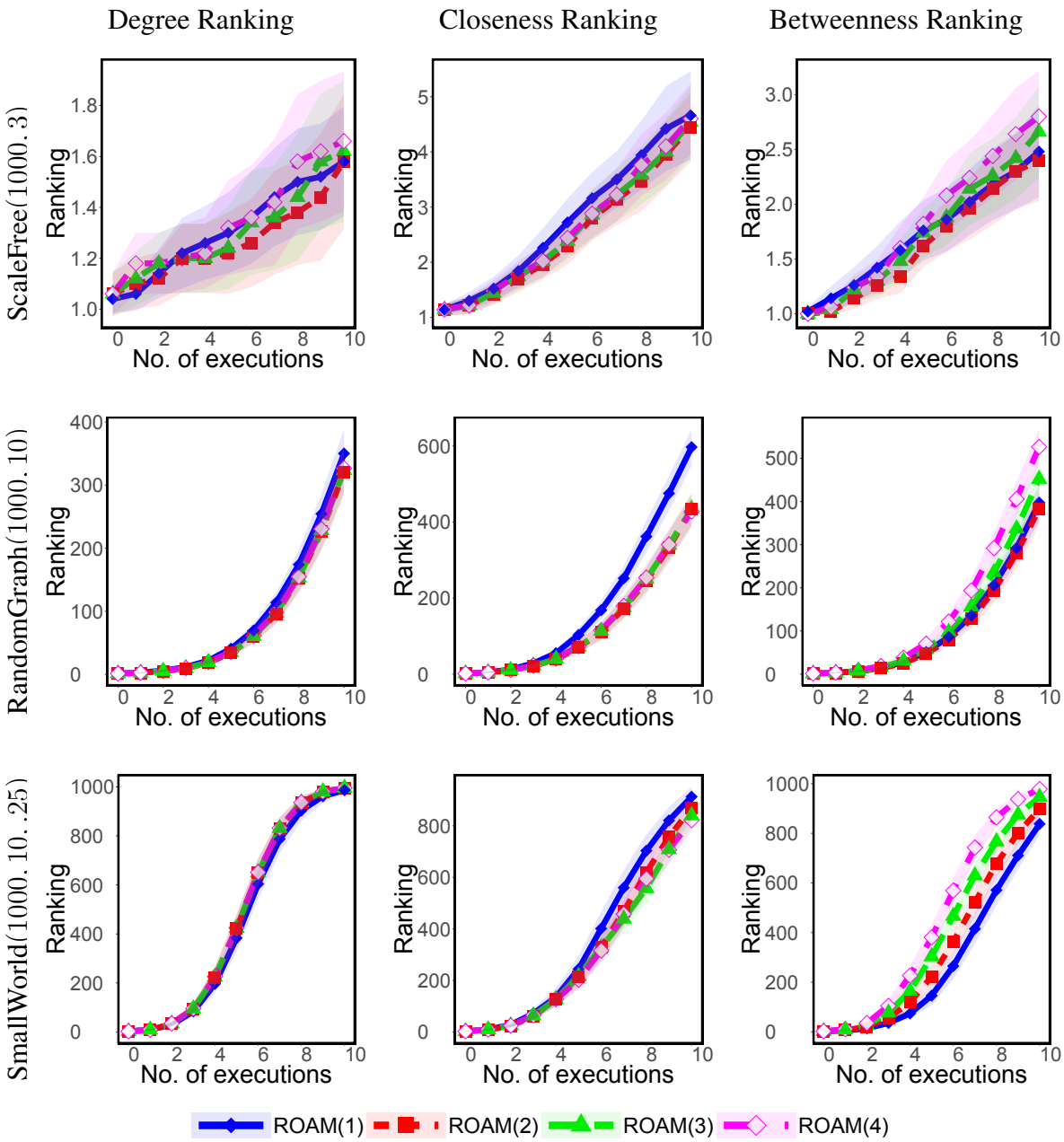
Supplementary Figure 7: Undirected network used to reduce the *Set cover* problem to our influence recovery problem under the Linear Threshold model. To solve both problems, we consider adding (some of) the dashed edges.



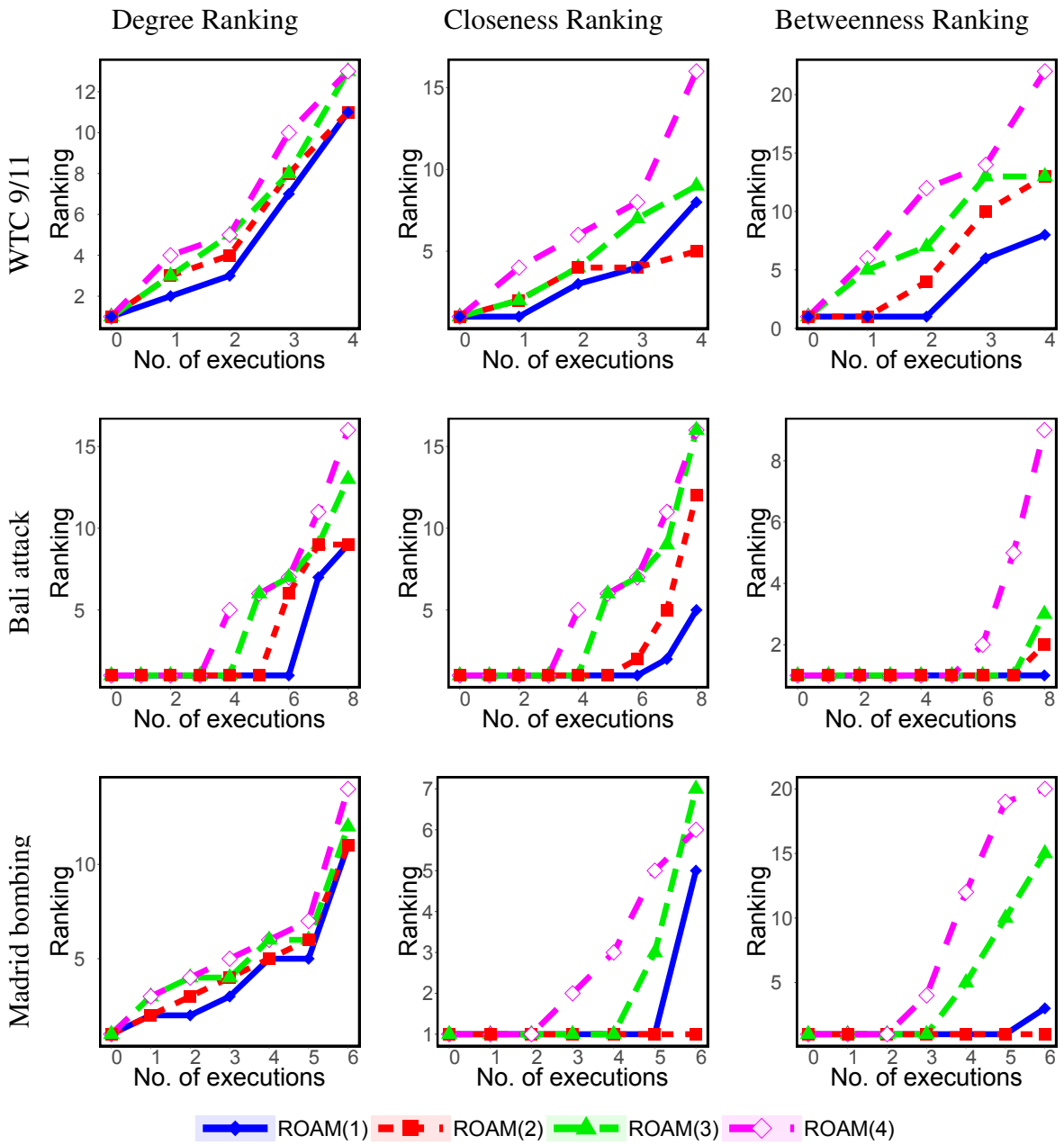
Supplementary Figure 8: Comparing different settings of ROAM on 50 randomly generated scale-free networks consisting of 100 nodes, with 3 edges added in each step of the generation process. For each such network, ROAM is executed multiple, consecutive times (the x-axis represents the number of executions). The subfigures show the evader's ranking (according to different centrality measures), and the relative change in its influence value (according to different influence models).



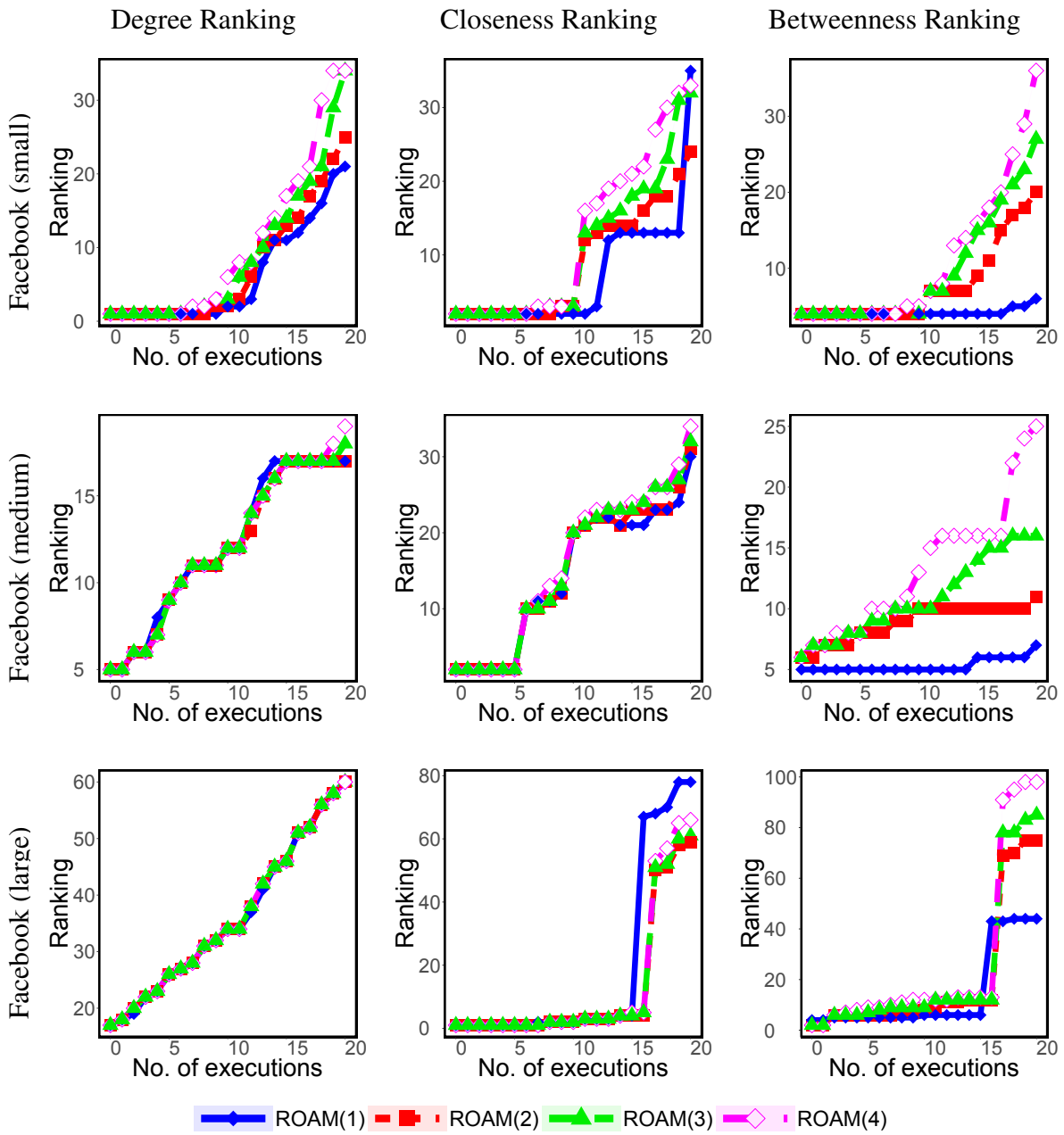
Supplementary Figure 9: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



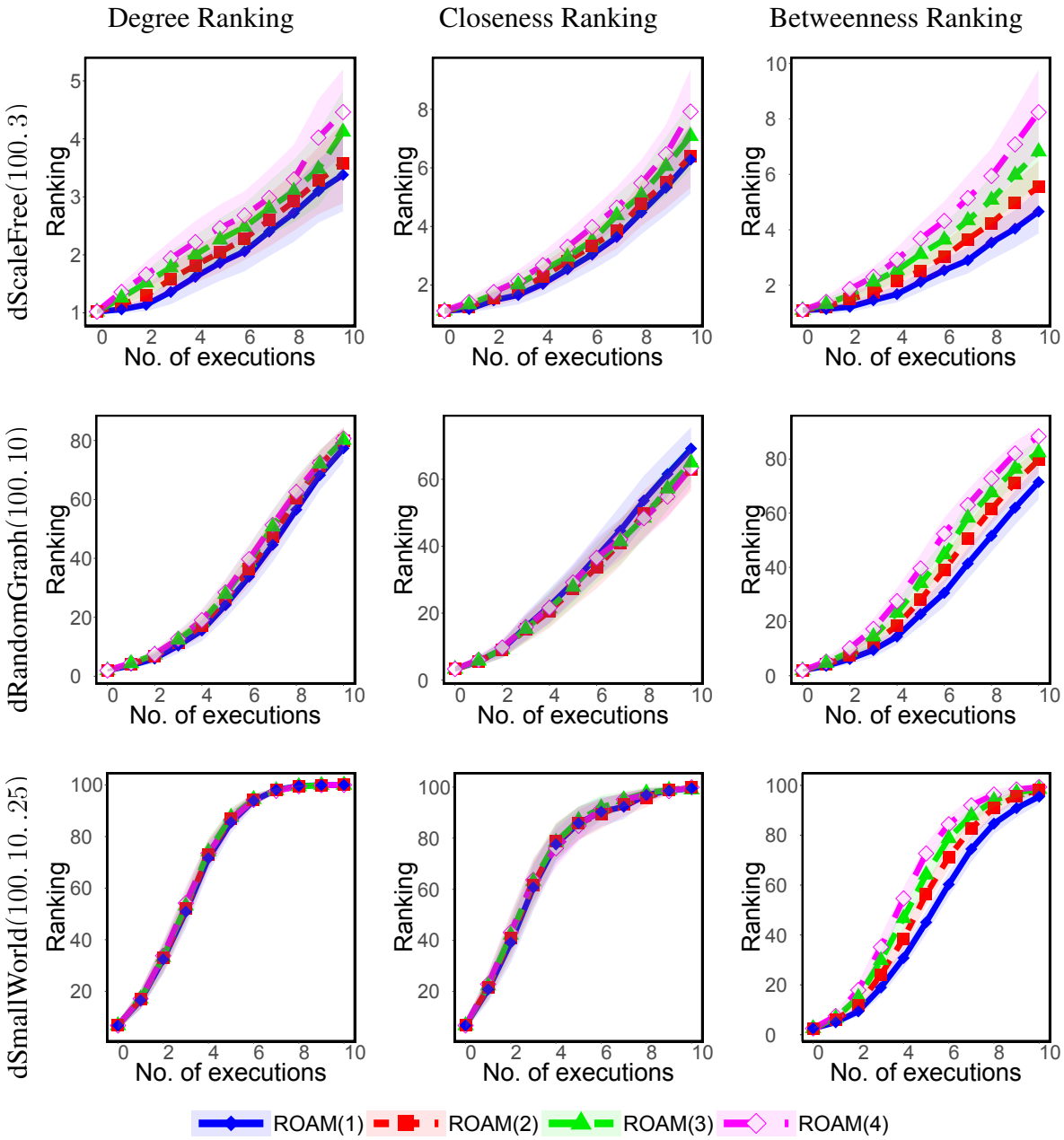
Supplementary Figure 10: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



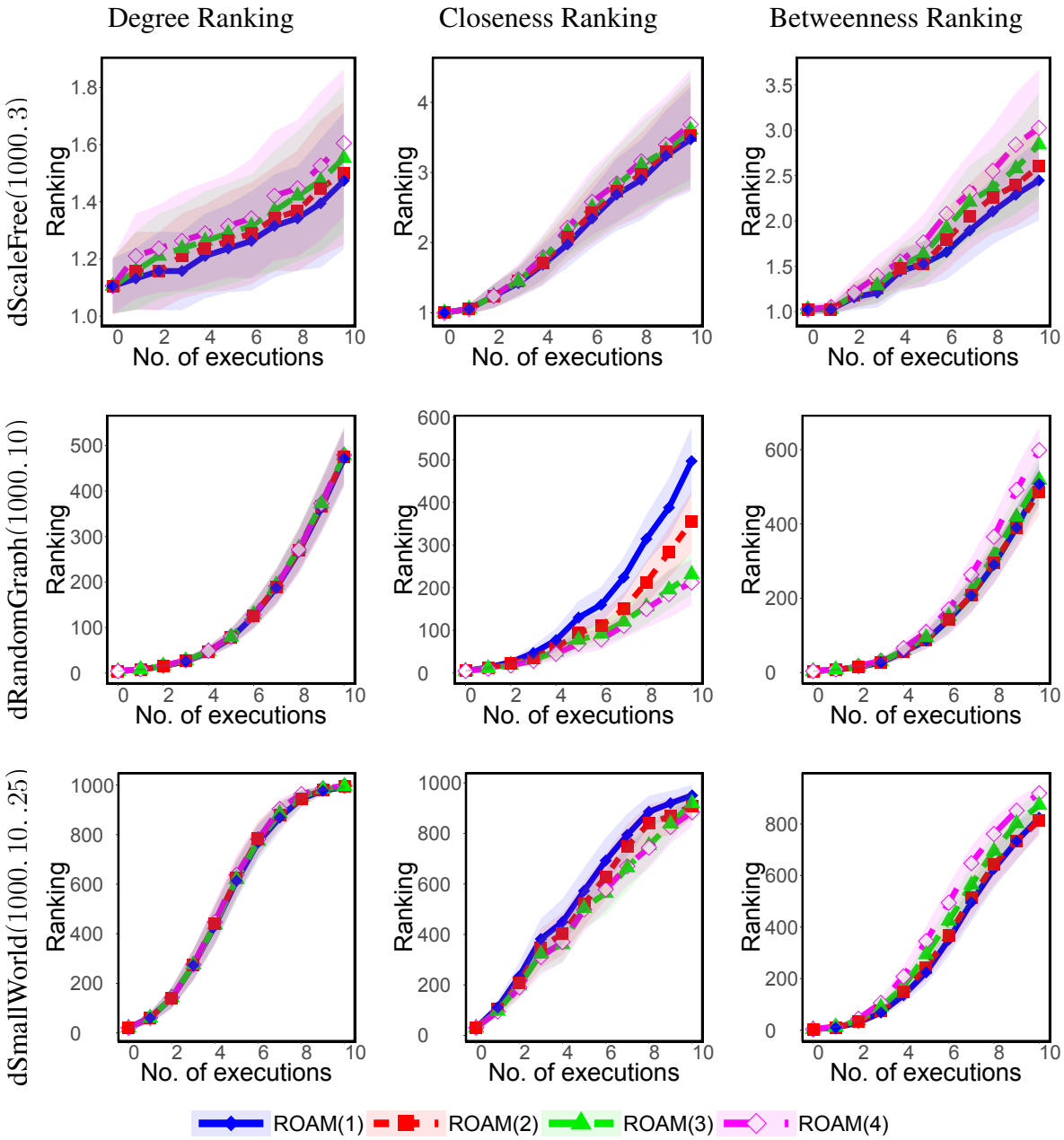
Supplementary Figure 11: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



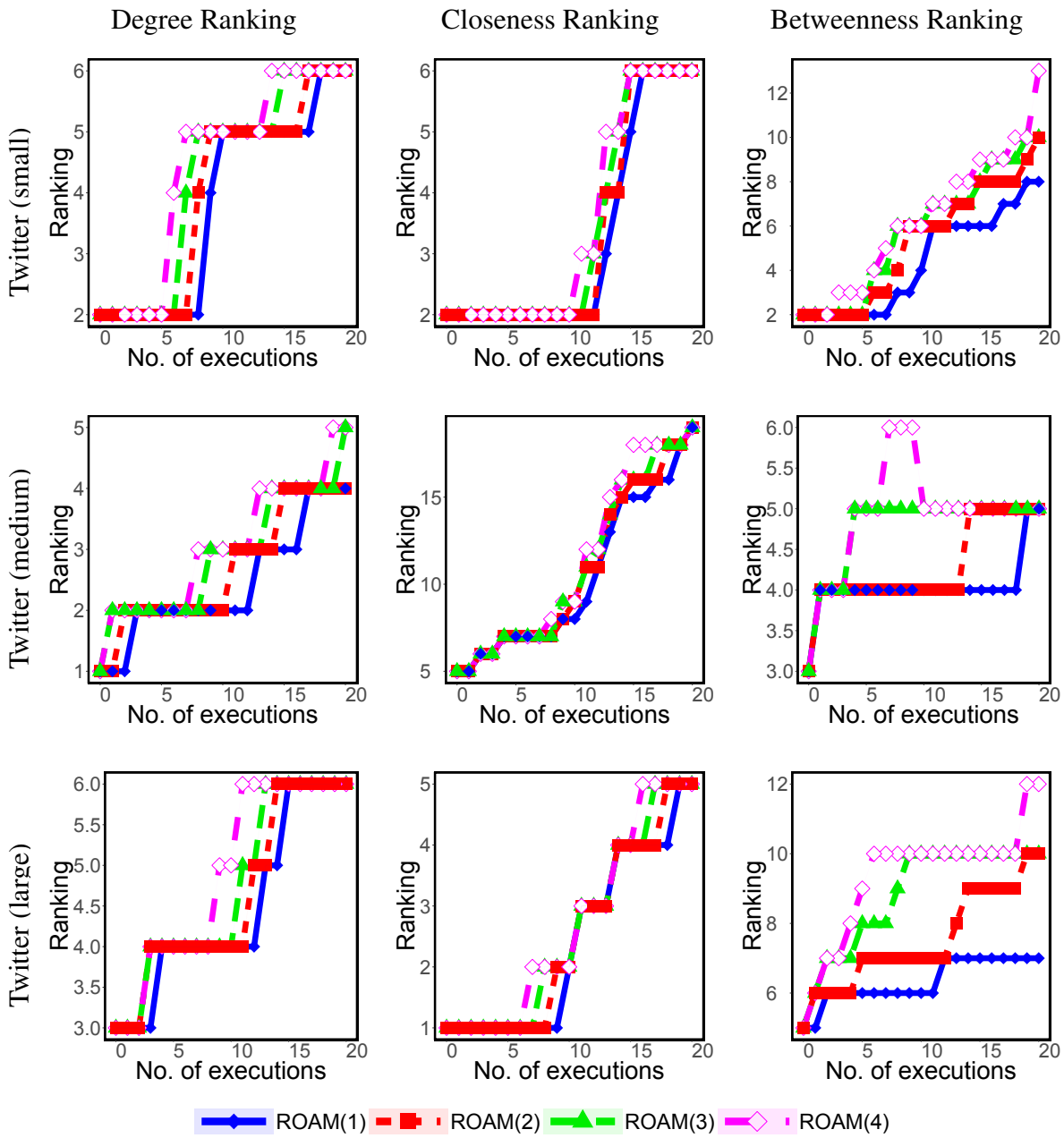
Supplementary Figure 12: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



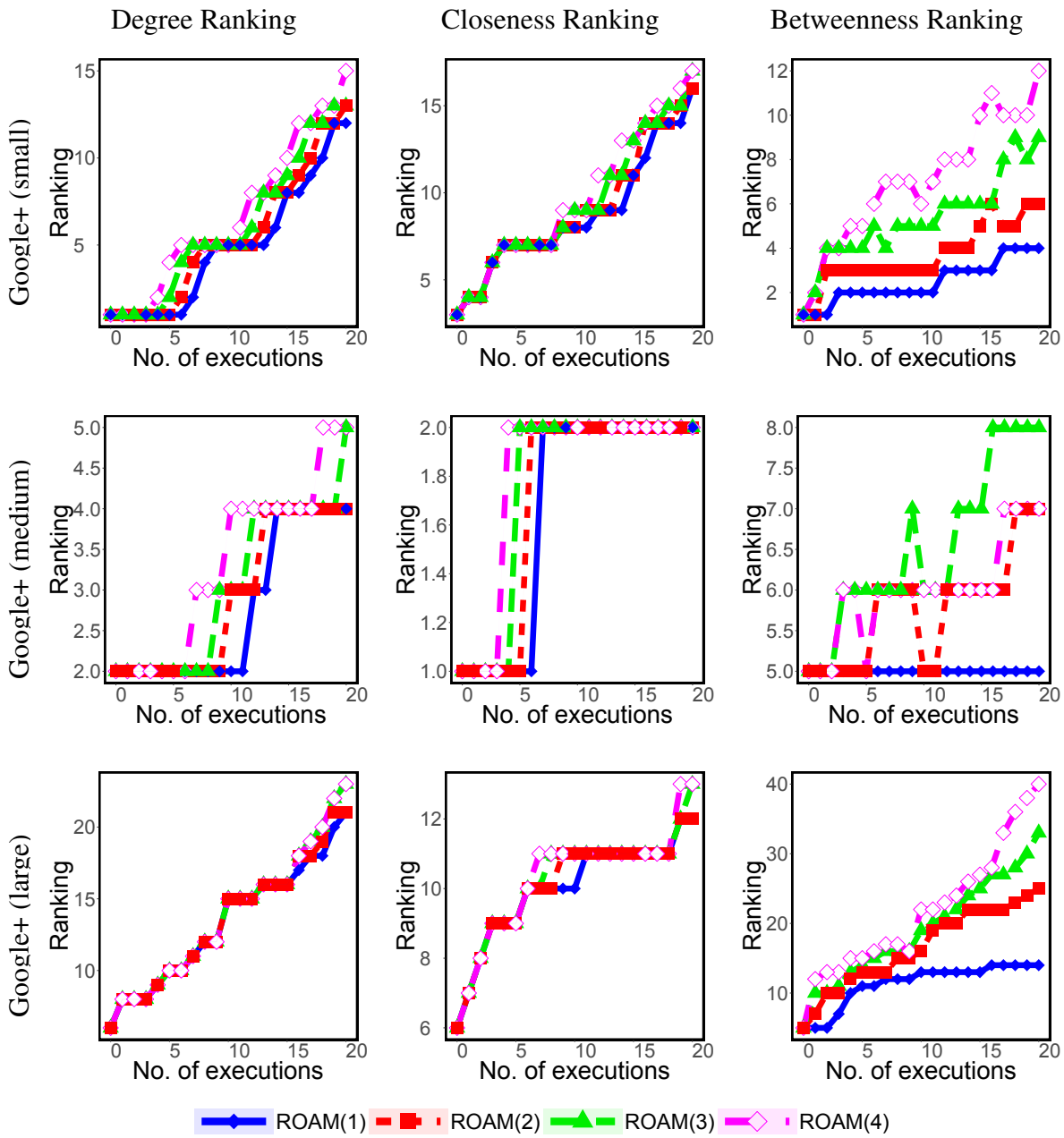
Supplementary Figure 13: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



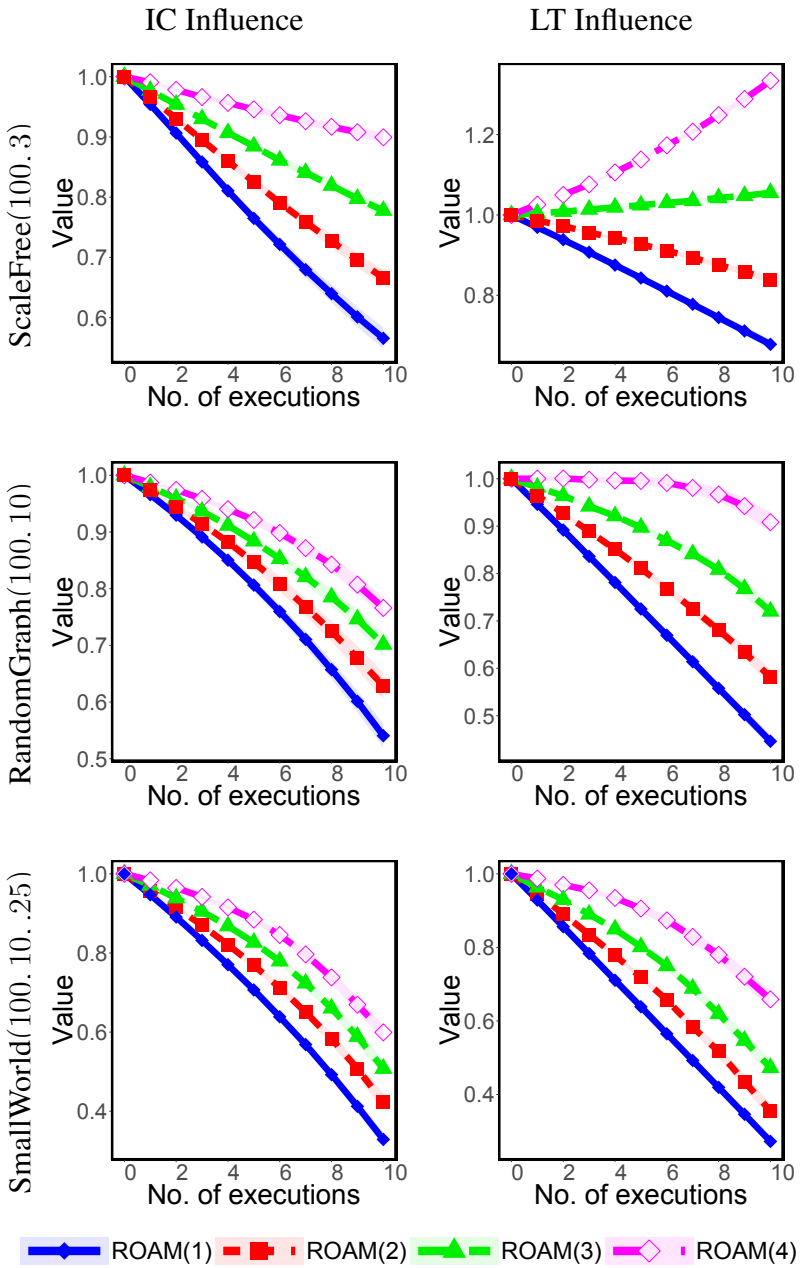
Supplementary Figure 14: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



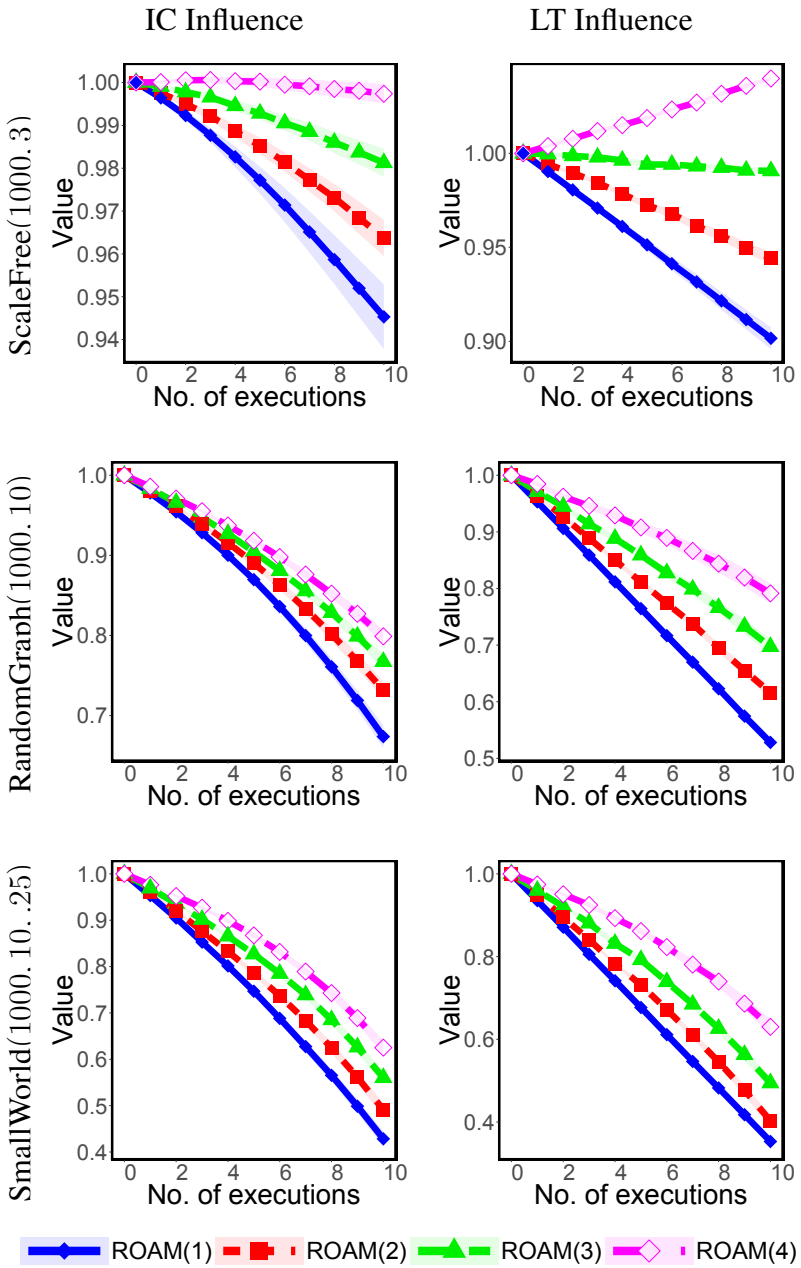
Supplementary Figure 15: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



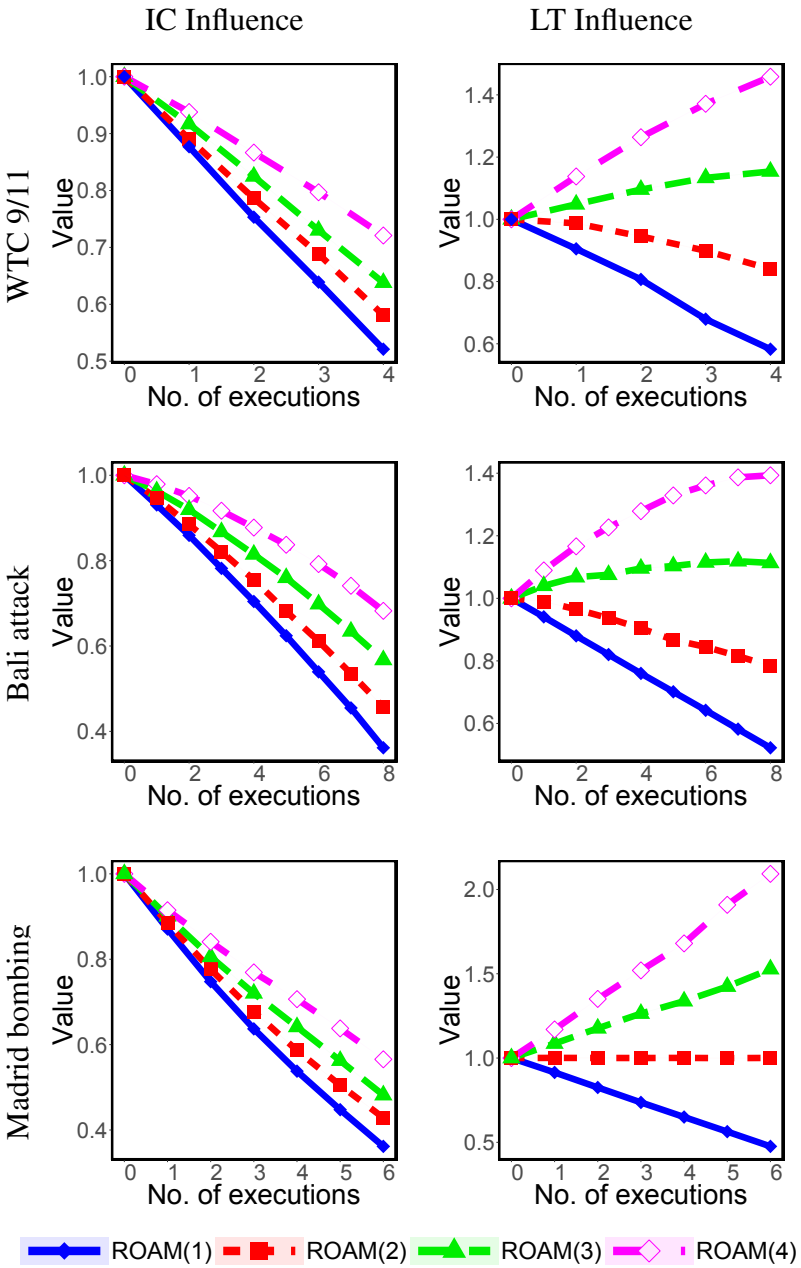
Supplementary Figure 16: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



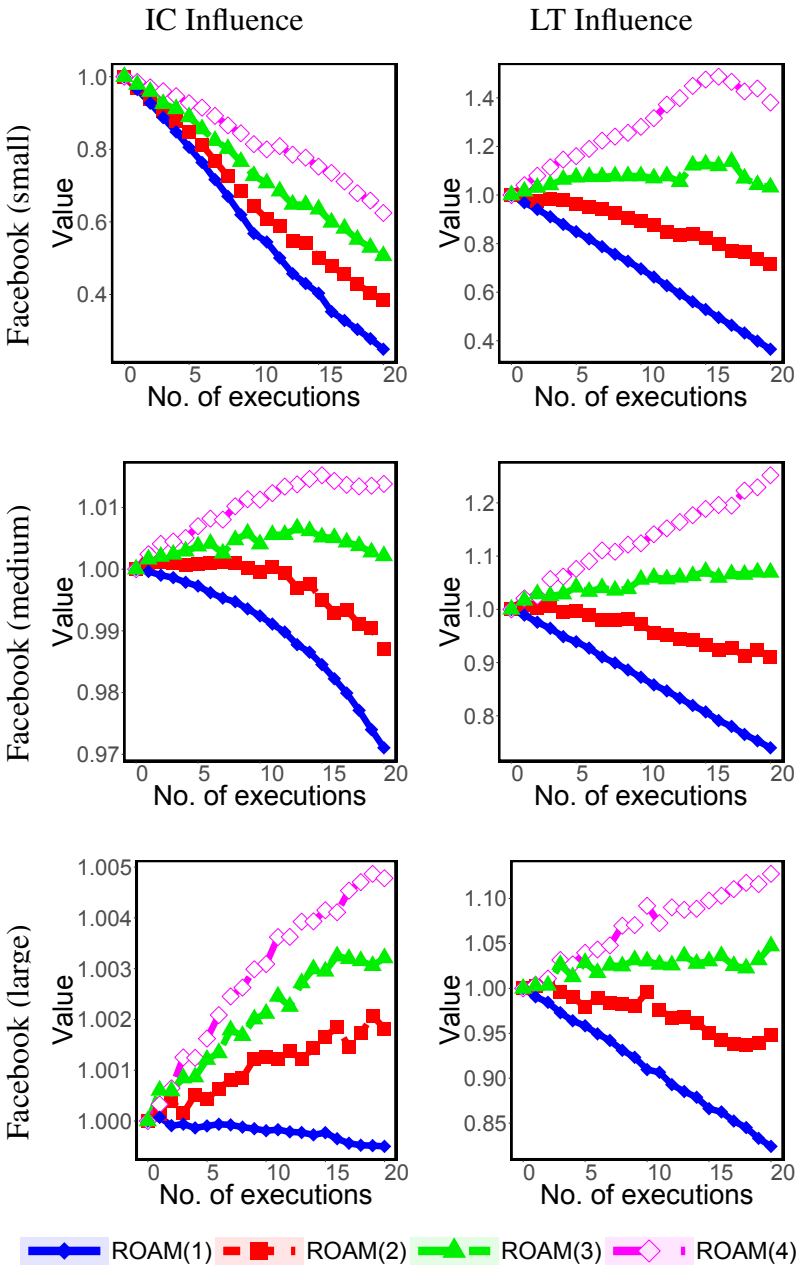
Supplementary Figure 17: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



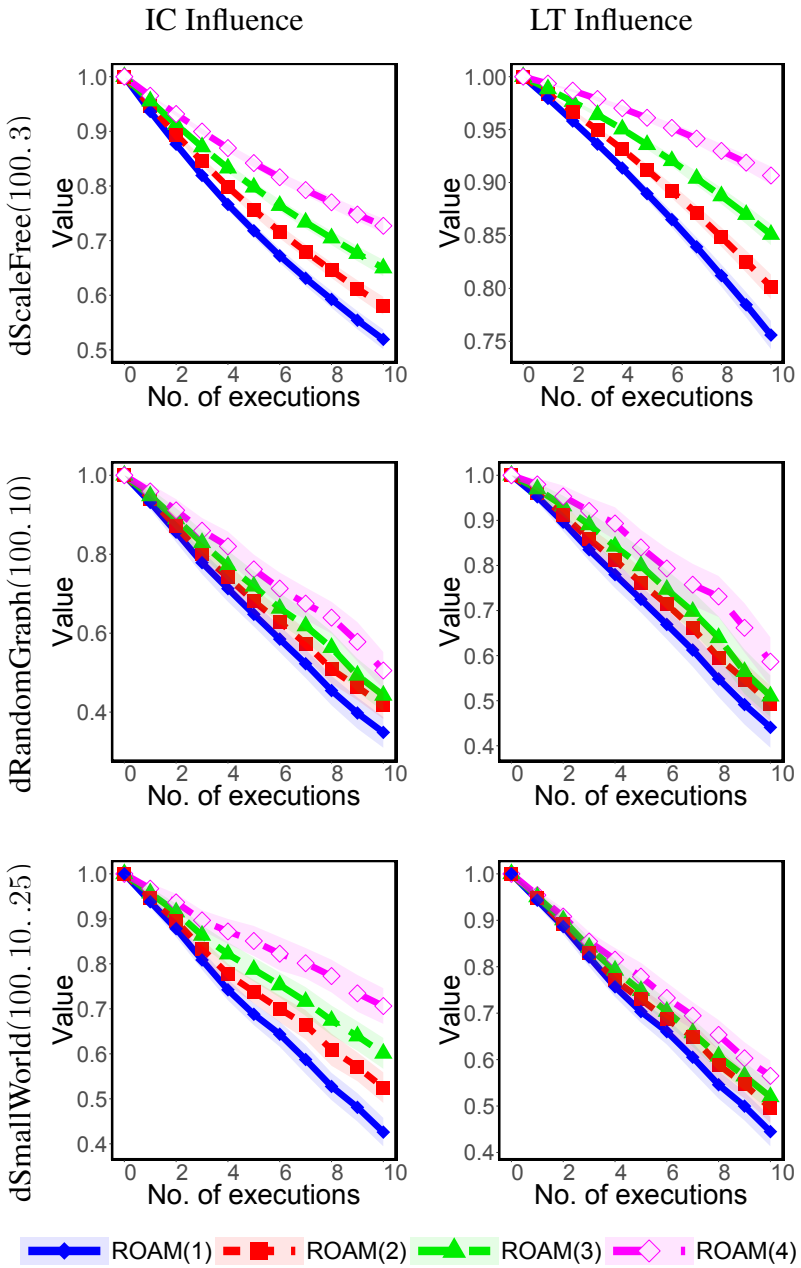
Supplementary Figure 18: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



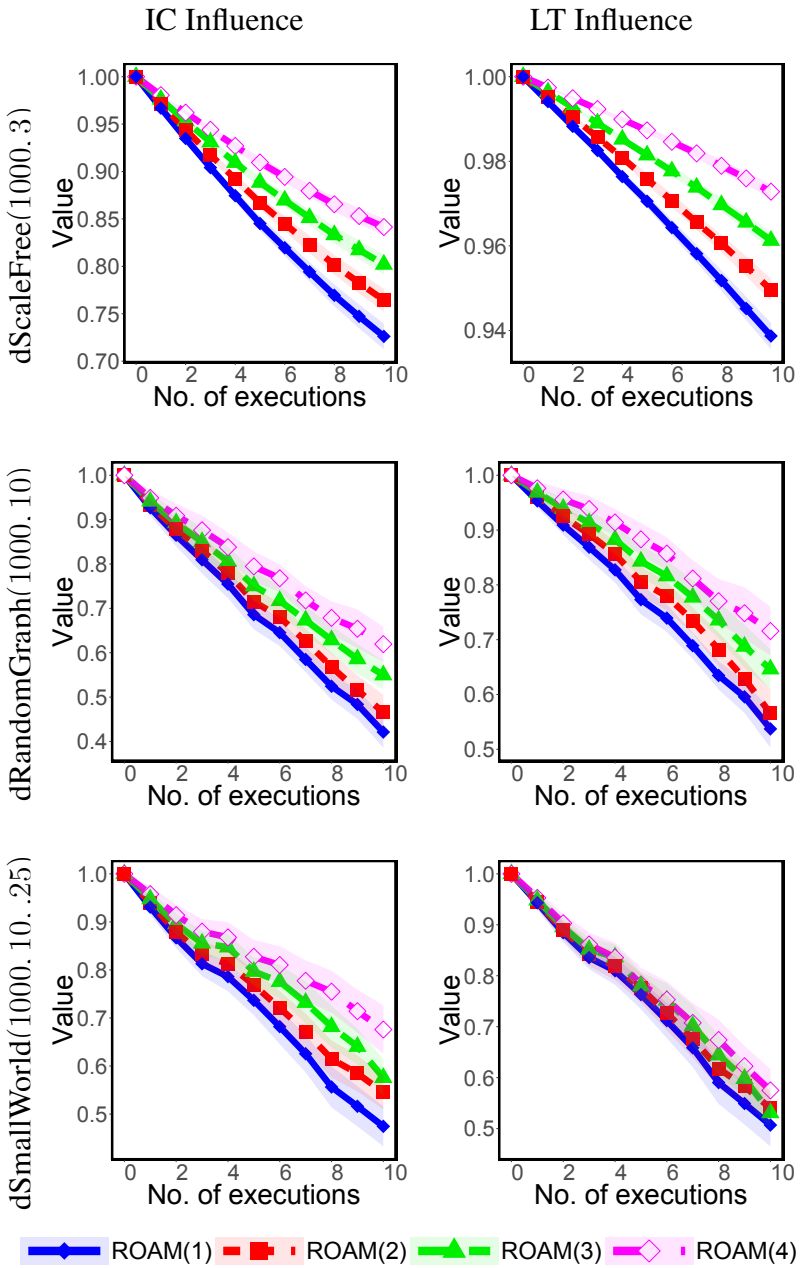
Supplementary Figure 19: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



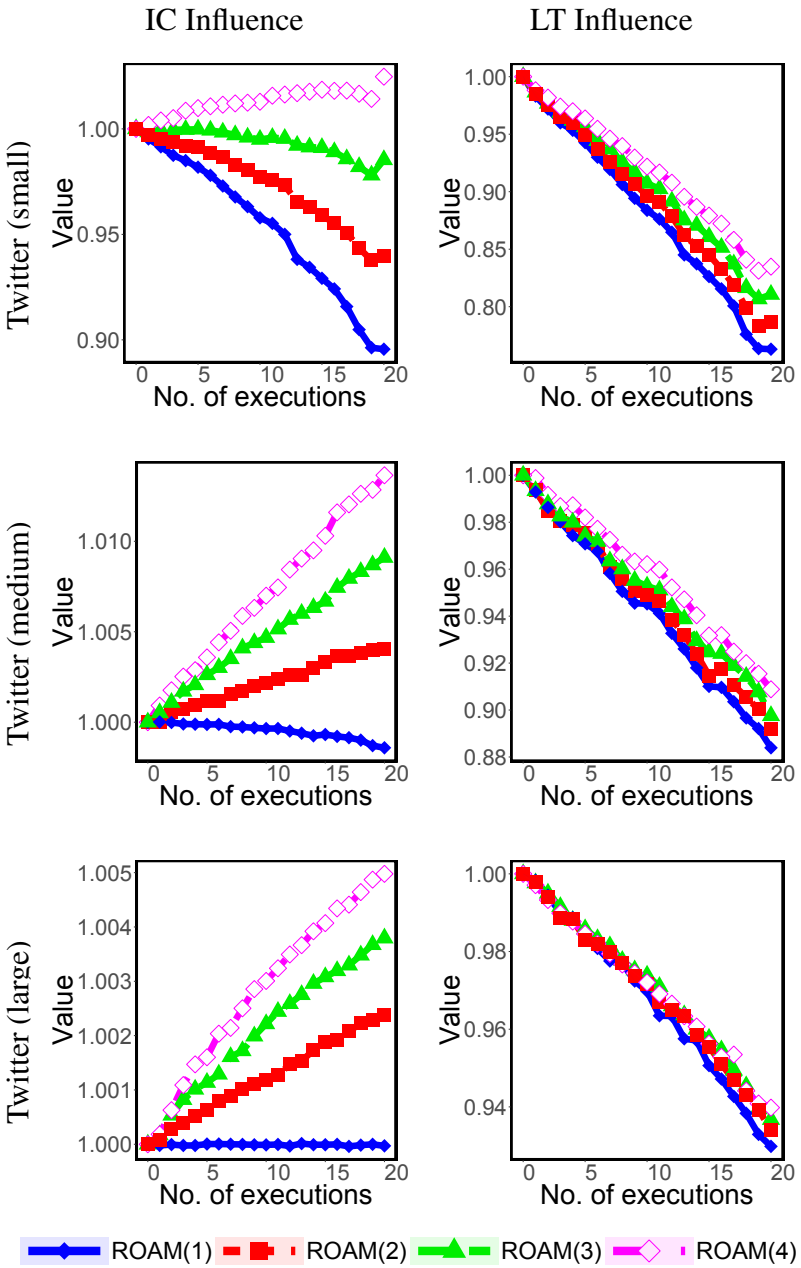
Supplementary Figure 20: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



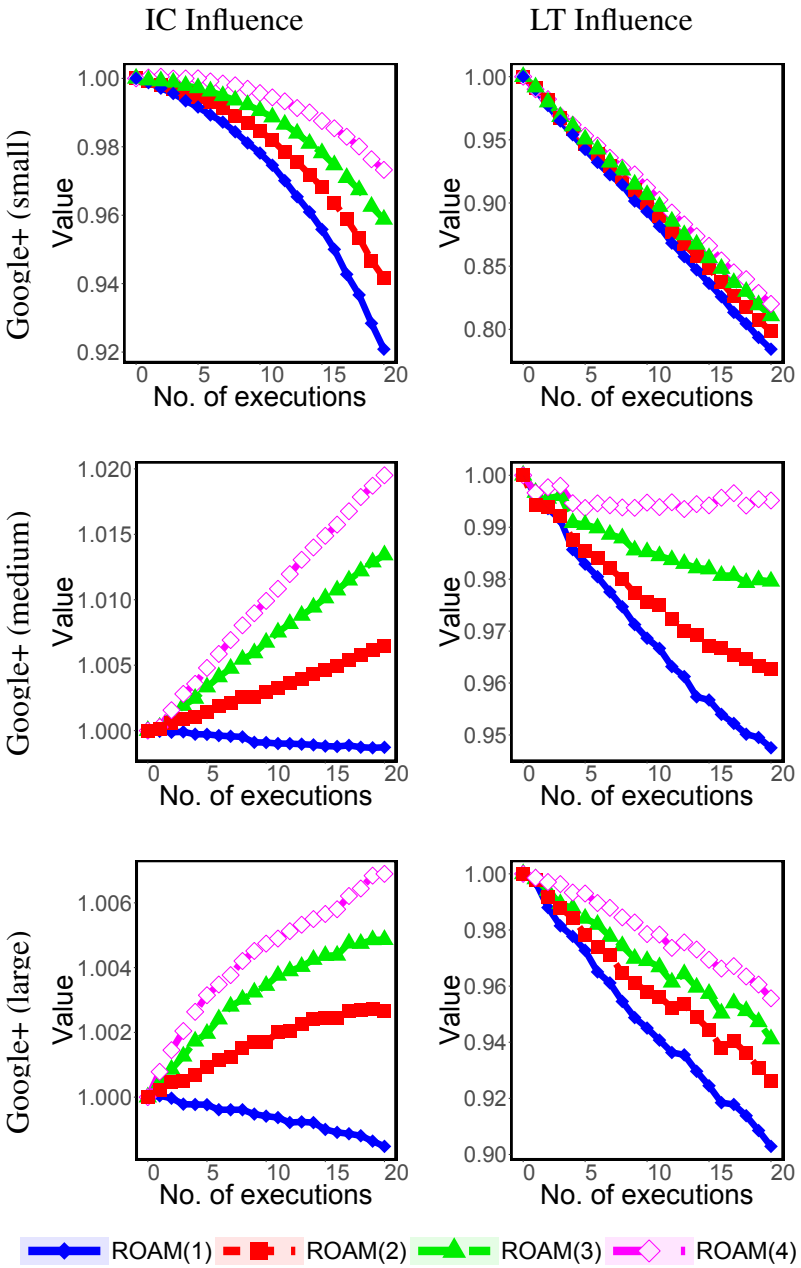
Supplementary Figure 21: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



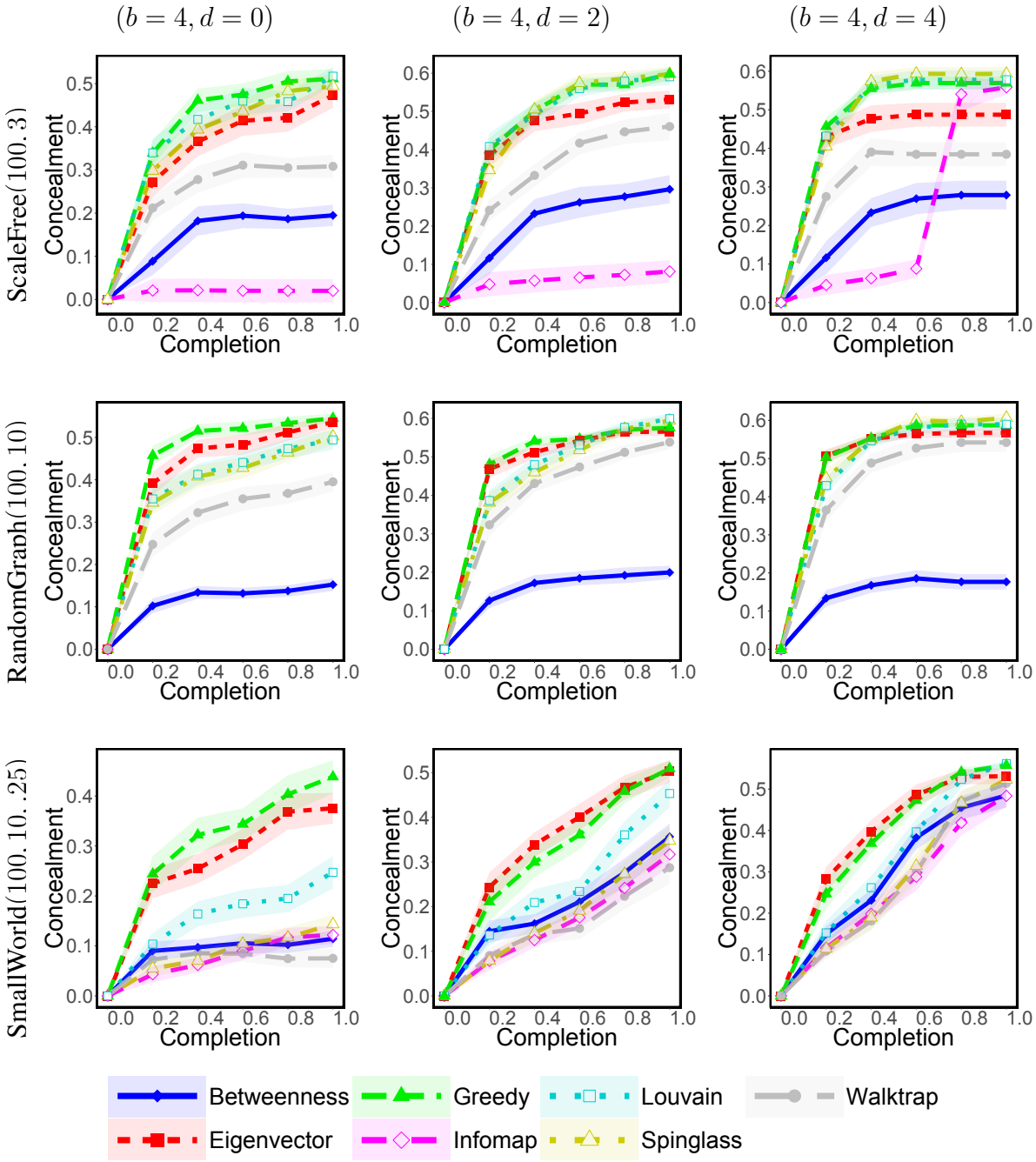
Supplementary Figure 22: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



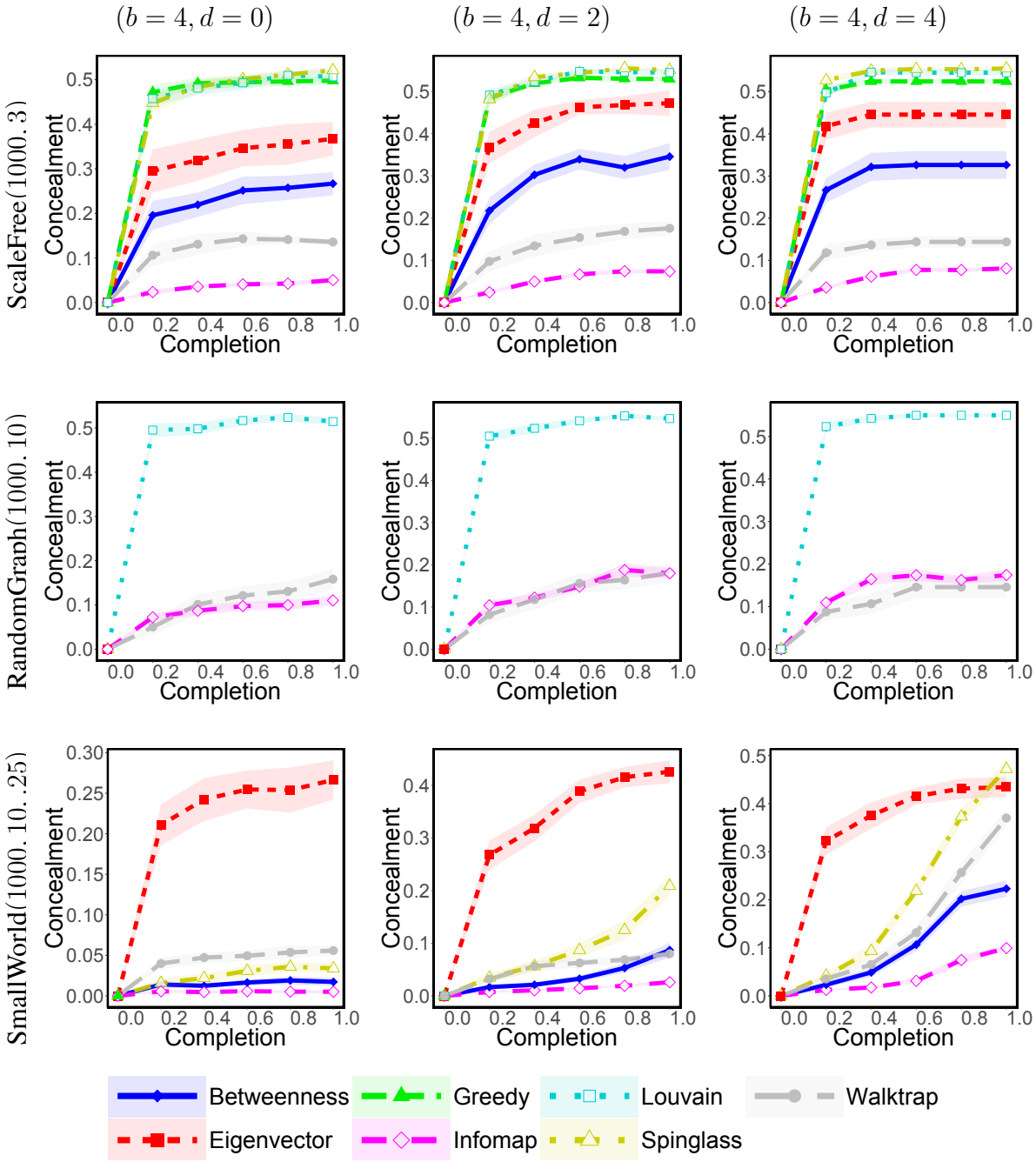
Supplementary Figure 23: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



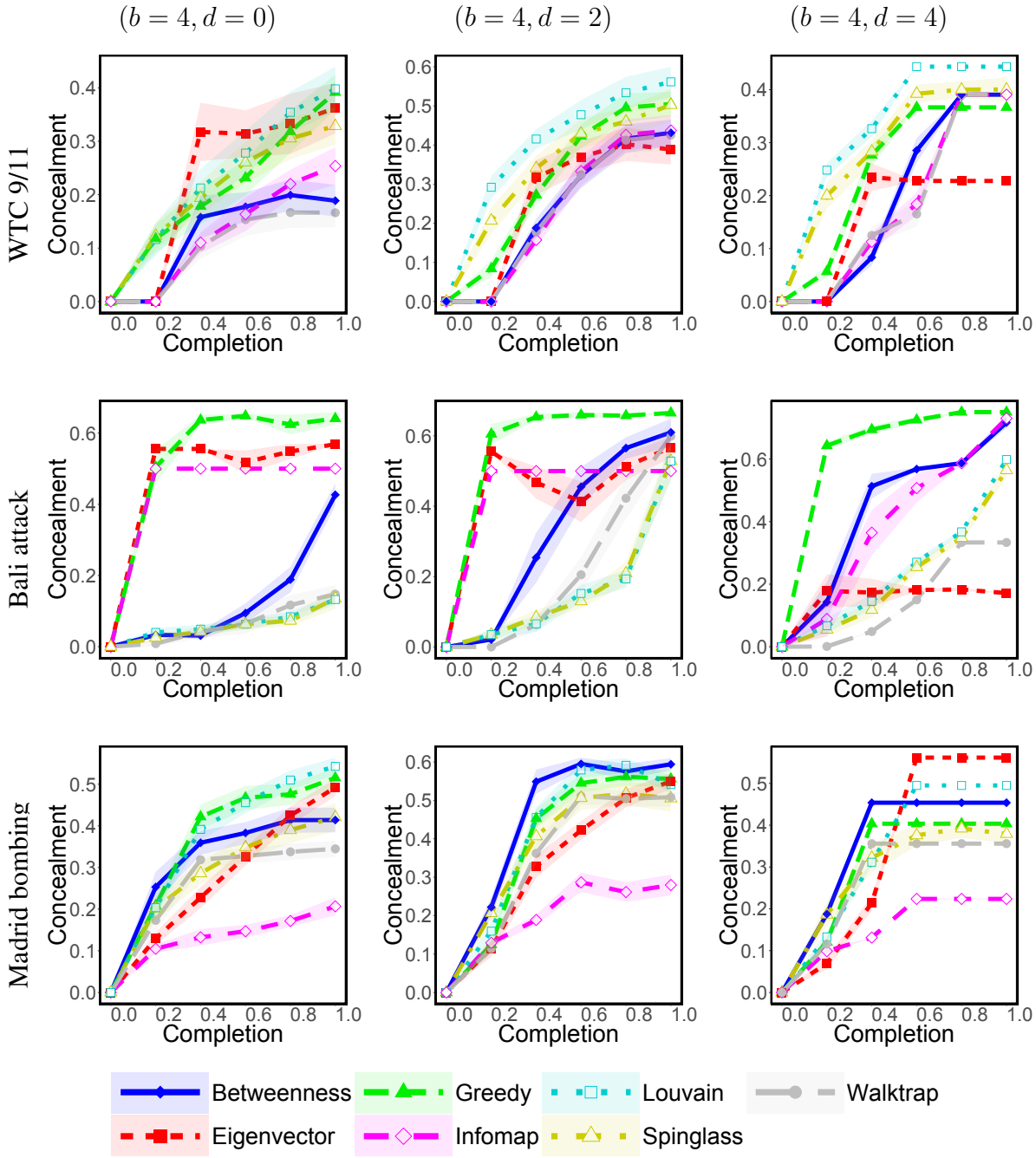
Supplementary Figure 24: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



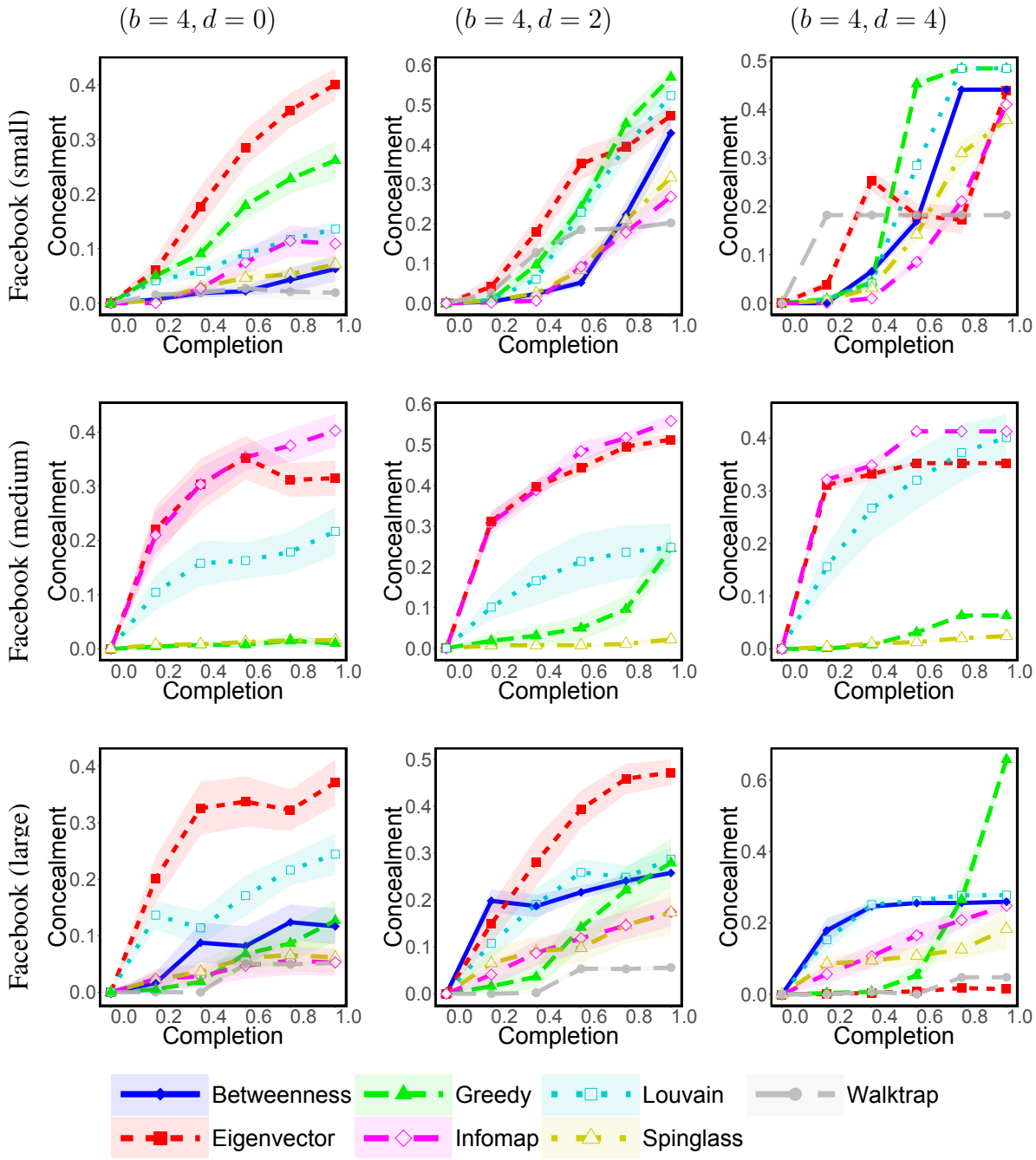
Supplementary Figure 25: Consecutive execution of DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE$(b, d) : b = 4; d \in \{0, 2, 4\}$, where b is the budget in each execution, and d is the number of removed internal edges.



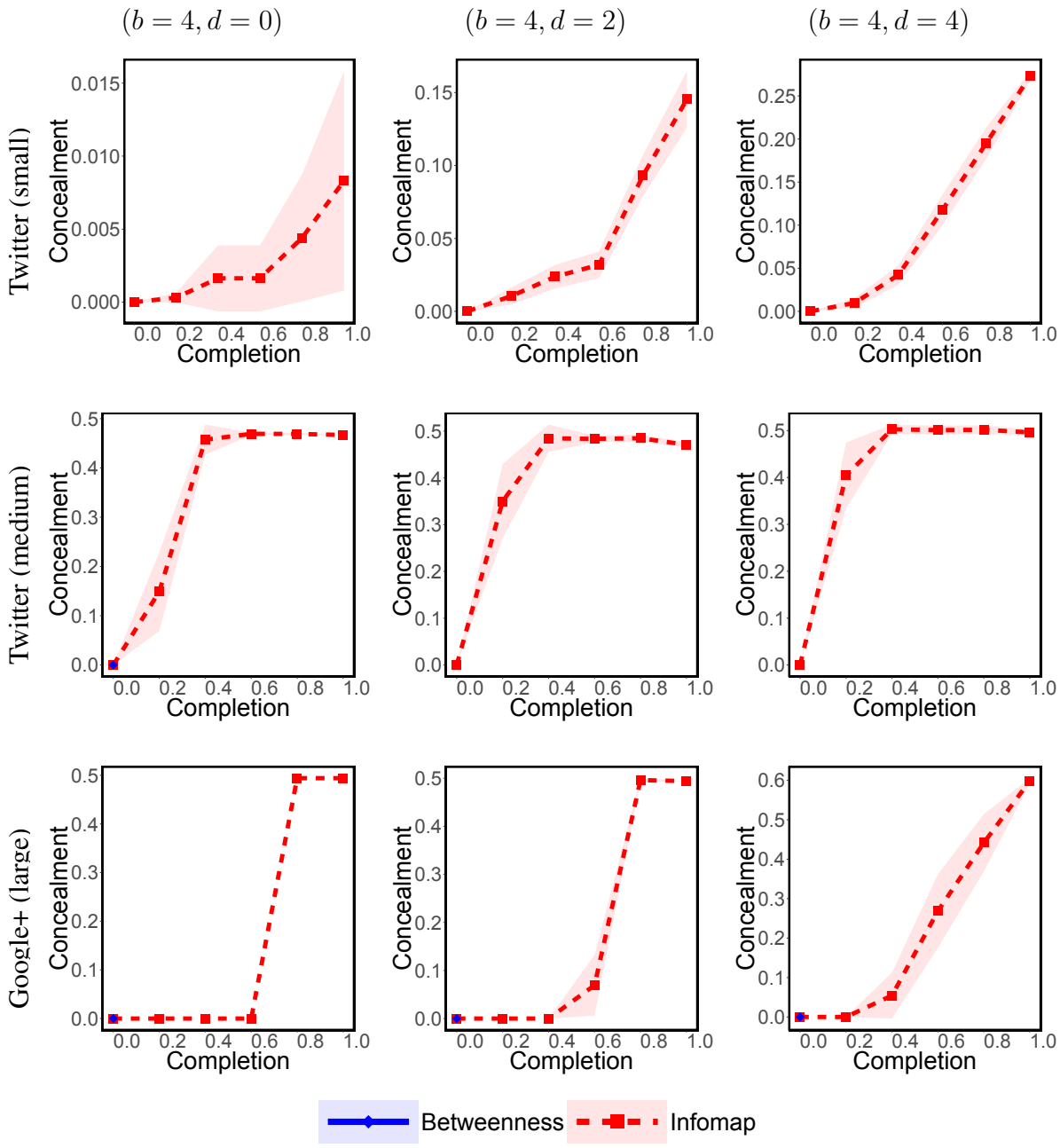
Supplementary Figure 26: Consecutive execution of DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE$(b, d) : b = 4; d \in \{0, 2, 4\}$, where b is the budget in each execution, and d is the number of removed internal edges.



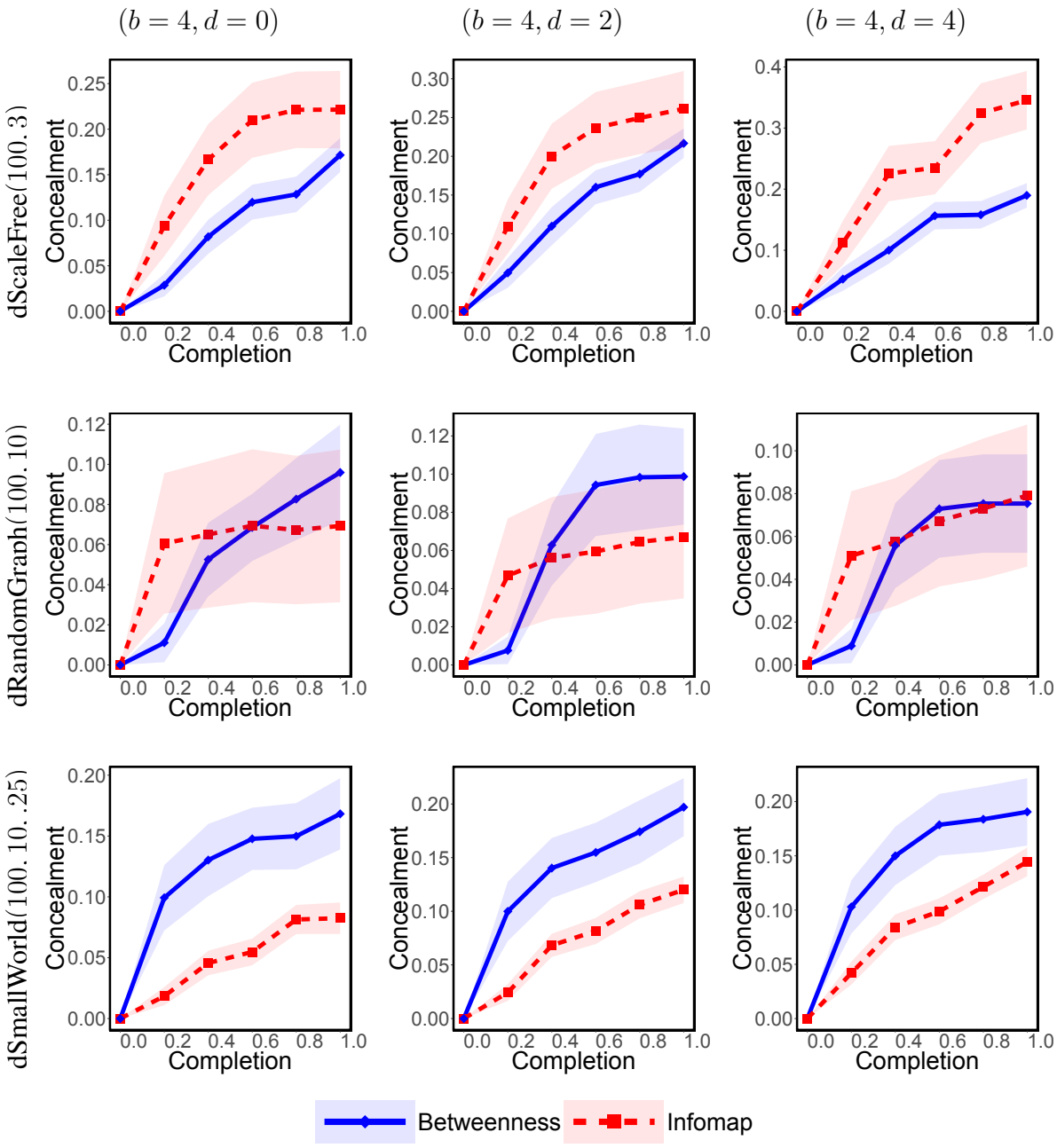
Supplementary Figure 27: Consecutive execution of DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE$(b, d) : b = 4; d \in \{0, 2, 4\}$, where b is the budget in each execution, and d is the number of removed internal edges.



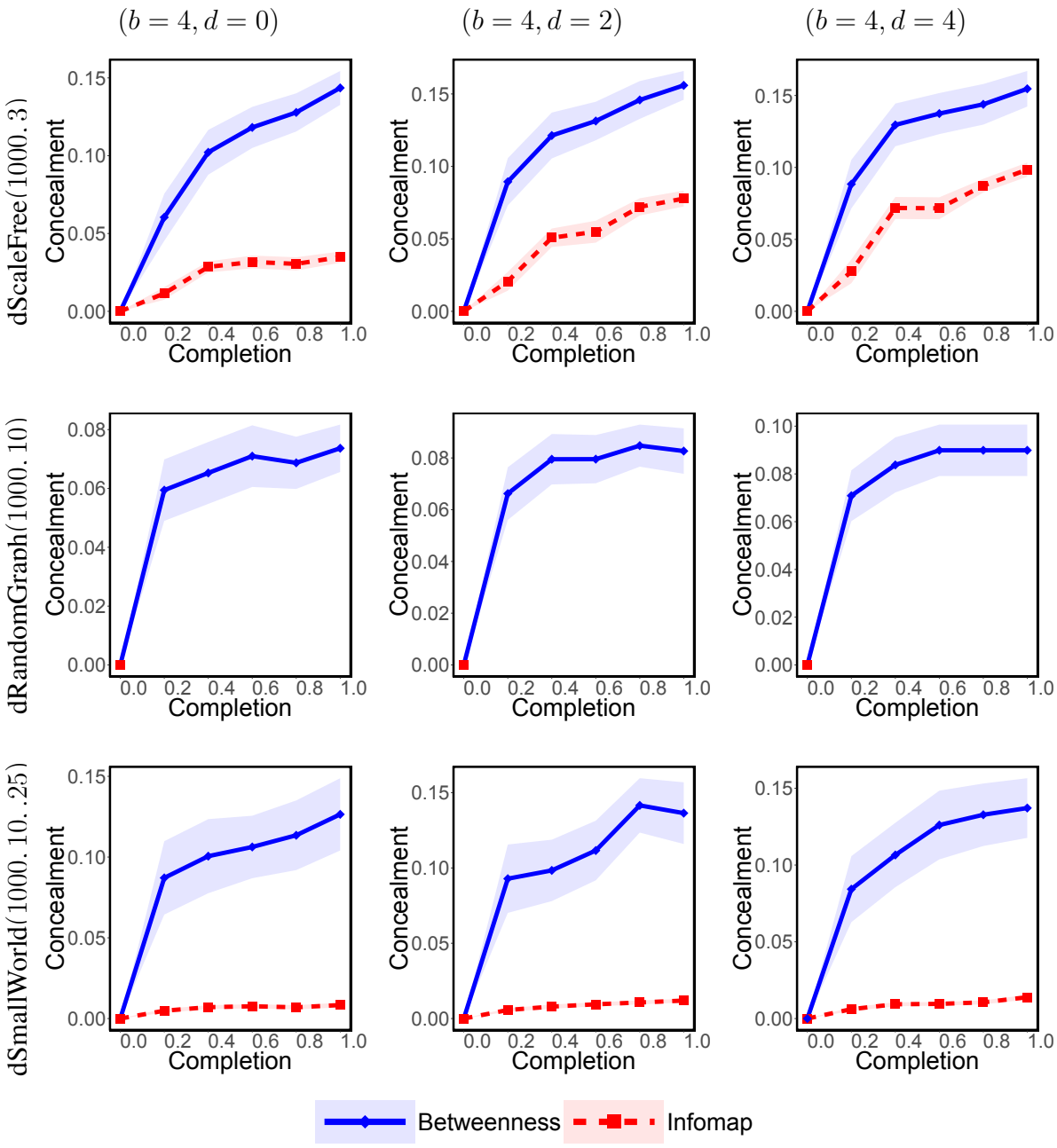
Supplementary Figure 28: Consecutive execution of DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE$(b, d) : b = 4; d \in \{0, 2, 4\}$, where b is the budget in each execution, and d is the number of removed internal edges.



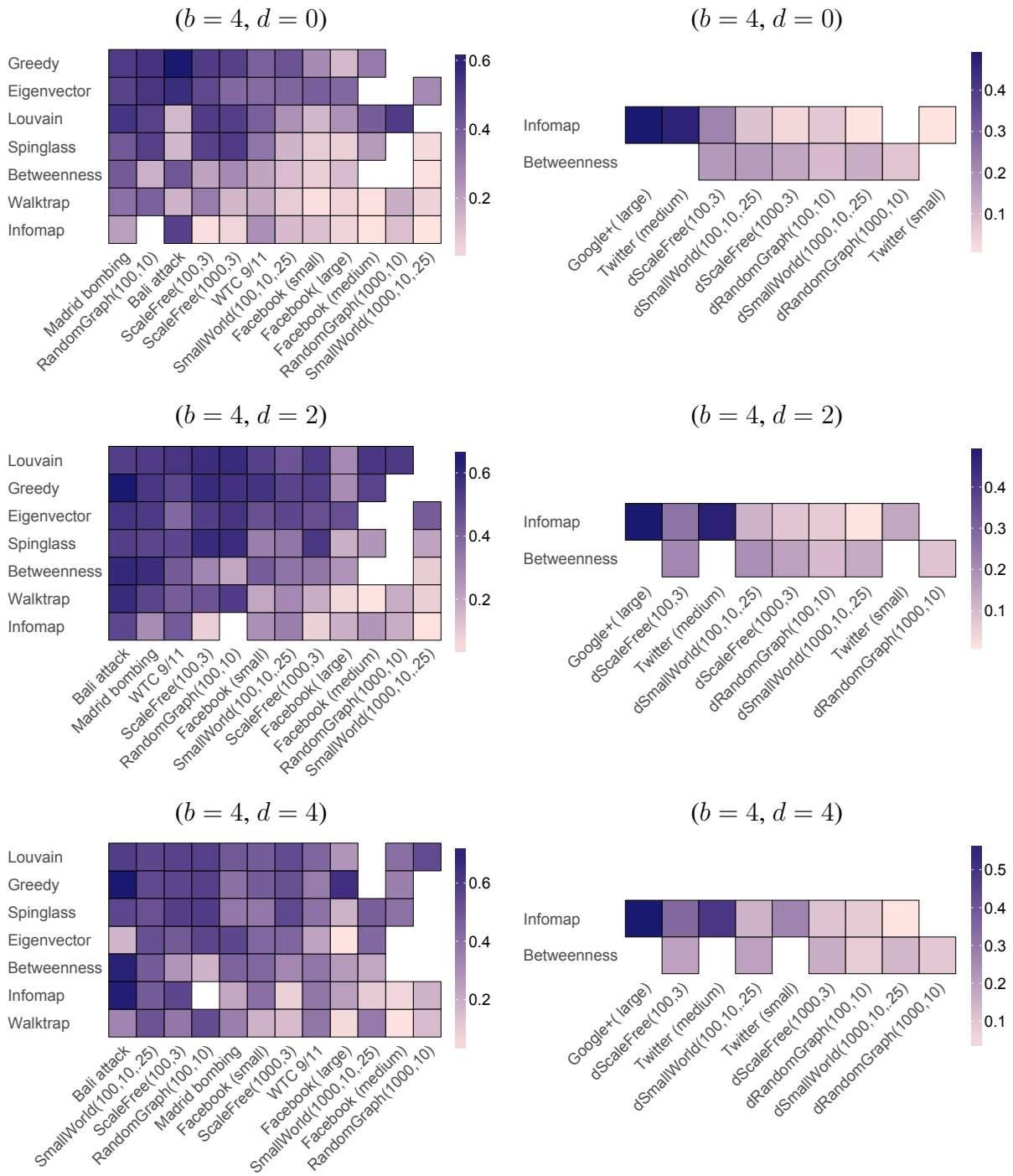
Supplementary Figure 29: Consecutive execution of DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE$(b, d) : b = 4; d \in \{0, 2, 4\}$, where b is the budget in each execution, and d is the number of removed internal edges.



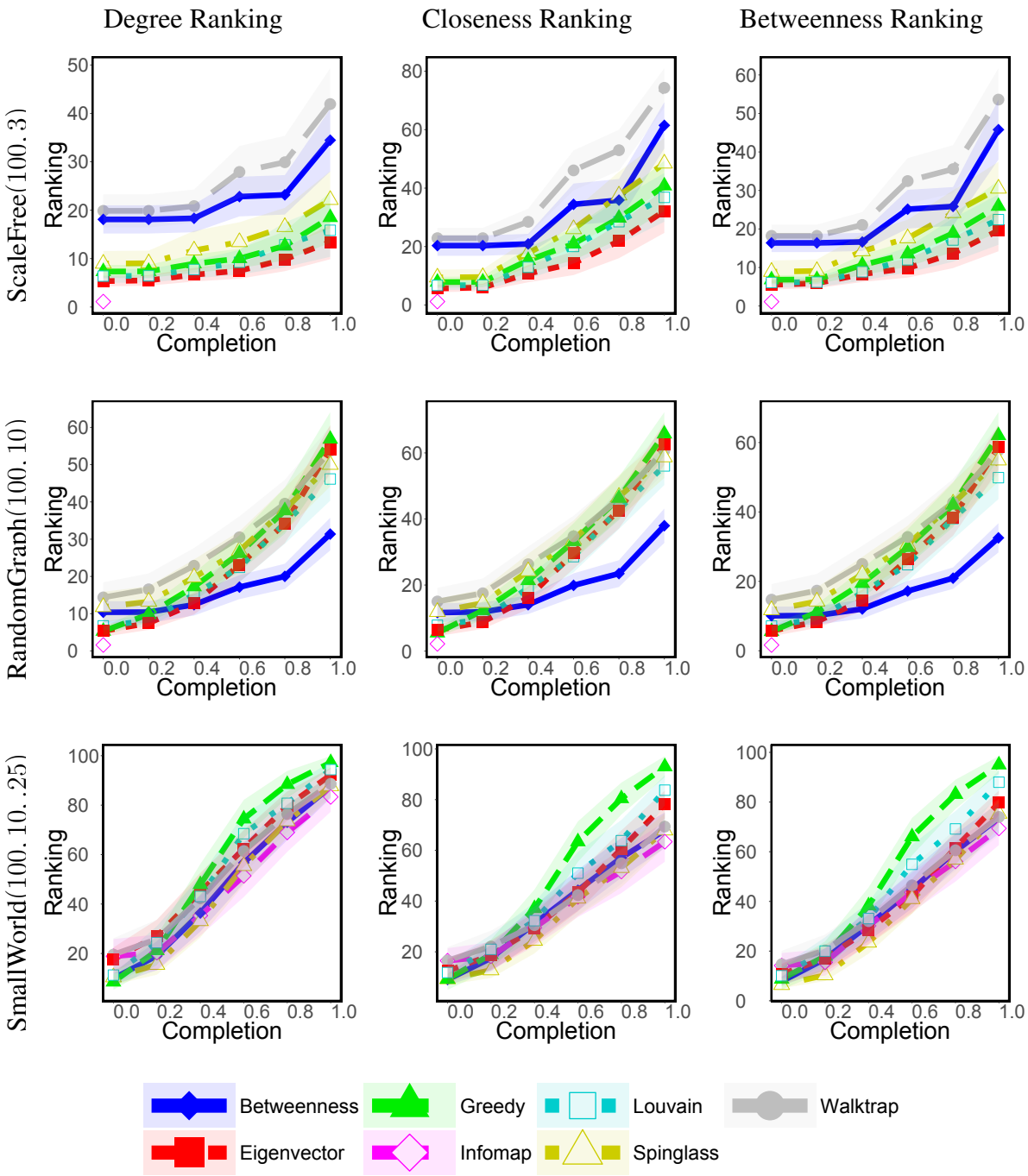
Supplementary Figure 30: Consecutive execution of DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE$(b, d) : b = 4; d \in \{0, 2, 4\}$, where b is the budget in each execution, and d is the number of removed internal edges.



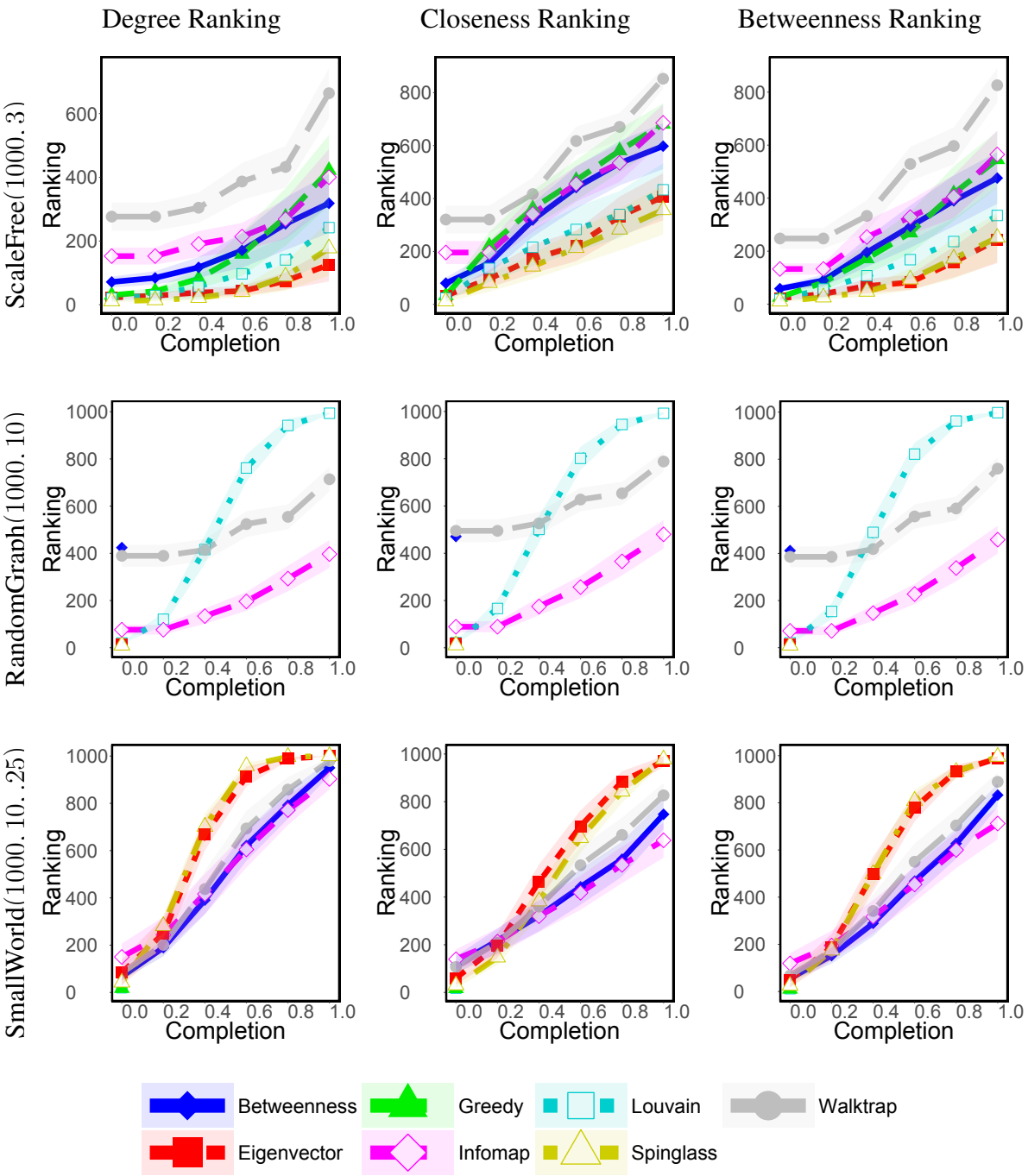
Supplementary Figure 31: Consecutive execution of DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE$(b, d) : b = 4; d \in \{0, 2, 4\}$, where b is the budget in each execution, and d is the number of removed internal edges.



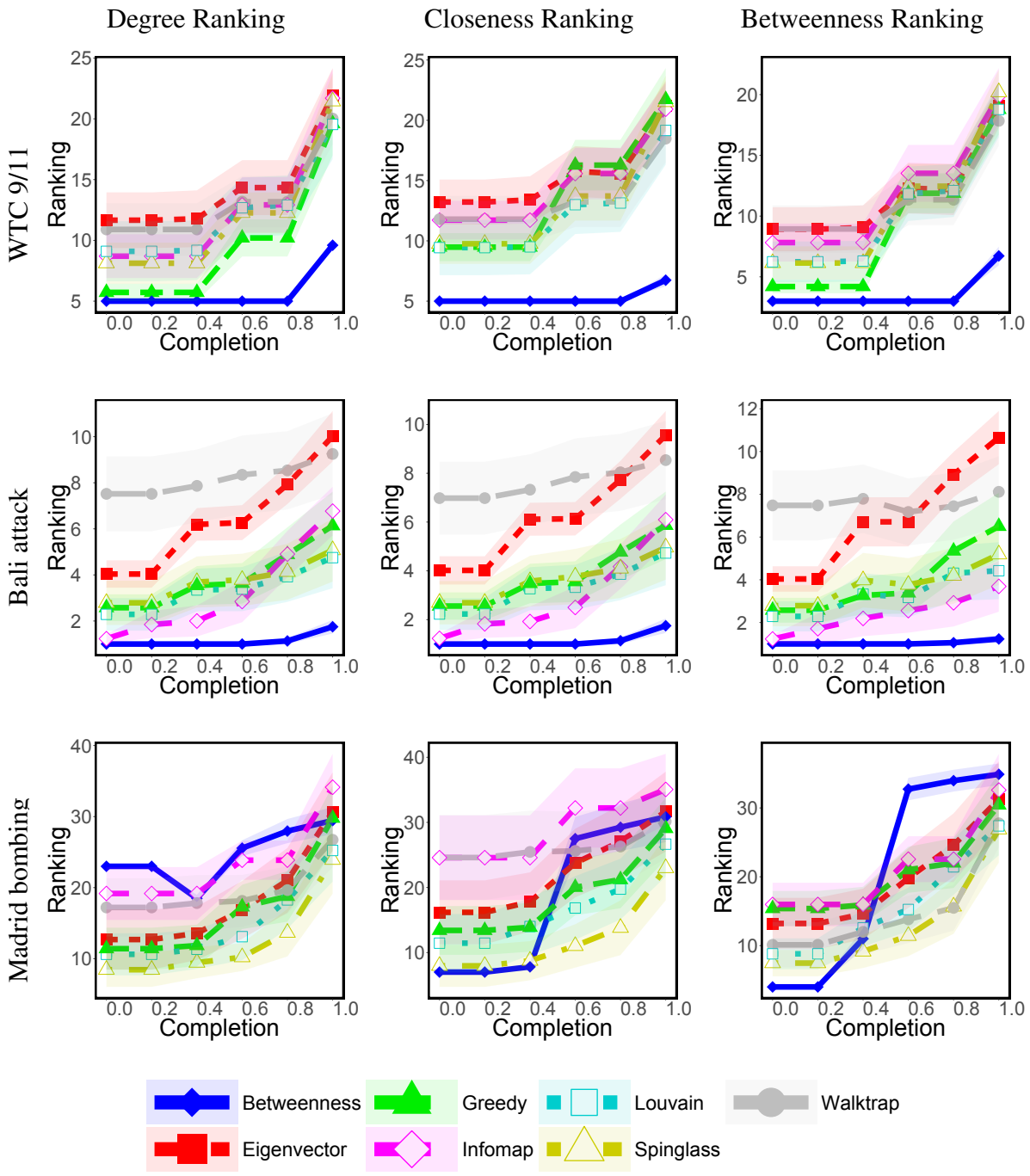
Supplementary Figure 32: Average concealment-measure value in each experiment. The results for the directed networks (namely the fragments of Twitter and Google+) are presented on the right, whereas the results for the undirected networks are presented on the left. Results are shown for DICE$(b, d) : b = 4; d \in \{0, 2, 4\}$, where b is the budget in each execution, and d is the number of removed internal edges.



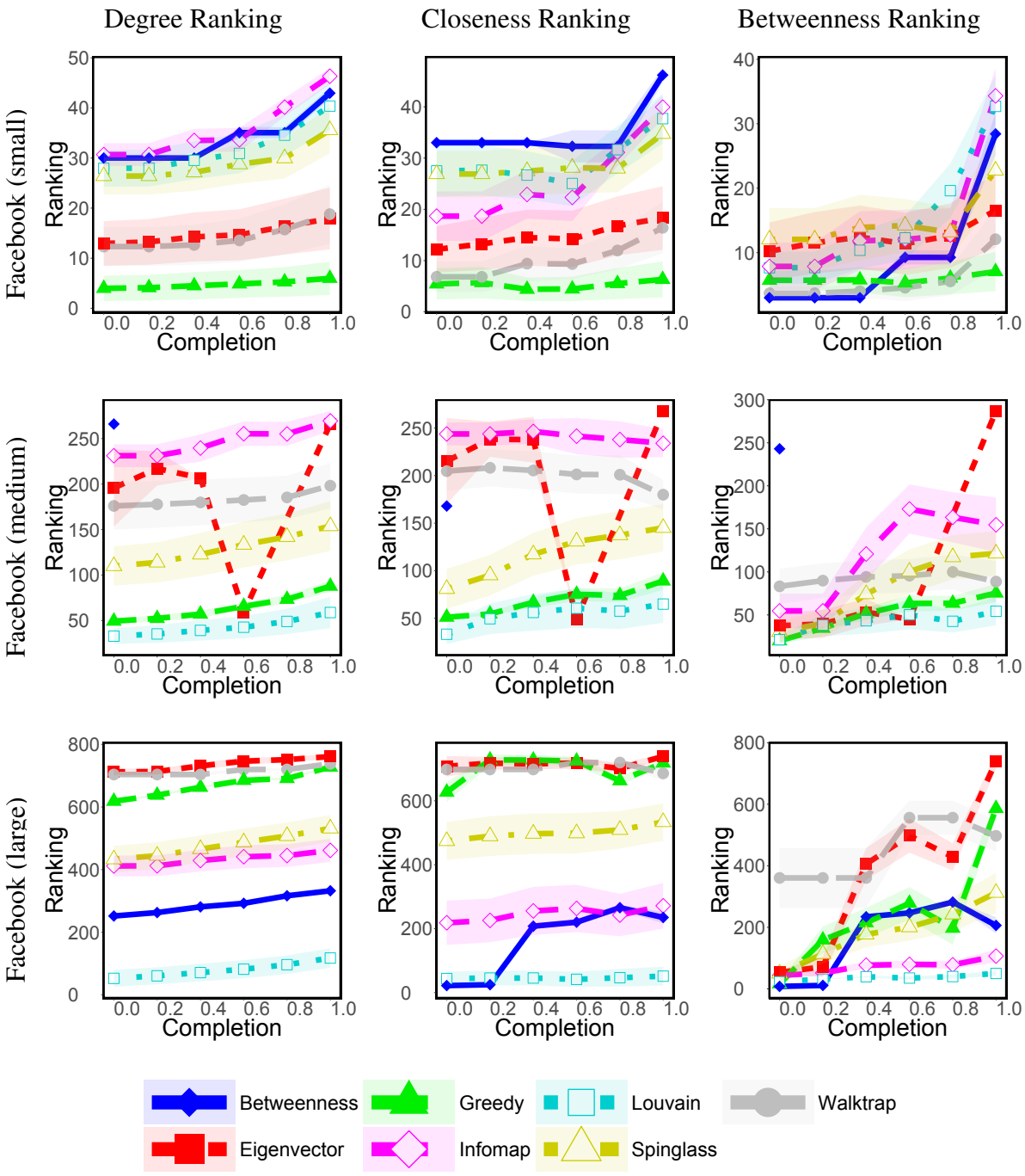
Supplementary Figure 33: Consecutive execution of both ROAM and DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for DICE(b, d) : $b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM(b) : $b = 3$, where b is the budget in each execution.



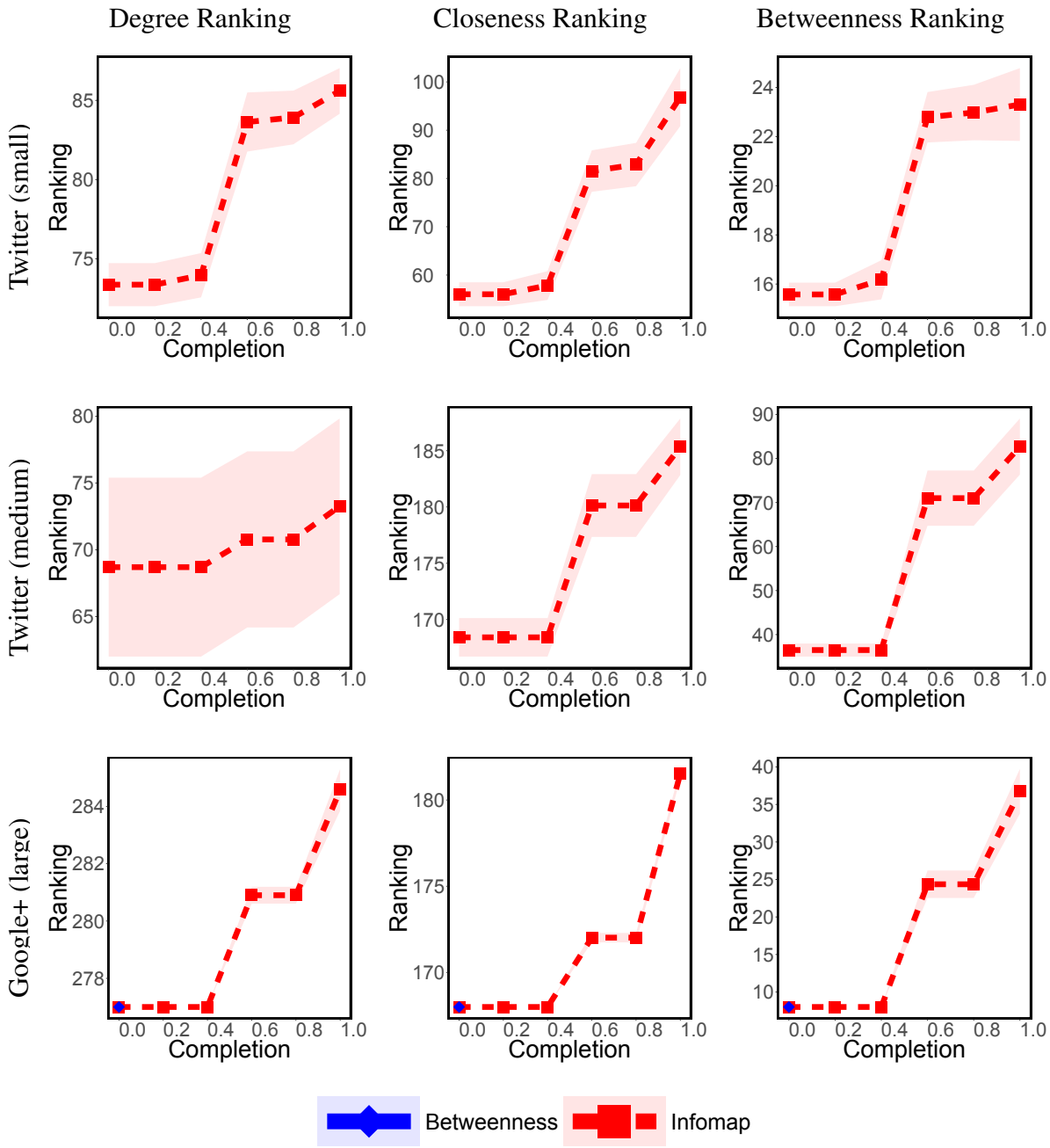
Supplementary Figure 34: Consecutive execution of both ROAM and DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for DICE(b, d) : $b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM(b) : $b = 3$, where b is the budget in each execution.



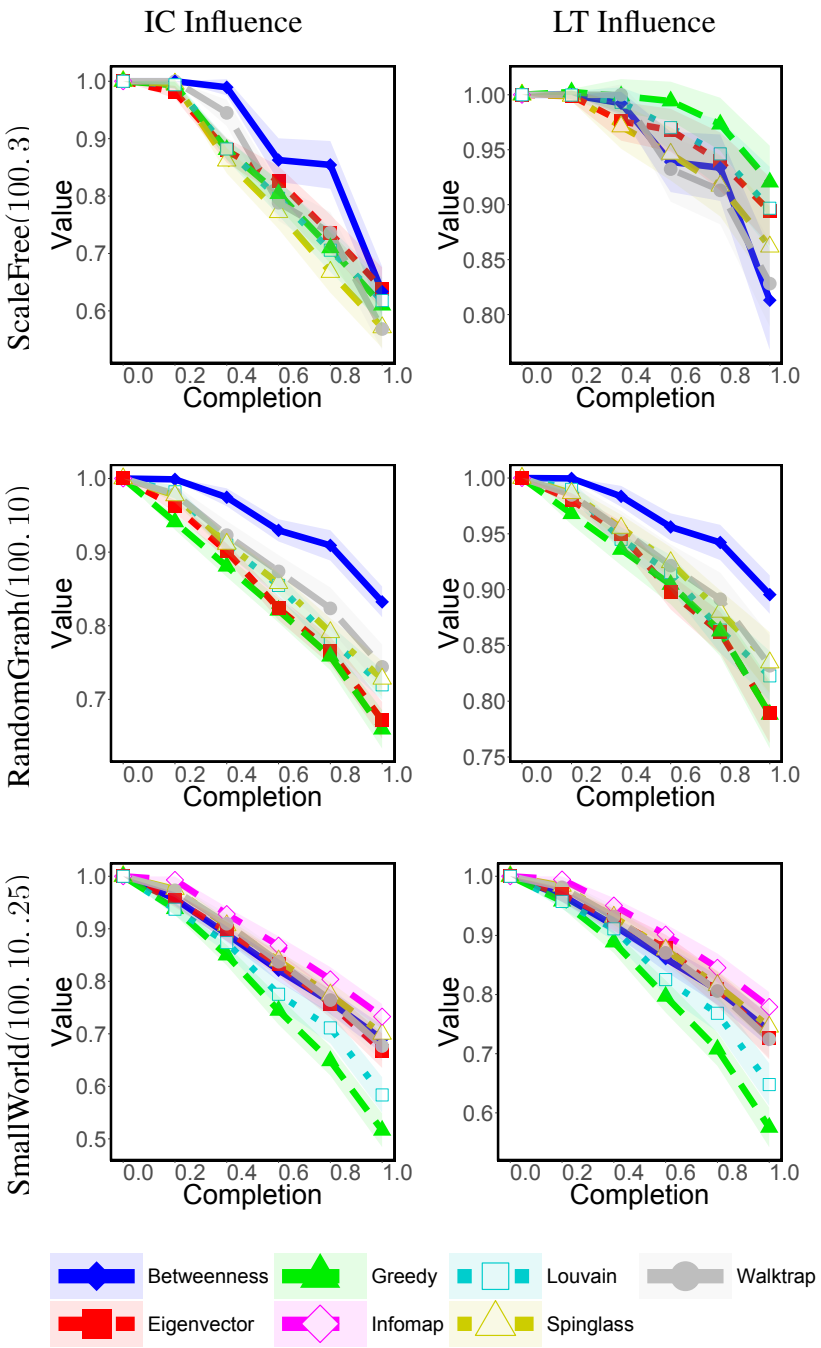
Supplementary Figure 35: Consecutive execution of both ROAM and DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for DICE(b, d) : $b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM(b) : $b = 3$, where b is the budget in each execution.



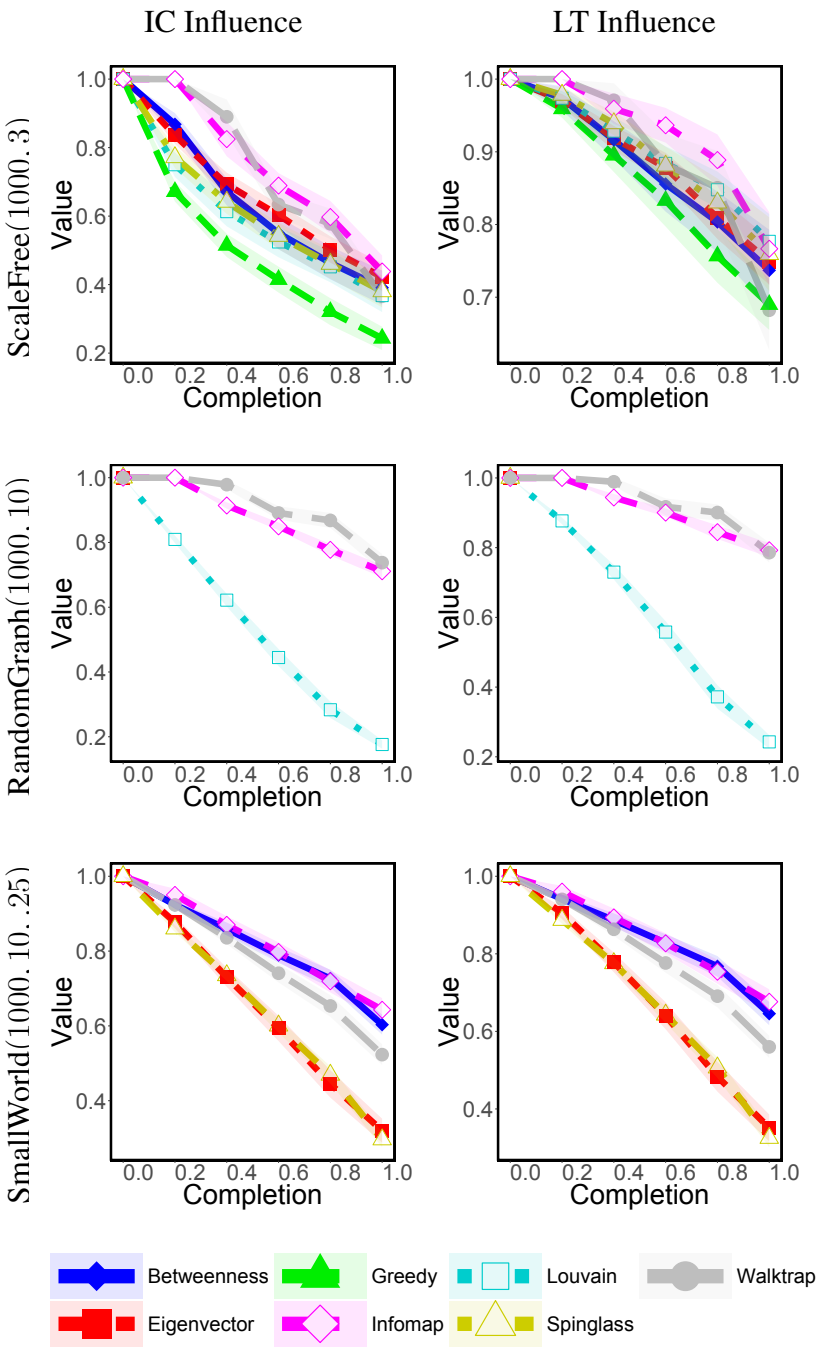
Supplementary Figure 36: Consecutive execution of both ROAM and DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for DICE(b, d) : $b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM(b) : $b = 3$, where b is the budget in each execution.



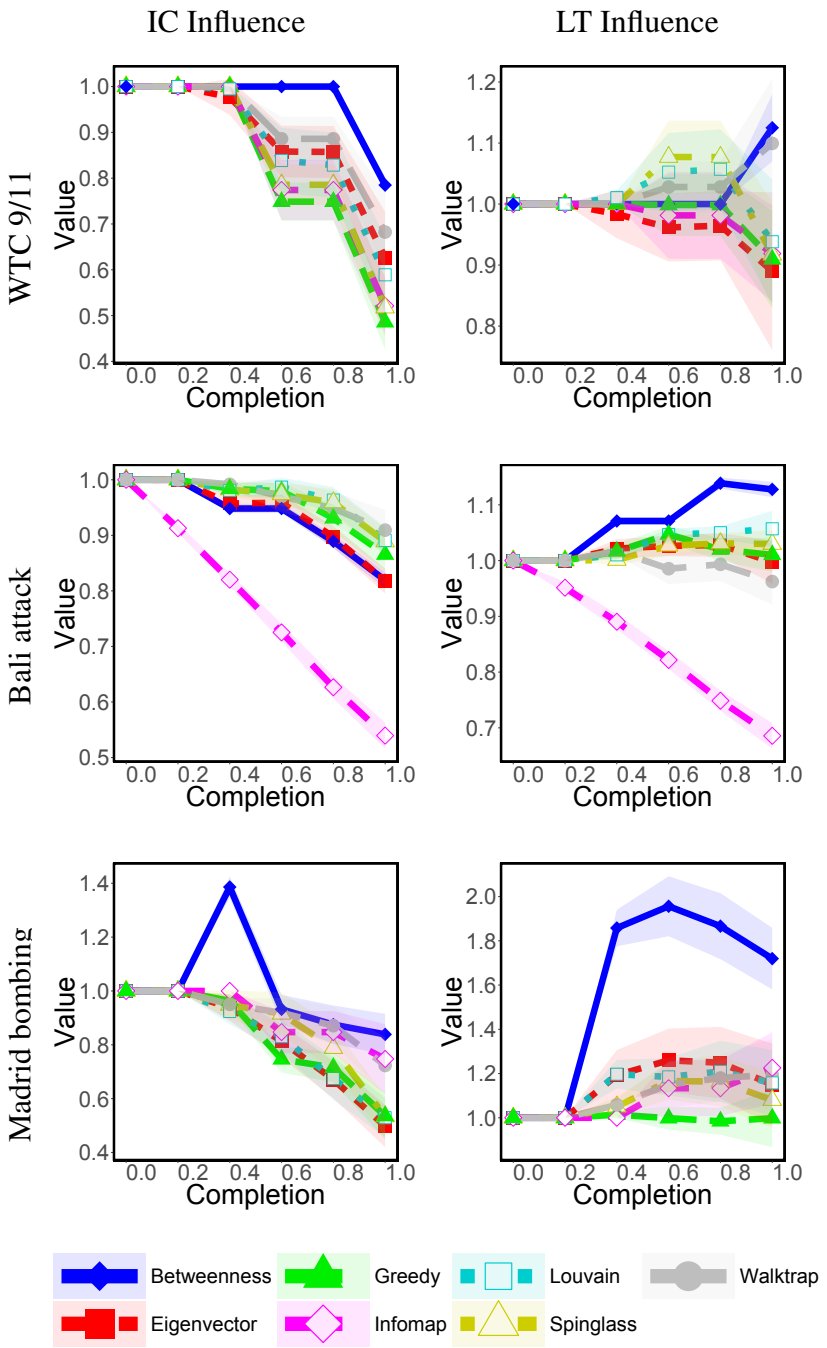
Supplementary Figure 37: Consecutive execution of both ROAM and DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for DICE(b, d) : $b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM(b) : $b = 3$, where b is the budget in each execution.



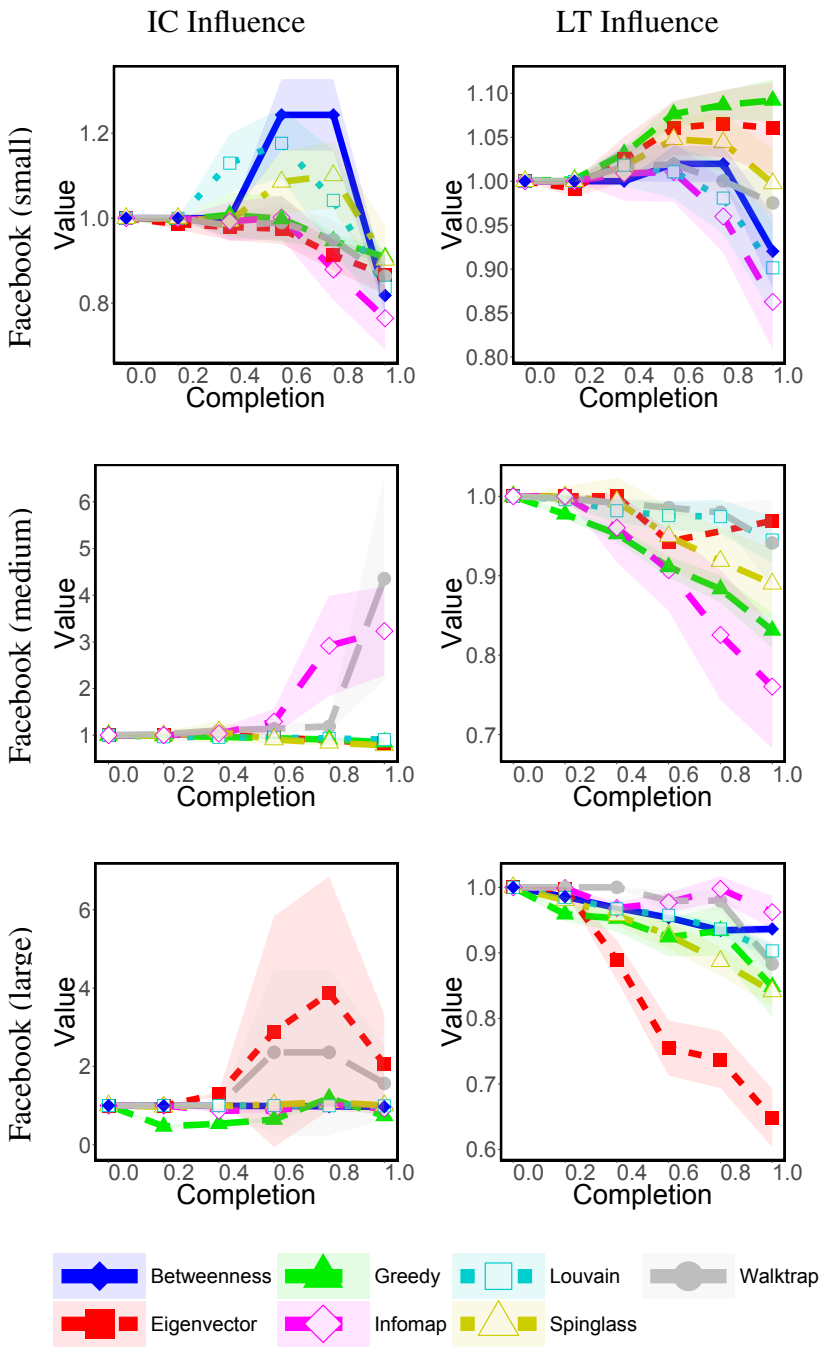
Supplementary Figure 38: Consecutive execution of both ROAM and DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for DICE(b, d) : $b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM(b) : $b = 3$, where b is the budget in each execution.



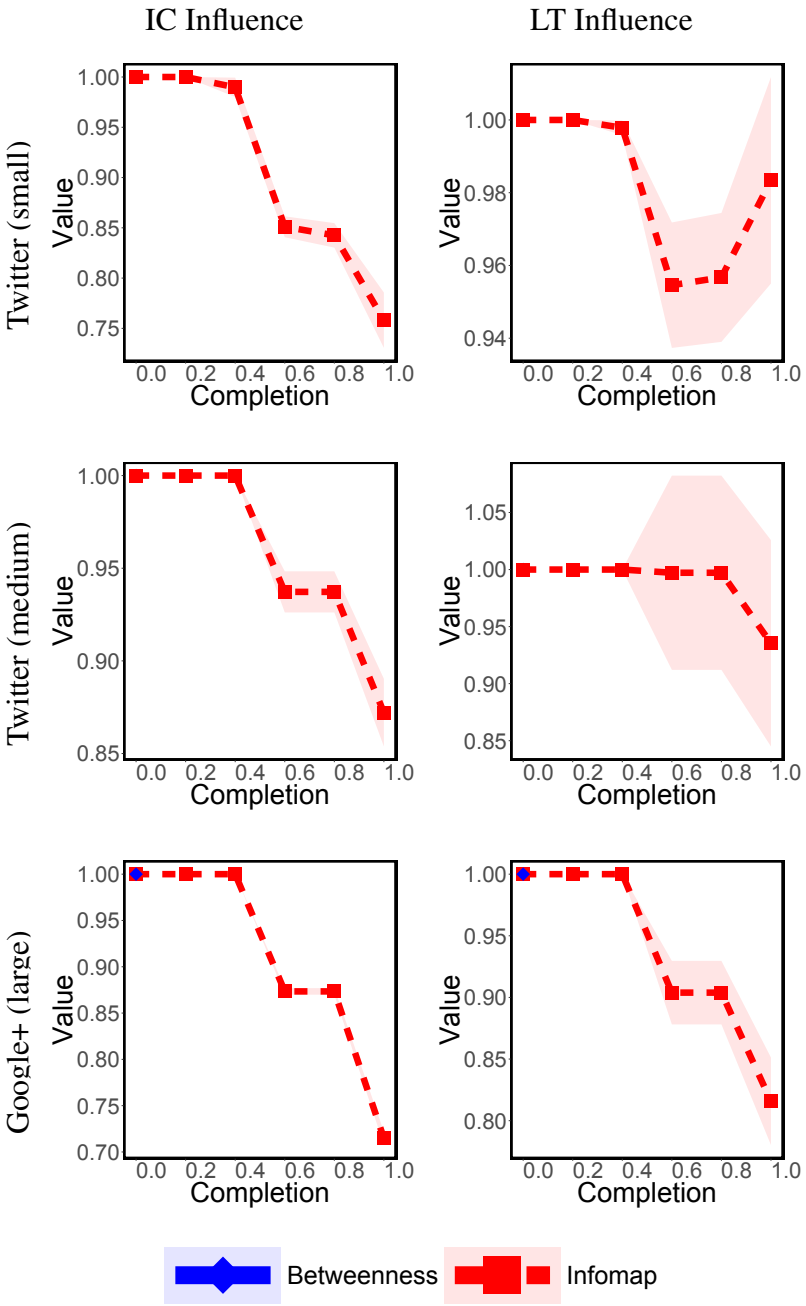
Supplementary Figure 39: Consecutive execution of both ROAM and DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for DICE(b, d) : $b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM(b) : $b = 3$, where b is the budget in each execution.



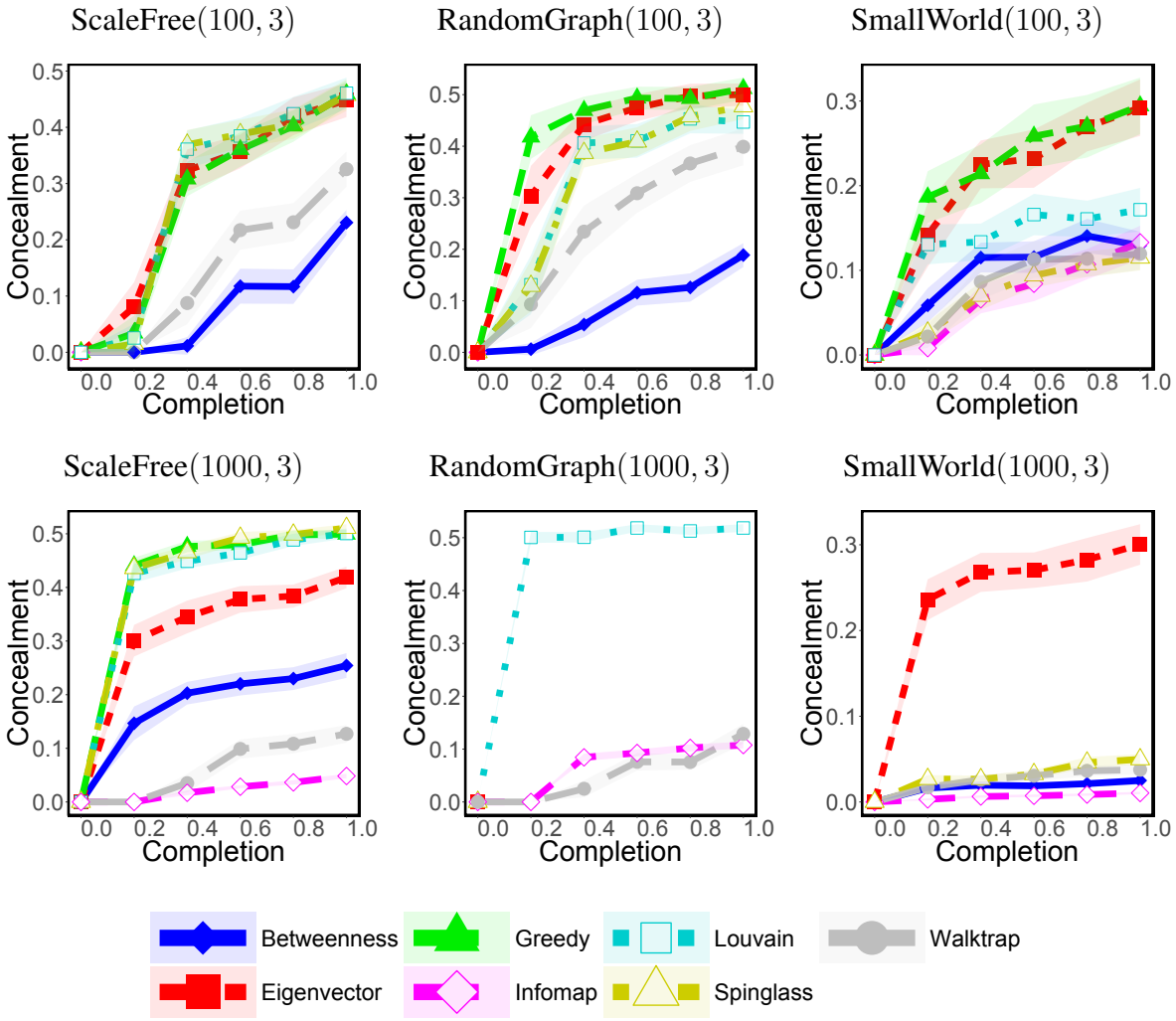
Supplementary Figure 40: Consecutive execution of both ROAM and DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for DICE(b, d) : $b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM(b) : $b = 3$, where b is the budget in each execution.



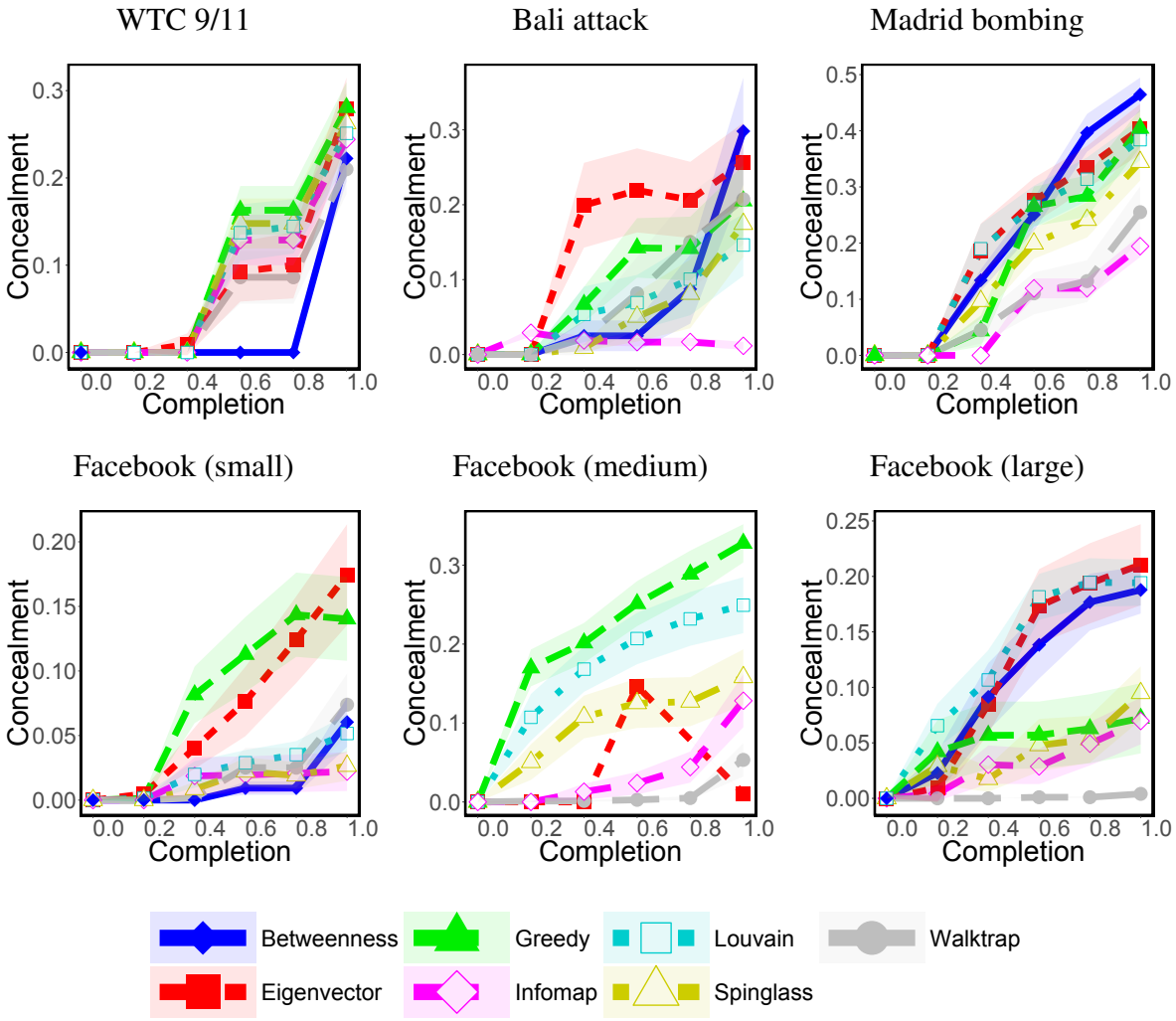
Supplementary Figure 41: Consecutive execution of both ROAM and DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for DICE(b, d) : $b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM(b) : $b = 3$, where b is the budget in each execution.



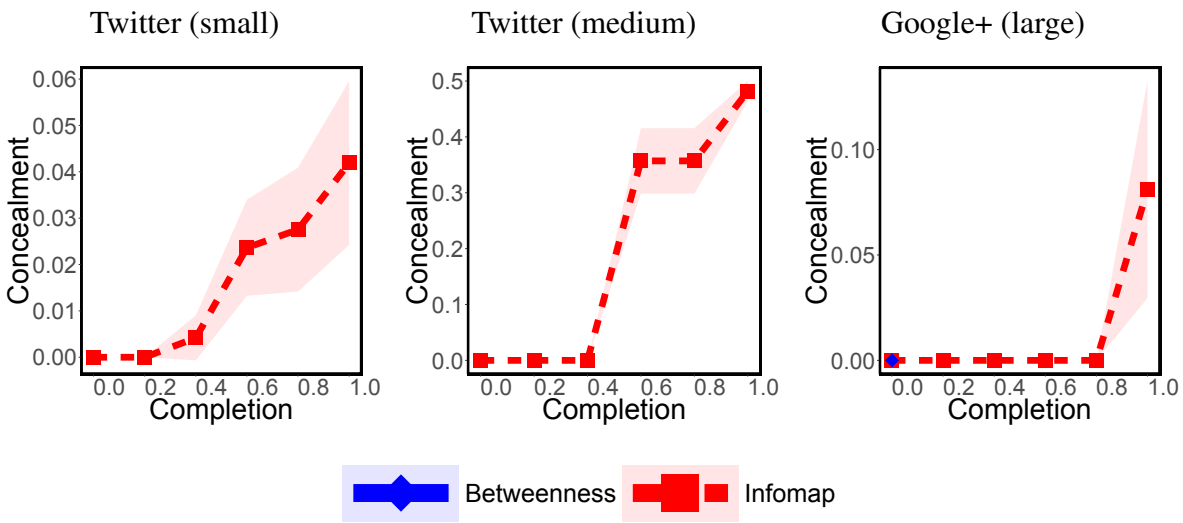
Supplementary Figure 42: Consecutive execution of both ROAM and DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for DICE(b, d) : $b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM(b) : $b = 3$, where b is the budget in each execution.



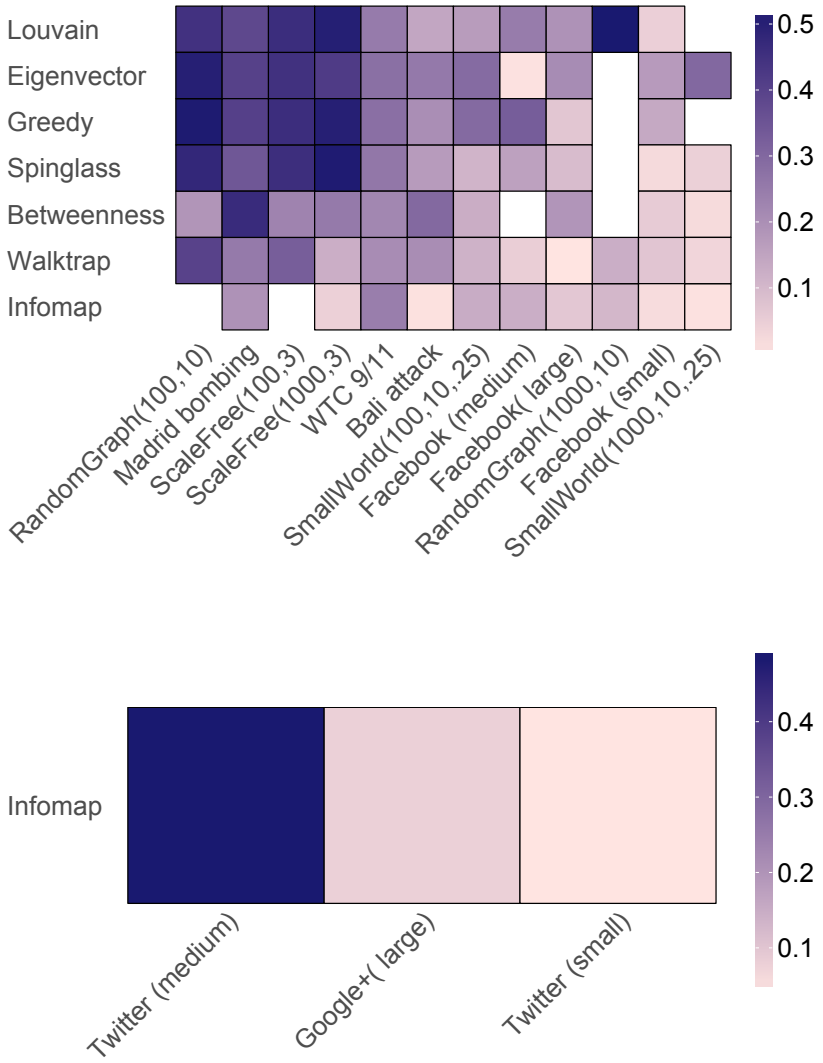
Supplementary Figure 43: Consecutive execution of both ROAM and DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE$(b, d) : b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM$(b) : b = 3$, where b is the budget in each execution.



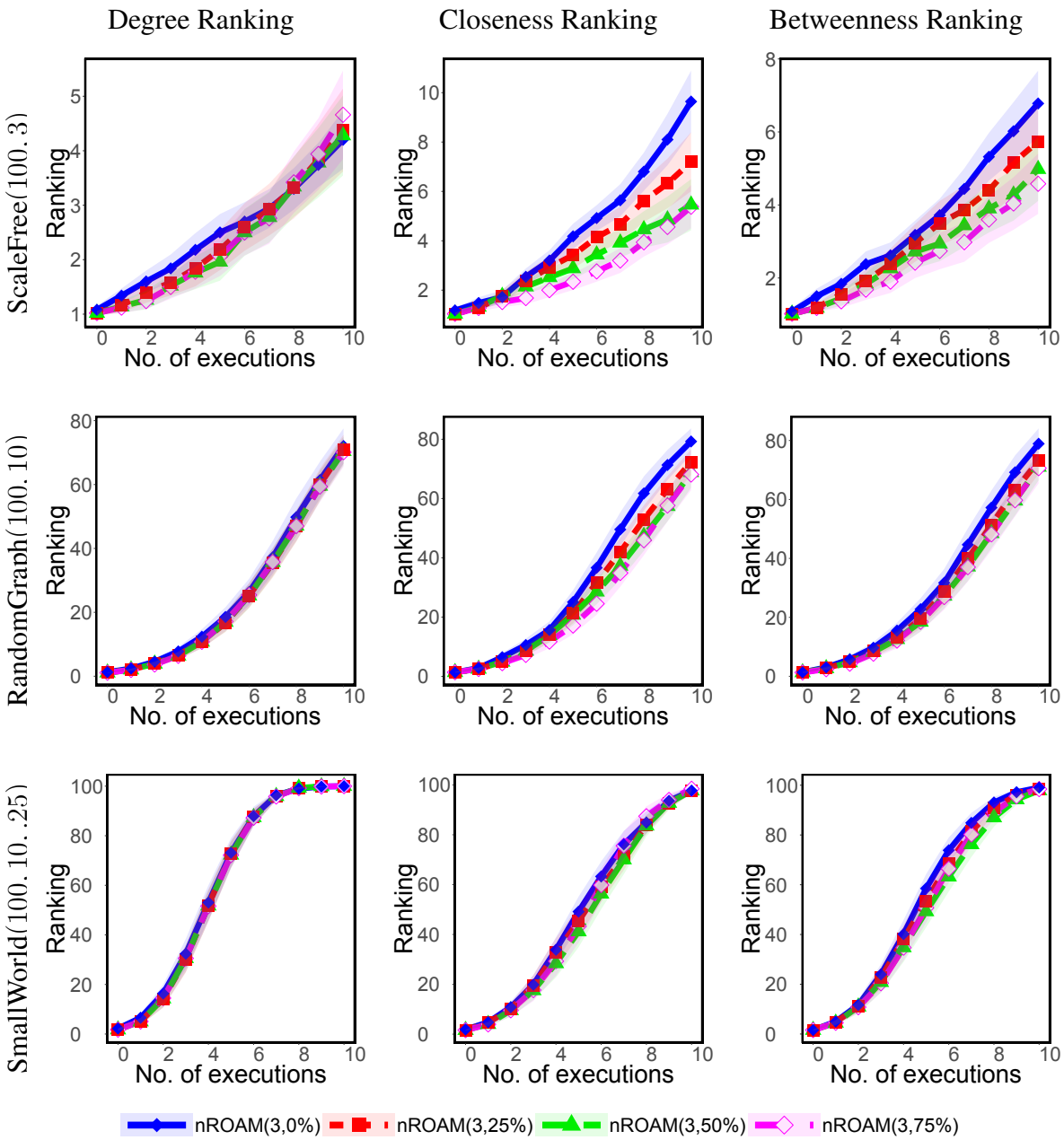
Supplementary Figure 44: Consecutive execution of both ROAM and DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE$(b, d) : b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM$(b) : b = 3$, where b is the budget in each execution.



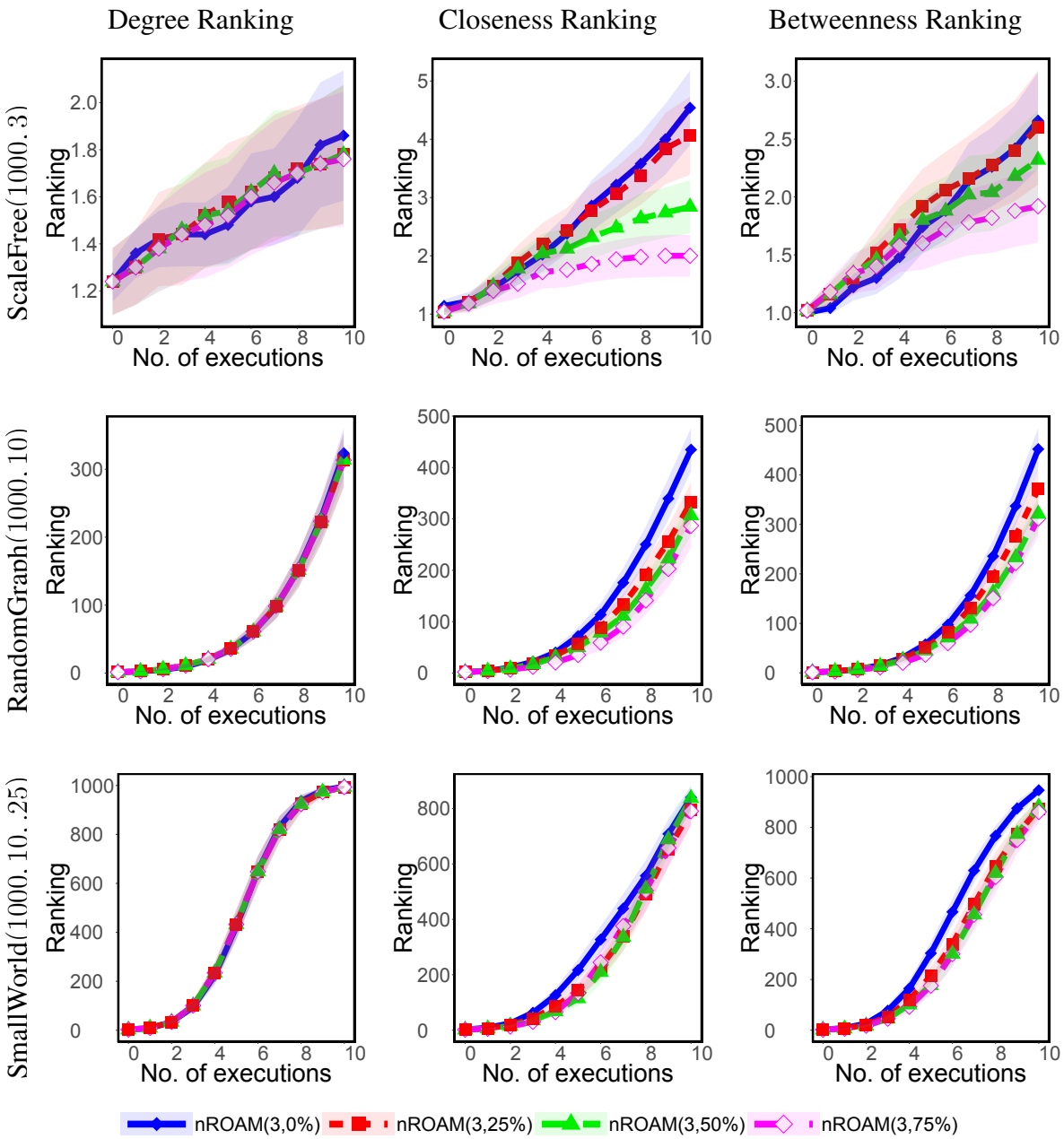
Supplementary Figure 45: Consecutive execution of both ROAM and DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE$(b, d) : b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM$(b) : b = 3$, where b is the budget in each execution.



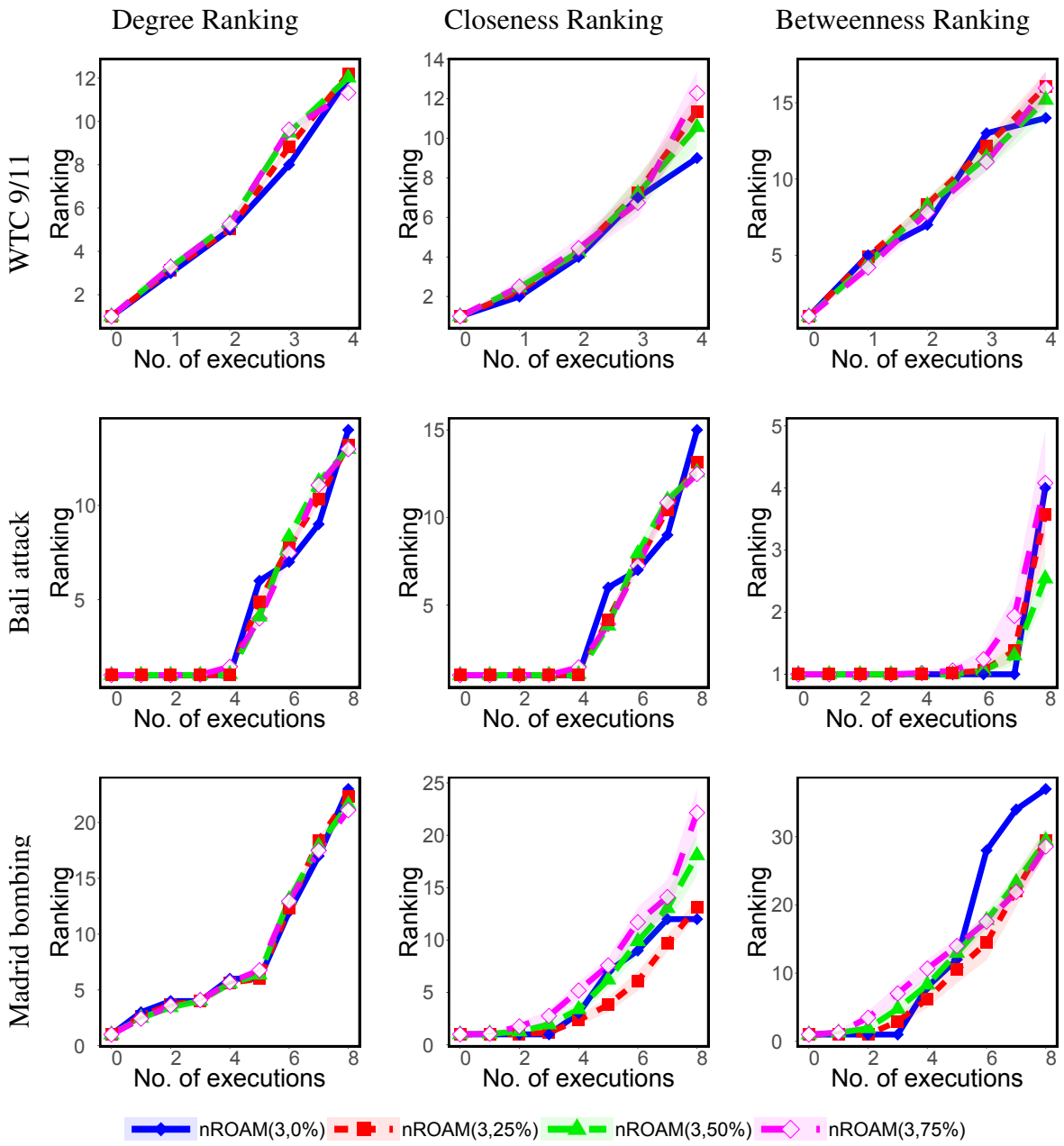
Supplementary Figure 46: Average concealment-measure value in each experiment on running ROAM and DICE simultaneously. The results for the directed networks (namely the fragments of Twitter and Google+) are presented on the right, whereas the results for the undirected networks are presented on the left. Results are shown for DICE$(b, d) : b = 4; d = 2$, where b is the budget in each execution, and d is the number of removed internal edges, and ROAM$(b) : b = 3$, where b is the budget in each execution.



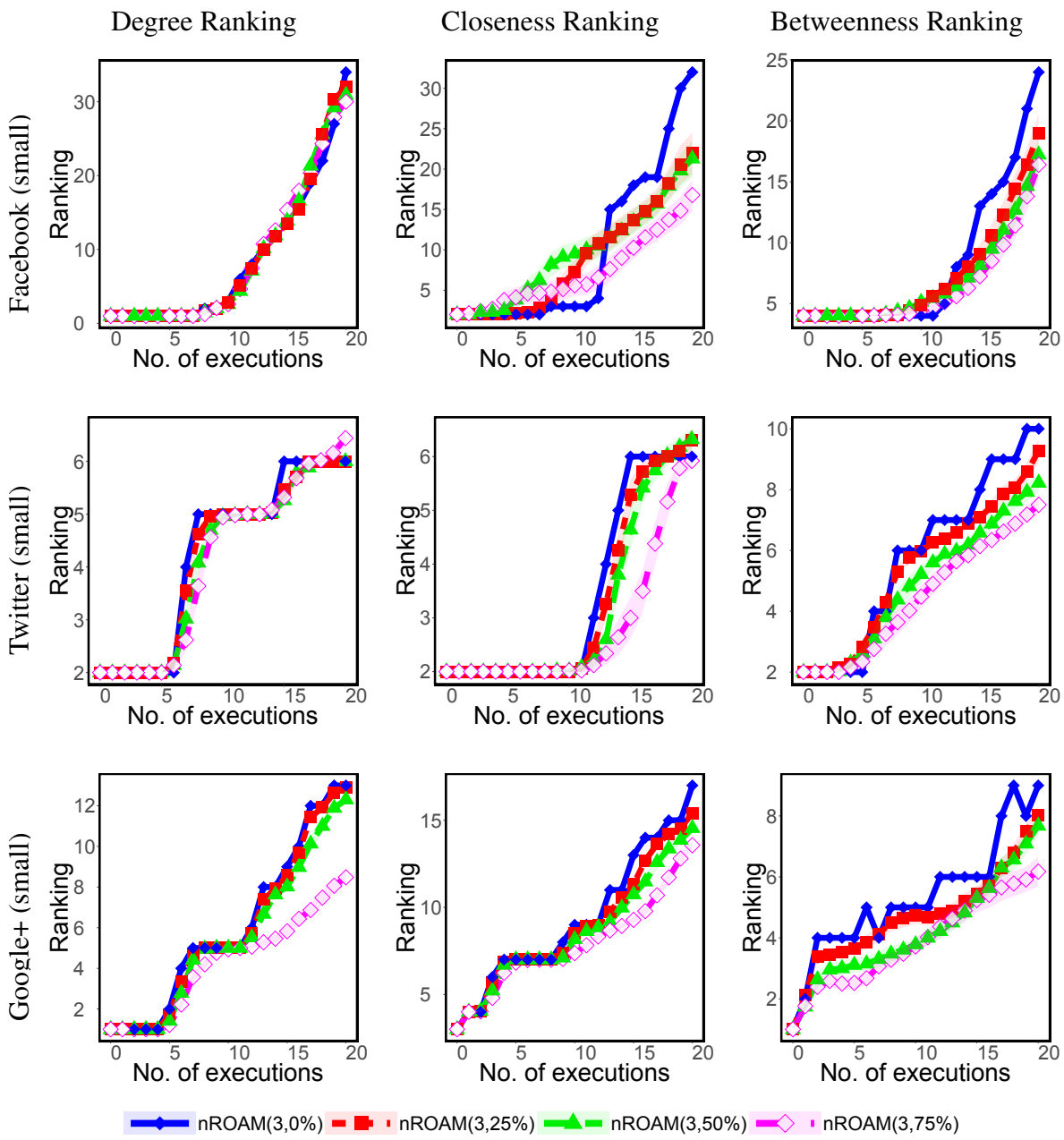
Supplementary Figure 47: Consecutive execution of nROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for nROAM$(b, p) : b = 3$, where b is the budget in each execution and p is the percentage of nodes that make their lists of neighbors invisible.



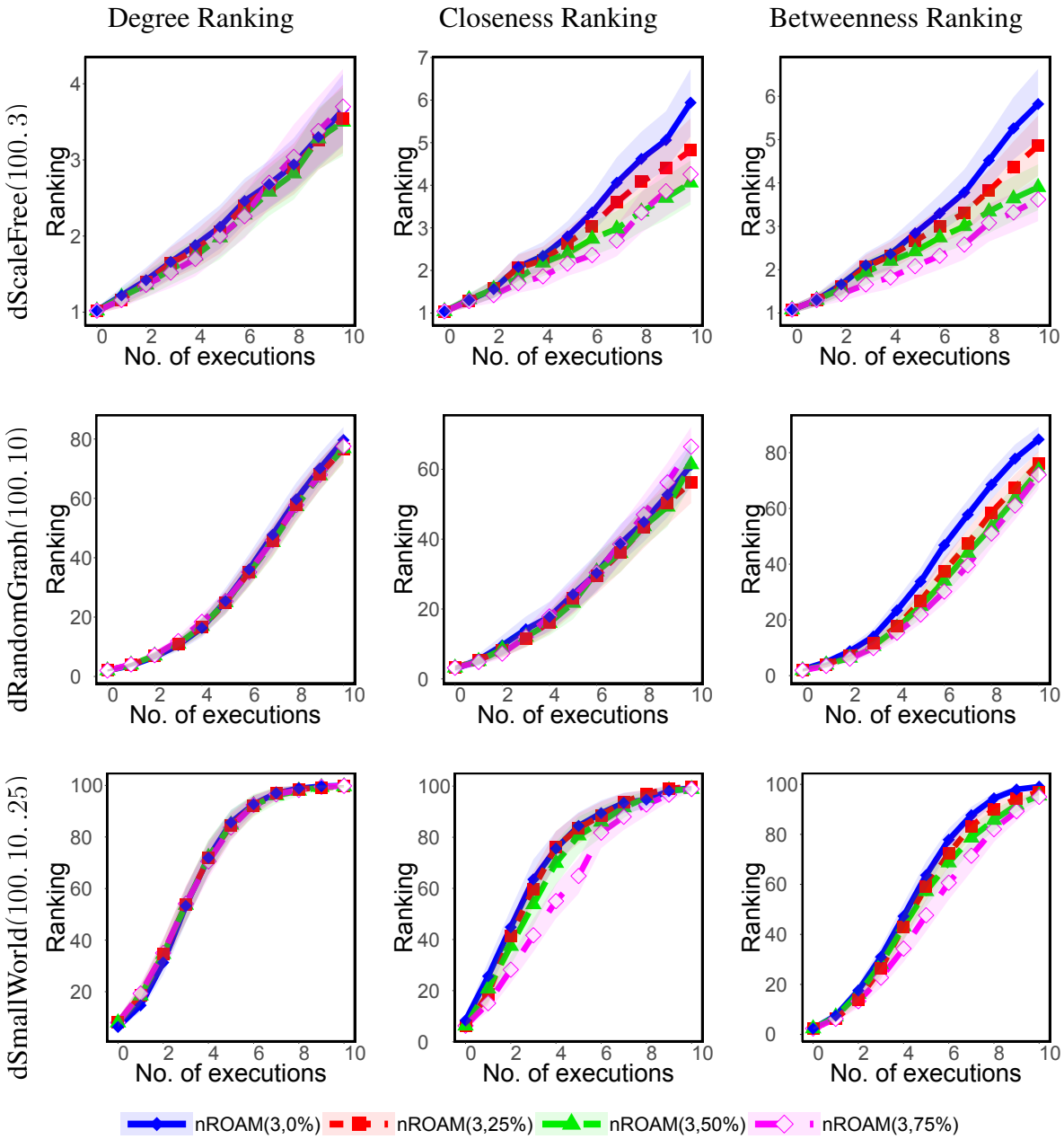
Supplementary Figure 48: Consecutive execution of nROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for nROAM$(b, p) : b = 3$, where b is the budget in each execution and p is the percentage of nodes that make their lists of neighbors invisible.



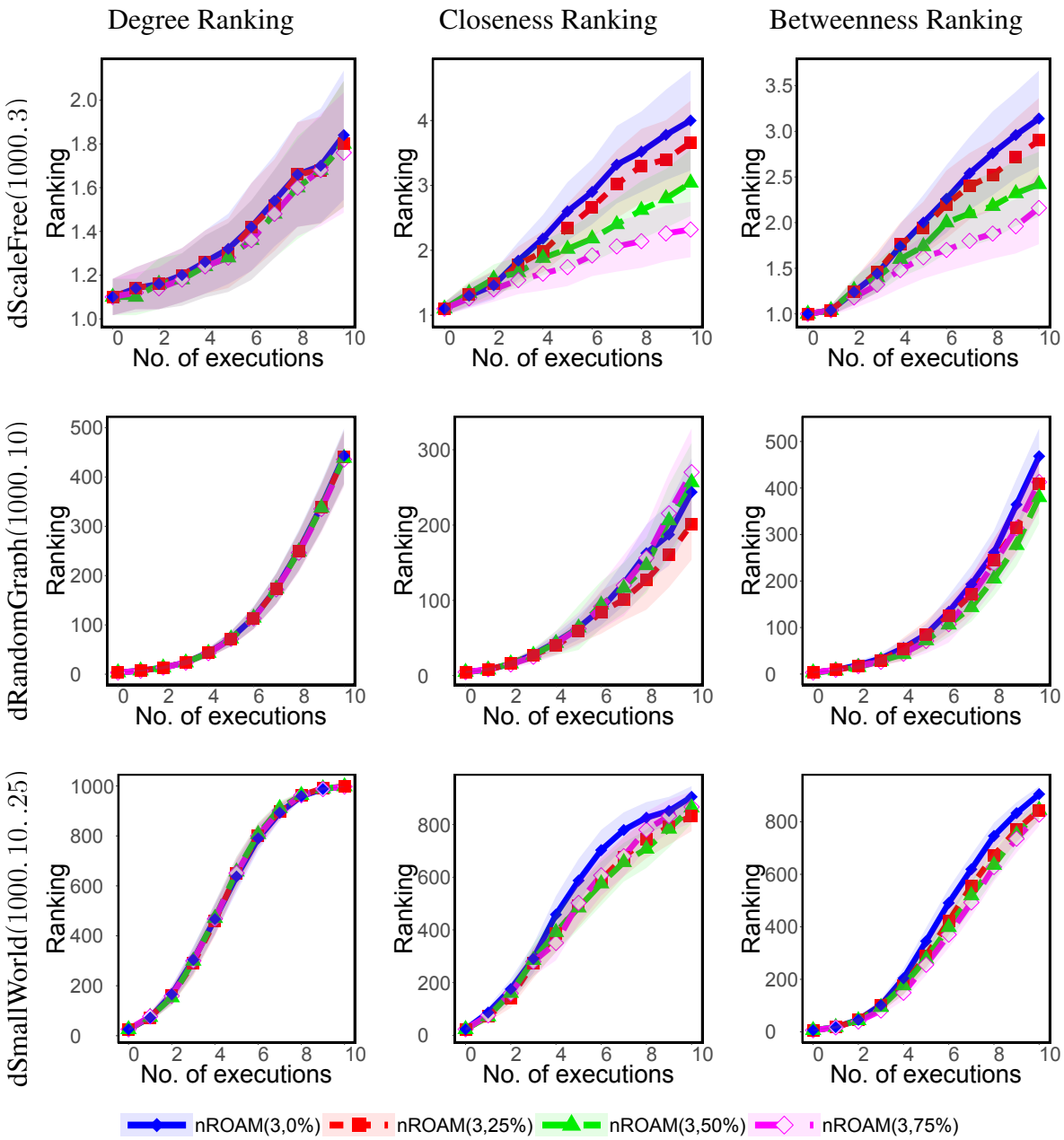
Supplementary Figure 49: Consecutive execution of nROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for nROAM$(b, p) : b = 3$, where b is the budget in each execution and p is the percentage of nodes that make their lists of neighbors invisible.



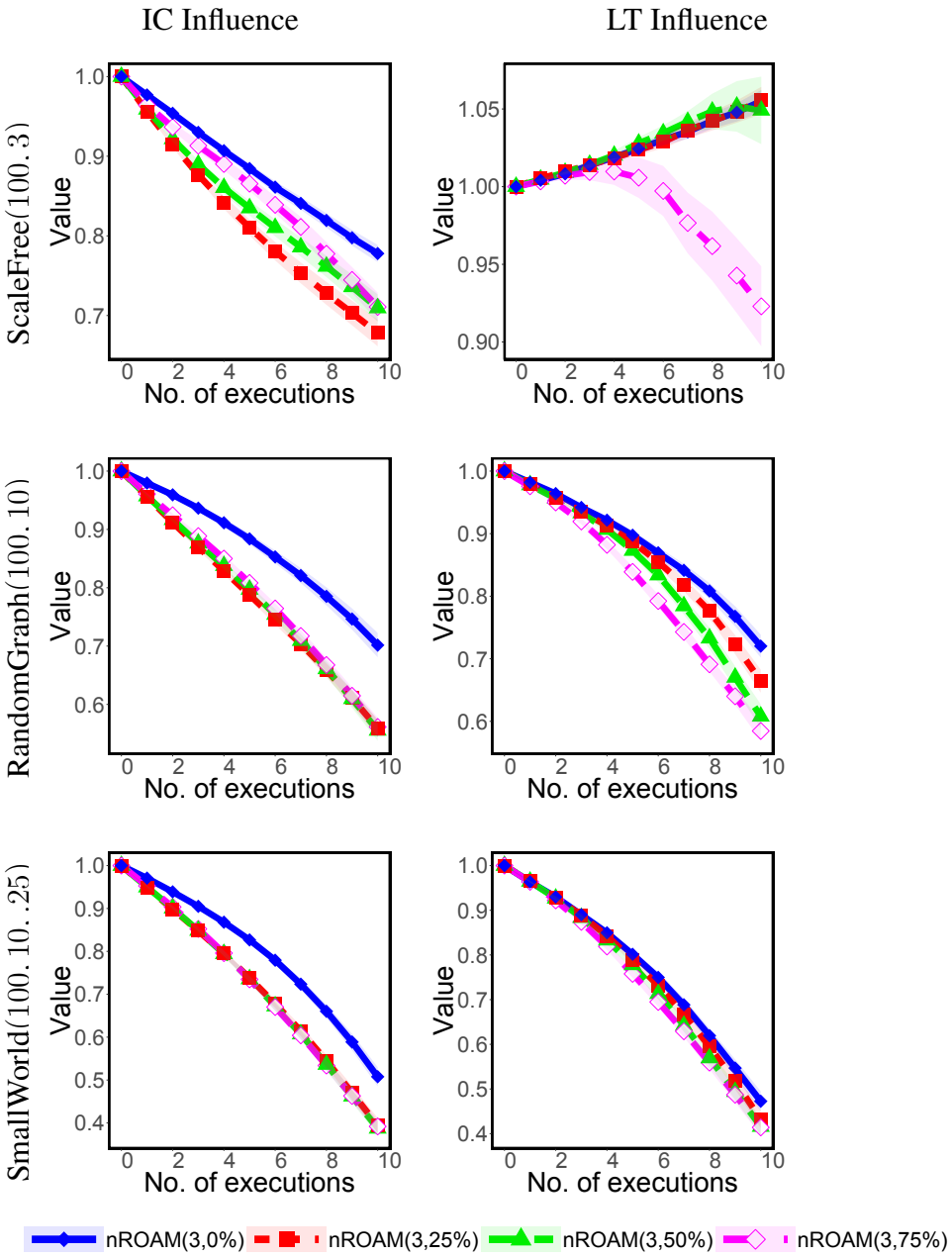
Supplementary Figure 50: Consecutive execution of nROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for nROAM$(b, p) : b = 3$, where b is the budget in each execution and p is the percentage of nodes that make their lists of neighbors invisible.



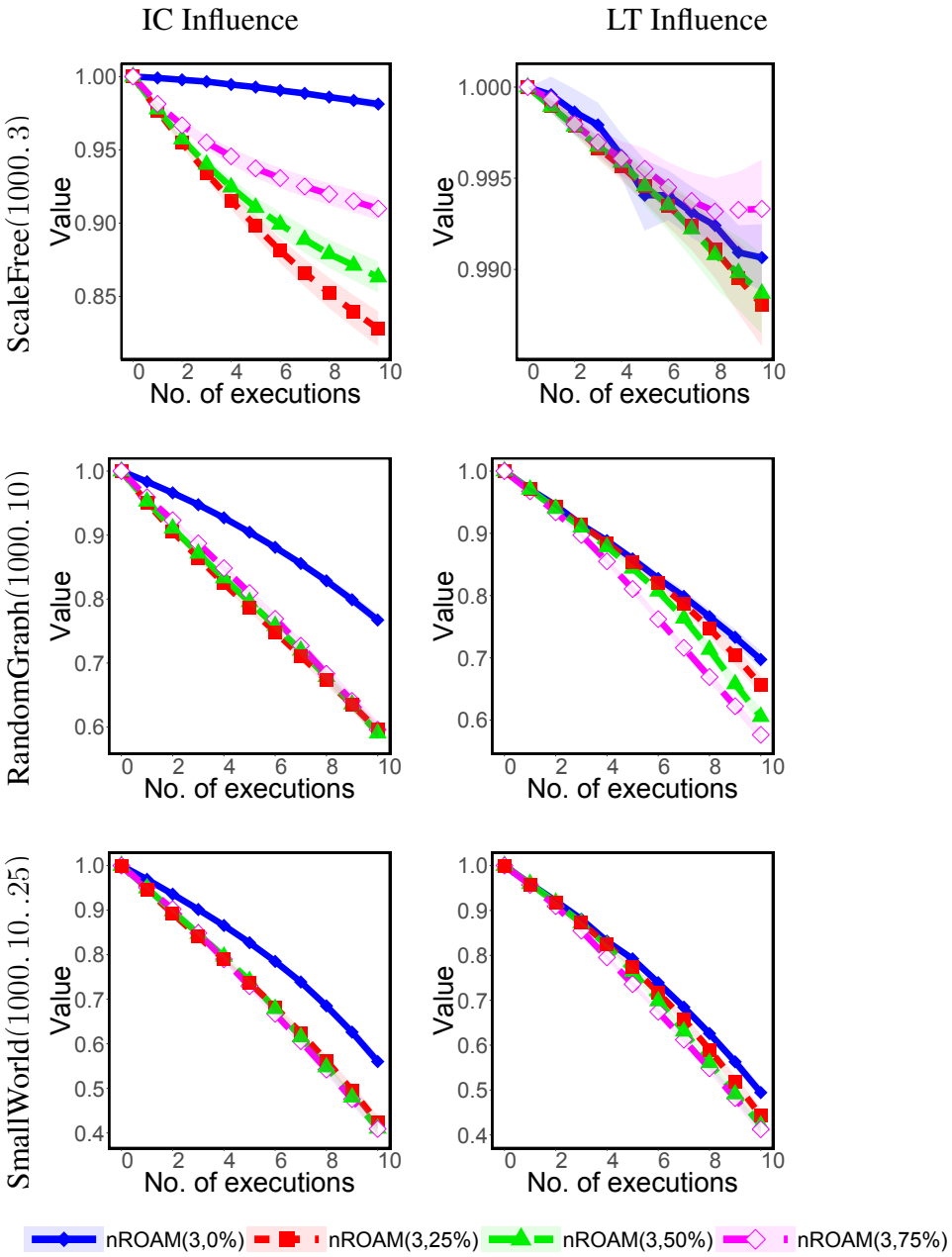
Supplementary Figure 51: Consecutive execution of nROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for nROAM$(b, p) : b = 3$, where b is the budget in each execution and p is the percentage of nodes that make their lists of neighbors invisible.



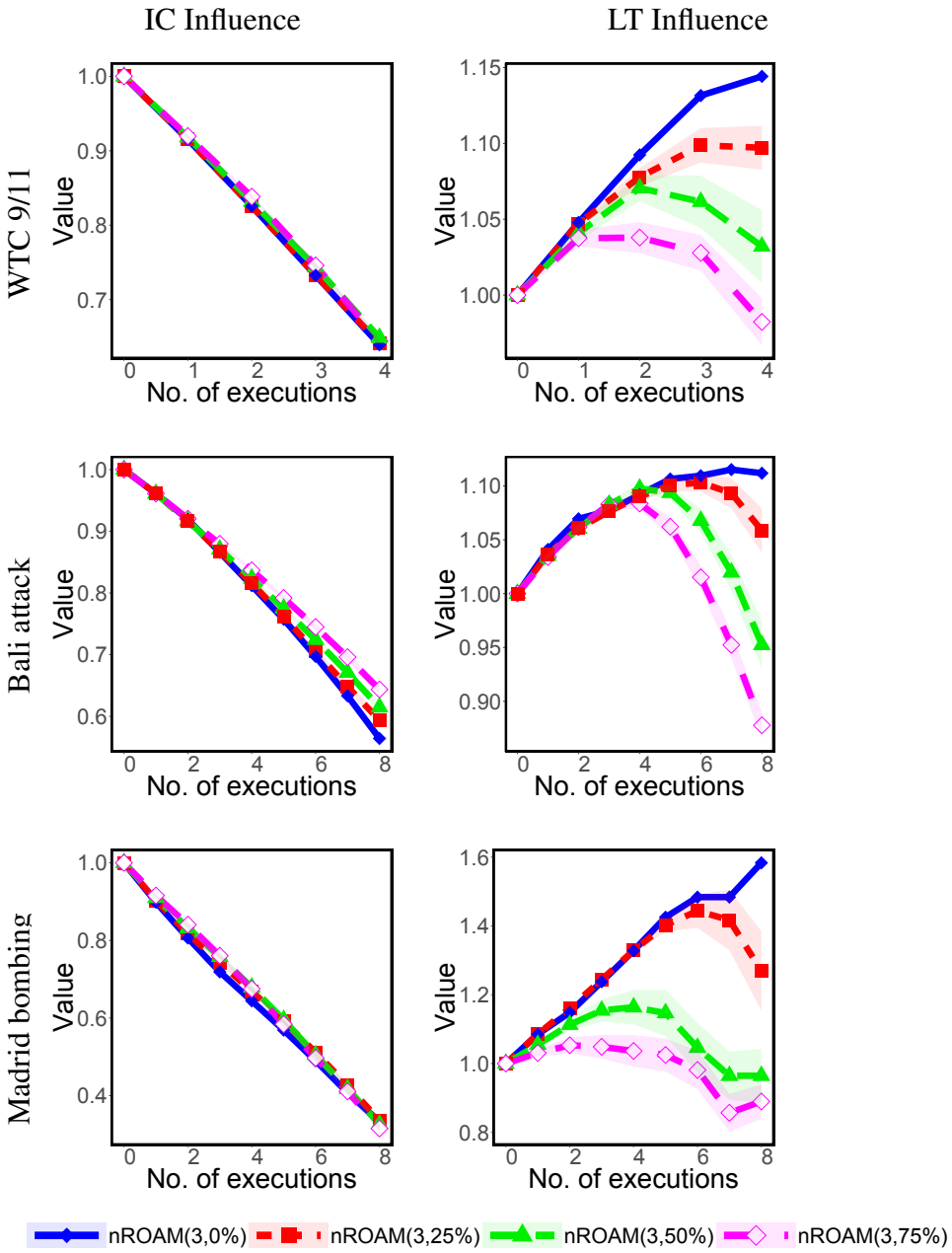
Supplementary Figure 52: Consecutive execution of nROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for nROAM$(b, p) : b = 3$, where b is the budget in each execution and p is the percentage of nodes that make their lists of neighbors invisible.



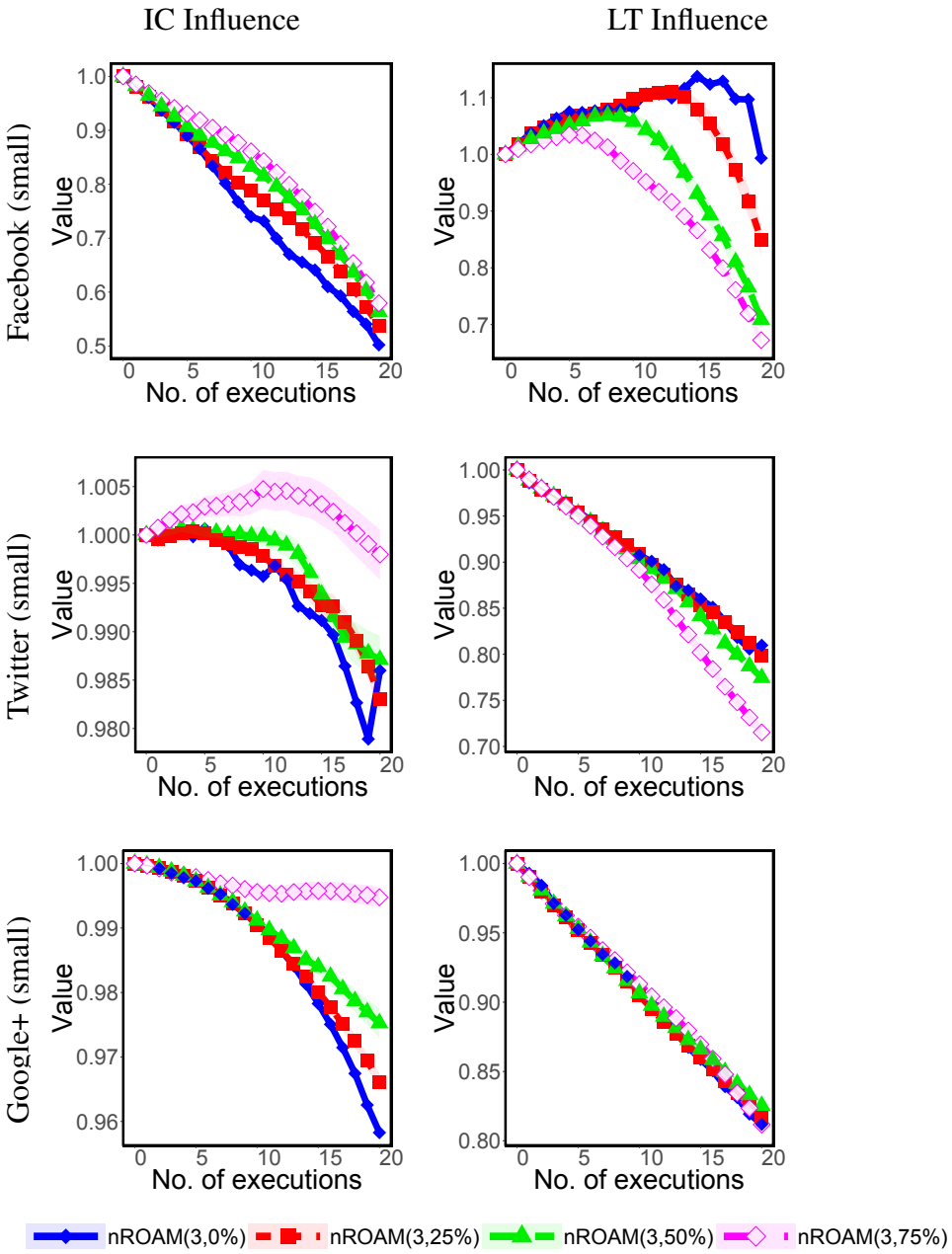
Supplementary Figure 53: Consecutive execution of nROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for nROAM$(b, p) : b = 3$, where b is the budget in each execution and p is the percentage of nodes that make their lists of neighbors invisible.



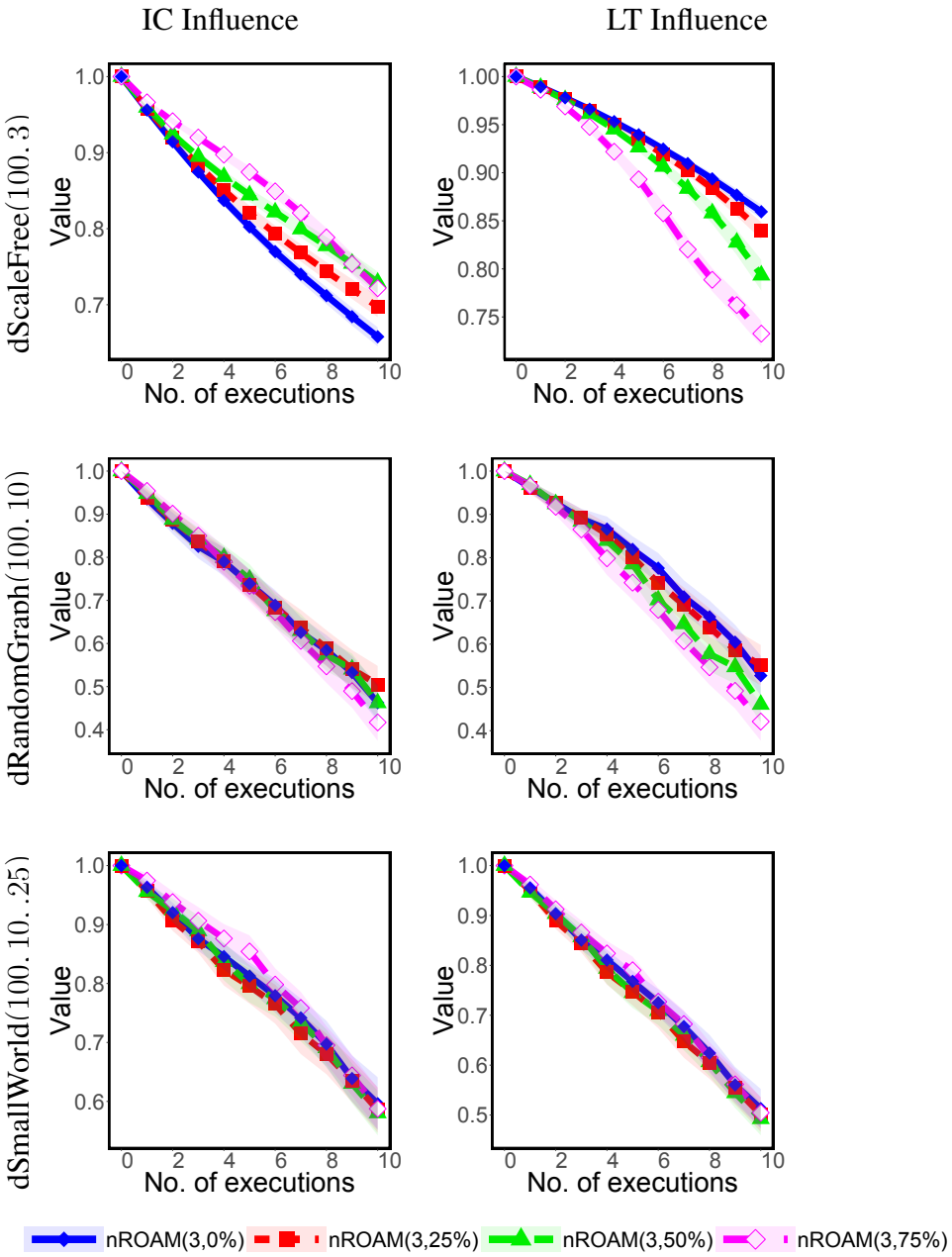
Supplementary Figure 54: Consecutive execution of nROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for nROAM$(b, p) : b = 3$, where b is the budget in each execution and p is the percentage of nodes that make their lists of neighbors invisible.



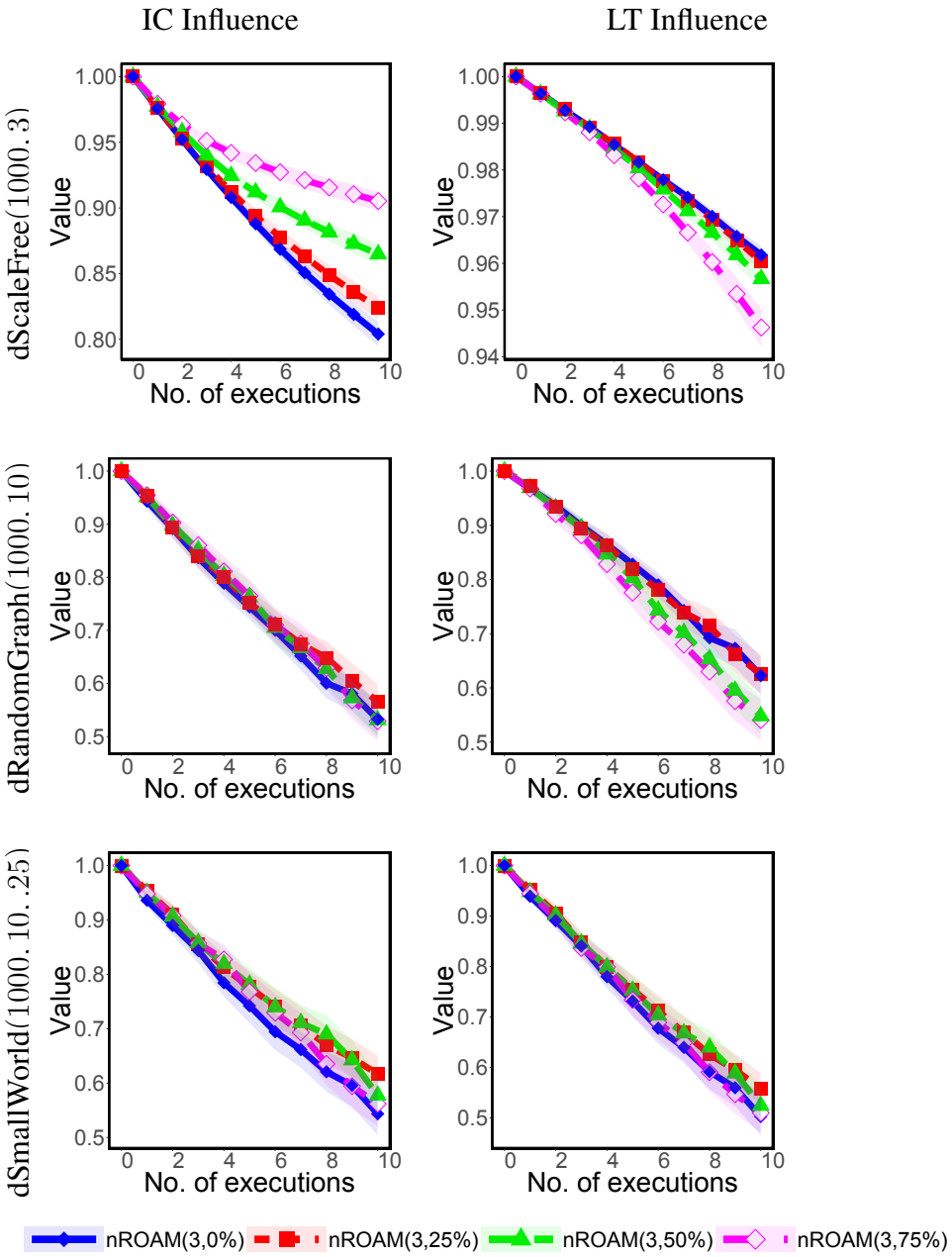
Supplementary Figure 55: Consecutive execution of nROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for nROAM$(b, p) : b = 3$, where b is the budget in each execution and p is the percentage of nodes that make their lists of neighbors invisible.



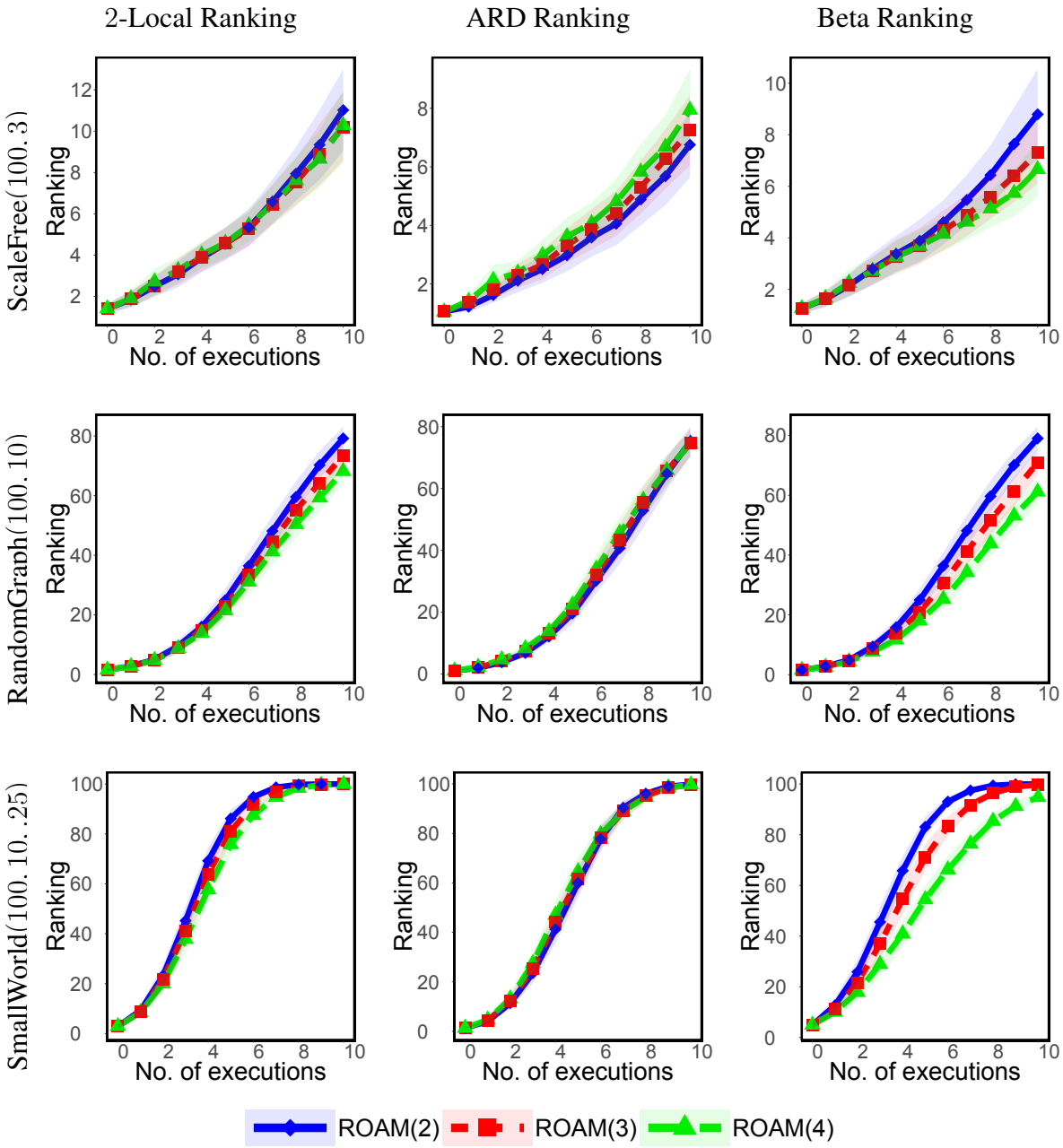
Supplementary Figure 56: Consecutive execution of nROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for nROAM(b, p) : $b = 3$, where b is the budget in each execution and p is the percentage of nodes that make their lists of neighbors invisible.



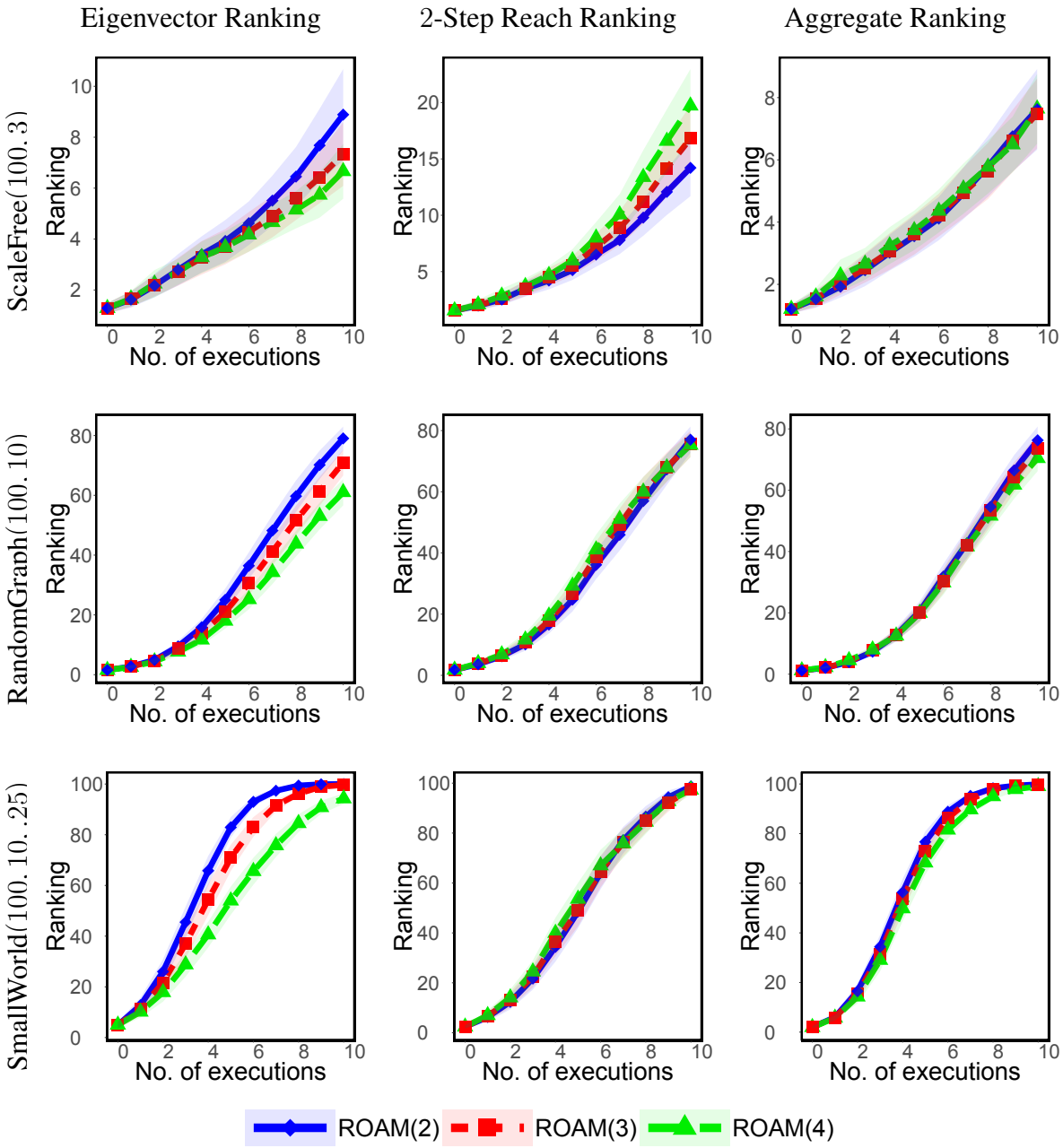
Supplementary Figure 57: Consecutive execution of nROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for nROAM(b, p) : $b = 3$, where b is the budget in each execution and p is the percentage of nodes that make their lists of neighbors invisible.



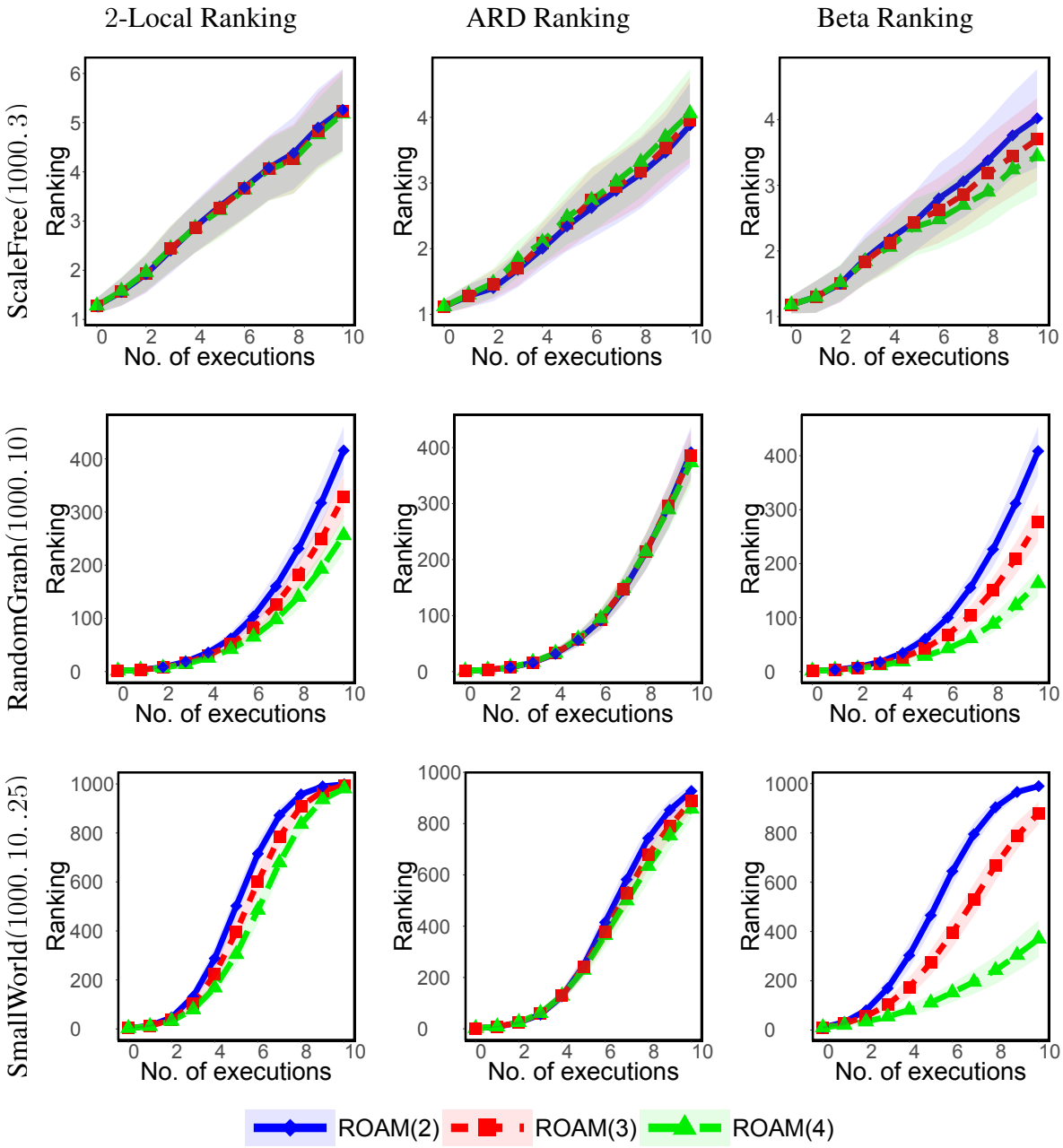
Supplementary Figure 58: Consecutive execution of nROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for nROAM(b, p) : $b = 3$, where b is the budget in each execution and p is the percentage of nodes that make their lists of neighbors invisible.



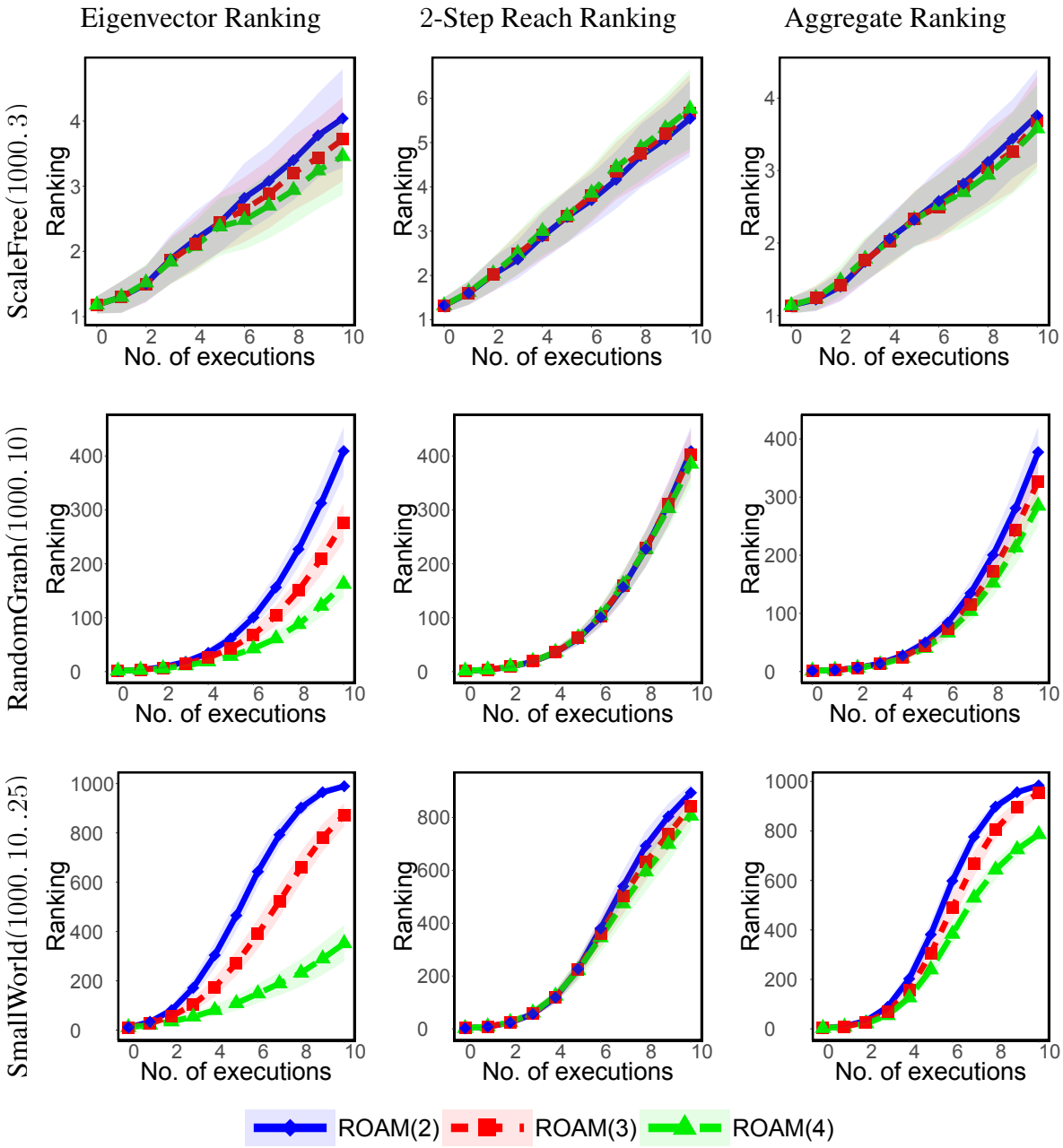
Supplementary Figure 59: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



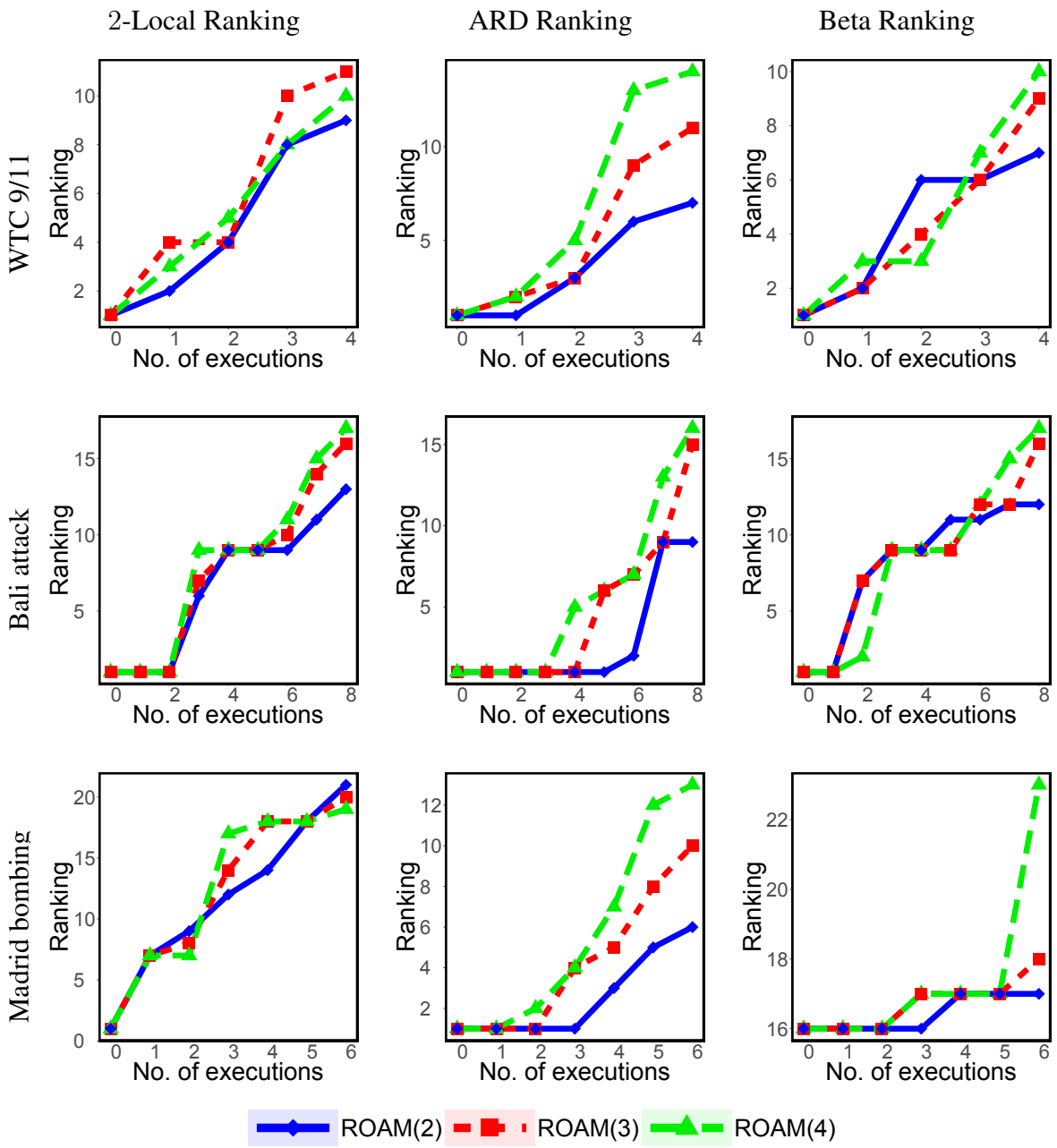
Supplementary Figure 60: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



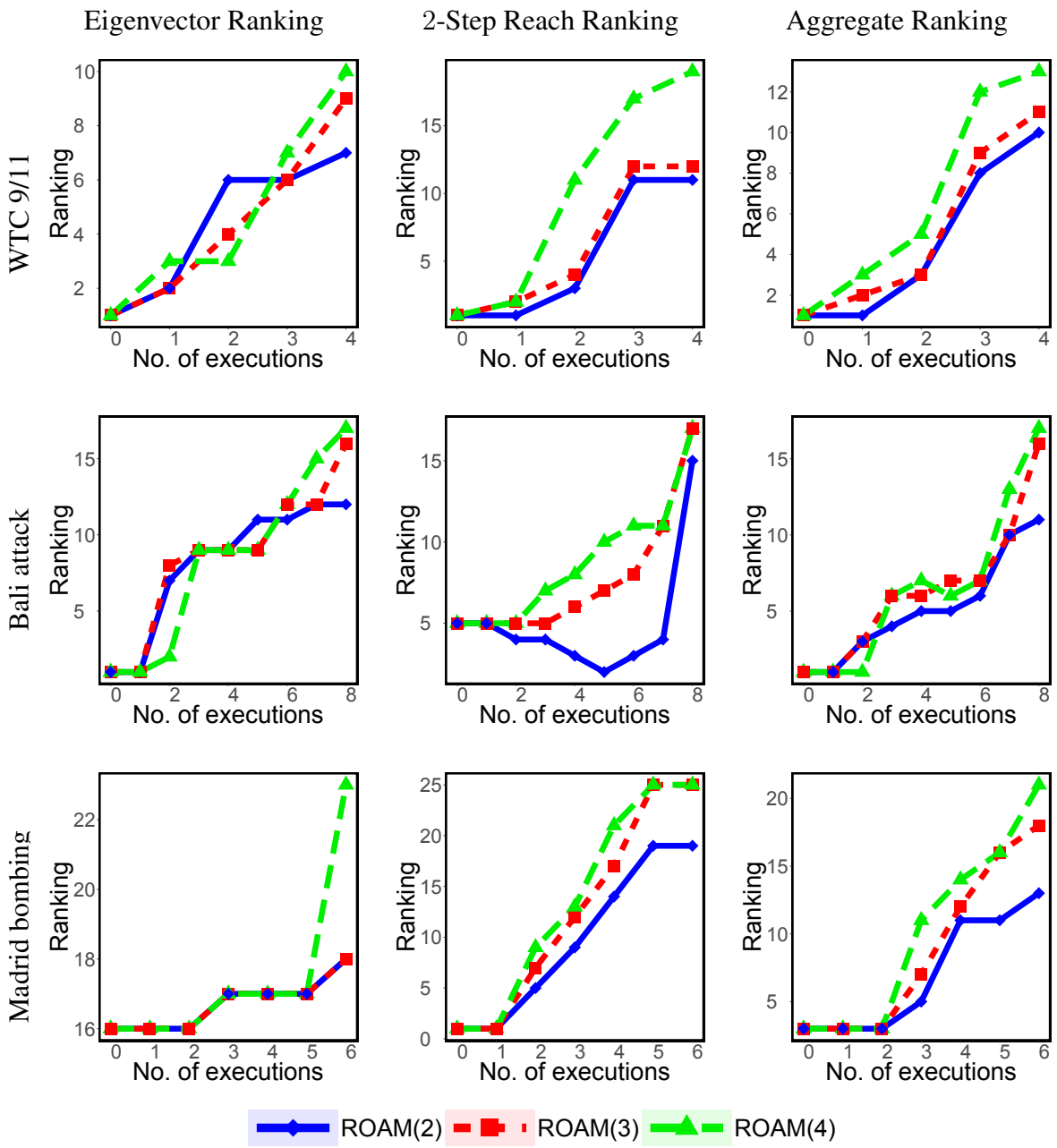
Supplementary Figure 61: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



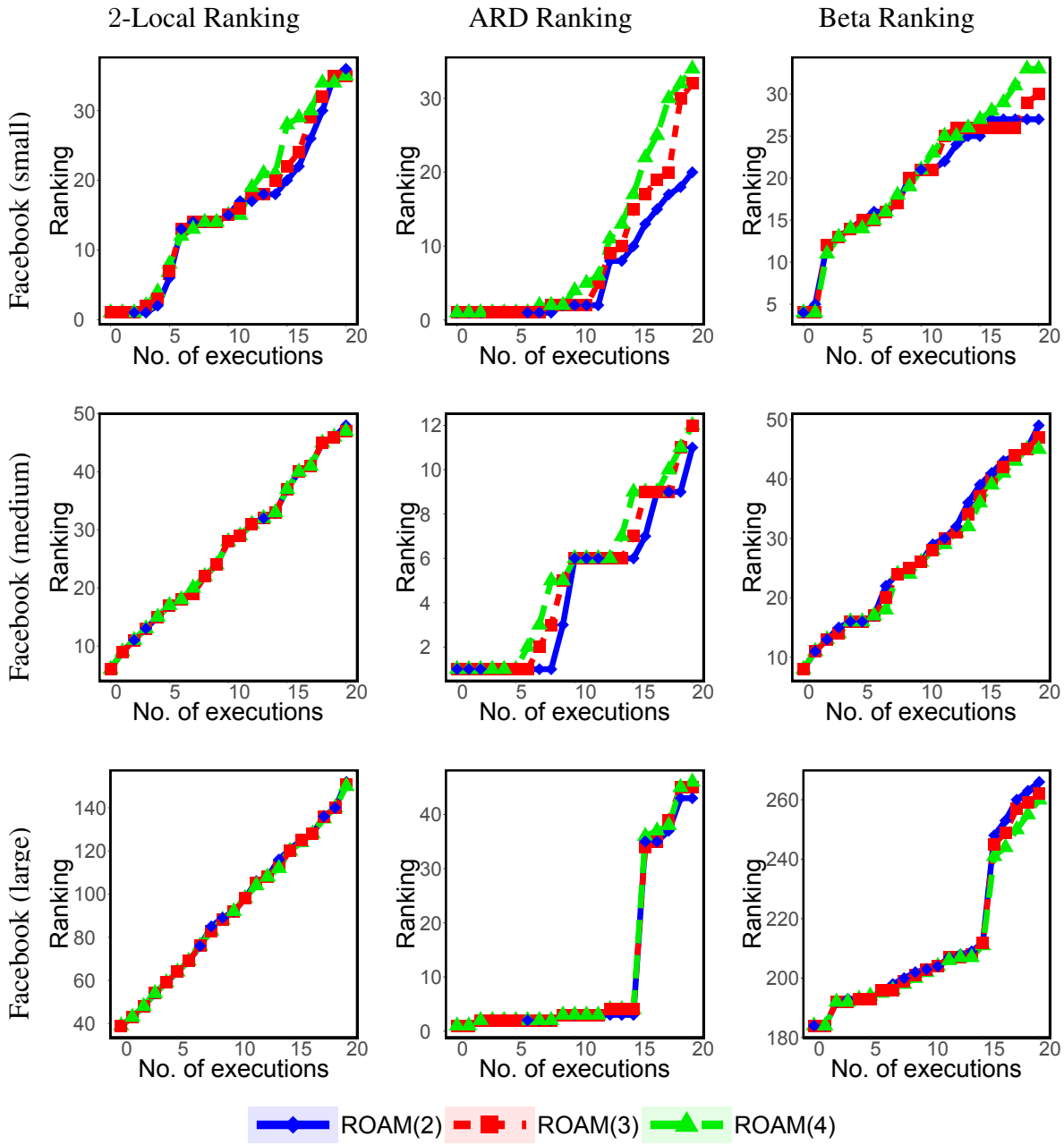
Supplementary Figure 62: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



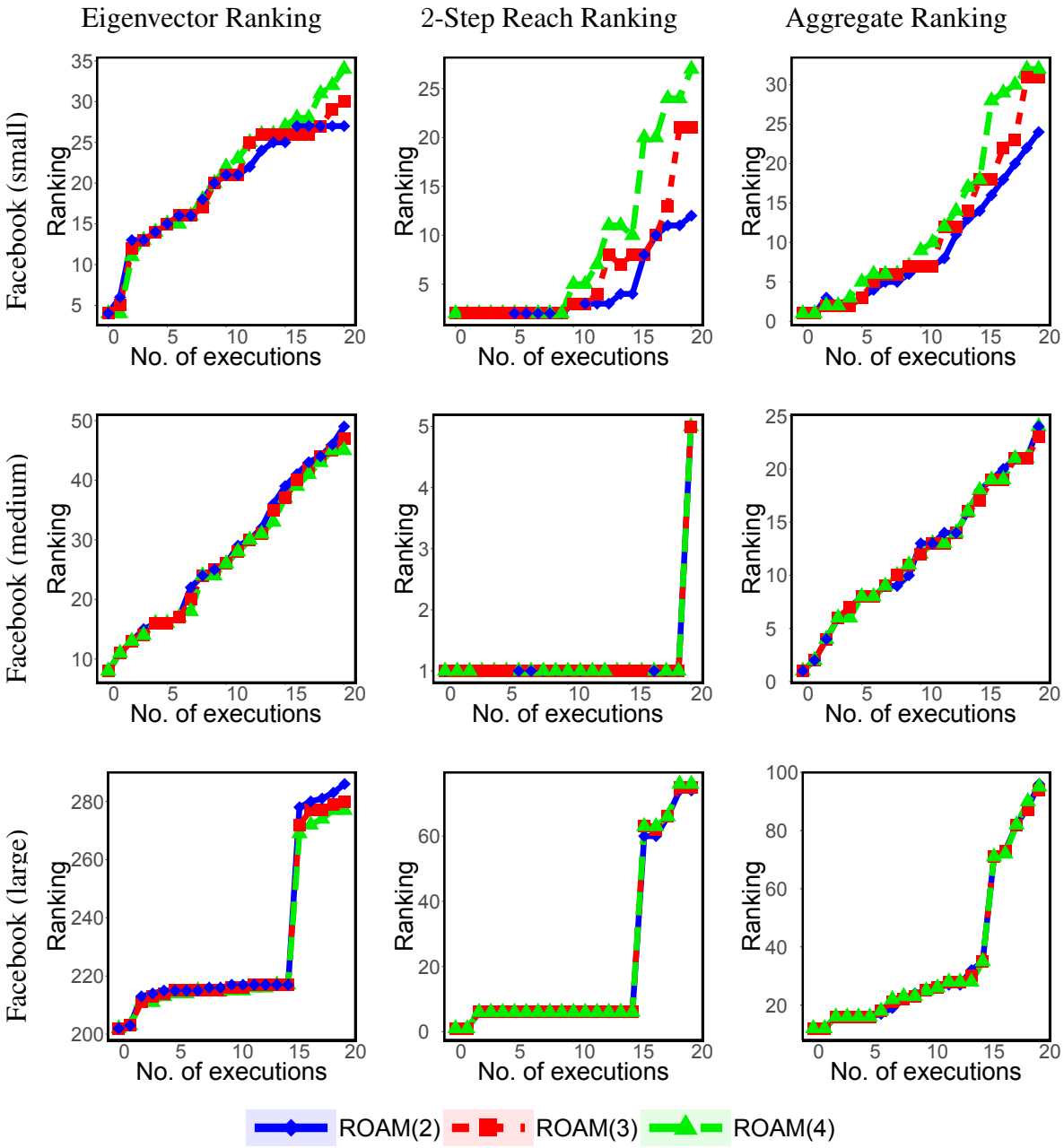
Supplementary Figure 63: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



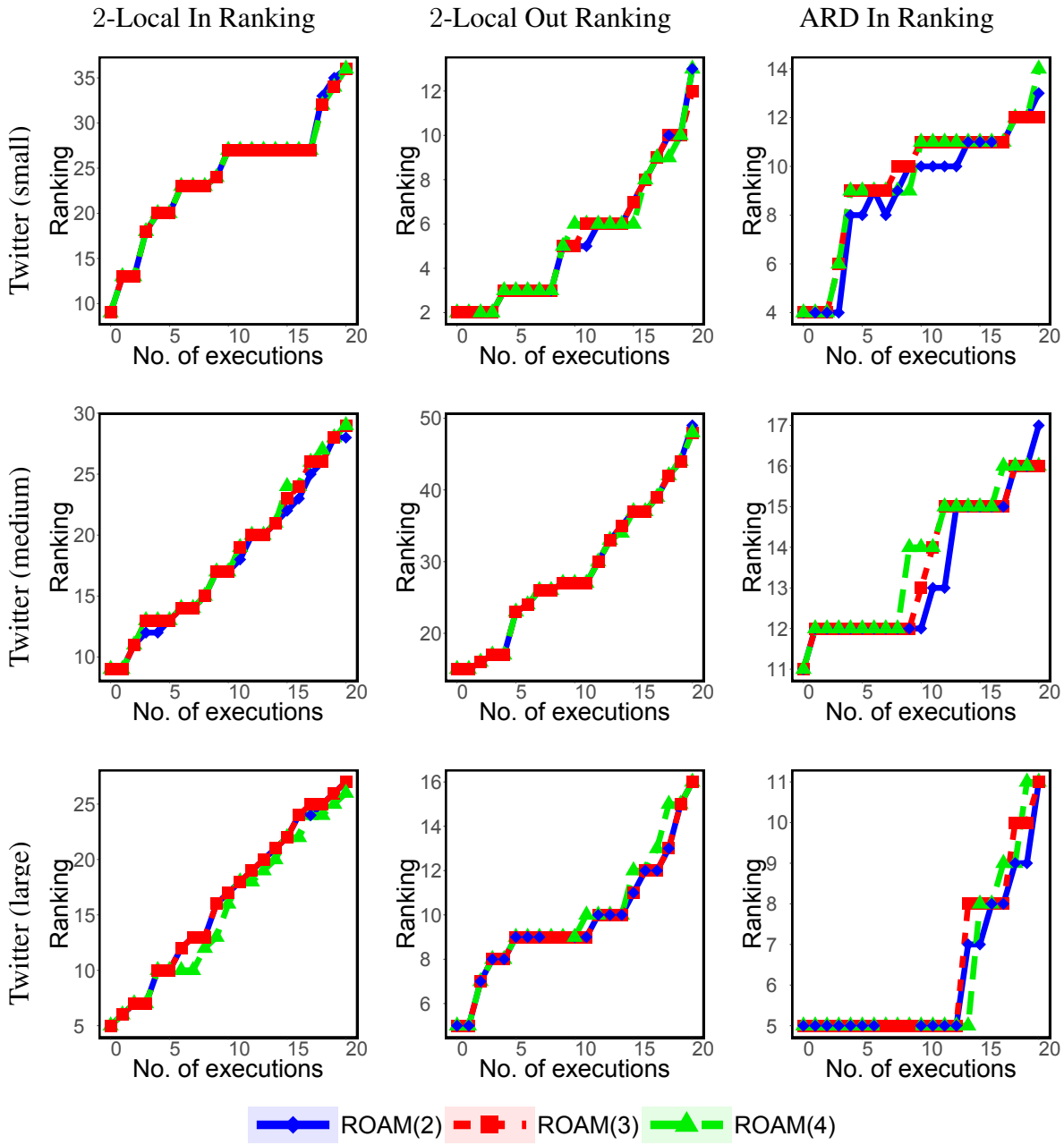
Supplementary Figure 64: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



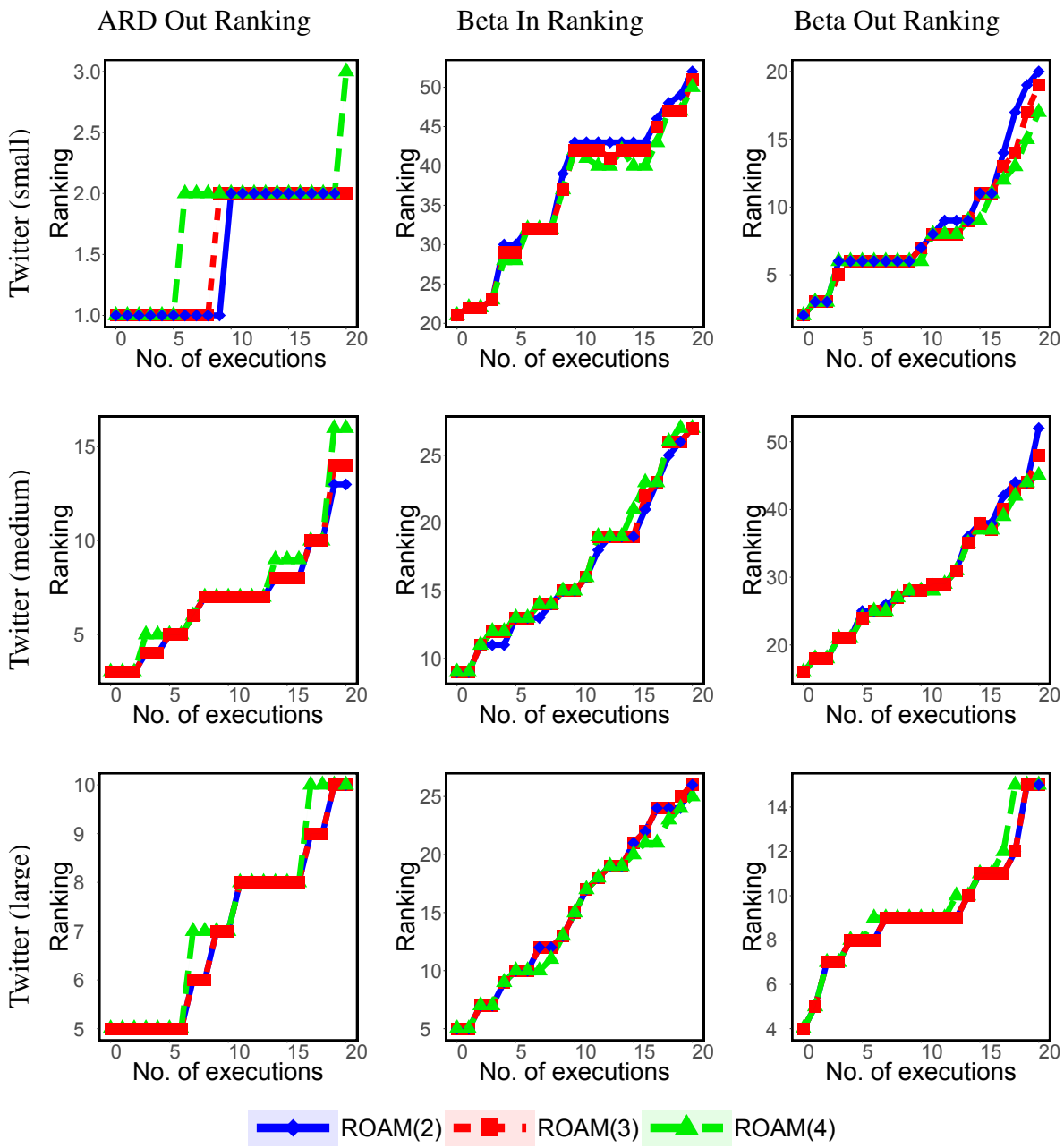
Supplementary Figure 65: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



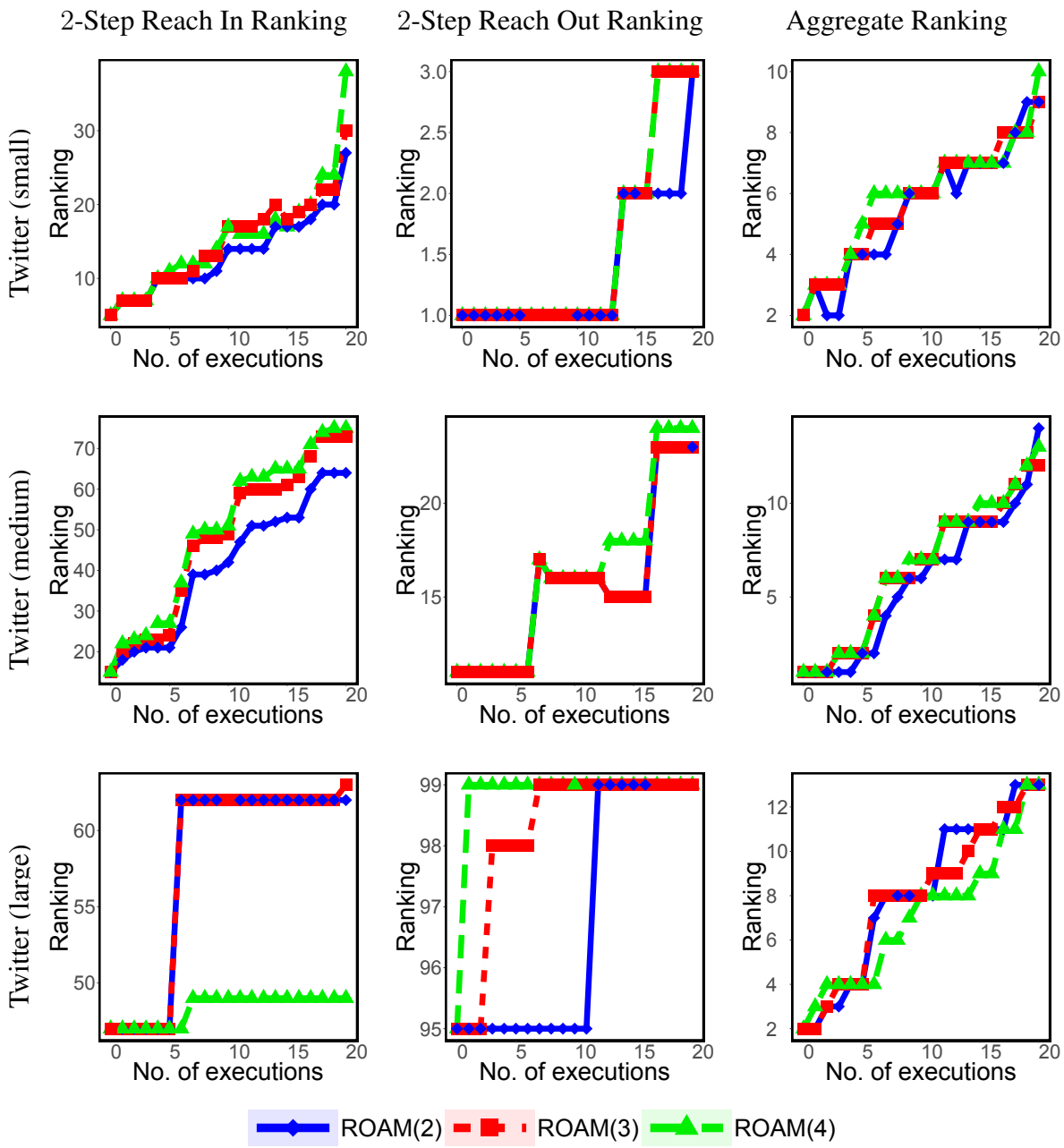
Supplementary Figure 66: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



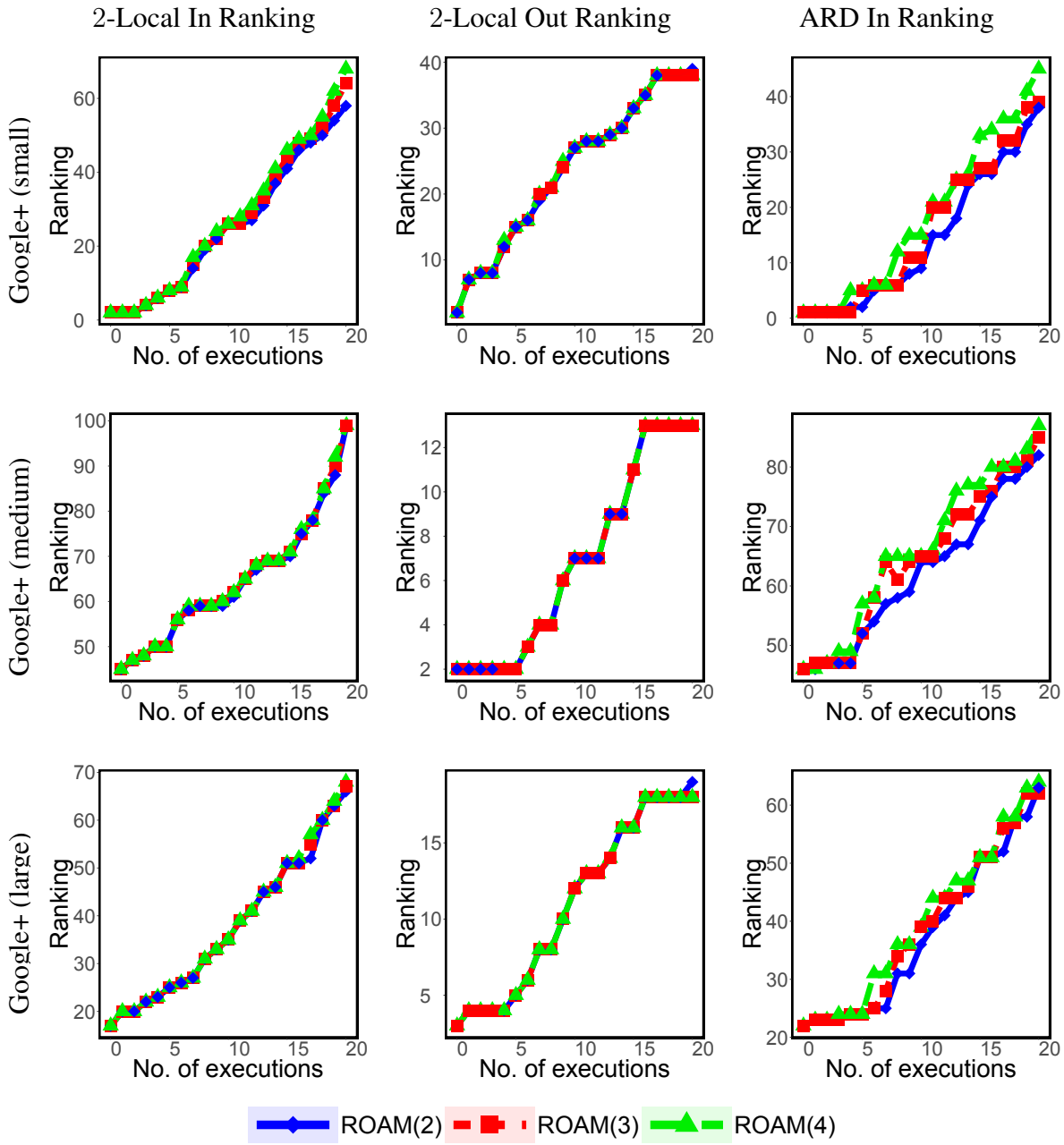
Supplementary Figure 67: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



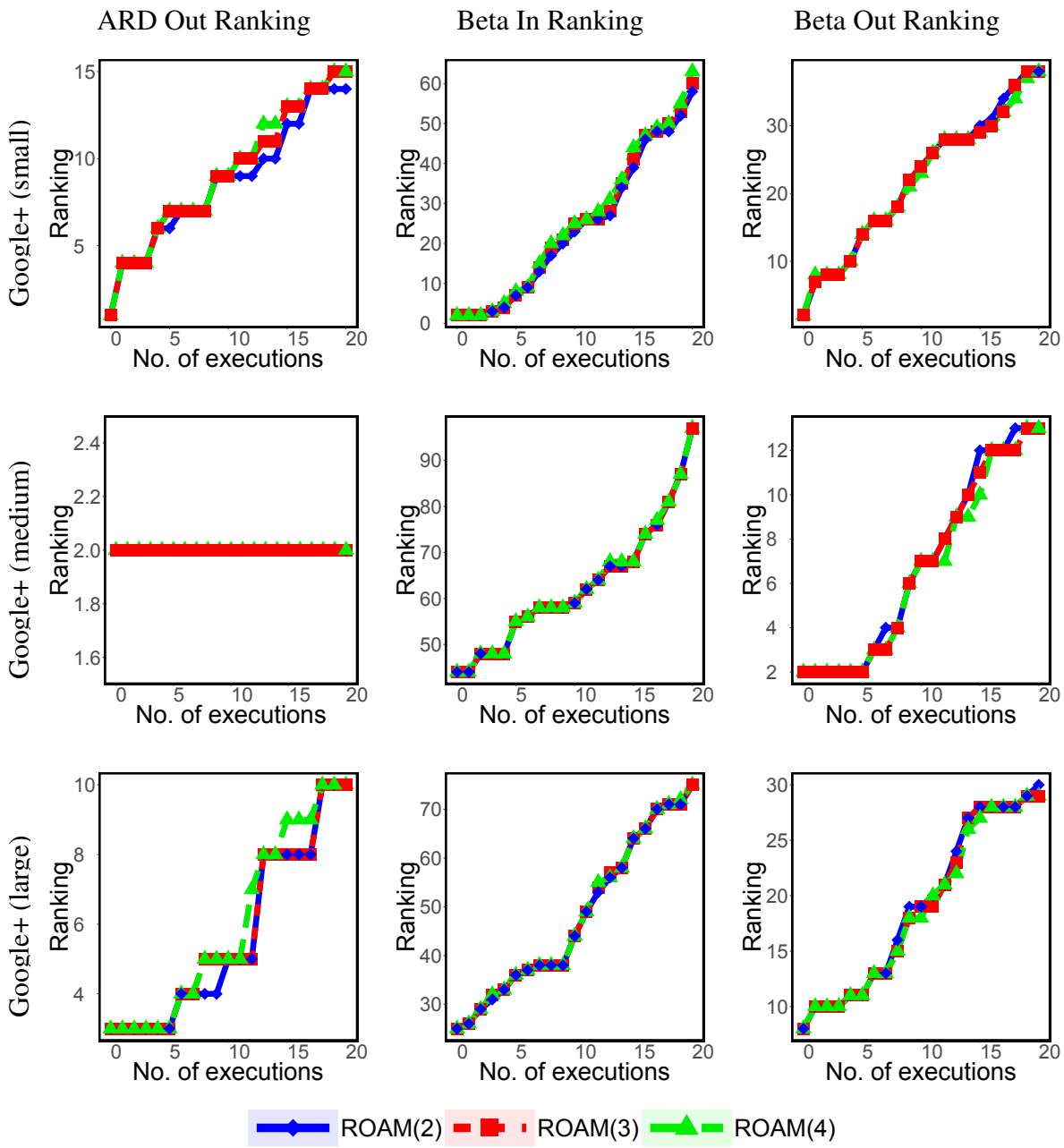
Supplementary Figure 68: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



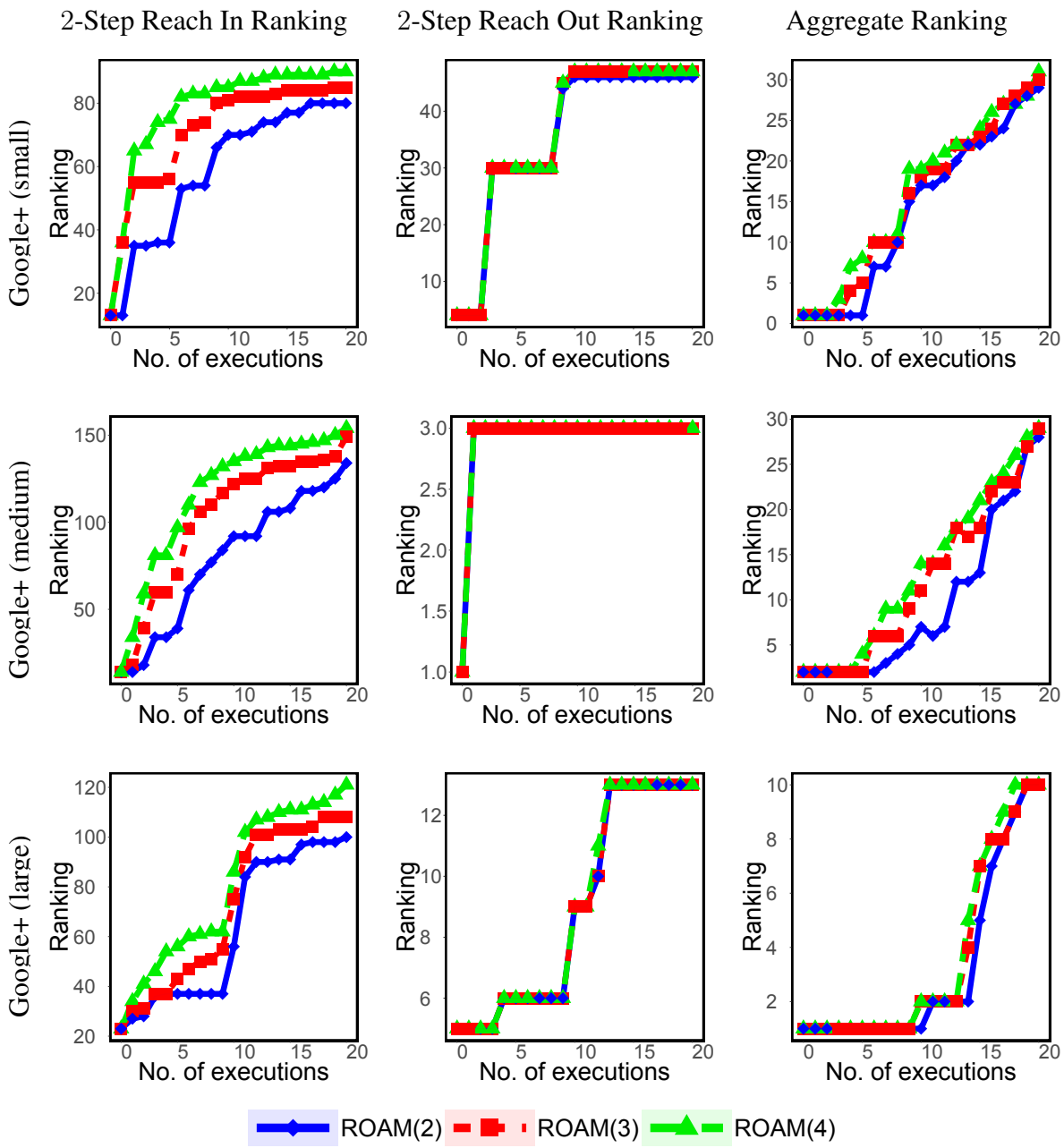
Supplementary Figure 69: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



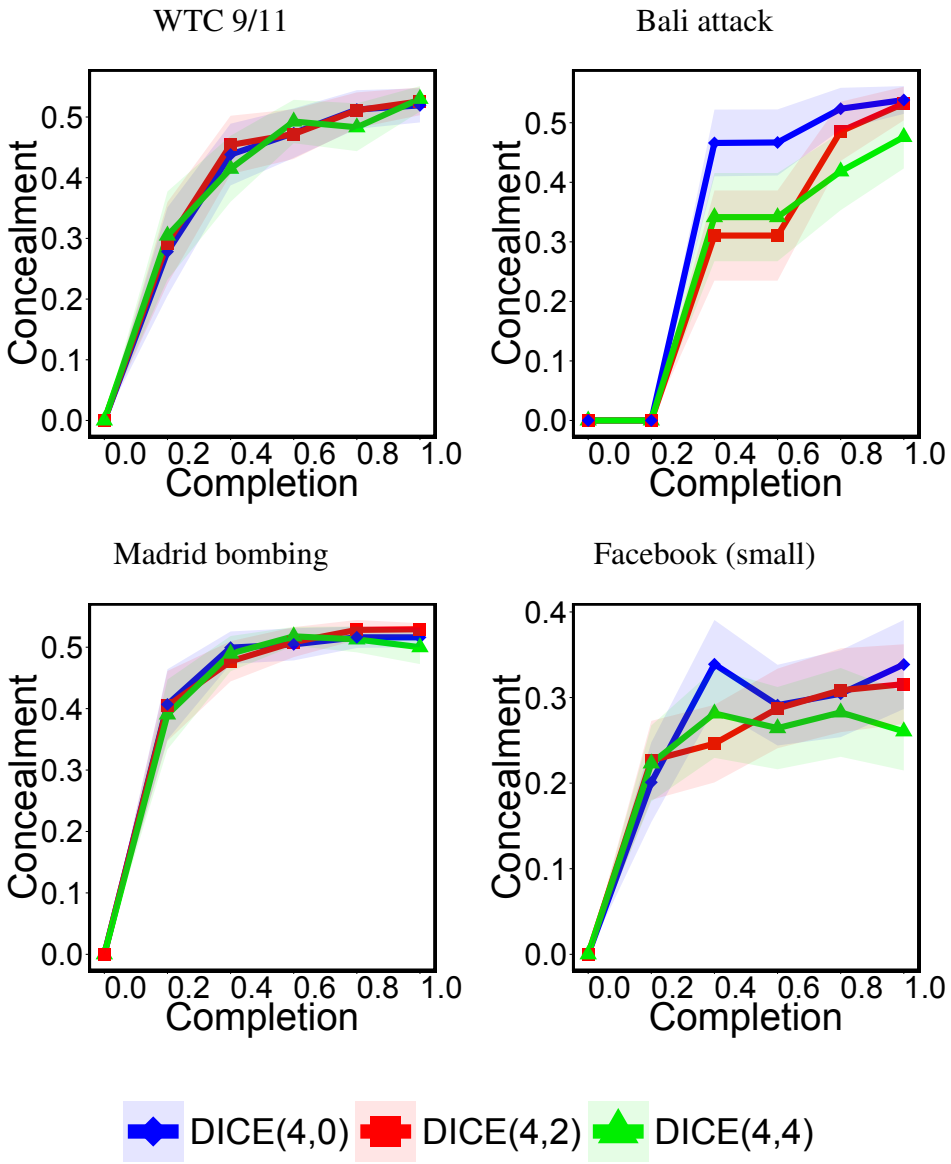
Supplementary Figure 70: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



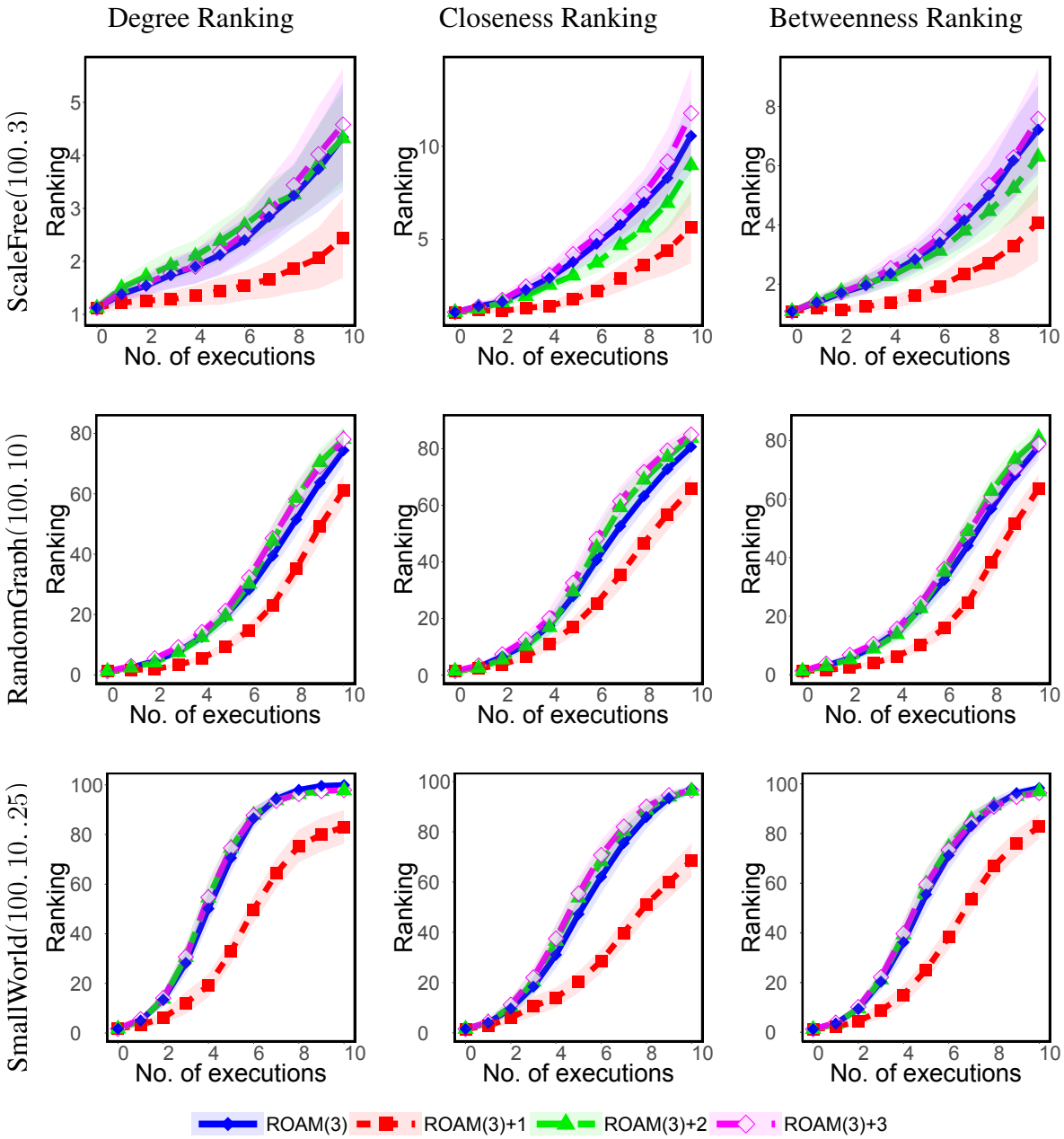
Supplementary Figure 71: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



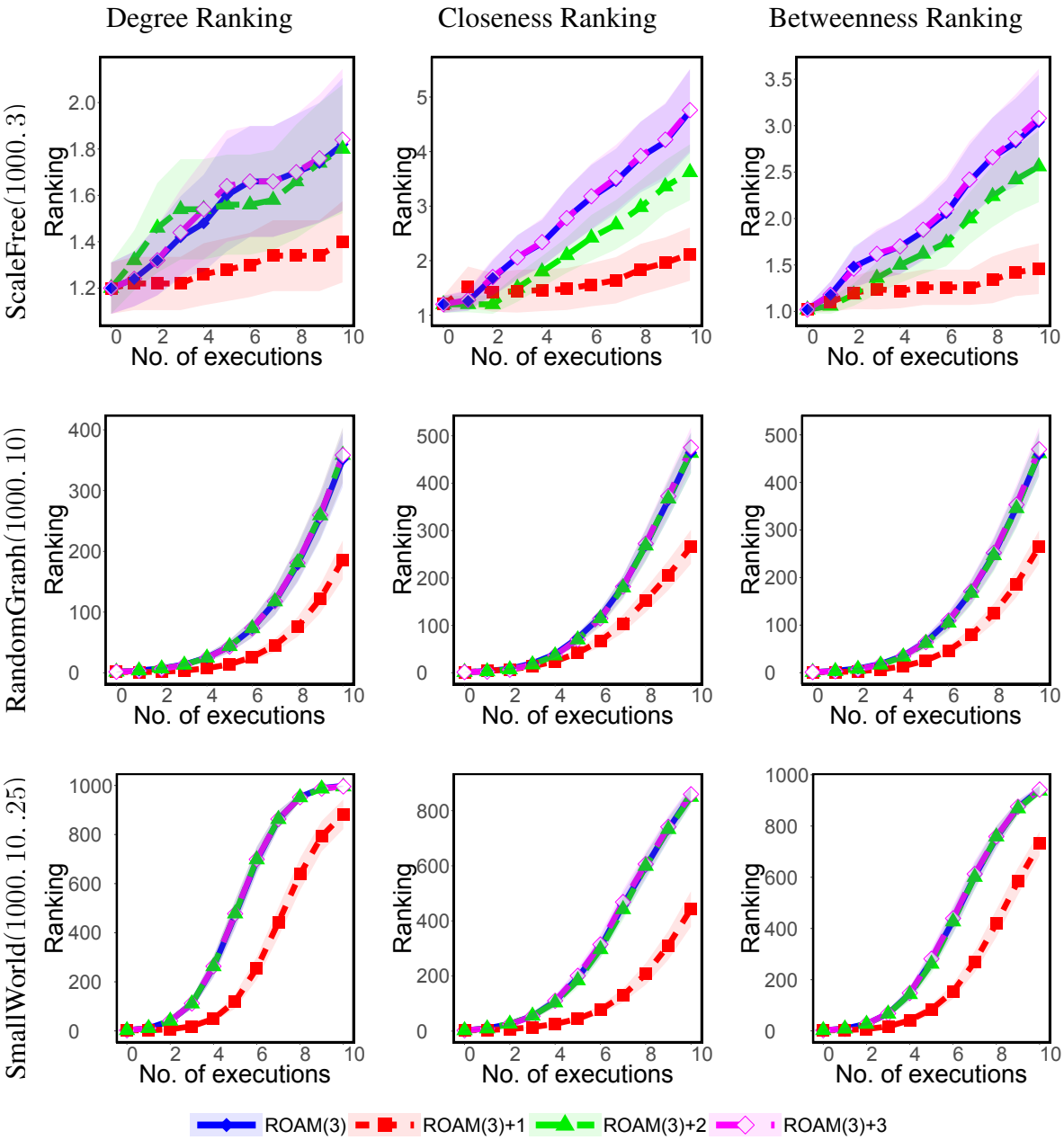
Supplementary Figure 72: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 2, 3, 4$, where b is the budget in each execution.



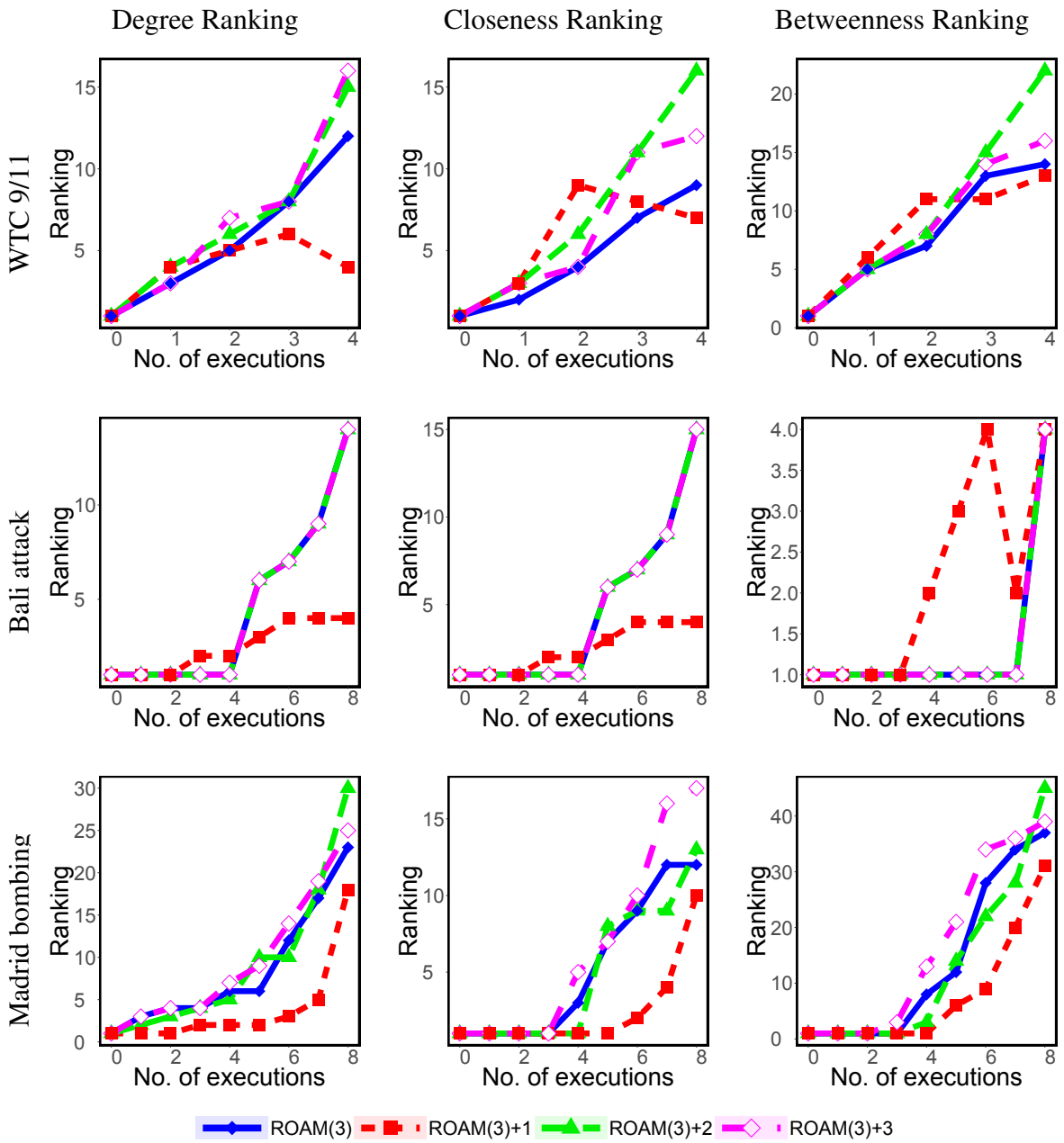
Supplementary Figure 73: Consecutive execution of DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value against Factions algorithm. Results are shown for DICE(b, d) : $b = 4; d \in \{0, 2, 4\}$, where b is the budget in each execution, and d is the number of removed internal edges.



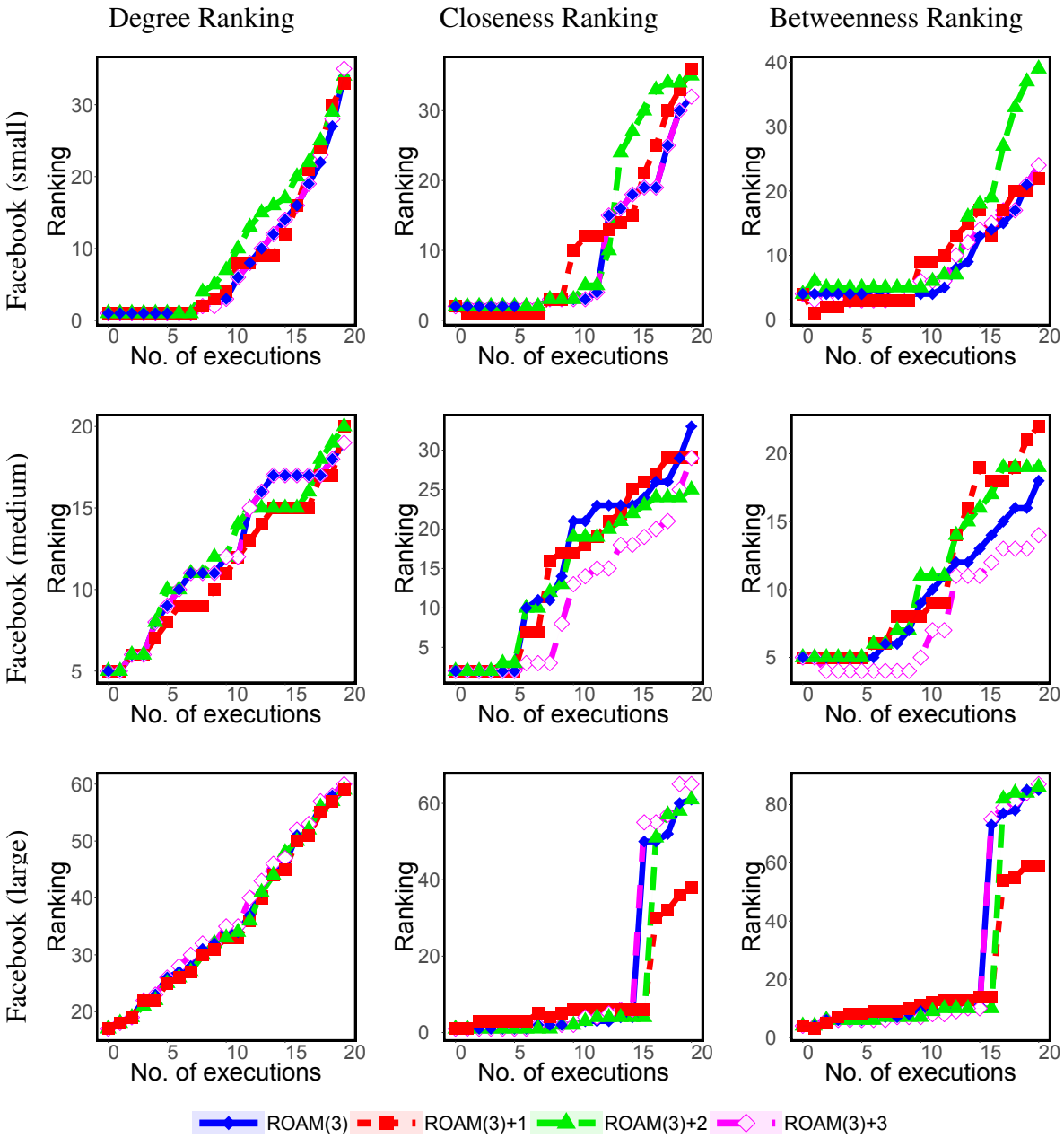
Supplementary Figure 74: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the ranking (according to the centrality measures) of the evader $v^{\dagger}$ when running ROAM(b) : $b = 3$ alone, and when running ROAM(b) : $b = 3$ along with three other evaders that are also running ROAM(b) : $b = 3$, and are exactly k steps away from $v^{\dagger}$; we refer to those latter experiments by writing: ROAM(b)+k.



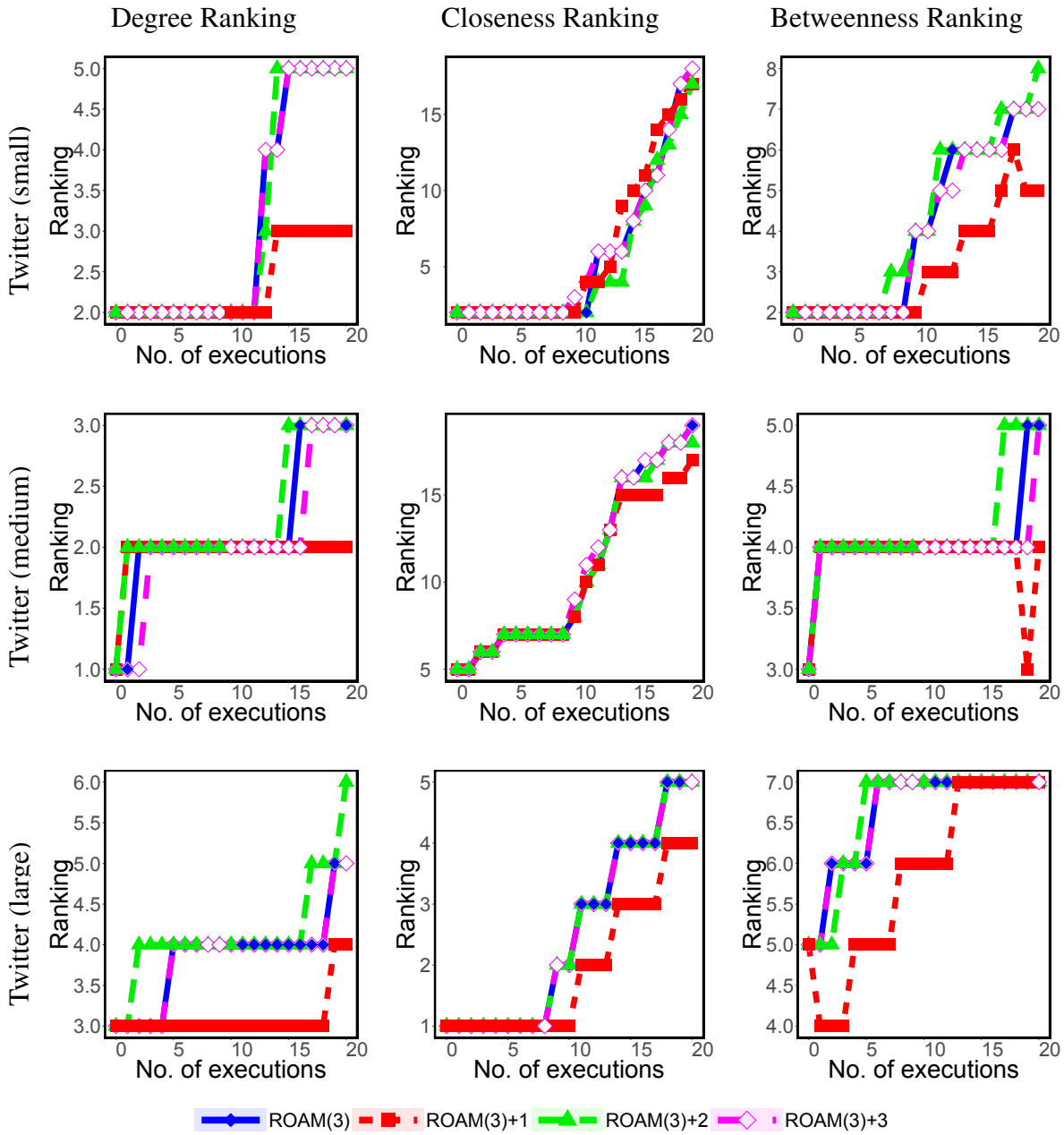
Supplementary Figure 75: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the ranking (according to the centrality measures) of the evader $v^{\dagger}$ when running ROAM(b) : $b = 3$ alone, and when running ROAM(b) : $b = 3$ along with three other evaders that are also running ROAM(b) : $b = 3$, and are exactly k steps away from $v^{\dagger}$; we refer to those latter experiments by writing: ROAM(b)+k.



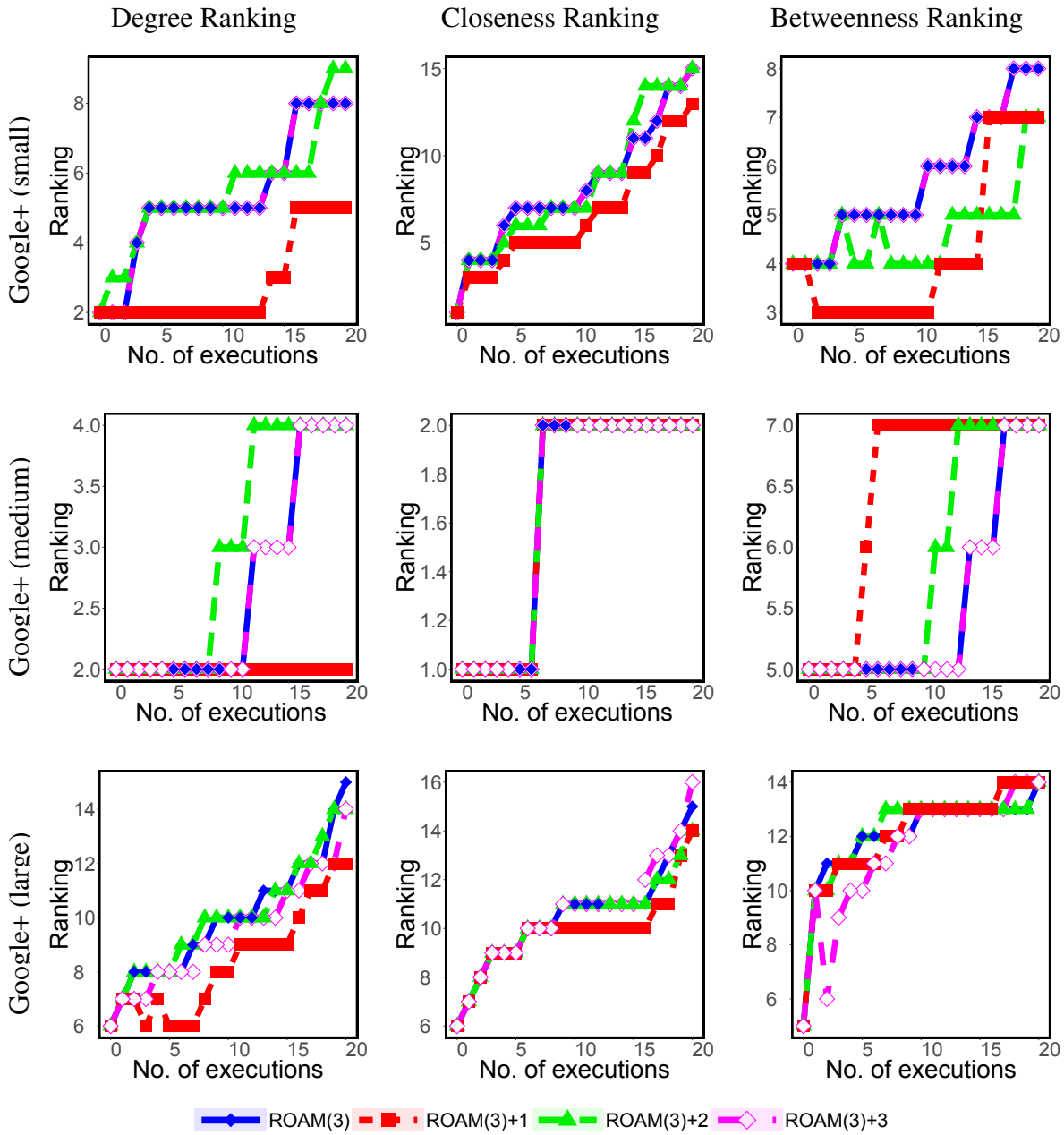
Supplementary Figure 76: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the ranking (according to the centrality measures) of the evader $v^{\dagger}$ when running ROAM(b) : $b = 3$ alone, and when running ROAM(b) : $b = 3$ along with three other evaders that are also running ROAM(b) : $b = 3$, and are exactly k steps away from $v^{\dagger}$; we refer to those latter experiments by writing: ROAM(b)+k.



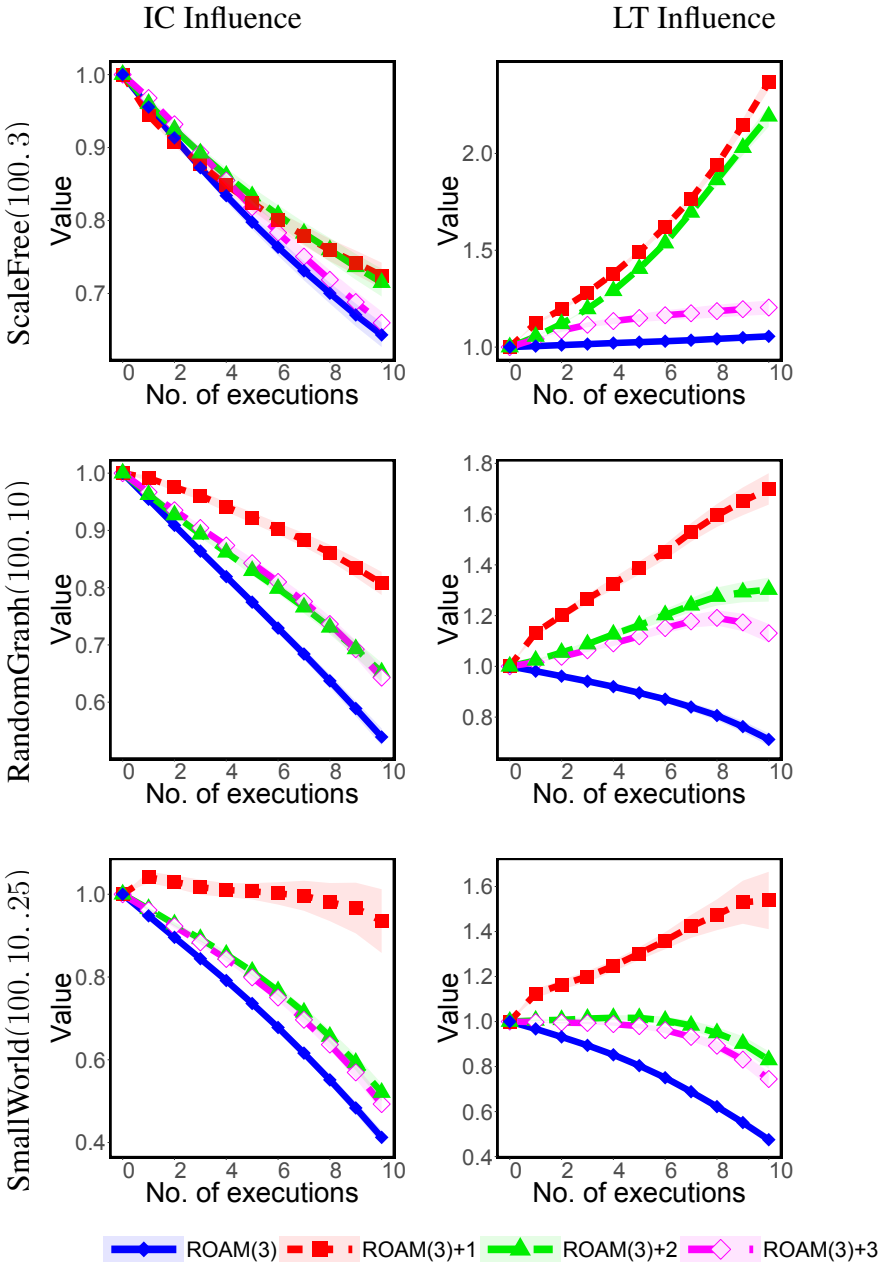
Supplementary Figure 77: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the ranking (according to the centrality measures) of the evader $v^{\dagger}$ when running ROAM(b) : $b = 3$ alone, and when running ROAM(b) : $b = 3$ along with three other evaders that are also running ROAM(b) : $b = 3$, and are exactly k steps away from $v^{\dagger}$; we refer to those latter experiments by writing: ROAM(b)+k.



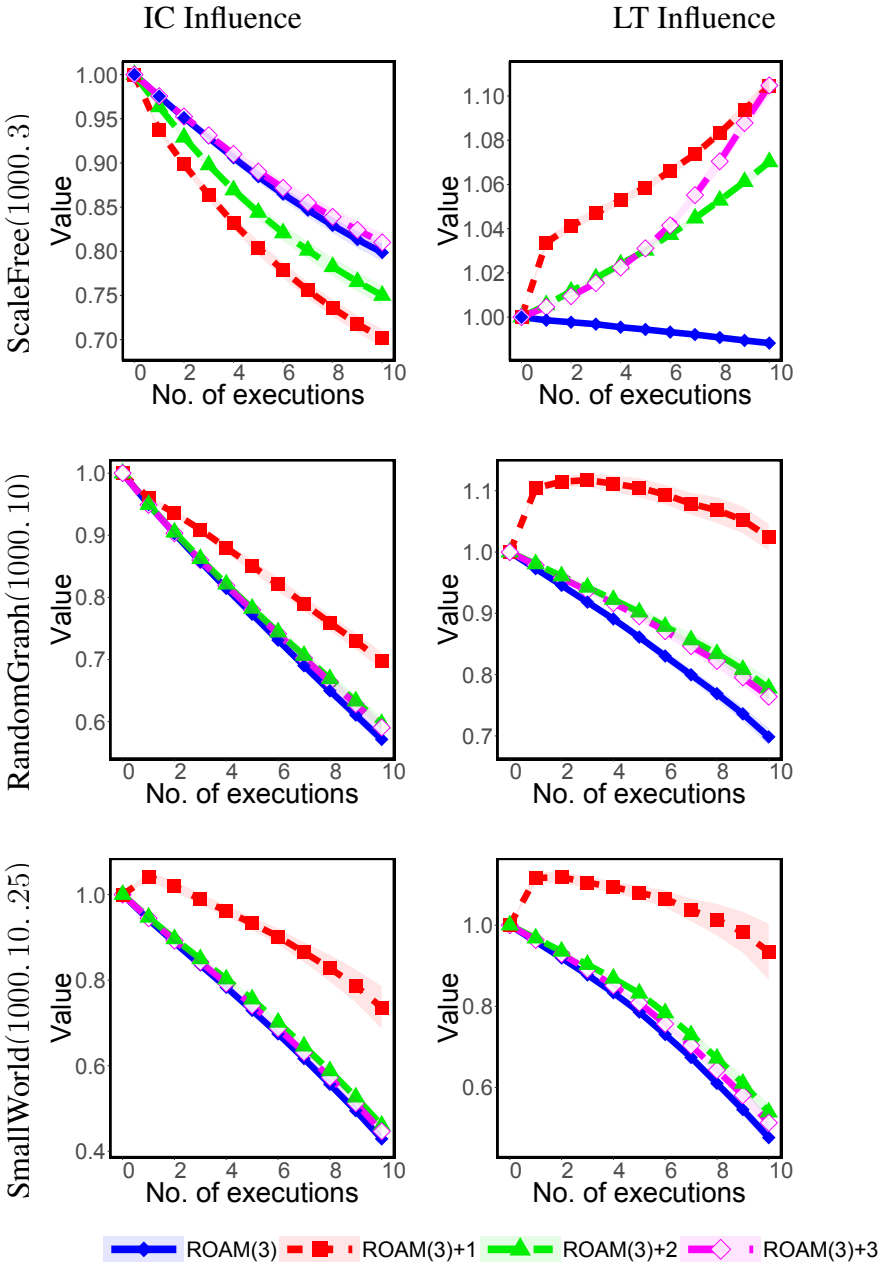
Supplementary Figure 78: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the ranking (according to the centrality measures) of the evader $v^{\dagger}$ when running ROAM(b) : $b = 3$ alone, and when running ROAM(b) : $b = 3$ along with three other evaders that are also running ROAM(b) : $b = 3$, and are exactly k steps away from $v^{\dagger}$; we refer to those latter experiments by writing: ROAM(b)+k.



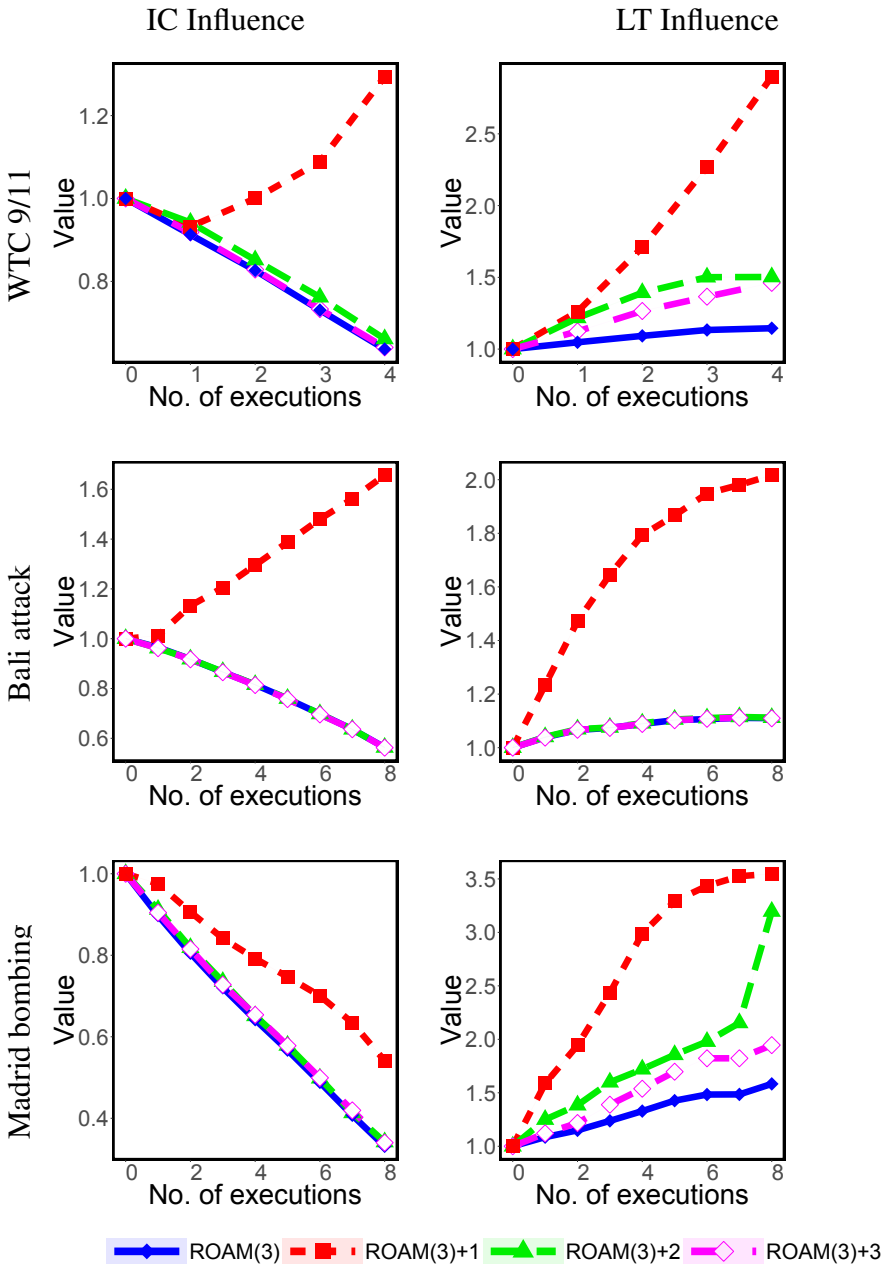
Supplementary Figure 79: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the ranking (according to the centrality measures) of the evader $v^{\dagger}$ when running ROAM(b) : $b = 3$ alone, and when running ROAM(b) : $b = 3$ along with three other evaders that are also running ROAM(b) : $b = 3$, and are exactly k steps away from $v^{\dagger}$; we refer to those latter experiments by writing: ROAM(b)+k.



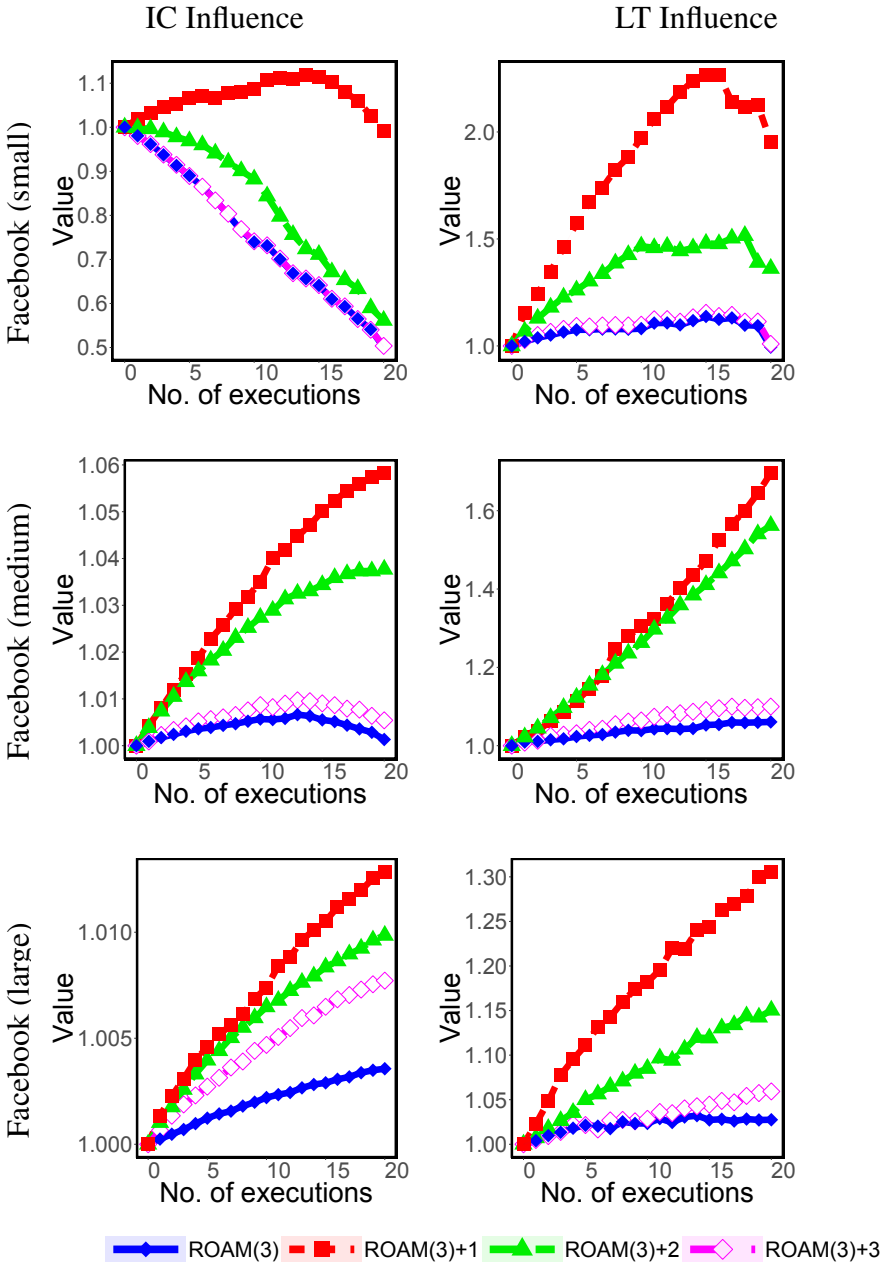
Supplementary Figure 80: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures the relative change in the influence value (according to the influence models) of the evader $v^\dagger$ when running ROAM(b) : $b = 3$ alone, and when running ROAM(b) : $b = 3$ along with three other evaders that are also running ROAM(b) : $b = 3$, and are exactly k steps away from $v^\dagger$; we refer to those latter experiments by writing: ROAM(b)+k.



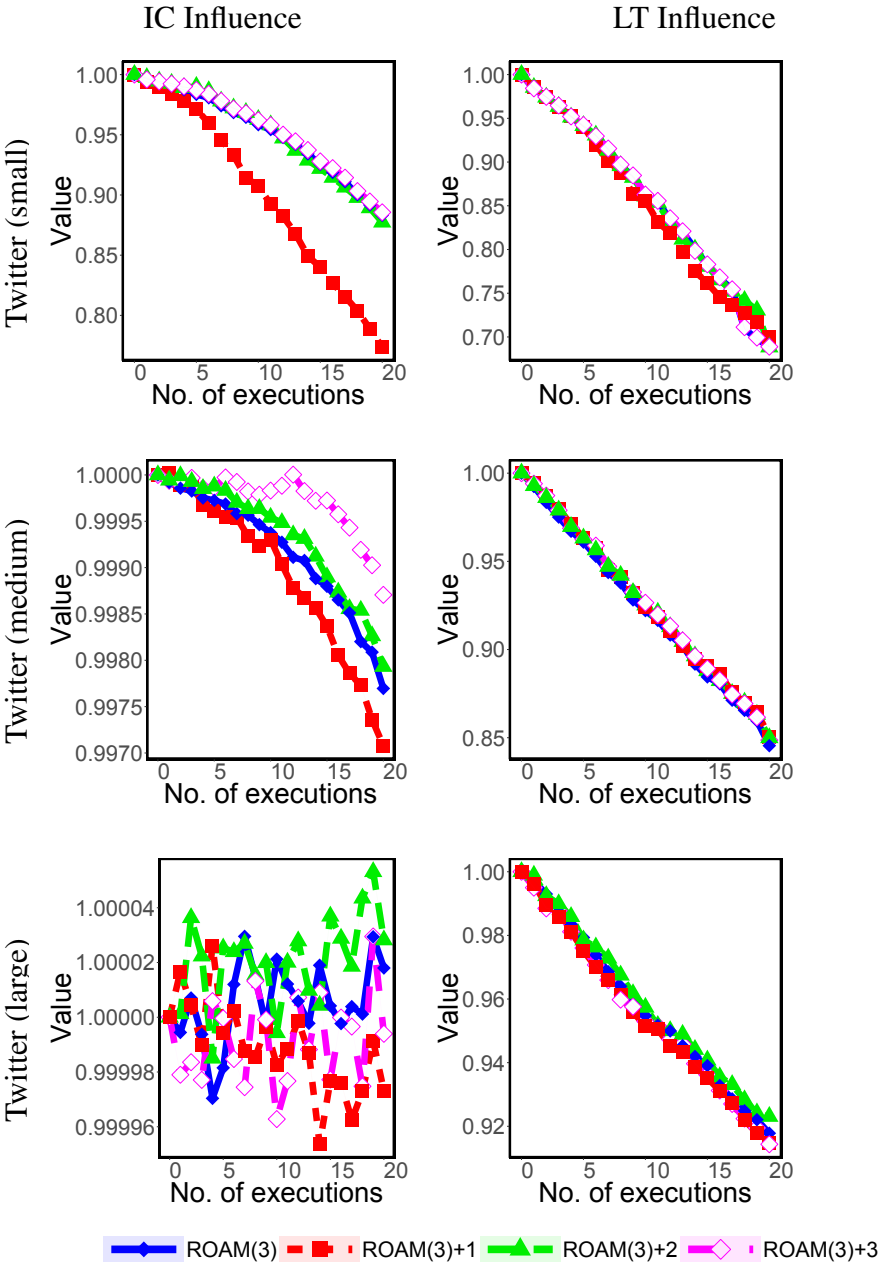
Supplementary Figure 81: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures the relative change in the influence value (according to the influence models) of the evader $v^{\dagger}$ when running ROAM(b) : $b = 3$ alone, and when running ROAM(b) : $b = 3$ along with three other evaders that are also running ROAM(b) : $b = 3$, and are exactly k steps away from $v^{\dagger}$; we refer to those latter experiments by writing: ROAM(b)+k.



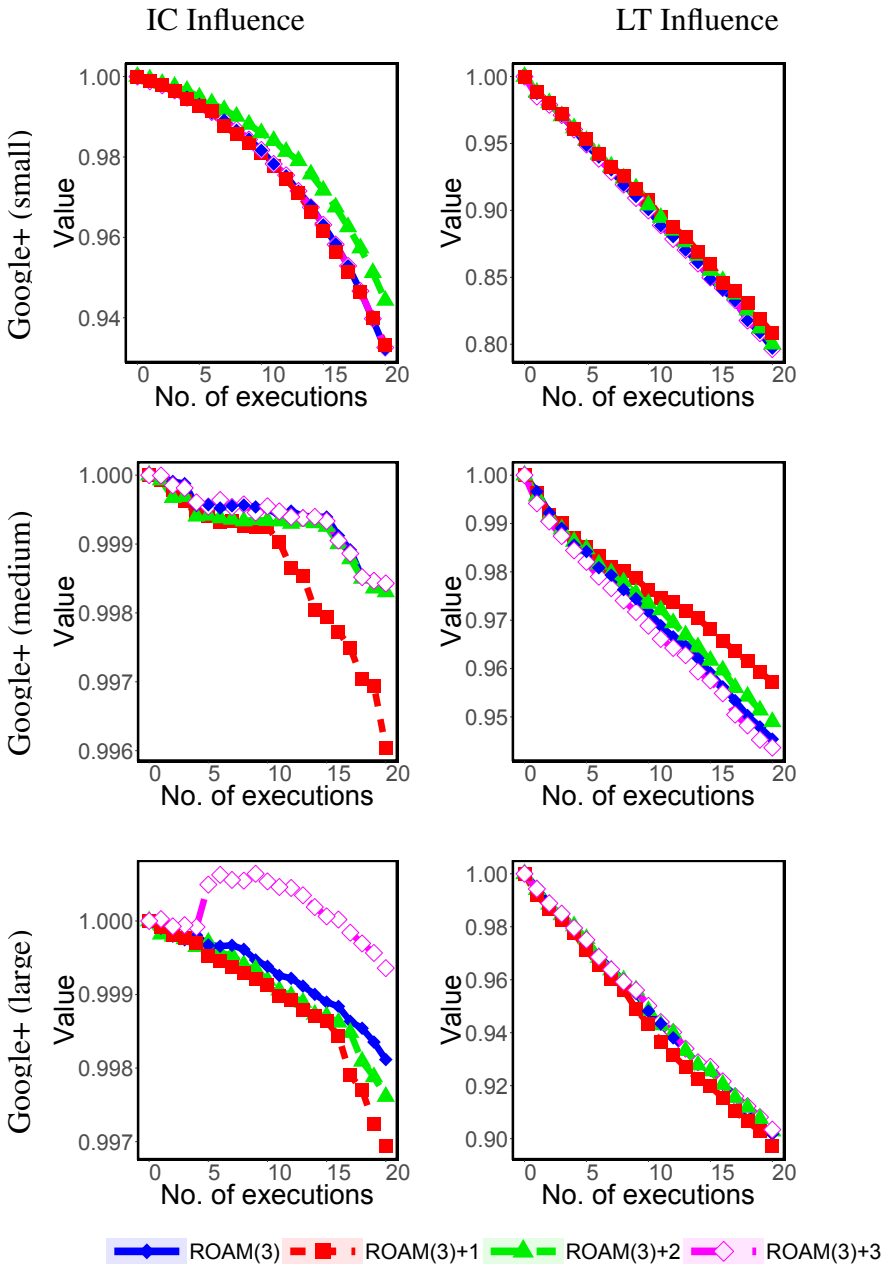
Supplementary Figure 82: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures the relative change in the influence value (according to the influence models) of the evader $v^{\dagger}$ when running ROAM(b) : $b = 3$ alone, and when running ROAM(b) : $b = 3$ along with three other evaders that are also running ROAM(b) : $b = 3$, and are exactly k steps away from $v^{\dagger}$; we refer to those latter experiments by writing: ROAM(b)+k.



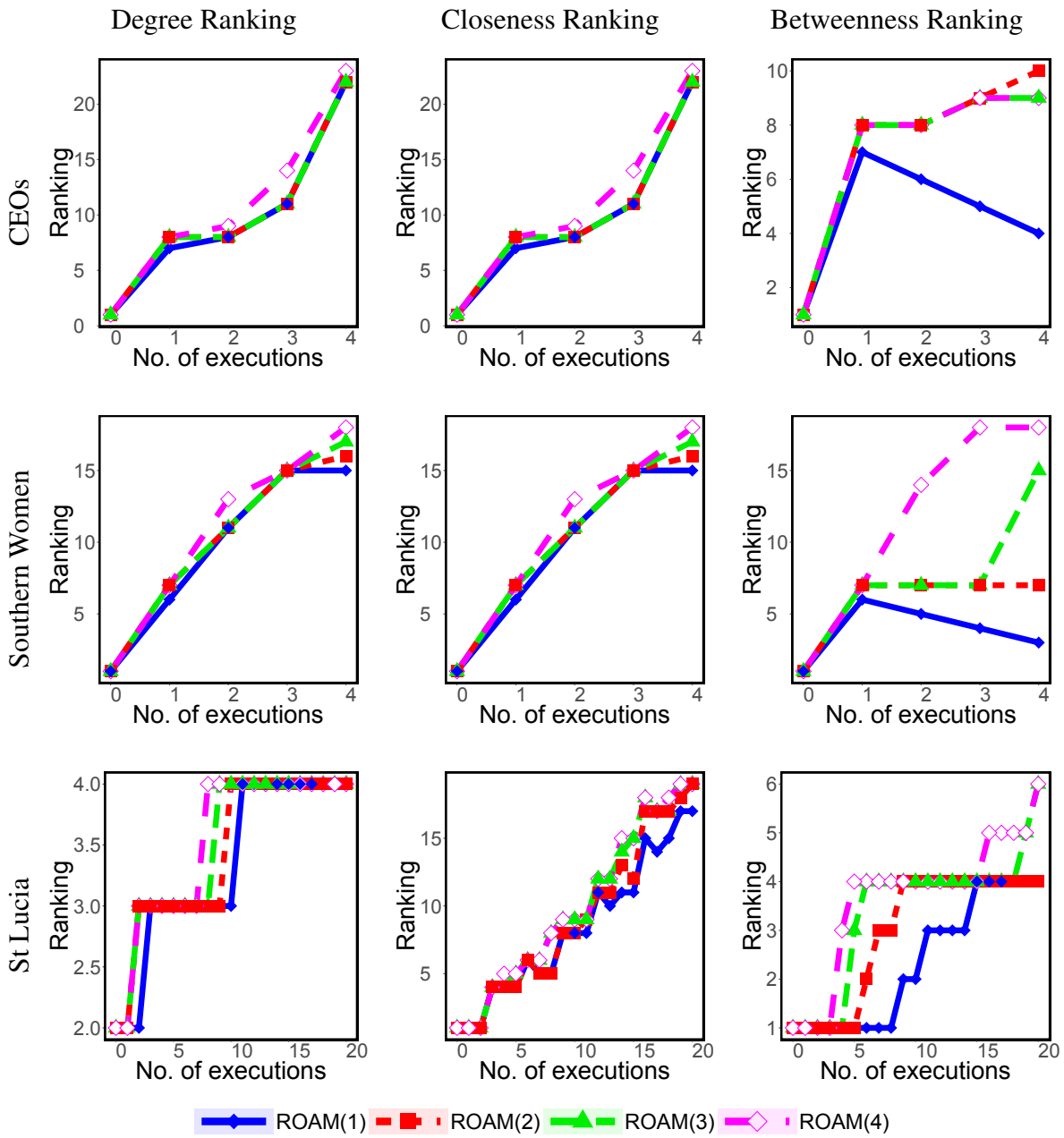
Supplementary Figure 83: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures the relative change in the influence value (according to the influence models) of the evader $v^{\dagger}$ when running ROAM(b) : $b = 3$ alone, and when running ROAM(b) : $b = 3$ along with three other evaders that are also running ROAM(b) : $b = 3$, and are exactly k steps away from $v^{\dagger}$; we refer to those latter experiments by writing: ROAM(b)+k.



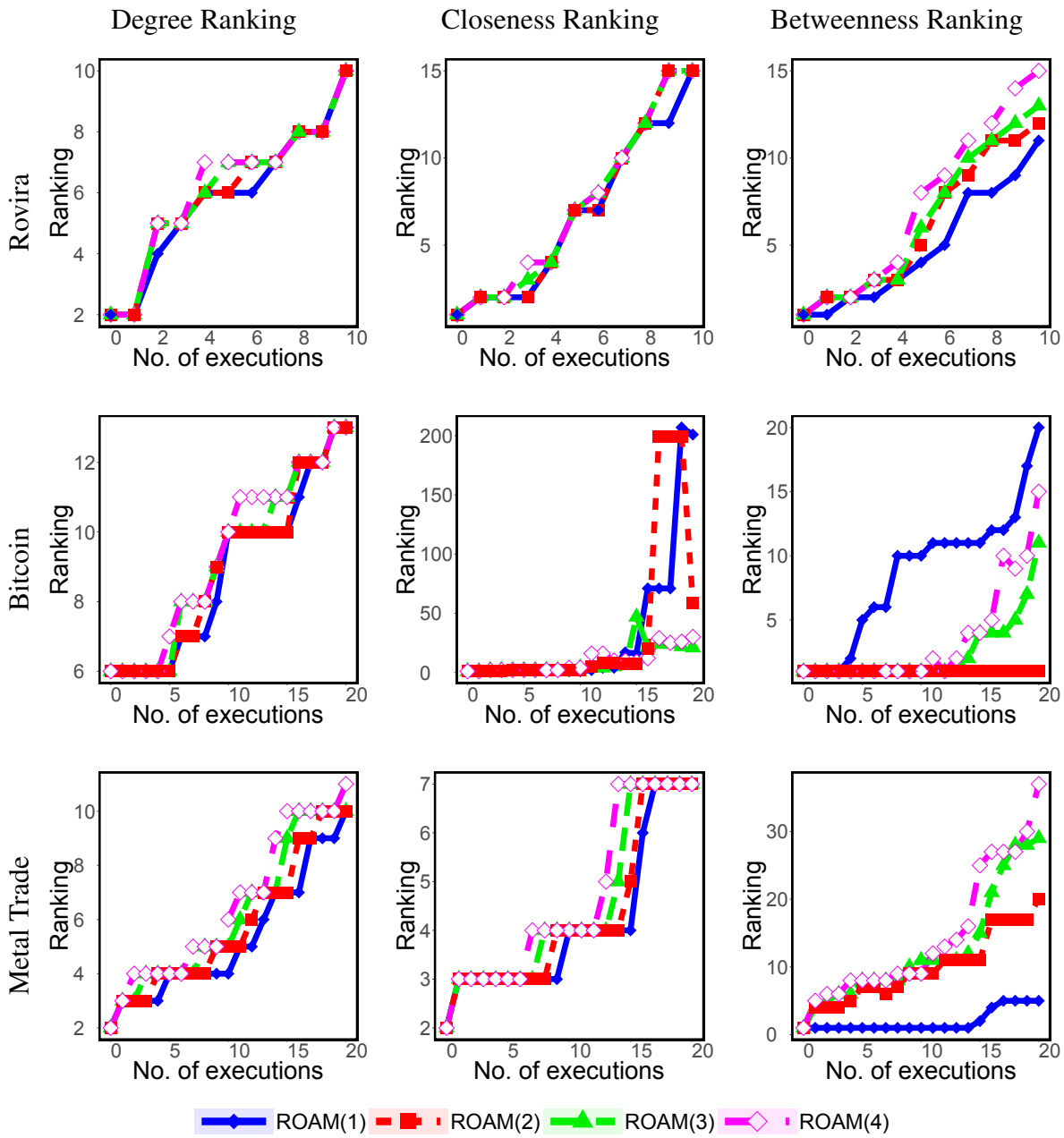
Supplementary Figure 84: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures the relative change in the influence value (according to the influence models) of the evader $v^{\dagger}$ when running ROAM(b) : $b = 3$ alone, and when running ROAM(b) : $b = 3$ along with three other evaders that are also running ROAM(b) : $b = 3$, and are exactly k steps away from $v^{\dagger}$; we refer to those latter experiments by writing: ROAM(b)+k.



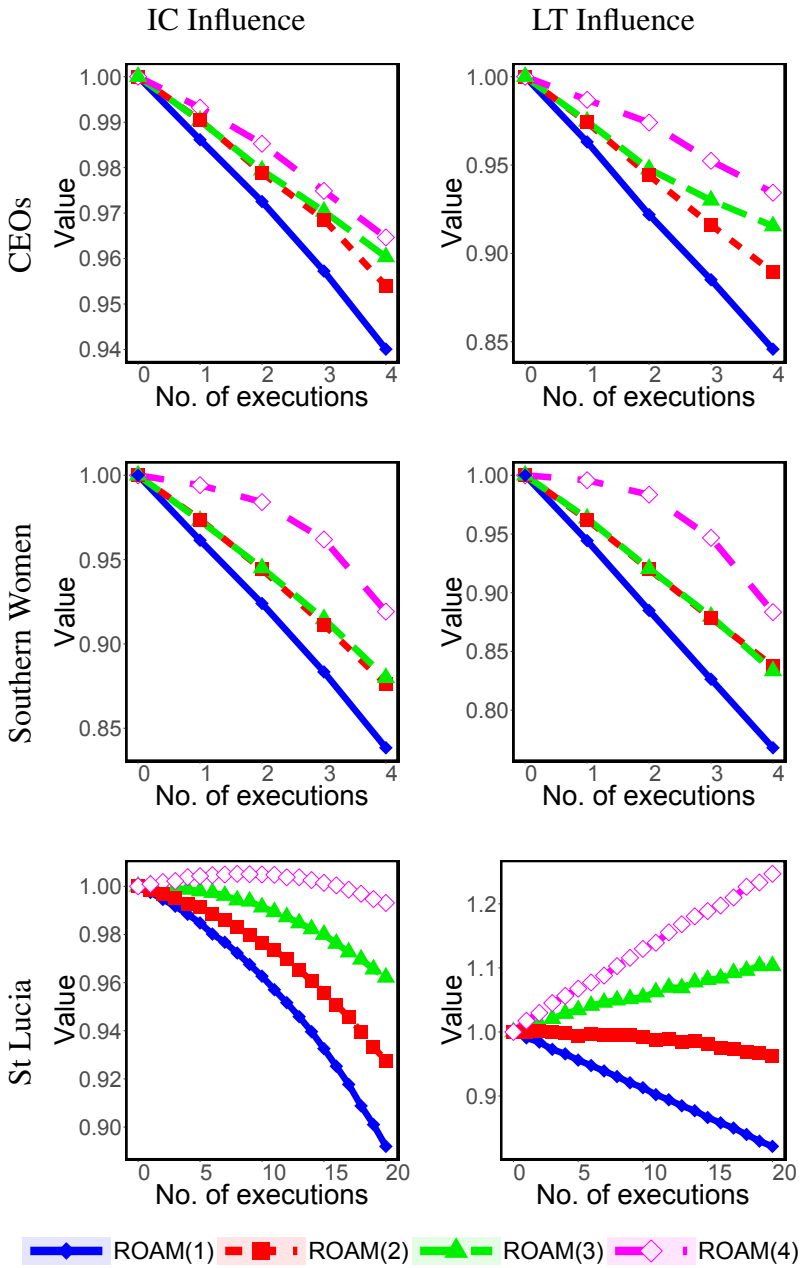
Supplementary Figure 85: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures the relative change in the influence value (according to the influence models) of the evader $v^{\dagger}$ when running ROAM(b) : $b = 3$ alone, and when running ROAM(b) : $b = 3$ along with three other evaders that are also running ROAM(b) : $b = 3$, and are exactly k steps away from $v^{\dagger}$; we refer to those latter experiments by writing: ROAM(b)+k.



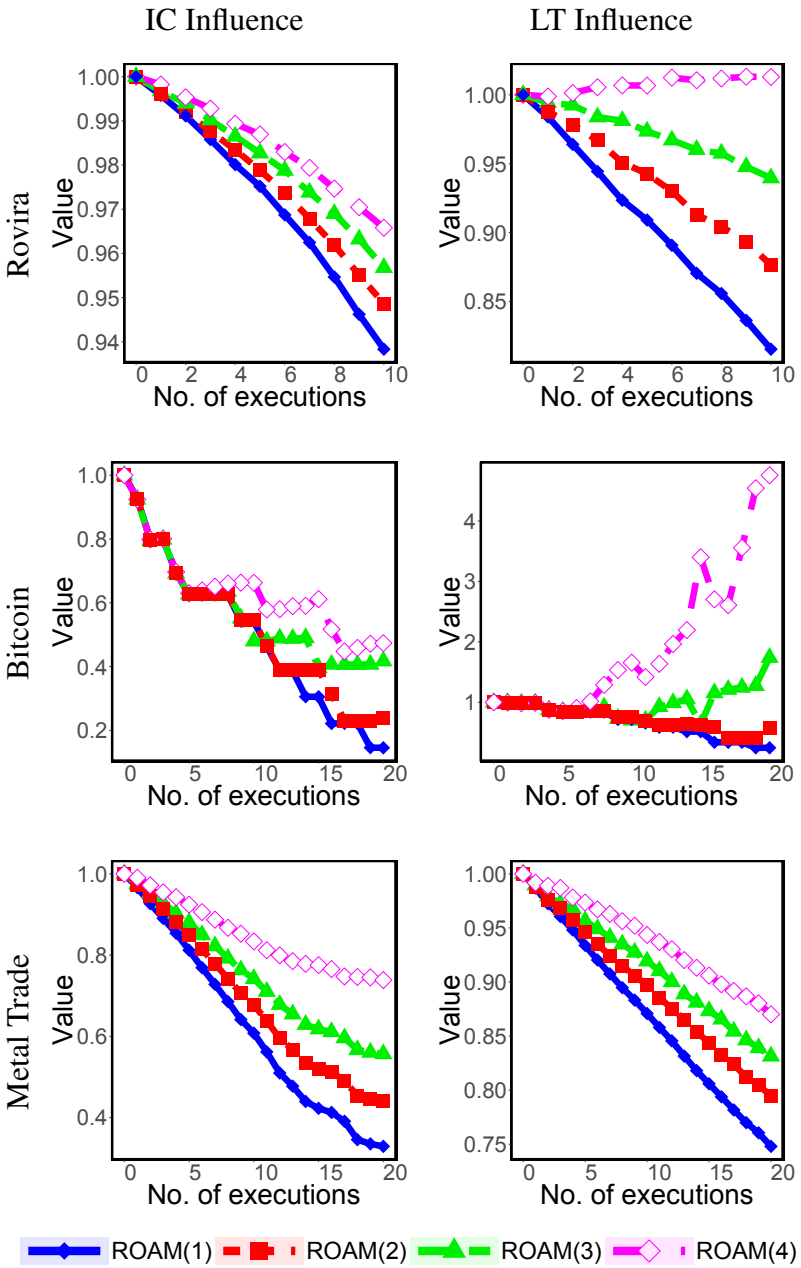
Supplementary Figure 86: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



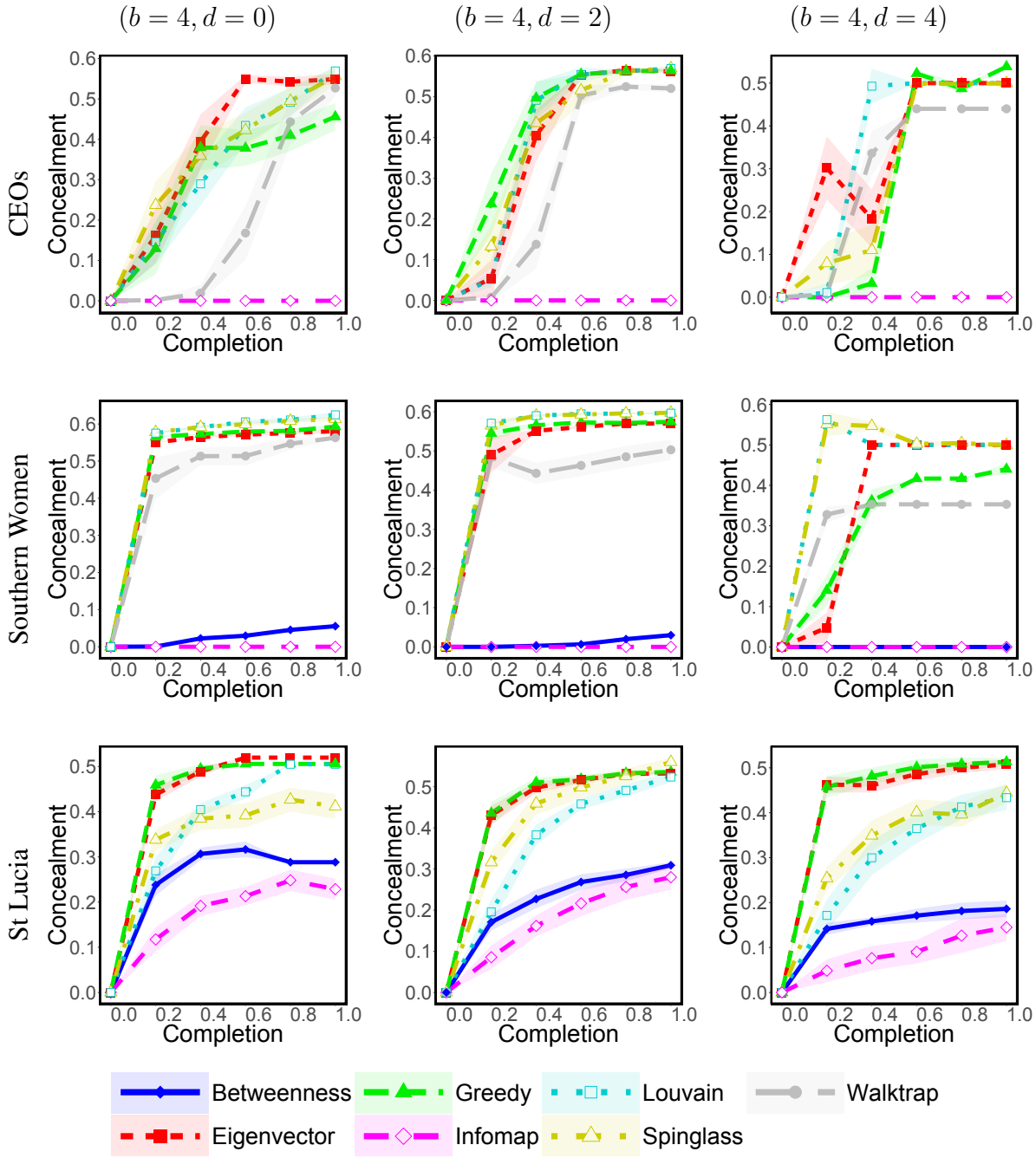
Supplementary Figure 87: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the evader's ranking (according to the centrality measures). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



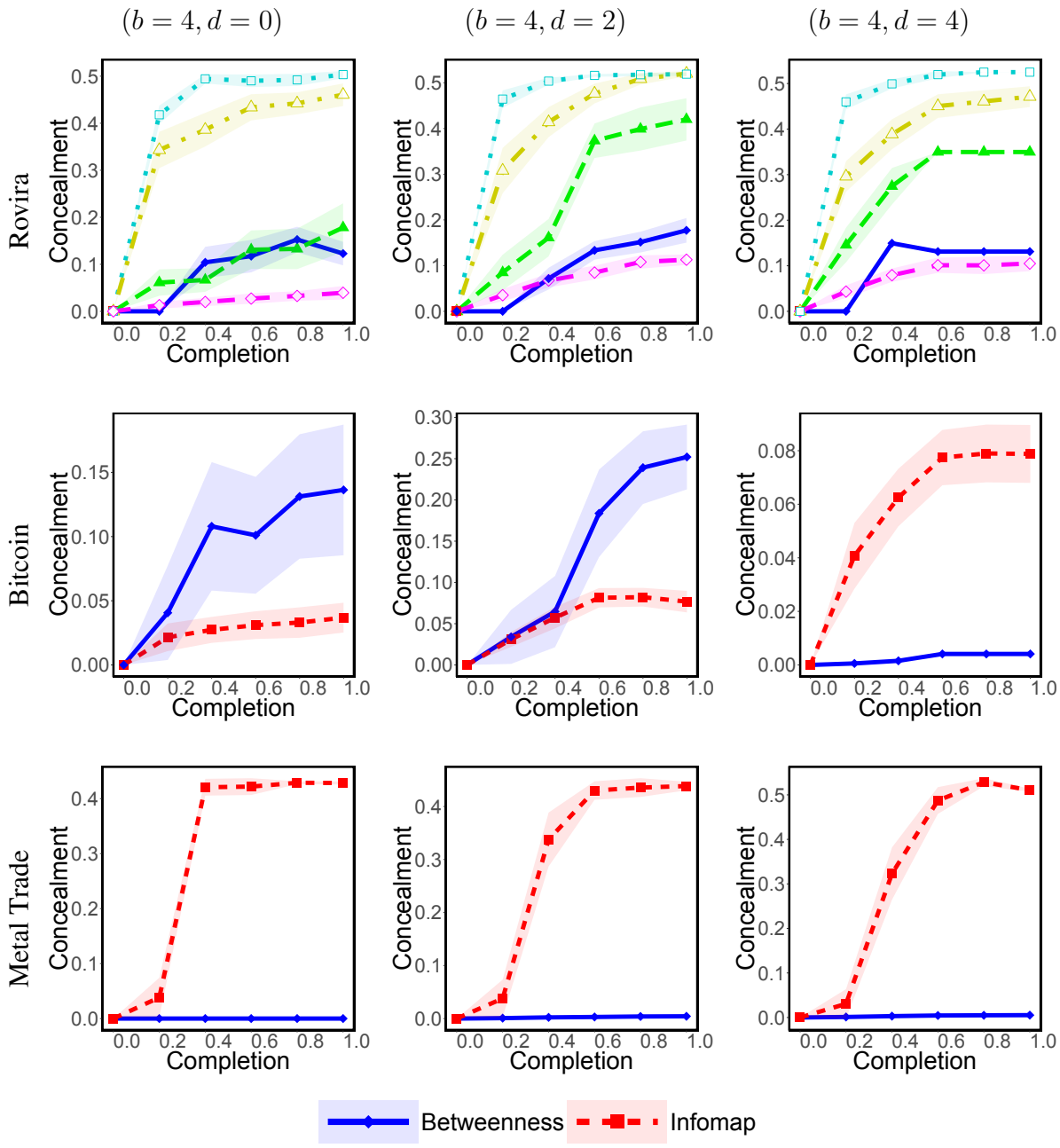
Supplementary Figure 88: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



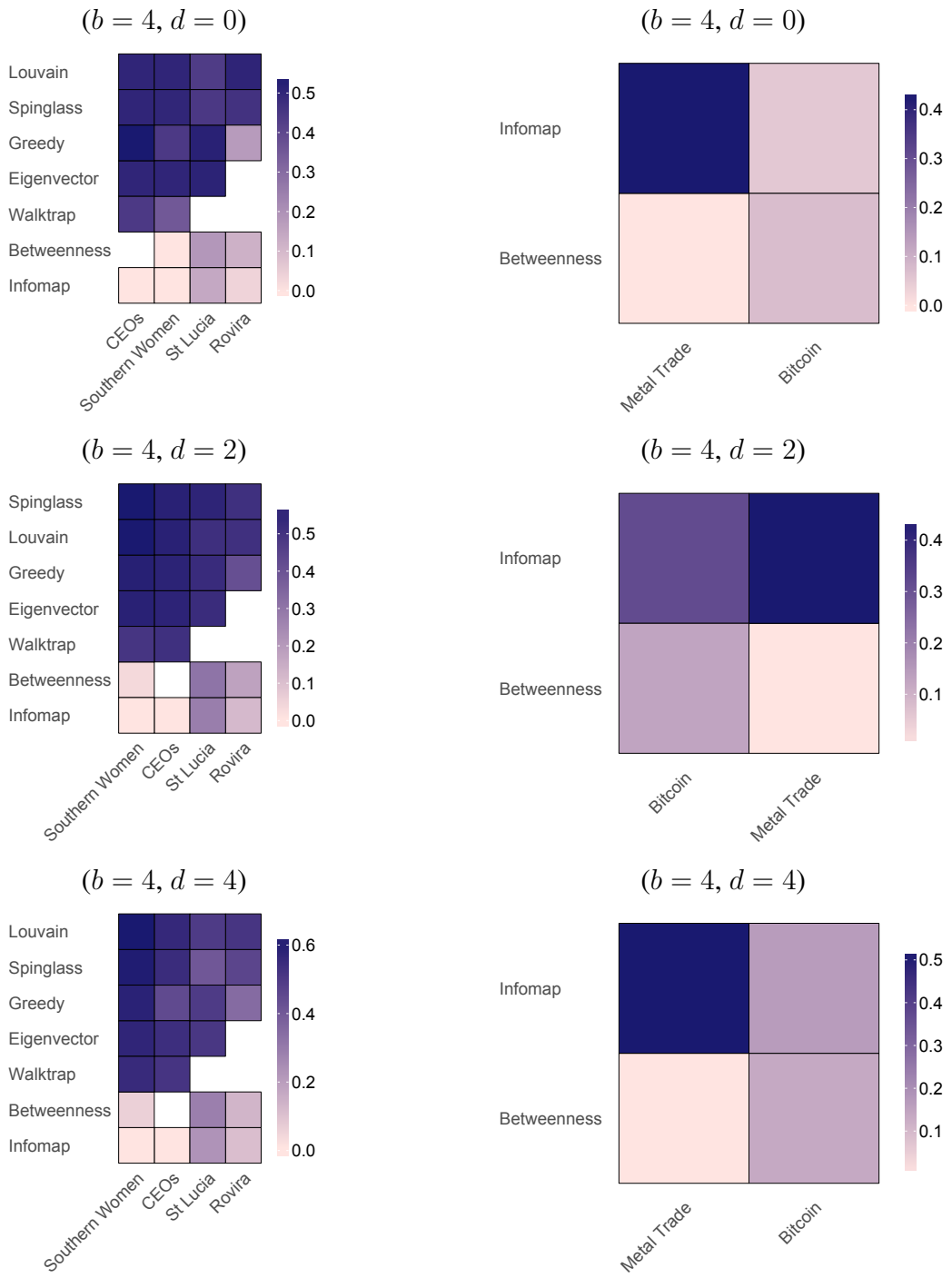
Supplementary Figure 89: Consecutive execution of ROAM (the x-axis represents the number of executions). Specifically, given different networks, the subfigures show the relative change in the evader's influence value (according to the influence models). Results are shown for ROAM(b) : $b = 1, 2, 3, 4$, where b is the budget in each execution.



Supplementary Figure 90: Consecutive execution of DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE$(b, d) : b = 4; d \in \{0, 2, 4\}$, where b is the budget in each execution, and d is the number of removed internal edges.



Supplementary Figure 91: Consecutive execution of DICE (the x-axis represents the completion of process). Specifically, given different networks, the subfigures show the community concealment measure value. Results are shown for DICE$(b, d) : b = 4; d \in \{0, 2, 4\}$, where b is the budget in each execution, and d is the number of removed internal edges.



Supplementary Figure 92: Average concealment-measure value in each experiment. The results for the directed networks are presented on the right, whereas the results for the undirected networks are presented on the left. Results are shown for DICE$(b, d) : b = 4; d \in \{0, 2, 4\}$, where b is the budget in each execution, and d is the number of removed internal edges.

Supplementary References

[1] J. M. Anthonisse. The rush in a graph. *Amsterdam: University of Amsterdam Mathematical Centre*, 1971.

[2] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

[3] M. A. Beauchamp. An improved index of centrality. *Behavioral Science*, 10(2):161–163, 1965.

[4] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

[5] P. Bonacich. Power and centrality: A family of measures. *American journal of sociology*, 92(5):1170–1182, 1987.

[6] S. P. Borgatti and M. G. Everett. Network analysis of 2-mode data. *Social networks*, 19(3):243–269, 1997.

[7] S. P. Borgatti, M. G. Everett, and L. C. Freeman. Ucinet for windows: Software for social network analysis. 2002.

[8] A. Clauset, M. E. Newman, and C. Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[9] G. Csardi and T. Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

[10] A. Davis, B. B. Gardner, and M. R. Gardner. *Deep South: A social anthropological study of caste and class*. Univ of South Carolina Press, 2009.

[11] H. Du, D. R. White, Y. Ren, and S. Li. A normalized and a hybrid modularity. *Draft paper, eclectic. ss. uci. edu/˜ drwhite/links2pdf. htm*, 2008.

[12] P. Erdős and A. Rényi. On random graphs i. *Publ. Math. Debrecen*, 6:290–297, 1959.

[13] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[14] L. C. Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1979.

[15] J. Galaskiewicz. *Social organization of an urban grants economy: A study of business philanthropy and nonprofit organizations*. Elsevier, 2016.

[16] F. Glover. Tabu search, part i. *ORSA Journal on computing*, 1(3):190–206, 1989.

[17] J. Goldenberg, B. Libai, and E. Muller. Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata. *Academy of Marketing Science Review*, 9(3):1–18, 2001.

[18] R. Guimera, L. Danon, A. Diaz-Guilera, F. Giralt, and A. Arenas. Self-similar community structure in a network of human interactions. *Physical review E*, 68(6):065103, 2003.

[19] B. Hayes. Connecting the dots can the tools of graph theory and social-network studies unravel the next big plot? *American Scientist*, 94(5):400–404, 2006.

[20] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[21] V. Krebs. Mapping networks of terrorist cells. *Connections*, 24:43–52, 2002.

[22] J. Leskovec and J. J. Mcauley. Learning to discover social circles in ego networks. In *Advances in neural information processing systems*, pages 539–547, 2012.

[23] M. E. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical review E*, 74(3):036104, 2006.

[24] M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[25] P. Pons and M. Latapy. Computing communities in large networks using random walks. In *Computer and Information Sciences-ISCIS 2005*, pages 284–293. Springer, 2005.

[26] J. Reichardt and S. Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1):016110, 2006.

[27] F. Reid and M. Harrigan. An analysis of anonymity in the bitcoin system. In *Security and privacy in social networks*, pages 197–223. Springer, 2013.

[28] M. Rosvall, D. Axelsson, and C. T. Bergstrom. The map equation. *The European Physical Journal Special Topics*, 178(1):13–23, 2010.

[29] M. E. Shaw. Group structure and the behavior of individuals in small groups. *The Journal of Psychology*, 38(1):139–149, 1954.

[30] D. A. Smith and D. R. White. Structure and dynamics of the global economy: network analysis of international trade 1965–1980. *Social forces*, 70(4):857–893, 1992.

[31] B. Thomas, R. Jurdak, K. Zhao, and I. Atkinson. Diffusion in colocation contact networks: The impact of nodal spatiotemporal dynamics. *PloS one*, 11(8):e0152624, 2016.

[32] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *nature*, 393(6684):440–442, 1998.

[33] J. Xie, S. Kelley, and B. K. Szymanski. Overlapping community detection in networks: The state-of-the-art and comparative study. *ACM Computing Surveys (csur)*, 45(4):43, 2013.