UNIVERSITY OF CALIFORNIA, SAN DIEGO

Local Algorithms for Graph Partitioning and Finding Dense Subgraphs

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Mathematics

by

Reid Andersen

Committee in charge:

Professor Fan Chung Graham, Chair
Professor Sam Buss
Professor T.C. Hu
Professor Russell Impagliazzo
Professor Jeff Remmel

2007

The dissertation of Reid Andersen is approved, and it is acceptable in quality and form for publication on microfilm:

_____

_____

_____

_____

_____

Chair

University of California, San Diego

2007

TABLE OF CONTENTS

LIST OF TABLES

ACKNOWLEDGEMENTS

Chapter 3 contains material that appeared in the following article.
R. Andersen, F. Chung, and K. Lang. *Local graph partitioning using PageRank vectors*. In *FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 475–486, Washington, DC, USA, 2006. IEEE Computer Society. The dissertation author was the primary investigator and author of this paper.

Chapter 4 contains material that appeared in the following article.
R. Andersen and F. Chung. *Detecting sharp drops in PageRank and a simplified local partitioning algorithm*. To appear in *The 4th Annual Conference on Theory and Applications of Models of Computation (TAMC'07)*. The dissertation author was the primary investigator and author of this paper.

Chapter 6 contains material to appear in the following article.
R. Andersen, F. Chung and K. Lang. *Local partitioning in directed graphs*. Manuscript in preparation, 2007. The dissertation author was the primary investigator and author of this paper.

Chapter 7 contains material to appear in the following article.
R. Andersen. *A local algorithm for finding dense subgraphs*, Manuscript in preparation, 2007.

VITA

2001	S.B., Massachussetts Institute of Technology
2002	M.A., University of California, San Diego
2007	Ph.D., University of California, San Diego

ABSTRACT OF THE DISSERTATION

Local Algorithms for Graph Partitioning and Finding Dense Subgraphs

by

Reid Andersen
Doctor of Philosophy in Mathematics
University of California San Diego, 2007
Professor Fan Chung Graham, Chair

This thesis is concerned with a new type of approximation algorithm for the fundamental problems of partitioning a graph and identifying its dense subgraphs. We develop *local approximation algorithms* for these two key problems, which search for a small set of vertices near a specified seed vertex, have a running time that is independent of the size of the graph, and for which we can prove a *local approximation guarantee*.

In the first part, we develop a local algorithm for graph partitioning that finds a cut with small conductance near a specified seed vertex, and has a running time that is nearly linear in the number of vertices in the small side of the cut. Within any cut S with conductance Φ, we prove there are a significant number of starting vertices for which the algorithm produces a cut with conductance $O(\sqrt{\Phi}\log n)$. By combining many small cuts found by this algorithm we can find a balanced cut quickly, producing in time $O(m\log^3 m/\Phi)$ a cut with conductance $O(\sqrt{\Phi}\log n)$ that is nearly as large as any cut with conductance Φ, where m is the number of edges in the graph. The local algorithm finds its cut by computing a variation of PageRank with a specified starting vertex, and ordering the vertices of the graph according to that ranking.

In the second part, we develop a local algorithm for finding dense subgraphs of bipartite graphs. This algorithm takes as input a bipartite graph with a specified seed vertex, and attempts to find a subgraph near the seed vertex that is dense

according to the definition proposed by Kannan and Vinay. For any subgraph S with k vertices and density θ, there are a significant number of starting vertices within S for which the algorithm produces a subgraph with density $\Omega(\theta/\log\Delta)$, where Δ is the maximum degree in the graph. This local algorithm finds a dense subgraph by computing a sequence of vectors, which are similar to the vectors produced by the power method for computing the largest eigenvector of A, but which are pruned at each step to keep their support small.

1 Introduction

One of the central problems in algorithmic design is the problem of partitioning a graph. This may be formulated as the task of finding a cut with small conductance, where the ratio between the number of edges crossing the cut and the volume of the smaller side of the cut is small. There is a large literature of research papers on this topic in theoretical computer science, including the breakthrough results of Leighton-Rao and Arora-Rao-Vazirani, which provide approximation algorithms for the problem. Graph partitioning algorithms have been applied in numerous fields, including parallel computing, VLSI design, and image processing.

This thesis is concerned with a relatively new type of approximation algorithm for this fundamental problem. Our main contribution is to develop an improved *local approximation algorithm* that can find a cut with small conductance at a computational cost that depends on the size of output of the algorithm rather than the size of the graph. This algorithm searches for a set of vertices with a specified target size and small conductance near a specified seed vertex. We prove a *local approximation guarantee* for this algorithm; for any set with small conductance, there are many seed vertices for which the algorithm is guaranteed to produce a set with conductance nearly as small as the original, provided the target size is large enough. The task our algorithm performs may be described as local partitioning, or targeted community identification.

Local partitioning may be applied to find a small community of nodes near an interesting target vertex, a task for which the local approximation guarantee is directly relevant. Because the running time of a local partitioning algorithm is independent of the size of the graph, it is well-suited to identifying the numerous small communities of vertices that may be found in large graphs derived from

modern datasets. Local partitioning may also be applied as a subroutine to quickly produce approximate solutions of more complicated partitioning problems. In particular, the small sets found by local partitioning can be combined to form a larger set with small conductance and nearly optimal balance in nearly linear time.

This thesis contains two main results. We develop an improved algorithm for local partitioning, continuing a line of work initiated by Spielman and Teng. We also develop a novel local approximation algorithm for the problem of identifying dense subgraphs, an important task in the analysis of large networks.

Our improved local partitioning algorithm is based on new techniques for computing and analyzing personalized PageRank vectors (which will be described in more detail in Section 2). These include an algorithm for approximating a personalized PageRank vector by another PageRank vector with a different starting distribution, based on a technique introduced by Jeh and Widom [30], and a mixing bound for PageRank vectors that is analogous to the mixing bound for random walks proved by Lovász and Simonovits [40, 41].

1.1 History

Approximation algorithms for the minimum-conductance cut problem. Consider the problem of finding a cut where the ratio between the number of edges crossing the cut and the size of the smaller side of the cut is small. There are several variations of this basic problem, including the sparsest cut problem, normalized cut problem, ratio cut problem, and minimum-conductance cut problem.

Spectral partitioning methods find low-conductance cuts by sorting the vertices of the graph according to their entries in an eigenvector, and produce cuts with conductance $O(\sqrt{\Phi})$ when there exists a cut with conductance Φ. Finding a cut from the Laplacian matrix of a graph was first proposed by Fiedler [20], and independently by Donath and Hoffman [15, 16]. The discrete analogue of Cheeger's inequality, which proves the approximation guarantee for spectral partitioning, was first proved by Alon and Milman [1] and Jerrum and Sinclair [31]. Mihail gave a

combinatorial proof that shows a cut with small conductance can be obtained by sweeping over any vector with a small Rayleigh quotient [42]. Cheeger's inequality for the normalized Laplacian, which applies to graphs that are not necessarily regular, was proved by Fan Chung [14].

The algorithm of Leighton and Rao, which is based on multicommodity flow, finds a cut with conductance $O(\Phi \log n)$ in a graph with conductance Φ. The algorithm of Arora, Rao, and Vazirani [7], which is based on semidefinite programming, produces a cut with conductance $O(\Phi\sqrt{\log n})$, the best approximation ratio known for the problem. An algorithm with the same approximation ratio and faster running time was given by Arora, Hazan, and Kale [4]. Recent results on metric embeddings [6, 38] have led to improved approximation algorithms for other partitioning problems, including finding vertex separators [17] and finding sparse cuts in directed graphs [13].

Partitioning algorithms that find a single two-way cut can be applied recursively to solve more complicated problems, including finding balanced cuts, k-way partitions, and hierarchical clusterings [10, 32, 39, 45, 46]. The running time of these recursive algorithms can be large if the cuts found at each step are unbalanced. This is particularly evident when applying spectral partitioning recursively, since spectral partitioning is guaranteed to produce a cut with approximately optimal conductance, but provides no guarantee on the balance of the cut. It was observed by Lang [37] that in many large graphs, the best partitions in terms of conductance are obtained by separating a small set of vertices from rest of the graph.

Local partitioning. Spielman and Teng developed the first local partitioning algorithm, and used it to address the problem of slow recursive partitioning. They introduced a local partitioning algorithm called `Nibble` that finds a small cut near a specified starting vertex in time proportional to the size of the small side of the cut. The analysis of the `Nibble` algorithm is based on a result of Lovász and Simonovits [40, 41] that shows a cut with small conductance can be found by simulating a random walk starting from a single vertex for sufficiently many steps. The small cuts found by `Nibble` can be combined to form a cut with

low conductance and nearly optimal balance in nearly linear time. In particular, `Nibble` was used to derive an algorithm `Partition` that produces a cut with conductance $O((\Phi \log^2 m)^{1/3})$, and with volume at least half as large as the largest cut with conductance Φ, in time $m(\log m/\Phi)^{O(1)}$, where m is the number of edges in the graph. The algorithm `Partition` was then used as a subroutine for finding multiway partitions, sparsifying graphs, and solving diagonally dominant linear systems [47].

Balanced cuts. When Spielman and Teng introduced `Partition`, it was the only algorithm known that could produce a cut with small conductance and nearly optimal balance in time $o(n^2)$. There are now two algorithms that solve the closely related problem of finding a b-balanced separator in time $o(n^2)$ using methods quite different from local partitioning. The algorithm of Khandekar, Rao, and Vazirani [34] produces a cut with conductance $O(\Phi_b \log^2 n)$ and balance at least $b/2$, where Φ_b is the smallest conductance of any cut with balance at least b. Their algorithm performs $O(\log^4 n)$ single commodity flow computations, and runs in time $\tilde{O}(min(n^{1.5}, n/\Phi))$. This was improved by the recent algorithm of Arora and Kale [5], which produces a cut with conductance $O(\Phi_b \log n)$ and balance at least $b/2$ in time $\tilde{O}(m + n^{1.5})$.

Community identification. The problem of identifying communities of nodes in large graphs like the world wide web is a challenging problem that can be approached using graph partitioning techniques. The community identification algorithm of Flake, Lawrence, and Giles [22], which identifies communities of pages in the web graph, may be viewed as a heuristic for local partitioning. This algorithm begins with a seed set of vertices, and expands this seed set into a community by examining a small portion of the entire web. The algorithm expands the seed set in several stages, at each stage augmenting the current set with vertices with high indegree to the current set, and then expanding to the minimum cut separating the current set from the rest of the graph. This algorithm performs well in practice, and is efficient enough to run on large datasets, but is not known to produce an approximation guarantee.

1.2 Results

Local partitioning. In Chapter 3, we give a local partitioning algorithm with the current best asymptotic running time and approximation ratio, improving upon the local partitioning algorithm of Spielman and Teng [47]. Our local partitioning algorithm finds cuts by computing and examining PageRank vectors. A PageRank vector is a weighted sum of the probability distributions obtained by taking a sequence of random walk steps starting from a specified initial distribution. The weight placed on the distribution obtained after t walk steps decreases exponentially in t, with the rate of decay determined by a parameter called the *teleport probability*. A PageRank vector can also be viewed as the solution of a system of linear equations, which we will describe in more detail in Chapter 2. Each of the PageRank vectors we compute has its starting distribution on a single starting vertex, and we prove that a PageRank vector produces a cut which approximates the best cut near its starting vertex. This cut can be found by performing a *sweep* over the PageRank vector, which involves examining the vertices of the graph in an order determined by the PageRank vector, and computing the conductance of each set produced by this order. The analysis of our algorithm is based on the following results:

- We give an algorithm for approximating a PageRank vector by another PageRank vector with a slightly different starting distribution, based on a technique introduced by Jeh and Widom [30]. This allows us to compute a PageRank vector with teleport probability α, and with an amount of error sufficiently small for finding a cut with volume k, in time $O(k/\alpha)$. (The volume of a set S is defined to be the sum of degrees over all vertices in S, and the volume of a cut is the minimum of the volumes of the two sides.)

- We prove a mixing result for PageRank vectors, which shows that if a PageRank vector with teleport probability α has significantly more probability than the stationary distribution on some set of vertices with volume k, a sweep over that PageRank vector will produce a cut with conductance $O(\sqrt{\alpha \log k})$.

- We show that for any set C, and for many vertices v contained in C, a PageRank vector whose starting vertex is v, and whose teleport probability is greater than the conductance of C, has a significant fraction of its probability contained in C.

Combining these results yields a partitioning result for PageRank vectors: for any set C with conductance Φ, there are a significant number of starting vertices within C for which a sweep over an appropriate PageRank vector finds a cut with conductance $O(\sqrt{\Phi \log k})$, where k is the volume of C. Such a cut can be found in time $O(k/\Phi + k \log k)$.

To produce cuts with approximately optimal balance in nearly linear time, we must be able to remove a small piece of the graph in time proportional to the volume of that small piece, rather than the volume of C or the volume of the entire graph. We present an algorithm `PageRank-Nibble` that does this. For many starting vertices within a set C of conductance Φ, this algorithm finds a cut with conductance $O\left((\Phi \log^2 m)^{1/2}\right)$ and volume at least k in time $O(k/\Phi)$, provided we can guess the volume of the smaller side of the cut within a factor of 2. This improves the algorithm `Nibble`, which finds a cut with conductance $O((\Phi \log^2 m)^{1/3})$ in time $O(k \log^{2/3} m/\Phi^{5/3})$. We can combine cuts found by `PageRank-Nibble` into a cut whose conductance is $O\left((\Phi \log^2 m)^{1/2}\right)$ and whose volume is at least half that of any set with conductance at most Φ, in time $O(m \log^2 m/\Phi)$. This improves the algorithm `Partition`, which obtains a cut whose conductance is $O\left(\Phi \log^2 m)^{1/3}\right)$ and whose volume is at least half that of any set with conductance at most Φ, in time $O(m \log^{8/3} m/\Phi^{5/3})$.

Related partitioning results. In chapter 4, we give a simpler construction of a nearly linear time local partitioning algorithm. We show that whenever there is a sharp drop in the numerical rank defined by a personalized PageRank vector, the location of the drop reveals a cut with small conductance. We then show that for any cut in the graph, and for many starting vertices within that cut, an approximate personalized PageRank vector will have a sharp drop sufficient to produce a cut with conductance nearly as small as the original cut.

In chapter 5, we give an intuitive proof of Cheeger's inequality, which is the result that shows a sweep over an eigenvector produces an approximation of the minimum conductance cut in the graph. Using techniques developed in earlier chapters, we provide an interesting characterization of a set with conductance $O(\sqrt{\Phi})$.

In chapter 6, we generalize portions of our local partitioning results to strongly connected directed graphs, and describe one possible approach for applying the result to directed graphs that are not strongly connected.

Local algorithm for finding dense subgraphs. In Chapter 7, we describe a local approximation algorithm for a variation of the densest subgraph problem. Kannan and Vinay [33] introduced a notion of density that is well-suited to large bipartite graphs derived from incidence matrices. Given a weighted bipartite graph with adjacency matrix A and a fixed bipartition L and R, the density of a pair of sets $S \subseteq L$ and $T \subseteq R$ is defined to be

$$d(S,T) = \frac{\langle 1_S A, 1_T \rangle}{\sqrt{|S|}\sqrt{|T|}}.$$

There is a spectral approximation algorithm for the corresponding version of the densest subgraph problem; Kannan and Vinay showed that by examining the largest eigenvector of A, one can produce a pair of sets (S', T') such that $d(S', T') = \Omega(\theta/\log n)$ where θ is the maximum density of any pair.

Our algorithm takes as input a graph with a specified starting vertex, and attempts to find a pair of sets near that vertex with high density. We prove that for any pair of sets (S, T) that has density θ and satisfies $|S| + |T| \leq k$, there are a significant number of starting vertices within S and T for which the algorithm produces a pair of sets (S', T') that has density $\Omega(\theta/\log \Delta)$ and that contains $O(\Delta k^2)$ vertices, where Δ is the maximum degree in the graph. The running time of the algorithm is $O(\Delta k^2)$, independent of the number of vertices in the graph. The analysis of the algorithm involves examining the eigenvalues and eigenvectors of certain submatrices of the adjacency matrix A.

There are several applications for this local algorithm. We can find a dense subgraph near a vertex of interest, while examining only a portion of the entire graph. We can also find many dense subgraphs in parallel, which is not possible

with existing algorithms. The algorithm provides an upper bound on the size of the sets it produces, which may make it a useful tool for the problem of maximizing $d(S,T)$ over all pairs (S,T) on exactly k vertices, an analogue of the densest k-subgraph problem.

2 Preliminaries

In the first few chapters, we consider graphs that are undirected and unweighted. Let $G = (V, E)$ be a graph with $|V| = n$ and $|E| = m$. We write $d(v)$ for the degree of a vertex v.

The adjacency matrix $A = A(G)$ is the $n \times n$ matrix where $A_{i,j} = 1$ if and only if there is an edge between v_i and v_j, given some fixed ordering $v_1, \ldots, v_n$ of the vertices. The degree matrix $D = D(G)$ is the $n \times n$ diagonal matrix where $D_{i,i} = d(v_i)$.

We view a vector p over the vertex set V as a $1 \times n$ row vector, so the product of p with a matrix A is written pA. Two vectors we use frequently are the degree-normalized uniform distribution over a set S,

$$\psi_S(x) = \begin{cases} \frac{d(x)}{\mathrm{vol}(S)} & \text{if } x \in S, \\ 0 & \text{otherwise,} \end{cases}$$

and the indicator function,

$$1_v(x) = \begin{cases} 1 & \text{if } x = v, \\ 0 & \text{otherwise.} \end{cases}$$

The *support* $\mathrm{Supp}(p)$ of a vector p is the set of vertices where p is nonzero. The sum of the vector p over a set S of vertices is written as follows,

$$p(S) = \sum_{u \in S} p(u).$$

If the entries of p are positive and $p(V)$ is at most 1, we refer to $p(S)$ as the amount of probability from p on S.

2.1 Cuts and conductance

The *volume* of a subset $S \subseteq V$ of vertices is defined to be

$$\mathrm{vol}(S) = \sum_{x \in S} d(x).$$

The volume of the entire graph is $\mathrm{vol}(V) = 2m$, where $m = |E|$ is the number of undirected edges in the graph.

We define $e(S, T)$ to be the set of edges between two sets of vertices S and T. A special case is the *edge boundary* of a set, which is defined to be

$$\partial(S) = e(S, \bar{S}) = \{\{x, y\} \in E \mid x \in S, y \notin S\}.$$

The quantity $|\partial(S)|$ is the *cutsize* of the cut $(S, \bar{S})$.

The *conductance* of a set is

$$\Phi(S) = \frac{|\partial(S)|}{\min\left(\mathrm{vol}(S), 2m - \mathrm{vol}(S)\right)}.$$

The conductance of the graph, Φ_G, is the minimum value $\Phi(S)$ of any subset $S \subseteq V$.

2.2 Lovász and Simonovits' technique for analyzing random walks

As part of their work in estimating the volume of convex bodies, Lovász and Simonovits introduced a novel technique for bounding the distance between the distribution of a lazy random walk after t steps and the stationary distribution of the walk, in terms of the conductance of certain sets [40, 41]. We will describe a portion of their results here in a general form that we will reuse later.

The random walk matrix W is defined to be

$$W = D^{-1}A.$$

To apply a random walk step to a vector p, we multiply by the matrix W to obtain a new vector $p' = pW$. If p is a probability distribution, then p' is the distribution

obtained by sampling a vertex from p, and then choosing a neighbor of that vertex at random.

To consider how the distribution of probability changes when a random walk step W is applied to an arbitrary vector p, we view each undirected edge $\{u, v\}$ as a pair of oriented edges (u, v) and (v, u). For each oriented edge (u, v), define

$$p(u, v) = \frac{p(u)}{d(u)},$$

which is the amount of probability that moves from u to v when the random walk step is applied to p. For any set of oriented edges A, we define

$$p(A) = \sum_{(u,v)\in A} p(u, v).$$

For any set S of vertices, we define the set of oriented edges whose heads are in S,

$$\text{in}(S) = \{(u, v) \in E \mid v \in S\},$$

and the set of oriented edges whose tails are in S,

$$\text{out}(S) = \{(u, v) \in E \mid u \in S\}.$$

We remark that an edge between two vertices of S is contained in both $\text{in}(S)$ and $\text{out}(S)$. The following lemma describes the amount of probability from pW on a set S in terms of the amount of probability moving across the edges in the sets $\text{in}(S)$ and $\text{out}(S)$.

Lemma 2.1. *For any vector p, and any set S of vertices,*

$$pW(S) = p\left(\text{in}(S) \cap \text{out}(S)\right) + p\left(\text{in}(S) \cup \text{out}(S)\right) - p(S).$$

Proof of Lemma 2.1. The amount of probability from pM on a set S can be written as follows.

$$\begin{aligned} pM(S) &= p(\text{in}(S)) \\ &= p\left(\text{in}(S) \cap \text{out}(S)\right) + p\left(\text{in}(S) \setminus \text{out}(S)\right) \\ &= p\left(\text{in}(S) \cap \text{out}(S)\right) + p\left(\text{in}(S) \cup \text{out}(S)\right) - p(S). \end{aligned}$$

□

Definition 2.2. Given a vector p on the vertex set V of a graph, let $v_1, \ldots, v_n$ be an ordering of the vertices such that

$$\frac{p(v_i)}{d(v_i)} \geq \frac{p(v_{i+1})}{d(v_{i+1})}.$$

For each integer j in $\{1, \ldots, n\}$, we define $S_j^p = \{v_1, \ldots, v_j\}$ to be the set containing the top j vertices in this ordering.

We call the sets S_j^p the *sweep sets* of the vector p. We define $\Phi(p)$ to be the smallest conductance of any sweep set,

$$\Phi(p) = \min_{j \in [1,n]} \Phi(S_j^p).$$

We perform a *sweep* over p by sorting the vertices by $p(v)/d(v)$, computing the conductance of each of the $O(n)$ sweep sets, and choosing a sweep set that attains the minimum conductance $\Phi(p)$. This can be done in time $O(m + n \log n)$.

To measure how a vector p is distributed in the graph, Lovász and Simonovits define the following function $p\,[x]$, which is determined by the volume of and the amount of probability on the sweep sets.

Definition 2.3. Let p be a vector over the vertex set V of a graph. We define $p\,[x]$ be the unique function from $[0, 2m]$ to $[0, 1]$ such that

$$p\left[\mathrm{vol}(S_j^p)\right] = p(S_j^p) \quad \text{for each } j \in [0, n],$$

and such that $p\,[x]$ is piecewise linear between these points. For any point $x \in [0, 2m]$, if j is the unique index where x is between $\mathrm{vol}(S_j^p)$ and $\mathrm{vol}(S_{j+1}^p)$, then

$$p\,[x] = p(S_j^p) + \frac{x - \mathrm{vol}(S_j^p)}{d(v_{j+1})} p(v_{j+1}).$$

Proposition 2.4. *We have the following facts about the function $p\,[x]$.*

1. *The function $p\,[x]$ is concave.*

2. *For any set S of vertices,*

$$p(S) \leq p\,[\mathrm{vol}(S)]\,.$$

3. For any set of oriented edges A, we have

$$p(A) \leq p\left[\mathrm{vol}(A)\right].$$

Proof. These facts are proved in [40], and are not difficult to verify. □

We now show how this function can be used to give an upper bound on the number of steps required for a lazy random walk to become close to its stationary distribution. The lazy random walk transition matrix is defined to be

$$M = \frac{1}{2}I + \frac{1}{2}D^{-1}A.$$

Given an initial probability distribution p_0, we define the lazy walk distribution after t steps by the rule $p_t = p_{t-1}M$. This random walk is called lazy because only half of the probability moves at each step, and the other half remains where it is. In a bipartite graph, the usual random walk W may alternate between the two sides of a bipartition, and thus may not converge to a stationary distribution. In contrast, the lazy random walk converges to its unique stationary distribution ψ_V in any connected undirected graph. The following theorem shows that the difference between the distribution of a walk after t steps and its stationary distribution decreases by a significant amount at each step.

Theorem 2.5 (Lovász-Simonovits)**.** *Let p be some vector, let $q = pM$ be the result of applying a lazy walk step to p, and let $\phi = \Phi(q)$ be the smallest conductance found by a sweep over q. If $C \geq 0$ is a constant such that*

$$p\left[x\right] \leq x/2m + C\sqrt{\min(x, 2m - x)} \qquad \textit{for all } x \in [0, 2m],$$

then

$$q\left[x\right] \leq x/2m + \left(1 - \frac{\phi^2}{8}\right) C\sqrt{\min(x, 2m - x)} \qquad \textit{for all } x \in [0, 2m].$$

Proof. Applying Lemma 2.1, we obtain

$$\begin{aligned} q(S) &= \frac{1}{2}p(S) + \frac{1}{2}pW(S) \\ &= \frac{1}{2}p(S) + \frac{1}{2}\left(p\left(\text{in}(S) \cap \text{out}(S)\right) + p\left(\text{in}(S) \cup \text{out}(S)\right) - p(S)\right) \\ &= \frac{1}{2}\left(p\left(\text{in}(S) \cap \text{out}(S)\right) + p\left(\text{in}(S) \cup \text{out}(S)\right)\right). \end{aligned}$$

Recall from Proposition 2.4 that $q\left[\text{vol}(S_j^q)\right] = q(S_j^q)$ for any integer $j \in [0, n]$, and that for any set of oriented edges A, we have the bound $p(A) \leq p\left[|A|\right]$.

$$\begin{aligned} q\left[\text{vol}(S_j^q)\right] &= q(\text{vol}(S_j^q) \\ &= \frac{1}{2}p\left(\text{in}(S_j^q) \cap \text{out}(S_j^q)\right) + \frac{1}{2}p\left(\text{in}(S_j^q) \cup \text{out}(S_j^q)\right) \\ &\leq \frac{1}{2}p\left[|\text{in}(S_j^q) \cap \text{out}(S_j^q)|\right] + \frac{1}{2}p\left[|\text{in}(S_j^q) \cup \text{out}(S_j^q)|\right] \\ &\leq \frac{1}{2}p\left[\text{vol}(S_j^q) - |\partial(S_j^p)|\right] + \frac{1}{2}p\left[\text{vol}(S_j^q) + |\partial(S_j^p)|\right]. \end{aligned}$$

The last line follows from Proposition 2.6, which we will state immediately after this proof.

There are two cases to consider, depending on whether $\text{vol}(S_j^q) \leq m$ or $\text{vol}(S_j^q) \geq m$. We will first carry out the proof assuming that $\text{vol}(S_j^q) \leq m$, in which case we have $|\partial(S_j^q)| = \Phi(S_j^q)\text{vol}(S_j^q)$.

$$\begin{aligned} q\left[\text{vol}(S_j^q)\right] &\leq \frac{1}{2}p\left[\text{vol}(S_j^q) - \Phi(S_j^q)\text{vol}(S_j^q)\right] + \frac{1}{2}p\left[\text{vol}(S_j^q) + \Phi(S_j^q)\text{vol}(S_j^q)\right] \\ &\leq \frac{1}{2}p\left[\text{vol}(S_j^q) - \phi\text{vol}(S_j^q)\right] + \frac{1}{2}p\left[\text{vol}(S_j^q) + \phi\text{vol}(S_j^q)\right] \\ &\leq \frac{1}{2}(\text{vol}(S_j^q) - \phi\text{vol}(S_j^q))/2m + \frac{1}{2}(\text{vol}(S_j^q) + \phi\text{vol}(S_j^q))/2m \\ &\quad + \frac{1}{2}C\sqrt{\text{vol}(S_j^q) - \phi\text{vol}(S_j^q)} + \frac{1}{2}C\sqrt{\text{vol}(S_j^q) + \phi\text{vol}(S_j^q)} \\ &\leq \text{vol}(S_j^q)/2m + C\sqrt{\text{vol}(S_j^q)}\left(\frac{1}{2}\sqrt{1-\phi} + \frac{1}{2}\sqrt{1+\phi}\right) \\ &\leq \text{vol}(S_j^q)/2m + \left(1 - \frac{\phi^2}{8}\right)C\sqrt{\text{vol}(S_j^q)}. \end{aligned}$$

The second line above follows from the concavity of $p\left[x\right]$. The last step follows from Proposition 2.7, which we will state immediately after this proof.

In the case where $\mathrm{vol}(S_j^q) \geq m$, we have $|\partial(S_j^q)| = \Phi(S_j^q)(2m - \mathrm{vol}(S_j^q))$, and a similar argument shows

$$q\left[\mathrm{vol}(S_j^q)\right] \leq \mathrm{vol}(S_j^q)/2m + \left(1 - \frac{\phi^2}{8}\right) C\sqrt{2m - \mathrm{vol}(S_j^q)}.$$

Then, we have that for every $j \in [0, n]$,

$$q\left[\mathrm{vol}(S_j^q)\right] \leq \mathrm{vol}(S_j^q)/2m + \left(1 - \frac{\phi^2}{8}\right) C\sqrt{\min(\mathrm{vol}(S_j^q), 2m - \mathrm{vol}(S_j^q))}.$$

Because the function $\sqrt{\min(x, 2m - x)}$ is concave and $p[x]$ is linear between the points $\mathrm{vol}(S_j)$ and $\mathrm{vol}(S_{j+1})$, it follows that for every $x \in [0, 2m]$, we have

$$q[x] \leq x/2m + \left(1 - \frac{\phi^2}{8}\right) C\sqrt{\min(x, 2m - x)}.$$

□

Proposition 2.6. *For any set S of vertices,*

$$|\mathrm{in}(S) \cup \mathrm{out}(S)| = \mathrm{vol}(S) + |\partial(S)|,$$

and

$$|\mathrm{in}(S) \cap \mathrm{out}(S)| = \mathrm{vol}(S) - |\partial(S)|.$$

Proof. This is straightforward. □

Proposition 2.7. *For any $\phi \in [0, 1]$,*

$$\frac{1}{2}\left(\sqrt{1 - \phi} + \sqrt{1 + \phi}\right) \leq 1 - \frac{\phi^2}{8}.$$

Proof. We examine the first few terms of the power series for $\sqrt{1 + \phi}$ at $\phi = 0$,

$$\sqrt{1 + \phi} = 1 + \frac{1}{2}\phi - \frac{1}{8}\phi^2 + \frac{1}{16}\phi^3 + \ldots,$$
$$\sqrt{1 - \phi} = 1 - \frac{1}{2}\phi - \frac{1}{8}\phi^2 - \frac{1}{16}\phi^3 - \ldots.$$

The odd terms of the two series cancel, and the even terms are negative. The result follows. □

2.3 PageRank and personalized PageRank

PageRank was introduced by Brin and Page [11, 43] as a way to evaluate the relative importance of web pages.

The PageRank vector $\mathrm{pr}(\alpha, s)$ is defined to be the unique solution of the linear system

$$\mathrm{pr}(\alpha, s) = \alpha s + (1-\alpha)\mathrm{pr}(\alpha, s)W. \tag{2.1}$$

Here, α is a constant in $(0, 1]$ called the *teleport probability*, s is a vector called the *starting vector*, and W is the random walk transition matrix $W = D^{-1}A$.

The PageRank vector that is usually associated with search ranking has a starting vector equal to the uniform distribution $\frac{1}{n}1_V$. PageRank vectors whose starting vectors are concentrated on a smaller set of vertices are often called *personalized PageRank vectors*. These were introduced by Haveliwala [29], and have been used to provide personalized search ranking and context-sensitive search [8, 23, 30].

We are particularly interested in PageRank vectors whose starting vectors are equal to the indicator function 1_v for a single vertex v. The vertex v will be called the *starting vertex*, and we will use the notation $\mathrm{pr}(\alpha, v) = \mathrm{pr}(\alpha, 1_v)$. We will also allow PageRank vectors where the starting vector s has both positive and negative entries. We define the positive part of s as follows,

$$s_+(x) \quad = \quad \begin{cases} s(x) & \text{if } s(x) \geq 0, \\ 0 & \text{otherwise.} \end{cases}$$

Here are some useful properties of PageRank vectors (also see [29] and [30]).

Proposition 2.8. *For any starting vector s, and any constant α in $(0, 1]$, there is a unique vector $\mathrm{pr}(\alpha, s)$ satisfying $\mathrm{pr}(\alpha, s) = \alpha s + (1-\alpha)\mathrm{pr}(\alpha, s)W$.*

Proof of Proposition 2.8. The solutions to the equation $p = \alpha s + (1-\alpha)pW$ are exactly the solutions of the linear system $p\,(I - (1-\alpha)W) = \alpha s$. When $\alpha \in (0, 1]$, the matrix $(I - (1-\alpha)W)$ is nonsingular since it is strictly diagonally dominant, so this linear system has a unique solution for each vector s. □

Proposition 2.9. *For any fixed value of* α *in* $(0, 1]$*, there is a linear transformation* R_α *such that* $\mathrm{pr}(\alpha, s) = sR_\alpha$*. Furthermore,* R_α *is given by the matrix*

$$R_\alpha = \alpha I + \alpha \sum_{t=1}^{\infty} (1-\alpha)^t W^t. \tag{2.2}$$

This implies that a PageRank vector is a weighted average of random walk vectors,

$$\mathrm{pr}(\alpha, s) = \alpha s + \alpha \sum_{t=1}^{\infty} (1-\alpha)^t \left(sW^t\right). \tag{2.3}$$

This also implies that a PageRank vector $\mathrm{pr}(\alpha, s)$ *is linear in its starting vector* s*.*

Proof of Proposition 2.9. The sum that defines R_α in equation (2.2) is absolutely convergent for $\alpha \in (0, 1]$, and the following computation shows that sR_α obeys the steady state equation for $\mathrm{pr}(\alpha, s)$.

$$\begin{aligned}
&\alpha s + (1-\alpha) s R_\alpha W \\
&= \alpha s + (1-\alpha) s \left(\alpha I + \alpha \sum_{t=1}^{\infty} (1-\alpha)^t W^t \right) W \\
&= \alpha s + s \left(\alpha \sum_{t=1}^{\infty} (1-\alpha)^t W^t \right) \\
&= s R_\alpha.
\end{aligned}$$

Since the solution to the this equation is unique by Proposition 2.8, it follows that $\mathrm{pr}(\alpha, s) = sR_\alpha$. □

2.4 Notation

As an example of the notation we will use throughout this document, if we let $p = \mathrm{pr}(\alpha, v)$, then the amount of probability from p on a set S will be written $p(S)$ or $[\mathrm{pr}(\alpha, v)](S)$, and the value of the Lovász-Simonovits curve at the point $x = \mathrm{vol}(S)$ will be written with the square bracket notation $p[\mathrm{vol}(S)]$.

3 Local Partitioning Using Personalized PageRank

In this chapter, we present a local graph partitioning algorithm that finds cuts by computing and examining personalized PageRank vectors. We derive a mixing result for PageRank vectors similar to that for random walks, and show that the ordering of the vertices produced by a PageRank vector reveals a cut with small conductance. In particular, we show that for any set C with conductance Φ and volume k, a PageRank vector whose starting vertex is one of a large number of good starting vertices within C can be used to produce a set with conductance $O(\sqrt{\Phi \log k})$. We present an improved algorithm for computing approximate PageRank vectors, which allows us to find such a set in time proportional to its size.

3.1 Mixing bounds for personalized PageRank

In this section, we prove a mixing result for PageRank vectors that is an analogue of the Lovász-Simonovits mixing result for random walks. For any PageRank vector $\mathrm{pr}(\alpha, s)$, we give an upper bound on $\mathrm{pr}(\alpha, s)\,[x]$ that depends on the smallest conductance $\Phi(\mathrm{pr}(\alpha, s))$ found by performing a sweep over $\mathrm{pr}(\alpha, s)$. We use this mixing result to prove the following theorem, which shows that if there exists a set of vertices S that contains a constant amount more probability from $\mathrm{pr}(\alpha, s)$ than from the stationary distribution ψ_V, then a sweep over $\mathrm{pr}(\alpha, s)$ finds a cut with conductance $O(\sqrt{\alpha \log(\mathrm{vol}(S))})$.

Theorem 3.1. *If* $\mathrm{pr}(\alpha, s)$ *is a PageRank vector with* $\|s_+\|_1 \leq 1$*, and there exists a set* S *of vertices and a constant* δ *satisfying*

$$[\mathrm{pr}(\alpha, s)](S) - \psi_V(S) > \delta,$$

then

$$\Phi(\mathrm{pr}(\alpha, s)) < \sqrt{\frac{12\alpha \log(4\sqrt{\mathrm{vol}(S)}/\delta)}{\delta}}.$$

We will prove this theorem later in this section, but first we need a few lemmas. Consider a PageRank vector defined by the usual equation,

$$\mathrm{pr}(\alpha, s) = \alpha s + (1-\alpha)\mathrm{pr}(\alpha, s)W.$$

If we write $p = \mathrm{pr}(\alpha, s)$, then the equation above relates $p = \mathrm{pr}(\alpha, s)$ to pW. By writing $pW(S)$ in terms of $p\left(\mathrm{in}(S)\right)$ and $p\left(\mathrm{out}(S)\right)$ using Lemma 2.1, we can relate $\mathrm{pr}(\alpha, s)$ to itself. As a result, we can bound the height of the curve $p[x]$ in terms of other points on the same curve.

Lemma 3.2. *If* $p = \mathrm{pr}(\alpha, s)$ *is a PageRank vector, then for any set* S *of vertices,*

$$p(S) = \frac{\alpha}{2-\alpha} s(S) + (1 - \frac{\alpha}{2-\alpha}) \left(\frac{1}{2} p\left(\mathrm{in}(S) \cap \mathrm{out}(S)\right) + \frac{1}{2} p\left(\mathrm{in}(S) \cup \mathrm{out}(S)\right) \right).$$

Furthermore, for each $j \in [1, n-1]$*,*

$$p\left[\mathrm{vol}(S_j^p)\right] \leq \frac{\alpha}{2-\alpha} s\left[\mathrm{vol}(S_j^p)\right] + (1 - \frac{\alpha}{2-\alpha}) \left(\frac{1}{2} p\left[\mathrm{vol}(S_j^p) - |\partial(S_j^p)|\right] + \frac{1}{2} p\left[\mathrm{vol}(S_j^p) + |\partial(S_j^p)|\right] \right).$$

Proof of Lemma 3.2. Since $p = \mathrm{pr}(\alpha, s)$ is a PageRank vector, it satisfies the following equation for any set S of vertices,

$$p(S) = \alpha s(S) + (1-\alpha) pW(S).$$

Applying Lemma 2.1, we obtain

$$\begin{aligned} p(S) &\leq \alpha s(S) + (1-\alpha) pW(S) \\ &\leq \alpha s(S) + (1-\alpha) \left(p\left(\mathrm{in}(S) \cap \mathrm{out}(S)\right) + p\left(\mathrm{in}(S) \cup \mathrm{out}(S)\right) - p(S) \right). \end{aligned}$$

By adding the term $(1-\alpha)p(S)$ to both sides and then dividing by $2-\alpha$, we obtain

$$p(S) \leq \frac{\alpha}{2-\alpha} s(S) + (1 - \frac{\alpha}{2-\alpha}) \left(\frac{1}{2} p\left(\text{in}(S) \cap \text{out}(S)\right) + \frac{1}{2} p\left(\text{in}(S) \cup \text{out}(S)\right) \right).$$

This proves the first part of the lemma. To prove the second part, recall from Proposition 2.4 that $p\left[\text{vol}(S_j^p)\right] = p(S_j^p)$ for any integer $j \in [0,n]$, and that for any set of directed edges A, we have the bound $p(A) \leq p\left[|A|\right]$.

$$\begin{aligned}
p\left[\text{vol}(S_j^p)\right] &= p(S_j^p) \\
&= \frac{\alpha}{2-\alpha} s(S_j^p) + \\
&\quad (1 - \frac{\alpha}{2-\alpha}) \left(\frac{1}{2} p\left(\text{in}(S_j^p) \cap \text{out}(S_j^p)\right) + \frac{1}{2} p\left(\text{in}(S_j^p) \cup \text{out}(S_j^p)\right) \right) \\
&\leq \frac{\alpha}{2-\alpha} s\left[\text{vol}(S_j^p)\right] + \\
&\quad (1 - \frac{\alpha}{2-\alpha}) \left(\frac{1}{2} p\left[\left|\text{in}(S_j^p) \cup \text{out}(S_j^p)\right|\right] + \frac{1}{2} p\left[\left|\text{in}(S_j^p) \cap \text{out}(S_j^p)\right|\right] \right) \\
&\leq \frac{\alpha}{2-\alpha} s\left[\text{vol}(S_j^p)\right] + \\
&\quad (1 - \frac{\alpha}{2-\alpha}) \left(\frac{1}{2} p\left[\text{vol}(S_j^p) - |\partial(S_j^p)|\right] + \frac{1}{2} p\left[\text{vol}(S_j^p) + |\partial(S_j^p)|\right] \right).
\end{aligned}$$

Here we have used Proposition 2.6.

□

One consequence of the previous lemma is that the curve $\text{pr}(\alpha, s)\left[x\right]$ of a PageRank vector lies below the curve $s\left[x\right]$ of its starting vector.

Lemma 3.3. *For any starting vector s, and any $x \in [0, 2m]$,*

$$\text{pr}(\alpha, s)\left[x\right] \leq s\left[x\right].$$

Proof of Lemma 3.3. If we let $p = \text{pr}(\alpha, s)$, Lemma 3.2 implies that for each $j \in [1, n-1]$,

$$\begin{aligned}
p\left[\text{vol}(S_j^p)\right] &\leq \frac{\alpha}{2-\alpha} s\left[\text{vol}(S_j^p)\right] + \\
&\quad (1 - \frac{\alpha}{2-\alpha}) \left(\frac{1}{2} p\left[\text{vol}(S_j^p) - |\partial(S_j^p)|\right] + \frac{1}{2} p\left[\text{vol}(S_j^p) + |\partial(S_j^p)|\right] \right) \\
&\leq \frac{\alpha}{2-\alpha} s\left[\text{vol}(S_j^p)\right] + (1 - \frac{\alpha}{2-\alpha}) p\left[\text{vol}(S_j^p)\right].
\end{aligned}$$

The last line follows from the concavity of $p[k]$. This implies that $p\left[\text{vol}(S_j^p)\right] \leq s\left[\text{vol}(S_j^p)\right]$ for each $j \in [1, n-1]$. The same equation then holds for all $x \in [0, 2m]$, because $s[x]$ is concave and $p[x]$ is linear between the points $\text{vol}(S_j^p)$ and $\text{vol}(S_{j+1}^p)$. □

We can prove a stronger upper bound on the curve $\text{pr}(\alpha, s)[x]$ than the bound in Lemma 3.3. Lemma 3.2 shows that for $p = \text{pr}(\alpha, s)$, the height of the curve $\text{pr}(\alpha, s)[x]$ at the point $\text{vol}(S_j^p)$ is at most the average of the heights of the same curve at the two points $\text{vol}(S_j^p) - |\partial(S_j^p)|$ and $\text{vol}(S_j^p) + |\partial(S_j^p)|$, plus a small term that depends on α. In the following theorem, we will use this averaging equation to derive a series of upper bounds on the curve $\text{pr}(\alpha, s)[x]$. As a result, we can show that one of the following must be true: either $p(S) - \psi_V(S)$ is not very large for any set S of vertices in the graph, or else there is some sweep set S_j^p that has small conductance. In fact we will prove a stronger statement, which we will use in the next chapter. If $\text{pr}(\alpha, s)$ is far from ψ_V, then there is a sweep set S_j^p that has small conductance and for which $p(S_j^p) - \psi_V(S_j^p)$ is fairly large.

Theorem 3.4. *Let $p = \text{pr}(\alpha, s)$ be a PageRank vector with $\|s_+\|_1 \leq 1$. Let ϕ and γ be any constants in $[0, 1]$. Either the following bound holds for any set of vertices S and any integer t:*

$$p(S) - \psi_V(S) \leq \gamma + \alpha t + \sqrt{\min(\text{vol}(S), 2m - \text{vol}(S))}\left(1 - \frac{\phi^2}{8}\right)^t,$$

or else there exists a sweep cut S_j^p, for some $j \in [1, |\text{Supp}(p)|]$, with the following properties:

1. $\Phi(S_j^p) < \phi$,
2. *For some integer t,*

$$p(S_j^p) - \psi_V(S_j^p) > \gamma + \alpha t + \sqrt{\min(\text{vol}(S_j^p), 2m - \text{vol}(S_j^p))}\left(1 - \frac{\phi^2}{8}\right)^t.$$

Proof of Theorem 3.4. Let $f_t(x) = \gamma + \alpha t + \sqrt{\min(x, 2m - x)}\left(1 - \frac{\phi^2}{8}\right)^t$, and consider the equation

$$p[x] - \frac{x}{2m} \leq f_t(x) \qquad \text{for all } x \in [0, 2m]. \tag{3.1}$$

Assuming that there does not exist a sweep cut with both of the properties stated in the theorem, we will prove by induction that this equation holds for all $t \geq 0$. The theorem will follow.

For the base case $t = 0$, notice that for any value of x in the interval $[0, 2m]$,

$$p\left[x\right] - \frac{x}{2m} \leq \min(1, \min(x, 2m - x)) \leq \sqrt{\min(x, 2m - x)}.$$

Here, we have used that $\|s_+\|_1 \leq 1$. This shows that Equation (3.1) holds at $t = 0$ for any choice of γ and ϕ.

Now assume for the sake of induction that equation (3.1) holds for some specific t. To prove that equation (3.1) holds for $t+1$ at every point $x \in [0, 2m]$, it suffices to show that it holds for $t+1$ at the points $x_j = \mathrm{vol}(S_j^p)$ for each $j \in [1, |\mathrm{Supp}(p)|]$. We already know that the equation holds trivially at $x_0 = 0$ and $x_n = 2m$. The result will follow because we will have shown that $p\left[x\right] - x/2m$ is piecewise linear between a set of points for which equation (3.1) holds, and because $f_{t+1}(x)$ is concave.

Consider an index $j \in [1, |\mathrm{Supp}(p)|]$. We know that either property 1 or property 2 of the theorem does not hold for S_j. If property 2 does not hold, this directly implies that equation (3.1) holds with $t + 1$ and $x = x_j$. If property 1 does not hold, then we have $\Phi(S_j^p) \geq \phi$, and we will apply Lemma 3.2. There are two cases to consider, depending on whether $x_j \leq m$ or $x_j \geq m$. We will carry out the proof assuming that $x_j \leq m$. The proof for the other case is similar.

$$\begin{aligned}
p\left[\mathrm{vol}(S_j^p)\right] &\leq \frac{\alpha}{2-\alpha} s\left[\mathrm{vol}(S_j^p)\right] + \\
&\qquad (1 - \frac{\alpha}{2-\alpha}) \left(\frac{1}{2} p\left[\mathrm{vol}(S_j^p) - |\partial(S_j^p)|\right] + \frac{1}{2} p\left[\mathrm{vol}(S_j^p) + |\partial(S_j^p)|\right]\right) \\
&\leq \alpha + \left(\frac{1}{2} p\left[\mathrm{vol}(S_j^p) - |\partial(S_j^p)|\right] + \frac{1}{2} p\left[\mathrm{vol}(S_j^p) + |\partial(S_j^p)|\right]\right) \\
&= \alpha + \left(\frac{1}{2} p\left[x_j - \Phi(S_j^p) x_j\right] + \frac{1}{2} p\left[x_j + \Phi(S_j^p) x_j\right]\right) \\
&\leq \alpha + \left(\frac{1}{2} p\left[x_j - \phi x_j\right] + \frac{1}{2} p\left[x_j + \phi x_j\right]\right).
\end{aligned}$$

The second step above uses Proposition 2.6, and the last step above follows from the concavity of $p\left[x\right]$.

Using the induction hypothesis,

$$\begin{aligned} p\left[x_j\right] &\leq \alpha + \frac{1}{2}\left(f_t(x_j - \phi x_j) + \frac{x_j - \phi x_j}{2m} + f_t(x_j + \phi x_j) + \frac{x_j + \phi x_j}{2m}\right). \\ &= \alpha + \frac{x_j}{2m} + \frac{1}{2}\left(f_t(x_j - \phi x_j) + f_t(x_j + \phi x_j)\right). \end{aligned}$$

Therefore,

$$\begin{aligned} p\left[x_j\right] - \frac{x_j}{2m} &\leq \alpha + \frac{1}{2}\left(f_t(x_j - \phi x_j) + f_t(x_j + \phi x_j)\right) \\ &= \gamma + \alpha + \alpha t + \frac{1}{2}\left(\sqrt{x_j - \phi x_j} + \sqrt{x_j + \phi x_j}\right)\left(1 - \frac{\phi^2}{8}\right)^t \\ &= \gamma + \alpha + \alpha t + \frac{1}{2}\sqrt{x_j}\left(\sqrt{1-\phi} + \sqrt{1+\phi}\right)\left(1 - \frac{\phi^2}{8}\right)^t \\ &\leq \gamma + \alpha + \alpha t + \sqrt{x_j}\left(1 - \frac{\phi^2}{8}\right)\left(1 - \frac{\phi^2}{8}\right)^t \\ &= f_{t+1}(x_j). \end{aligned}$$

The second-to-last line follows from Proposition 2.7. □

We will now derive Theorem 3.1 from the previous theorem.

Proof of Theorem 3.1. Let $\phi = \Phi(\mathrm{pr}(\alpha, s))$. Theorem 3.4 implies that for any integer $t \geq 0$ and any set S,

$$[\mathrm{pr}(\alpha, s)](S) - \psi_V(S) \leq \alpha t + \sqrt{\min(\mathrm{vol}(S), 2m - \mathrm{vol}(S))}\left(1 - \frac{\phi^2}{8}\right)^t.$$

If we set

$$t = \left\lceil \frac{8}{\phi^2}\log(4\sqrt{\mathrm{vol}(S)}/\delta)) \right\rceil \leq \frac{9}{\phi^2}\log(4\sqrt{\mathrm{vol}(S)}/\delta),$$

then we have

$$\sqrt{\min(\mathrm{vol}(S), 2m - \mathrm{vol}(S))}\left(1 - \frac{\phi^2}{8}\right)^t \leq \frac{\delta}{4}.$$

This gives the bound

$$[\mathrm{pr}(\alpha, s)](S) - \psi_V(S) \leq \alpha\frac{9}{\phi^2}\log(4\sqrt{\mathrm{vol}(S)}/\delta) + \frac{\delta}{4}.$$

On the other hand, we have assumed that $[\mathrm{pr}(\alpha, s)](S) - \psi_V(S) > \delta$. Combining these upper and lower bounds yields the following inequality,

$$\frac{3\delta}{4} < \alpha\frac{9}{\phi^2}\log(4\sqrt{\mathrm{vol}(S)}/\delta).$$

The result follows by solving for ϕ. □

3.2 Lower bounds on PageRank concentration

In this section, we describe a way to lower bound the curve $p[x]$ of certain personalized PageRank vectors. For any set C with small conductance, we identify a set of good starting vertices v within C for which we can give a lower bound on the amount of probability from $\mathrm{pr}(\alpha, v)$ on the set C.

Definition 3.5. Given a set C, let C_α be the set of vertices v within C such that $\mathrm{pr}(\alpha, v)(\bar{C})$ is at most $2\Phi(C)/\alpha$.

Lemma 3.6 (Probability Capturing Lemma)**.** *For any set C and value of α, we have* $\mathrm{vol}(C_\alpha) \geq (1/2)\mathrm{vol}(C)$.

Proof of Lemma 3.6. We are trying to prove that a set C with small conductance contains a significant amount of probability from $\mathrm{pr}(\alpha, v)$ for many of the vertices v in C. We first show that this holds for an average of the vertices in C, by showing that the PageRank vector $\mathrm{pr}(\alpha, \psi_C)$ satisfies

$$[\mathrm{pr}(\alpha, \psi_C)](\bar{C}) \leq \Phi(C)\frac{1-\alpha}{\alpha}. \tag{3.2}$$

We then observe that if we sample a vertex from the distribution ψ_C, at least half of the time $\mathrm{pr}(\alpha, v)$ is less than twice its expectation, which is $\mathrm{pr}(\alpha, \psi_C)$.

We will prove equation (3.2) by examining the movement of probability during the single step from $\mathrm{pr}(\alpha, \psi_C)$ to $\mathrm{pr}(\alpha, \psi_C)W$. The amount of probability that moves from C to $\bar{C}$ in the step from $\mathrm{pr}(\alpha, \psi_C)$ to $\mathrm{pr}(\alpha, \psi_C)W$ is bounded by $\mathrm{pr}(\alpha, \psi_C)\left[|\partial(C)|\right]$, so

$$[\mathrm{pr}(\alpha, \psi_C)W]\left(\bar{C}\right) \leq [\mathrm{pr}(\alpha, \psi_C)]\left(\bar{C}\right) + \mathrm{pr}(\alpha, \psi_C)\left[|\partial(C)|\right].$$

By Lemma 3.3,

$$\begin{aligned} \mathrm{pr}(\alpha, \psi_C)\left[|\partial(C)|\right] &\leq \psi_C\left[|\partial(C)|\right] \\ &= \frac{|\partial(C)|}{\mathrm{vol}(C)} \\ &= \Phi(C). \end{aligned}$$

We combine this observation with the PageRank equation to obtain the following.

$$\begin{aligned}
[\mathrm{pr}(\alpha,\psi_C)]\left(\bar{C}\right) &= [\alpha\psi_C + (1-\alpha)\mathrm{pr}(\alpha,\psi_C)W]\left(\bar{C}\right) \\
&= (1-\alpha)[\mathrm{pr}(\alpha,\psi_C)W]\left(\bar{C}\right) \\
&\leq (1-\alpha)[\mathrm{pr}(\alpha,\psi_C)]\left(\bar{C}\right) + (1-\alpha)\mathrm{pr}(\alpha,\psi_C)\left[|\partial(C)|\right].
\end{aligned}$$

This implies that

$$[\mathrm{pr}(\alpha,\psi_C)](\bar{C}) \leq \frac{1-\alpha}{\alpha}\mathrm{pr}(\alpha,\psi_C)\left[|\partial(C)|\right] \leq \frac{1-\alpha}{\alpha}\Phi(C).$$

To complete the proof, let C_α be the set of vertices v in C satisfying

$$\mathrm{pr}(\alpha,v)(\bar{C}) \leq 2\frac{\Phi(C)}{\alpha}.$$

Let v be a vertex chosen randomly from the distribution ψ_C, and define the random variable $X = \mathrm{pr}(\alpha,v)(\bar{C})$. The linearity property of PageRank vectors from Proposition 2.9 implies the following bound on the expectation of X.

$$\mathrm{E}\left[X\right] = \mathrm{pr}(\alpha,\psi_C)(\bar{C}) \leq \frac{1-\alpha}{\alpha}\Phi(C) \leq \frac{\Phi(C)}{\alpha}.$$

Applying Markov's inequality yields

$$\Pr\left[v \notin C_\alpha\right] \leq \Pr\left[X > 2\mathrm{E}\left[X\right]\right] \leq \frac{1}{2}.$$

Since $\Pr\left[v \in C_\alpha\right] \geq \frac{1}{2}$, the volume of C_α is at least $\frac{1}{2}\mathrm{vol}(C)$. □

3.3 Computing approximate PageRank vectors

Instead of computing the personalized PageRank vector $\mathrm{pr}(\alpha,v)$ exactly, we will approximate it by another PageRank vector with a slightly different starting vector, $\mathrm{pr}(\alpha,1_v - r)$, where r is a vector with nonnegative entries. If $r(u) \leq \epsilon d(u)$ for every vertex in the graph, then we say $\mathrm{pr}(\alpha,1_v - r)$ is an ϵ-approximate PageRank vector for $\mathrm{pr}(\alpha,v)$.

Definition 3.7. Given a starting vector s, an *ϵ-approximate* PageRank vector for $\mathrm{pr}(\alpha,s)$ is a PageRank vector $\tilde{p}$ such that $\tilde{p} \geq 0$, and $\tilde{p} = \mathrm{pr}(\alpha,s-r)$ where the vector r is nonnegative and satisfies $r(u) \leq \epsilon d(u)$ for every vertex u in the graph.

We remark that the total difference between an ϵ-approximate PageRank vector $\mathrm{pr}(\alpha, s-r)$ and the PageRank vector $\mathrm{pr}(\alpha, s)$ on a given set S can be bounded in terms of ϵ and $\mathrm{vol}(S)$. Such a bound is given in Section 3.4.

In this section, we give an algorithm $\texttt{ApproximatePR}(v, \alpha, \epsilon)$ for computing an ϵ-approximate PageRank vector for $\mathrm{pr}(\alpha, v)$ with small support. The running time of the algorithm depends on ϵ and α, but is independent of the size of the graph.

Theorem 3.8. *The algorithm* $\texttt{ApproximatePR}(v, \alpha, \epsilon)$ *has the following properties. For any starting vertex* v *and any constant* $\epsilon \in (0, 1]$*, the algorithm computes an* ϵ*-approximate PageRank vector* p *for* $\mathrm{pr}(\alpha, v)$*. The support of* p *satisfies* $\mathrm{vol}(\mathrm{Supp}(p)) \leq \frac{2}{\epsilon}$*, and the running time of the algorithm is* $O(\frac{1}{\epsilon\alpha})$*.*

The proof of Theorem 3.8 is based on a series of facts which we describe below. The starting point is the observation that the PageRank operator commutes with the random walk matrix W,

$$\mathrm{pr}(\alpha, s)W = \mathrm{pr}(\alpha, sW). \tag{3.3}$$

Proof of Equation 3.3. The following sequence of equations shows that the vector $\mathrm{pr}(\alpha, s)W$ obeys the PageRank equation for $\mathrm{pr}(\alpha, sW)$. This equation has a unique solution, and so $\mathrm{pr}(\alpha, s)W = \mathrm{pr}(\alpha, sW)$.

$$\begin{aligned}
\mathrm{pr}(\alpha, s) &= \alpha s + (1-\alpha)\mathrm{pr}(\alpha, s)W \\
\mathrm{pr}(\alpha, s)W &= \alpha sW + (1-\alpha)\mathrm{pr}(\alpha, s)W^2 \\
(\mathrm{pr}(\alpha, s)W) &= \alpha(sW) + (1-\alpha)(\mathrm{pr}(\alpha, s)W)W.
\end{aligned}$$

□

By combining Equation (3.3) with the equation that defines PageRank, we derive the following equation,

$$\begin{aligned}
\mathrm{pr}(\alpha, s) &= \alpha s + (1-\alpha)\mathrm{pr}(\alpha, s)W \\
&= \alpha s + (1-\alpha)\mathrm{pr}(\alpha, sW).
\end{aligned} \tag{3.4}$$

Jeh and Widom derived this equation, and used it to design an algorithm for computing many PageRank vectors simultaneously [30]. The same equation was

used by Berkhin [8] in the Bookmark Coloring Algorithm. The algorithms they proposed can be used to compute a single approximate PageRank vector in time $O(\frac{\log n}{\epsilon\alpha})$. The difference of $\log n$ between their running time and ours is the overhead that they incur by using a heap or priority queue instead of a FIFO queue.

Our algorithm maintains a pair of vectors p and r, starting with the trivial approximation $p = \vec{0}$ and $r = 1_v$, and applies a series of push operations that move probability from r to p while maintaining the invariant $p = \mathrm{pr}(\alpha, 1_v - r)$. Each push operation takes all but $\epsilon/2$ units of the probability from r at a single vertex u, moves an α fraction of this probability to $p(u)$, and then spreads the remaining $(1-\alpha)$ fraction within r by applying a random walk step. We will define the push operation more formally below, and then verify that each push operation does maintain the invariant $p = \mathrm{pr}(\alpha, 1_v - r)$.

`push` (u):
Let $p' = p$ and $r' = r$, except for these changes:

1. $p'(u) = p(u) + \alpha(r(u) - \epsilon/2)$.
2. $r'(u) = \epsilon/2$.
3. For each vertex v such that $(u, v) \in E$:
   $r'(v) = r(v) + (1-\alpha)(r(u) - \epsilon/2)/d(u))$.

Lemma 3.9. *Let p' and r' be the result of performing* `push`(u) *on p and r. Then*

$$p = \mathrm{pr}(\alpha, 1_v - r) \quad \Longrightarrow \quad p' = \mathrm{pr}(\alpha, 1_v - r').$$

Proof of Lemma 3.9. After the push operation, we have

$$p' = p + \alpha(r(u) - \epsilon/2)1_u.$$
$$r' = r - (r(u) - \epsilon/2)1_u + (1-\alpha)(r(u) - \epsilon/2)1_u W.$$

We apply equation (3.4) to to show that $p + \mathrm{pr}(\alpha, r) = p' + \mathrm{pr}(\alpha, r')$. The lemma

will follow by the linearity property of PageRank vectors.

$$\begin{aligned}
\mathrm{pr}(\alpha, r) &= \mathrm{pr}(\alpha, r - (r(u) - \epsilon/2)1_u) + \mathrm{pr}(\alpha, (r(u) - \epsilon/2)1_u) \\
&= \mathrm{pr}(\alpha, r - (r(u) - \epsilon/2)1_u) + \alpha(r(u) - \epsilon/2)1_u \\
&\quad + \mathrm{pr}(\alpha, (1-\alpha)(r(u) - \epsilon/2)1_u W) \\
&= \mathrm{pr}(\alpha, r - (r(u) - \epsilon/2)1_u + (1-\alpha)(r(u) - \epsilon/2)1_u W) + \alpha(r(u) - \epsilon/2)1_u \\
&= \mathrm{pr}(\alpha, r') + p' - p.
\end{aligned}$$

□

During each push operation, some probability is moved from r to p, where it remains. By performing pushes only from vertices where $r(u) \geq \epsilon d(u)$, we can ensure that at least $\epsilon/2$ units of probability are moved at each step. This allows us to bound the number of pushes required to compute an ϵ-approximate PageRank vector.

`ApproximatePR` (s, α, ϵ):

1. Let $p = \vec{0}$, and $r = 1_v$.
2. While $r(u) \geq \epsilon d(u)$ for some vertex u:
   (a) Pick any vertex u where $r(u) \geq \epsilon d(u)$.
   (b) Apply `push` (u).
3. Return p and r.

This algorithm can be implemented by maintaining a queue containing those vertices u satisfying $r(u) \geq \epsilon d(u)$. Initially, the starting vertex v is the only vertex in the queue. At each step, a push operation is performed on the vertex u at the front of the queue, and u is then removed from the queue. If a push operation raises the value of $r(x)$ above $\epsilon d(x)$ for some neighbor x of u, then x is added to the back of the queue. This continues until the queue is empty, at which point all vertices in the graph satisfy $r(u) < \epsilon d(u)$. We now show that this algorithm has the properties promised in Theorem 3.8.

Proof of Theorem 3.8. Throughout the running of the algorithm the invariant $p = \mathrm{pr}(\alpha, 1_v - r)$ is preserved, and the stopping criterion ensures that r satisfies $r(u) < \epsilon d(u)$ at every vertex, so the algorithm returns an ϵ-approximate PageRank vector for $\mathrm{pr}(\alpha, s)$.

To bound the running time, let T be the total number of push operations performed by `ApproximatePR`, and let d_i be the degree of the vertex where the ith push operation was performed. When the ith push operation was performed, the amount of probability on this vertex was at least ϵd_i, so $\|r\|_1$ decreased by at least $\alpha(\epsilon/2)d_i$. Since $\|r\|_1$ was at most 1 initially, we must have $\alpha(\epsilon/2)\sum_{i=1}^{T} d_i \leq 1$, so

$$\sum_{i=1}^{T} d_i \leq \frac{2}{\epsilon\alpha}. \tag{3.5}$$

It is possible to perform a push operation on the vertex u, and to perform the necessary queue updates, in time proportional to $d(u)$. The running time bound for `ApproximatePR` follows from equation (3.5).

To bound the support volume, notice that for each vertex v in $\mathrm{Supp}(p)$, there are at least $\epsilon/2$ units of probability remaining on $r(v)$ when the algorithm terminates. It follows that $\mathrm{vol}(\mathrm{Supp}(p)) \leq \frac{2}{\epsilon}$. □

3.4 Finding a cut from an approximate PageRank vector

Consider the following procedure: pick a starting vertex v and a value of α, compute an ϵ-approximate PageRank vector for $\mathrm{pr}(\alpha, v)$, and perform a sweep over the resulting approximation. In this section, we show that for any set C having conductance $\alpha/20$, and for many of the vertices within C, this procedure finds a set having conductance $O(\sqrt{\alpha \log(\mathrm{vol}(C))})$.

Lemma 3.6 identifies a set of vertices v within C for which we can give a lower bound on the amount of probability from $\mathrm{pr}(\alpha, v)$ on the set C. We can give a similar lower bound on the amount of probability within C from an ϵ-approximate

PageRank vector. We use the following lemma to bound the amount of probability that is lost in the approximation.

Lemma 3.10. *For any ϵ-approximate PageRank vector $\mathrm{pr}(\alpha, s-r)$, and any set S of vertices,*

$$[\mathrm{pr}(\alpha, s)](S) \geq [\mathrm{pr}(\alpha, s-r)](S) \geq [\mathrm{pr}(\alpha, s)](S) - \epsilon \mathrm{vol}(S).$$

Proof of Lemma 3.10. Since the vector r is nonnegative,

$$\mathrm{pr}(\alpha, s-r) = \mathrm{pr}(\alpha, s) - \mathrm{pr}(\alpha, r) \leq \mathrm{pr}(\alpha, s).$$

To prove the other half of the lemma, we use the monotonicity property from Lemma 3.3 to bound the difference between $\mathrm{pr}(\alpha, s)$ and an ϵ-approximate PageRank vector $\mathrm{pr}(\alpha, s-r)$. For any set S, we have

$$[\mathrm{pr}(\alpha, r)](S) \leq \mathrm{pr}(\alpha, r)\left[\mathrm{vol}(S)\right] \leq r\left[\mathrm{vol}(S)\right],$$

and so

$$\begin{aligned}
[\mathrm{pr}(\alpha, s-r)](S) &= [\mathrm{pr}(\alpha, s)](S) - [\mathrm{pr}(\alpha, r)](S) \\
&\geq [\mathrm{pr}(\alpha, s)](S) - r\left[\mathrm{vol}(S)\right] \\
&\geq [\mathrm{pr}(\alpha, s)](S) - \epsilon \mathrm{vol}(S).
\end{aligned}$$

□

If v is a vertex in C_α, and if $\mathrm{pr}(\alpha, 1_v - r)$ is an ϵ-approximate PageRank vector, then

$$[\mathrm{pr}(\alpha, 1_v - r)](C) \geq 1 - 2\frac{\Phi(C)}{\alpha} - \epsilon \mathrm{vol}(C).$$

If both $\Phi(C)/\alpha$ and ϵ are small, there is still a significant amount of probability from $\mathrm{pr}(\alpha, 1_v - r)$ on the set C. Since $\mathrm{pr}(\alpha, 1_v - r)$ is a PageRank vector, we can then apply the mixing result from Theorem 3.1 to show that a sweep over $\mathrm{pr}(\alpha, 1_v - r)$ finds a cut with small conductance.

Theorem 3.11. *Let α be a constant in $(0, 1]$, and let C be a set satisfying*

1. $\Phi(C) \leq \alpha/20$,

2. $\mathrm{vol}(C) \leq \frac{2}{3}\mathrm{vol}(G)$.

If $\tilde{p} = \mathrm{pr}(\alpha, 1_v - r)$ *is an* ϵ*-approximate PageRank vector where* $v \in C_\alpha$ *and* $\epsilon \leq \frac{1}{10\mathrm{vol}(C)}$*, then a sweep over* $\tilde{p}$ *produces a cut with conductance*

$$\Phi(\tilde{p}) = O(\sqrt{\alpha \log(\mathrm{vol}(C))}).$$

Proof of Theorem 3.11. Let $\tilde{p} = \mathrm{pr}(\alpha, 1_v - r)$ be an ϵ-approximate PageRank vector for $\mathrm{pr}(\alpha, v)$ satisfying the assumptions of the theorem. Combining Lemma 3.6 with Lemma 3.10 gives a lower bound on $\tilde{p}(C)$,

$$\tilde{p}(C) \geq 1 - 2\frac{\Phi(C)}{\alpha} - \epsilon\mathrm{vol}(C).$$

Since $\Phi(C)/\alpha \leq 1/20$ and $\epsilon \leq 1/(10\mathrm{vol}(C))$, we have $\tilde{p}(C) \geq 4/5$, which implies

$$\tilde{p}(C) - \psi_V(C) \geq \frac{4}{5} - \frac{2}{3} = \frac{2}{15}.$$

Theorem 3.1 then implies

$$\Phi(\tilde{p}) < \sqrt{90\alpha \log(30\sqrt{\mathrm{vol}(C)})}.$$

□

As a corollary, there is some starting vertex v and value of α for which a sweep over $\mathrm{pr}(\alpha, 1_v)$ finds a cut with conductance near the minimum conductance in the graph.

Corollary 3.12. *Let* Φ_G *be the minimum conductance of any set of vertices in the graph, and let* C^{opt} *be a set achieving this minimum. If* $\tilde{p}$ *is an* ϵ*-approximate PageRank vector for* $\mathrm{pr}(\alpha, 1_v)$*, where* $\alpha = 10\Phi_G$*,* $v \in C^{\mathrm{opt}}_\alpha$*, and* $\epsilon \leq \frac{1}{10\mathrm{vol}(C^{\mathrm{opt}})}$*, then*

$$\Phi(\tilde{p}) = O(\sqrt{\Phi_G \log(\mathrm{vol}(C^{\mathrm{opt}}))}).$$

Corollary 3.12 follows from Theorem 3.11 by setting $C = C^{\mathrm{opt}}$.

3.5 Finding a cut in time proportional to its size

In the previous section, we showed that to find a cut within a set C, it suffices to compute an ϵ-approximate PageRank vector with ϵ roughly $1/\mathrm{vol}(C)$. This requires time proportional to $\mathrm{vol}(C)$, but the resulting cut S may have much smaller volume. This leaves us with a problem similar to the one facing spectral partitioning: we do not want to spend a large amount of time to find a cut whose volume is small.

In this section, we extend our local partitioning techniques to find a cut with small conductance in time proportional to the volume of the smaller side of the cut found. Essentially, we consider what would happen if we were to compute an approximate PageRank vector with different levels of error. We show that if a given level of error yields a cut whose volume is too small, we could have found a cut with similar volume more quickly by using a larger amount of error.

The result is an algorithm called `PageRank-Nibble` that takes a constant $\phi \in (0, 1]$ and a scale $b \in [1, \log m]$ as part of its input, and attempts to find a cut with conductance ϕ and volume at least 2^{b-1}. We prove that `PageRank-Nibble` finds a set with these properties for at least one choice of b in the range $[1, \lceil \log m \rceil]$ whenever v is a good starting vertex for a set C with sufficiently small conductance. In addition, we show that the resulting set has a large intersection with the set C.

`PageRank-Nibble`(v, ϕ, b):

Input: a vertex v, a constant $\phi \in (0, 1]$, and an integer $b \in [1, B]$, where $B = \lceil \log m \rceil$.

1. Let $\alpha = \frac{\phi^2}{225 \log(100\sqrt{m})}$.
2. Compute an ϵ-approximate PageRank vector $p = \mathrm{pr}(\alpha, 1_v - r)$, with $\epsilon = \left(2^b \cdot 48\lceil \log m \rceil\right)^{-1}$.
3. For each $j \in [1, |\mathrm{Supp}(p)|]$, check whether S_j^p obeys the following conditions:

   conductance: $\Phi(S_j^p) < \phi$,

   volume: $2^{b-1} < \mathrm{vol}(S_j^p) < \frac{2}{3}\mathrm{vol}(G)$,

   probability: $p\left[2^b\right] - p\left[2^{b-1}\right] > \frac{1}{48\lceil \log m \rceil}$,
4. If some set S_j^p satisfies all of these conditions, return S_j^p. Otherwise, return nothing.

Theorem 3.13. `PageRank-Nibble`(v, ϕ, b) *can be implemented with running time* $O(2^b \frac{\log^2 m}{\phi^2})$.

Proof of Theorem 3.13. An ϵ-approximate PageRank vector p with ϵ set to $\left(2^b \cdot 48\lceil \log m \rceil\right)^{-1}$ can be computed in time $O(2^b \frac{\log m}{\alpha})$ using `ApproximatePR`. By Theorem 3.8, the support of this vector has volume $O(2^b \log m)$, and the number of vertices in the support is $N_p = O(2^b \log m)$. It is possible to check each of the conditions in step 3 of `PageRank-Nibble`, for every set S_j^p with $j \in [1, N_p]$, in the amount of time required to sort and perform a sweep over p, which is

$$O(\mathrm{vol}(\mathrm{Supp}(p)) + N_p \log N_p) = O(2^b \log^2 m).$$

Since we have set $\alpha = \Omega(\phi^2 / \log m)$, the running time of `PageRank-Nibble` is

$$O(2^b \frac{\log m}{\alpha} + 2^b \log^2 m) = O(2^b \frac{\log^2 m}{\phi^2}).$$

□

Theorem 3.14. *Let C be a set whose volume satisfies $\mathrm{vol}(C) \leq \frac{1}{2}\mathrm{vol}(G)$ and whose conductance satisfies $\Phi(C) \leq \phi^2/(55000 \log^2 100m)$, and let v be a vertex in C_α for the value of α used in* `PageRank-Nibble`*, which is $\phi^2/(225 \log(100\sqrt{m}))$. Then, there is some integer $b \in [1, \lceil \log m \rceil]$ for which* `PageRank-Nibble`*(v, ϕ, b) finds a set S meeting all of its criteria. Any such set has the following properties:*

conductance: $\Phi(S) < \phi$,

volume: $2^{b-1} < \mathrm{vol}(S) < \frac{2}{3}\mathrm{vol}(G)$,

intersection: $\mathrm{vol}(S \cap C) > 2^{b-2}$.

Proof of Theorem 3.14. Consider the PageRank vector $\mathrm{pr}(\alpha, 1_v)$. We have assumed that v is in C_α, and have set $\Phi(C)$ to ensure that $\Phi(C) \leq \alpha/(200\lceil \log m \rceil)$. This implies that

$$\begin{aligned} \mathrm{pr}(\alpha, 1_v)\left[\mathrm{vol}(C)\right] - \psi_V(C) &\geq (1 - 2\frac{\Phi(C)}{\alpha}) - \frac{1}{2} \\ &\geq \frac{1}{2} - \frac{1}{100}. \end{aligned}$$

We have set α so that if $t_0 = \lceil \frac{8}{\phi^2} \log(100\sqrt{m}) \rceil$, then $\alpha t_0 \leq 1/25$ when and with this choice of t_0 we have

$$\alpha t_0 + \sqrt{\mathrm{vol}(C)} \left(1 - \frac{\phi^2}{8}\right)^{t_0} < \frac{1}{25} + \frac{1}{100}.$$

Since $\frac{1}{2} - \frac{1}{100} > \frac{5}{12} + \frac{1}{25} + \frac{1}{100}$, the following equation holds.

$$\mathrm{pr}(\alpha, 1_v)\left[\mathrm{vol}(C)\right] - \frac{\mathrm{vol}(C)}{2m} > 5/12 + \alpha t_0 + \sqrt{\mathrm{vol}(C)} \left(1 - \frac{\phi^2}{8}\right)^{t_0}. \tag{3.6}$$

Let $B = \lceil \log m \rceil$. For each integer b in $[1, B]$, let $\gamma_b = \frac{5}{12}(\frac{9}{10} + \frac{1}{10}\frac{b}{B})$. We will consider the following equation.

$$\mathrm{pr}(\alpha, 1_v)\left[x\right] - \frac{x}{2m} > \gamma_b + \alpha t + \sqrt{x} \left(1 - \frac{\phi^2}{8}\right)^{t}. \tag{3.7}$$

We have already shown that this holds with $b = B$, $x = m$, and $t = t_0$, in Equation (3.6). Let b_0 be the smallest value of b for which this equation holds

for some $x_0 \leq 2^b$ and for some value of t. We will show that `PageRank-Nibble` successfully returns a cut when it is run with $b = b_0$.

When `PageRank-Nibble` is run with $b = b_0$, it computes an ϵ-approximate PageRank vector $\mathrm{pr}(\alpha, 1_v - r)$ with $\epsilon \leq (2^{b_0} 48B)^{-1}$. With this amount of error, we have

$$\begin{aligned}
\mathrm{pr}(\alpha, 1_v - r)\,[x_0] &\geq \mathrm{pr}(\alpha, 1_v)\,[x_0] - \left(\frac{2^{-b_0}}{48B}\right) x_0 \\
&\geq \mathrm{pr}(\alpha, 1_v)\,[x_0] - \frac{1}{48B} \\
&\geq \mathrm{pr}(\alpha, 1_v)\,[x_0] - (\gamma_{b_0} - \gamma_{b_0 - 1}) + \frac{1}{48B},
\end{aligned}$$

where the last line follows because $\gamma_{b_0} - \gamma_{b_0 - 1} \leq \frac{1}{24B}$. Since equation (3.7) holds for b_0 and x_0, we have for some integer $t \geq 0$,

$$\begin{aligned}
\mathrm{pr}(\alpha, 1_v - r)\,[x_0] - \frac{x_0}{2m} &> \left(\gamma_{b_0} + \alpha t + \sqrt{x_0}\left(1 - \frac{\phi^2}{8}\right)^t\right) - (\gamma_{b_0} - \gamma_{b_0 - 1}) + \frac{1}{48B} \\
&> (\gamma_{b_0 - 1} + \frac{1}{48B}) + \alpha t + \sqrt{x_0}\left(1 - \frac{\phi^2}{8}\right)^t.
\end{aligned}$$

Theorem 3.4 then shows that there exists a sweep cut S_j, with $S_j = S_j^{\mathrm{pr}(\alpha, 1_v - r)}$ for some value of j in the range $[1, |\mathrm{Supp}(\mathrm{pr}(\alpha, 1_v - r))|]$, such that $\Phi(S_j) \leq \phi$, and such that following lower bound holds for some integer t':

$$\mathrm{pr}(\alpha, 1_v - r)(S_j) - \frac{\mathrm{vol}(S_j)}{2m} > (\gamma_{b_0 - 1} + \frac{1}{48B}) + \alpha t' + \sqrt{\overline{\mathrm{vol}}(S_j)}\left(1 - \frac{\phi^2}{8}\right)^{t'}, \quad (3.8)$$

where $\overline{\mathrm{vol}}(S_j) = \min(\mathrm{vol}(S_j), 2m - \mathrm{vol}(S_j))$. We will show that this cut S_j meets all the requirements of `PageRank-Nibble`, which will prove that the algorithm outputs some cut when run with $b = b_0$.

First, assume for the sake of contradiction that $\mathrm{vol}(S_j) \leq 2^{b_0 - 1}$. Since equation (3.7) cannot hold with $b = b_0 - 1$ and $x \leq 2^{b_0 - 1}$, this implies that for any integer $t \geq 0$,

$$\begin{aligned}
\mathrm{pr}(\alpha, 1_v - r)(S_j) - \frac{\mathrm{vol}(S_j)}{2m} &= \mathrm{pr}(\alpha, 1_v - r)\,[\mathrm{vol}(S_j)] - \frac{\mathrm{vol}(S_j)}{2m} \\
&\leq \mathrm{pr}(\alpha, 1_v)\,[\mathrm{vol}(S_j)] - \frac{\mathrm{vol}(S_j)}{2m} \\
&\leq \gamma_{b_0 - 1} + \alpha t + \sqrt{\mathrm{vol}(S_j)}\left(1 - \frac{\phi^2}{8}\right)^t.
\end{aligned}$$

Since $\overline{\text{vol}}(S_j) = \text{vol}(S_j)$ when $x \leq 2^{b_0-1}$, this contradicts the lower bound from equation (3.8). Therefore, it must be true that $\text{vol}(S_j) > 2^{b_0-1}$.

It must also be true that $\text{vol}(S_j) < \frac{2}{3}\text{vol}(G)$. Otherwise, the lower bound from equation (3.8) would imply that for some integer $t' \geq 0$,

$$\begin{aligned} \text{pr}(\alpha, 1_v - r)(S_j) &> \frac{\text{vol}(S_j)}{2m} + \gamma_{b_0-1} + \alpha t' + \sqrt{\text{vol}(S_j)} \left(1 - \frac{\phi^2}{8}\right)^{t'} \\ &> \frac{2}{3} + \gamma_{b_0-1} \\ &\geq \frac{2}{3} + \frac{9}{10}\frac{5}{12}. \end{aligned}$$

This implies $\text{pr}(\alpha, 1_v - r)(S_j) > 1$, which is a contradiction.

We will now prove that there is a significant difference in probability between $\text{pr}(\alpha, 1_v - r)\left[2^{b_0}\right]$ and $\text{pr}(\alpha, 1_v - r)\left[2^{b_0-1}\right]$. Since equation (3.8) does not hold with $b = b_0 - 1$ and $x = 2^{b_0-1}$, we know that for every integer $t \geq 0$,

$$\text{pr}(\alpha, 1_v - r)\left[2^{b_0-1}\right] - \frac{x_0}{2m} \leq \gamma_{b_0-1} + \alpha t + \sqrt{2^{b_0-1}} \left(1 - \frac{\phi^2}{8}\right)^{t}. \tag{3.9}$$

We also know that for some integer t',

$$\text{pr}(\alpha, 1_v - r)\left[x_0\right] - \frac{x_0}{2m} > (\gamma_{b_0-1} + \frac{1}{48B}) + \alpha t' + \sqrt{x_0} \left(1 - \frac{\phi^2}{8}\right)^{t'}. \tag{3.10}$$

By plugging t' into Equation (3.9), we obtain the following inequality.

$$\begin{aligned} \text{pr}(\alpha, 1_v - r)\left[2^{b_0}\right] - \text{pr}(\alpha, 1_v - r)\left[2^{b_0-1}\right] &\geq \text{pr}(\alpha, 1_v - r)\left[x_0\right] - \text{pr}(\alpha, 1_v - r)\left[2^{b_0-1}\right] \\ &> \frac{1}{48B}. \end{aligned}$$

We have shown that S_j meets all the requirements of `PageRank-Nibble`, which proves that the algorithm outputs some cut when run with $b = b_0$. We now prove a lower bound on $\text{vol}(S \cap C)$, which holds for any cut S output by `PageRank-Nibble`, regardless of whether the algorithm was run with $b = b_0$ or with some other value of b. Let $p'[x] = p[x] - p[x-1]$. Since $p'[x]$ is a decreasing function of x,

$$\begin{aligned} p'\left[2^{b-1}\right] &\geq \frac{p\left[2^b\right] - p\left[2^{b-1}\right]}{2^b - 2^{b-1}} \\ &> \frac{1}{2^{(b-1)}48B}. \end{aligned}$$

It is not hard to see that combining this lower bound on $p'\left[2^{b-1}\right]$ with the upper bound $p(\bar{C}) \leq 2\frac{\Phi(C)}{\alpha}$ gives the following bound on the volume of the intersection.

$$\begin{aligned} \mathrm{vol}(S_j \cap C) &\geq 2^{b-1} - \frac{p(\bar{C})}{p'\left[2^{b-1}\right]} \\ &> 2^{b-1} - 2^{b-1}(96B\frac{\Phi(C)}{\alpha}). \end{aligned}$$

Since we have assumed that $\frac{\Phi(C)}{\alpha} \leq \frac{1}{200B}$, we have

$$\mathrm{vol}(S \cap C) > 2^{b-1} - 2^{b-2} = 2^{b-2}.$$

□

3.6 Comparison of local partitioning algorithms

`PageRank-Nibble` improves the running time and approximation ratio of the `Nibble` algorithm of Spielman and Teng [47]. In that paper, `Nibble` was called repeatedly with randomly chosen starting vertices and scales to create an algorithm called `Partition`, which finds a cut with small conductance and approximately maximal volume. `Partition` was applied recursively to create algorithms for multiway partitioning, graph sparsification, and solving diagonally dominant linear systems.

An algorithm `PageRank-Partition` can be created by calling `PageRank-Nibble` instead of `Nibble`. The resulting algorithm takes as input a parameter Φ and a graph, and has expected running time $O(m \log^2 m/\Phi)$. If there exists a set C with conductance at most Φ, then with high probability `PageRank-Partition` finds a set S with conductance $O\left((\Phi \log^2 m)^{1/2}\right)$ and volume at least $\mathrm{vol}(C)/2$.

In the table below, we compare our local partitioning algorithms with the algorithms of Spielman and Teng from [47]. The running times are stated in terms of Φ, which is the conductance of the set C. The approximation column states the bound $f(\Phi)$ on the conductance of the cut returned by the algorithm.

Table 3.1: Comparison of local partitioning algorithms

	Running time	Approximation Ratio
Nibble	$2^b \log^{2/3} m/\Phi^{5/3}$	$f(\Phi) = (\Phi \log^2 m)^{1/3}$
PR-Nibble	$2^b/\Phi$	$f(\Phi) = (\Phi \log^2 m)^{1/2}$
Partition	$m \log^{8/3} m/\Phi^{5/3}$	$f(\Phi) = (\Phi \log^2 m)^{1/3}$
PR-Partition	$m \log^2 m/\Phi$	$f(\Phi) = (\Phi \log^2 m)^{1/2}$

3.7 Acknowledgement

Chapter 3 contains material that appeared in the following article.

R. Andersen, F. Chung, and K. Lang. *Local graph partitioning using PageRank vectors*. In *FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 475–486, Washington, DC, USA, 2006. IEEE Computer Society. The dissertation author was the primary investigator and author of this paper.

4 Detecting Sharp Drops in Personalized PageRank and a Simplified Local Partitioning Algorithm

In this chapter, we consider the following consequence of the personalized PageRank equation,

$$p = \alpha 1_v + (1 - \alpha)pW.$$

When the random walk step W is applied to the personalized PageRank vector p, every vertex in the graph has more probability from the vector pW than it has from p, except for the seed vertex v. This implies something strong about the ordering of the vertices produced by the PageRank vector p: there cannot be many links between any set of vertices with high probability in p and any set of vertices with low probability in p. More precisely, whenever there is a *sharp drop* in probability, where the kth highest ranked vertex has much more probability than the $k(1+\delta)$th vertex, there must be few links between the k highest ranked vertices and the vertices not ranked in the top $k(1+\delta)$.

We will make this observation precise in Lemma 4.2, which provides an intuitive proof that personalized PageRank identifies a set with small conductance. In the section that follows, we prove a series of lemmas that describe necessary conditions for a sharp drop to exist. In the final section, we use these techniques to produce an efficient local partitioning algorithm that produces a cut in time proportional

to its volume.

4.1 A sharp drop reveals a good cut

We describe a normalized rank function derived from a PageRank vector, and the ordering of the vertices induced by that rank function.

Definition 4.1. Given a PageRank vector p, we define the following.

- Define the rank function q to be $q(u) = p(u)/d(u)$.
- Let π be a permutation that places the vertices in nonincreasing order of rank, so that
$$q(\pi(1)) \geq q(\pi(2)) \geq \cdots \geq q(\pi(n)).$$
This is the ordering induced by the PageRank vector. An integer $j \in [1, n]$ will be called an index, and we will say that $\pi(j)$ is the vertex at index j.
- Let $S_j = \{\pi(1), \ldots, \pi(j)\}$ be the set containing the j vertices of highest rank. The set S_j is called the jth level set of the PageRank vector.
- Define the shorthand notation $q(j) = q(\pi(j))$ and $V(j) = \mathrm{vol}(S_j)$.

We now prove the main lemma. If there is a sharp drop in rank at S_j, then the set S_j has small conductance. We will prove the contrapositive instead, because that is how we will eventually apply the lemma. Namely, we will prove that either the set S_j has small conductance, or else there is an index $k > j$ where the volume $\mathrm{vol}(S_k)$ is significantly larger than $\mathrm{vol}(S_j)$, and the rank $q(k)$ is not much smaller than $q(j)$.

Lemma 4.2 (Sharp Drop Lemma)**.** *Let $p = \mathrm{pr}(\alpha, 1_v - r)$ be an approximate PageRank vector. Let $\phi \in (0, 1)$ be a real number, and let j be any index in $[1, n]$. Either the number of edges leaving S_j satisfies $e(S_j, \bar{S}_j) < 2\phi\mathrm{vol}(S_j)$, or else there is some index $k > j$ such that*

$$\mathrm{vol}(S_k) \geq \mathrm{vol}(S_j)(1 + \phi) \qquad \textit{and} \qquad q(k) \geq q(j) - \alpha/\phi\mathrm{vol}(S_j).$$

Proof. For any set S of vertices,

$$pW(S) = p(S) - \sum_{(u,v)\in e(S,\bar{S})} q(u) - q(v).$$

Since $p = \mathrm{pr}(\alpha, 1_v - r)$ and the vector r is nonnegative,

$$pW = (1-\alpha)^{-1}(p - \alpha(1_v - r)) \geq p - \alpha 1_v.$$

The two equations above imply the following,

$$\sum_{(u,v)\in e(S,\bar{S})} q(u) - q(v) \leq \alpha. \tag{4.1}$$

Now consider the level set S_j. If $\mathrm{vol}(S_j)(1+\phi) > \mathrm{vol}(G)$, then

$$e(S_j\bar{S}_j) \leq \mathrm{vol}(G)(1 - \frac{1}{1+\phi}) \leq \phi\mathrm{vol}(S_j),$$

and the theorem holds trivially. If $\mathrm{vol}(S_j)(1+\phi) \leq \mathrm{vol}(G)$, then there is a unique index k such that

$$\mathrm{vol}(S_{k-1}) \leq \mathrm{vol}(S_j)(1+\phi) \leq \mathrm{vol}(S_k).$$

If $e(S_j, \bar{S}_j) < 2\phi\mathrm{vol}(S_j)$, we are done. If $e(S_j, \bar{S}_j) \geq 2\phi\mathrm{vol}(S_j)$, then $e(S_j, \bar{S}_{k-1})$ is also large by the following argument,

$$e(S_j, \bar{S}_{k-1}) \geq \partial(S_j) - \mathrm{vol}(S_{k-1} \setminus S_j) \geq 2\phi\mathrm{vol}(S_j) - \phi\mathrm{vol}(S_j) = \phi\mathrm{vol}(S_j).$$

Using equation (4.1),

$$\begin{aligned}
\alpha \geq \sum_{(u,v)\in e(S_j,\bar{S}_j)} q(u) - q(v) &\geq \sum_{(u,v)\in e(S_j,\bar{S}_{k-1})} q(u) - q(v) \\
&\geq e(S_j, \bar{S}_{k-1})(q(j) - q(k)) \\
&\geq \phi\mathrm{vol}(S_j) \cdot (q(j) - q(k)).
\end{aligned}$$

This shows that $q(j) - q(k) \leq \alpha/\phi\mathrm{vol}(S_j)$, completing the proof.

□

4.2 Ensuring that a sharp drop exists

In this section, we introduce several tools that will allow us to show that a sharp drop in rank exists for many personalized PageRank vectors. When we present the local partitioning algorithm in the next section, these tools will be used to prove its approximation guarantee.

Throughout this section and the next, we have two PageRank vectors to consider, the PageRank vector $p = \mathrm{pr}(\alpha, v)$, and the approximate PageRank vector $\tilde{p} = \mathrm{pr}(\alpha, 1_v - r)$ that will be computed by the local partitioning algorithm. These two PageRank vectors induce two different orderings π and $\tilde{\pi}$, which lead to two different rank functions $q(k)$ and $\tilde{q}(k)$, which produce two collections of level sets S_k and $\tilde{S}_k$, which have different volumes $V(k) = \mathrm{vol}(S_k)$ and $\tilde{V}(k) = \mathrm{vol}(\tilde{S}_k)$.

We start by showing there is some index i where the rank $q(i)$ is not much smaller than $1/V(i)$. This lemma doesn't use any special properties of PageRank vectors, and is true for any nonnegative vector whose entries sum to 1.

Lemma 4.3 (Integration Lemma)**.** *Let q be the rank function of any vector p for which $\|p\|_1 = 1$. Then, there exists an index i such that $q(i) \geq \frac{1}{H(2m)V(i)}$, where $H(2m) = \sum_{k=1}^{2m} 1/k = O(\log m)$.*

Proof. If we assume that $q(i) < c/V(i)$ for all $i \in [1, n]$, then

$$\begin{aligned}\sum_{i=1}^{n} q(i)d(i) &< c\sum_{i=1}^{n} \frac{d(i)}{V(i)} \\ &\leq c\sum_{k=1}^{2m} \frac{1}{k} \\ &= cH(2m).\end{aligned}$$

If $c = 1/H(2m)$, this would imply $\|p\|_1 = \sum_{i=1}^{n} q(i)d(i) < 1$, so we must have $q(i) \geq \frac{1}{H(2m)V(i)}$ for some index i. □

We now give a lower bound on the rank function of an ϵ-approximate PageRank vector $\tilde{p} = \mathrm{pr}(\alpha, 1_v - r)$ that depends on the rank function of the PageRank vector $p = \mathrm{pr}(\alpha, v)$ that is being approximated, and on the error parameter ϵ.

Lemma 4.4 (Approximation Error Lemma)**.** *Let q be the rank function for a PageRank vector $p = \mathrm{pr}(\alpha, v)$, and let $\tilde{q}$ be the rank for an ϵ-approximate PageRank vector $\tilde{p} = \mathrm{pr}(\alpha, 1_v - r)$. For any index i, there is an index j such that*

$$\tilde{q}(j) \geq q(i) - \epsilon \qquad \textit{and} \qquad \mathrm{vol}(\tilde{S}_j) \geq \mathrm{vol}(S_i).$$

Proof. If $v \in S_i$, then $p(v)/d(v) \geq q(i)$. Since $\tilde{p}$ is an ϵ-approximation of p,

$$\tilde{p}(v)/d(v) \geq p(v)/d(v) - \epsilon \geq q(i) - \epsilon.$$

Therefore, the set of vertices for which $\tilde{p}(v)/d(v) \geq q(i) - \epsilon$ has volume at least $\mathrm{vol}(S_j)$, which proves the lemma. □

The following lemma shows what happens if we repeatedly apply the Sharp Drop Lemma, but fail to find a cut with small conductance. In that case, we obtain a sequence of larger and larger indices for which the rank does not drop very quickly. We give a lower bound on the rank of the final index in the sequence. We will eventually contradict this lower bound, which will show that the Sharp Drop Lemma eventually finds a cut with small conductance.

Lemma 4.5 (Chaining Lemma)**.** *Let $\{k_0 \ldots k_f\}$ be an increasing sequence of indices for which the following holds for each $i \in [0, f-1]$,*

$$q(k_{i+1}) \geq q(k_i) - \alpha/\phi V(k_i) \qquad \textit{and} \qquad V(k_{i+1}) \geq (1+\phi)V(k_i).$$

Then, the last index k_f satisfies

$$q(k_f) \geq q(k_0) - 2\alpha/\phi^2 V(k_0).$$

Proof. The bound on the change in volume implies that $V(k_i) \geq (1+\phi)^i V(k_0)$ for all $i \in [0, f-1]$. Therefore,

$$\begin{aligned}
q(k_f) &\geq q(k_0) - \frac{\alpha}{\phi V(k_0)} - \frac{\alpha}{\phi V(k_1)} - \cdots - \frac{\alpha}{\phi V(k_{f-1})} \\
&\geq q(k_0) - \frac{\alpha}{\phi V(k_0)} \left(1 - \frac{1}{(1+\phi)} - \cdots - \frac{1}{(1+\phi)^{f-1}}\right) \\
&\geq q(k_0) - \frac{\alpha}{\phi V(k_0)} (\frac{1+\phi}{\phi}) \\
&\geq q(k_0) - \frac{2\alpha}{\phi^2 V(k_0)}.
\end{aligned}$$

This completes the proof of the Chaining Lemma. □

4.3 A simplified local partitioning algorithm

The local partitioning algorithm can be described as follows.

`Local Partition`(v, ϕ, x):

The input to the algorithm is a starting vertex v, a target conductance $\phi \in (0, 1/3)$, and a target volume $x \in [0, 2m]$. The output is either a set of vertices or a failure message.

PageRank computation:

1. Let $\gamma = H(2m)$, let $\alpha = \frac{\phi^2}{8\gamma}$, and let $\epsilon = \frac{1}{2\gamma x}$.
2. Compute an ϵ-approximate PageRank vector $\tilde{p} = \mathrm{pr}(\alpha, 1_v - r)$, using `ApproximatePR`$(v, \alpha, \epsilon)$.
3. Order the vertices so that $\tilde{q}(1) \geq \tilde{q}(2) \geq \cdots \geq \tilde{q}(n)$.

Finding a cut:

The algorithm examines a sequence of vertices, looking for a sharp drop. We let $\{k_i\}$ be the indices of the vertices examined by the algorithm. The first index examined by the algorithm will be k_0, and the last index examined will be k_f. We now describe how these indices are chosen.

1. Let the starting index k_0 be the largest index such that $\tilde{q}(k_0) \geq 1/2\gamma\tilde{V}(k_0)$. If no such index exists, halt and output `FAIL: no starting index`.
2. While the algorithm is still running:
   (a) If $(1+\phi)\tilde{V}(k_i) > \mathrm{vol}(G)$ or if $\tilde{V}(k_i) > \mathrm{vol}(\mathrm{Supp}(\tilde{p}))$, then let $k_f = k_i$, halt and output `FAIL: no cut found`.
   (b) Otherwise, let k_{i+1} be the smallest index such that $\tilde{V}(k_{i+1}) \geq \tilde{V}(k_i)(1+\phi)$.
   (c) If $\tilde{q}(k_{i+1}) \leq \tilde{q}(k_i) - \alpha/\phi\tilde{V}(k_i)$, then let $k_f = k_i$, output the set S_{k_i}, and quit. Otherwise, repeat the loop.

Theorem 4.6. *The running time of* `Local Partition`(v, ϕ, x) *is* $O(x\frac{\log^2 m}{\phi^2})$.

Proof. The running time of the algorithm is dominated by the time to compute and sort $\tilde{p}$. Computing $\tilde{p}$ can be done in time $O(1/\epsilon\alpha) = O(x\gamma/\alpha) = O(x\frac{\log m}{\alpha})$ using `ApproximatePR`. The support of $\tilde{p}$ has volume $O(1/\epsilon) = O(\gamma x) = O(x \log m)$, so the time required to sort $\tilde{p}$ is

$$O(|\text{Supp}(\tilde{p})| \log |\text{Supp}(\tilde{p})|) = O(x \log^2 m).$$

Since we have set $\alpha = \Omega(\phi^2/\log m)$, the total running time is

$$O(x\frac{\log m}{\alpha} + x \log^2 m) = O(x\frac{\log^2 m}{\phi^2}).$$

□

Theorem 4.7. *Consider a run of the algorithm* `Local Partition` *on the input values v, ϕ, and x. Let $\tilde{p} = \text{pr}(\alpha, 1_v - r)$ be the ϵ-approximate PageRank vector computed by the algorithm, for which $\alpha = \phi^2/8\gamma$ and $\epsilon = 1/2\gamma x$. The following statements are true.*

1. *Let q be the rank function of the PageRank vector $p = \text{pr}(\alpha, v)$. There exists an index K such that $q(K) \geq 1/\gamma\text{vol}(S_K)$. Furthermore, if the target volume x satisfies $x \geq \text{vol}(S_K)$, then the algorithm finds a starting index k_0 that satisfies $\text{vol}(\tilde{S}_{k_0}) \geq \text{vol}(S_K)$.*

2. *Assume there exists a set C whose volume satisfies $\text{vol}(C) \leq \frac{1}{2}\text{vol}(G)$, whose conductance satisfies $\Phi(C) \leq \phi^2/6400\gamma = \alpha/80\gamma$, and for which the starting vertex v is contained in C_α. If the target volume x satisfies $x \geq \text{vol}(S_K)$, then the algorithm successfully outputs a set. Furthermore, the set S output by the algorithm has the following properties.*

   (a) *(Approximation guarantee)* $\Phi(S) \leq 3\phi = 3\sqrt{8\gamma\alpha}$.

   (b) *(Volume lower bound)* $\text{vol}(S) \geq \text{vol}(S_K)$.

   (c) *(Volume upper bound)* $\text{vol}(S) \leq (5/9)\text{vol}(G)$.

   (d) *(Intersection with C)* $\text{vol}(S \cap C) \geq (9/10)\text{vol}(S)$.

Proof. We begin by observing that regardless of whether the algorithm fails or successfully outputs a cut, the sequence of indices $\{k_0, \ldots, k_f\}$ examined by the algorithm satisfies the conditions of Lemma 4.5. If the algorithm fails during the loop of step 2, then k_f satisfies either $(1+\phi)\tilde{V}(k_f) > \mathrm{vol}(G)$ or $\tilde{V}(k_f) > \mathrm{vol}(\mathrm{Supp}(\tilde{p}))$.

We now consider the PageRank vector $p = \mathrm{pr}(\alpha, v)$ in order to prove claim 1. Lemma 4.3 shows that there is some index K for which $q(K) \geq 1/\gamma\mathrm{vol}(S_K)$, which proves part of the claim. To prove the other part, we assume that $x \geq \mathrm{vol}(S_K)$, and show that the algorithm finds a starting index k_0 that satisfies $\mathrm{vol}(\tilde{S}_{k_0}) \geq \mathrm{vol}(S_K)$. Lemma 4.4 shows that there exists an index j such that $\mathrm{vol}(\tilde{S}_j) \geq \mathrm{vol}(S_K)$ and

$$\tilde{q}(j) \geq q(K) - \epsilon \geq \frac{1}{\gamma\mathrm{vol}(S_K)} - \epsilon.$$

Since $x \geq \mathrm{vol}(S_K)$, we have $\epsilon = 1/2\gamma x \leq 1/2\gamma\mathrm{vol}(S_K)$, which implies the following.

$$\tilde{q}(j) \geq \frac{1}{\gamma\mathrm{vol}(S_K)} - \frac{1}{2\gamma x} \geq \frac{1}{2\gamma\mathrm{vol}(S_K)} \geq \frac{1}{2\gamma\mathrm{vol}(\tilde{S}_j)}.$$

This shows that j may be chosen as a starting index, so the algorithm is assured of choosing some starting index $k_0 \geq j$, which we know will satisfy

$$\mathrm{vol}(\tilde{S}_{k_0}) \geq \mathrm{vol}(S_K) \qquad \text{and} \qquad \tilde{q}(k_0) \geq \frac{1}{2\gamma\mathrm{vol}(\tilde{S}_{k_0})}.$$

This proves Claim 1 of the theorem.

We now move on to the proof of Claim 2. Let k_f be the index of the last vertex considered by the algorithm. We will give a lower bound on $\tilde{q}(k_f)$. Because the rank $\tilde{q}(k_{i+1})$ is not much smaller than $\tilde{q}(k_i)$ at each step, Lemma 4.5 shows that

$$\tilde{q}(k_f) \geq \tilde{q}(k_0) - \frac{(2\alpha/\phi^2)}{\mathrm{vol}(\tilde{S}_{k_0})}.$$

We have set $\alpha = \phi^2/8\gamma$ to ensure that $2\alpha/\phi^2 \leq 1/4\gamma$, and so

$$\begin{aligned}
\tilde{q}(k_f) &\geq \tilde{q}(k_0) - \frac{(2\alpha/\phi^2)}{\mathrm{vol}(\tilde{S}_{k_0})} \\
&\geq \frac{1}{2\gamma\mathrm{vol}(\tilde{S}_{k_0})} - \frac{1}{4\gamma\mathrm{vol}(\tilde{S}_{k_0})} \\
&\geq \frac{1}{4\gamma\mathrm{vol}(\tilde{S}_{k_0})}.
\end{aligned}$$

We now use the assumptions that $v \in C_\alpha$ and $\Phi(C) \leq \alpha/80\gamma$, and apply Lemma 3.6 to give the following bound on $\tilde{p}(\bar{C})$.

$$\tilde{p}(\bar{C}) \leq p(\bar{C}) \leq 2\Phi(C)/\alpha \leq 1/40\gamma.$$

By combining our lower bound on $\tilde{q}(k_f)$ with our upper bound on $\tilde{p}(\bar{C})$, we can bound the intersection of $\tilde{S}_{k_f}$ with $\bar{C}$. First notice that

$$\mathrm{vol}(\tilde{S}_{k_f} \cap \bar{C})\tilde{q}(k_f) \leq p(\tilde{S}_{k_f} \cap \bar{C}) \leq p(\bar{C}).$$

Therefore,

$$\begin{aligned}
\mathrm{vol}(\tilde{S}_{k_f} \cap \bar{C}) &\leq \frac{\tilde{p}(\bar{C})}{\tilde{q}(k_f)} \\
&\leq \frac{4\gamma \mathrm{vol}(\tilde{S}_{k_0})}{40\gamma} \\
&\leq \frac{1}{10}\mathrm{vol}(\tilde{S}_{k_f}).
\end{aligned}$$

This implies

$$\mathrm{vol}(\tilde{S}_{k_f}) \leq \mathrm{vol}(C) + \mathrm{vol}(\tilde{S}_{k_f} \cap \bar{C}) \leq \mathrm{vol}(C) + \frac{1}{10}\mathrm{vol}(\tilde{S}_{k_f}).$$

We now use the fact that $\mathrm{vol}(C) \leq (1/2)\mathrm{vol}(G)$,

$$\mathrm{vol}(\tilde{S}_{k_f}) \leq (10/9)\mathrm{vol}(C) \leq (5/9)\mathrm{vol}(G) \leq \frac{\mathrm{vol}(G)}{1+\phi}.$$

The last step follows by assuming that $\phi \leq 1/3$. We can do so without loss of generality because the approximation guarantee of the theorem is vacuous if $\phi \geq 1/3$.

The equation above shows that the algorithm will not experience a failure caused by $(1+\phi)\mathrm{vol}(\tilde{S}_{k_f}) > \mathrm{vol}(G)$, and our lower bound on $\tilde{q}(k_f)$ ensures that the algorithm will not experience a failure caused by $\mathrm{vol}(\tilde{S}_{k_f}) > \mathrm{vol}(\mathrm{Supp}(\tilde{p}))$. This ensures that the algorithm does not fail and output the message `FAIL: no cut found`. We have already ensured that the algorithm does not fail and output the message `FAIL: no starting index`. Since we have ruled out all of the possible failure conditions, the algorithm must successfully output a set.

We must still prove that the set output by the algorithm satisfies all the properties in claim 2. We have already proved that $\text{vol}(\tilde{S}_{k_f}) \leq (5/9)\text{vol}(G)$, which proves claim (2b). We have proved $\text{vol}(\tilde{S}_{k_f} \cap \bar{C}) \leq \frac{1}{10}\text{vol}(\tilde{S}_{k_f})$, which proves claim (2d). We have proved $\text{vol}(\tilde{S}_{k_f}) \geq \text{vol}(\tilde{S}_{k_0}) \geq \text{vol}(S_K)$, which proves claim (2c).

For the coup de grâce, we will apply the Sharp Drop Lemma. Since the set $\tilde{S}_{k_f}$ was output by the algorithm, that set must have the following property: if k_{f+1} is the smallest index such that $\text{vol}(\tilde{S}_{k_{f+1}}) \geq \text{vol}(\tilde{S}_{k_f})(1+\phi)$, then $\tilde{q}(k_{f+1}) < \tilde{q}(k_f) - \alpha/\phi\text{vol}(\tilde{S}_{k_f})$. We can then apply the Sharp Drop Lemma to show that the number of edges leaving S_{k_f} satisfies $e(\tilde{S}_{k_f}, \bar{\tilde{S}}_{k_f}) < 2\phi\text{vol}(\tilde{S}_{k_f})$. Since $\text{vol}(\tilde{S}_{k_f}) \leq \frac{5}{9}\text{vol}(G)$, we have $\text{vol}(G) - \text{vol}(\tilde{S}_{k_f}) \geq \frac{4}{9}\text{vol}(G) \geq \frac{4}{5}\text{vol}(\tilde{S}_{k_f})$, and so

$$\begin{aligned}
\Phi(\tilde{S}_{k_f}) &= \frac{e(\tilde{S}_{k_f}, \bar{\tilde{S}}_{k_f})}{\min(\text{vol}(\tilde{S}_{k_f}), \text{vol}(G) - \text{vol}(\tilde{S}_{k_f}))} \\
&\leq \frac{2\phi\text{vol}(\tilde{S}_{k_f})}{(4/5)\text{vol}(\tilde{S}_{k_f})} \\
&\leq 3\phi.
\end{aligned}$$

This proves the approximation guarantee of claim (2a).

□

4.4 Acknowledgement

Chapter 4 contains material that appeared in the following article.
R. Andersen and F. Chung. *Detecting sharp drops in PageRank and a simplified local partitioning algorithm.* To appear in *The 4th Annual Conference on Theory and Applications of Models of Computation (TAMC'07).* The dissertation author was the primary investigator and author of this paper.

5 An Intuitive Proof of Cheeger's Inequality

In this chapter, we present an intuitive proof of Cheeger's inequality, using ideas from the previous chapters. Cheeger's inequality relates the conductance of a graph to its spectral gap. We will prove only the hard half of the inequality, which is the theoretical basis for spectral partitioning algorithms. We will show that for any eigenvector φ of the random walk matrix W that has eigenvalue $1-\lambda$ for some $\lambda > 0$, a sweep over φ produces a cut that satisfies

$$\Phi(\varphi) \leq 2\sqrt{\lambda}.$$

The other easier half of the inequality shows that $\lambda \leq 2\Phi_G$, where Φ_G is the smallest conductance of any set in the graph. Combining these two halves we learn that $\Phi(\varphi) \leq \sqrt{8\Phi_G}$. Our proof gives an interesting characterization of a sweep set that attains the desired conductance.

5.1 Related work

The idea of using an eigenvector to partition a graph was first described by Fiedler [20], and later by Donath and Hoffman [16, 16]. Cheeger's inequality for graphs was proved by Jerrum and Sinclair [31], who used it as a tool for bounding the mixing rate of Markov chains. A similar result was proved independently by Alon and Milman [2]. Milena Mihail gave a combinatorial proof of Cheeger's inequality that is useful for spectral partitioning, because it shows a good cut can be obtained by sweeping over any vector of the combinatorial Laplacian matrix

that is orthogonal to the all-ones vector and has a small Rayleigh quotient [42]. The form of Cheeger's inequality that we consider, which uses the random walk matrix or normalized Lapalacian matrix, and which applies to graphs that are not necessarily regular, was proved by Fan Chung [14].

5.2 A proof of Cheeger's inequality

Let $W = D^{-1}A$ be the transition matrix of a random walk in a connected undirected graph. The matrix W has the trivial right eigenvector $\vec{1}$ with eigenvalue 1, and a corresponding left eigenvector ψ_V with eigenvalue 1. In matrix notation, we have

$$\psi_V W = \psi_V \qquad \text{and} \qquad W\vec{1} = \vec{1}.$$

If φ is a left eigenvector of W with eigenvalue $1-\lambda$, where $\lambda > 0$, then φ is orthogonal to $\vec{1}$. If the graph is connected and nonbipartite, then all left eigenvectors of W other than ψ_V (up to scalar multiplication), have eigenvalues strictly less than 1. We do not require the graph be nonbipartite.

Our proof has a simple premise. We sort the vertices of the graph from highest to lowest values of $\varphi(v)/d(v)$. Using techniques from the previous sections, we show that for some $x \in [0, 2m]$, the value $\varphi W[x]$ is smaller than $\varphi[x]$ by a factor of roughly $1 - \Phi(\varphi)^2$. On the other hand, the eigenvector equation shows that $\varphi W[x] = (1-\lambda)\varphi[x]$ for all x. Relating these two bounds yields the hard part of Cheeger's inequality.

We first present a preliminary Lemma that describes the shape of the curve $\varphi[x]$.

Lemma 5.1. *Let φ be a left eigenvector of $W = D^{-1}A$ whose associated eigenvalue is $1-\lambda$ for some $\lambda \in (0,1)$. For any set S of vertices,*

$$(1-\lambda/2)\varphi(S) \leq \frac{1}{2}\varphi\left(\text{in}(S) \cap \text{out}(S)\right) + \frac{1}{2}\varphi\left(\text{in}(S) \cup \text{out}(S)\right).$$

Furthermore, for each $j \in [1, n-1]$, and for each level set $S_j = S_j^{\varphi}$, we have

$$(1-\lambda/2)\varphi(S_j) \leq \frac{1}{2}\varphi\left[\text{vol}(S_j) - |\partial(S_j)|\right] + \frac{1}{2}\varphi\left[\text{vol}(S_j) + |\partial(S_j)|\right].$$

Proof of Lemma 5.1. The eigenvector φ satisfies the following equation for any set S of vertices,

$$(1-\lambda)\varphi(S) = \varphi W(S).$$

Applying Lemma 2.1,

$$(1-\lambda)\varphi(S) \leq \varphi\left(\text{in}(S) \cap \text{out}(S)\right) + \varphi\left(\text{in}(S) \cup \text{out}(S)\right) - \varphi(S).$$

By adding the term $(1-\lambda)\varphi(S)$ to both sides and then dividing by $(2-\lambda)/2$, we obtain

$$\begin{aligned}(2-\lambda)\varphi(S) &\leq \varphi\left(\text{in}(S) \cap \text{out}(S)\right) + \varphi\left(\text{in}(S) \cup \text{out}(S)\right) \\ (1-\lambda/2)\varphi(S) &\leq \frac{1}{2}\varphi\left(\text{in}(S) \cap \text{out}(S)\right) + \frac{1}{2}\varphi\left(\text{in}(S) \cup \text{out}(S)\right).\end{aligned}$$

That proves the first part of the Lemma. We now consider a level set $S_j = S_j^{\varphi}$. Using the same argument as in Lemma 3.2, we obtain

$$\begin{aligned}(1-\lambda/2)\varphi(S_j) &\leq \frac{1}{2}\varphi\left(\text{in}(S_j) \cap \text{out}(S_j)\right) + \frac{1}{2}\varphi\left(\text{in}(S_j) \cup \text{out}(S_j)\right) \\ &\leq \frac{1}{2}p\left[|\text{in}(S_j) \cap \text{out}(S_j)|\right] + \frac{1}{2}p\left[|\text{in}(S_j) \cup \text{out}(S_j)|\right] \\ &\leq \frac{1}{2}p\left[\text{vol}(S_j) - |\partial(S_j)|\right] + \frac{1}{2}p\left[\text{vol}(S_j) + |\partial(S_j)|\right].\end{aligned}$$

□

We now prove the hard half of Cheeger's inequality. Notice that the theorem provides a characterization of a level set with small conductance.

Theorem 5.2. *Let φ be a left eigenvector of $W = D^{-1}A$ whose associated eigenvalue is $1-\lambda$ for some $\lambda \in (0,1)$, and let $S_j = S_j^{\varphi}$. Given an index j in $[1, n-1]$, we define*

$$f(j) = \frac{\varphi[\text{vol}(S_j)]}{\sqrt{\min(\text{vol}(S_j), 2m - \text{vol}(S_j))}}.$$

If J is a index in $[1, n-1]$ that achieves the maximum value $f(J) = \max_{j\in[1,n-1]} f(j)$, then the level set S_J satisfies $\Phi(S_J)^2 \leq 4\lambda$.

Proof. Since φ is a left eigenvector of W whose eigenvalue is strictly less than 1, we know φ is orthogonal to $\vec{1}$, and so $\varphi[0] = \varphi[2m] = 0$. Since φ is not uniformly equal to zero, there exists a smallest real number $C(\varphi) > 0$ such that

$$\varphi[x] \leq C(\varphi) \cdot \sqrt{\min(x, 2m - x)} \qquad \text{for all } x \in [0, 2m].$$

Since $\varphi[x]$ is piecewise linear between the set of points x where $x = \text{vol}(S_j)$ for some index j, and since $\sqrt{\min(x, 2m-x)}$ is concave, this bound is attained by some level set S_j. In particular, it is attained by the level set S_J that maximizes $f(j)$. We may assume without loss of generality that the level set S_J has at most half the volume of the entire graph, since otherwise we could reverse the ordering of the vertices by considering the eigenvector $(-1) \cdot \varphi$. By our choice of $C(\varphi)$ and S_J, we know $\varphi(S_J)$ satisfies

$$\varphi(S_J) = C(\varphi) \cdot \sqrt{\text{vol}(S_J))}.$$

On the other hand, Lemma 5.1 implies the following.

$$\begin{aligned}
(1 - \lambda/2)\varphi(S_J) &\leq \frac{1}{2} p\left[\text{vol}(S_J) - |\partial(S_J)|\right] + \frac{1}{2} p\left[\text{vol}(S_J) + |\partial(S_J)|\right] \\
&\leq \frac{1}{2}\varphi[\text{vol}(S_J) - \Phi(S_J)\text{vol}(S_J)] + \frac{1}{2}\varphi[\text{vol}(S_J) + \Phi(S_J)\text{vol}(S_J)] \\
&\leq \frac{1}{2} C(\varphi)\sqrt{\text{vol}(S_J) - \Phi(S_J)\text{vol}(S_J)} + \\
&\qquad \frac{1}{2} C(\varphi)\sqrt{\text{vol}(S_J) + \Phi(S_J)\text{vol}(S_J)} \\
&= \frac{1}{2}\left(\sqrt{1 - \Phi(S_J)} + \sqrt{1 + \Phi(S_J)}\right) C(\varphi)\sqrt{\text{vol}(S_J)} \\
&\leq (1 - \Phi(S_J)^2/8) \cdot C(\varphi)\sqrt{\text{vol}(S_J)}.
\end{aligned}$$

The last line follows from Proposition 2.7. We have proved that

$$(1 - \lambda/2)C(\varphi)\sqrt{\text{vol}(S_J)} = (1 - \lambda/2)\varphi(S_J) \leq (1 - \Phi(S_J)^2/8) \cdot C(\varphi)\sqrt{\text{vol}(S_J)},$$

which implies $\Phi(S_J)^2 \leq 4\lambda$. □

5.3 Variations

In Theorem 5.2, we used a constant multiple of the function $\sqrt{x}$ to place an upper bound on $\varphi[x]$. The same argument will apply with any bounding function $f(x)$ in place of $\sqrt{x}$, provided that f is concave and increasing, $f(0) = 0$, and $f(x) > 0$ when $x > 0$. We now consider bounding functions of the form x^b for $b \in (0,1)$. For any $b \in (0,1)$, and for any $\phi \in [0,1]$,

$$(1+\phi)^b = 1 + b\phi + \frac{b(b-1)}{2!}\phi^2 + \frac{b(b-1)(b-2)}{3!}\phi^3 + \ldots,$$
$$(1-\phi)^b = 1 - b\phi + \frac{b(b-1)}{2!}\phi^2 - \frac{b(b-1)(b-2)}{3!}\phi^3 + \ldots.$$

The odd terms of the two series cancel, and the even terms are negative, so we have

$$\frac{1}{2}\left((1-\phi)^b + (1+\phi)^b\right) \leq 1 - \frac{1}{2}b(1-b)\phi^2.$$

The best constant is obtained when $b = 1/2$, which is a good reason to use $\sqrt{x}$. However, there is still a reason to use a different bounding function, since we obtain an upper bound on the conductance of a different cut. The cut found using $b < 1/2$ is no smaller than the cut found using $b = 1/2$, and may be larger.

Corollary 5.3. *Let φ be a left eigenvector of $W = D^{-1}A$ whose associated eigenvalue is $1-\lambda$ for some $\lambda \in (0,1)$, and let $S_j = S_j^{\varphi}$. Let $f(x) = x^b$ for some $b \in (0,1)$, and define*

$$h(j) = \frac{\varphi[\mathrm{vol}(S_j)]}{f(\min(\mathrm{vol}(S_j), 2m - \mathrm{vol}(S_j)))}.$$

If J is a index in $[1, n-1]$ that achieves the maximum value $h(J) = \max_{j\in[1,n-1]} h(j)$, then the level set S_J satisfies $\Phi(S_J)^2 \leq \frac{1}{b(1-b)}\lambda$.

Proof. This follows from the same argument as Theorem 5.2, by replacing $\sqrt{x}$ with $f(x)$. □

We can also find a cut with small conductance by bounding $\varphi[x]$ with a multiple of the function $\log(1+x)$. With this bounding function, there is a stronger tendency to find a cut with good balance than with the bounding function $\sqrt{x}$.

Theorem 5.4. *Consider the following modified definition of conductance,*

$$\Phi'(S) = \frac{|\partial(S)|}{1 + \min(\mathrm{vol}(S), 2m - \mathrm{vol}(S))},$$

which is always within a factor of 2 *of the usual conductance in an unweighted graph. Let* φ *be a left eigenvector of* $W = D^{-1}A$ *whose associated eigenvalue is* $1 - \lambda$ *for some* $\lambda \in (0, 1)$*, and let* $S_j = S_j^{\varphi}$*. Given an index* j *in* $[1, n-1]$*, we define*

$$h(j) = \frac{\varphi[\mathrm{vol}(S_j)]}{\log(1 + \min(\mathrm{vol}(S_j), 2m - \mathrm{vol}(S_j)))}.$$

If J *is a index in* $[1, n-1]$ *that achieves the maximum value* $h(J) = \max_{j \in [1,n-1]} h(j)$*, then the corresponding level set satisfies* $\Phi'(S_J)^2 \leq \lambda \log(1 + \mathrm{vol}(S_J))$.

Proof. The function $\log(1 + x)$ is 0 when $x = 0$, and we know $\varphi[0] = \varphi[2m] = 0$, so there exists a smallest real number $C(\varphi) > 0$ such that

$$\varphi[x] \leq C(\varphi) \cdot \log(1 + \min(x, 2m - x)) \qquad \text{for all } x \in [0, 2m].$$

Since $\log(1 + x)$ is concave, this bound is attained by the level set that maximizes $f(j)$. We may assume without loss of generality that the level set S_J has at most half the volume of the entire graph. Since S_J attains this bound, we know $\varphi(S_J)$ exactly,

$$\varphi(S_J) = C(\varphi) \cdot \log(1 + \mathrm{vol}(S_J)).$$

As before, we use Lemma 5.1 to place an upper bound on $\varphi(S_J)$. We will use the properties of $\log(1 + x)$ to show that the bound decreases by an additive

amount rather than a multiplicative factor.

$$\begin{aligned}
(1-\lambda/2)\varphi(S_J) &\leq \frac{1}{2}\varphi[\mathrm{vol}(S_J) - |\partial(S_J)|] + \frac{1}{2}\varphi[\mathrm{vol}(S_J) + |\partial(S_J)|] \\
&= \frac{1}{2}C(\varphi)\log(1 + \mathrm{vol}(S_J) - |\partial(S_J)|) \\
&\quad + \frac{1}{2}C(\varphi)\log(1 + \mathrm{vol}(S_J) + |\partial(S_J)|) \\
&= \frac{1}{2}C(\varphi)\log(1 + \mathrm{vol}(S_J) - \Phi'(S_J)(1 + \mathrm{vol}(S_J))) \\
&\quad + \frac{1}{2}C(\varphi)\log(1 + \mathrm{vol}(S_J) + \Phi'(S_J)(1 + \mathrm{vol}(S_J))) \\
&= \frac{1}{2}C(\varphi)\log\left((1 + \mathrm{vol}(S_J))(1 - \Phi'(S_J))\right) \\
&\quad + \frac{1}{2}C(\varphi)\log\left((1 + \mathrm{vol}(S_J))(1 + \Phi'(S_J))\right) \\
&= C(\varphi)\log(1 + \mathrm{vol}(S_J)) + C(\varphi)\log((1 - \Phi'(S_J))(1 + \Phi'(S_J))) \\
&= C(\varphi)\log(1 + \mathrm{vol}(S_J)) + C(\varphi)\log(1 - \Phi'(S_J)^2)).
\end{aligned}$$

We have shown that

$$(1-\lambda/2)C(\varphi)\log(1 + \mathrm{vol}(S_J)) \leq C(\varphi)\log(1 + \mathrm{vol}(S_J)) + C(\varphi)\log(1 - \Phi'(S_J)^2)).$$

Therefore,

$$-(\lambda/2)\log(1 + \mathrm{vol}(S_J)) \leq \frac{1}{2}\log(1 - \Phi'(S_J)^2)) \leq -\Phi'(S_J)^2/2,$$

and we learn that $\Phi'(S_J)^2 \leq \lambda \log(1 + \mathrm{vol}(S_J))$. □

We remark that the cut S_J described in the theorem above must satisfy $|\varphi(S_J)| \geq |\varphi|_1/\log(1+m)$, so if $\mathrm{vol}(S_J)$ is much smaller than $1/\log(m)$ then the eigenvector itself is somewhat unbalanced.

6 Generalizing Local Partitioning to Directed Graphs

In this chapter, we generalize a basic local partitioning result to to strongly connected directed graphs. We prove that a sweep over a personalized PageRank vector in a directed graph produces a set with small conductance, provided the notion of conductance is generalized appropriately. The main difficulty we encounter is that the appropriate generalization of conductance depends on the stationary distribution π of a random walk in the directed graph, which is more complicated than the stationary distribution in an undirected graph.

Directed graphs that arise in practice are typically not strongly connected, so we also propose one possible way to apply our results to an arbitrary directed graph. We describe how to apply our algorithm to the PageRank Markov chain of a directed graph, which is ergodic even when the underlying graph is not strongly connected. Our local partitioning algorithm has a natural interpretation when applied to the PageRank Markov chain: it computes a global PageRank vector with a uniform starting distribution, and a personalized PageRank vector with a single starting vertex, and ranks the vertices according to the ratio of their entries in the personalized PageRank vector and global PageRank vector. By sorting the vertices of the graph according to this ratio, our algorithm finds a set with small conductance in the PageRank Markov chain.

6.1 Related work

The results of Fill [21] and Mihail [42] bound the mixing rate of an asymmetric ergodic Markov chain in terms of its conductance. The result of Lovász and Simonovits, which we used to analyze our local partitioning algorithm for undirected graphs, also applies to the more general case of asymmetric ergodic Markov chains [40], and this will be the basis for our directed local partitioning algorithm. The idea of comparing contributions from different PageRank vectors has been used previously in spam identification [28].

6.2 Preliminaries for directed graphs

Let G be a directed graph, consisting of a vertex set V and a set of directed edges E, each of which is an ordered pair (u, v) of vertices from V. Let n be the number of vertices, and m be the number of directed edges. We write $d_{out}(v)$ for the outdegree of a vertex v.

The adjacency matrix $A = A(G)$ is the $n \times n$ matrix where $A_{i,j} = 1$ if and only if there is a directed edge (v_i, v_j), given some fixed ordering $v_1, \ldots, v_n$ of the vertices. The outdegree matrix $D = D(G)$ is the $n \times n$ diagonal matrix where $D_{i,i} = d_{out}(v_i)$.

For a given directed graph, we will consider several different Markov chains. For our purposes, a Markov chain M is the matrix of a random walk on a weighted directed graph on the vertex set V. Equivalently, it is an $n \times n$ probability matrix, for which the sum of each row is 1. A Markov chain is said to be *ergodic* if the corresponding random walk converges to a unique stationary distribution. That is, if there exists a vector π that is nonzero at each vertex, that satisfies $\pi = \pi M$, and such that for every vertex v in V, we have $\lim_{t\to\infty} 1_v M^t = \pi$. The vector π is the *stationary distribution* of M. A Markov chain is ergodic if and only if it is a random walk on a graph that is strongly connected and aperiodic.

Let p be a probability distribution on the vertices of V, and let M be a Markov chain. For each vertex u, we define the outdegree of u in M to be

$$d_{out}(u) = \sum_{v \in V} M(u, v).$$

For each set $S \subseteq V$, we define the sum of p over S to be

$$p(S) = \sum_{u \in S} p(u),$$

For each edge (u, v), we define

$$p(u, v) = p(u) \frac{M(u, v)}{d_{out}(u)}.$$

This is the amount of probability that moves from u to v when a step of the Markov chain is applied to the vector p. For each set A of directed edges, we define

$$p(A) = \sum_{(u,v) \in A} p(u, v),$$

which is the total amount of probability moving over the set of directed edges. This notation is overloaded, but it is unambiguous if the type of input is known.

6.2.1 Conductance and sweeps

We now assume that the Markov chain M is ergodic with a unique stationary distribution π, and define the generalizations of conductance, of the sweep procedure, and of the potential function $p[x]$, all of which are now normalized by π.

Given a set S of states, we define $\bar{\pi}(S) = min(\pi(S), 1 - \pi(S))$ to be the measure of the smaller side of the partition induced by S, and define the outgoing edge border $\partial(S)$ as follows,

$$\partial(S) = \{(u, v) \in E \mid u \in S \text{ and } v \in \bar{S}\}.$$

Definition 6.1. Let M be an ergodic Markov chain, and let π be its unique stationary distribution. We define the M-conductance $\Phi_M(S)$ of a set of vertices S to be

$$\Phi_M(S) = \frac{\pi(\partial(S))}{\bar{\pi}(S)}.$$

Definition 6.2. Given a probability distribution p on the states of an ergodic Markov chain M with stationary distribution π, let $v_1, \ldots, v_n$ be an ordering of the vertices such that

$$\frac{p(v_i)}{\pi(v_i)} \geq \frac{p(v_{i+1})}{\pi(v_{i+1})}.$$

For each integer j in $\{1, \ldots, n\}$, we define $S_j^p = \{v_1, \ldots, v_j\}$ to be the set containing the top j vertices in this ordering. We define $\Phi_M(p)$ to be the smallest M-conductance among the sets $S_1^p, \ldots S_n^p$,

$$\Phi_M(p) = \min_{j \in [1,n]} \Phi_M(S_j^p).$$

Definition 6.3. Let M be an ergodic Markov chain with stationary distribution π, and let p be a probability distribution on the states of Markov chain. We define $p\left[x\right]$ to be the unique function from $[0,1]$ to $[0,1]$ such that

$$p\left[\pi(S_j^p)\right] = p(S_j^p) \quad \text{for each } j \in [0,n],$$

and such that $p\left[x\right]$ is piecewise linear between these points.

Proposition 6.4. *We have the following facts about the function $p\left[x\right]$.*

1. *The function $p\left[x\right]$ is concave.*

2. *For any set S of vertices,*

$$p(S) \leq p\left[\pi(S)\right].$$

3. *For any set of directed edges A, we have*

$$p(A) \leq p\left[\pi(A)\right].$$

These facts are proved in [40], and are not difficult to verify.

6.2.2 Global PageRank and personalized PageRank

Definition 6.5. Given a Markov chain M, the PageRank vector $\mathrm{pr}_M(\alpha, s)$ is defined to be the unique solution of the linear system

$$\mathrm{pr}_M(\alpha, s) = \alpha s + (1-\alpha)\mathrm{pr}_M(\alpha, s)M. \tag{6.1}$$

Here, α is a constant in $(0, 1]$ called the *teleport probability*, s is a probability distribution called the *starting vector*.

We will use the following basic facts about PageRank.

Proposition 6.6. *For any Markov chain M, any starting vector s, and any constant α in $(0, 1]$, there is a unique vector $\mathrm{pr}_M(\alpha, s)$ satisfying*

$$\mathrm{pr}_M(\alpha, s) = \alpha s + (1-\alpha)\mathrm{pr}_M(\alpha, s)M.$$

Proposition 6.7. *For any Markov chain M and any fixed value of α in $(0, 1]$, there is a linear transformation R_α such that $\mathrm{pr}_M(\alpha, s) = sR_\alpha$. Furthermore, R_α is given by the matrix*

$$R_\alpha = \alpha I + \alpha \sum_{t=1}^{\infty} (1-\alpha)^t M^t. \tag{6.2}$$

The proofs are the same as those given in Chapter 2.

We let $\psi = \frac{1}{n}1_V$ be the uniform distribution. If a PageRank vector has ψ for its starting vector, we call it a *global PageRank vector*. If a PageRank vector has for its starting vector the indicator vector 1_v, with all probability on a single vertex v, we call it a *personalized PageRank vector*, and use the shorthand notation $\mathrm{pr}_M(\alpha, v) = \mathrm{pr}_M(\alpha, 1_v)$.

There are a glut of algorithms for computing global PageRank and personalized PageRank, so we will treat the computation of PageRank as a primitive operation. We assume we have the following two black-box algorithms,

- `GlobalPR`(M, α) computes the global PageRank vector $\mathrm{pr}_M(\alpha, \psi)$.
- `LocalPR`(M, α, v) computes the personalized PageRank vector $\mathrm{pr}_M(\alpha, v)$.

We make the distinction between these two black boxes because personalized PageRank can be computed more efficiently that global PageRank. One may use for `LocalPR` any of the algorithms described by Jeh and Widom [30], Berkhin [8], Sarlos [44], or Gleich [26], each of which can compute an approximation of the personalized PageRank vector $\mathrm{pr}_M(\alpha, v)$ by examining only a small fraction of the input graph near v, provided that M is a sparse matrix. We will endeavor to use `LocalPR` instead of `GlobalPR` as much as possible.

6.3 Local partitioning for ergodic Markov chains

We now state the main theorem of the paper, which shows that a sweep over a personalized PageRank vector in an ergodic Markov chain M can produce a set with small M-conductance. This is a natural generalization of the theorem we proved for undirected graphs in Chapter 3.

Theorem 6.8. *Let M be an ergodic Markov chain with stationary distribution π. Let S be a set of vertices such that $\pi(S) \leq \frac{1}{2}$ and $\Phi_M(S) \leq \alpha/16$, for some constant α. If v is sampled from S according to the probability distribution $\pi(v)/\pi(S)$, then with probability at least $1/2$, we have $\Phi_M(\mathrm{pr}_M(\alpha, v)) = O(\sqrt{\alpha \log |S|})$.*

The proof of the theorem is given at the end of this section. Here is outline of how we will proceed. Given a personalized PageRank vector $p = \mathrm{pr}_M(\alpha, s)$ in an ergodic Markov chain M, we place an upper bound on $p[x]$ that depends on α and $\Phi(p)$, and place a lower bound on $p[\pi(S)]$ that depends on the conductance of a certain set S near the starting vertex. These upper and lower bounds will be combined to show that $\Phi(p)$ is small. We establish the upper and lower bounds in the following lemmas.

Lemma 6.9. *Let M be an ergodic Markov chain with stationary distribution π, let $p = \mathrm{pr}_M(\alpha, v)$ be a personalized PageRank vector in M, and let $\phi = \Phi_M(p)$ be the smallest M-conductance found by the sweep over p. Then,*

$$p[x] \leq x + \alpha t + \left(1 - \tfrac{\phi^2}{8}\right)^t \sqrt{x/\pi(v)} \quad \text{for all } x \in [0,1] \text{ and all } t \geq 0.$$

Lemma 6.10. *Let M be an ergodic Markov chain with stationary distribution π, let S be a set of vertices, and let v be a vertex sampled from S according to the probability distribution $\pi(v)/\pi(S)$. With probability at least $3/4$,*

$$\mathrm{pr}_M(\alpha, v)(S) \geq 1 - 4\frac{\Phi_M(S)}{\alpha}.$$

These two lemmas will be proved in the following section. We use them now to derive the main theorem.

Proof of Theorem 6.8. Let $p = \mathrm{pr}_M(\alpha, v)$ and let $\Phi(p) = \phi$. If v is sampled from S with probability $\pi(v)/\pi(S)$, Lemma 6.10 implies the following bound holds with probability at least $3/4$,

$$\mathrm{pr}_M(\alpha, v)(S) \geq 1 - 4\frac{\Phi_M(S)}{\alpha} \geq 1 - 4\frac{\alpha/16}{\alpha} \geq 3/4. \tag{6.3}$$

We will now show that with probability at least $3/4$,

$$\frac{\pi(v)}{\pi(S)} \geq \frac{1}{4|S|}. \tag{6.4}$$

To see this, consider the set of vertices S' in S such that $\pi(v) \geq \frac{\pi(S)}{4|S|}$. Clearly $\pi(S \setminus S') < \pi(S)/4$, which shows that $\pi(S') > (3/4)\pi(S)$.

The probability that the two events described above in (6.3) and (6.4) both occur is at least $1/2$. We will assume for the rest of the proof that both events hold.

Lemma 6.9 gives us the following upper bound on $\mathrm{pr}_M(\alpha, v)(S)$.

$$\begin{aligned}
\mathrm{pr}_M(\alpha, v)(S) &\leq \mathrm{pr}_M(\alpha, v)[\pi(S)] \\
&\leq (4/3)\pi(S) + \alpha T + \left(1 - \frac{\phi^2}{8}\right)^T \sqrt{\pi(S)/\pi(v)} \\
&\leq (4/3)(1/2) + \alpha T + \left(1 - \frac{\phi^2}{8}\right)^T \sqrt{4|S|}.
\end{aligned}$$

If we let $T = (72/\phi^2) \ln 24\sqrt{4|S|}$, then

$$\mathrm{pr}_M(\alpha, v)(S) \leq 2/3 + \alpha T + 1/24.$$

This contradicts our lower bound from (6.3) if $\alpha < 1/25T$, so we have shown that $\alpha \geq 1/25T$, which implies the following bound,

$$\phi \leq \sqrt{72 \cdot 25 \cdot \alpha \ln 24\sqrt{4|S|}} = O(\sqrt{\alpha \log |S|}).$$

□

6.4 Mixing bounds for personalized PageRank

In this section, we prove upper and lower bounds on the curve $p[x]$ of a PageRank vector $p = \mathrm{pr}_M(\alpha, s)$.

Lemma 6.11. *Let M be an ergodic Markov chain with stationary distribution π, let $p = \mathrm{pr}_M(\alpha, v)$ be a personalized PageRank vector in M, and let $S_j = S_j^p$. For each $j \in [1, n-1]$, we have*

$$p[\pi(S_j)] \leq \frac{\alpha}{2-\alpha} s\left[\pi(S_j)\right] + \left(1 - \frac{\alpha}{2-\alpha}\right)\left(p\left[\pi(S_j) + \pi(\partial(S_j))\right] + p\left[\pi(S_j) - \pi(\partial(S_j))\right]\right).$$

Proof of Lemma 6.11. For any set S of vertices, we define the set of directed edges whose heads are in S,

$$\mathrm{in}(S) = \{(u, v) \in E \mid v \in S\},$$

and the set of edges whose tails are in S,

$$\mathrm{out}(S) = \{(u, v) \in E \mid u \in S\}.$$

The following describes the amount of probability from pM on a set S, in terms of the amount of probability moving across the edges in the sets $\mathrm{in}(S)$ and $\mathrm{out}(S)$.

$$\begin{aligned} pM(S) &= p(\mathrm{in}(S)) \\ &= p\left(\mathrm{in}(S) \cap \mathrm{out}(S)\right) + p\left(\mathrm{in}(S) \setminus \mathrm{out}(S)\right) \\ &= p\left(\mathrm{in}(S) \cap \mathrm{out}(S)\right) + p\left(\mathrm{in}(S) \cup \mathrm{out}(S)\right) - p(S). \end{aligned}$$

We will now calculate the total measure of the edges in the sets $(\text{in}(S) \cup \text{out}(S))$ and $(\text{in}(S) \cap \text{out}(S))$. The following holds because π is the stationary distribution of M,

$$\pi(\text{in}(S)) = \pi(\text{out}(S)) = \pi(S).$$

It is not hard to observe the following two equalities.

$$\pi(\text{in}(S) \cup \text{out}(S)) + \pi(\text{in}(S) \cap \text{out}(S)) = 2\pi(S),$$

$$\pi(\text{in}(S) \cup \text{out}(S)) - \pi(\text{in}(S) \cap \text{out}(S)) = 2\pi(\partial(S)).$$

Solving the system of equations above yields the following.

$$\pi\left(\text{in}(S) \cup \text{out}(S)\right) = \pi(S) + \pi(\partial(S)),$$

$$\pi\left(\text{in}(S) \cap \text{out}(S)\right) = \pi(S) - \pi(\partial(S)).$$

Now let $p = \text{pr}_M(\alpha, s)$ be a personalized PageRank vector. For any set S of vertices, we have

$$\begin{aligned} p(S) &\leq \alpha s(S) + (1-\alpha) pW(S) \\ &\leq \alpha s(S) + (1-\alpha)\left(p\left(\text{in}(S) \cap \text{out}(S)\right) + p\left(\text{in}(S) \cup \text{out}(S)\right) - p(S)\right). \end{aligned}$$

By adding the term $(1-\alpha)p(S)$ to both sides and then dividing by $2-\alpha$, we obtain

$$p(S) \leq \frac{\alpha}{2-\alpha} s(S) + (1 - \frac{\alpha}{2-\alpha})\left(\frac{1}{2} p\left(\text{in}(S) \cap \text{out}(S)\right) + \frac{1}{2} p\left(\text{in}(S) \cup \text{out}(S)\right)\right).$$

Now let $S_j = S_j^p$, and recall from Proposition 6.4 that $p\left[\pi(S_j)\right] = p(S_j)$ for any integer $j \in [0, n]$, and that for any set of directed edges A, we have the bound

$p(A) \leq p\left[\pi(A)\right]$.

$$\begin{aligned}
p\left[\pi(S_j)\right] &= p(S_j) \\
&= \frac{\alpha}{2-\alpha} s(S_j) + \\
&\quad (1 - \frac{\alpha}{2-\alpha}) \left(\frac{1}{2} p\left(\text{in}(S_j) \cap \text{out}(S_j)\right) + \frac{1}{2} p\left(\text{in}(S_j) \cup \text{out}(S_j)\right) \right) \\
&\leq \frac{\alpha}{2-\alpha} s\left[\pi(S_j)\right] + \\
&\quad (1 - \frac{\alpha}{2-\alpha}) \left(\frac{1}{2} p\left[\pi\left(\text{in}(S_j) \cap \text{out}(S_j)\right)\right] + \frac{1}{2} p\left[\pi\left(\text{in}(S_j) \cup \text{out}(S_j)\right)\right] \right) \\
&\leq \frac{\alpha}{2-\alpha} s\left[\pi(S_j)\right] + \\
&\quad (1 - \frac{\alpha}{2-\alpha}) \left(\frac{1}{2} p\left[\pi(S_j) - \pi(\partial(S_j))\right] + \frac{1}{2} p\left[\pi(S_j) + \pi(\partial(S_j))\right] \right).
\end{aligned}$$

□

Lemma 6.12. *For any ergodic Markov chain M, any starting vector s, any value $\alpha \in (0, 1]$, and any $x \in [0, 1]$, we have*

$$\text{pr}_M(\alpha, s)\left[x\right] \leq s\left[x\right].$$

Proof of Lemma 6.12. If we let $p = \text{pr}(\alpha, s)$, Lemma 6.11 implies that for each $j \in [1, n-1]$,

$$\begin{aligned}
p\left[\pi(S_j^p)\right] &\leq \frac{\alpha}{2-\alpha} s\left[\pi(S_j^p)\right] + \\
&\qquad (1 - \frac{\alpha}{2-\alpha}) \left(\frac{1}{2} p\left[\pi(S_j^p) - \pi(\partial(S_j^p))\right] + \frac{1}{2} p\left[\pi(S_j^p) + \pi(\partial(S_j^p))\right] \right) \\
&\leq \frac{\alpha}{2-\alpha} s\left[\pi(S_j^p)\right] + (1 - \frac{\alpha}{2-\alpha}) p\left[\pi(S_j^p)\right].
\end{aligned}$$

The last line follows from the concavity of $p\left[k\right]$. This implies that $p\left[\pi(S_j^p)\right] \leq s\left[\pi(S_j^p)\right]$ for each $j \in [1, n-1]$. The same equation then holds for all $x \in [0, 1]$, because $s\left[x\right]$ is concave and $p\left[x\right]$ is linear between the points $\pi(S_j^p)$ and $\pi(S_{j+1}^p)$. □

We now prove the two lemmas we used in the previous section.

Proof of Lemma 6.9. We define the function

$$f_t(x) = \alpha t + \left(1 - \frac{\phi^2}{8}\right)^t \sqrt{x/\pi(v)}.$$

We will prove by induction that the following inequality holds for all $t \geq 0$.

$$p\left[x\right] \leq \frac{4}{3}x + f_t(x) \qquad \text{for all } x \in [0,1].$$

We call this inequality I_t.

To prove that the base case I_0 holds, notice for any value of x in the interval $[0,1]$, we have $p[x] = \mathrm{pr}_M(\alpha, v)[x] \leq 1_v[x]$, by Lemma 6.12. This implies

$$p\left[x\right] \leq 1_v[x] \leq \min(1, x/\pi(v)) \leq x + \sqrt{x/\pi(v)},$$

which shows that I_0 holds.

We now assume that I_t holds, and prove that I_{t+1} holds. For each $j \in [0,n]$, let $x_j = \pi(S_j^p)$. It suffices to show that I_{t+1} holds at the points $x_0, \ldots, x_n$, because $p\left[x\right]$ is piecewise linear between these points, and $f_{t+1}(x)$ is concave.

The inequality I_{t+1} holds trivially when $x = 0$, and also when $x \geq 3/4$, so it suffices to consider an arbitrary index j such that $j > 1$ and $\pi(S_j) \leq 3/4$. Because $\pi(S_j) \leq 3/4$, we have $\bar{\pi}(S_j) \geq (1/3)\pi$, and so

$$\pi(\partial(S_j)) = \Phi(S_j)\bar{\pi}(S_j) \geq (1/3)\Phi(S_j)\pi(S_j) \geq (1/3)\phi x_j.$$

We now apply Lemma 6.11.

$$\begin{aligned}
p\left[\pi(S_j)\right] &\leq \frac{\alpha}{2-\alpha} s\left[\pi(S_j)\right] + \\
&\qquad (1 - \frac{\alpha}{2-\alpha})\left(p\left[\pi(S_j) - \pi(\partial(S_j))\right] + p\left[\pi(S_j) + \pi(\partial(S_j))\right]\right) \\
&\leq \frac{\alpha}{2-\alpha} + \left(\frac{1}{2}p\left[\pi(S_j) - \pi(\partial(S_j))\right] + \frac{1}{2}p\left[\pi(S_j) + \pi(\partial(S_j))\right]\right) \\
&\leq \alpha + \left(\frac{1}{2}p\left[x_j - (1/3)\phi x_j\right] + \frac{1}{2}p\left[x_j + (1/3)\phi x_j\right]\right).
\end{aligned}$$

The last step above follows from the concavity of $p\left[x\right]$, and the fact that $\pi(\partial(S_j)) \geq$

$(1/3)\phi x_j$. We now use the induction assumption that I_t holds,

$$\begin{aligned}
p\left[x_j\right] &\leq \alpha + \frac{1}{2}\big((4/3)(x_j - (1/3)\phi x_j) + f_t(x_j - (1/3)\phi x_j)\big) \\
&\quad + \frac{1}{2}\big((4/3)(x_j + (1/3)\phi x_j) + f_t(x_j + (1/3)\phi x_j)\big) \\
&= (4/3)x_j + \alpha + \frac{1}{2}\big(f_t(x_j - (1/3)\phi x_j) + f_t(x_j + (1/3)\phi x_j)\big) \\
&\leq (4/3)x_j + \alpha + \frac{1}{2}\big(f_t(x_j - (1/3)\phi x_j) + f_t(x_j + (1/3)\phi x_j)\big) \\
&= (4/3)x_j + \alpha(t+1) \\
&\quad + \frac{1}{2}\left(\sqrt{x_j - (1/3)\phi x_j} + \sqrt{x_j + (1/3)\phi x_j}\right)\frac{1}{\sqrt{\pi(v)}}\left(1 - \frac{\phi^2}{8}\right)^t.
\end{aligned}$$

We now use the fact that for any $x \geq 0$ and $z \in [0,1]$,

$$\frac{1}{2}\left(\sqrt{x - zx} + \sqrt{x + zx}\right) \leq \sqrt{x}\left(1 - z^2/8\right).$$

Applying this bound with $x = x_j$ and $z = (1/3)\phi$, we obtain the following.

$$\begin{aligned}
p\left[x_j\right] &\leq (4/3)x + \alpha(t+1) + \sqrt{x_j/\pi(v)}\left(1 - \frac{\phi^2}{8}\right)\left(1 - \frac{\phi^2}{8}\right)^t \\
&= (4/3)x + f_{t+1}(x_j).
\end{aligned}$$

This completes the proof. □

Proof of Lemma 6.10. Let π_S be the probability distribution described in the statement of the lemma, the one obtained by sampling a vertex v from the distribution π, conditioned on the event that $v \in S$.

The amount of probability that moves from S to $\bar{S}$ in the step from $\mathrm{pr}(\alpha, \pi_S)$ to $\mathrm{pr}(\alpha, \pi_S)M$ is equal to $[\mathrm{pr}(\alpha, \pi_S)](\partial(S))$, so we have

$$\begin{aligned}
[\mathrm{pr}(\alpha, \pi_S)M]\left(\bar{S}\right) &\leq [\mathrm{pr}(\alpha, \pi_S)]\left(\bar{S}\right) + [\mathrm{pr}(\alpha, \pi_S)](\partial(S)) \\
&\leq [\mathrm{pr}(\alpha, \pi_S)]\left(\bar{S}\right) + \mathrm{pr}(\alpha, \pi_S)\left[\pi(\partial(S))\right].
\end{aligned}$$

By Lemma 6.12,

$$\begin{aligned}
\mathrm{pr}(\alpha, \pi_S)\left[\pi(\partial(S))\right] &\leq \pi_S\left[\pi(\partial(S))\right] \\
&= \frac{\pi(\partial(S))}{\pi(S)} \\
&= \Phi_M(S).
\end{aligned}$$

We combine this observation with the personalized PageRank equation.

$$\begin{aligned}
[\mathrm{pr}_M(\alpha, \pi_S)]\left(\bar{S}\right) &= [\alpha\pi_S + (1-\alpha)\mathrm{pr}_M(\alpha, \pi_S)M]\left(\bar{S}\right) \\
&= (1-\alpha)[\mathrm{pr}_M(\alpha, \pi_S)M]\left(\bar{S}\right) \\
&\leq (1-\alpha)[\mathrm{pr}_M(\alpha, \pi_S)]\left(\bar{S}\right) + \Phi_M(S).
\end{aligned}$$

This implies the following,

$$[\mathrm{pr}(\alpha, \pi_S)](\bar{S}) \leq \frac{\Phi_M(S)}{\alpha}.$$

If we sample a vertex from the distribution π_S, then at least $3/4$ of the time $\mathrm{pr}_M(\alpha, 1_v)(\bar{S})$ is at most 4 times its expected value of $\mathrm{pr}_M(\alpha, \pi_S)(\bar{S})$, and the result follows.

□

6.5 Partitioning a strongly connected graph

In the next two sections we describe two possible approaches to partitioning a directed graph. In this section, we describe the straightforward method that applies only when the directed graph is strongly connected.

If the graph is strongly connected, then we may apply Theorem 6.8 to the lazy random walk Markov chain LW, which is defined to be

$$LW = LW(A) = \frac{1}{2}(I + AD^{-1}).$$

Here, D is the diagonal matrix whose nonzero elements are the outdegrees of the vertices. The laziness of the walk ensures that LW is ergodic whenever A is strongly connected, which allows us to apply our main theorem to LW.

To apply Theorem 6.8 to the lazy walk Markov chain LW, we must compute and perform a sweep over a personalized PageRank vector. When performing the sweep, we must know the stationary distribution of LW to sort the vertices into the proper order. The stationary distribution needs to be computed only once, and afterwards we can find numerous cuts by computing a single personalized PageRank vector per cut. The necessary computation is summarized below.

Applying Theorem 6.8 to the lazy walk Markov chain of a strongly connected graph.

We are given as input a strongly connected directed graph with lazy walk matrix LW. The following procedure may be used to apply Theorem 6.8 with several different starting vertices and values of α. The offline preprocessing must be done once, after which the local computation may be performed as many times as desired.

Offline Preprocessing:

1. Compute the stationary distribution π of LW.

Local computation:

1. Pick a starting vertex v and a value of α.
2. Compute $p = \mathrm{pr}_{LW}(\alpha, v)$, using `LocalPR`.
3. Sort the vertices in nonincreasing order of $p(x)/\pi(x)$.
4. Let S_j be the set of the top j vertices in this ranking.
5. Compute the LW-conductance of each set S_j^p, and output the set with the smallest LW-conductance.

6.6 Partitioning the PageRank Markov chain

The majority of directed graphs that arise in practice are not strongly connected, so we cannot directly apply the results of the previous section to such a graph. In this section, we describe how Theorem 6.8 can be applied to the PageRank Markov chain of an arbitrary graph, which is always ergodic. We show that the notion of conductance associated with this Markov chain has a natural interpretation in terms of PageRank. We describe how to find a large number of sets with low conductance in the PageRank Markov chain by performing a small number (two) of global PageRank computations as a preprocessing step, followed

by any desired number of local computations.

6.6.1 The PageRank Markov chain

We now define the PageRank Markov chain $M_\beta = M_\beta(A)$ in terms of the adjacency matrix A of an arbitrary directed graph. We first add a self-loop to each vertex of the directed graph to ensure that no vertex has outdegree zero. This ensures the random walk matrix $W = D^{-1}A$ is a Markov chain, where D is the diagonal matrix containing the modified outdegrees after the self-loops have been added.

Let $\psi = \frac{1}{n}1_V$ be the uniform distribution, and let β be a constant in $[0, 1]$. Recall that the global PageRank vector $\mathrm{pr}_M(\beta, \psi)$ is the unique solution of the linear system

$$\mathrm{pr}_M(\beta, \psi) = \beta\psi + (1 - \beta)\mathrm{pr}_M(\beta, \psi)W. \tag{6.5}$$

We refer to β as the *global jump probability*. The PageRank Markov chain M_β is defined to be

$$M_\beta = \beta K_\psi + (1 - \beta)W,$$

where $K_\psi = \vec{1}^T\psi$ is the dense rank-1 matrix obtained by taking the outer product of ψ with the all-ones vector. The global PageRank vector $\mathrm{pr}_M(\beta, \psi)$, is the stationary distribution for M_β, satisfying the equation $\mathrm{pr}_M(\beta, \psi) = \mathrm{pr}_M(\beta, \psi)M_\beta$. The PageRank Markov chain M_β is ergodic for any value of $\beta \in (0, 1]$.

The notion of conductance associated with the PageRank Markov chain M_β has a natural interpretation in terms of the global PageRank vector $\mathrm{pr}_M(\beta, \psi)$. We will use the shorthand notation $\mathrm{pr}_\beta = \mathrm{pr}_M(\beta, \psi)$ and $\Phi_\beta(S) = \Phi_{M_\beta}(S)$. Then, for any a set of vertices S, we have

$$\Phi_\beta(S) = \frac{\mathrm{pr}_\beta(\partial(S))}{\mathrm{pr}_\beta(S)}.$$

This is the probability that if we choose a vertex from S with probability proportional to its PageRank, and then take a single step in the PageRank Markov chain M_β, we end up at a vertex outside of S.

6.6.2 Computing personalized PageRank in the PageRank Markov chain

To apply our local partitioning theorem to M_β, we must compute a personalized PageRank vector in the Markov chain M_β. The personalized PageRank vector $\mathrm{pr}_{M_\beta}(\alpha, s)$ is the unique solution of the linear system

$$\mathrm{pr}_{M_\beta}(\alpha, s) = \alpha s + (1-\alpha)\mathrm{pr}_{M_\beta}(\alpha, s)M_\beta.$$

Although this is a personalized PageRank vector, the Markov chain M_β is dense because of its global random jump, so it is not possible to compute $\mathrm{pr}_{M_\beta}(\alpha, s)$ efficiently using $\texttt{LocalPR}(M_\beta, \alpha, s)$. We will show that $\mathrm{pr}_{M_\beta}(\alpha, s)$ can be computed efficiently in another way, by taking a linear combination of a personalized PageRank vector and a global PageRank vector in the random walk Markov chain W.

We now present two interpretations of the PageRank vector $\mathrm{pr}_{M_\beta}(\alpha, s)$. By definition, $\mathrm{pr}_{M_\beta}(\alpha, s)$ is a personalized PageRank vector in the Markov chain M_β. It can also be viewed as a PageRank vector in the random walk Markov chain W. When viewed as a PageRank vector in W, its starting vector is a linear combination of the uniform distribution ψ and the starting distribution s, and its jump probability is $\gamma = \alpha + \beta - \alpha\beta$.

$$\begin{aligned}
\mathrm{pr}_{M_\beta}(\alpha, s) &= \alpha s + (1-\alpha)\mathrm{pr}_\beta(\alpha, s)M_\beta \\
&= \alpha s + (1-\alpha)\beta\psi + (1-\alpha)(1-\beta)\mathrm{pr}_\beta(\alpha, s)W \\
&= \gamma(\frac{\alpha}{\gamma}s + \frac{(1-\alpha)\beta}{\gamma}\psi) + (1-\gamma)\mathrm{pr}_\beta(\alpha, s)W \\
&= \mathrm{pr}_W(\gamma, s').
\end{aligned}$$

Here $\gamma = \alpha + \beta - \alpha\beta$, and $s' = \frac{\alpha}{\gamma}s + \frac{(1-\alpha)\beta}{\gamma}\psi$. Using the fact that a PageRank vector is a linear function of its starting distribution, we can write

$$\begin{aligned}
\mathrm{pr}_{M_\beta}(\alpha, s) &= \mathrm{pr}_W(\gamma, \frac{\alpha}{\gamma}s + \frac{(1-\alpha)\beta}{\psi}) \\
&= \frac{\alpha}{\gamma}\mathrm{pr}_W(\gamma, s) + \frac{(1-\alpha)\beta}{\gamma}\mathrm{pr}_W(\gamma, \psi).
\end{aligned}$$

In summary, we have taken a personalized PageRank vector $\mathrm{pr}_{M_\beta}(\alpha, s)$ from the PageRank Markov chain M_β, and written it as a linear combination of two PageRank vectors from the walk Markov chain W. One of these is a personalized PageRank vector in W with starting distribution s, and the other is a global PageRank vector in W with starting distribution ψ.

6.6.3 Local partitioning in the PageRank Markov chain

By applying our main theorem to the PageRank Markov chain, we obtain the following corollary, which shows a sweep over the PageRank vector $\mathrm{pr}_{M_\beta}(\alpha, v)$ produces a set with small M_β-conductance.

Corollary 6.13. *Let S be a set of vertices such that $\mathrm{pr}_\beta(S) \leq \frac{1}{2}$ and $\Phi_\beta(S) \leq \alpha/16$, for some constants α and β. If a vertex v is sampled from S according to the probability distribution $\mathrm{pr}_\beta(v)/\mathrm{pr}_\beta(S)$, then with probability at least $1/2$ we have $\Phi_\beta(\mathrm{pr}_{M_\beta}(\alpha, v)) = O(\sqrt{\alpha \log |S|})$.*

Proof. The corollary is immediate, by applying Theorem 6.8 to the ergodic Markov chain M_β. □

To carry out the computation required by the corollary, we need to compute the stationary distribution of M_β, which is just the global PageRank vector $\mathrm{pr}_W(\beta, \psi)$. For each cut we want to find, we also need to compute a personalized PageRank vector $\mathrm{pr}_{M_\beta}(\alpha, v)$ in the Markov chain M_β. This can be done by computing $\mathrm{pr}_W(\gamma, v)$ and $\mathrm{pr}_W(\gamma, \psi)$, and then taking a linear combination of these two PageRank vectors, as described in the previous section. If we fix the values of α and β, we can compute the two global PageRank vectors $\mathrm{pr}_W(\beta, \psi)$ and $\mathrm{pr}_W(\gamma, \psi)$ ahead of time, and then compute a large number of personalized PageRank vectors $\mathrm{pr}_W(\gamma, v)$ using `LocalPR`. This procedure is summarized below.

Applying Corollary 6.13 to the PageRank Markov chain.
We are given as input the adjacency matrix A of a directed graph (not necessarily strongly connected), the global jump probability α, and the local jump probability β. The following procedure may be used to apply Theorem 6.8 at several different starting vertices with these fixed values of α and β. The offline preprocessing must be done once, after which the local computation may be performed as many times as desired.

Offline Preprocessing:
We must compute two global PageRank vectors.

1. Let $\gamma = \alpha + \beta - \alpha\beta$.
2. Let $W = W(A)$ be the random walk matrix of A.
3. Compute the two global PageRank vectors $\mathrm{pr}_W(\beta, \psi)$ and $\mathrm{pr}_W(\gamma, \psi)$ using the algorithm `GlobalPR`.

Local Computation:

1. Pick a starting vertex v.
2. Compute $\mathrm{pr}_W(\gamma, v)$, using `LocalPR`.
3. Obtain $p = \mathrm{pr}_{M_\beta}(\alpha, v)$ by taking a linear combination of $\mathrm{pr}_W(\gamma, v)$ and $\mathrm{pr}_W(\gamma, \psi)$,
$$p = \mathrm{pr}_{M_\beta}(\alpha, v) = \frac{\alpha}{\gamma}\mathrm{pr}_W(\gamma, v) + \frac{(1-\alpha)\beta}{\gamma}\mathrm{pr}_W(\gamma, \psi).$$
4. Rank the vertices in nonincreasing order of $p(x)/\mathrm{pr}_\beta(x)$.
5. Let S_j be the set of the top j vertices in this ranking.
6. Compute the β-conductances $\Phi_\beta(S_j)$ for each set S_j, and output the set with the smallest β-conductance.

6.6.4 When is partitioning the PageRank Markov chain effective?

It is not immediately clear when Corollary 6.13 yields a meaningful approximation guarantee, and when it does not. The answer depends on the structure of the graph. As we increase β, we increase the probability of the global jump, which ensures that the β-conductance of every set in the graph is at least roughly β. If we partition the PageRank Markov chain of a graph with no edges, all sets will have conductance roughly β, and the result will be vacuous. On the other hand, if we partition the PageRank Markov chain of an undirected graph, using a very small value of β, there will be good partitions of the graph that have β-conductance larger than β, so we obtain a meaningful approximation guarantee.

Loosely speaking, partitioning the PageRank Markov chain M_β gives interesting results exactly when there are interesting partitions of the graph that have β-conductance larger than β. We can provide some evidence for this claim by separating the β-conductance $\Phi_\beta(S)$ into two parts: the contribution $\Psi_\beta(S)$ from real graph edges in W, and the contribution from the random jump. We define

$$\Psi_\beta(S) = \frac{\sum_{(u,v)\in S\times\bar{S})} \mathrm{pr}_\beta(u)W(u,v)}{\mathrm{pr}_\beta(S)}.$$

It is not hard to see that $\Phi_\beta(S)$ and $\Psi_\beta(S)$ are related by the following equation.

$$\Phi_\beta(S) = (1-\beta)\Psi_\beta(S) + \beta\frac{|\bar{S}|}{n}.$$

If a set S has β-conductance significantly larger than β, our algorithm finds a set S' for which $\Psi_\beta(S')$ is nearly as small as $\Psi_\beta(S)$. In particular, if S is a set of vertices for which $\Psi_\beta(S) = \Omega(\Phi_\beta(S))$, and S' is a set of vertices for which $\Phi_\beta(S') = O(\sqrt{\Phi_\beta(S)\log n})$, which is the conductance guaranteed by Corollary 6.13, then we also have

$$\Psi_\beta(S') = O(\sqrt{\Psi_\beta(S)\log n}).$$

6.6.5 Commentary

We have extended a limited subset of our local partitioning results to directed graphs. In the case of undirected graphs, we went on to develop a more complicated local partitioning algorithm that provides stronger guarantees about the sizes of the sets it produces. This required showing that a sweep over an approximate PageRank vector produces a low-conductance cut, and showing that a sufficient approximation can be computed quickly. We leave the question of whether this can be done efficiently for directed graphs for future work.

A related problem we have not addressed is that to create the vector $\mathrm{pr}_{M_\beta}(\alpha, v)$, we take a linear combination of the vector $\mathrm{pr}_W(\gamma, v)$, which has small support, and the vector $\mathrm{pr}_W(\gamma, \psi)$, which is nonzero everywhere. One would hope it suffices to perform the sweep only over those vertices in the support of $\mathrm{pr}_W(\gamma, v)$, rather than every vertex in the graph, but we have not proved this.

6.7 Acknowledgement

Chapter 6 contains material to appear in the following article.
R. Andersen and F. Chung and K. Lang. *Local partitioning in directed graphs*. Manuscript in preparation, 2007. The dissertation author was the primary investigator and author of this paper.

7 A Local Algorithm for Finding Dense Subgraphs

We present a local algorithm for finding dense subgraphs of bipartite graphs, according to the measure of density proposed by Kannan and Vinay. The algorithm takes as input a bipartite graph with a specified starting vertex, and attempts to find a dense subgraph near that vertex. We prove that for any subgraph S with k vertices and density θ, there are a substantial number of starting vertices within S for which the algorithm produces a subgraph S' with density $\Omega(\theta/\log\Delta)$, where Δ is the maximum degree. The running time of the algorithm is $O(\Delta k^2)$, independent of the number of vertices in the graph.

7.1 Introduction

Identifying dense subgraphs has become an important task in the analysis of large networks, and a collection of dense subgraphs may reveal a wealth of information about a graph. In particular, it has been observed that clusters in a graph often contain dense subgraphs as cores [36], and that dense subgraphs in the web often indicate link spam [25].

The notion of density we consider in this paper was introduced by Kannan and Vinay [33], and is well-suited to finding dense subgraphs within bipartite graphs derived from large datasets. As an example, consider the bipartite graph that describes the membership relationships between a set of groups $\mathcal{G}$ and a set of users $\mathcal{U}$, with an edge between a user and a group when the user belongs to the group. The density of the induced subgraph that consists of the set of groups

$S \subseteq \mathcal{G}$ and the set of members $T \subseteq \mathcal{U}$ is defined to be

$$d(S,T) = \frac{e(S,T)}{\sqrt{|S|}\sqrt{|T|}},$$

which is the total number of incidences between groups and members in the subgraph, divided by the geometric mean of the number of groups and number of members.

There are various algorithms for identifying subgraphs with high density according to the measure $d(S,T)$. Kannan and Vinay [33] gave a spectral algorithm that produces a subgraph whose density is within an $O(\log n)$ factor of optimal. Charikar [12] showed that a subgraph with optimal density can be identified in polynomial time by solving a linear program, and gave a greedy algorithm that produces a 2-approximation in linear time. To identify community cores or link spam in a large network, we would like to identify a large number of small dense subgraphs. Gibson, Kumar, and Tomkins [25] introduced a fast heuristic for this task using the technique of fingerprinting by shingles.

In this paper, we introduce a local approximation algorithm for finding dense subgraphs. This algorithm takes as input a graph and a specified *starting vertex*, and attempts to find a dense subgraph near the starting vertex. We prove a *local approximation guarantee*: for any subgraph S with density θ and k vertices, there are a substantial number of starting vertices within S for which the algorithm produces a subgraph with density $\Omega(\theta/\log\Delta)$. The running time of the algorithm is $O(\Delta k^2)$, where Δ is the maximum degree in the graph.

This local algorithm can perform several tasks that cannot be performed by known approximation algorithms for the densest subgraph problem. The algorithm can find a dense subgraph near a vertex of interest by examining only a portion of the entire graph. The algorithm can find a large collection of small dense subgraphs by searching in parallel from many different starting vertices. In addition, the algorithm provides an upper bound on the size of the subgraph it produces, which may make it a useful tool for producing a dense subgraph of a specified size.

The analysis of the algorithm is based on spectral techniques, and exploits the close relationship between the densest subgraph of a graph and the largest eigenvalue of the graph's adjacency matrix. We define a deterministic process

called the 'pruned growth process', which is closely related to the power method, except the vectors are rounded at each step to ensure that the number of nonzero elements is small. We show that by computing these sparse vectors we can identify a subgraph with high density.

In Section 7.2, we survey related work on the identification of dense subgraphs, and compare the measure of density introduced by Kannan and Vinay with others that have appeared in the literature. In Section 7.3, we define the 'pruned growth process'. In Section 7.4, we state our local algorithm, analyze its running time, and prove its approximation guarantee.

7.2 Preliminaries and related work

Let $G = (V, E)$ be an undirected bipartite graph with adjacency matrix A, and let L and R be the left and right sides of a fixed bipartition. We will allow the edges of the graph to have integer weights. The entry $A_{i,j}$ is the weight of edge $\{i, j\}$. The degree of a vertex v is the sum of the weights of the edges incident to v. We define Δ to be the maximum degree in the graph, and assume that $\Delta = O(poly(n))$.

For any two sets $S \subseteq L$ and $T \subseteq R$, we let (S, T) denote the induced bipartite subgraph of G on the set of vertices $S \cup T$, and we define $\mathrm{e}(S, T)$ to be the sum of the weights of the edges between S and T. We will sometimes use the inner product notation $\mathrm{e}(S, T) = \langle 1_S A, 1_T \rangle$, where 1_S is the indicator function for membership in S. Here and elsewhere, 1_S is taken to be a row vector, so we can multiply by matrices on the right. We define the *support* of a vector x to be the set of vertices on which x is nonzero. The two-norm of x is written $\|x\|$.

We will identify induced subgraphs of G that are dense according the following measure of density, which was introduced by Kannan and Vinay [33].

Definition 7.1. For any induced subgraph (S, T), we define

$$d(S, T) = \frac{\mathrm{e}(S, T)}{\sqrt{|S|}\sqrt{|T|}}.$$

We define the density $d(A)$ of the graph to be the maximum value of $d(S,T)$ over all induced subgraphs.

Our algorithm may be applied to an arbitrary directed graph (not necessarily bipartite), using the following standard trick. Given a directed graph with vertex set X, define a bipartite graph where $L = R = X$. For each edge $x \to y$ in the directed graph, place an undirected edge between the copy of x in L and the copy of y in R.

7.2.1 Related work

A different measure of density, equivalent to the average degree of the subgraph, has been studied extensively [27, 24, 35, 12].

Definition 7.2. Let $G = (V, E)$ be an undirected graph (not necessarily bipartite). For any set $S \subseteq V$, we define

$$g(S) = \frac{\mathrm{e}(S,S)}{|S|}.$$

We define $g(A)$ to be the maximum value of $g(S)$ over all subsets of V.

Both $d(A)$ and $g(A)$ can be computed exactly in polynomial time. Goldberg showed that a set S achieving $g(S) = g(A)$ can be found using maximum flow computations [27]. Such a set can also be found using the parametric flow algorithm of Gallo, Grigoriadis, and Tarjan [24]. Charikar showed that a subgraph (S,T) achieving $d(S,T) = d(A)$ can be found by solving a linear program [12].

Kortsarz and Peleg [35] gave a greedy 2-approximation algorithm for $g(A)$. Charikar [12] gave a greedy 2-approximation algorithm for $d(A)$. The running time of these algorithms is $O(m)$ in an unweighted graph, and $O(m + n \log n)$ in a weighted graph. Kannan and Vinay gave a spectral approximation algorithm for $d(A)$, which produces a subgraph (S,T) with density $d(S,T) = \Omega(d(A)/\log n)$ from the top singular vectors of A.

The densest k-subgraph problem is to identify the subgraph with the largest number of edges among all subgraphs of exactly k vertices. This problem is considerably more difficult than the densest subgraph problem; there is a large gap between the best approximation algorithms and hardness results known (see [19, 18]).

7.2.2 Comparison of the two objective functions

It is easier to compare the two objective functions $d(S,T)$ and $g(S)$ if we restrict $g(S)$ to bipartite graphs. In this case, $g(S)$ takes the following form.

Definition 7.3. For any subgraph (S,T), we define

$$g(S,T) = \frac{\mathrm{e}(S,T)}{|S|+|T|}.$$

We define $g(A)$ to be the maximum value of $g(S,T)$ over all induced subgraphs.

It is always true that $d(S,T) \geq 2g(S,T)$, becuase $2\sqrt{|S|}\sqrt{|T|} \leq |S|+|T|$. It is possible for $d(S,T)$ to be much larger than $g(S,T)$ if the sets S and T have very different sizes. In the complete bipartite graph $K_{a,b}$, we have $d(A) = \sqrt{ab}$, while $g(A) = ab/(a+b)$. In the case where $a = 1$, we have $d(A) = \sqrt{b}$ while $g(A) = b/(b+1) \sim 1$.

The relative merits of $d(S,T)$ and $g(S)$ as objective functions for density were discussed in [12, 33]. Frankly, we are considering $d(S,T)$ because it is more amenable to approximation by spectral algorithms than $g(S)$. The largest eigenvalue of the adjacency matrix A is closely related to $d(A)$, and this is essential to our algorithm.

7.3 The pruned growth process

We now define the deterministic process on which our local algorithm is based. The process generates a sequence of vectors $x_0, \ldots, x_T$ from a starting vector x_0. The main operation performed at each step is multiplication by the adjacency matrix A, as in the power method. The resulting vector is then pruned by setting to zero each entry whose value is below a certain threshold. This reduces the size of the support, which reduces the amount of computation required to compute the sequence.

Definition 7.4. Given a vector z and a nonnegative real number ϵ, we define $\mathrm{prune}_\epsilon(z)$ to be the vector obtained by setting to zero any entry of z whose value

is at most $\epsilon\|z\|$,

$$[\text{prune}_\epsilon(z)](u) = \begin{cases} z(u) & \text{if } z(u) > \epsilon\|z\|, \\ 0 & \text{otherwise.} \end{cases}$$

We remark that the number of nonzero entries in $\text{prune}_\epsilon(z)$ is at most $1/\epsilon^2$.

Definition 7.5. Given a starting vector x_0 and a sequence of real numbers $\epsilon_t \in [0, 1]$, we define the *pruned growth process* to be the sequence of vectors $x_0, \ldots, x_T$ defined by the following rule:

$$x_{t+1} = \text{prune}_{\epsilon_{t+1}}(x_t A).$$

We will eventually show that a subgraph with high density can be produced from the vector x_t whenever $\|x_{t+1}\|$ is significantly larger than $\|x_t\|$. To find such a subgraph, we will compute $y = x_t A$, partition the vertices in the support of x_t and y_t into disjoint sets according to their values in the direction vectors $\hat{x_t}$ and $\hat{y_t}$, and choose the highest density that occurs among all pairs of such sets.

Definition 7.6. Given a nonnegative vector z, we define $\hat{z} = z/\|z\|$. For each integer $i \geq 0$, we define the set $S_i(z)$ to be

$$\{\, v \,:\, \hat{z}(v) \in (2^{-(i+1)}, 2^{-i}] \,\}.$$

For each $t < T$, we let $y_t = x_t A$, and define $X_i^t = S_i(x_t)$ and $Y_j^t = S_j(y_t)$.

We remark that $\hat{z}$ can be reconstructed approximately from the sets $S_i(z)$. For any nonnegative vector z,

$$\hat{z} \leq \sum_{i \geq 0} 2^{-i} 1_{S_i(z)} \leq 2\hat{z}.$$

7.4 Local approximation algorithm

In this section, we state and analyze the main algorithm. The input to the algorithm is a graph, along with a *starting vertex* v and a *target size* K. We prove

that the running time depends mainly on the target size K, and is independent of the number of vertices in the graph. We prove that for any subgraph (S, T), there are a substantial number of starting vertices in S for which the algorithm produces a subgraph whose density is within an $O(\log \Delta)$ factor of $d(S, T)$.

`LocalDensity(v, K)`

Input: A vertex v and a target size K.

Output: A subgraph (X, Y).

1. Let $x_0 = 1_v$, let $T = \log(\sqrt{2|K|})$, and let $\epsilon_t = \frac{2^t}{8K}$.
2. Compute the vectors $x_0, \ldots, x_T$ of the pruned growth process.
3. For each time $t < T$, compute the density $d(X_i^t, Y_j^t)$ for each pair i, j where X_i^t and Y_j^t are both nonempty and $|i - j| \leq 6 + \log \Delta$.
4. Output a subgraph (X, Y) achieving the highest density encountered during step 3.

Theorem 7.7. *Let (S, T) be a subgraph such that $d(S, T) \geq 2\theta$, where $\theta \geq 1$. Then there exists a set $\mathrm{S}_\theta \subseteq S$, with the following properties.*

1. *$e(\mathrm{S}_\theta, T) \geq e(S, T)/2$,*
2. *If $v \in \mathrm{S}_\theta$ and $K \geq \max(|S|, |T|)$, then* `LocalDensity(v, K)` *outputs a subgraph (X, Y) such that*

$$d(X, Y) \geq \frac{\theta}{4(72 + 16 \log 4\Delta/\theta)} = \Omega\left(\frac{\theta}{\log \Delta}\right).$$

Theorem 7.8. `LocalDensity(v, K)` *runs in time $O(\Delta K^2)$.*

The proofs of Theorems 7.7 and 7.8 are given in section 7.4.3.

Here is an outline of how we will proceed. In Subsection 7.4.1, we show that the algorithm will produce a dense subgraph if, at any step, the norm $\|x_{t+1}\|$ is significantly larger than the previous norm $\|x_t\|$. In Subsection 7.4.2, we show that for any dense subgraph, there are many starting vertices for which can give a

lower bound on the growth of the norms $\|x_t A\|$. In Subsection 7.4.3, we combine these results to prove the approximation guarantee, which requires proving that our lower bound on $\|x_t A\|$ is robust with respect to the pruning process.

7.4.1 Fast growth implies a dense subgraph

The following lemma will be used to show that if $\|x_{t+1}\|$ is significantly larger than $\|x_t\|$, then one of the subgraphs (X_i^t, Y_j^t) has high density. The proof combines elements from [33, 9].

Lemma 7.9. *Given a nonnegative vector x, let $y = xA$. Let $X_i = S_i(\hat{x})$ and let $Y_j = S_j(\hat{y})$. Let P be the set of pairs (i, j) such that X_i and Y_j are nonempty and $|i - j| \leq 6 + \log \Delta$, and let*

$$d(x) = \max_{(i,j)\in P} d(X_i, Y_j).$$

If $\|y\| \geq \beta\|x\|$, where $\beta \geq 1/4$, then

$$d(x) \geq \beta/(72 + 16 \log \Delta/\beta).$$

Proof. Notice that $\langle \hat{x}A, \hat{y} \rangle = \|y\|/\|x\| \geq \beta$. To prove the theorem, we will bound $\langle \hat{x}A, \hat{y} \rangle$ in terms of $d(x)$.

$$\begin{aligned}\langle \hat{x}A, \hat{y} \rangle &\leq \left\langle \sum_{i\geq 0} 2^{-i} 1_{X_i} A, \sum_{j\geq 0} 2^{-j} 1_{Y_j} \right\rangle \\ &= \sum_{(i,j)\in I} 2^{-i} 2^{-j} \left\langle 1_{X_i} A, 1_{Y_j} \right\rangle.\end{aligned}$$

Here, I is the set of pairs $(i \geq 0, j \geq 0)$ for which X_i and Y_j are nonempty. We will partition I into three disjoint sets of pairs, P_γ, Q_γ, and R_γ, depending on the difference between i and j.

$$\begin{aligned}P_\gamma &= \{\, (i,j) \in I \,:\, |j - i| \leq \gamma \,\}, \\ Q_\gamma &= \{\, (i,j) \in I \,:\, i > j + \gamma \,\}, \\ R_\gamma &= \{\, (i,j) \in I \,:\, j > i + \gamma \,\}.\end{aligned}$$

We set $\gamma = 4 + \log \Delta/\beta$. Since $\beta \geq 1/4$, this ensures that $\gamma \leq 6 + \log \Delta$, and so $P_\gamma \subseteq P$.

We will sum over each set of pairs separately. The sum over indices in P_γ is

$$\begin{aligned} S(P_\gamma) &= \sum_{(i,j)\in P_\gamma} 2^{-i}2^{-j} \left\langle 1_{X_i} A, 1_{Y_j} \right\rangle \\ &= \sum_{(i,j)\in P_\gamma} 2^{-i}2^{-j} d(X_i, Y_j) \sqrt{|X_i|}\sqrt{|Y_j|} \\ &\leq \frac{1}{2} \sum_{(i,j)\in P_\gamma} d(X_i, Y_j) \left(2^{-2i}|X_i| + 2^{-2j}|Y_j|\right) \\ &\leq \frac{1}{2} d(x) \sum_{(i,j)\in P_\gamma} \left(2^{-2i}|X_i| + 2^{-2j}|Y_j|\right). \end{aligned}$$

The second-to-last line follows from the inequality $2ab \leq a^2 + b^2$, applied with $a = 2^{-i}\sqrt{|X_i|}$ and $b = 2^{-j}\sqrt{|Y_j|}$.

Now we observe that a given index i appears as the first element in at most $1 + 2\gamma$ pairs in P_γ. Similarly, an index j appears as the second element in at most $1 + 2\gamma$ pairs in P_γ. Therefore,

$$\begin{aligned} S(P_\gamma) &\leq \frac{1}{2}(1 + 2\gamma) d(x) \left(\sum_{i\geq 0} 2^{-2i}|X_i| + \sum_{j\geq 0} 2^{-2j}|Y_j| \right) \\ &\leq \frac{1}{2}(1 + 2\gamma) d(x) (\|2\hat{x}\|^2 + \|2\hat{y}\|^2) \\ &\leq 4(1 + 2\gamma) d(x). \end{aligned}$$

We bound the sums over Q_γ and R_γ in terms of the maximum degree Δ. The

sum over indices in Q_γ is

$$\begin{aligned} S(Q_\gamma) &= \sum_{(i,j)\in Q_\gamma} 2^{-i}2^{-j}\left\langle 1_{X_i}A, 1_{Y_j}\right\rangle \\ &\le \sum_{(i,j)\in Q_\gamma} 2^{-i}2^{-j}\Delta|X_i| \\ &\le \sum_{i\ge 0} 2^{-i}\Delta|X_i| \sum_{j>i+\gamma} 2^{-j} \\ &= \sum_{i\ge 0} 2^{-i}\Delta|X_i|2^{-(i+\gamma)} \\ &= \Delta 2^{-\gamma}\sum_{i\ge 0} 2^{-2i}|X_i| \\ &\le \Delta 2^{-\gamma}\|2\hat{x}\|^2 \\ &\le 4\Delta 2^{-\gamma}. \end{aligned}$$

Similarly, the sum over R_γ is

$$S(R_\gamma) \le \Delta 2^{-\gamma}\|2\hat{y}\|^2 \le 4\Delta 2^{-\gamma}.$$

By setting $\gamma = 4 + \log \Delta/\beta$, we have ensured that

$$\begin{aligned} \beta \le \langle \hat{x}A, \hat{y}\rangle &\le S(P_\gamma) + S(Q_\gamma) + S(R_\gamma) \\ &\le 4(1+2\gamma)d(x) + 8\Delta 2^{-\gamma} \\ &= 4(1+2\gamma)d(x) + \beta/2. \end{aligned}$$

Then, $\beta \ge 8(1+2\gamma)d(x) = (72 + 16\log\Delta/\beta)d(x)$.

□

7.4.2 Lower bounds on growth within a dense subgraph

To prove a lower bound on the growth of the norms $\|x_t\|$, we start with the fact that the maximum density $d(A)$ gives a lower bound on the largest eigenvalue of A.

Fact 7.10. *Let A be the adjacency matrix of an undirected graph, and let λ be the largest eigenvalue of A. Then, $\lambda \ge d(A)$. Furthermore, there is an eigenvector ϕ with eigenvalue λ whose entries are nonnegative.*

Proof. To prove that $\lambda \geq d(A)$, notice that for any sets $S \subseteq L$ and $T \subseteq R$,

$$\lambda \geq \max_{x,y} \frac{\langle xA, y\rangle}{\|x\|\,\|y\|} \geq \left\langle \frac{1_S}{\sqrt{|S|}} A, \frac{1_T}{\sqrt{|T|}} \right\rangle = \frac{\mathrm{e}(S,T)}{\sqrt{|S|}\sqrt{|T|}} = d(S,T).$$

It is not hard to see that if ϕ is an eigenvector with eigenvalue λ, then the vector whose entries are the absolute values of the entries of ϕ is also an eigenvector with eigenvalue λ. □

The fact above implies the lower bound $\|x_0 A^t\| \geq \langle \phi, x_0\rangle\, d(A)^t$, which depends on the maximum density $d(A)$ in the graph. This bound is not good enough for our purposes, because the constant term $\langle \phi, x_0\rangle$ may be small for a particular starting vertex. Instead, we give a lower bound that depends on the density of a particular subgraph (S,T); for many starting vertices within the set S, we give a lower bound of the form $\|x_0 A^t\| \geq \langle \psi, x_0\rangle\, \Omega(d(S,T)^t)$, where the constant term $\langle \psi, x_0\rangle$ is not too small. The vector ψ used to obtain this bound is an eigenvector of an induced subgraph of (S,T).

Definition 7.11. For any induced subgraph (S,T), we define $A_{(S,T)}$ to be the restriction of the adjacency matrix A to (S,T),

$$A_{(S,T)}(x,y) = \begin{cases} A(x,y) \text{ if } x \in S \text{ and } y \in T, \text{ or if } x \in T \text{ and } y \in S. \\ 0 \text{ otherwise.} \end{cases}$$

The following lemma identifies, for any subgraph (S,T), a set of starting vertices in S for which we can give a good lower bound on the norms $\|x_t\|$. This set of good starting vertices touches a set of edges that constitutes at least half of the total weight in the induced subgraph (S,T).

Lemma 7.12. *If (S,T) is a subgraph such that $d(S,T) \geq 2\theta$, then there exists a subset $\mathrm{S}_\theta \subseteq S$ with the following properties:*

1. $e(\mathrm{S}_\theta, T) \geq e(S,T)/2$
2. *For each $v \in \mathrm{S}_\theta$, there is a nonnegative unit vector ψ such that*
   - *(a)* $\mathrm{Supp}(\psi) \subseteq S \cup T$,

(b) $\psi A \geq \theta\psi$,

(c) $\psi(v) \geq \frac{1}{\sqrt{2|S|}}$.

Proof. Let S_θ be the largest subset of S for which property (2) holds, and consider the set $S' = S \setminus \mathrm{S}_\theta$. If S_θ does not satisfy property (1), then

$$e(S', T) = e(S, T) - e(\mathrm{S}_\theta, T) \geq \frac{e(S, T)}{2},$$

and so $d(S', T) \geq d(S, T)/2 \geq \theta$.

Let λ be the largest eigenvalue of $A_{(S',T)}$. We know from fact 7.10 that there is an eigenvector ψ of $A_{(S',T)}$ with unit length whose entries are all nonnegative, and whose corresponding eigenvalue λ satisfies

$$\lambda \geq d(S', T) \geq \theta.$$

It is easy to see that ψ satisfies properties (a) and (b). We will now identify a vertex in S' for which $\psi(v) \geq 1/\sqrt{2|S|}$. This will imply that v is in S_θ, which will show that S_θ must satisfy property (1), and thus complete the proof.

Let $\psi_{S'}$ and ψ_T be the projections of ψ onto S' and T, and observe that $\|\psi_{S'}\| = \|\psi_T\| = \frac{1}{\sqrt{2}}$. This is true because $\psi_{S'} A_{(S',T)} = \lambda\psi_T$, which implies that $\lambda\|\psi_{S'}\| \geq \|\psi_{S'} A_{(S',T)}\| = \lambda\|\psi_T\|$. There must at least one vertex v in S' that satisfies $\psi(v) \geq 1/\sqrt{2|S'|}$, since otherwise we would have $\|\psi_{S'}\|^2 < 1/2$. □

7.4.3 Analysis of the local algorithm

Proof of Theorem 7.7. Let (S, T) be a subgraph for which $d(S, T) \geq 2\theta$, where $\theta \geq 1$. We will prove that for each vertex v in the set S_θ, which was described in Lemma 7.12, the algorithm `LocalDensity(v,K)` outputs a subgraph with density at least $\theta/4L$, where $L = (72 + 16 \log 4\Delta/\theta)$, provided that $K \geq \max(|S|, |T|)$.

Let $x_0, \ldots, x_T$ be the pruned growth process vectors computed by the algorithm. If for some time $t < T$ the vector x_t satisfies $\|x_t A\|/\|x_t\| \geq \theta/4$, then Lemma 7.9 shows that $d(x_t) \geq \theta/4L$, which ensures that the algorithm will output a set with the desired density. We now assume $\|x_t A\|/\|x_t\| < \theta/4$ for every $t < T$,

and attempt to derive a contradiction. Since $\|x_0\| = 1$ and $\|x_{t+1}\| \leq \|x_t A\|$, this assumption implies

$$\|x_t\| \leq \|x_{t-1}A\| < (\theta/4)^t \quad \text{for every } t \leq T. \tag{7.1}$$

Since $v \in \mathrm{S}_\theta$, there exists a nonnegative unit vector ψ such that $\psi A \geq \theta\psi$, such that $\mathrm{Supp}(\psi) \subseteq S \cup T$, and such that $\psi(v) \geq \frac{1}{\sqrt{2|S|}}$. We will prove the following lower bound on the inner product of x_t with ψ.

$$\langle x_t, \psi \rangle \geq \frac{1}{\sqrt{2|S|}}(\theta/2)^t \quad \text{for every } t \leq T. \tag{7.2}$$

When we prove equation (7.2), it will contradict equation (7.1) when $t = T = \log(\sqrt{2|S|})$, and the theorem will be proved.

We will prove that equation (7.2) holds by induction. We know it holds for $t = 0$. The only difficulty in the inductive step is to bound the effect of pruning on the projection of x_t onto ψ. We define r_t to be the vector that is removed during the pruning.

$$\begin{aligned} r_t &= x_{t-1}A - x_t \\ &= x_{t-1}A - \mathrm{prune}_{\epsilon_t}(x_{t-1}A). \end{aligned}$$

The value of r_t at any given vertex is at most $\epsilon_t \|x_{t-1}A\|$. Let r'_t be the vector that is equal to r_t on the support of ψ, and zero elsewhere. Since the support of ψ is contained in $S \cup T$, and the support of r_t is contained in either L or R, the support of the vector r'_t contains at most $\max(|S|, |T|) \leq K$ vertices. The inner product of r_t and ψ can then be bounded as follows.

$$\begin{aligned} \langle r_t, \psi \rangle &= \langle r'_t, \psi \rangle \\ &\leq \|r'_t\| \\ &\leq \epsilon_t \|x_{t-1}A\| \sqrt{K}. \end{aligned}$$

Now assume the induction hypothesis holds for $t-1$, which means that $\langle x_{t-1}, \psi \rangle \geq (1/\sqrt{2|S|})(\theta/2)^{t-1}$, and recall from (7.1) that $\|x_t\| \leq \|x_{t-1}A\| < (\theta/4)^t$. Then,

$$\begin{aligned}
\langle x_t, \psi \rangle &= \langle x_{t-1}A - r_t, \psi \rangle \\
&= \langle x_{t-1}A, \psi \rangle - \langle r_t, \psi \rangle \\
&\geq \theta \langle x_{t-1}, \psi \rangle - \epsilon_t \|x_{t-1}A\| \sqrt{K} \\
&\geq \frac{\theta}{\sqrt{2|S|}} \left(\frac{\theta}{2}\right)^{t-1} - \epsilon_t \sqrt{K} \left(\frac{\theta}{4}\right)^t \\
&\geq \left(\frac{\theta}{2}\right)^t \left(\frac{2}{\sqrt{2|S|}} - 4\epsilon_t 2^{-t} \sqrt{K}\right) \\
&\geq \left(\frac{\theta}{2}\right)^t \frac{1}{\sqrt{2|S|}}.
\end{aligned}$$

The last step follows because $\epsilon_t = 2^t/8K$. This completes the proof. □

Proof of Theorem 7.8. We bound the running time of `LocalDensity(v,K)` by bounding the number of vertices in the support of x_t at each step. Since x_t is at least $\epsilon_t \|x_t\|$ wherever it is nonzero, we have

$$\|x_t\|^2 \geq |\mathrm{Supp}(x_t)| \epsilon_t^2 \|x_t\|^2,$$

and so

$$|\mathrm{Supp}(x_t)| \leq \frac{1}{\epsilon_t^2}.$$

We let D_t be the sum of the degrees of the vertices in $\mathrm{Supp}(x_t)$, which is at most

$$O(\Delta |\mathrm{Supp}(x_t)|) = O(\Delta/\epsilon_t^2) = O(\Delta K^2 2^{-2t}).$$

Consider the amount of computation required at step t. We can compute $x_t A$ and x_{t+1} from x_t in time $O(D_t)$. This computation dominates the running time, so the total running time of the algorithm is

$$\sum_{t=0}^{T} O(\Delta K^2 2^{-2t}) = O(\Delta K^2).$$

We still need to show that the rest of the compuation at time t can be carried out in time $O(D_t)$. Namely, we need to compute the density $d(X_i^t, Y_j^t)$ for each

pair $(i, j) \in P_t$, where P_t is the set of pairs for which X_i^t and Y_j^t are both nonempty and $|i - j| \leq 6 + \log \Delta$. To compute this, we maintain a set of variables $W_{i,j}$ that record the total weight of edges between X_i^t and Y_j^t. The largest value of i we need to consider is $O(1/\epsilon_t) = O(\log n)$, and the largest value of j that we need to consider is $O(1/\epsilon_t) + O(\log \Delta) = O(\log n)$, so the number of variables we need to maintain is $O(\log^2 n)$. We loop through the vertices in $\mathrm{Supp}(x_t)$ and examine their incident edges, each of which belongs to at most one of the subgraphs (X_i^t, Y_j^t). For each edge, we add its weight to the appropriate variable $W_{i,j}$. This can be done in time $O(D_t)$. Once we have the values $\mathrm{e}(X_i^t, Y_j^t)$ for each pair in P_t, it is easy to compute the density of each subgraph and choose the densest one.

□

7.5 Acknowledgement

Chapter 7 contains material to appear in the following article.

R. Andersen. *A local algorithm for finding dense subgraphs*. Manuscript in preparation, 2007.

Bibliography

[1] N. Alon. Eigenvalues and expanders. *Combinatorica*, 6(2):83–96, 1986.

[2] N. Alon and V. Milman. Isoperimetric inequalities for graphs and superconcentrators. *J. Combin. Theory B*, 38:73–88, 1985.

[3] R. Andersen, F. Chung, and K. Lang. Local graph partitioning using pagerank vectors. In *FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 475–486, Washington, DC, USA, 2006. IEEE Computer Society.

[4] S. Arora, E. Hazan, and S. Kale. $0(sqrt(logn))$ approximation to sparsest cut in $\tilde{O}(n^2)$ time. In *FOCS*, pages 238–247, 2004.

[5] S. Arora and S. Kale. A combinatorial, primal-dual approach to semidefinite programs. To appear in 39th STOC, 2007.

[6] S. Arora, J. R. Lee, and A. Naor. Euclidean distortion and the sparsest cut. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 553–562, New York, NY, USA, 2005. ACM Press.

[7] S. Arora, S. Rao, and U. Vazirani. Expander flows, geometric embeddings and graph partitioning. In *STOC '04: Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 222–231, New York, NY, USA, 2004. ACM Press.

[8] P. Berkhin. Bookmark-coloring algorithm for personalized pagerank computing. *Internet Math.*, 3(1):41–62, 2006.

[9] Y. Bilu and N. Linial. Constructing expander graphs by 2-lifts and discrepancy vs. spectral gap. In *FOCS '04: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS'04)*, pages 404–412, Washington, DC, USA, 2004. IEEE Computer Society.

[10] C. Borgs, J. T. Chayes, M. Mahdian, and A. Saberi. Exploring the community structure of newsgroups. In *KDD*, pages 783–787, 2004.

[11] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[12] M. Charikar. Greedy approximation algorithms for finding dense components in a graph. In *APPROX 2000, Proceedings of the Third International Workshop on Approximation Algorithms for Combinatorial Optimization*, 2000.

[13] M. Charikar, K. Makarychev, and Y. Makarychev. Directed metrics and directed graph partitioning problems. In *SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 51–60, New York, NY, USA, 2006. ACM Press.

[14] F. Chung. *Spectral graph theory*, volume Number 92 in CBMS Regional Conference Series in Mathematics. American Mathematical Society, 1997.

[15] W. E. Donath and A. J. Hoffman. Algorithms for partitioning graphs and computer logic based on eigenvectors of connection matrices. *IBM Technical Disclosure Bulletin*, 15(3):938–944, 1972.

[16] W. E. Donath and A. J. Hoffman. Lower bounds for the partitioning of graphs. *IBM J. Res. Develop*, 17:420–425, 1973.

[17] U. Feige, M. Hajiaghayi, and J. R. Lee. Improved approximation algorithms for minimum-weight vertex separators. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 563–572, New York, NY, USA, 2005. ACM Press.

[18] U. Feige, D. Peleg, and G. Kortsarz. The dense k -subgraph problem. *Algorithmica*, 29(3):410–421, 2001.

[19] U. Feige and M. Seltser. On the densest k-subgraph problem. Technical Report CS97-16, 1, 1997.

[20] M. Fiedler. A property of eigenvectors of non-negative symmetric matrices and its application to graph theory. *Czeckoslovak Mathematical Journal*, 25(619672), 1975.

[21] J. A. Fill. Eigenvalue bounds on convergence to stationarity for nonreversible Markov chains, with an application to the exclusion process. *Ann. Appl. Probab.*, 1(1):62–87, 1991.

[22] G. Flake, S. Lawrence, and C. L. Giles. Efficient identification of web communities. In *Sixth ACM SIGKDD*, pages 150–160, Boston, MA, August 20–23 2000.

[23] D. Fogaras and B. Racz. Towards scaling fully personalized pagerank. In *Proceedings of the 3rd Workshop on Algorithms and Models for the Web-Graph (WAW)*, pages 105–117, October 2004.

[24] G. Gallo, M. D. Grigoriadis, and R. E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM J. Comput.*, 18(1):30–55, 1989.

[25] D. Gibson, R. Kumar, and A. Tomkins. Discovering large dense subgraphs in massive graphs, 2005.

[26] D. Gleich and M. Polito. Approximating personalized pagerank with minimal use of webgraph data. To appear in Internet Mathematics.

[27] A. Goldberg. Finding a maximum density subgraph. Technical Report UCB CSD 84/71, University of California, Berkeley, 1984.

[28] Z. Gyöngyi, P. Berkhin, H. Garcia-Molina, and J. Pedersen. Link spam detection based on mass estimation. In *Proceedings of the 32nd International Conference on Very Large Databases*. ACM, 2006.

[29] T. H. Haveliwala. Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *IEEE Trans. Knowl. Data Eng.*, 15(4):784–796, 2003.

[30] G. Jeh and J. Widom. Scaling personalized web search. In *Proceedings of the 12th World Wide Web Conference (WWW)*, pages 271–279, 2003.

[31] M. Jerrum and A.Sinclair. Conductance and the rapid mixing property for markov chains: the approximation of the permanent resolved. In *Symposium on Theory of Computing*, pages 235–243, May 1988.

[32] R. Kannan, S. Vempala, and A. Vetta. On clusterings: Good, bad and spectral. *J. ACM*, 51(3):497–515, 2004.

[33] R. Kannan and V. Vinay. Analyzing the structure of large graphs. Manuscript, 1999.

[34] R. Khandekar, S. Rao, and U. Vazirani. Graph partitioning using single commodity flows. In *STOC '06: Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 385–390, New York, NY, USA, 2006. ACM Press.

[35] G. Kortsarz and D. Peleg. Generating sparse 2-spanners. *J. Algorithms*, 17(2):222–236, 1994.

[36] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the Web for emerging cyber-communities. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(11–16):1481–1493, 1999.

[37] K. J. Lang. Fixing two weaknesses of the spectral method. In *NIPS*, 2005.

[38] J. R. Lee. On distance scales, embeddings, and efficient relaxations of the cut cone. In *SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 92–101, Philadelphia, PA, USA, 2005. Society for Industrial and Applied Mathematics.

[39] F. T. Leighton and S. Rao. An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms. In *FOCS*, pages 422–431, 1988.

[40] L. Lovász and M. Simonovits. The mixing rate of markov chains, an isoperimetric inequality, and computing the volume. In *FOCS*, pages 346–354, 1990.

[41] L. Lovász and M. Simonovits. Random walks in a convex body and an improved volume algorithm. *Random Struct. Algorithms*, 4(4):359–412, 1993.

[42] M. Mihail. Conductance and convergence of markov chains—a combinatorial treatment of expanders. In *Proc. of 30th FOCS*, pages 526–531, 1989.

[43] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[44] T. Sarlós, A. A. Benczúr, K. Csalogány, D. Fogaras, and B. Rácz. To randomize or not to randomize: space optimal summaries for hyperlink analysis. In *WWW*, pages 297–306, 2006.

[45] H. D. Simon and S.-H. Teng. How good is recursive bisection? *SIAM Journal on Scientific Computing*, 18(5):1436–1445, 1997.

[46] D. A. Spielman and S.-H. Teng. Spectral partitioning works: Planar graphs and finite element meshes. In *IEEE Symposium on Foundations of Computer Science*, pages 96–105, 1996.

[47] D. A. Spielman and S.-H. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *ACM STOC-04*, pages 81–90, New York, NY, USA, 2004. ACM Press.