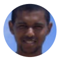
Group Centrality Maximization via Network Design

Conference Paper · May 2018

DOI: 10.1137/1.9781611975321.14

Sourav Medya
Northwestern University
29 PUBLICATIONS **61** CITATIONS
SEE PROFILE



Arlei Silva
University of California, Santa Barbara
37 PUBLICATIONS **500** CITATIONS
SEE PROFILE

Prithwish Basu
Raytheon Technologies
77 PUBLICATIONS **1,120** CITATIONS
SEE PROFILE

Ananthram Swami
Institute of Electrical and Electronics Engineers
455 PUBLICATIONS **14,681** CITATIONS
SEE PROFILE

Group Centrality Maximization via Network Design

Sourav Medya[*] Arlei Silva[†] Ambuj Singh[‡] Prithwish Basu[§] Ananthram Swami[¶]

Abstract

Network centrality plays an important role in many applications. Central nodes in social networks can be influential, driving opinions and spreading news or rumors. In hyperlinked environments, such as the Web, where users navigate via clicks, central content receives high traffic, becoming target for advertising campaigns. While there is an extensive amount of work on centrality measures and their efficient computation, controlling nodes' centrality via network updates is a more recent and challenging task. Performing minimal modifications to a network to achieve a desired property falls under the umbrella of network design problems. This paper is focused on improving group (coverage and betweenness) centrality, which is a function of the shortest paths passing through a set of nodes, by adding edges to the network. Several variations of the problem, which are NP-hard as well as APX-hard, are introduced. We present a greedy algorithm, and even faster sampling algorithms, for group centrality maximization with theoretical quality guarantees under realistic constraints. The experimental results show that our sampling algorithms outperform the best baseline solution in terms of centrality by up to 5 times while being 2-3 orders of magnitude faster than our greedy approach.

1 Introduction

Network design is a recent area of study focused on modifying or redesigning a network in order to achieve a desired property [8, 26]. As networks become a popular framework for modeling complex systems (e.g. VLSI, transportation, communication, society), network design provides key controlling capabilities over these systems, especially when resources are constrained. Existing work has investigated the optimization of global network properties, such as minimum spanning tree [12], shortest-path distances [13, 7, 16], diameter [6], and information diffusion-related metrics [11, 23] via a few local (e.g. vertex, edge-level) upgrades. Due to the large scale of real networks, computing a global network property becomes time-intensive. For instance, it is prohibitive to compute all-pairs shortest paths in large networks. As a consequence, design problems are inherently challenging. Moreover, because of the combinatorial nature of these local modifications, network design problems are often NP-hard, and thus, require the development of efficient approximation algorithms.

We focus on a novel network design problem, that improves the *group centrality*. Given a node v, its coverage centrality is the number of distinct node pairs for which a shortest path passes through v, whereas its betweenness centrality is the sum of the fraction of shortest paths between all distinct pair of nodes passing through v. The centrality of a group X is a function of the shortest paths that go through members of X [25]. *Our goal is to maximize group centrality, for a target group of nodes, via a small number of edge additions.*

There are several applications for group centrality optimization. Broadly speaking, whenever computing the centrality of a single node, or a group of nodes, is a problem of interest, one might as well pose the question of how to improve the centrality of one or more nodes. For instance, in online advertising, links can be added to boost the traffic towards a target set of Web pages. In a professional network, such as *LinkedIn*, the centrality of some users (e.g. employees of a given company) might be increased via connection recommendations/advertising. In military settings, where networks might include adversarial elements, inducing the flow of information towards key agents can enhance communication and decision making [21].

From a theoretical standpoint, for any objective function of interest, we can define a *search* and a corresponding *design* problem. In this paper, we show that, different from its search version [25], group centrality maximization cannot be approximated by a simple greedy algorithm. Furthermore, we study several variations of the problem and show that, under two realistic constraints, the problem has a constant factor approximation algorithm. In fact, we are able to prove that our approximation for the constrained problem is *optimal*, in the sense that the best algorithm cannot achieve a better approximation than ours. In order to scale our

[*] University of California, Santa Barbara (medya@cs.ucsb.edu)

[†] University of California, Santa Barbara (arlei@cs.ucsb.edu)

[‡] University of California, Santa Barbara (ambuj@cs.ucsb.edu)

[§] Raytheon BBN Technologies, Cambridge (prithwish.basu@raytheon.com)

[¶] Army Research Laboratory, Adelphi (ananthram.swami.civ@mail.mil) greedy solution to large datasets, we also propose efficient sampling schemes, with approximation guarantees, for group centrality maximization.

Our Contributions. The main contributions of this paper can be summarized as follows:

- We study a novel general network design problem, the group centrality optimization, and prove that it is NP-hard as well as APX-hard.
- We propose a greedy algorithm and faster sampling algorithms for group centrality maximization.
- We show the effectiveness of our algorithms on several datasets and also prove their theoretical guarantees for a constrained version of the problem.

2 Problem Definition

We assume $G(V, E)$ to be an undirected[1] graph with sets of vertices V and edges E. A shortest path between vertices s and t is a path with minimum distance (in hops) among all paths between s and t, with length $d(s, t)$. By convention, $d(s, s) = 0$, for all $s \in V$. Let P_{st} denote the set of vertices in the shortest paths (multiple ones might exist) between s and t where $s, t \notin P_{st}$. We want to maximize the centrality of the group of nodes X. We define Z as the set of candidate pairs of vertices, $Z \subseteq V \setminus X \times V \setminus X$, which we want to cover. The *coverage centrality* of a vertex is defined as:

$$C(v) = |\{(s, t) \in Z | v \in P_{st}, s \neq v, t \neq v\}| \tag{2.1}$$

$C(v)$ gives the number of pairs of vertices with at least one shortest path going through (i.e. covered by) v. The *coverage centrality* of a set $X \subseteq V$ is defined as:

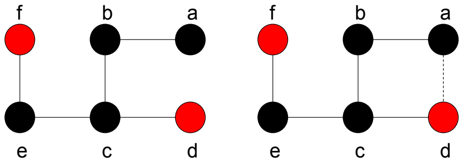
$$C(X) = |\{(s, t) \in Z | v \in P_{st}, v \in X \wedge s, t \notin X\}| \tag{2.2}$$

A set X covers a pair (s, t) iff $X \cap P_{st} \neq \varnothing$, i.e., at least one vertex in X is part of a shortest path from s to t. Our goal is to maximize the coverage centrality of X over a set of pairs Z by adding edges from a set of candidate edges Γ to G. Let G_m denote the modified graph after adding edges $E_s \subseteq \Gamma$, $G_m = (V, E \cup E_s)$. We define the coverage centrality of X (over pairs in Z) in the modified graph G_m as $C_m(X)$.

Problem 1. ***Coverage Centrality Optimization (CCO):*** *Given a network $G = (V, E)$, a set of vertices $X \subset V$, a candidate set of edges Γ, a set of vertex pairs Z and a budget k, find a set of edges $E_s \subseteq \Gamma$, such that $|E_s| = k$ and $C_m(X)$ is maximized.*

Symbols	Definitions and Descriptions
$d(s,t)$	Shortest path (s.p.) distance between s and t
n	Number of nodes in the graph
m	Number of edges in the graph
$G(V, E)$	Given graph (vertex set V and edge set E)
X	Target set of nodes
$C(v), C(X)$	Coverage centrality of node v, node set X
Γ	Candidate set of edges
k	budget
P_{st}	The set of nodes on the s.p.s between s and t
G_m, C_m	Modified graph and modified centrality
Z	Pairs of vertices to be covered
m_u	Number of uncovered pairs, $\|Z\|$

Table 1: Frequently used symbols



(a) Initial graph (b) Modified graph

Figure 1: Example of Coverage Centrality Optimization problem. We want to optimize the centrality of $\{d, f\}$ with a budget of one edge from the candidates $\{(d, a), (d, b), (f, b)\}$. The coverage centrality of $\{d, f\}$ is 0 in the initial graph (a) and 3 in the modified graph (b). Node d belongs to the shortest paths between (a, e), (a, c) and (a, f) in (b).

For simplicity, in the rest of the paper, we assume $Z = V \setminus X \times V \setminus X$ unless stated otherwise. Thus,

$$C(X) = |\{(s, t) \in V \setminus X \times V \setminus X | v \in P_{st}, v \in X, s < t\}| \tag{2.3}$$

where $s < t$ implies ordered pairs of vertices. Fig. 1 shows a solution for the CCO problem with budget $k = 1$ for an example network where the target set $X = \{d, f\}$ and the candidate set $\Gamma = \{(d, a), (d, b), (f, b)\}$.

Similarly, we can also formulate the group betweenness centrality optimization problem. Given a vertex set $X \subseteq V$, its *group betweenness centrality* is defined as:

$$B(X) = \sum_{s,t \in V \setminus X} \frac{\sigma_{s,t}(X)}{\sigma_{s,t}} \tag{2.4}$$

where $\sigma_{s,t}$ is the number of shortest paths between s and t, $\sigma_{s,t}(X)$ is the number of shortest paths between s and t passing through X. We define the group betweenness centrality of X in the modified graph G_m as $B_m(X)$.

Problem 2. ***Betweenness Centrality Optimization (BCO):*** *Given a network $G = (V, E)$, a node set $X \subset V$, a candidate edge set Γ, a set of node pairs Z and a budget k, find a set of edges $E_s \subseteq \Gamma$, such that $|E_s| \leq k$ and $B_m(X)$ is maximized.*

While we focus on the CCO problem, the results described here can be easily mapped to BCO.

[1] We discuss how our methods can be generalized to directed networks in [1].

3 Hardness and Inapproximability

This section provides complexity analysis of the CCO problem. We show that CCO is NP-hard as well as APX-hard. More specifically, CCO cannot be approximated within a factor grater than $(1 - \frac{1}{e})$.

Theorem 1. *The CCO problem is NP-hard.*

The proof is in [1]. While computing an optimal solution for CCO is infeasible in practice, a natural question is whether it has a polynomial-time approximation. The next theorem shows that CCO is also NP-hard to approximate within a factor greater than $(1 - \frac{1}{e})$. Interestingly, different from its search counterpart [25], CCO is not submodular (see [1]). These two results provide strong evidence that, for group centrality, network design is strictly harder than search.

Theorem 2. *CCO cannot be approximated within a factor greater than* $(1 - \frac{1}{e})$.

Proof. We give an L-reduction [24] from the maximum coverage (MSC) problem with parameters x and y. Given a collection of subsets $S_1, S_2, ..., S_m$ for a universal set of items $U = \{u_1, u_2, ..., u_n\}$, the MSC problem is to choose at most k sets to cover as many elements as possible. Our reduction is such that following two equations are satisfied:

$$OPT(I_{CCO}) \leq xOPT(I_{MSC})$$
$$OPT(I_{MSC}) - s(T^M) \leq y(OPT(I_{CCO}) - s(T^C))$$

where I_{MSC} and I_{CCO} are problem instances, and $OPT(Y)$ is the optimal value for instance Y. $s(T^M)$ and $s(T^C)$ denote any solution of the MSC and CCO instances, respectively. If the conditions hold and CCO has an α approximation, then MSC has an $(1 - xy(1-\alpha))$ approximation. However, MSC is NP-hard to approximate within a factor greater than $(1 - \frac{1}{e})$. It follows that $(1 - xy(1-\alpha)) < (1 - \frac{1}{e})$, or, $\alpha < (1 - \frac{1}{xye})$. So, if the conditions are satisfied, CCO is NP-hard to approximate within a factor greater than $(1 - \frac{1}{xye})$.

We use the same construction as in Theorem 1. For CCO, the set Z contains pairs in the form (b, u), $u \in U$. Let the solution of I_{CCO} be $s(T^C)$. The centrality of node a will increase by $s(T^C)$ to cover the pairs in Z. Note that $s(T^C) = 2s(T^M)$ from the construction (as the graph is undirected, the covered pair is unordered). It follows that both the conditions are satisfied when $x = 2$ and $y = \frac{1}{2}$. So, CCO is NP-hard to approximate within a factor grater than $(1 - \frac{1}{e})$.

Theorem 2 shows that there is no polynomial-time approximation better than $(1 - \frac{1}{e})$ for CCO. Given such an inapproximation result, we propose an efficient greedy heuristic for our problem in the next section.

Algorithm 1 Greedy Edge Set (GES)

```
Require: Network $G = (V, E)$, target node set $X$,
    Candidate set of edges $\Gamma$, Budget $k$
Ensure: A subset $E_s$ from $\Gamma$ of $k$ edges
 1: $E_s \leftarrow \emptyset$
 2: Compute all-pairs shortest path distances
 3: while $|E_s| \leq k$ do
 4:    for $e \in \Gamma \setminus E_s$ do
 5:       $Count(e) \leftarrow$# new covered pairs after adding $e$
 6:    end for
 7:    $e^* \leftarrow \arg\max_{e \in \Gamma \setminus E_s} \{Count(e)\}$
 8:    $E_s \leftarrow E_s \cup e^*$ and $E \leftarrow E \cup e^*$
 9:    Update the shortest path distances
10: end while
11: return $E_s$
```

4 Algorithms

4.1 Greedy Algorithm (GES) Algorithm 1 (GES) selects the best edge to be added in each of k iterations, where k is the budget. Its most important steps are 2 and 7. In step 2, it computes all-pairs shortest paths in time $O(n(m+n))$. Next, it chooses, among the candidate edges Γ, the one that maximizes the marginal coverage centrality gain of X (step 7), which takes $O(|\Gamma|n^2)$ time. After adding the best edge, shortest path distances are updated. Then, the algorithm checks the pairwise distances in $O(n^2)$ time (step 9). The total running time of GES is $O(n(m+n) + k|\Gamma|n^2)$.

We illustrate the execution of GES on the graph from Figure 1a for a budget $k = 2$, a candidate set of edges $\Gamma = \{(d, a), (d, b), (f, b)\}$, and a target set $X = \{d, f\}$. Initially, adding $(d, a), (d, b)$ and (f, b) increases the centrality of X by 3, 0, and 2, respectively, and thus (d, a) is chosen. In the second iteration, (d, b) and (f, b) increase the centrality of X by 0 and 1, respectively, and (f, b) is chosen.

4.2 Sampling Algorithm (BUS) The execution time of GES increases with $|\Gamma|$ and m. In particular, if $m = O(n^2)$ and $|\Gamma| = O(n)$, the complexity reaches $O(n^3)$, which is prohibitive for large graphs. To address this challenge, we propose a sampling algorithm that is nearly optimal, regarding each greedy edge choice, with probabilistic guarantees (see Section 5.3). Instead of selecting edges based on all the uncovered pairs of vertices, our scheme does it based on a small number of sampled uncovered pairs. This strategy allows the selection of edges with probabilistic guarantees using a small number of samples, thus ensuring scalability to large graphs. We show that the error in estimating the improvement in coverage based on the samples is small.

Algorithm 2 Best Edge via Uniform Sampling (BUS)

```
Require: Network $G = (V, E)$, target node set $X$,
    Candidate set of edges $\Gamma$, Budget $k$
Ensure: A subset $\gamma$ from $\Gamma$ of $k$ edges
 1: Choose $q$ pairs of vertices in $Q$ from $M_u$
 2: $E_s \leftarrow \emptyset$
 3: while $|E_s| \leq k$ do
 4:     for $(s, t) \in Q$ do
 5:         Compute and store shortest path distances
            $d(s, v)$ and $d(t, v)$ for all $v \in V$
 6:     end for
 7:     for $e \in \Gamma \setminus E_s$ do
 8:         $Count(e) \leftarrow$ # new covered pairs in $Q$ after
            adding $e$
 9:     end for
10:     $e^* \leftarrow \arg\max_{e \in \Gamma \setminus E_s} \{Count(e)\}$
11:     $E_s \leftarrow E_s \cup e^*$ and $E \leftarrow E \cup e^*$
12: end while
13: Return $E_s$
```

Algorithm 2 (Best Edge via Uniform Sampling, or BUS) is a sampling scheme to select the best edge to be added in each of the k iterations based on sampled uncovered node pairs. For each pair of samples, we compute the distances from each node in the pair to all others. These distances are used to estimate the true number of covered pairs after the addition of an edge. In Section 5.3, we provide a theoretical analysis of the approximation achieved by BUS.

In terms of time complexity, steps 4-6, where BUS performs shortest-path computations, take $O(q(n+m))$ time. Next, the algorithm estimates the additional number of shortest pairs covered by X after adding each of the edges based on the samples (steps 7-9) in $O(|\Gamma|q^2)$ time. Given such an estimate, the algorithm chooses the best edge to be added (step 10). The total running time of BUS is $O(kq(m + n) + k|\Gamma|q^2)$.

5 Analysis

In the previous section, we described a greedy heuristic and an efficient sampling algorithm to approximate the greedy approach. Next, we show that, under some realistic assumptions, the described greedy algorithm provides a constant-factor approximation for a modified version of CCO. More specifically, our approximation guarantees are based on the addition of two extra constraints to the general CCO described in Section 2.

5.1 Constrained Problem The extra constraints, S^1 and S^2, considered are the following: (1) S^1: We assume that edges are added from the target set X to the remaining nodes, i.e. edges in a given candidate set Γ have the form (a, b) where $a \in X$ and $b \in V \setminus X$ [4, 5]; and (2) S^2: Each pair (s, t) can be covered by at most one single newly added edge [3, 16].

S^1 is a reasonable assumption in many applications. For instance, in online advertising, adding links to a third-party page gives away control over the navigation, which is undesirable. S^2 is motivated by the fact that, in real-life graphs, centrality follows a skewed distribution (e.g. power-law), and thus most of the new pairs will have shortest paths through a single edge in Γ. Generalizing our methods to the case where shortest paths are covered by any fixed number of edges in Γ is straightforward. In our experiments (see Section 6.1), we show that solutions for the constrained and general problem are often close. Moreover, both constraints have been considered by previous work [3, 16]. Next, we show that COO under S^1 and S^2, or RCCO (Restricted CCO), for short, is still NP-hard.

Corollary 3. *RCCO is NP-hard.*

Proof. Follows directly from Theorem 1, as our construction respects both the constraints.

5.2 Analysis: Greedy Algorithm The next theorem shows that RCCO's optimization function is monotone and submodular. As a consequence, the greedy algorithm described in Section 4.1 leads to a well-known constant factor approximation of $(1 - 1/e)$ [17].

Theorem 4. *The objective function $f(E_s) = C_m(X)$ in RCCO is monotone and submodular.*

Proof. Monotonicity: Follows from the definition of a shortest path. Adding an edge $(u, v) \in E_s$ cannot increase $d(s, t)$ for any (s, t) already covered by X. Since $u \in X$ for any $(u, v) \in E_s$, the coverage $C_m(X)$ is also non-decreasing.

Submodularity: We consider addition of two sets of edges, E_a and E_b where $E_a \subset E_b$, and show that $f(E_a \cup \{e\}) - f(E_a) \geq f(E_b \cup \{e\}) - f(E_b)$ for any edge $e \in \Gamma$ such that $e \notin E_a$ and $e \notin E_b$. Let $F(A)$ be the set of node pairs (s, t) which are covered by an edge $e \in A$ ($|F(E_s)| = C_m(X)$). Then $f(.)$ is submodular if $F(E_b \cup \{e\}) \setminus F(E_b) \subseteq F(E_a \cup \{e\}) \setminus F(E_a)$. To prove this claim, we make use of S^B. Therefore, each pair $(s, t) \in F(E_b)$ is covered by only one edge in E_b. As $E_a \subset E_b$, adding e to E_a will cover some of the pairs which are already covered by $E_b \setminus E_a$. Then, for any newly covered pair $(s, t) \in F(E_b \cup \{e\}) \setminus F(E_b)$, it must hold that $(s, t) \in F(E_a \cup \{e\}) \setminus F(E_a)$.

Based on Theorem 4, if OPT is the optimal solution for an instance of the RCCO problem, GES will return a set of edges E_s such that $f(E_s) \geq (1 - 1/e)OPT$. The existence of such an approximation algorithm shows that the constraints S^1 and S^2 make the CCO problem easier, compared to its general version. On the other hand, whether GES is a good algorithm for the modified CCO (RCCO) remains an open question. In order to show that our algorithm is optimal, in the sense that the best algorithm for this problem cannot achieve a better approximation from those of GES, we also prove an inapproximability result for the constrained problem.

COROLLARY 5. *RCCO cannot be approximated within a factor greater than $(1 - \frac{1}{e})$.*

Proof. Follows directly from Thm. 2, as the construction applied in the proof respects both the constraints.

Corollary 5 certifies that GES achieves the best approximation possible for constrained CCO (RCCO).

5.3 Analysis: Sampling Algorithm In Section 4.2, we presented BUS, a fast sampling algorithm for the general CCO problem. Here, we study the quality of the approximation provided by BUS as a function of the number of sampled node pairs. The analysis will assume the constrained version of CCO (RCCO), but the general case will also be discussed.

Let us assume that X covers a set M_c of pairs of nodes. The set of remaining pairs is $M_u = \{(s,t)|s \in V, t \in V, s \neq t, X \cap P_{st} = \emptyset\}$ and $m_u = |M_u| = n(n-1) - |M_c|$. We sample, uniformly with replacement, a set of ordered pairs Q ($|Q| = q$) from M_u. Let $g^q(.)$ denote the number of *new* pairs covered by the candidate edges based on the samples Q. For an edge set $\gamma \subset \Gamma$, X_i is a random variable that denotes whether the ith sampled pair is covered by any edge in γ. In other words, $X_i = 1$ if the pair is covered and 0, otherwise. Each pair is chosen with probability $\frac{1}{m_u}$.

LEMMA 5.1. *Given q sampled node pairs from M_u:*

$$E(g^q(\gamma)) = \frac{q}{m_u} f(\gamma)$$

From the samples, we get $g^q(\gamma) = \Sigma_{i=1}^{q} X_i$. By the linearity and additive rule, $E(g^q(\gamma)) = \Sigma_{i=1}^{q} E(X_i) = q.E(X_i)$. As the probability $P(X_i) = \frac{f(\gamma)}{m_u}$ and X_is are i.i.d., $E(g^q(\gamma)) = \frac{q}{m_u} f(\gamma)$. Also, let us define $f^q = \frac{m_u}{q} g^q$ as the estimated coverage.

LEMMA 5.2. *Given ϵ $(0 < \epsilon < 1)$, a positive integer l, a budget k, and a sample of independent uncovered node pairs $Q, |Q| = q$, where $q(\epsilon) \geq \frac{3 m_u (l+k) log(|\Gamma|)}{\epsilon^2 \cdot OPT}$; then:*

$$Pr(|f^q(\gamma) - f(\gamma)| < \epsilon \cdot OPT) \geq 1 - 2|\Gamma|^{-l}$$

For all $\gamma \subset \Gamma$, $|\gamma| \leq k$, where OPT denotes the optimal coverage ($OPT = Max\{f(\gamma)|\gamma \subset \Gamma, |\gamma| \leq k\}$).

Proof. Using Lemma 5.1:

$$Pr(|f^q(\gamma) - f(\gamma)| \geq \delta \cdot f(\gamma))$$

$$Pr\Big(|\frac{q}{m_u} f^q(\gamma) - \frac{q}{m_u} f(\gamma)| \geq \frac{q}{m_u} \cdot \delta \cdot f(\gamma)\Big)$$

$$Pr\Big(|g^q(\gamma) - \frac{q}{m_u} f(\gamma)| \geq \frac{q}{m_u} \cdot \delta f(\gamma)\Big)$$

$$Pr(|g^q(\gamma) - E(g^q(\gamma))| \geq \delta E(g^q(\gamma)))$$

As samples are independent, the Chernoff bound gives:

$$Pr\Big(|g^q(\gamma) - \frac{q}{m_u} f(\gamma)| \geq \frac{q}{m_u} \delta f(\gamma)\Big) \leq 2 \exp\Big(-\frac{\delta^2}{3}\frac{q}{m_u} f(\gamma)\Big)$$

Substituting $\delta = \frac{\epsilon OPT}{f(\gamma)}$ and q:

$$Pr(|f^q(\gamma) - f(\gamma)| \geq \epsilon \cdot OPT) \leq 2 \exp\Big(-\frac{OPT}{f(\gamma)}(l+k) log(\Gamma)\Big)$$

Using the fact that $OPT \geq f(\gamma)$:

$$Pr(|f^q(\gamma) - f(\gamma)| \geq \epsilon \cdot OPT) \leq 2|\Gamma|^{-(l+k)}$$

Applying the union bound over all possible size-k subsets of $\gamma \subset \Gamma$ (there are $|\Gamma|^k$) we conclude that:

$$Pr(|f^q(\gamma) - f(\gamma)| \geq \epsilon \cdot OPT) < 2|\Gamma|^{-l}, \forall \gamma \subset \Gamma$$

$$Pr(|f^q(\gamma) - f(\gamma)| < \epsilon \cdot OPT) \geq 1 - 2|\Gamma|^{-l}, \forall \gamma \subset \Gamma$$

Now, we prove our main theorem which shows an approximation bound of $(1 - \frac{1}{e} - \epsilon)$ by Algorithm 2 whenever the number of samples is at least $q(\epsilon/2) = \frac{12 m_u (l+k) log(|\Gamma|)}{\epsilon^2 \cdot OPT}$ (l and ϵ are as in Lemma 5.2).

THEOREM 6. *Algorithm 2 ensures $f(\gamma) \geq (1 - \frac{1}{e} - \epsilon)OPT$ with high probability $(1 - \frac{2}{|\Gamma|^l})$ if at least $q(\epsilon/2)$ samples are considered.*

Proof. $f(.)$ is monotonic and submodular (Thm. 4) and one can prove the same for $f^q(.)$. Given the following:

1. From Lemma 5.2, the number of samples is at least $q(\epsilon/2)$. So, with probability $1 - \frac{2}{|\Gamma|^l}$, $f(\gamma) \geq f^q(\gamma) - \frac{\epsilon}{2} OPT$;
2. $f^q(\gamma) \geq (1 - \frac{1}{e}) f^q(\gamma*)$, $\gamma* = \arg\max_{\gamma' \subset \Gamma, |\gamma'| \leq k} f^q(\gamma')$ (submodularity property of $f^q(.)$);
3. $f^q(\gamma*) \geq f^q(\bar{\gamma})$, $\bar{\gamma} = \arg\max_{\gamma' \subset \Gamma, |\gamma'| \leq k} f(\gamma')$ (Note that, $OPT = f(\bar{\gamma})$)

We can prove with probability $1 - \frac{2}{|\Gamma|^l}$ that:

$$\begin{aligned} f(\gamma) &\geq f^q(\gamma) - \frac{\epsilon}{2} OPT \\ &\geq \Big(1 - \frac{1}{e}\Big) f^q(\gamma*) - \frac{\epsilon}{2} OPT \\ &\geq \Big(1 - \frac{1}{e}\Big) f^q(\bar{\gamma}) - \frac{\epsilon}{2} OPT \\ &\geq \Big(1 - \frac{1}{e}\Big)\Big(f(\bar{\gamma}) - \frac{\epsilon}{2} OPT\Big) - \frac{\epsilon}{2} OPT \\ &> \Big(1 - \frac{1}{e} - \epsilon\Big) OPT \end{aligned}$$

Thm.	#Samples	Approximations
Thm. 6	$O(\frac{m_u klog(\|\Gamma\|)}{\epsilon^2 . OPT})$	$f(\gamma) > (1 - \frac{1}{e} - \epsilon)OPT$
Cor. 8	$O(\frac{klog(\|\Gamma\|)}{\epsilon^2})$	$f(\gamma) > (1 - \frac{1}{e})OPT - \epsilon.m_u$

Table 2: Summary of the probabilistic approximations.

While we are able to achieve a good probabilistic approximation with respect to the optimal value OPT, deciding the number of samples is not straightforward. In practice, we do not know the value of OPT beforehand, which affects the number of samples needed. However, notice that OPT is bounded by the number of uncovered pairs m_u. Moreover, the number of samples $q(\epsilon/2)$ depends on the ratio $\frac{m_u}{OPT}$. Increasing this ratio while keeping the quality constant requires more samples. If OPT (which depends on X) is close to the number of uncovered pairs m_u, we need fewer samples to achieve the bound. In the experiments, we assume this ratio to be constant. Next, we propose another approximation scheme where we can reduce the number of samples by avoiding the term OPT in the sample size while waiving the assumption involving constants.

Let M_u and m_u be the set and number of uncovered pairs by X, respectively, in the initial graph. Moreover, we define $\bar{q}(\epsilon)$ so that:

$$\bar{q}(\epsilon) \geq \frac{3(l+k)log(|\Gamma|)}{\epsilon^2}$$

Corollary 7. *Given ϵ $(0 < \epsilon < 1)$, a positive integer l, a budget k, and a sample of independent uncovered node pairs $Q, |Q| = \bar{q}(\epsilon)$, then:*

$$Pr(|f^q(\gamma) - f(\gamma)| < \epsilon \cdot m_u) \geq 1 - 2|\Gamma|^{-l}, \forall \gamma \subset \Gamma, |\gamma| \leq k$$

The proof is given in [1]. Next, we provide an approximation bound by our sampling scheme for at least $\bar{q}(\epsilon/2) = \frac{12(l+k)log(|\Gamma|)}{\epsilon^2}$ samples.

Corollary 8. *Algorithm 2 ensures $f(\gamma) \geq (1 - \frac{1}{e})OPT - \epsilon.m_u$ with high probability $(1 - \frac{2}{|\Gamma|^l})$ if at least $\bar{q}(\epsilon/2)$ samples are used.*

This proof is also in [1]. Table 2 summarizes the number of samples and corresponding bounds for Algorithm 2. Theorem 6 ensures higher quality with higher number of samples than Corollary 8. On the other hand, Corollary 8 does not assume anything about the ratio $\frac{m_u}{OPT}$. The results reflect a trade-off between number of samples and accuracy.

Theorem 6 and Corollary 8 assume that a greedy approach achieves a constant-factor approximation of $(1 - 1/e)$, which holds only for the RCCO problem (see Sections 5.1 and 5.2). As a consequence, in the case of the general problem, the guarantees discussed in this Section apply only for each iteration of our sampling algorithm, but not for the final results. In other words, BUS provides theoretical quality guarantees that each edge selected in an iteration of the algorithm achieves a coverage within bounded distance from the optimal edge. Nonetheless, experimental results show, in practice, BUS is also effective in the general setting.

Dataset Name	$\|V\|$	$\|E\|$
Network Science Coauthorship (NS)	0.3k	1k
email-Eu-core (EU)	1k	25k
ca-GrQc (CG)	5K	14K
email-Enron (EE)	36K	183K
loc-Brightkite (LB)	58K	214K
loc-Gowalla (LG)	196K	950K
web-Stanford (WS)	280K	2.3M
DBLP (DB)	1.1M	5M

Table 3: Dataset description and statistics.

	Ratio		
Data	$k = 5$	$k = 10$	$k = 15$
NS	1.02	1.14	1.17
EU	1.0	1.1	1.08
Synthetic	1.0	1.0	1.0

Table 4: The ratio between the improvement in coverage produced by GES for CCO and RCCO.

6 Experimental Results

Experimental Setup: We evaluate our algorithms on real-world networks. All experiments were conducted on a 3.30GHz Intel Core i7 machine with 30 GB RAM and Ubuntu. Algorithms were implemented in Java.

Dataset: All datasets applied are available online[2]. Table 3 shows dataset statistics. The graphs are undirected and we consider the largest connected component for our experiments. The datasets are from different categories: EE and EU are constructed from email communication; NS, CG and DB are collaboration networks; LB and LG are OSNs and WS is a webgraph.

Other Settings: We set the candidate of edges Γ as those edges from X to the remaining vertices that are absent in the initial graph (i.e. $\Gamma = \{(u, v)|u \in X \wedge v \in V \setminus X \wedge (u, v) \notin E\}$). The set of target nodes X is randomly selected from the set of all nodes. Results reported are averages of 10 repetitions.

Baselines: We consider three baselines in our experiments: 1) **High-ACC [25, 14]:** Finds the top k central nodes based on *maximum adaptive centrality coverage* and adds edges between target nodes X and the set of top-k central nodes; 2) **High-Degree:** Selects edges between the target nodes X and the top k high degree

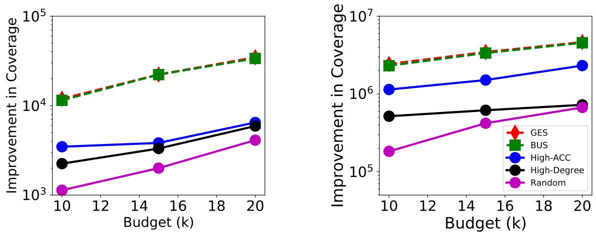
[2] Datasets are from (1) `https://snap.stanford.edu`, (2) `http://dblp.uni-trier.de`, (3) `http://www-personal.umich.edu/~mejn/netdata/` and the code is available at `http://cs.ucsb.edu/~medya/CODE/SDM18/`.

	Coverage of BUS (relative to baselines)				**Time [sec.]**			**# Samples**
Budget	GES	High-ACC	High-Degree	Random	GES	High-ACC	BUS	BUS
$k = 10$	0.95	2.46	5.41	14.45	> 7200	157.1	5.1	2560
$k = 15$	0.97	2.92	7.29	9.98	> 7200	156.9	10.1	3840
$k = 20$	0.98	2.78	9.96	9.59	> 7200	157.2	18.2	5120

Table 5: **CG data:** Comparison between our sampling algorithm (BUS) and the baselines, including our Greedy (GES) approach, using the CG dataset and varying the budget k. We evaluate the coverage of BUS relative to the baselines—i.e. how many times more new pairs are covered by BUS compared to the baseline.

	Coverage of BUS (relative to baselines)				**Time [sec.]**			**# Samples**
Budget	GES	High-ACC	High-Degree	Random	GES	High-ACC	BUS	BUS
$k = 10$	0.96	3.3	5.1	10.1	271	2.3	1.8	2093
$k = 15$	0.97	5.8	6.7	11.1	423	2.4	3.4	3139
$k = 20$	0.97	5.2	5.7	8.2	531	2.5	4.5	4186

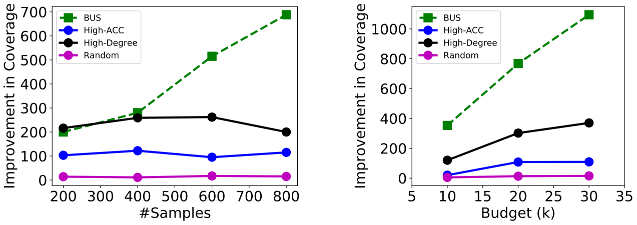
Table 6: **EU data:** Comparison between our sampling algorithm (BUS) and the baselines using the EU dataset.



(a) Quality on EU

(b) Quality on CG

Figure 2: BUS vs. Greedy: Improvement in coverage centrality produced by different algorithms.



(a) Fixed Budget

(b) Fixed #Sample

Figure 3: Comparison with baselines on the EE dataset varying (b) the number of samples and (c) the budget.

nodes; 3) **Random:** Randomly chooses k edges from Γ which are not present in the graph. We also compare our sampling algorithm (BUS) against our Greedy solution (GES) and show that BUS is more efficient while producing similar results in terms of quality.

Performance Metric: The quality of a solution set (a set of edges produced by the algorithm) is the number of newly covered pairs by the target set of nodes after addition of these edges in the intial graph. We call it *improvement in coverage.*

6.1 GES: RCCO vs CCO We compare coverage centrality optimization (CCO) and its restricted version (RCCO) by applying GES to two small real (NS and EU) and one synthetic (Barabasi) network ($|V| = 2k$, $|E| = 10k$). The target set size $|X|$ is set to 5. Table 4 shows the ratio between results for CCO and RCCO varying the budget k. The results, close to 1, support the RCCO assumptions discussed in Section 5.1.

6.2 BUS vs. GES We apply only the smallest dataset (CG) in this experiment, as the GES algorithm is not scalable—it requires the computation of all-pairs shortest paths. For BUS, we set the error $\epsilon = 0.3$. First, we evaluate the effect of sampling on quality, which we theoretically analyzed in Theorem 6 and Corollary 8.

Fig. 2 shows the number of new pairs covered by the algorithms. Table 5 and 6 show the running times and the quality of BUS relative to the baselines—i.e. how many times more pairs are covered by BUS compared to a given baseline on CG and EU data, respectively. BUS and GES produce results at least 2 times better than the baselines. Moreover, BUS achieves results comparable to GES while being 2-3 orders of magnitude faster.

6.3 Results for Large Graphs: We compare our sampling algorithm against the baseline methods using large graphs (EE, LB, LG, WS and DB). Due to the high cost of computing all-pairs shortest paths, we estimate the centrality based on $10K$ randomly selected pairs. For High-ACC, we also use sampling for adaptive coverage centrality computation [25, 14] and the same number of samples is used. The budget and target set sizes are set as 20 and 5, respectively.

Table 7 shows the results, where the quality is relative to BUS results. BUS takes a few minutes ($8, 15, 17, 45, 85$ minutes for EE, LB, WS, LG and DB respectively) to run and significantly outperforms the baselines. This happens as the existing approaches do not take into account the dependencies between the edges selected. BUS selects the edges sequentially, considering the effect of edges selected in previous steps.

6.4 Parameter Sensitivity:

The main parameters of BUS are the budget and the number of samples—both affect the error ϵ, as discussed in Thm. 6 and Cor. 8. We study the impact of these two parameters on performance. Again, we estimate coverage using $10K$ randomly selected pairs of nodes.

Fig. 3a shows the results on EE data for budget 20 and target set size 5. With 600 samples, BUS produces results at least 2 times better than the baselines. Next, we fix the number of samples and vary the budget. Figure 3b shows the results with $10K$ samples and 5 target nodes. BUS produces results at least 2.5 times better than the baselines. Moreover, BUS takes only 30 seconds to run with budget of 30 and 1000 samples. We find that the running time grows linearly with the budget for a fixed number of samples. These results validate the running time analysis from Sec. 4.2.

	BUS (relative to baselines)			**# Samples**
Data	High-Acc	High-Degree	Random	BUS
EE	4.88	2.74	51	6462
LB	3.3	2.3	33.8	6796
LG	3.3	4.2	62	4255
WS	1.89	1.95	4.8	2000
DB	2.5	1.6	5	875

Table 7: Coverage centrality of BUS relative to baselines for large datasets.

6.5 Impact on other Metrics:

While this paper is focused on optimizing Coverage Centrality, it is interesting to analyze how our methods affect other relevant metrics. Here, we look at the following ones: 1) influence, 2) average shortest-path distance, and 3) closeness centrality. The idea is to assess how BUS improves the influence of the target nodes, decreases the distances from the target to the remaining nodes, and increases the closeness centrality of these nodes as new edges are added to the graph. For influence analysis, we consider the popular independent cascade model [10] with edge probabilities as 0.1. In all the experiments, we fix the number of sampled pairs at 1000 and choose 10 nodes, uniformly at random, as the target set X. The metrics are computed before and after the addition of edges and presented as the relative improvement (in percentage). Because target nodes are chosen at random, increasing the budget does not necessarily lead to an increase in the metrics considered.

Results are presented in Table 8. There is a significant improvement of the three metrics as the budget (k) increases. For influence, the number of seed nodes is small, and thus the relative improvement for increasing k is large. The improvement of the other metrics is also significant. For instance, in EE, the decrease in distance is nearly 5%, which is approximately $72K$, for a budget of 75.

	Influence			Distance			Closeness		
k	EE	LB	LG	EE	LB	LG	EE	LB	LG
25	57.7	12.2	10.7	2.7	1.2	2.2	2.0	2.0	1.0
50	96.8	17.5	92.7	3.8	3.5	3.3	4.9	3.9	4.0
75	134.3	29.1	45.9	5.2	2.1	2.3	5.9	2.3	1.9

Table 8: Improvement of other metrics after adding the edges found by BUS: the numbers are improvement in percentage with respect to the value for the initial graph.

7 Previous Work

General network design problems: A set of design problems were introduced by Paik et al. [18]. They focused on vertex upgrades to improve the delays on adjacent edges. Krumke et al. [12] generalized this model and proposed minimizing the cost of the minimum spanning tree. Lin et al. [13] also proposed a shortest path optimization problem via improving edge weights under a budget constraint. In [7, 15], the authors studied the path optimization problem under node improvement.

Design problems via edge addition: Meyerson et al. [16] proposed approximation algorithms for single-source and all-pairs shortest paths minimization. Faster algorithms for the same problems were presented in [19]. Demaine et al. [6] minimized the diameter of a network and the node eccentricity by adding shortcut edges with a constant factor approximation algorithm. Past research had also considered eccentricity minimization in a composite network [21]. However, all aforementioned problems are based on improving distances and hence are complementary to our objective.

Centrality computation and related optimization problems: The first efficient algorithm for betweenness centrality computation was proposed by Brandes [2]. Recently, [22] introduced an approach for computing the top-k nodes in terms of betweenness centrality via VC-dimension theory. Yoshida [25] studied similar problems —for both betweenness and coverage centrality— in the adaptive setting, where shortest paths already covered by selected nodes are not taken into account. Yoshida's algorithm was later improved using a different sampling scheme [14]. Here, we focus on the design version of the problem, where the goal is to optimize the coverage centrality of a target set of nodes by adding edges. Previous work has studied a constrained version of our problem where the target set size is one [4, 9, 5]. Note that, as the target set X can be chosen arbitrarily in our problem, our solutions and theoretical analysis differ significantly from theirs. In [20], the authors also assume a single target node while maximizing the expected decrease in shortest path distances to the remaining nodes via edge addition. Our work is the first to address the more general and challenging problem of maximizing the centrality of a group of nodes via budgeted edge additions.

8 Conclusions

We studied several variations of a novel network design problem, the group centrality optimization. This problem has applications in a variety of domains including social, collaboration, and communication networks. From a theoretical perspective, we have shown that these variations of the problem are NP-hard as well as APX-hard. Moreover, we have proposed a greedy algorithm, and even faster sampling algorithms, for group centrality optimization. Our algorithms provide theoretical quality guarantees under realistic assumptions and also outperform the baseline methods by up to 5 times in several datasets. From a broader point of view, we believe that this paper highlights interesting properties of network design problems compared to their, more well-studied, search counterparts.

Acknowledgment

Research was sponsored by the Army Research Laboratory and accomplished under Cooperative Agreement Number W911NF-09-2-0053 (the ARL Network Science CTA). The views and conclusions in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] https://arxiv.org/abs/1702.04082.

[2] U. Brandes. A faster algorithm for betweenness centrality. *Journal of mathematical sociology*, pages 163–177, 2001.

[3] V. Chaoji, S. Ranu, R. Rastogi, and R. Bhatt. Recommendations to boost content spread in social networks. In *WWW*, pages 529–538, 2012.

[4] P. Crescenzi, G. D'Angelo, L. Severini, and Y. Velaj. Greedily improving our own centrality in a network. In *SEA*, pages 43–55, 2015.

[5] G. D'Angelo, L. Severini, and Y. Velaj. On the maximum betweenness improvement problem. *Electronic Notes in TCS*, 322:153 – 168, 2016.

[6] E. D. Demaine and M. Zadimoghaddam. Minimizing the diameter of a network using shortcut edges. *Lecture Notes in Computer Science*, pages 420–431, 2010.

[7] B. Dilkina, K. J. Lai, and C. P. Gomes. Upgrading shortest paths in networks. In *CPAIOR*, pages 76–91, 2011.

[8] A. Gupta and J. Könemann. Approximation algorithms for network design: A survey. *Surveys in Operations Research and Management Science*, pages 3–20, 2011.

[9] V. Ishakian, D. Erdos, E. Terzi, and A. Bestavros. A framework for the evaluation and management of network centrality. In *SDM*, pages 427–438, 2012.

[10] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *KDD*, pages 137–146, 2003.

[11] E. B. Khalil, B. Dilkina, and L. Song. Scalable diffusion-aware optimization of network topology. In *KDD*, pages 1226–1235, 2014.

[12] S. Krumke, M. Marathe, H. Noltemeier, R. Ravi, and S. Ravi. Approximation algorithms for certain network improvement problems. *Journal of Combinatorial Optimization*, 2:257–288, 1998.

[13] Y. Lin and K. Mouratidis. Best upgrade plans for single and multiple source-destination pairs. *GeoInformatica*, 19(2):365–404, 2015.

[14] A. Mahmoody, E. Charalampos, and E. Upfal. Scalable betweenness centrality maximization via sampling. In *KDD*, pages 1765–1773, 2016.

[15] S. Medya, P. Bogdanov, and A. Singh. Towards scalable network delay minimization. In *ICDM*, pages 1083–1088, 2016.

[16] A. Meyerson and B. Tagiku. Minimizing average shortest path distances via shortcut edge addition. In *APPROX-RANDOM*, pages 272–285, 2009.

[17] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. Best algorithms for approximating the maximum of a submodular set function. *Math. Oper. Res.*, pages 177–188, 1978.

[18] D. Paik and S. Sahni. Network upgrading problems. *Networks*, pages 45–58, 1995.

[19] N. Parotisidis, E. Pitoura, and P. Tsaparas. Selecting shortcuts for a smaller world. In *SDM*, pages 28–36, 2015.

[20] N. Parotsidis, E. Pitoura, and P. Tsaparas. Centrality-aware link recommendations. In *WSDM*, pages 503–512, 2016.

[21] S. Perumal, P. Basu, and Z. Guan. Minimizing eccentricity in composite networks via constrained edge additions. In *MILCOM*, pages 1894–1899, 2013.

[22] M. Riondato and E. M. Kornaropoulos. Fast approximation of betweenness centrality through sampling. In *WSDM*, pages 413–422, 2014.

[23] H. Tong, B. A. Prakash, T. Eliassi-Rad, M. Faloutsos, and C. Faloutsos. Gelling, and melting, large graphs by edge manipulation. In *CIKM*, pages 245–254, 2012.

[24] D. P. Williamson and D. B. Shmoys. *The design of approximation algorithms*. Cambridge, 2011.

[25] Y. Yoshida. Almost linear-time algorithms for adaptive betweenness centrality using hypergraph sketches. In *KDD*, pages 1416–1425, 2014.

[26] Q. K. Zhu. *Power distribution network design for VLSI*. John Wiley & Sons, 2004.