Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science

Electronic Notes in Theoretical Computer Science 322 (2016) 153–168

www.elsevier.com/locate/entcs

# On the Maximum Betweenness Improvement Problem

Gianlorenzo D'Angelo[1] Lorenzo Severini[1] Yllka Velaj[1]

*Gran Sasso Science Institute (GSSI)*
*Viale F. Crispi 7, L'Aquila, Italy.*

**Abstract**

The betweenness is a well-known measure of centrality of a node in a network. We consider the problem of determining how much a node can increase its betweenness centrality by creating a limited amount of new edges incident to it. If the graph is directed, this problem does not admit a polynomial-time approximation scheme (unless $P = NP$) and a simple greedy approximation algorithm guarantees an almost tight approximation ratio [10].
In this paper we focus on the undirected graph case: we show that also in this case the problem does not admit a polynomial-time approximation scheme (unless $P = NP$). Moreover, we show that, differently from the directed case, the greedy algorithm can have an unbounded approximation ratio. In order to test the practical performance of the greedy algorithm, we experimentally measured its efficiency in term of ranking improvement, comparing it with another algorithm that simply adds edges to the nodes that have highest betweenness. Our experiments show that the greedy algorithm adds only few edges in order to increase the betweenness of a node and to reach the top positions in the ranking. Moreover, the greedy algorithm outperforms the second approach.

*Keywords:* Betweenness centrality, approximation algorithms, graph augmentation

## 1 Introduction

One of the main goals of network analysis is that of determining the most important nodes in a given complex network. Several measures of importance have been explicitly formalized in the literature to try to quantify how much a node is important (or "*central*"). The way of defining such so called *centrality measures* depends on the particular feature of the network that the measure wants to capture.

One of the most popular measures of importance is the *betweenness centrality* (see, for example, [8]). It intuitively quantifies how much a node controls the information flow between all pairs of nodes in a graph. More formally, the betweenness centrality of a given node $v$ is the portion of shortest paths that pass through $v$

[1] Email: gianlorenzo.dangelo@gssi.infn.it, lorenzo.severini@gssi.infn.it, yllka.velaj@gssi.infn.it

http://dx.doi.org/10.1016/j.entcs.2016.03.011

over all the possible shortest paths between all pairs of nodes different from $v$. Having high betweenness centrality can have positive impact on the node itself. Let us consider a network where there are messages passing along the edges by using shortest paths (e.g. information is a social network or packets in the Internet). Then, the number of messages passing through a given node $v$ is, to some extent, proportional to the number of shortest paths passing through $v$. Hence, a node with high betweenness has an high probability of receiving an high number of messages and hence it is central.

Computing betweenness centrality of a node or its ranking in the network can be done in polynomial time but requires $O(nm)$ time [9] on unweighted graphs which is clearly infeasible for huge networks. Therefore, several randomized or approximation algorithms have been proposed [7,14,22]

Besides computing the betweenness centrality, another interesting problem is that of *increasing* the betweenness centrality of a given node. Increasing the centrality of a node can clearly have positive consequences on the node itself. For example, in the field of transportation network analysis, the betweenness centrality seems to be positively related to the efficiency of an airport (see [17] where a network of 57 European airports has been analyzed). On the other hand, in many complex networks, a node can decide to connect itself to some other nodes in the network. Therefore, in this paper, we focus on the problem of maximizing the betweenness centrality of a node by adding a limited number of edges incident to it. More specifically, we consider the problem of efficiently determining, for a given node $v$, the set of $k$ edges incident to $v$ that, when added to the original graph, allows $v$ to increase as much as possible its betweenness centrality (and as a consequence its ranking according to this measure).

### *1.1 Related work*

The problem of increasing the centrality of a node in a network has been studied for several centrality measures different from betweenness, i.e. page-rank [4,20], eccentricity [11], stress and some measures related to the number of paths passing through a given node [15], closeness [10], and average distance [18].

Regarding betweenness centrality, if the network is directed and the arcs to be added are all directed towards node $v$, it has been shown that the problem does not admit a polynomial-time approximation scheme (unless $P = NP$) and that a simple greedy approximation algorithm exhibits an almost tight approximation ratio [10]. In detail, the problem cannot be approximated within a factor of $1 - \frac{1}{2e}$ and the greedy algorithm guarantees an approximation factor of $(1 - \frac{1}{e})$. The main part of the proof of the approximation ratio of the greedy algorithm consists in proving that the objective function is monotone and submodular which is not true for the undirected case.

### 1.2 Our results

In this paper we focus on undirected graphs and show that, also in this case, the problem is hard to be approximated within an approximation factor greater than $1 - \frac{1}{2e}$. Surprisingly, we show that the approximation ratio of the greedy algorithm can be arbitrarily small. This is in contrast with the results for the directed case. Another natural algorithm that, in the directed case, performs well in practice is the one that connects $v$ to the $k$ nodes that have the highest betweenness (and that are different from $v$ and its neighbors). Also in this case we show that the approximation ratio of such algorithm can be arbitrarily small.

Therefore, we study the practical performance of the aforementioned algorithms by means of experiments. We conducted two types of experiments: in the first type we measured the improvement in the value of betweenness of $v$ and, in the second type, we evaluate the improvement in the betweenness ranking of $v$ within the network. Our experiments show that the greedy algorithm adds only few edges in order to increase the betweenness of a node and to reach the top positions in the ranking. Regarding the algorithm that connects $v$ to the $k$ nodes that have the highest betweenness, we show that in many cases it requires to add many edges in order to significantly increase the ranking of a node.

### 1.3 Outline

In the next section we formally define the problem and give the notation used in the paper. In Section 3 we give the hardness of approximation result. In Section 4, we describe the two algorithms and show that their approximation ratio can be arbitrarily small. In Section 5 we present our experimental study. Section 6 concludes the paper and outlines some research directions.

## 2 Definitions and notation

Let $G = (V, E)$ be an undirected weighted graph, where the weights are given by function $w : E \to \mathbb{R}$. For each node $v$, $N_v$ denotes the set of neighbors of $v$, i.e. $N_v = \{u \mid \{u, v\} \in E\}$. Given two nodes $s$ and $t$, we denote by $d_{st}$, $\sigma_{st}$, and $\sigma_{stv}$ the distance between $s$ and $t$ in $G$, the number of shortest paths from $s$ to $t$ in $G$, and the number of shortest paths from $s$ to $t$ in $G$ that contain $v$, respectively. For each node $v$ the *betweenness centrality* [8] of $v$ is defined as

$$b_v = \sum_{\substack{s,t \in V \\ s \neq t; s,t \neq v \\ \sigma_{st} \neq 0}} \frac{\sigma_{stv}}{\sigma_{st}}.$$

Note that all the definitions hold also for the unweighted case since it is a particular instance of the weighted graph case where $w(u, v) = 1$, $\forall \{u, v\} \in E$.

The betweenness centrality of a node clearly depends on the graph structure: if we augment a graph by adding a set of edges $S$ having weight $\delta$, then the centrality of a node might change. Generally speaking, adding edges incident to some node $v$

can only increase the centrality of $v$. Given a set $S$ of edges not in $E$, we denote by $G(S)$ the graph augmented by adding the edges in $S$ to $G$, i.e. $G(S) = (V, E \cup S)$. For a parameter $x$ of $G$, we denote by $x(S)$ the same parameter in graph $G(S)$, e.g. the distance from $s$ to $t$ in $G(S)$ is denoted as $d_{st}(S)$.

We are interested in finding the set $S$ of edges incident to a particular node $v$ that maximizes $b_v(S)$. Therefore, we define the following optimization problem.

| **Maximum Betweenness Improvement (MBI)** | |
|---|---|
| **Given:** | An undirected weighted graph $G = (V, E)$ with weight function $w$; a node $v \in V$; an integer $k \in \mathbb{N}$, and a real number $\delta \in \mathbb{R}_0^+$ |
| **Solution:** | A set $S$ of edges having weight $\delta$ incident to $v$, $S = \{\{u, v\} \mid u \in V \setminus N_v\}$, such that $\lvert S\rvert \leq k$ |
| **Goal:** | Maximize $b_v(S)$ |

## 3 Hardness of approximation

In this section we prove that MBI cannot be approximated within a factor greater than $1 - \frac{1}{2e}$. The proof is based on an approximation preserving reduction to the Maximum Set Coverage (MSC) problem defined as follows. Given a ground set $X$, a family of subsets of $X$, $\mathcal{F} = \{S_1, S_2, \dots S_{|\mathcal{F}|}\}$, and an integer $k'$, find a family $\mathcal{F}' \subseteq \mathcal{F}$ such that $|\mathcal{F}'| \leq k'$ that maximize $s(\mathcal{F}') = |\cup_{S_i \in \mathcal{F}'} S_i|$.

**Theorem 3.1** *Problem* MBI *cannot be approximated within a factor greater than* $1 - \frac{1}{2e}$*, unless* $P = NP$*.*

**Proof.** We will give an $L$-reduction with parameters $a$ and $b$ [23]. In particular, we will give a polynomial-time algorithm that transforms any instance $I_{\text{MSC}}$ of MSC into an instance $I_{\text{MBI}}$ of MBI and a polynomial-time algorithm that transforms any solution $S$ for $I_{\text{MBI}}$ into a solution $\mathcal{F}'$ for $I_{\text{MSC}}$ such that the following two conditions are satisfied for some values $a$ and $b$:

$$OPT(I_{\text{MBI}}) \leq aOPT(I_{\text{MSC}}), \tag{1}$$

$$OPT(I_{\text{MSC}}) - s(\mathcal{F}') \leq b\left(OPT(I_{\text{MBI}}) - b_v(S)\right), \tag{2}$$

where $OPT$ denotes the optimal value of an instance of an optimization problem. If the above conditions are satisfied and there exists a $\alpha$-approximation algorithm for MBI, then there exists a $(1 - ab(1-\alpha))$-approximation algorithm for MSC [23]. Since it is $NP$-hard to approximate MSC within a factor greater than $1 - \frac{1}{e}$ [13], then $1 - ab(1-\alpha) < 1 - \frac{1}{e}$, unless $P = NP$. This implies that $\alpha < 1 - \frac{1}{abe}$.

Given an instance $I_{\text{MSC}} = (X, \mathcal{F}, k')$ of MSC, where $\mathcal{F} = \{S_1, S_2, \dots S_{|\mathcal{F}|}\}$, we define an instance $I_{\text{MBI}} = (G, v, k)$ of MBI, where:

- $G = (V, E)$;
- $V = \{v, t\} \cup \{v_{x_i} \mid x_i \in X\} \cup \{v_{S_j} \mid S_j \in \mathcal{F}\}$;

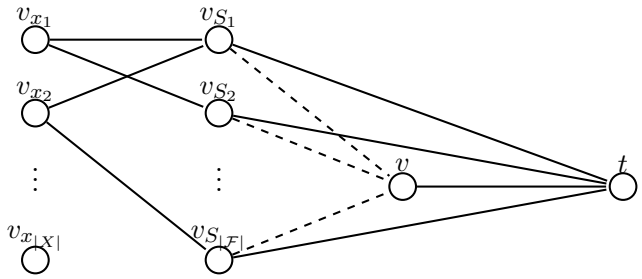



Fig. 1. Reduction used in Theorem 3.1. In the example $x_1 \in S_1$, $x_1 \in S_2$, $x_2 \in S_1$, and $x_2 \in S_{\mathcal{F}}$. The dashed edges denote those added in a solution.

- $E = \{(v,t)\} \cup \{\{v_{x_i}, v_{S_j}\} \mid x_i \in S_j\} \cup \{\{v_{S_j}, t\} \mid S_j \in \mathcal{F}\}$;
- $w(\{v_{x_i}, v_{S_j}\}) = 2\epsilon$, for each $x_i \in S_j$, $w(\{v_{S_j}, t\}) = 2 + \epsilon$, for each $S_j \in \mathcal{F}$, and $w(v,t) = 1$, where $0 < \epsilon < \min\{\frac{1}{4|X|}, \frac{1}{4|\mathcal{F}|}\}$;
- $\delta = 1$;
- $k = k'$.

See Fig.1 for a visualization.

First of all, note that since $\epsilon < \min\{\frac{1}{4|X|}, \frac{1}{4|\mathcal{F}|}\}$, then the distance between any two nodes in $\{v_{x_i} \mid x_i \in X\} \cup \{v_{S_j} \mid S_j \in \mathcal{F}\}$ is smaller than 1. Therefore no shortest path between any pair of such nodes can pass through $v$ by adding an edge of weight $\delta = 1$. Moreover, without loss of generality, we can assume that any solution $S$ to MBI contains only edges $\{v_{S_j}, v\}$ for some $S_j \in \mathcal{F}$. In fact, if a solution does not satisfy this property, then we can improve it in polynomial time by repeatedly apply the following rule.

- If $S$ contains an edge $\{v_{x_i}, v\}$, for some $x_i \in X$, then exchange such edge with an edge $\{v_{S_j}, v\}$ such that $x_i \in S_j$ and $\{v_{S_j}, v\} \notin S$ if it exists or remove such an edge. Note that if no edge $\{v_{S_j}, v\}$ such that $x_i \in S_j$ and $\{v_{S_j}, v\} \notin S$ exists, then all the shortest paths from $x_i$ to $t$ pass through $v$ and therefore the edge $\{v_{x_i}, v\}$ can be removed without changing the value of $b_v(S)$.

The above rule increases the value of $b_v(S)$. In fact, all the shortest paths passing through $v$ in the original solution still passes through $v$ also in the obtained solution. Moreover, if Condition (2) is satisfied for the obtained solution, then it is satisfied also for the original solution.

In such a solution, the distance between nodes $v_{S_j}$ and $t$ is either 2, if $\{S_j, v\} \in S$, or $2 + \epsilon$, if $\{S_j, v\} \notin S$. In the former case, all the shortest paths from $v_{S_j}$ to $t$ pass through $v$ and therefore the ratio $\frac{\sigma_{v_{S_j} tv}(S)}{\sigma_{v_{S_j} t}(S)}$ is 1, in the latter case no shortest path from $v_{S_j}$ to $t$ pass through $v$ and therefore such ratio is 0. Similarly, the distance between nodes $v_{x_i}$ and $t$ is either $2 + 2\epsilon$, if $x_i \in S_j$ for some $\{S_j, v\} \in S$, or $2 + 3\epsilon$, if $x_i \notin \cup_{\{S_j, v\} \in S} S_j$. Therefore, the ratio $\frac{\sigma_{v_{x_i} tv}(S)}{\sigma_{v_{S_j} t}(S)}$ is 1 if $x_i \in S_j$, for some $\{S_j, v\} \in S$, otherwise it is 0. As already mentioned, no shortest path between any pair of nodes in $\{v_{x_i} \mid x_i \in X\} \cup \{v_{S_j} \mid S_j \in \mathcal{F}\}$ can pass through $v$ by adding an edge of weight

$\delta = 1$. Therefore, for such pairs of nodes $(x, y)$ the ratio $\frac{\sigma_{xyv}}{\sigma_{xy}}$ is equal to 0.

Given a solution $S = \{\{v_{S_j}, v\} \mid S_j \in \mathcal{F}\}$ to MBI, we obtain the solution $\mathcal{F}' = \{S_j \mid \{v_{S_j}, v\} \in S\}$ to MSC. By construction, $|\mathcal{F}'| = |S| = k = k'$. Therefore, the betweenness centrality of $v$ in $G(S)$ is:

$$\begin{aligned} b_v(S) = & \sum_{\substack{x_i \in X \\ x_i \in S_j, \{v_{S_j}, v\} \in S}} \frac{\sigma_{v_{x_i} t v}(S)}{\sigma_{v_{x_i} t}(S)} + \sum_{\substack{S_j \in \mathcal{F} \\ \{v_{S_j}, v\} \in S}} \frac{\sigma_{v_{S_j} t v}(S)}{\sigma_{v_{S_j} t}(S)} \\ = & |\{x_i \in S_j \mid \{v_{S_j}, v\} \in S\}| + |\{S_j \mid \{v_{S_j}, v\} \in S\}| \\ = & \left| \bigcup_{S_j \in \mathcal{F}'} S_j \right| + |\mathcal{F}'| \\ = & s(\mathcal{F}') + k. \end{aligned}$$

It follows that Conditions (1) and (2) are satisfied for $a = 2$, $b = 1$ since: $OPT(I_{\text{MBI}}) = OPT(I_{\text{MSC}}) + k \leq 2OPT(I_{\text{MSC}})$ and $OPT(I_{\text{MSC}}) - s(\mathcal{F}') = OPT(I_{\text{MBI}}) - b_v(S)$. The first inequality is due to the fact that $OPT(I_{\text{MSC}}) \geq k$, since, if $OPT(I_{\text{MSC}}) < k$, then the greedy algorithm finds an optimal solution for MSC. The statement follows by plugging the values of $a$ and $b$ into $\alpha < 1 - \frac{1}{abe}$. □

## 4 Algorithms

In this paper we analyze the following two natural algorithms.

- **Greedy**. The greedy algorithm starts with the empty set, and repeatedly adds an edge $\{u, v\}$, where $u \notin N_v(S)$, that mostly increases the value of $b_v(S \cup \{\{u, v\}\})$. The algorithm is given in Figure 2. Since it requires to compute $k \cdot n$ times the betweenness of $v$ on a graph that has at most $m + k = O(m + n)$ edges, the algorithm requires $O(kn \cdot g(n, m + k))$, where $g(n, m)$ is the complexity of computing $b_v$.
- **TopK**[21]. The algorithm adds $k$ edges between $v$ and the $k$ nodes not in $N_v$ having the highest betweenness in $G$. The algorithm is given in Figure 3. Since this algorithm computes only one value of betweenness, it requires $O(g(n, m) + n \log n)$ computational time.

In the directed version of the problem in which we want to find a set $S$ of arcs directed towards $v$, it has been shown that the greedy algorithm almost matches the approximation lower bound, exhibiting an approximation ratio of $1 - \frac{1}{e}$ [10]. The proof of such statement was not reported in [10]Surprisingly, the same algorithm does not provide the same approximation ratio in the case of undirected graphs. Indeed, in the remainder of the section we show that both the above algorithms can have an arbitrarily small approximation ratio in the worst case. The bounds are given for unweighted graphs but note that hold also for the weighted case where $w(u, v) = 1$, $\forall \{u, v\} \in E$ and $\delta = 1$. The main difference between the directed and the undirected case is that in the former case any shortest path passing through $v$

**Algorithm**: GREEDY
**Input** : An undirected graph $G = (V, E)$; a node $v \in V$; an integer $k \in \mathbb{N}$, and a real number $\delta \in \mathbb{R}_0^+$
**Output**: set $S$ of edges having weight $\delta$ incident to $v$, $S = \{\{u, v\} \mid u \in V \setminus N_v\}$, such that $|S| \leq k$

```
S := ∅;
for i = 1, 2, ..., k do
    foreach u ∈ V \ N_v(S) do
        Compute b_v(S ∪ {{u, v}}), where w({u, v}) = δ;
    u_max := arg max{b_v(S ∪ {{u, v}}) | u ∈ V \ N_v(S)};
    S := S ∪ {{u_max, v}};
return S;
```

Fig. 2. Greedy algorithm.

**Algorithm**: TOPK
**Input** : An undirected graph $G = (V, E)$; a node $v \in V$; an integer $k \in \mathbb{N}$, and a real number $\delta \in \mathbb{R}_0^+$
**Output**: set $S$ of edges having weight $\delta$ incident to $v$, $S = \{\{u, v\} \mid u \in V \setminus N_v\}$, such that $|S| \leq k$

```
S := ∅;
foreach u ∈ V do
    Compute b_u
Let V be the set of nodes V sorted in non-increasing order according to b_u;
for i = 1, 2, ..., k do
    S := S ∪ {{V[i], v}}
return S;
```

Fig. 3. TOPK algorithm.

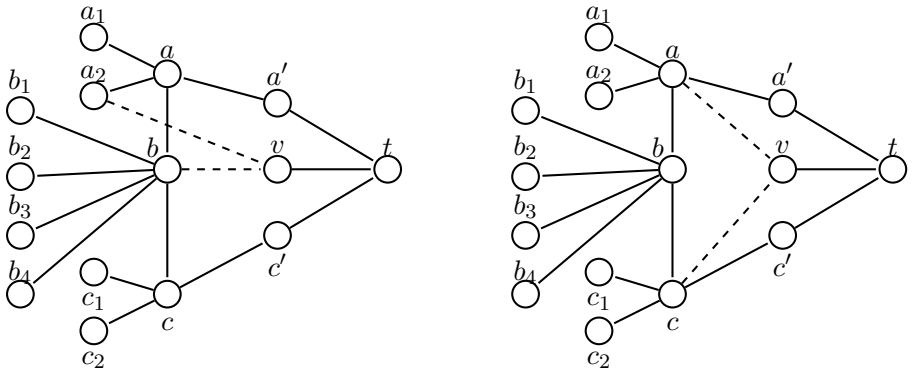
can use at most one of the inserted edges (being all of them directed towards $v$), while in the latter case a shortest path can in principle use two of such edges.

From an experimental point of view, in terms of solution quality, the greedy algorithm outperforms the TOPK algorithm in directed graphs [10]. However, the TOPK algorithm requires a smaller computational time.

### *4.1 Worst case approximation ratio of the greedy algorithm*

We now show that the greedy algorithm exhibits an arbitrary small approximation ratio. Consider the following instance of MBI, see Figure 4 for an example.

- Graph $G = (V, E)$.
- $V = \{v, t, a, b, c, a', b', c'\} \cup A \cup B \cup C$, where $A = \{a_i\}_{i=1}^{y}$, $B = \{b_i\}_{i=1}^{x}$, $C = \{c_i\}_{i=1}^{y}$, and $y = x - 2$, for some $x > 2$;




(a) Greedy algorithm (b) Optimal solution

Fig. 4. Counterexample for the greedy algorithm for $x = 4$. The dashed edges are those in a solution to MBI. The greedy algorithm (left) in the first iteration adds edge $\{b, v\}$ since it increases the most the centrality of $v$. After adding such edge the new value of $b_v$ is 5. In the second iteration the algorithm adds edge $\{a_2, v\}$, then the value of $b_v$ becomes 11. An optimal solution (right), has value $b_v(\{\{a, v\}, \{c, v\}\}) = 13$.

- $E = \{\{v, t\}, \{a, b\}, \{b, c\}, \{a, a'\}, \{b, b'\}, \{c, c'\}, \{a', t\}, \{b', t\}, \{c', t\}\} \cup \{\{a_i, a\} \mid a_i \in A\} \cup \{\{b_i, b\} \mid b_i \in B\} \cup \{\{c_i, c\} \mid c_i \in C\}$;
- All the edges have weight 1;
- $k = 2$.

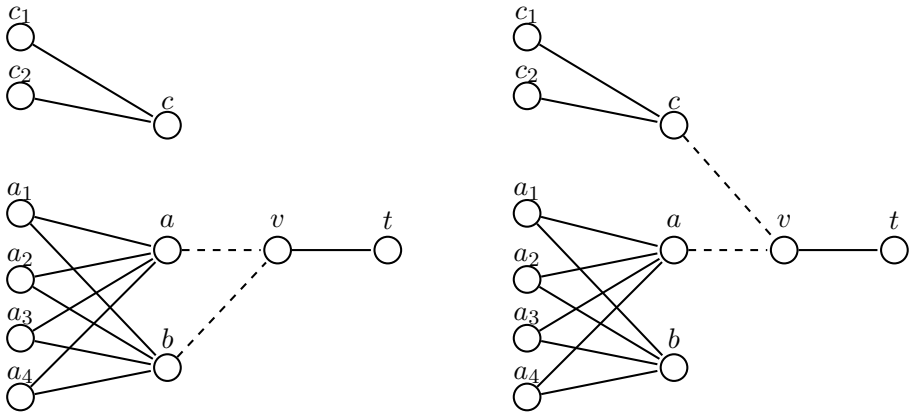
The initial value of $b_v$ is zero. The greedy algorithm first chooses edge $\{b, v\}$ and then edge $\{a_i, v\}$, for some $a_i \in A$ (or equivalently $\{c_i, v\}$, for some $c_i \in A$). The value of $b_v(\{b, v\}, \{a_i, v\})$ is $2x + 3$. In fact, the following pairs have shortest paths passing through $v$ in $G(\{b, v\}, \{a_i, v\})$: nodes in $B \cup \{b\}$ and $t$ ($x+1$ shortest paths), $a_i$ and $t$ (1 shortest path), $a_i$ and nodes in $B \cup \{b\}$ ($\frac{x+1}{2}$ shortest paths), $a_i$ and nodes in $C \cup \{c\}$ ($\frac{y+1}{2}$ shortest paths), and $a_i$ and $c'$ (1 shortest path). An optimal solution, instead, is made of edges $\{a, v\}$ and $\{c, v\}$ where $b_v(\{\{a, v\}, \{c, v\}\}) = \frac{x^2+3x-2}{2}$, where the quadratic term comes from the fact that there are $(y + 1)^2$ paths passing through $v$ between nodes in $A \cup \{a\}$ and nodes in $C \cup \{c\}$. Therefore, the approximation ratio of the greedy algorithm tends to be arbitrarily small as $x$ increases. The bad approximation ratio of the greedy algorithm is due to the fact that it does not consider the shortest paths that pass through $v$ by using both edges.

### *4.2 Worst case approximation ratio of the* TopK *algorithm*

We now show that also the TopK algorithm has an arbitrary small approximation ratio.

Consider the following instance of MBI, see Figure 5 for an example.

- graph $G = (V, E)$.
- $V = \{v, t, a, b, c\} \cup A \cup C$, where $A = \{a_i\}_{i=1}^{x}$, $C = \{c_i\}_{i=1}^{y}$, and $y = x - 2$, for some $x > 2$;
- $E = \{\{v, t\}\} \cup \{\{a_i, a\} \mid a_i \in A\} \cup \{\{a_i, b\} \mid a_i \in A\} \cup \{\{c_i, c\} \mid c_i \in C\}$;




(a) TopK algorithm (b) Optimal solution

Fig. 5. Counterexample for the TopK algorithm for $x = 4$. The dashed edges are those in a solution to MBI. The TopK algorithm (left) adds edges $\{a, v\}$ and $\{b, v\}$ and the value $b_v(\{a, v\}, \{b, v\})$ is 6.20. An optimal solution (right), has value $b_v(\{\{a, v\}, \{c, v\}\}) = 27$.

- All the edges have weight 1;
- $k = 1$.

The initial values of betweenness are $b_v = 0$, $b_a = b_b = \frac{x(x-1)}{4}$, $b_c = \frac{y(y-1)}{2}$, $b_{a_i} = \frac{1}{x}$, for each $i = 1, 2, \ldots, x$. Therefore, the two nodes with the highest betweenness are $a$ and $b$ and the TopK algorithm adds edges $\{a, v\}$ and $\{b, v\}$. The solution obtained has a value $b_v(\{a, v\}) = x + 2 + \frac{1}{x+1}$, since there are $x + 2$ paths between nodes in $A \cup \{a, b\}$ and $t$ passing through $v$ and 1 path over $x + 1$ paths from $a$ to $b$ passing through $v$. Adding edges $\{a, v\}$ and $\{c, v\}$, instead increases $b_v$ by $y \cdot x + y + 1 + x + 1 + y + x + 1 + x = x^2 + 3x - 1$, since all the paths from $A \cup \{a, b\}$ to $C \cup \{c\}$ pass through $v$. Therefore, the approximation ratio of the TopK algorithm tends to be arbitrarily small as $x$ increases.

## 5 Experiments

In this section we report the results of our experimental study. We conducted two types of experiments: in the first type we measured the improvement in the value of betweenness of $v$ and, in the second type, we evaluate the improvement in the betweenness ranking of $v$ within the network. These experiments are conducted on both synthetic and real-world networks. All our experiments have been performed on a computer equipped with an AMD Opteron 6376 CPU with 16 cores clocked at 2.30GHz and 64GB of main memory, and our programs have been implemented in `C++` (gcc compiler v4.8.2 with optimization level O3).

We executed the experiments on three types of randomly generated networks, namely Preferential Attachment (in short, `PA`) [5], Erdős-Rényi (in short, `ER`) [12], and Configuration (in short, `CONF`) [6,19] and on several real-world graphs. The size of the graphs is reported in Table 1 and in Table 2. All the edges have unitary weights and $\delta = 1$.

For each random model, we generated five graphs and used five nodes as $v$. These nodes have been chosen on the basis of their original betweenness ranking. In particular, we divided the list of nodes sorted by their original ranking in five parts and choose the nodes in the boundaries. We denote by $v_{X\%}$ the node on the boundary of the top $X$th percentile (e.g. $v_{25\%}$ is a nodes on the boundary of the top 25th percentile). The value of $k$ ranges from 1 to 20 for both Greedy and TopK algorithms. We show the results for three different random graphs: a `CONF` graph ($n = 100$, $m \approx 200$), an `ER` graph ($n = 100$, $m = 200$) and a `PA` graph ($n = 100$, $m \approx 200$). The results of `CONF` are plotted in Fig. 6. In the two top charts we plot the betweenness centrality and the ranking of vertex $v$ as a function of $k$. We observe that any vertex become central by adding just few edges. For example a vertex with the smallest betweenness centrality which initially has betweenness 0 and is ranked 100, improves its betweenness and ranking to 615.32 and 5, respectively, by adding only 6 edges. In the charts on the bottom, we compare the Greedy algorithm with the TopK algorithm that adds the edges from the $k$ vertices with the highest betweenness centrality to $v$. We report the ratio between the betweenness value obtained by the TopK algorithm and that obtained by our one. The experiments show that the Greedy algorithm outperforms the TopK approach. In fact, the solution computed by this latter is worst in terms of ranking as has been shown in the chart on the right bottom. In this case the node with smallest betweenness centrality needs 19 edges to be added in order to be in the top-5 nodes of the ranking. The results for `PA` and `ER` are similar and are plotted in Fig. 7 and in Fig. 8.

In Fig. 9 we show the comparison between the Greedy algorithm with the TopK considering the average value of the ratio between the betweenness value obtained by the TopK algorithm and that obtained by our one. The average is computed considering the five nodes $v$ from the five random graphs `CONF`, `PA` and `ER` with $n = 100$ and $m = 200$. Its easy to see that the Greedy algorithm is by far better than the TopK one. The worst performance of the TopK algorithm can be explained as follows: let $u$ be a node with high betweenness centrality, hence short paths between other nodes pass through it, adding an edge to $u$ will not change these paths because they are short. Therefore, adding an edge to connect to high betweenness node may not improve the betweenness value so much. The same positive result holds for the improvement in the ranking position for the node $v$.

For the second type of experiments we analyse four real-world networks: `Jazz` is the collaboration network between Jazz musicians that have played together in a band, `Karate` is Zachary karate club network where edges represent a tie between two members of the club. These networks are taken from the Konect repository [16]. `Easyjet` is a network obtained by crawling the EasyJet website [1]. `Coli1` is a biological network taken from Uri AlonLab [2]. The size of the networks is reported in Table 2. As in the previous experiments, for each graph we used five nodes as $v$: one of the nodes on the boundary of the top 25th (respectively, 50th and 75th) percentile, and one of the nodes with the smallest betweenness centrality. The value of $k$ ranges from 1 to 20.

| **Network** | $n = \|V\|$ | $m = \|E\|$ |
|---|---|---|
| PA | 50 | ≈100 |
| PA | 100 | ≈ 200 |
| ER | 50 | 100, 250, 500 |
| ER | 100 | 200, 500, 1000 |
| CONF | 50 | ≈100 |
| CONF | 100 | ≈200 |

Table 1
Betweenness centrality:type and size of the generated random graphs.

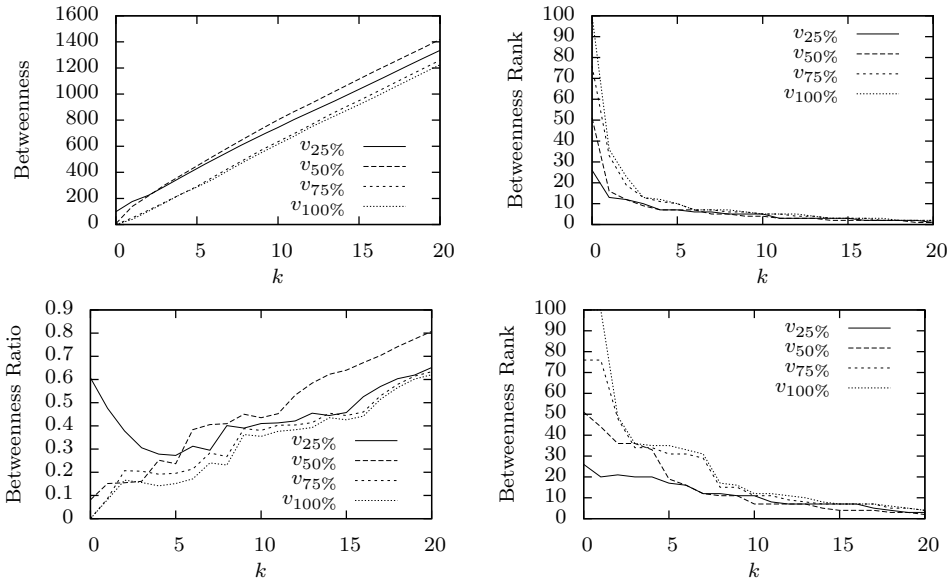



Fig. 6. Betweenness centrality: (Top Left) Betweenness value computed by the Greedy algorithm on network `Conf` and (Top Right) ranking of nodes. (Bottom Left) Comparison of the Greedy algorithm with the TopK method on network `Conf`. (Bottom Right) Ranking computed by the TopK method on network `Conf`.

The results for the `Jazz` are plotted in Fig. 10. As in the previous case, in the two top charts we plot the betweenness centrality and the ranking of vertex $v$ as a function of $k$. It is easy to observe that any vertex becomes central by adding just few edges. For example a vertex with the smallest betweenness centrality which initially has betweenness 0 and is ranked 197, improves its betweenness and ranking to 748.36 and 5, respectively, by adding only 6 edges. In the charts on the bottom, we compare the Greedy algorithm with the TopK algorithm. We report the ratio between the betweenness value obtained by the TopK algorithm and that obtained by Greedy. The experiments show that the greedy algorithm outperforms the TopK approach. In fact, the solution computed by this latter is worst in terms of

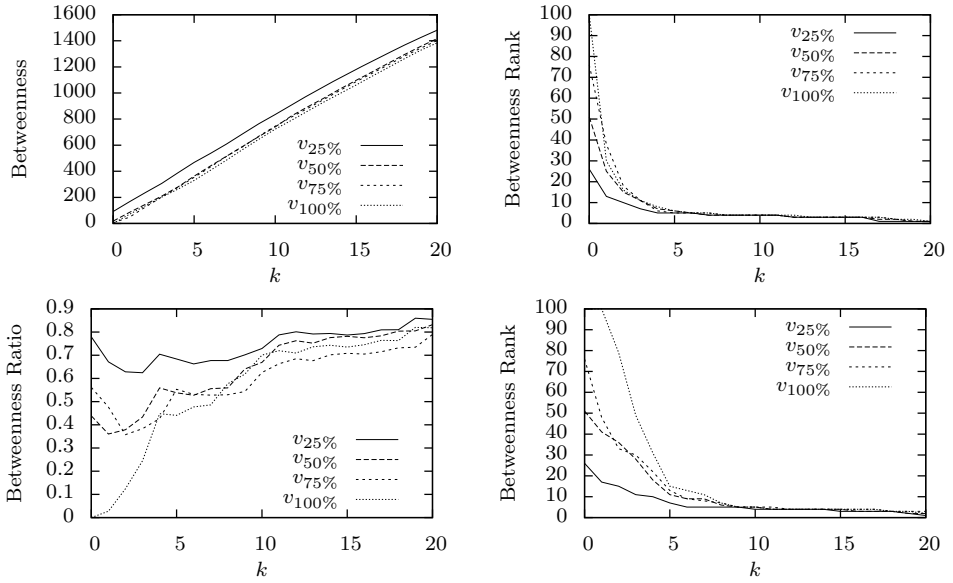



Fig. 7. Betweenness centrality: (Top Left) Betweenness value computed by the GREEDY algorithm on network PA and (Top Right) ranking of nodes. (Bottom Left) Comparison of the GREEDY algorithm with the TOPK method on network PA. (Bottom Right) Ranking computed by the TOPK method on network PA.

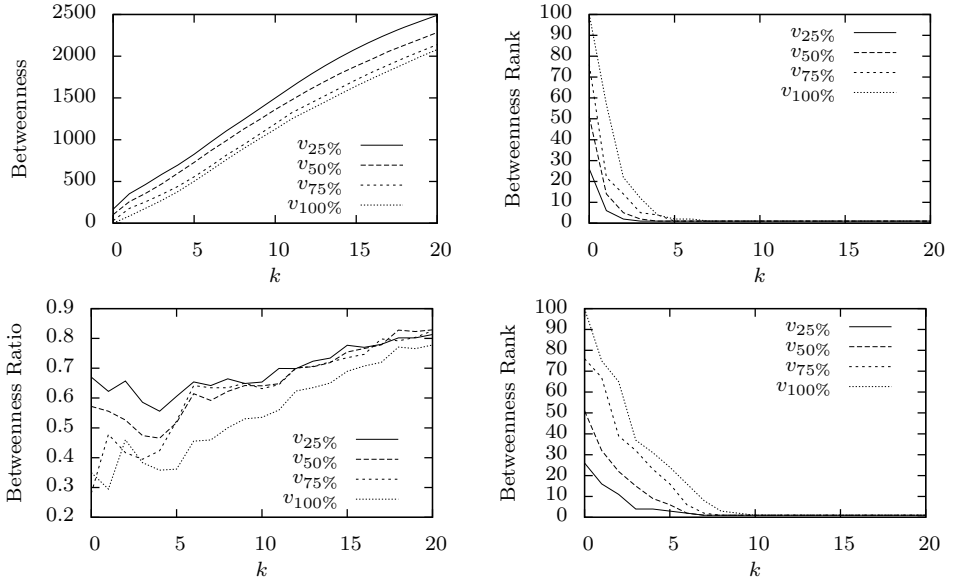



Fig. 8. Betweenness centrality: (Top Left) Betweenness value computed by the GREEDY algorithm on network ER and (Top Right) ranking of nodes. (Bottom Left) Comparison of the GREEDY algorithm with the TOPK method on network ER. (Bottom Right) Ranking computed by the TOPK method on network ER.

ranking as has been shown in the chart on the right bottom. In this case the node with smallest betweenness centrality needs more than 20 edges to be added in order to be in the top-5 nodes of the ranking. The results for Easyjet are similar and are

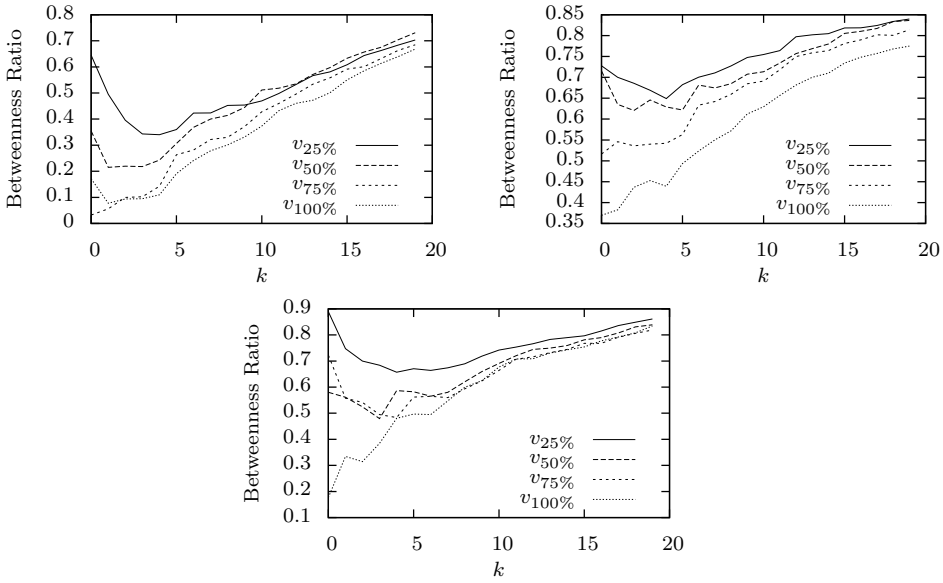



Fig. 9. Betweenness centrality: Average value of the comparison of the Greedy algorithm with the TopK method on random networks. `CONF` (top left), `ER` (top right), and `PA` (bottom)

| **Network** | $n$ | $m$ | **25th** | | **50th** | | **75th** | | **Last** | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $\|V\|$ | $\|E\|$ | $k_{\min}$ G | $k_{\min}$ T | $k_{\min}$ G | $k_{\min}$ T | $k_{\min}$ G | $k_{\min}$ T | $k_{\min}$ G | $k_{\min}$ T |
| `Karate` | 39 | 78 | 1 | 5 | 2 | 7 | 1 | 1 | 2 | 2 |
| `EasyJet` | 136 | 750 | 7 | $> 20$ | 9 | $> 20$ | 10 | $> 20$ | 12 | $> 20$ |
| `Jazz` | 198 | 2742 | 2 | 6 | 3 | 9 | 3 | 17 | 6 | $> 20$ |
| `Coli1` | 328 | 456 | 2 | 3 | 2 | 2 | 2 | 3 | 2 | 3 |

Table 2
Betweenness centrality: the first three columns report the type and size of the graphs. The last four report the minimum number $k_{min}$ of edges that have to be added in order to let the node reach the first 5 nodes in the final ranking. We distinguish between those added by the Greedy algorithm (G) and those added by the TopK (T) one.

plotted in Fig. 11.

For all the real-world graphs we report in the Table 2 the minimum number $k_{min}$ of edges that have to be added in order to let the node reach the first five nodes in the final ranking. As it can be seen from the table, very few edges are necessary to drastically increase the ranking of a node using the Greedy algorithm, many more are needed if we use the TopK one.

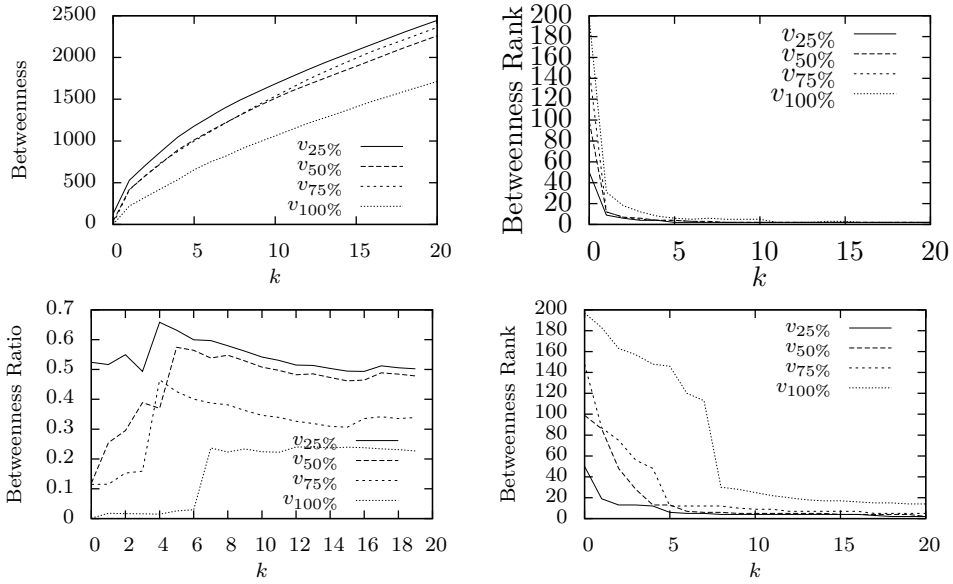



Fig. 10. Betweenness centrality: (Top Left) Betweenness value computed by the GREEDY algorithm on network `Jazz` and (Top Right) ranking of nodes. (Bottom Left) Comparison of the GREEDY algorithm with the TOPK method on network `Jazz`. (Bottom Right) Ranking computed by the TOPK method on network `Jazz`.

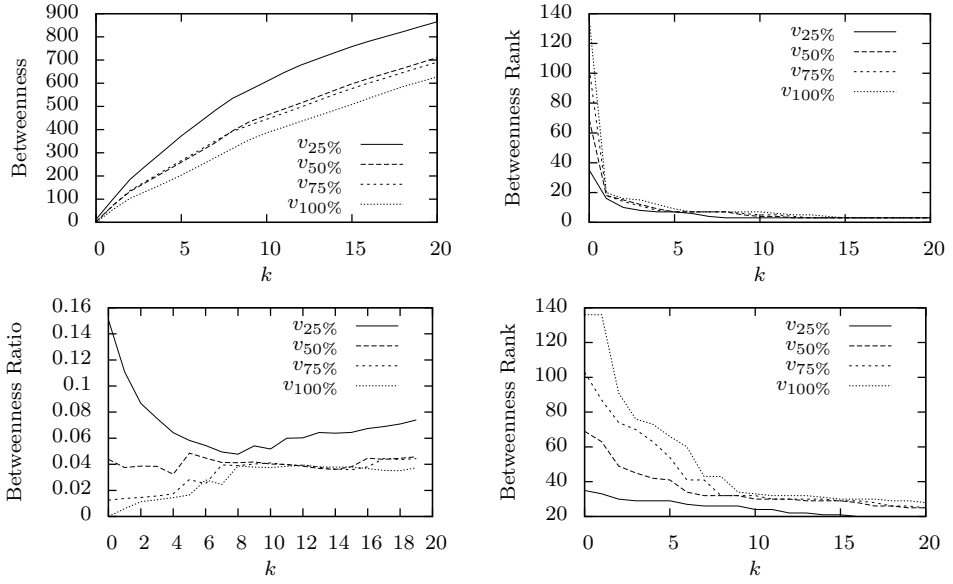



Fig. 11. Betweenness centrality: (Top Left) Betweenness value computed by the GREEDY algorithm on network `Easyjet` and (Top Right) ranking of nodes. (Bottom Left) Comparison of the GREEDY algorithm with the TOPK method on network `Easyjet`. (Bottom Right) Ranking computed by the TOPK method on network `Easyjet`.

# 6 Conclusion and future research

In this paper we studied the problem of computing a set of edges that a node can decide to add to a graph in order to increase its betweenness centrality. We have shown that the problem is hard to approximate within some constant factor. We have tested two algorithms on several relatively small random graphs and, then, applied to several real-world networks. The experiments show that the Greedy algorithm outperforms the TopK one in terms of solution quality and requires to add few edges in order to increase the ranking of a node to reach the top positions.

As future works, we plan to fill the gap between approximation guarantees and hardness of approximation, i.e. we plan to devise an algorithm with a constant approximation ratio. This task is challenging since in this paper we have shown that the algorithmic technique used for the directed case produces solutions with an unbounded approximation ratio in the worst case. Another interesting future research is to investigate if there exists an equilibrium for the greedy algorithm. Suppose that a node can remove an incident edge and replace it with a new one, then an equilibrium is a graph where no such move can increase the centrality of the corresponding node. Similar concepts were studied also for the network creation game [3].

# References

[1] Easyjet. http://www.easyjet.com. Accessed: 2015-01-15.

[2] Uri alonlab. http://www.weizmann.ac.il/mcb/UriAlon/. Accessed: 2015-01-15.

[3] N. Alon, E. D. Demaine, M. Hajiaghayi, and T. Leighton. Basic network creation games. In *Proceedings of the Twenty-second Annual ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '10, pages 106–113, New York, NY, USA, 2010. ACM.

[4] K. Avrachenkov and N. Litvak. The effect of new links on google pagerank. *Stoc. Models*, 22(2):319–331, 2006.

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

[6] E. A. Bender and E. Canfield. The asymptotic number of labeled graphs with given degree sequences. *Journal of Combinatorial Theory, Series A*, 24(3):296 – 307, 1978.

[7] E. Bergamini, H. Meyerhenke, and C. Staudt. Approximating betweenness centrality in large evolving networks. In *Proceedings of the Seventeenth Workshop on Algorithm Engineering and Experiments, ALENEX*, pages 133–146. SIAM, 2015.

[8] P. Boldi and S. Vigna. Axioms for centrality. *Internet Math.*, 10(3–4):222–262, 2014.

[9] U. Brandes. A faster algorithm for betweenness centrality. *J. Math. Sociol.*, 25(2):163–177, 2001.

[10] P. Crescenzi, G. D'Angelo, L. Severini, and Y. Velaj. Greedily improving our own centrality in a network. In *Proceedings of the 14th International Symposium on Experimental Algorithms (SEA 2015)*, volume 9125 of *Lecture Notes in Computer Science*, pages 43–55. Springer, 2015.

[11] E. D. Demaine and M. Zadimoghaddam. Minimizing the diameter of a network using shortcut edges. In *Proc. of the 12th Scandinavian Symp. and Work. on Algorithm Theory (SWAT)*, volume 6139 of *LNCS*, pages 420–431. Springer, 2010.

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

[13] U. Feige. A threshold of ln n for approximating set cover. *J. ACM*, 45(4), 1998.

[14] R. Geisberger, P. Sanders, and D. Schultes. Better approximation of betweenness centrality. In *Proceedings of the Tenth Workshop on Algorithm Engineering and Experiments, ALENEX*, pages 90–100. SIAM, 2008.

[15] V. Ishakian, D. Erdös, E. Terzi, and A. Bestavros. A framework for the evaluation and management of network centrality. In *Proc. of the 12th SIAM Int. Conf. on Data Mining (SDM)*, pages 427–438. SIAM, 2012.

[16] J. Kunegis. KONECT - The Koblenz network collection. In *Proc. of the 1st Int. Web Observatory Work. (WOW)*, pages 1343–1350, 2013.

[17] P. Malighetti, G. Martini, S. Paleari, and R. Redondi. The impacts of airport centrality in the EU network and inter-airport competition on airport efficiency. Technical Report MPRA-7673, 2009.

[18] A. Meyerson and B. Tagiku. Minimizing average shortest path distances via shortcut edge addition. In *Proc. of the 13th Int. Work. on Approx. Alg. for Comb. Opt. Prob. (APPROX)*, volume 5687 of *LNCS*, pages 272–285. Springer, 2009.

[19] M. Molloy and B. Reed. A critical point for random graphs with a given degree sequence. *Random Structures & Algorithms*, 6(2–3):161–180, 1995.

[20] M. Olsen and A. Viglas. On the approximability of the link building problem. *Theor. Comput. Sci.*, 518:96–116, 2014.

[21] R. Puzis, Y. Elovici, and S. Dolev. Finding the most prominent group in complex networks. *AI Commun.*, 20(4):287–296, dec 2007.

[22] M. Riondato and E. M. Kornaropoulos. Fast approximation of betweenness centrality through sampling. In *Proc. of the 7th ACM Int. Conf. on Web Search and Data Mining, (WSDM)*, pages 413–422. ACM, 2014.

[23] D. Williamson and D. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.