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, 20]. 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, 16, 17], 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 \( X \) [20].

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
of a vertex is defined as:

\[ C(v) = |\{(s,t) \in Z | v \in P_{st}, s \neq v, t \neq v\}| \]

The coverage centrality of a set \( X \subseteq V \) is defined as:

\[ C(X) = |\{(s,t) \in Z | v \in P_{st}, v \in X \land s,t \notin X\}| \]

A set \( X \) covers a pair \((s,t)\) iff \( X \cap P_{st} \neq \emptyset \), 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 \( \Gamma \) 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 \( X \subseteq V \), a candidate set of edges \( \Gamma \), 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.

**Problem 2. Betweenness Centrality Optimization (BCO):** Given a network \( G = (V,E) \), a node set \( X \subseteq V \), a candidate edge set \( \Gamma \), 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.

...
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 greater 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 [23], 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 $\alpha$ 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 greater 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.

### 4 Algorithms

#### 4.1 Greedy Algorithm (GES)

**Algorithm 1 (GES)**

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. $E_s \leftarrow E_s \cup e^*$
   9. Update the shortest path distances
10. end while
11. return $E_s$

The CCO problem is NP-hard. 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 $\alpha$ 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 greater 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 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 $M_a$
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 $\Gamma$ have the form $(a,b)$ where $a \in X$ and $b \in V \setminus X$ [3, 15]; and (2) $S^2$: Each pair $(s,t)$ can be covered by at most one single newly added edge [3, 15].

$S^2$ 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 $\Gamma$. Generalizing our methods to the case where shortest paths are covered by any fixed number of edges in $\Gamma$ is straightforward. In our experiments (see Section 5.3), we show that solutions for the constrained and general problem are often close. Moreover, both constraints have been considered by previous work [3, 15]. 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 4.2 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 $(u,v) \in E_a$ cannot increase $d(s,t)$ for any $(s,t)$ already covered by $X$. Since $u \in X$ for any $(u,v) \in E_a$, 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 $S^2$. 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 $E_b$. Then, for any newly covered pair $(s,t) \in F(E_a \cup \{e\}) \setminus F(E_a)$, 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 $S^1 \delta S^2$ 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 $(1 - \frac{1}{2})$.

**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 $i$th sampled pair is covered by any edge in $\gamma$. 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 $g^q(\gamma) = \sum_{i=1}^{q} X_i$. By the linearity and additive rule, $E(g^q(\gamma)) = \sum_{i=1}^{q} E(X_i) = q, E(X_i)$. The probability $P(X_i) = \frac{f^q(\gamma)}{m_u}$ and $X_i$s are i.i.d., $E(g^q(\gamma)) = \frac{q}{m_u} f(\gamma)$. Also, we let $f^q = \frac{q}{m_u} q^q$ as the estimated coverage.

**Lemma 5.2.** Given $\epsilon (0 < \epsilon < 1)$, a positive integer $l$, a budget $k$, and a sample of independent uncovered node pairs $Q, |Q| = q$, where $q(\epsilon) \geq \frac{2m_u((l+k)\log(|\Gamma|))}{\epsilon^2 \cdot OPT}$; 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 ($OPT = \text{Max}\{f(\gamma) | \gamma \subset \Gamma, |\gamma| \leq k\}$).

**Proof.** Using Lemma 5.1

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

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

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

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

As samples are independent, the Chernoff bound gives:

$$Pr(|f^q(\gamma) - f(\gamma)| \geq \epsilon \cdot OPT) \leq 2 \exp \left(-\frac{\epsilon^2}{2} \cdot q f(\gamma)\right)$$

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

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

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

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

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 $(1 - \frac{1}{2} - \epsilon)$ by Algorithm 2 whenever the number of samples is at least $q(\epsilon/2) = \frac{2m_u((l+k)\log(|\Gamma|))}{\epsilon^2 \cdot OPT}$ (and $\epsilon$ as are in Lemma 5.2).

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

**Proof.** $f(.)$ is monotonic and submodular (Thm. 1) 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 $1 - \frac{2}{|\Gamma|}$, $f(\gamma) \geq f^q(\gamma) - \frac{1}{2} OPT$.
2. $f^q(\gamma) \geq (1 - \frac{1}{2}) f^q(\gamma^*)$, $\gamma^* = \arg\max_{\gamma \subset \Gamma, |\gamma| \leq k} f^q(\gamma)$ (submodularity property of $f^q(.)$);
3. $f^q(\gamma^*) \geq f^q(\tilde{\gamma})$, $\tilde{\gamma} = \arg\max_{\gamma \subset \Gamma, |\gamma| \leq k} f^q(\gamma)$ (Note that, $OPT = f(\gamma)$)

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

$$f(\gamma) \geq f^q(\gamma) - \frac{\epsilon}{2} OPT$$

$$\geq (1 - \frac{1}{2}) f^q(\gamma^*) - \frac{\epsilon}{2} OPT$$

$$\geq (1 - \frac{1}{2}) f(\tilde{\gamma}) - \frac{\epsilon}{2} OPT$$

$$\geq (1 - \frac{1}{2} - \epsilon) OPT$$

Copyright © 2018 by SIAM
Unauthorized reproduction of this article is prohibited
Table 2: Summary of the probabilistic approximations.

<table>
<thead>
<tr>
<th>Thm.</th>
<th>#Samples</th>
<th>Approximations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thm. 6</td>
<td>$O\left(\frac{m_u \log(1/k)}{\epsilon^3} \right)$</td>
<td>$f(\gamma) &gt; (1 - \frac{1}{\epsilon} - \epsilon)OPT$</td>
</tr>
<tr>
<td>Cor. 8</td>
<td>$O\left(\frac{m_u \log(1/k)}{\epsilon^2} \right)$</td>
<td>$f(\gamma) &gt; (1 - \frac{1}{\epsilon})OPT - \epsilon \cdot m_u$</td>
</tr>
</tbody>
</table>

Table 3: Dataset description and statistics.

<table>
<thead>
<tr>
<th>Dataset Name</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Network Science Coauthorship (NS)</td>
<td>0.3k</td>
<td>1k</td>
</tr>
<tr>
<td>email-Eu-core (EU)</td>
<td>1k</td>
<td>25k</td>
</tr>
<tr>
<td>ca-GrQc (CG)</td>
<td>5K</td>
<td>14K</td>
</tr>
<tr>
<td>email-Enron (EE)</td>
<td>36K</td>
<td>183K</td>
</tr>
<tr>
<td>loc-Brightkite (LB)</td>
<td>58K</td>
<td>214K</td>
</tr>
<tr>
<td>loc-Gowalla (LG)</td>
<td>196K</td>
<td>950K</td>
</tr>
<tr>
<td>web-Stanford (WS)</td>
<td>280K</td>
<td>2.3M</td>
</tr>
<tr>
<td>DBLP (DB)</td>
<td>1.1M</td>
<td>5M</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Data</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>k = 5</td>
<td>1.00</td>
</tr>
<tr>
<td>k = 10</td>
<td>1.14</td>
</tr>
<tr>
<td>k = 15</td>
<td>1.17</td>
</tr>
<tr>
<td>Synthetic</td>
<td>1.00</td>
</tr>
</tbody>
</table>

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. Datasets: All datasets applied are available online\footnote{Databases are from (1) \url{https://snap.stanford.edu} (2) \url{http://dblp.uni-trier.de} (3) \url{http://www-personal.umich.edu/~mejn/netdata} and the code is available at \url{http://cs.ucsd.edu/~medya/CODE/SDM18/}}. 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 $\Gamma$ as those edges from $X$ to the remaining vertices that are absent in the initial graph (i.e. $\Gamma = \{(u, v) | u \in X \land v \in V \setminus X \land (u, v) \notin E\}$). 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\footnote{Algorithm 2 ensures $f(\gamma) \geq \gamma \in \Gamma$ with high probability $(1 - \frac{\epsilon}{2})$ if at least $q(\gamma) / 2$ samples are used.} \cite{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 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.
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.

<table>
<thead>
<tr>
<th>Budget</th>
<th>Coverage of BUS (relative to baselines)</th>
<th>Time [sec.]</th>
<th># Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GES</td>
<td>High-ACC</td>
<td>High-Degree</td>
</tr>
<tr>
<td>( k = 10 )</td>
<td>0.95</td>
<td>2.46</td>
<td>5.41</td>
</tr>
<tr>
<td>( k = 15 )</td>
<td>0.97</td>
<td>2.92</td>
<td>7.29</td>
</tr>
<tr>
<td>( k = 20 )</td>
<td>0.98</td>
<td>2.78</td>
<td>9.96</td>
</tr>
</tbody>
</table>

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

![Quality on EU](image1.png) ![Quality on CG](image2.png)

(a) Quality on EU (b) Quality on CG

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

![Fixed Budget](image3.png) ![Fixed #Sample](image4.png)

(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.

2) **Random:** Randomly chooses \( k \) edges from \( \Gamma \) 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 initial 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](image5.png)

Fig. 2 shows the number of new pairs covered by the algorithms. Table 4 and 5 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 10\( k \) randomly selected pairs. For High-ACC, we also use sampling for adaptive coverage centrality computation \cite{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 $\epsilon$, 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.

<table>
<thead>
<tr>
<th>k</th>
<th>Influence</th>
<th>Distance</th>
<th>Closeness</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>EE</td>
<td>LB</td>
<td>LG</td>
</tr>
<tr>
<td>51.7</td>
<td>12.2</td>
<td>10.7</td>
<td></td>
</tr>
<tr>
<td>2.7</td>
<td>1.2</td>
<td>2.2</td>
<td></td>
</tr>
<tr>
<td>2.0</td>
<td>2.0</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>EE</td>
<td>LB</td>
<td>LG</td>
</tr>
<tr>
<td>96.8</td>
<td>17.8</td>
<td>92.7</td>
<td></td>
</tr>
<tr>
<td>5.8</td>
<td>2.5</td>
<td>4.3</td>
<td></td>
</tr>
<tr>
<td>4.9</td>
<td>3.9</td>
<td>4.0</td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>EE</td>
<td>LB</td>
<td>LG</td>
</tr>
<tr>
<td>134.3</td>
<td>29.1</td>
<td>45.9</td>
<td></td>
</tr>
<tr>
<td>5.2</td>
<td>2.1</td>
<td>2.3</td>
<td></td>
</tr>
<tr>
<td>5.9</td>
<td>2.3</td>
<td>1.9</td>
<td></td>
</tr>
</tbody>
</table>

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.

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.

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 [17, 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, 8, 9]. 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