Local Graph Partitioning using PageRank Vectors

Reid Andersen
University of California, San Diego
Dept. of Mathematics
La Jolla, CA 92093-0112
randerse@math.ucsd.edu

Fan Chung
University of California, San Diego
Dept. of Mathematics
La Jolla, CA 92093-0112
fan@ucsd.edu

Kevin Lang
Yahoo! Research
2821 Mission College Blvd.
Santa Clara, CA 95054
langk@yahoo-inc.com

Abstract

A local graph partitioning algorithm finds a cut near a specified starting vertex, with a running time that depends largely on the size of the small side of the cut, rather than the size of the input graph. In this paper, we present a local partitioning algorithm using a variation of PageRank with a specified starting distribution. 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 with a certain starting distribution 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. In particular, we can find a cut with conductance at most ϕ, whose small side has volume at least 2^b, in time $O(2^b \log^2 m/\phi^2)$ where m is the number of edges in the graph. By combining small sets found by this local partitioning algorithm, we obtain a cut with conductance ϕ and approximately optimal balance in time $O(m \log^4 m/\phi^2)$.

1 Introduction

One of the central problems in algorithmic design is 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 is a large literature of research papers on this topic, with applications in numerous areas.

Partitioning algorithms that find such cuts can be applied recursively to solve more complicated problems, including finding balanced cuts, k-way partitions, and hierarchical clusterings [3, 8, 10, 15, 16]. 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, which produces a cut with approximately optimal conductance, but with no guarantee on the balance.

Recently, Spielman and Teng addressed this problem by introducing a local partitioning algorithm called `Nibble`, which finds a small cut near a specified starting vertex in time proportional to the size of the small side of the cut. The small cuts found by `Nibble` can be combined to form balanced cuts in nearly linear time, and the resulting balanced cut algorithm `Partition` is used as a subroutine for finding multiway partitions, sparsifying graphs, and solving diagonally dominant linear systems [17]. The analysis of the `Nibble` algorithm is based on a mixing result by Lovász and Simonovits [11, 12], which shows that a cut with small conductance can be found by simulating a random walk starting from a single vertex for sufficiently many steps.

In this paper, we present a local graph partitioning algorithm that 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 Section 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 [7]. 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 balanced cuts 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` which does this. For many starting vertices within a set C of conductance $O(\phi^2/\log^2 m)$, this algorithm finds a cut with conductance ϕ and volume $O(k)$ in time $O(k \log^2 m/\phi^2)$, provided that we can guess the volume of the smaller side of the cut within a factor of 2. This improves the algorithm `Nibble`, which runs in time $O(k \log^4 m/\phi^5)$, and requires that C have conductance $O(\phi^3/\log^2 m)$.

We can combine cuts found by `PageRank-Nibble` into a cut whose conductance is at most ϕ, and whose volume is at least half that of any set with conductance $O(\phi^2/\log^2 m)$, in time $O(m \log^4 m/\phi^2)$, where m is the number of edges in the graph. This improves the algorithm `Partition`, which obtains a cut whose volume is at least half that of any set with conductance $O(\phi^3/\log^2 m)$ in time $O(m \log^6 m/\phi^5)$.

2 Preliminaries

Throughout the paper we will consider a graph G that is undirected and unweighted. This graph has a vertex set $V = \{v_1, \ldots, v_n\}$ and an edge set E with m undirected edges. We write $d(v)$ for the degree of vertex v, and let D be the diagonal matrix where $D_{i,i} = d(v_i)$. We let A be the adjacency matrix, where $A_{i,j} = 1$ if and only if there is an edge joining v_i and v_j.

When we consider vectors over the vertex set V, we will write them as row vectors, so the product of a vector p and a matrix A will be written pA. Two vectors we will use frequently are the stationary distribution,

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

and the indicator function,

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

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

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

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

2.1 PageRank Vectors

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

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

Here, α is a constant in $(0, 1]$ called the *teleport probability*, s is a vector called the *starting vector*, and W is the lazy random walk transition matrix $W = \frac{1}{2}(I + D^{-1}A)$. This is superficially different from the standard definition of PageRank, which uses the standard random walk matrix $D^{-1}A$ instead of the lazy random walk matrix W, but the two definitions are equivalent up to a change in α. A proof of this is given in the Appendix.

PageRank was introduced by Brin and Page [4, 14], who proposed using PageRank with the starting vector $s = \vec{1}/n$ for search ranking. PageRank vectors whose starting vectors are not uniform, but instead represent a combination of topics and web pages, are called *personalized PageRank vectors*, and have been used to provide personalized search ranking and context-sensitive search [2, 5, 6, 7].

We will consider PageRank vectors where the starting vector is a single vertex. We will also sometimes allow PageRank vectors where the starting vector s has both positive and negative entries, so 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 [6] and [7]). The proofs are given in the Appendix.

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

Proposition 2. *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*

$$R_\alpha = \alpha I + \alpha \sum_{t=1}^{\infty} (1 - \alpha)^t W^t. \tag{2}$$

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

$$\mathrm{pr}_\alpha(s) = \alpha s + \alpha \sum_{t=1}^{\infty} (1 - \alpha)^t \left(sW^t\right). \tag{3}$$

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

2.2 Conductance, sweeps, and mixing

We recall that the *volume* of a subset $S \subseteq V$ of vertices is

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

and the volume of the entire graph is $\mathrm{vol}(V) = 2m$, where m is the number of edges in the graph. The *edge boundary* of a set is defined to be

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

The *conductance* of a set is

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

A *sweep* is a technique for producing a cut from a vector, and is widely used in spectral partitioning [13, 16]. We will use the following degree-normalized version of a sweep. Given a PageRank vector $p = \mathrm{pr}_\alpha(s)$ with support size $N_p = |\mathrm{Supp}(p)|$, let $v_1, \ldots, v_{N_p}$ be an ordering of the vertices from highest to lowest probability-per-degree, so that $p(v_i)/d(v_i) \geq p(v_{i+1})/d(v_{i+1})$. This produces a collection of sets, with one set $S_j^p = \{v_1, \ldots, v_j\}$ for each integer j in $\{1, \ldots, N_p\}$, which we call *sweep sets*. We let $\Phi(p)$ be the smallest conductance of any of these sweep sets,

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

A cut with conductance $\Phi(p)$ can be found by sorting p/d and computing the conductance of each sweep set. This can be done in time $O(\mathrm{vol}(\mathrm{Supp}(p)) + N_p \log N_p)$.

To measure how a vector p is distributed in the graph, we define a function $p\,[x]$ that gives an upper bound on the amount of probability on any set of vertices with volume x. We refer to this function as the Lovász-Simonovits curve, since it was introduced by Lovász and Simonovits [11, 12]. This function is defined for all real numbers x in the interval $[0, 2m]$, and is determined by the amount of probability on the sweep sets; we set

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

and define $p\,[x]$ to be piecewise linear between these points. In other words, 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

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

The function $p\,[x]$ is increasing and concave. It is not hard to see that $p\,[x]$ is an upper bound on the amount of probability from p on any set with volume x; for any set S, we have

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

As an example of the notation we will use throughout the paper, the PageRank vector with teleport probability α and starting vector χ_v is written $\text{pr}_\alpha(\chi_v)$. If we let $p = \text{pr}_\alpha(\chi_v)$, the amount of probability from this PageRank vector on a set S is written as either $p\,(S)$ or $[\text{pr}_\alpha(\chi_v)]\,(S)$, and the value of the Lovász-Simonovits curve at the point $x = \text{vol}(S)$ is written $p\,[\text{vol}(S)]$.

3 Computing approximate PageRank vectors

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

Definition 1. *An ϵ-approximate PageRank vector for $\text{pr}_\alpha(s)$ is a PageRank vector $\text{pr}_\alpha(s-r)$ where the vector r is nonnegative and satisfies $r(v) \le \epsilon d(v)$ for every vertex v in the graph.*

The difference between an ϵ-approximate PageRank vector $\text{pr}_\alpha(s-r)$ and the PageRank vector $\text{pr}_\alpha(s)$ on a set S can be bounded in terms of ϵ and $\text{vol}(S)$.

Lemma 1. *For any ϵ-approximate PageRank vector $\text{pr}_\alpha(s-r)$, and any set S of vertices,*

$$[\text{pr}_\alpha(s)](S) \ge [\text{pr}_\alpha(s-r)](S) \ge [\text{pr}_\alpha(s)](S) - \epsilon\text{vol}(S).$$

We omit the proof, which can be found in the full version of the paper [1]. In this section, we give an algorithm `ApproximatePR`(s, α, ϵ) for computing an ϵ-approximate PageRank vector with small support. The running time of the algorithm depends on ϵ and α, but is independent of the size of the graph.

Theorem 1. *The algorithm* `ApproximatePR`(s, α, ϵ) *has the following properties. For any starting vector s with $\|s\|_1 \le 1$, and any constant $\epsilon \in (0, 1]$, the algorithm computes an ϵ-approximate PageRank vector p for $\text{pr}_\alpha(s)$. The support of p satisfies $\text{vol}(\text{Supp}(p)) \le \frac{2}{(1-\alpha)\epsilon}$, and the running time of the algorithm is $O(\frac{1}{\epsilon\alpha})$.*

The proof of Theorem 1 is based on a series of facts which we describe below. The starting point is the observation that the PageRank operator commutes with the lazy walk matrix W,

$$\text{pr}_\alpha(s)W = \text{pr}_\alpha(sW). \tag{4}$$

By combining Equation (4) with the equation that defines PageRank, we derive the following equation,

$$\begin{aligned}\text{pr}_\alpha(s) &= \alpha s + (1-\alpha)\text{pr}_\alpha(s)W \\ &= \alpha s + (1-\alpha)\text{pr}_\alpha(sW).\end{aligned} \tag{5}$$

Jeh and Widom derived this equation, and showed that it provides a flexible way to compute many PageRank vectors simultaneously [7]. A similar approach was used by Berkhin [2]. 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 which 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 $r = s$, and applies a series of push operations which move probability from r to p while maintaining the invariant $p = \text{pr}_\alpha(s-r)$. Each push operation takes 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 lazy random walk step to the vector $(1-\alpha)r(u)\chi_u$. This operation is defined more formally below. It can be verified using Equation (5) that this push operation does maintain the invariant $p = \text{pr}_\alpha(s-r)$.

```
push (u):
Let $p' = p$ and $r' = r$, except for these changes:
    1. $p'(u) = p(u) + \alpha r(u)$.
    2. $r'(u) = (1-\alpha)r(u)/2$.
    3. For each vertex $v$ such that $(u,v) \in E$:
       $r'(v) = r(v) + (1-\alpha)r(u)/(2d(u))$.
```

During each push operation, some probability is moved from r to p, where it remains. Our algorithm performs pushes only on vertices where $r(u) \ge \epsilon d(u)$, which ensures that a significant amount of probability is moved at each step, and allows us to bound the number of pushes required to compute an ϵ-approximate PageRank vector.

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

This algorithm can be implemented by maintaining a queue containing those vertices u satisfying $r(u) \geq \epsilon d(u)$. At each step, a push operation is performed on the first vertex u in the queue. If $r(u)$ is still at least $\epsilon d(u)$ after the push is performed, then u is placed at the back of the queue, otherwise u is 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 satisfy $r(u) < \epsilon d(u)$. We now show that this algorithm has the properties promised in Theorem 1.

Proof of Theorem 1. Each push operation preserves the property $p = \mathrm{pr}_\alpha(s - r)$, 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 $\alpha\epsilon d_i$. Since $\|r\|_1$ was at most 1 initially, we must have $\alpha\epsilon \sum_{i=1}^{T} d_i \leq 1$, so

$$\sum_{i=1}^{T} d_i \leq \frac{1}{\epsilon\alpha}. \tag{6}$$

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 (6).

To bound the support volume, notice that for each vertex v in $\mathrm{Supp}(p)$, there is some probability remaining on $r(v)$ when the algorithm terminates. In fact, we must have $r(v) \geq ((1-\alpha)/2) \cdot \epsilon d(v)$, because when the last push operation was performed at vertex v, $r(v)$ was at least $\epsilon d(v)$, and a $\frac{1-\alpha}{2}$ fraction of that probability remained on $r(v)$. It follows that $\mathrm{vol}(\mathrm{Supp}(p)) \leq \frac{2}{(1-\alpha)\epsilon}$. □

4 A mixing result for PageRank vectors

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 $\mathrm{pr}_\alpha(s)[x]$ which 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 2. *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

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

In the remainder of this section, we will outline the proof of this theorem and the more general mixing result from which it is derived. When we eventually apply this theorem, we will apply it to an ϵ-approximate PageRank vector. For the conditions of the theorem to hold, ϵ must be small enough that the approximate PageRank vector contains more probability than the stationary distribution on some set of vertices.

The first step toward the proof is to consider how probability moves when a lazy random step is applied to an arbitrary vector p. To do so, we view each undirected edge $\{u, v\}$ as a pair of directed edges (u, v) and (v, u). For each directed edge (u, v) we let

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

and for any set of directed edges A, we define

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

When a lazy walk step is applied to the vector p, the amount of probability that moves from u to v is $\frac{1}{2}p(u, v)$. To measure how much probability moves into and out of a given set S, we define the set of directed edges into S,

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

and the set of directed edges out of S,

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

It is not hard to check that for any vector p, and any set S of vertices,

$$\begin{aligned} pW(S) &= \frac{1}{2}p(\mathrm{in}(S)) + \frac{1}{2}p(\mathrm{out}(S)) \qquad (7) \\ &= \frac{1}{2}p\left(\mathrm{in}(S) \cup \mathrm{out}(S)\right) + \frac{1}{2}p\left(\mathrm{in}(S) \cap \mathrm{out}(S)\right). \end{aligned}$$

Now consider a PageRank vector defined by the usual equation,

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

This equation relates $\mathrm{pr}_\alpha(s)$ to $\mathrm{pr}_\alpha(s)W$. By writing $[\mathrm{pr}_\alpha(s)W](S)$ in terms of $\mathrm{pr}_\alpha(s)$ using equation (7), we can relate $\mathrm{pr}_\alpha(s)$ to itself. As a result, we can bound the height of the Lovasz-Simonovits curve for $\mathrm{pr}_\alpha(s)$ in terms of other points on the same curve.

Lemma 2. *If $p = \mathrm{pr}_\alpha(s)$ is a PageRank vector, then for any set S of vertices,*

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

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

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

The preceding lemma shows that for the PageRank vector $p = \mathrm{pr}_\alpha(s)$, the height of the curve $p[x]$ at the point $\mathrm{vol}(S^p_j)$ is at most the average of the heights of the same curve at the two points $\mathrm{vol}(S^p_j) - |\partial(S^p_j)|$ and $\mathrm{vol}(S^p_j) + |\partial(S^p_j)|$, plus a small term that depends on α. Using this observation, we can give a series of upper bounds on the curve. The result is the following theorem, which states 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^p_j where $p\left(S^p_j\right) - \psi_V(S)$ is fairly large, and where S^p_j has small conductance.

Theorem 3. *Let $p = \mathrm{pr}_\alpha(s)$ 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:*

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

where $\overline{\mathrm{vol}}(S) = \min(\mathrm{vol}(S), 2m - \mathrm{vol}(S))$, or else there exists a sweep cut S^p_j, for some $j \in [1, |\mathrm{Supp}(p)|]$, with the following properties:

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

$$p\left(S^p_j\right) - \psi_V(S^p_j) > \gamma + \alpha t + \sqrt{\overline{\mathrm{vol}}(S^p_j)}\left(1 - \frac{\phi^2}{8}\right)^t.$$

The proofs of Lemma 2 and Theorem 3 are included in the full version of the paper. We will now derive Theorem 2.

Proof of Theorem 2. Let $\phi = \Phi(\mathrm{pr}_\alpha(s))$. Theorem 3 implies that for any integer $t \geq 0$ and any $k \in [0, 2m]$,

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

If we set

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

then we have

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

This gives the bound

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

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

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

The result follows by solving for ϕ. □

5 Cuts from PageRank vectors

Consider the following procedure: pick a starting vertex v and a value of α, compute an ϵ-approximate PageRank vector for $\mathrm{pr}_\alpha(s)$, and perform a sweep over the resulting approximation. In this section, we show that for any set C of conductance $O(\alpha)$, and for many of the vertices v within C, this procedure finds a set with conductance $O(\sqrt{\alpha \log(\mathrm{vol}(C))})$. To prove this, we identify a set of vertices v within C for which we can give a lower bound on the amount of probability from $\mathrm{pr}_\alpha(\chi_v)$ on the set C.

Theorem 4. *For any set C and any constant α in $(0,1]$, there is a subset $C_\alpha \subseteq C$, with $\mathrm{vol}(C_\alpha) \geq \mathrm{vol}(C)/2$, such that for any vertex $v \in C_\alpha$, the PageRank vector $\mathrm{pr}_\alpha(\chi_v)$ satisfies*

$$[\mathrm{pr}_\alpha(\chi_v)](C) \geq 1 - \frac{\Phi(C)}{\alpha}.$$

The proof is in the full version of the paper. We can give a similar bound for an ϵ-approximate PageRank vector $\text{pr}_\alpha(\chi_v - r)$ using Lemma 1. If v is a vertex in C_α, then

$$[\text{pr}_\alpha(\chi_v - r)](C) \geq 1 - \frac{\Phi(C)}{\alpha} - \epsilon\text{vol}(C).$$

If both $\Phi(C)/\alpha$ and ϵ are small, there is a significant amount of probability on the set C, so we can apply the mixing result from Theorem 2 to show that a sweep over $\text{pr}_\alpha(\chi_v - r)$ finds a cut with small conductance. This gives the following guarantee on the procedure proposed above.

Theorem 5. *Let α be a constant in $(0, 1]$, and let C be a set satisfying*

1. $\Phi(C) \leq \alpha/10$,
2. $\text{vol}(C) \leq \frac{2}{3}\text{vol}(G)$.

If $\tilde{p} = \text{pr}_\alpha(\chi_v - r)$ is an ϵ-approximate PageRank vector where $v \in C_\alpha$ and $\epsilon \leq \frac{1}{10\text{vol}(C)}$, then a sweep over $\tilde{p}$ produces a cut with conductance $\Phi(\tilde{p}) = O(\sqrt{\alpha \log(\text{vol}(C))})$.

Proof of Theorem 5. Let $\tilde{p} = \text{pr}_\alpha(\chi_v - r)$ be an ϵ-approximate PageRank vector for $\text{pr}_\alpha(\chi_v)$ satisfying the assumptions of the theorem. Combining Theorem 4 with Lemma 1 gives the following lower bound on $\tilde{p}(C)$.

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

Since $\Phi(C)/\alpha \leq 1/10$ and $\epsilon \leq 1/(10\text{vol}(C))$, 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 2 then implies

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

□

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

Corollary 1. *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 $\text{pr}_\alpha(\chi_v)$, where $\alpha = 10\Phi_G$, $v \in C^{\text{opt}}$, and $\epsilon \leq \frac{1}{10\text{vol}(C^{\text{opt}})}$, then*

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

Corollary 1 follows from Theorem 5 by setting $C = C^{\text{opt}}$.

6 Finding small cuts in nearly linear time

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 $1/\text{vol}(C)$. This requires time proportional to $\text{vol}(C)$, but the resulting cut S may have much smaller volume. This leaves us with a problem similar to the one facing recursive 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($v, \phi, b$):
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 $\epsilon$-approximate PageRank vector
   $p = \text{pr}_\alpha(\chi_v - r)$, with $\epsilon \leq \left(2^b \cdot 48\lceil \log m \rceil\right)^{-1}$.
3. For each $j \in [1, |\text{Supp}(p)|]$,
   check whether $S_j^p$ obeys the following conditions:
       conductance: $\Phi(S_j^p) < \phi$,
       volume: $2^{b-1} < \text{vol}(S_j^p) < \frac{2}{3}\text{vol}(G)$,
       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 6. `PageRank-Nibble`(v, ϕ, b) *can be implemented with running time* $O(2^b \frac{\log^2 m}{\phi^2})$.

Proof of Theorem 6. An ϵ-approximate PageRank vector p with $\epsilon \le \left(2^b \cdot 48\lceil \log m \rceil\right)^{-1}$ can be computed in time $O(2^b \frac{\log m}{\alpha})$ using `ApproximatePR`. By Theorem 1, 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, 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

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

$\square$

Theorem 7. *Let C be a set such that $\mathrm{vol}(C) \le \frac{1}{2}\mathrm{vol}(G)$ and $\Phi(C) \le \phi^2/(22500 \log^2 100m)$, 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: $\mathrm{vol}(S \cap C) > 2^{b-2}$.

The proof of Theorem 7 is included in the full version of the paper.

7 Local graph partitioning

`PageRank-Nibble` improves the running time and approximation ratio of the `Nibble` algorithm of Spielman and Teng [17]. In their 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 optimal 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 algorithm takes as input a parameter ϕ and a graph, and has expected running time $O(m \log^4 m/\phi^2)$. If there exists a set C with $\Phi(C) = O(\phi^2/\log^2 m)$, then with high probability `PageRank-Partition` finds a set S such that $\mathrm{vol}(S) \ge \mathrm{vol}(C)/2$ and $\Phi(S) \le \phi$.

In the table below, we compare our local partitioning algorithms with the existing ones. The running times are stated in terms of ϕ, which is the conductance of the cut returned by the algorithm. The approximation ratios are described by stating what $\Phi(C)$ must be to guarantee that the algorithm will find a cut of conductance ϕ with high probability.

	Running time	Approximation
Nibble	$2^b \log^4 m/\phi^5$	$\phi^3/\log^2 m$
PR-Nibble	$2^b \log^2 m/\phi^2$	$\phi^2/\log^2 m$
Partition	$m \log^6 m/\phi^5$	$\phi^3/\log^2 m$
PR-Partition	$m \log^4 m/\phi^2$	$\phi^2/\log^2 m$

Finding balanced cuts in nearly linear time with `PageRank-Partition` is one important application of our local partitioning techniques. Recently, Khandekar, Rao, and Vazirani [9], introduced an algorithm which produces balanced cuts quickly using a different method. Their algorithm produces an $O(\log^2 n)$ approximation for the balanced cut problem using $O(\log^4 n)$ single commodity flow computations. Both of these algorithms can be used in the linear system solver of Spielman and Teng.

References

[1] R. Andersen, F. Chung, and K. J. Lang. Graph partitioning using pagerank vectors (full version), A preliminary version is available from the authors' homepages.

[2] P. Berkhin. Bookmark-coloring approach to personalized pagerank computing. *Internet Mathematics*, to appear.

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

[4] 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.

[5] 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.

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

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

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

[9] 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.

[10] 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.

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

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

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

[14] 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.

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

[16] 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.

[17] 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.

8 Appendix

The following proposition shows that the definition of PageRank used in this paper, which uses the lazy random walk matrix $W = \frac{1}{2}(I + D^{-1}A)$, is equivalent to the standard definition, which uses the random walk matrix $M = D^{-1}A$.

Proposition 3. *Let $p = \mathrm{pr}_\alpha(s)$ be the unique solution to the equation*

$$p = \alpha + (1-\alpha)pW,$$

where $W = \frac{1}{2}(I + D^{-1}A)$. Then p is also the unique solution of the traditional PageRank equation

$$p = \alpha' + (1-\alpha')pM,$$

where $\alpha' = 2\alpha/(1+\alpha)$ and $M = D^{-1}A$.

Proof. The proof is just algebra.

$$\begin{aligned}\mathrm{pr}_\alpha(s) &= \alpha s + (1-\alpha)\mathrm{pr}_\alpha(s)W \\ &= \alpha s + (\frac{1-\alpha}{2})\mathrm{pr}_\alpha(s) + (\frac{1-\alpha}{2})\mathrm{pr}_\alpha(s)(D^{-1}A).\end{aligned}$$

This implies

$$(\frac{1+\alpha}{2})\mathrm{pr}_\alpha(s) = \alpha s + (\frac{1-\alpha}{2})\mathrm{pr}_\alpha(s)(D^{-1}A),$$

and so

$$\mathrm{pr}_\alpha(s) = (\frac{2\alpha}{1+\alpha})s + (1 - \frac{2\alpha}{1+\alpha})\mathrm{pr}_\alpha(s)(D^{-1}A).$$

The result follows. □

Proof of Proposition 1. The solutions of the PageRank equation $p = \alpha s + (1-\alpha)pW$ are the solutions of the linear system $p\,(I - (1-\alpha)W) = \alpha s$. The matrix $(I - (1-\alpha)W)$ is nonsingular, since it is strictly diagonally dominant, so this equation has a unique solution. □

Proof of Proposition 2. The sum that defines R_α in equation (2) is absolutely convergent for $\alpha \in (0,1]$, and the following computation shows that sR_α obeys the steady state equation for $\mathrm{pr}_\alpha(s)$.

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

Since the solution to the this equation is unique by Proposition 1, it follows that $\mathrm{pr}_\alpha(s) = sR_\alpha$. □