Exploiting Optimization for Local Graph Clustering

Kimon Fountoulakis* Xiang Cheng† Julian Shun‡ Farbod Roosta-Khorasani§
Michael. W. Mahoney¶

January 12, 2017

Abstract

Modern graph clustering applications require the analysis of large graphs which can be computationally expensive. In this regard, local spectral graph clustering methods aim to identify well-connected clusters around a given "seed set" of reference nodes without accessing the entire graph. This can be achieved, for example, by using the Approximate Personalized PageRank algorithm in the seminal paper by Andersen et al. [1]. Here, we adopt an optimization perspective on this problem and draw connections between the local spectral algorithm of [1] and an iterative shrinkage-thresholding algorithm. In particular, we show that, appropriately initialized ISTA can recover the sought-after local cluster in a time that only depends on the number of non-zeros of the optimal solution instead of the entire graph. In the process, we show that an optimization algorithm which apparently requires accessing the entire graph, can be made to behave completely local by accessing only a small number of nodes. This viewpoint builds a bridge across two seemingly disjoint fields of graph processing and numerical optimization, and allows one to leverage well-studied, numerically robust, and efficient optimization algorithms for processing today's large graphs.

1 Introduction

Modern graph clustering applications require the analysis of large graphs [13, 10] and in many cases the large size of recent graph data has rendered classical global approaches (i.e., those whose running time depends on the size of the entire graph) [9, 12, 17, 8, 3] computationally too expensive. This is so since the running time of these algorithms typically increases with the size of the input graph. This problem sparked the development of methods [1, 2, 18, 15, 11, 21] that are *local*, in that their running time depends only on the size of the output or on the size of an input seed set of reference nodes (and not the size of the full graph). This property yields local graph clustering methods which are more efficient for today's large-scale graphs. In addition to computational considerations, many real-world graphs tend to have "good" local and small/medium size clusters, as opposed to "good" large ones [13, 10], making the application of such local algorithms even more so appealing in practice[1]

Approximate Personalized PageRank (APPR) algorithm in the seminal paper [1] has been the cornerstone of most local graph clustering algorithms. However, most of these methods take a discrete and combinatorial perspective and, on the surface, might even appear "too complicated and unintuitive". Here, we focus on APPR and show that local graph clustering can be cast as a simple optimization problem. This viewpoint indeed decouples the combinatorial properties of the graph from the characteristics of the optimization algorithm used

*K. Fountoulakis is with the International Computer Science Institute, Department of Statistics, University of California Berkeley, 1947 Center St, Ste. 600, Berkeley, CA 94704, USA. e-mail: kfount@berkeley.edu.

†X. Cheng is with the Department of Electrical Engineering and Computer Science, University of California Berkeley, 253 Cory Hall, Berkeley, CA 94720-1770, USA. e-mail: x.cheng@berkeley.edu.

‡J. Shun is with the Department of Electrical Engineering and Computer Science, Department of Statistics, University of California Berkeley, 253 Cory Hall, Berkeley, CA 94720-1770, USA. e-mail: jshun@eecs.berkeley.edu.

§F. Roosta-Khorasani is with the International Computer Science Institute, Department of Statistics, University of California Berkeley, 1947 Center St, Ste. 600, Berkeley, CA 94704, USA. e-mail: farbod@icsi.berkeley.edu.

¶M. Mahoney is with the International Computer Science Institute, Department of Statistics, University of California Berkeley, Evans Hall, 2594 Hearst Ave., Berkeley, CA 94720, USA. e-mail: mmahoney@stat.berkeley.edu.

[1]In between global and local algorithms, there are a class of *locally-biased algorithms*, e.g., that of [14], whose running time depends on the whole graph, however the solution is locally-biased toward some input seed set of reference nodes. We don't consider them in this paper.

to solve the new formulation. More importantly, we will demonstrate that by using a popular optimization algorithm, namely iterative shrinkage-thresholding algorithms (ISTA), [19], and with proper initialization, one can indeed guarantee similar local properties as those of APPR. The high level objective of this work is to build a bridge between two seemingly disjoint fields of graph processing and numerical optimization. It is hoped that once this viewpoint is extended to other graph processing problems, faster and more efficient algorithms emerge as a result.

2 Preliminaries and Notation

We will denote the i'th coordinate of a vector p with $p(i)$. The vector of all ones is denoted by e and the vector whose i^{th} coordinate is one and zero elsewhere is denoted by e_i. The square root of a vector is taken component-wise, i.e., $q^{1/2} := [q_1^{1/2}, \ldots, q_n^{1/2}]$. We assume that we are given an undirected graph $\mathcal{G}$ with no self-loops, whose number of nodes and edges are denoted by n and m, respectively. The set of nodes of the graph is denoted by $\mathcal{V}$. By $j \sim i$ we mean that j is a neighbor of i and vice-versa. For a set of nodes S, the relation $j \sim S$ indicates that a node j is a neighbor of at least one node in S. Let A and D be the adjacency matrix and the diagonal degree matrix of $\mathcal{G}$, respectively. The i^{th} diagonal of D is denoted by d_i. We define $Q := D^{-1/2}\left\{D - \frac{1-\alpha}{2}(D+A)\right\}D^{-1/2}$ and

$$f(q) := \frac{1}{2}q^T Q q - \alpha s^T D^{-1/2} q. \tag{1}$$

Denoting $[n] = \{1, 2, \ldots, n\}$, let $S \subseteq [n]$ and $I_S \in \mathbb{R}^{n \times |S|}$ be a selection of columns of the identity matrix with indices in S. Further, we define $\nabla_S f(q) := I_S^T \nabla f(q)$, $d_S := diag(I_S^T D I_S)$ and $Q_S := I_S^T Q I_S$, where $diag()$ extracts the diagonal of the input matrix and returns it as a vector. One can easily see that the function ∇f is block $(1+\alpha)/2$-Lipschitz continuous and the function f is α-strongly-convex, both w.r.t. ℓ_2 norm. Finally, we define the support set of a vector as supp$(q) := \{i \in [n] \mid q_i \neq 0\}$.

3 Approximate Personalized Page-Rank in a Nutshell

Personalized PageRank (PPR) is a ubiquitous problem for ranking web pages, social and information network analysis, recommendation systems, analysis of biology, neuroscience and physics networks [6]. In this section, we give a brief background on PPR and its approximate variant (APPR) introduced in [1].

After arbitrarily ordering the nodes of $\mathcal{V}$, consider an input node, say i, and a vector $s \in \mathbb{R}^n$ such that $s_i = 1$ and zero elsewhere. For a lazy random walk matrix, $W = (I + AD^{-1})/2$, PPR is defined as the linear system

$$p = \alpha s + (1-\alpha)Wp, \tag{2}$$

where p is the sought-after solution vector. Starting from the zero vector, $p_0 = 0$, APPR sets out to solve (2) approximately until

$$\|D^{-1}r\|_\infty \leq \rho\alpha, \tag{3}$$

where

$$r := (I - (1-\alpha)W)p - \alpha s, \tag{4}$$

denotes the residual and $\rho \in (0,1)$ is a given tolerance. One of many contributions of [1] is giving explicit values for ρ which, in turn, guarantees that APPR will recover local clusters with some combinatorial properties; see Theorems 5 and 7 in [1]. APPR is indeed an iterative linear system solver applied to (2) where at each iteration, a properly chosen coordinate is updated as $p_{k+1} = p_k - r_k(i)e_i$. Any such coordinate can be chosen as long as $r_k(i) \geq \rho\alpha d_i$. Another contribution of [1] is a novel procedure for selecting such coordinates in $\mathcal{O}(1)$ time, as opposed to $\mathcal{O}(n)$, i.e., without having to access the entire nodes of the graph.

4 ℓ_1-Regularized Page-Rank and Local Graph Clustering

One can view APPR as a coordinate descent algorithm applied to for the optimization problem

$$\min f(q),$$

where f is defined as in (1). To see this, note that the residual r in (4) can be written as $r = D^{1/2}\nabla f(q)$ where $\nabla f(q) = D^{-1/2}\left\{D - \frac{1-\alpha}{2}(D+A)\right\}D^{-1/2}q - \alpha D^{-1/2}s$ and $q := D^{-1/2}p$. This way, APPR's termination criterion (3) can be alternatively written as

$$\|D^{-1/2}\nabla f(q_k)\|_\infty \leq \rho\alpha. \tag{5}$$

The following lemma, whose proof can be found in the supplementary materials, shows that APPR generates a sequence of vectors $\{q_k\}$ for which we have $\nabla f(q_k) \leq 0\ \forall k$.

Lemma 1. *If APPR is initialized with $q_0 = 0$ and $s \geq 0$, then $q_{k+1} \geq q_k$ and $\nabla f(q_k) \leq 0\ \forall k$.*

As an immediate consequence of Lemma 1, one can simplify the termination criterion of APPR as

$$\nabla_i f(q_k) \geq -\rho\alpha d_i^{1/2}\ \forall i. \tag{6}$$

Now consider ℓ_1-regularized variant of Personalized PageRank problem written as

$$\text{minimize} \quad \psi(q) := \rho\alpha\|D^{1/2}q\|_1 + f(q), \tag{7}$$

with the optimal solution denoted by q_*. It is easy to see that the first-order optimality conditions of (7) are

$$\nabla_i f(q_*) = \begin{cases} -\rho\alpha d_i^{1/2}, & \text{if } q_*(i) > 0 \\ \rho\alpha d_i^{1/2}, & \text{if } q_*(i) < 0 \\ \in \rho\alpha d_i^{1/2}[-1,1], & \text{if } q_*(i) = 0 \end{cases}.$$

In one of our main results, Theorem 1, we will show that $q_* \geq 0$, and hence, the optimality conditions of problem (7) can be simplified to

$$\nabla_i f(q_*) = \begin{cases} -\rho\alpha d_i^{1/2}, & \text{if } q_*(i) > 0 \\ \in \rho\alpha d_i^{1/2}[-1,0], & \text{if } q_*(i) = 0 \end{cases}. \tag{8}$$

Before presenting our main algorithm, we make a few remarks regarding the formulation (7) and its termination criteria (8).

- Notice that the optimality conditions (8) imply the termination criterion (6) of APPR, but the converse is not necessarily true. This is because (6) does not distinguish between positive and zero components of q_*. Moreover, depending on which coordinate is chosen at every iteration, APPR can yield a different output on multiple runs. In other words, the output solution depends completely on the setting of the algorithm. In contrast, ℓ_1-regularized PPR formulation (7) decouples the locality/sparsity of the solution from properties of the algorithm, i.e., which nodes are chosen at every iteration. More specifically, if there exists a good local cluster, then any optimization algorithm applied to ℓ_1-reg. PPR obtains the same solution, and the differences merely boil down to running time and locality as opposed to the actual output solution.
- The proposed optimization formulation (7) is motivated by Theorem 3 in [7]. However, by drawing a clear connection between the termination criterion of APPR, (6), and the first-order optimality conditions of ℓ_1-regularized PPR, (8), we get a much simpler formulation than the one presented in [7]. In particular, unlike the formulation of [7], problem (7) does not require any additional tuning parameters other than the ones used for APPR, nor does it introduce any constraints, such as non-negativity. More importantly, the formulation in [7] only implies the sparsity of the final solution as opposed to the intermediate iterates produced by any iterative procedure applied to solve the corresponding optimization problem. In sharp contrast, in Section 5, we will show that the application of properly initialized ISTA to our formulation (7) maintains sparsity for all generated iterates, a property which is crucial to obtaining a local algorithm.
- It has to be noted that, currently, the number of non-zeros/nodes in the optimal solution of (7) is a priori unknown. This is in contrast to APPR with a guaranteed output solution with $\mathcal{O}(1/(\rho\alpha))$ non-zeros; see Theorem 1 in [1]. Despite the very well known fact that ℓ_1 regularization promotes sparsity, [4, 20], it is difficult to have certain guarantees on the number of non-zeros unless one makes particular assumptions about the underlying model for the given graph. This will result in interesting statistical objectives, i.e., that of recovering assumed sparse models over the nodes of the graph [22]. Although, this is not the focus of our paper, we do provide numerical evidence in Section 6 showing that the sparsity pattern of the output of APPR and that of ISTA applied to (7) are indeed similar.

5 ISTA for ℓ_1-Regularized Page-Rank

In this section, we investigate the application of ISTA for approximately minimizing (7) and study its theoretical properties such as locality and running time. The adaptation of ISTA to our particular problem is depicted in Algorithm 1.

Recall that main computational advantage of APPR is the *locality* of its iterations. More specifically, throughout the iterations, APPR only updates the nonzero coordinates of the output solution vector. In other words, APPR never requires access to the entire graph and iterations are performed efficiently which makes the application of APPR very appealing for modern large graphs. Interestingly, we now show that Algorithm 1, which incorporates a presumably global optimization routine such as ISTA, exhibits this desired locality property while inheriting the fast convergence properties of ISTA.

Algorithm 1 ISTA-equivalent solver for (7)

1: **Initialize**: $\epsilon \in (0,1)$, $\alpha > 0$, $q_0 = 0$, s such that $e^T s = 1$ and $s \geq 0$, set $\nabla f(q_0) = -\alpha D^{-1/2} s$.

2: **while** $\|D^{-1/2}\nabla f(q_k)\|_\infty > (1+\epsilon)\rho\alpha$ **do**

3: Set $S_k := \{i \in [n] \mid q_k(i) - \frac{2}{1+\alpha}\nabla_i f(q_k) \geq \frac{2\rho\alpha}{1+\alpha} d_i^{1/2}\}$

4: $\Delta q_k := -\frac{2}{1+\alpha}(\nabla_{S_k} f(q_k) + \rho\alpha d_{S_k}^{1/2})$ and $q_{k+1}(S_k) = q_k(S_k) + \Delta q_k$

5: For each $i \in S_k$ set

$$\nabla_i f(q_{k+1}) = -\rho\alpha d_i^{1/2} - \frac{1-\alpha}{2d_i^{1/2}} \sum_{l \sim i, l \in S_k} \frac{A_{i,l}[I_{S_k}\Delta q_k]_l}{d_l^{1/2}}$$

6: For each $j \notin S_k$ such that $j \sim S_k$ set

$$\nabla_j f(q_{k+1}) = \nabla_j f(q_k) - \frac{1-\alpha}{2d_j^{1/2}} \sum_{l \sim j, l \in S_k} \frac{A_{j,l}[I_{S_k}\Delta q_k]_l}{d_l^{1/2}}$$

7: For each $j \notin S_k$ such that $j \nsim S_k$ set

$$\nabla_j f(q_{k+1}) = \nabla_j f(q_k)$$

8: $k = k + 1$

9: **end while**

10: **return** $p_k := D^{1/2} q_k$

Theorem 1 shows the equivalence between Algorithm 1 and ISTA, and more importantly, establishes the desired locality property. In particular, part ii of Theorem 1 states that if Algorithm 1 is initialized properly, then despite the fact that the set S_k changes at every iteration (Step 3 of Algorithm 1), its size, $|S_k|$, indeed never grows larger than the total number of non-zeros of the optimal solution. As such, in the worst case where one might update all the coordinates in S_k at every iteration, the per-iteration cost depends only on the sparsity of the final solution vector, as opposed to the size of the full graph.

Theorem 1. *Let q_* be the optimal solution of* (7) *and consider a vector $s \geq 0$ such that $e^T s = 1$ and $s_i \geq \rho$ for all i such that $s_i \neq 0$. The following are the properties of Algorithm 1:*

(i) Algorithm 1 is equivalent to ISTA in [5],

(ii) $|S_k| \leq |S_{k+1}| \leq |supp(q_*)|, \quad \forall k,$

(iii) $0 \leq q_k \leq q_{k+1}, \quad \forall k,$ *which implies that* $q_* \geq 0,$

(iv) $\nabla f(q_k) \leq 0,$ *and moreover* $\nabla_i f(q_k) \leq -\rho\alpha d_i^{1/2}\ \forall i \in S_k$ *and* $\nabla_i f(q_k) > -\rho\alpha d_i^{1/2}\ \forall i \in [n]\backslash S_k\ \forall k.$

Proof. Define

$$\tilde{f}(q; q_k) := f(q_k) + (q - q_k)^T \nabla f(q_k) + \frac{1+\alpha}{4}\|q - q_k\|_2^2,$$
$$\tilde{\psi}(q; q_k) := \rho\alpha\|D^{1/2} q\|_1 + \tilde{f}(q; q_k).$$

It is easy to see that

$$\arg\min_q \tilde{\psi}(q; q_k) = \arg\min_q \rho\alpha\|D^{1/2}q\|_1 + \frac{1}{2}\|q - (q_k - \frac{2}{1+\alpha}\nabla f(q_k))\|_2^2,$$

and hence

$$q(i) = \mathbf{prox}_{\rho\alpha d_i^{1/2}\|.\|_1}\left(q_k(i) - \frac{2}{1+\alpha}\nabla_i f(q_k)\right),$$

where **prox** is the proximal operator [16]. Now let us define the sets

$$\begin{aligned}
S_k &:= \{i \in [n] \mid q_k(i) - \frac{2}{1+\alpha}\nabla_i f(q_k) \geq \frac{2}{1+\alpha}\rho\alpha d_i^{1/2}\},\\
\widehat{S}_k &:= \{i \in [n] \mid -\frac{2}{1+\alpha}\rho\alpha d_i^{1/2} < q_k(i) - \frac{2}{1+\alpha}\nabla_i f(q_k) < \frac{2}{1+\alpha}\rho\alpha d_i^{1/2}\},\\
\widetilde{S}_k &:= \{i \in [n] \mid q_k(i) - \frac{2}{1+\alpha}\nabla_i f(q_k) \leq -\frac{2}{1+\alpha}\rho\alpha d_i^{1/2}\}.
\end{aligned}$$

For convenience we rewrite below ISTA from [5]. To show that Algorithm 1 and 2 are equivalent, it suffices to

Algorithm 2 ISTA for (7)

1: **Initialize**: $\rho > 0$, $q_0 = 0$, thus $\nabla f(q_0) = -\alpha D^{-1/2}s$
2: **while** termination criteria are not satisfied **do**
3: $q_{k+1}(i) = \mathbf{prox}_{\rho\alpha d_i^{1/2}\|.\|_1}\left(q_k(i) - \frac{2}{1+\alpha}\nabla_i f(q_k)\right)$, $\forall i$, whose closed-form solution is given by

$$q_{k+1}(i) = \begin{cases} q_k(i) - \frac{2}{1+\alpha}(\nabla_i f(q_k) + \rho\alpha d^{1/2}(i)) & \text{if } i \in S_k \\ q_k(i) - \frac{2}{1+\alpha}(\nabla_i f(q_k) - \rho\alpha d^{1/2}(i)) & \text{if } i \in \widetilde{S}_k \\ 0 & \text{if } i \in \widehat{S}_k. \end{cases}$$

4: Calculate new gradient $\nabla f(q_{k+1})$.
5: $k = k + 1$
6: **end while**
7: **return** $p_k := D^{1/2}q_k$

show that $\widetilde{S}_k = \emptyset, \forall k$. We will prove the result by induction. Let us assume that at iteration k we have a $q_k \geq 0$, $\nabla f(q_k) \leq 0$ and $\nabla_i f(q_k) \leq -\rho\alpha d_i^{1/2}$ $\forall i \in S_k$. As a result of the first two assumptions, we have $\widetilde{S}_k = \emptyset$ and $S_k \cap \widehat{S}_k = [n]$. Hence, Step 3 of ISTA Algorithm 2 can be simplified as

$$q_{k+1}(i) = \begin{cases} q_k(i) - \frac{2}{1+\alpha}(\nabla_i f(q_k) + \rho\alpha d^{1/2}(i)) & \text{if } i \in S_k \\ 0 & \text{if } i \in \widehat{S}_k \end{cases}. \tag{9}$$

Define $\Delta q_k := -\frac{2}{1+\alpha}I_{S_k}^T\left(\nabla f(q_k) + \rho\alpha D^{1/2}e\right)$, where I_{S_k} is defined in Section 2. Consequently, at iteration k, the new gradient components are updated as follows

$$\nabla_i f(q_{k+1}) = \begin{cases} -\rho\alpha d_i^{1/2} - \frac{1-\alpha}{2d_i^{1/2}} \sum\limits_{l\sim i, l\in S_k} \frac{A_{i,l}[I_{S_k}\Delta q_k]_l}{d_l^{1/2}}, & i \in S_k \\ \nabla_i f(q_k) - \frac{1-\alpha}{2d_i^{1/2}} \sum\limits_{l\sim i, l\in S_k} \frac{A_{i,l}[I_{S_k}\Delta q_k]_l}{d_l^{1/2}}, & i \in \widehat{S}_k \text{ and } i \sim S_k \\ \nabla_i f(q_k), & i \in \widehat{S}_k \text{ and } i \nsim S_k, \end{cases} \tag{10}$$

where A is the adjacency matrix of the given graph. By induction hypothesis and noticing that $\Delta q_k \geq 0$ and $A_{i,l} \geq 0, \forall i, l$, it is easy to see that by (9), we have $q_{k+1} \geq 0$, and by (10), we get $\nabla f(q_{k+1}) \leq 0$. Hence, it follows that $\widetilde{S}_{k+1} = \emptyset$. In addition, for any $i \in S_k$, we get $\nabla_i f(q_{k+1}) \leq -\rho\alpha d_i^{1/2}$ and, as such, $i \in S_{k+1}$. In other words, once an index i enters the set S_k at iteration k, it will continue to stay in that set for all subsequent iterations, and so we always have $q_{k+1}(i) \geq q_k(i)$. In fact, the only indices entering S_{k+1}

are those from $\widehat{S}_k$ which are neighbors of S_k (see (9) and (10) for $i \in \widehat{S}_k$) thus, $|S_k| \leq |S_{k+1}|$. In this case, suppose that $i \in \widehat{S}_k \cap S_{k+1}$. By (9), we have $q_{k+1}(i) = 0$, which combined with the definition of S_{k+1}, yields $\nabla_i f(q_{k+1}) \leq -\rho\alpha d_i^{1/2}$. As a result, we have $\nabla_i f(q_{k+1}) \leq -\rho\alpha d_i^{1/2}, \forall i \in S_{k+1}$. All is left to do is to start the iterations with the proper initial conditions, so that the base case of the induction holds. Set ρ small enough that $s_i \geq \rho \; \forall i$. Now since $s \geq 0$, by choosing $q_0 = 0$, we have that $\nabla f(q_0) = -\alpha D^{1/2} s \leq 0$ and $\nabla_i f(q_0) \leq -\rho\alpha d_i^{1/2} \; \forall i \in S_0$. In addition, such a choice of q_0, (9) as well as decreasing nature of $\widehat{S}_k$ imply that $q_{k+1} \geq q_k, \forall k$. Finally, since $q_{k+1} \geq q_k \; \forall k$ then Algorithm 1 will not update more than $|\text{supp}(q_*)|$ coordinates. Thus, we have that $|S_k| \leq |\text{supp}(q_*)| \; \forall k$. □

Theorem 2 gives the overall iteration complexity and total running time[2] of Algorithm 1. The proof is given in the supplementary material.

Theorem 2. *Let $s_i \geq \rho$ for all i such that $s_i \neq 0$. Algorithm 1 requires at most $T \in \mathcal{O}\left(\frac{1+\alpha}{2\alpha} \log\left(\frac{1+\alpha}{\epsilon^2 \rho\alpha^2 \min_j d_j}\right)\right)$ iterations to converge to a solution that satisfies the termination criterion at Step 2. Let $\mathcal{S}_* := supp(q_*)$, then the running time of Algorithm 1 is at most $\mathcal{O}\left(\frac{|\mathcal{S}_*|(1+\max_{j\in\mathcal{S}^*} d_j)(1+\alpha)}{\alpha} \log\left(\frac{1+\alpha}{\epsilon^2 \rho\alpha^2 \min_j d_j}\right)\right)$.*

According to Theorem 2, the worst-case running time of Algorithm 1 is $\mathcal{O}(|\mathcal{S}_*|/\alpha)$ (ignoring small terms). On the other hand, Theorems 1 and 5 in [1] state that APPR has a worst-case running time of $\mathcal{O}(1/(\rho\alpha))$. While we cannot prove anything about the relationship between$|\mathcal{S}_*|$ and $1/\rho$, in Section 6, we provide empirical evidence showing that the number of non-zeros of the output of Algorithm 1 is indeed similar to that of the output of APPR. Since APPR never accesses nodes of the graph that are not part of the output solution, we can conclude that both algorithms, in practice, have roughly the same running time; this is also illustrated in Section 6.

Finally, in order to ensure that the solutions of Algorithm 1 and APPR share the same theoretical clustering guarantees, the parameter ρ of Algorithm 1 must be set with respect to that of APPR. More specifically, let $\rho, \tilde{\rho} \in (0, 1)$ be the parameters of the ℓ_1-regularized PPR problem (7) and APPR, respectively. Moreover, let the vector $s \geq 0$ be chosen such that $s_i \geq \max(\rho, \tilde{\rho})$ for all i with $s_i \neq 0$, e.g., $s_i = 1$ for the reference node i and zero elsewhere. Then APPR algorithm at termination gives an output which satisfies (5) while Algorithm 1 is terminated when $\|D^{-1/2}\nabla f(q_k)\|_\infty \leq (1+\epsilon)\rho\alpha$. Hence, one can set $\rho \leq \tilde{\rho}/(1+\epsilon)$ to ensure that the termination criterion of Algorithm 1 matches that of APPR.

The theoretical clustering guarantees of Algorithm 1 is further discussed in the supplementary materials.

6 Experiments

This section describes experimental results for our implementations of the algorithms described in this paper.

The experiments are performed on a single thread of a 64-core machine with four 2.4 GHz 16-core AMD Opteron 6278 processors. The implementations are written using C++ code and compiled with the g++ compiler version 4.8.0. We use a set of undirected, unweighted real-world graphs from the Stanford Network Analysis Project (`http://snap.stanford.edu/data`), whose sizes are shown in Table 1. All implementations will be made publicly available. We present the performance of ISTA and two versions of APPR, greedy and heuristic. In

Input Graph	Num. Vertices	Num. Edges[†]
wiki-Talk	2,394,385	4,659,565
soc-LJ	4,847,571	42,851,237
cit-Patents	6,009,555	16,518,947
com-Orkut	3,072,627	117,185,083

Table 1: Graph inputs. [†]Number of unique undirected edges.

particular, for APPR GREEDY the i'th coordinate with the largest partial derivative $\nabla_i f(q_k)$ in absolute value is selected at each iteration. For APPR HEURISTIC a priority queue of coordinates is maintained which initially

[2]Iteration complexity refers to the worst-case number of iterations to satisfy the termination criterion and running time refers to the total amount of work, i.e., the per-iteration cost times iteration complexity.

contains the starting vertex only. On each iteration we select the highest-priority coordinate in the queue and update the coordinate and its neighbors accordingly. For each neighbor, insert it in the queue if its partial derivative is above the threshold with priority equal to the chosen coordinate. Note that this is a heuristic because we select coordinates based on their priority when they are initially inserted in the queue, and do not update their priorities later on. It is important to mention that the heuristic versions of the algorithms are guaranteed to converge in theory. The Cheeger-like guarantees on the output quality hold for the heuristic versions as well.

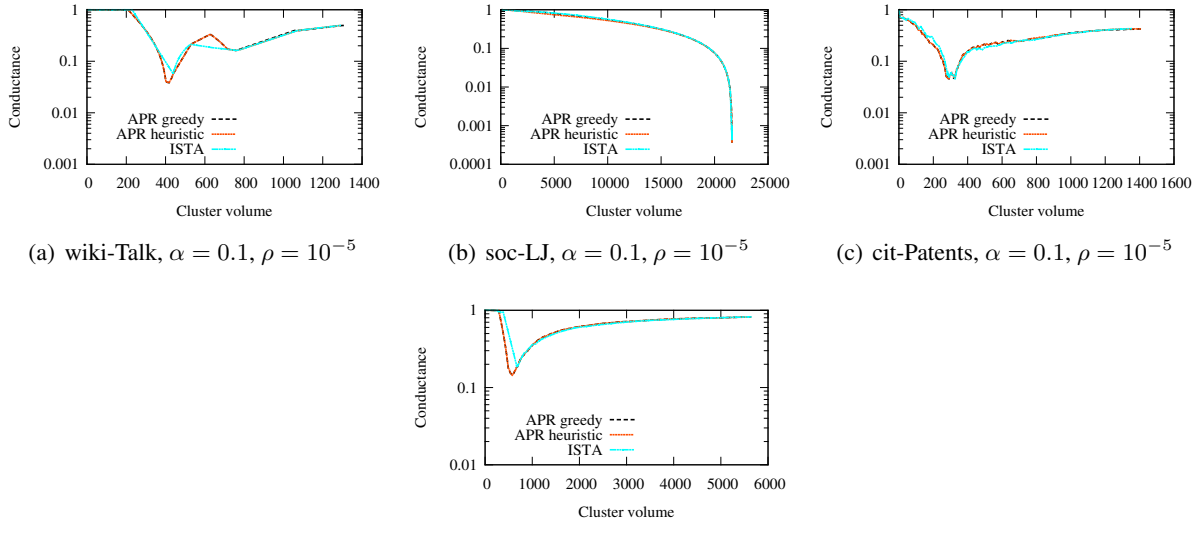
For all experiments we set $s = e_v$, where the coordinate/node v is chosen based on a search of over 10^4 starting nodes. We used the starting vertex that gave the best conductance. We conduct all experiments by fixing $\alpha = 0.1$ and choose the ρ values empirically such that we get clusters with at least 100 nodes each. This agrees with the observations in [13] regarding the size of local clusters in large-scale graphs.

6.1 ℓ_1-reg. PR $\approx$ APPR

We demonstrate that ℓ_1-reg. PR problem achieves in practice similar graph cut guarantees as APPR. We use the same sweep procedure as the one described in Section 2.2 in [1] for the original APPR algorithm, which is based on the conductance criterion. In Figure 1 we present the conductance criterion (y-axis) versus the volume ($\text{vol}(S) := \sum_{i \in S} d_i$) of the clusters ($x$-axis) produced by the sweep procedure in increasing order. All algorithms obtain approximately the same conductance value after the sweep procedure. The number of non-zeros of the output p_k for each algorithm is given in Table 2. Notice that the output of the ℓ_1-reg. PR problem, which is obtained by ISTA, has at most the same number of non-zeros as the greedy and the heuristic versions of APPR.

Input Graph	APPR GREEDY	APPR HEUR.	ISTA
wiki-Talk	326	334	326
soc-LJ	159	159	159
cit-Patents	210	211	198
com-Orkut	447	448	442

Table 2: Number of non-zeros for the output solution p_k of each algorithm for the four experiments in Figure 1.



(a) wiki-Talk, $\alpha = 0.1$, $\rho = 10^{-5}$

(b) soc-LJ, $\alpha = 0.1$, $\rho = 10^{-5}$

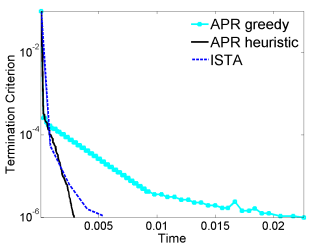
(c) cit-Patents, $\alpha = 0.1$, $\rho = 10^{-5}$

(d) com-Orkut, $\alpha = 0.1$, $\rho = 10^{-5}$

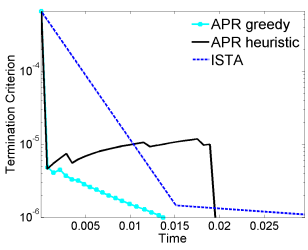
Figure 1: Conductance vs. cluster volume. The axes of all plots are in log-scale. This figure shows the conductance criterion for the clusters which are produced by the sweep procedure applied on the output of each algorithm. The volume of the clusters is shown in increasing size.

6.2 Running time

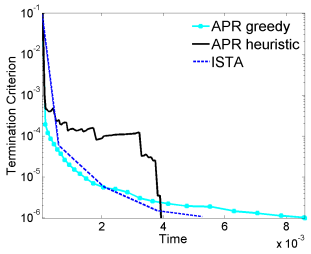
In Figure 2 we plot the value of the quantity $\|D^{-1/2}\nabla f(q_k)\|_\infty$ against the running time of each algorithm. The quantity $\|D^{-1/2}\nabla f(q_k)\|_\infty$ is used as a termination criterion for all algorithms. Notice that ISTA is nearly as efficient as APPR. Overall, the fastest algorithm is the variant of APPR where the next coordinates/nodes to be updated are chosen heuristically.



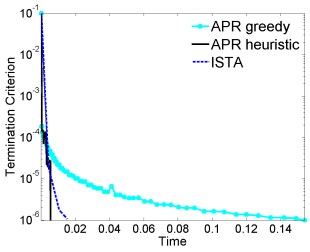
(a) wiki-Talk, $\alpha = 0.1$, $\rho = 10^{-5}$



(b) soc-LJ, $\alpha = 0.1$, $\rho = 10^{-5}$



(c) cit-Patents, $\alpha = 0.1$, $\rho = 10^{-5}$



(d) com-Orkut, $\alpha = 0.1$, $\rho = 10^{-5}$

Figure 2: Termination criterion, i.e., $\|D^{-1/2}\nabla f(q_k)\|_\infty$, vs. running time. The axes of all plots are in semi-log-vertical scale.

7 Conclusion

We discussed an optimization perspective of the local spectral clustering algorithm APPR [1]. We showed that the application of an existing state-of-the-art optimization algorithm, i.e., ISTA [19], can be made to result in a strongly local algorithm with a running time which only depends on the number of non-zeros of the solution (and not the entire graph). On a higher level, we hope that this optimization viewpoint builds a bridge across two seemingly disjoint fields of graph processing and numerical optimization, and allows one to leverage well-studied, numerically robust, and efficient optimization algorithms for processing today's large graphs. For example, one might be able to apply a modification of accelerated ISTA, i.e. FISTA [19], and improve the running time to $\mathcal{O}(1/\sqrt{\alpha})$ instead of the current $\mathcal{O}(1/\alpha)$. This can indeed be a direction for future research in this area.

References

[1] R. Andersen, F. Chung, and K. Lang. Local graph partitioning using pagerank vectors. *FOCS '06 Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 475–486, 2006.

[2] R. Andersen and K. Lang. An algorithm for improving graph partitions. *SODA '08 Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 651–660, 2008.

[3] S. Arora, S. Rao, and U. Vazirani. Expander flows, geometric embeddings and graph partitioning. *Journal of the ACM*, 56(2:5), 2009.

[4] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Optimization with sparsity-inducing penalties. *Found. Trends Mach. Learn.*, 4(1):1–106, 2011.

[5] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sciences*, pages 183–202, 2009.

[6] D. F. Gleich. Pagerank beyond the web. *SIAM Review*, 57(3):321–363, 2015.

[7] D. F. Gleich and M. W. Mahoney. Anti-differentiating approximation algorithms: A case study with min-cuts, spectral, and flow. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1018–1025, 2014.

[8] L. Grady and E. L. Schwartz. Isoperimetric partitioning:a new algorithm for graph partitioning. *SIAM Journal on Scientific Computing*, 27(6):1844–1866, 2006.

[9] K. M. Hall. An r-dimensional quadratic placement algorithm. *Management Science*, 17(3):219–229, 1970.

[10] L. G. S. Jeub, P. Balachandran, M. A. Porter, P. J. Mucha, and M. Mahoney. Think locally, act locally: The detection of small, medium-sized, and large communities in large networks. *Physical Review E*, 91:012821, 2015.

[11] K. Kloster and D. F. Gleich. Heat kernel based community detection. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1386–1395, 2014.

[12] T. Leighton and S. Rao. An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms. *Foundations of Computer Science, 1988., 29th Annual Symposium on*, pages 422–431, 1988.

[13] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet of Mathematics*, 6(1):29–123, 2011.

[14] M. Mahoney, L. Orecchia, and N. K. Vishnoi. A local spectral method for graphs: with applications to improving graph partitions and exploring data graphs locally. *Journal of Machine Learning Research*, 13:2339–2365, 2012.

[15] L. Orecchia and Z. A. Zhu. Flow-based algorithms for local graph clustering. *SODA '14 Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1267–1286, 2014.

[16] Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):123–231, 2013.

[17] A. Pothen, H. D. Simon, and K. P. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Applications*, 11(3):430–452, 1990.

[18] D. A. Spielman and S. H. Teng. A local clustering algorithm for massive graphs and its application to nearly linear time graph partitioning. *SIAM Journal on Scientific Computing*, 42(1):1–26, 2013.

[19] Suvrit Sra, Sebastian Nowozin, and Stephen J Wright. *Optimization for machine learning*. Mit Press, 2012.

[20] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[21] N. Veldt, D. F. Gleich, and M. W. Mahoney. A simple and strongly-local flow-based method for cut improvement. Accepted to ICML (2016).

[22] Y. X. Wang, J. Sharpnack, A. Smola, and R. Tibshirani. Trend filtering on graphs. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pages 1042–1050, 2014.

A Proof of Lemma 1

We rewrite the coordinate descent form of APPR in Algorithm 3 for convenience. We will prove this statement by induction. Let us assume that at the kth iteration we have $q_k \geq 0$ and $\nabla f(q_k) \leq 0$. Further, let assume that there exists coordinate i such that $\nabla_i f(q_k) < -\rho\alpha d_i^{1/2}$, otherwise, the termination criterion is satisfied. Algorithm 3 chooses one coordinate which satisfies $\nabla_i f(q_k) < -\rho\alpha d_i^{1/2}$. Then from Step 4 of Algorithm 3 we have that $q_{k+1} \geq q_k$. Moreover, from Steps 5, 6 and 7 we have that $\nabla_i f(q_k) < \nabla_i f(q_{k+1}) < 0$, $\nabla_j f(q_{k+1}) < \nabla_j f(q_k) \leq 0$ for each j such that $i \sim j$ and $\nabla_j f(q_{k+1}) = \nabla_j f(q_k) \leq 0$ for each j such that $i \nsim j$. Hence, $\nabla f(q_{k+1}) \leq 0$. Let $q_0 = 0$ and $s \geq 0$ in Algorithm 3. Then $\nabla f(q_0) = -\alpha s \leq 0$. We conclude that $q_{k+1} \geq q_k \geq 0$ and $\nabla f(q_k) \leq 0\ \forall k$.

B Iteration complexity and Running time

Let the assumption about s from Theorem 1 hold. Then from Theorem 1 we have that $q_k \geq 0\ \forall k$, i.e., we always remain in the the non-negative orthant. Denoting the restriction of $\psi(q)$ to $q \geq 0$, by

$$\widehat{\psi}(q) := \rho\alpha e^T D^{1/2} q + f(q),$$

it follows that $\psi(q) = \hat{\psi}(q)$ for all q in the non-negative orthant. From $(1+\alpha)/2$-Lipschitz continuity of ∇f w.r.t. ℓ_2 norm, it follows that $\widehat{\psi}$ is also smooth with the same parameter, i.e., $(1+\alpha)/2$. Hence, for any q_k from Algorithm 1, we have

$$\widehat{\psi}(q) \leq \psi(q_k) + (q - q_k)^T \nabla\widehat{\psi}(q_k) + \frac{1+\alpha}{4}\|q_k - q\|_2^2. \tag{11}$$

Since $q_{k+1} \geq 0$ (see Theorem 1), $q_{k+1} - q_k = I_{S_k}\Delta q_k$ and $\Delta q_k = -\frac{2}{1+\alpha}\nabla_{S_k}\widehat{\psi}(q_k)$ we have that

$$\psi(q_{k+1}) \leq \psi(q_k) - \frac{1}{1+\alpha}\|\nabla_{S_k}\widehat{\psi}(q_k)\|_2^2. \tag{12}$$

By α-strong convexity of ψ we have

$$\psi(q_k) - \psi(q_*) \leq \frac{1}{2\alpha}\|g\|_2^2 \quad \forall g \in \partial\psi(q_k),$$

where $\partial\psi(q_k)$ is the sub-differential of ψ at q_k. Notice that $I_{S_k}\nabla\widehat{\psi}_{S_k}(q_k)$ is a valid sub-gradient of ψ at q_k. This gives us

$$\psi(q_k) - \psi(q^*) \leq \frac{1}{2\alpha}\|\nabla_{S_k}\widehat{\psi}(q_k)\|_2^2. \tag{13}$$

Combining (12) and (13) and subtracting $\psi(q_*)$ from both sides we get

$$\psi(q_{k+1}) - \psi(q^*) \leq \left(1 - \frac{2\alpha}{1+\alpha}\right)(\psi(q_k) - \psi(q_*)),$$

which implies linear convergence. Applying the last inequality recursively we get that Algorithm 1 requires at most $T \in \mathcal{O}(\frac{1+\alpha}{2\alpha}\log(1/\hat{\epsilon}))$ iterations to obtain a solution q_T such that $\psi(q_T) - \psi(q_*) \leq \hat{\epsilon}$.

From (12) we have that

$$\psi(q_*) \leq \psi(q_k) - \frac{1}{1+\alpha}\|\nabla_{S_k}\widehat{\psi}(q_k)\|_2^2 \quad \forall k.$$

Using the above and $\psi(q_T) - \psi(q_*) \leq \hat{\epsilon}$, we get $\|\nabla_{S_k}\widehat{\psi}(q_T)\|_\infty^2 \leq (1+\alpha)\hat{\epsilon}$, which is equivalent to

$$-\rho\alpha - \left(\frac{(1+\alpha)\hat{\epsilon}}{d_i}\right)^{1/2} \leq \frac{\nabla_i f(q_T)}{d_i^{1/2}} \leq \rho\alpha + \left(\frac{(1+\alpha)\hat{\epsilon}}{d_i}\right)^{1/2}$$

$\forall i \in S_k$. From Theorem 1 we have that $\nabla_i f(q_T) > -\rho\alpha d_i^{1/2}\ \forall i \in [n]\backslash S_k$. Let $\epsilon \in (0,1)$ be the accuracy parameter of Algorithm 1. As a result, by setting $\hat{\epsilon} := (\epsilon^2\rho\alpha^2\min_j d_j)/(1+\alpha)$ and using the fact that $\nabla f(q_k) \leq 0\ \forall k$ from Lemma 1, we get that after

$$T \in \mathcal{O}\left(\frac{1+\alpha}{\alpha}\log\left(\frac{1+\alpha}{\epsilon^2\rho\alpha^2\min_j d_j}\right)\right) \tag{14}$$

iterations the output of Algorithm 1 satisfies $-(1+\epsilon)\rho\alpha d_i^{1/2} \leq \nabla_i f(q_T) \leq 0 \; \forall i$, which is the termination criterion in Step 2 of Algorithm 1.

Now we discuss the running time of Algorithm 1. Let $\mathcal{S}_* := \text{supp}(q_*)$. From Theorem 1 we have that $|S_k| \leq |\mathcal{S}_*|$. Hence, Step 4 of Algorithm 1 requires at most $\mathcal{O}(|\mathcal{S}_*|)$ operations. Similarly Steps 5 and 6 require at most $\mathcal{O}(|\mathcal{S}_*| + \text{vol}(\mathcal{S}^*))$ operations. This can be upper bounded by $\mathcal{O}(|\mathcal{S}_*|(1+\max_{i\in\mathcal{S}^*} d_i))$. Finally, Step 7 does not perform any computations. Putting the operations performed in all of the steps together and using the iteration complexity result in (14) we have that Algorithm 1 requires at most

$$\mathcal{O}\left(\frac{|\mathcal{S}_*|(1+\max_{j\in\mathcal{S}^*} d_j)(1+\alpha)}{\alpha}\log\left(\frac{1+\alpha}{\epsilon^2\rho\alpha^2\min_j d_j}\right)\right)$$

time.

C Theoretical Clustering guarantees

The main application of APPR in [1] is to approximately solve the NP-complete minimum-conductance problem. In this section we show that Algorithm 1 can be used instead of APPR to find low-conductance cuts.

Let w_{ij} be the weight of the edge between two neighbor nodes $i \sim j$. We define the conductance of a subset of nodes $S \subset \mathcal{V}$ as

$$\Phi(S) := \frac{\sum_{i\in S}\sum_{j\in\mathcal{V}\backslash S, j\sim i} w_{ij}}{\min\left(\text{vol}(S), \text{vol}(\mathcal{V}\backslash S)\right)},$$

where $\text{vol}(S) := \sum_{i\in S} d_i$. The minimum-conductance problem is defined as

$$\Phi(\mathcal{G}) := \min_{S\subset V} \Phi(S).$$

Let p_k be the output of APPR with input value α and let r_k be the residual of (2) calculated by APPR. Moreover, we define $C_\alpha \subseteq C$ such that $\text{vol}(C_\alpha) \geq \text{vol}(C)/2$. According to Theorem 5 in [1], we can use the output of APPR as an input to a sweep procedure (see Section 2.2 in [1]) to produce clusters of low-conductance. More precisely, the sweep procedure sorts the indices in $\text{supp}(p_k)$ in decreasing order w.r.t. to the values of the components of $D^{-1}p_k$. Let $i_1, i_2, \ldots, i_{|H_k|}$ be the sorted indices, where $H_k = \text{supp}(p_k)$. Using the sorted indices the sweep procedure generates a collection of sets $\mathcal{S}_j := \{i_1, i_2, \ldots, i_j\}$ for each $j \in \{1, 2, \ldots, |H_k|\}$. Provided that there exists a subset of nodes C that satisfies $\Phi(C) \leq \alpha/10$ and $\text{vol}(C) \leq 2\text{vol}(\mathcal{G})/3$, s is initialized within C_α and $\rho = 1/(10\text{vol}(C))$ then from Theorem 5 in [1] we get $\min_{j\in\{1,2,\ldots,|H_k|\}} \Phi(\mathcal{S}_j) \leq \sqrt{135\log(m)\alpha}$. Theorem 5 in [1] suggests setting $\alpha = 10\Phi(\mathcal{G})$ which gives us $\min_{j\in\{1,2,\ldots,|H_k|\}} \Phi(\mathcal{S}_j) \leq \sqrt{1350\Phi(\mathcal{G})\log(m)}$. Similar guarantees can be proved if we replace APPR with Algorithm 1.

Theorem 3. *Let $\rho = 1/(10vol(C))$, where C satisfies $\Phi(C) \leq \alpha/10$ and $vol(C) \leq 2vol(\mathcal{G})/3$, $\alpha = 10\Phi(\mathcal{G})$ and s is set within a subset of nodes $C_\alpha \subseteq C$. Algorithm 1 with $\rho' = \rho/(1+\epsilon)$ can be used instead of APPR to find a set with conductance at most $\sqrt{1350\Phi(\mathcal{G})\log(m)}$.*

The proof is based on the fact that Algorithm 1 satisfies the invariance property of APPR (see Section 3 in [1]). Moreover, all algorithms at termination satisfy $\|D^{-1/2}\nabla f(q_k)\|_\infty \leq \rho\alpha$. Below we provide a lemma that is used in proving Theorem 3.

Lemma 2. *Let $p := D^{1/2}q$, then the following is always true*

$$p + pr(\alpha, -\frac{1}{\alpha}D^{1/2}\nabla f(q)) = \alpha D^{1/2}QD^{-1/2}s.$$

In other words, $p + pr(\alpha, -\frac{1}{\alpha}D^{1/2}\nabla f(q))$ is a constant.

Proof of Lemma 2. Recall that $f(q) = \frac{1}{2}q^TQq - \alpha q^TD^{-1/2}s$, so $\nabla f(q) = Qq - \alpha D^{-1/2}s$, and $Q := D^{-1/2}\left\{D - \frac{1-\alpha}{2}(D+A)\right\}D^{-1/2}$. Let $pr(\alpha, s)$ be the solution to the Page-Rank system in (2), i.e., $pr(\alpha, s)$ satisfies the following

$$\frac{1-\alpha}{2}(I + AD^{-1})pr(\alpha, s) + \alpha s = pr(\alpha, s). \tag{15}$$

After some manipulations, one can obtain

$$pr(\alpha, s) = \alpha D^{1/2} Q^{-1} D^{-1/2} s. \tag{16}$$

Thus

$$\begin{aligned} p + pr(\alpha, -\frac{1}{\alpha} D^{1/2} \nabla f(q)) &= p + \alpha D^{1/2} Q^{-1} D^{-1/2} (-\frac{1}{\alpha} D^{1/2} \nabla f(q)) \\ &= p - D^{1/2} Q^{-1} \nabla f(q) \\ &= p - D^{1/2} Q^{-1} (Qq - \alpha D^{-1/2} s) \\ &= \alpha D^{1/2} Q^{-1} D^{-1/2} s. \end{aligned}$$

□

Proof of Theorem 3. The proof of Theorem 5 in [1] is based on two properties of APPR, which we discuss below, and the specific initialization of APPR that is given in the preamble of Theorem 3.

Property 1 This property is claimed in Section 3 in [1]. Let r be the residual of (2), i.e., $r := (I - (1 - \alpha)W)p - \alpha s$ and $pr(\alpha, s)$ be the solution to the Page-Rank system for some vector s. Let $p_k \; \forall k$ be generated by APPR and let us define $\tilde{r}_k := -(1/\alpha) r_k$. Then APPR satisfies the invariance property

$$p_{k+1} + pr(\alpha, \tilde{r}_{k+1}) = p_k + pr(\alpha, \tilde{r}_k) \quad \forall k. \tag{17}$$

Property 2 According to the termination criterion of APPR in (6) the residual r_k at termination satisfies

$$\|D^{-1} r_k\|_\infty \leq \rho\alpha. \tag{18}$$

Let us explain where these two properties are used in Theorem 5 in [1]. To prove Theorem 5 in [1] the authors used the results of Theorem 4 and Theorem 2 in [1]. Theorem 4 uses **Property 1** and **Property 2**. Theorem 2 makes the following calling sequence Theorem 2 ← Theorem 3 ← Lemma 5 ← Lemma 3, where the left arrow means "uses". The latter lemma uses **Property 1**. All other properties that are used to prove Theorem 5 are independent of APPR.

We will prove that Algorithm 1 satisfies **Property 1** and **Property 2**. Therefore, if they are initialized as mentioned in the preamble of Theorem 3 then they have the same graph cut guarantees as APPR.

We start by proving that Algorithm 1 satisfies the invariance property. Let $q_k \; \forall k$ obtained by Algorithm 1. Using $r_k = D^{1/2} \nabla f(q_k) \; \forall k$ and the definition $\tilde{r}_k = -(1/\alpha) r_k$ we get $-(1/\alpha) D^{1/2} \nabla f(q_k) = \tilde{r}_k$. Let us also define $p_k := D^{1/2} q_k \; \forall k$. Using Lemma 2 we get that Algorithm 1 satisfies

$$\begin{aligned} p_{k+1} + pr(\alpha, \tilde{r}_{k+1}) &= p_{k+1} + pr(\alpha, -(1/\alpha) D^{1/2} \nabla f(q_{k+1})) \\ &= \alpha D^{1/2} Q D^{-1/2} s \\ &= p_k + pr(\alpha, -(1/\alpha) D^{1/2} \nabla f(q_k)) \\ &= p_k + pr(\alpha, \tilde{r}_k)). \end{aligned}$$

Thus, **Property 1** (17) is satisfied for Algorithm 1

From the definition of the termination condition of Algorithm 1, we see that they return a q_k which satisfies $\|D^{-1/2} \nabla f(q_k)\|_\infty \leq (1 + \epsilon)\rho\alpha$, where $\epsilon \in (0, 1)$. Using $r_k = D^{1/2} \nabla f(q_k) \; \forall k$ we get that $\|D^{-1/2} \nabla f(q_k)\|_\infty \leq (1 + \epsilon)\rho\alpha$ is equivalent to $\|D^{-1} r_k\|_\infty \leq (1 + \epsilon)\rho\alpha$, which is the termination criterion (18) of APPR up to a factor $1 + \epsilon$. By initializing Algorithm 1 with $\rho\alpha = \rho'\alpha$ and $\rho' = \rho/1 + \epsilon$, we satisfy **Property 2** (18). Since Algorithm 1 satisfies both **Property 1** and **Property 2**, Theorem 5 in [1] holds for Algorithm 1 as well. □

Page-Rank-Nibble One issue with the result in Theorem 3 is that there is no lower bound on the volume of the output cluster which is obtained from the sweep procedure. This implies that a very small set might be found. In Section 6 in [1] a procedure called Page-Rank-Nibble is proposed which deals with this problem. Page-Rank-Nibble makes a single call to APPR. We show that Algorithm 1 can be used instead. We describe the

main result about Page-Rank-Nibble from Section 6 in [1], and we refer the reader to [1] for a more detailed discussion of the algorithm.

Let $\phi \in [0,1]$ be a parameter and let us assume that there exists $C \subset \mathcal{V}$ such that $\text{vol}(C) \leq \text{vol}(\mathcal{G})/2$ and $\Phi(C) \leq \phi^2/(22500\log^2(100m))$. Page-Rank-Nibble makes a single call to APPR and uses its output to produce the sweep sets, which were defined above. Theorem 7 in [1] suggests that if APPR is initialized with $\alpha = \phi^2/(225\log(100m^{1/2}))$ and s is set in C_α then there exists some $b \in [1, \lceil \log m \rceil]$ such that if $\rho \leq (2^b 48\lceil \log m \rceil)^{-1}$ then at least one sweep set $\mathcal{S}_j$ satisfies $\Phi(\mathcal{S}_j) \leq \phi$, $2^{b-1} < \text{vol}(\mathcal{S}_j) < 2\text{vol}(\mathcal{G})/3$ and $\text{vol}(\mathcal{S}_j \cap C) > 2^{b-2}$. Similarly to Theorem 3, by replacing APPR with Algorithm 1 we can obtain the same guarantees for Page-Rank-Nibble as in Theorem 7 in [1]. The proof is similar to that of Theorem 3 and we therefore omit it.

Theorem 4. *Let $\phi \in [0,1]$, $\alpha = \phi^2/(225\log(100m^{1/2}))$ and s is set within a subset of nodes $C_\alpha \subseteq C$, where C satisfies vol$(C) \leq$ vol$(\mathcal{G})/2$ and $\Phi(C) \leq \phi^2/(22500\log^2(100m))$. There exists $b \in [1, \lceil \log m \rceil]$ such that if $\rho \leq (2^b 48\lceil \log m \rceil)^{-1}$ then Algorithm 1 with $\rho' = \rho/(1+\epsilon)$ can be used in Page-Rank-Nibble to find a set S that satisfies $\Phi(S) \leq \phi$, $2^{b-1} <$ vol$(S) < 2$vol$(\mathcal{G})/3$ and vol$(S \cap C) > 2^{b-2}$.*

Algorithm 3 Coordinate descent version of APPR

Initialize: $\rho > 0$, $q_0 = 0$, thus $\nabla f(q_0) = -\alpha D^{-1/2} s$
while $\|D^{-1/2}\nabla f(q_k)\|_\infty > \rho\alpha$ **do**
  Choose an i such that $\nabla_i f(q_k) < -\alpha\rho d_i^{1/2}$
  $q_{k+1}(i) = q_k(i) - \nabla_i f(q_k)$
  $\nabla_i f(q_{k+1}) = \frac{1-\alpha}{2}\nabla_i f(q_k)$
  For each j such that $j \sim i$ set

$$\nabla_j f(q_{k+1}) = \nabla_j f(q_k) + \frac{(1-\alpha)}{2d_i^{1/2}d_j^{1/2}}\nabla_i f(q_k)$$

  For each j that $j \nsim i$ set $\nabla_j f(q_{k+1}) = \nabla_j f(q_k)$
  $k = k + 1$
end while
return $p_k := D^{1/2} q_k$