Detecting Communities Around Seed Nodes in Complex Networks

Christian L. Staudt, Yassine Marrakchi, Henning Meyerhenke
Department of Informatics, Karlsruhe Institute of Technology (KIT)
christian.staudt@kit.edu, yassine.marrakchi@student.kit.edu, meyerhenke@kit.edu

***Abstract*—The detection of communities (internally dense subgraphs) is a network analysis task with manifold applications. The special task of selective community detection is concerned with finding high-quality communities locally around seed nodes. Given the lack of conclusive experimental studies, we perform a systematic comparison of different previously published as well as novel methods. In particular we evaluate their performance on large complex networks, such as social networks. Algorithms are compared with respect to accuracy in detecting ground truth communities, community quality measures, size of communities and running time. We implement a generic greedy algorithm which subsumes several previous efforts in the field. Experimental evaluation of multiple objective functions and optimizations shows that the frequently proposed greedy approach is not adequate for large datasets. As a more scalable alternative, we propose selSCAN, our adaptation of a global, density-based community detection algorithm. In a novel combination with algebraic distances on graphs, query times can be strongly reduced through preprocessing. However, selSCAN is very sensitive to the choice of numeric parameters, limiting its practicality. The random-walk-based PageRankNibble emerges from the comparison as the most successful candidate.**

I. Introduction

Problem Motivation and Definition: Uncovering the community structure of real-world networks regularly yields interesting empirical insights about complex systems, and also has promising applications in the electronic networks that surround us. At the same time, community detection is a challenging graph mining and algorithm engineering problem, partly because of the imprecise concept of community structure in networks. Most formalizations (e.g. modularity, conductance) lead to $\mathcal{NP}$-hard optimization problems, but efficient heuristics have been created and successfully applied, and continue to be developed. In this work we consider the problem of quickly finding high-quality communities around given seed nodes, particularly in large complex networks. We refer to this task as *selective community detection* (SCD, also called local community detection or seed set expansion) to distinguish it from global community detection. In the global scenario, the graph is considered as a whole and an assignment of each node to a community is sought. In contrast, selective community detection begins with a small set of seed nodes as input and finds appropriate communities containing them, obtained by searching the network locally around the seed nodes. Such a targeted approach provides potential for speedup when solving the problem globally is not necessary or infeasible. Some SCD algorithms also enable us to find query-specific communities, assuming that the best community for seed node s_1 is different from the one for a nearby seed node s_2. SCD methods are especially applicable when we lack global knowledge of the network, e.g. in scenarios where the network structure is discovered on-the-fly. While it may seem that the difference between the global and selective scenario lies only in the amount of work performed, we show that the methods needed are somewhat different. Given these differences, applications are as numerous as for global community detection, and include e.g. finding functional complexes for a given protein in protein interaction networks [27]. The existing literature defines the task in slightly different ways, leading to a diverse set of SCD algorithms. We therefore begin by specifying the task as follows: Given a graph $G = (V, E)$ and a set of seed nodes $S \subset V$, return an assignment of each seed $s \in S$ to a community $C \subset V$ so that s is contained in C and all communities are pairwise disjoint. Within this definition, the following aspects need clarification: We discuss in Sec. II when a subset of nodes is considered a good community. The output is an assignment of seed node to community. Depending on the algorithm, the communities may overlap or coincide. We do not require the seed nodes to be structurally close to each other – the considered algorithms can handle both related and unrelated seeds appropriately.

Methods: While a variety of methods have been proposed, we observe a lack of conclusive experimental studies demonstrating their practical relevance. With our comparative study, we aim to close this gap. We are also the first to target large complex networks in the order of 10^5 to 10^6 of edges, requiring scalable algorithms and implementations. After a review of previous work on the subject (Sec. III), algorithmic approaches are compared and classified. We identify one widely used approach which we call Greedy Community Expansion (GCE, Sec. V-A). With our generic implementation of GCE, we evaluate existing objective functions and optimizations. In our experimental comparison we include a reimplementation of the random-walk based PageRankNibble [1], representing an important class of approaches to the problem. Furthermore, as our main algorithmic innovation, we adapt a global algorithm to the selective scenario (Sec. V-B): A modification of the density-based SCAN algorithm yields our variant selSCAN. The algorithm is generic with respect to a node distance measure, and we propose algebraic distances as an alternative to the original measure. The performance of algorithms is experimentally evaluated

with respect to accuracy, quality, community sizes and running time (Sec. VI). Accuracy is measured as the agreement with ground truth communities on synthetic LFR graphs, while quality is calculated with community quality measures which also serve as objective functions for GCE. We then apply the best performing algorithms to large real-world networks.

Results: We deliver an experimental comparison of different classes of SCD algorithms. After reimplementation and extensive analysis, we conclude that a widespread greedy quality optimization method is not likely to perform fast enough on large-scale graphs. A review of previously proposed objective functions narrows the choice for a viable function down to conductance (Φ). We show that the query time can become prohibitively high for networks with millions of edges. Especially for large networks, we propose selSCAN as a faster alternative. In contrast to most other methods, selSCAN also has a notion of outliers. Given an appropriate parameter choice, selSCAN is also qualitatively superior, although efficient parameter selection remains problematic. With selSCAN-AD, we explore a combination of density-based clustering and algebraic distance, which further reduces query time at the cost of precalculating node distances. From the comparison it becomes clear that PageRank-Nibble performs best in terms of running time and result quality.

II. Concepts and Definitions

Networks: Complex networks are a promising model for a wide range of systems and phenomena, including but not limited to social relations, the biochemical machinery of life, or the hyperlinked pages of the web [11]. In contrast to more regular graphs (e.g. technical meshes), these networks are commonly characterized as scale-free and small-world. Computational challenges arise from these properties: The presence of very high degree nodes and the small-world phenomenon imply that a large part of the graph is reachable within only a few hops from any seed node.

Graphs are the mathematical representation of networks. We denote a simple, undirected graph by $G = (V, E)$, where V is a node set of size n and $E \subseteq \binom{V}{2}$ is an edge set of size m. We consider unweighted graphs, but all methods discussed can be generalized to weighted graphs. The set $N(u) := \{v \in V : \{u, v\} \in E\}$ contains the neighbors of u, and the number of neighbors is called the degree $\deg(u)$ of u. In several cases we define the neighborhood of a node u as including u itself: $\Gamma(u) := N(u) \cup \{u\}$. Given a set of nodes A, its volume is the sum of degrees of nodes in A: $vol(A) = \sum_{v \in A} deg(v)$. For node sets A and B we denote the set of edges between them as $E(A, B) := \{\{u, v\} \in E : u \in A, v \in B\}$ with $E(A) := E(A, A)$.

Communities: A community is considered a subset of nodes that are more densely linked to each other than to the rest of the graph (intra-community density versus inter-community sparsity). The somewhat fuzzy intuitive definition leads to various formalizations, and no definitive consensus has emerged (see [20], [30] for a discussion). A quantification in the form of community quality measures generally leads to $\mathcal{NP}$-hard optimization problems.

Formally, a community detection solution $\zeta = \{C_1,....,C_k\}$ is a partition of the node set into disjoint subsets. For SCD we focus only on the communities that contain the seed nodes. We denote the set of internal edges of a community as $E_{\text{int}}(C) := E(C, C)$ and its external edges as $E_{\text{ext}}(C) := E(C, V \setminus C)$. Each community C induces three sets of nodes, a core, a boundary, and a shell. The core K of C are the nodes in C for which all neighbors are also in C: $K(C) := \{u \in C : \forall\{u, v\} \in E : v \in C\}$. Boundary nodes have neighbors both inside and outside of the community: $B(C) := \{u \in C : \exists\{u, v\} \in E : v \notin C\}$. Finally, C is surrounded by a shell of nodes which do not belong to C but have edges to nodes in C: $\Omega(C) := \{u \notin C : \exists\{u, v\} \in E : v \in C\}$.

III. Related Work

Though not nearly as extensively studied as the global scenario, SCD has been the focus of multiple previous publications. Because experimental data demonstrating their relative performance is scarce, we aim to clarify the state of the art and summarize and categorize previous efforts.

Global Community Detection: There has been extensive work on heuristics for global community detection, for which we refer the reader to existing surveys [20], [13]. In a previous work of ours [23], we developed fast parallel heuristics for global community detection, which could not be adapted to the SCD problem, though.

Community Expansion: A common approach to SCD starts from a seed node as a singleton community and expands the community one node at a time, selecting the candidate which gives the maximum gain for a community quality function. We will refer to this method as greedy community expansion. Examples of this approach are frequent in the literature and vary in the objective function and possible pre- and postprocessing optimizations. Clauset [9], Luo *et al.* [16], Chen *et al.* [8] and Bagrow [3] introduce different objective functions as "local modularity" (discussed in Sec. IV). An efficient implementation of the first three algorithms runs in $O(|C| \cdot d \cdot |\Omega(C)|)$ where d the average degree of nodes in the community and the last runs in $O(|C| \cdot log|C|)$. These publications consider only small to medium sized graphs in their evaluation.

Algorithms Based on Node Similarity: Rather than optimizing community quality one node at a time, this class of algorithms determines the similarity of candidate nodes with a given seed and finds a community among the most similar nodes. A common way to define this similarity is to perform a random walk from a seed, which is likely to get trapped in dense, community-like subgraphs. Nodes are then ordered by the resulting probability distribution and a high-quality community is found among the highest ranked nodes. Instances of this approach include [22], [1], [27]. For our experimental comparison, we implemented the PageRank-Nibble algorithm due to Andersen, Chung and Lang [1]. Treating the community around a seed node as a graph cut,

running time depends on the size of the small side of the cut rather than the size of the input graph. The algorithm offers theoretical guarantees with respect to the conductance of the cut.

Other Approaches: Apart from these main classes of algorithms, other approaches have also been explored. An example is bridge-bounding [18], which starts with a singleton community around a seed and attaches nodes from the shell as long as the edges connecting these nodes are not bridges. However, bridges are defined by globally scoring edges by centrality (e.g. by betweenness centrality). One of the few works targeting large networks [19] proposes an agglomerative algorithm: Starting from a handful of seed nodes and singleton communities, mergers are performed between communities containing a seed node and communities adjacent to them.

IV. Measuring Community Quality

In the following we explain and justify our choice of measures for the quality of a seeded community C_s. The measures we select then serve both as objective functions in GCE and quality criteria for the evaluation. On the right side of each function we denote the value range and whether it is maximized or minimized.

Discarding Modularity: Modularity [17] has become a popular measure for community quality, and several efficient heuristics for its $\mathcal{NP}$-hard [5] optimization have been published [10], [4]. The modularity of a partition $\zeta = \{C_1, ..., C_k\}$ into disjoint communities is defined as

$$mod(\zeta) := \sum_{C \in \zeta} \left(\frac{|E(C)|}{m} - \left(\frac{vol(C)}{2m} \right)^2 \right) \qquad [-0.5, 1] \text{ max}$$

Modularity is coverage (i.e. the fraction of internal edges) minus expected coverage under a null-model which preserves expected node degree. For global community detection, modularity has proven to be effective, despite a few drawbacks [14], [5]. We considered modularity in the context of selective community detection, but decided against it: Modularity depends strongly on the global structure of the graph via the number of edges and the global null-model and expects a full graph partition. In the remainder we examine community quality measures for a single community without global knowledge of the network.

Conductance: Consider a single community as a set of nodes cut from the rest of the graph. Sparsity of the cut is a necessary criterion for a good community. One important cut measure is conductance, the size of a cut divided by volume of the smaller section of the graph.

$$\Phi(C) := \frac{|E(C, V \setminus C)|}{\min\{vol(C), vol(V \setminus C)\}} \underset{|C| \ll |V|}{=} \frac{|E(C, V \setminus C)|}{vol(C)} \qquad [0, 1] \text{ min}$$

Minimizing the cut alone encourages small communities, while maximizing the volume requires larger ones. Minimizing conductance therefore helps to identify non-trivial subsets of nodes which are sparsely connected to the rest of the graph. It should be noted that determining the minimum conductance cut in a graph is $\mathcal{NP}$-hard [21]. Conductance is regularly used for measuring community quality locally (e.g. [27], [2]).

Custom measures for SCD: Considering that modularity is ill-suited as an objective function for SCD, several alternatives have been proposed under the name of "local modularity". In spite of the name, they are not directly related to each other and should not be considered localized variants of global modularity, since they do not rely on a null model. We will refer to these measures by their abbreviations in the literature instead. Clauset [9] defines R as a ratio of boundary edges to community nodes and all edges connected to boundary nodes:

$$R(C) := \frac{|E(B(C), C)|}{|E(B(C), V)|} \qquad [0, 1] \text{ max}$$

Luo *et al.* [16] propose M, which is the ratio of intra-community edges to inter-community edges:

$$M(C) := \frac{|E_{int}(C)|}{|E_{ext}(C)|} \qquad [0, \infty] \text{ max}$$

For the average internal degree in the community and the average external degree in its boundary, we obviously want to maximize the former and minimize latter, resulting in the L measure introduced by Chen *et al.* [8].

$$L(C) := \frac{2 \cdot |E_{int}(C)|}{|C|} \cdot \left(\frac{|E(B(C), \Omega(C))|}{|B(C)|} \right)^{-1} \qquad [0, \infty] \text{ max}$$

Selection of Measures: Previous empirical results [6], [28] show that M yields consistently better results than optimizing R and the measure proposed by Bagrow in terms of agreement with ground truth data. Furthermore, we show the equivalence of M and Φ as objective functions, a relationship not observed in [16].

Observation 1. *For a graph without self-loops and $|C| \ll |V|$, the following holds:*

$$\begin{aligned} \min! \, \Phi(C) &\iff \max! \, \Phi(C)^{-1} \\ &\iff \max! \left(\frac{|E_{ext}(C)|}{2 \cdot |E_{int}(C)| + |E_{ext}(C)|} \right)^{-1} \\ &\iff \max! \left(1 + 2 \cdot \frac{|E_{int}(C)|}{|E_{ext}(C)|} \right) \\ &\iff \max! \, M(C) \end{aligned}$$

Therefore, we will evaluate Φ and L as community quality measures and use M and L as objective functions for GCE to maximize.

V. Algorithms

A. GCE: Greedy Community Expansion

In the existing literature on SCD, many of the proposed algorithms are variations of one basic approach which we call greedy community expansion: A community C_s around seed

s is expanded by including the shell node which yields the highest gain ΔQ with respect to some community quality measure Q. After each inclusion, gains are recomputed for all candidates. Previously proposed greedy algorithms (e.g. [9], [16], [8], [28], [3]) vary in the objective function as well as different pre- and post-processing steps and optimizations. Many of these variants are similar enough to be implemented as one generic algorithm with exchangeable components. This yields GCE, which we use to evaluate the greedy method with respect to large complex networks. GCE has an exchangeable objective function as well as an optional optimization which we call acceptability. It is claimed to improve quality by prioritizing certain candidate nodes [28]. As explained in Sec. IV, we implement both L and M (equivalent to Φ) as objective functions. We indicate the configuration with the following naming scheme: For example, GCE-L optimizes L. In the following paragraphs, we give more details on objective functions and optimizations.

Objective Functions: We can efficiently calculate the quality gain for candidate nodes $v \in \Omega(C)$ in $O(\deg(v))$ by keeping track of interim values like the number of internal and external edges and the number of boundary nodes. Let $Q^{'}(C)$ be the current community quality and let $\deg_{\text{int}}(v, C) := |\{u \in C \cup \{v\} : \{u, v\} \in E\}|$ and $\deg_{\text{ext}}(v, C) := \deg(v) - \deg_{\text{int}}(v, C)$. Then the gain for M and L when moving the shell node v to the community C is

$$\Delta M(v, C) = \frac{|E_{\text{int}}(C)| + \deg_{\text{int}}(v, C)}{|E_{\text{ext}}(C)| - \deg_{\text{int}}(v, C) + \deg_{\text{ext}}(v, C)} - M^{'}(C)$$

$$\Delta L(v, C) = \left(\frac{2 \cdot (|E_{\text{int}}(C)| + \deg_{\text{int}}(v, C))}{|C| + 1}\right) \cdot \left(\frac{|E_{\text{ext}}(C)| - \deg_{\text{int}}(v, C) + \deg_{\text{ext}}(v, C)}{|B(C)| + \Delta|B(C)|}\right)^{-1} - L^{'}(C)$$

Each expansion step is followed by recalculation of the gains in $O(|\Omega(C)| \cdot d)$ where d is the average degree of shell nodes.

B. selSCAN: a Density-based Approach

As an alternative to community expansion we propose our adaptation of SCAN (Structural Clustering Algorithm for Networks) [29], a global community detection algorithm inspired by the data-clustering algorithm DBSCAN [12]. SCAN tries to transfer the characteristics of DBSCAN to community detection in networks. We briefly revisit the concepts on which SCAN is based. It operates with a similarity measure for pairs of connected nodes—we define it equivalently in terms of node distances in order to combine it with algebraic distances. The ϵ-neighborhood $N_\epsilon(v) = \{u \in N(v) : d(u, v) < \epsilon\}$ is the set of close neighbors which have at most ϵ distance from v. A node is a core if it has more than κ close neighbors, formally $\text{core}_{\kappa,\epsilon}(v) \iff |N_\epsilon(v)| \geq \kappa$. Two nodes from the graph are called density-connected if they are joined by a path where at least all inner nodes are cores: A SCAN community is then a maximal set of density-connected cores and their close neighbors. All remaining nodes are either hubs or outliers, i.e. central nodes lying between two or more communities or peripheral nodes not belonging to any community. SCAN runs in $O(m)$ time. We continue by describing selSCAN, our adaptation of SCAN to the SCD scenario. selSCAN receives as input a set of related or unrelated seed nodes and returns an assignment of seed to community, but in contrast to GCE each pair of communities is either disjoint or identical. Algorithm 1 denotes selSCAN in pseudocode. The algorithm considers each seed in turn, but can recognize that a seed belongs to a previously discovered community, saving time for sets of related seeds (line 1). Our adaptation differs from the original SCAN in the following aspects: We begin with seed nodes rather than random nodes. If a seed s is not a core, selSCAN tries to find a core in its neighborhood (line 8). If one is found, a community is constructed around it as denoted in Algorithm 2, which is best understood as a breadth-first search among close neighbors where only cores are added to the search queue. If no core is found, s is classified as an outlier (line 14). A distinction between hubs and outliers cannot be made, because this would require a global partition.

A downside which SCAN and selSCAN inherit from DBSCAN is that the method is not parameter-free: The result depends on the parameters μ and ϵ, which need to be estimated per network. We are aware of an algorithm [25] which has a similar community concept as SCAN and automatically determines an ϵ which is favorable for a community quality measure, but is based on a global spanning tree of the graph and deviates too strongly from the SCD scenario.

Algorithm 1: selSCAN: Selective Structural Clustering Algorithm for Networks

Input: Graph $G = (V, E)$, node distances d, seed set S, parameters κ, ϵ
Output: η: assignment of seed $s \in S$ to community C_i or outlier status $\emptyset$, default value undefined $\perp$

```
1  for s ∈ S : η(v) =⊥ do
2      create queue Q
3      if core_{κ,ε}(s) then
4          η(s) ← new community C
5          enqueue(Q, s)
6          coreSearch(Q,C)
7      else
8          if ∃c ∈ N_ε(s) : core_{κ,ε}(c) then
9              c ← core with minimum distance d(s, c)
10             η(s) ← new community C
11             enqueue(Q, c)
12             coreSearch(Q,C)
13         else
14             η(s) ← ∅
15         end
16     end
17 end
18 return ζ
```

Node Distance Measures: selSCAN is generic with respect to a node distance measure. The original SCAN is implemented with a node distance measure which expresses

Algorithm 2: coreSearch(Q,C)

```
while Q not empty do
    x ← dequeue node from Q
    for y ∈ N_ε(x) do
        if η(y) =⊥ then
            η(y) ← C
            if core_{κ,ε}(y) then
                enqueue(Q, y)
            end
        end
    end
end
```

the amount of overlap between node neighborhoods:

$$ND(u,v) = 1 - \frac{|\Gamma(u) \cap \Gamma(v)|}{\sqrt{|\Gamma(u)| \cdot |\Gamma(v)|}}$$

In addition to ND, we propose algebraic distance (AD) as concept for expressing the structural closeness of nodes. Algebraic distance on graphs was introduced as a way of measuring the connection strength between a pair of nodes, which are not necessarily neighbors [7]. The distances are calculated via an iterative algebraic method, conceptually related to random walks and diffusive processes. Each node is associated with a vector of r initially random values, where r is a small constant. Iterative smoothing is performed so that the vectors of nearby nodes assimilate. Algebraic node distance $AD(u,v)$ is then defined in terms of a norm on these vectors. $ND(u,v)$ can be calculated on the fly when discovering a community, but only takes the direct neighborhoods into account. $AD(u,v)$ requires a preprocessing phase, but can incorporate more structural information. Note that AD preprocessing can be implemented with a distributed algorithm because it involves only computations between neighboring nodes.

VI. Evaluation

We experimentally test the performance of GCE-M, GCE-L, selSCAN and PageRankNibble in terms of accuracy, quality and running time to evaluate whether they are appropriate for large complex networks. In extensive experiments, a subset of which is presented here, we used both synthetic LFR graphs and real-world social networks. Implementations are written in C++, extending our framework NetworKit [24], a collection of high-performance network analysis algorithms, available as free software (see http://networkit.iti.kit.edu). Experiments are performed on a compute server with 2 x 8 Cores: Intel(R) Xeon(R) E5-2680 0 at 2.70GHz, 32 threads and 256 GB RAM. The compiler is GCC 4.7.1 with -O3 optimization.

A. LFR Benchmark

A ground truth partition of a network is a partition which generates internal density and external sparsity of edges. Real-world complex networks are usually the result of multi-factorial processes generating nodes and edges, so a single ground truth partition does not necessarily exist. For synthetic graphs however, we can define a generative model in which the edge probability directly depends on a given partition. Such a model is the LFR graph generator, which we will use for evaluating the accuracy of our algorithms. Lancichinetti et. al [15] introduce a generative model intended as a benchmark for community detection, in which both node degrees and community sizes follow a power-law distribution. For our purposes, LFR offers a sufficiently realistic model for the complex networks we encounter in real data. For the evaluation, we vary the LFR mixing parameter μ, which stands for the fraction of inter-community edges. For higher mixing parameters, communities are less dense and their boundaries are less clearly defined, increasing the difficulty of the task. Distinctive communities can be identified up to $\mu = 0.5$. We quantify the accuracy of an algorithm in recognizing the ground truth in the following way: Let C_s be the detected community for seed s and T_s its ground truth community. We use the Jaccard index $J(C,T) := \frac{|C \cap T|}{|C \cup T|}$ to quantify their agreement. Jaccard values closer to 1 are better, and containment of T within C is not enough, because the Jaccard index penalizes the size of the set $C \setminus T$.

B. Parameter Studies

Both PageRankNibble and selSCAN depend on numeric parameters, the choice of which is crucial for community detection success. For PageRankNibble, we need to set α and β: α is the loop probability of the random walk, and smaller values tend to produce larger communities. β is the tolerance threshold for approximation of PageRank vectors, and smaller β leads to more accurate approximation and higher running time. For selSCAN, the parameters to estimate are ϵ, the threshold distance for two neighboring nodes to be considered close, and κ, the number of close neighbors required to be a core node. When using algebraic distances, we need to additionally estimate the number of iterations required for meaningful node distances, because values are increasingly smoothened with each iteration. The distance threshold ϵ needs to be selected depending on the resulting absolute distance values. In general, parameter dependence is a disadvantage of those methods, since community detection is usually an exploratory data analysis task and reliable ground truth against which to tune the parameters is not available. However, the LFR generator provides reliable ground truth, and we perform the following parameter study: For an LFR graph, we measure which combination of two parameters yields the best agreement with ground truth. By setting parameters this way, the algorithms are configured to recognize community structure that it similar to that found in the LFR benchmark graphs: Community sizes between 50 and 250 nodes are typical for actual social networks. From the results of the parameter study it became clear that we can fix κ at 2 and select an appropriate ϵ. Using the ND distance function, ground truth communities were best recognized for $\epsilon = 0.75$. For algebraic distances, we observed that 10 iterations and a distance threshold of $\epsilon = 0.01$ performed best. For PageRankNibble an approximation tolerance of 10^{-4} is sufficient and a loop probability $\alpha = 0.1$

performs well. We apply the same parameters to the real world networks in Section VI-E.

C. LFR Benchmark Results

Fig. 1 contains plots of LFR benchmark results. We let each algorithm find communities for 100 seed nodes in a 10^5 node LFR graph and record average accuracy, average community quality in terms of conductance, average community size and query time for the batch of seed nodes. Running times do not include the preprocessing phase of selSCAN-AD, which we discuss in more detail in Sec. VI-E. Additional experiments show that the results do not qualitatively change when we scale up the number of nodes in the graphs. The crucial factor is the size of the ground truth communities (here between 50 and 250 nodes) and the ratio of intra- to inter-community density controlled by the parameter μ. We increase the difficulty for algorithms by increasing μ up to 0.5. Algorithms tend to perform worse for increasing μ, the LFR parameter influencing the amount of inter-community edges. We consider those algorithms superior which can distinguish communities even with significant noise. GCE-L terminates without discovering the ground truth community, reflected in small community sizes. GCE variants optimizing L run slightly faster than those using M. In terms of accuracy, M clearly outperforms L. L leads to a low agreement with LFR ground truth. Inspecting precision and recall shows that on average about 80% of nodes are correctly assigned by GCE-L, but C_s is only insufficiently expanded to half the size of T_s. It seems to be a general limitation of L that big communities are not discovered because expansion halts after few iterations. This can be explained by the fact that the denominator L_{ext} grows faster than the numerator L_{int} during expansion. The same occurs for the non-greedy CE-L algorithm. In view of these results, we cannot consider the L measure an appropriate objective function for our purposes. GCE-L running times increase with μ because they are coupled to the growth of the shell. The same is not true for the density-based selSCAN-ND algorithm, whose running time even decreases as the size of discovered communities decreases. Qualitatively, selSCAN-ND performs well and behaves robustly with increasing noise between the communities, even outperforming PageRank-Nibble. selSCAN-AD starts as one of the fastest algorithms on the LFR graphs. It avoids the (re)computation of quality gains and can also return a result in constant time if the seed belongs to a previously discovered SCAN community. Query time is slightly faster than for selSCAN-ND, because node distances are calculated as pre-processing (see Sec. VI-E for discussion). It is important to note, however, that the high accuracy of selSCAN depends on the right choice for the parameter ϵ (see Sec. VI-B). Clearly, accuracy of selSCAN-AD peaks for the μ value on which the parameters have been trained, and then plummets. At the same time the size of the detected community escalates to encompass most of the network. These results indicate that selSCAN performs well only for correctly selected parameters, with a narrow margin of error. PageRank-Nibble needs parameter tuning as well,

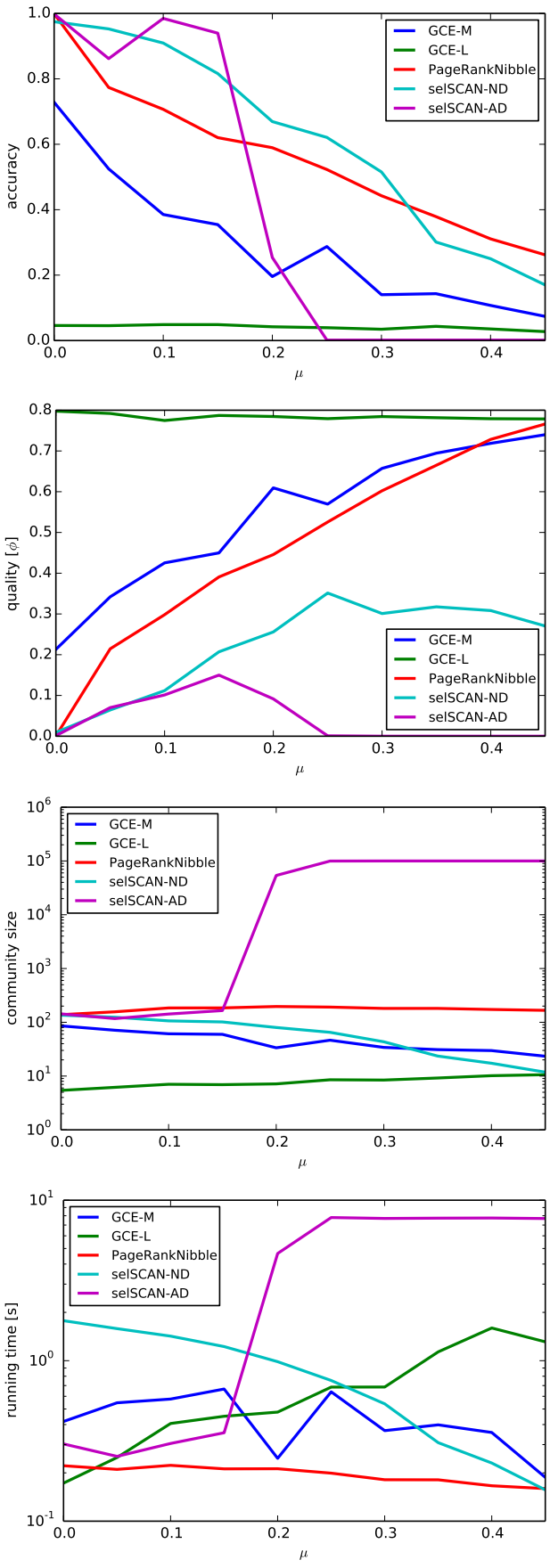

Fig. 1. LFR benchmark results: accuracy, quality, community size and running time (query time without preprocessing)

but is more robust and convinces with very low query times and a relatively high accuracy throughout the experiments.

D. Real-world Social Networks.

no	name	n	m	c	γ
0	Caltech36	769	16656	0.429	4.94
1	Smith60	2970	97133	0.291	10.44
2	Johns Hopkins55	5180	186586	0.276	4.37
3	UChicago30	6591	208103	0.261	3.78
4	Carnegie49	6637	249967	0.283	5.33
5	MIT8	6440	251252	0.279	3.84
6	Princeton12	6596	293320	0.240	4.87
7	Yale4	8578	405450	0.240	4.49
8	Harvard1	15126	824617	0.225	3.13
9	Oklahoma97	17425	892528	0.233	7.40

TABLE I
OVERVIEW OF REAL-WORLD SOCIAL NETWORKS USED

We compare the algorithms on a collection of real social networks collected in the early days of Facebook [26] (available from https://archive.org/details/oxford-2005-facebook-matrix). Table I contains graph sizes, an estimate of the average local clustering coefficient c, and the exponent γ of the degree distribution power law.

E. Results for Real-World Networks

For experiments on real-world networks, we select GCE-M, GCE-L, selSCAN-ND , selSCAN-AD and PageRank-Nibble as candidates. The results shown in Fig. 2 are averages of community quality (Φ), community size and total running time for a seed set of 10 nodes. Running times do not include AD preprocessing time. Because of the lack of reliable ground truth data, we cannot test accuracy. Quality as measured by conductance must be considered in combination with community sizes. We observed that good values of Φ can also be achieved for communities which encompass large parts of the graph, which are not likely to be desired solutions.

Strikingly, query times for GCE-M and GCE-L escalate completely on these real networks, being orders of magnitudes larger than the query times of selSCAN-AD and PageRank-Nibble, which are close to zero on this scale. We conjecture that this is due to high-degree nodes leading to an extreme growth of the size of the shell to be scanned. The resulting query times can only be called impractical. Query time for selSCAN-ND is also very high, for similar reasons as the overlap of large neighbourhoods has to be calculated. selSCAN-AD has potentially the fastest query times due to the precalculation of node distances, so that actual queries amount to little more than breadth-first search. While the preprocessing time is not insignificant, it can amortize if repeated queries on the same network need to be performed. We observe that AD preprocessing time is linear in the number of edges, with $t_{AD}(G) \approx 0.0022 \cdot |E|$ seconds on our machine. In practice, however, both selSCAN-ND and selSCAN-AD fail with the parameters trained on LFR graphs: selSCAN-ND produces giant communities while selSCAN-AD recognizes only outliers. This shows how sensitive the density-based approach is to the choice of numeric parameters, especially

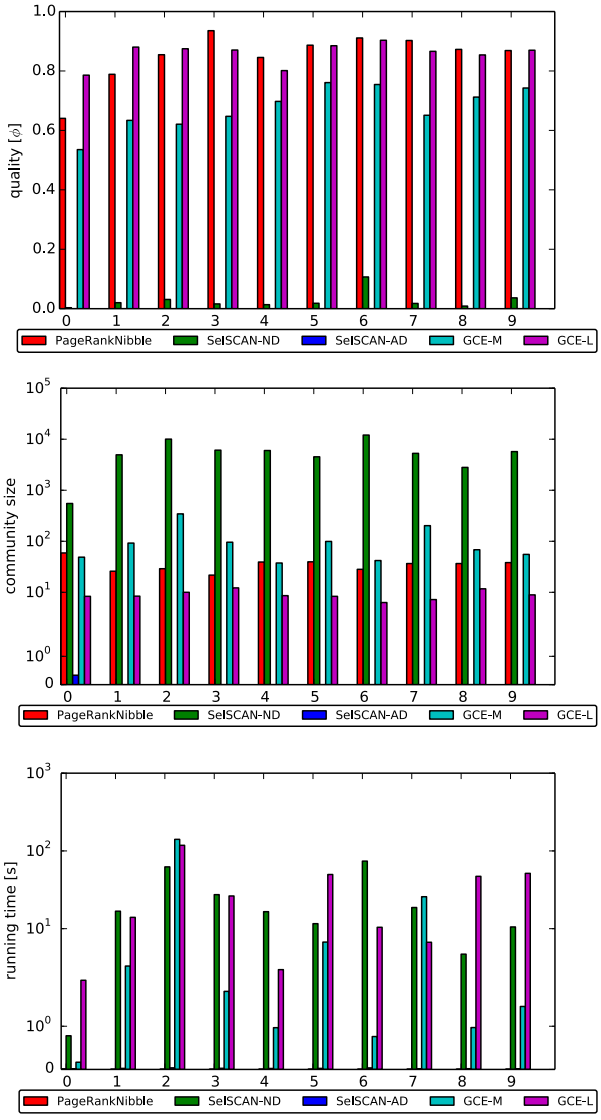

Fig. 2. Average community quality, size and running time for real complex networks (Table I)

the distance threshold ϵ. Values that achieved high accuracy on the LFR set fail completely for this set of real world social networks, although we can assume at least some similarity with respect to community structure and graph properties. In contrast, PageRank-Nibble performs rather well and in an expected way using the parameters estimated from LFR experiments.

VII. CONCLUSION

While various methods for selective community detection have been proposed, there is a gap in the literature with respect to an experimental comparison on real-world data. We contribute to the consolidation of the topic by reviewing several algorithms and objective functions proposed for the selective community detection problem. We highlight the need for scalable solutions, showing that a popular greedy approach is not fast enough to target large complex networks in the order of millions to billions of edges, which are not uncommon today. Greedy node-by-node optimization of community

quality may not be the best strategy at all, since the process can easily halt in unwanted local optima. We propose the density-based selSCAN as an alternative approach, which can be more accurate and faster especially when combined with algebraic distances. Although algebraic distances require preprocessing, they take more structural information into account and provide us with node distances appropriate for community detection. Calculating algebraic distances in a preprocessing step allows us to keep query times in the order of a few milliseconds even for graphs with millions of edges. The density-based approach is, however, very sensitive to the choice of a numeric parameter ϵ, which depends on the specific properties of a network. Solving the problem of parameter estimation without previous knowledge would allow us to harness the strengths of the density-based approach in practice. Finally our experiments show that the PageRank-Nibble algorithm is likely to be a practical method for selective community detection. Although the algorithm depends on numeric parameters that have to be estimated in parameter studies, these seem to be more robust and can be carried over to real-world networks. Our efficient implementation of the algorithm will be distributed with the publication as part of the NetworKit [24] framework.

Acknowledgements: This work was partially supported by the project *Parallel Analysis of Dynamic Networks — Algorithm Engineering of Efficient Combinatorial and Numerical Methods*, funded by the Ministry of Science, Research and the Arts Baden-Württemberg.

REFERENCES

[1] R. Andersen, F. Chung, and K. Lang. Local graph partitioning using pagerank vectors. In *FOCS '06: Proc. of the 47th Annual IEEE Symp. on Foundations of Computer Science*, pages 475–486, Los Alamitos, CA, USA, October 2006. IEEE Computer Society.

[2] R. Andersen and K. Lang. Communities from seed sets. In *Proc. of the 15th Int'l Conf. on World Wide Web*, page 232. ACM, 2006.

[3] J.P. Bagrow. Evaluating local community methods in networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008:P05001, 2008.

[4] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

[5] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *IEEE Trans. Knowledge and Data Engineering*, 20(2):172–188, 2008.

[6] L. Karl Branting. Incremental detection of local community structure. In *Proceedings of the 2010 International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '10, pages 80–87, Washington, DC, USA, 2010. IEEE Computer Society.

[7] Jie Chen and Ilya Safro. Algebraic distance on graphs. *SIAM J. Sci. Comput.*, 33(6):3468–3490, December 2011.

[8] Jiyang Chen, Osmar Zaïane, and Randy Goebel. Local community identification in social networks. In *Proceedings of the 2009 International Conference on Advances in Social Network Analysis and Mining*, ASONAM '09, pages 237–242. IEEE Computer Society, 2009.

[9] A. Clauset. Finding local community structure in networks. *Physical Review E*, 72(2):26132, 2005.

[10] A. Clauset, M.E.J. Newman, and C. Moore. Finding community structure in very large networks. *Physical Review E*, 70(6):66111, 2004.

[11] Luciano da Fontoura Costa, Osvaldo N Oliveira Jr, Gonzalo Travieso, Francisco Aparecido Rodrigues, Paulino Ribeiro Villas Boas, Lucas Antiqueira, Matheus Palhares Viana, and Luis Enrique Correa Rocha. Analyzing and modeling real-world phenomena with complex networks: a survey of applications. *Advances in Physics*, 60(3):329–412, 2011.

[12] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, volume 96, pages 226–231, 1996.

[13] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75 – 174, 2010.

[14] Santo Fortunato and Marc Barthelemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007.

[15] Andrea Lancichinetti and Santo Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Phys. Rev. E*, 80(1):016118, Jul 2009.

[16] F. Luo, J.Z. Wang, and E. Promislow. Exploring local community structures in large networks. *Web Intelligence and Agent Systems*, 6(4):387–400, 2008.

[17] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69(2):026113, Feb 2004.

[18] Symeon Papadopoulos, Andre Skusa, Athena Vakali, Yiannis Kompatsiaris, and Nadine Wagner. Bridge bounding: A local approach for efficient community discovery in complex networks. 2009.

[19] Jason Riedy, David A. Bader, Karl Jiang, Pushkar Pande, and Richa Sharma. Detecting communities from given seeds in social networks. Technical Report GT-CSE-11-01, Georgia Institute of Technology, February 2011. Expanded from submitted version.

[20] Satu Elisa Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

[21] Jiří Šíma and Satu Elisa Schaeffer. On the np-completeness of some graph cluster measures. In *SOFSEM 2006: Theory and Practice of Computer Science*, pages 530–537. Springer, 2006.

[22] Daniel A Spielman and Shang-Hua Teng. A local clustering algorithm for massive graphs and its application to nearly-linear time graph partitioning. *arXiv preprint arXiv:0809.3232*, 2008.

[23] Christian L. Staudt and Henning Meyerhenke. Engineering high-performance community detection heuristics for massive graphs. In *proceedings of the 2013 International Conference on Parallel Processing*. Conference Publishing Services (CPS), 2013.

[24] Christian L Staudt, Aleksejs Sazonovs, and Henning Meyerhenke. NetworKit: An interactive tool suite for high-performance network analysis. *arXiv preprint arXiv:1403.3005*, 2014.

[25] Heli Sun, Jianbin Huang, Jiawei Han, Hongbo Deng, Peixiang Zhao, and Boqin Feng. gskeletonclu: Density-based network clustering via structure-connected tree division or agglomeration. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 2010.

[26] Amanda L Traud, Peter J Mucha, and Mason A Porter. Social structure of facebook networks. *Physica A: Statistical Mechanics and its Applications*, 391(16):4165–4180, 2012.

[27] Konstantin Voevodski, Shang-Hua Teng, and Yu Xia. Finding local communities in protein networks. *BMC bioinformatics*, 10(1):297, 2009.

[28] Ying-Jun Wu, Han Huang, Zhi-Feng Hao, and Chen Feng. Local community detection using link similarity. *Journal of Computer Science and Technology*, 27(6), 2012.

[29] Bingying Xu, Zheng Liang, Yan Jia, Bin Zhou, and Yi Han. Local community detection using seeds expansion. In *Proceedings of the 2012 Second International Conference on Cloud and Green Computing*, CGC '12, pages 557–562, Washington, DC, USA, 2012. IEEE Computer Society.

[30] Jaewon Yang and Jure Leskovec. Defining and evaluating network communities based on ground-truth. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, page 3. ACM, 2012.