A Local Clustering Algorithm for Massive Graphs and its Application to Nearly-Linear Time Graph Partitioning

Daniel A. Spielman
Department of Computer Science
Program in Applied Mathematics
Yale University

Shang-Hua Teng
Department of Computer Science
Boston University

September 18, 2008

Abstract

We study the design of local algorithms for massive graphs. A local algorithm is one that finds a solution containing or near a given vertex without looking at the whole graph. We present a local clustering algorithm. Our algorithm finds a good cluster—a subset of vertices whose internal connections are significantly richer than its external connections—near a given vertex. The running time of our algorithm, when it finds a non-empty local cluster, is nearly linear in the size of the cluster it outputs.

Our clustering algorithm could be a useful primitive for handling massive graphs, such as social networks and web-graphs. As an application of this clustering algorithm, we present a partitioning algorithm that finds an approximate sparsest cut with nearly optimal balance. Our algorithm takes time nearly linear in the number edges of the graph.

Using the partitioning algorithm of this paper, we have designed a nearly-linear time algorithm for constructing spectral sparsifiers of graphs, which we in turn use in a nearly-linear time algorithm for solving linear systems in symmetric, diagonally-dominant matrices. The linear system solver also leads to a nearly linear-time algorithm for approximating the second-smallest eigenvalue and corresponding eigenvector of the Laplacian matrix of a graph. These other results are presented in two companion papers.

*This paper is the first in a sequence of three papers expanding on material that appeared first under the title “Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems” [ST03]. The second paper, “Spectral Sparsification of Graphs” [ST08b] contains further results on partitioning graphs, and applies them to producing spectral sparsifiers of graphs. The third paper, “Nearly-Linear Time Algorithms for Preconditioning and Solving Symmetric, Diagonally Dominant Linear Systems” [ST08a] contains the results on solving linear equations and approximating eigenvalues and eigenvectors.

This material is based upon work supported by the National Science Foundation under Grant Nos. 0325630, 0634957, 0635102 and 0707522. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.
1 Introduction

Given a vertex of interest in a massive graph, we would like to find a small cluster around that vertex, in time proportional to the size of the cluster. The algorithm we introduce will solve this problem while only examining vertices near the initial vertex, under some reasonable notion of nearness. We call such an algorithm a local algorithm.

Our local clustering algorithm provides a very powerful primitive for the design of fast graph algorithms. In Section 3 of this paper, we use it to design the first nearly-linear time algorithm for graph partitioning that produces a partition of nearly-optimal balance among those approximating a target conductance. In the papers [ST08] and [ST08a], we proceed to use this graph partitioning algorithm to design nearly-linear time algorithms for sparsifying graphs and for solving symmetric, diagonally-dominant linear systems.

1.1 Local Clustering

We say that a graph algorithm is a local algorithm if it is given a particular vertex as input, and at each step after the first only examines vertices connected to those it has seen before. The use of a local algorithm naturally leads to the question of in which order one should explore the vertices of a graph. While it may be natural to explore vertices in order of shortest-path distance from the input vertex, such an ordering is a poor choice in graphs of low-diameter, such as social network graphs [LH08]. We suggest first processing the vertices that are most likely to occur in short random walks from at the input vertex. That is, we consider a vertex to be near the input vertex if it is likely to appear in a short random walk from the input vertex.

In Section 3, we use a local graph exploration process to find a cluster that is near the input vertex. Following Kannan, Vempala and Vetta [KVV04], we say that a set of vertices is a good cluster if it has low conductance; that is, if it has many more external than internal edges. We give an efficient local clustering algorithm, Nibble, that runs in time proportional to the size of the cluster it outputs. Although our algorithm may not find a local cluster for some input vertices, we will show that it is usually successful. In particular, we prove the following theorem: There exists a constant $\alpha > 0$ such that for any target conductance $\phi$ and any cluster $C_0$ of conductance at most $\alpha \cdot \phi^2 / \log^3 n$, when given a random vertex $v$ sampled according to degree inside $C_0$, Nibble will return a cluster $C$ mostly inside $C_0$ and with conductance at most $\phi$, with probability at least $1/2$.

The local clustering algorithm Nibble makes a novel use of random walks. For a positive integer $t$, suppose $p_{t,v}$ is the probability distribution of the $t$-step random walk starting at $v$. As the support of $p_{t,v}$—the set of nodes with positive probability—could grow rapidly, Nibble maintains a truncated version of the distribution. At each step of the truncated random walks, Nibble looks for a cluster among only nodes with high probability. The truncation is critical to ensure that the clustering algorithm is output sensitive. It guarantees that the size of the support of the distribution that Nibble maintains is not too much larger than the size of the cluster it produces. The cluster that Nibble produces is local to the starting vertex $v$ in the sense that it consists of nodes that are among the most favored destinations of random walks starting from $v$.

By using the personal PageRank vector [PBMW98] to define nearness, Andersen, Chung and Lang [ACL06], have produced an improved version of our algorithm Nibble, which they call...
PageRank-Nibble. Following this work, other local algorithms have been designed by Andersen et. al. [ABC+07] for approximately computing Personal PageRank vectors, by Andersen [And08] for finding dense subgraphs and by Andersen, Chung and Lang [ACL07] for partitioning directed graphs.

1.2 Nearly Linear-Time Algorithms

Our local clustering algorithm provides a powerful tool for designing fast graph algorithms. In this paper and its two companion papers, we show how to use it to design randomized, nearly linear-time algorithms for several important graph-theoretic and numerical problems.

The need for algorithms whose running time is linear or nearly linear in their input size has increased as algorithms handle larger inputs. For example, in circuit design and simulation, an Intel Dual Core Itanium processor has more than one billion transistors, which is more than 100 times the number of transistors that the Pentium had in 2000 [Cor05]; in scientific computing, one often needs to solve linear systems that involve hundreds of millions of variables [SLM+02]; in modern information infrastructure, the web has grown into a graph of hundreds billions of nodes [GS05]. As a result of this rapid growth in problem size, what used to be considered an efficient algorithm, such as a $O(n^{1.5})$-time algorithm, may no longer be adequate for solving problems of these scales. Space complexity poses an even greater problem.

Many basic graph-theoretic problems such as connectivity and topological sorting can be solved in linear or nearly-linear time. The efficient algorithms for these problems are built on linear-time primitives such as Breadth-First-Search (BFS) and Depth-First-Search (DFS). Minimum Spanning Trees (MST) and Shortest-Path Trees are examples of other commonly used nearly linear-time primitives. We hope to build up the library of nearly-linear time graph algorithms that may be used as primitives. While the analyzable variants of the algorithms we present here, and even their improved versions by Andersen, Chung and Lang [ACL06], may not be immediately useful in practice, we believe practical algorithms may be derived from them by making less conservative choices of parameters.

Our local clustering algorithm provides an exciting new primitive for developing nearly linear-time graph algorithms. Because its running time is proportional to the size of the cluster it produces, we can repeatedly apply it remove many clusters from a graph, all within nearly-linear time.

In the second part of this paper, we use Nibble as a subroutine to construct a randomized graph partitioning algorithm that runs in nearly-linear time. To the best of our knowledge, this is the first nearly linear-time partitioning algorithm that finds an approximate sparsest cut with approximately optimal balance. In our first companion paper [ST08b], we apply this new partitioning algorithm to develop a nearly-linear-time algorithm for producing spectral sparsifiers of graphs. We begin that paper by extending the partitioning algorithm of this paper to obtain a stronger guarantee on its output: if it outputs a small set, then the complement must be contained in a subgraph whose conductance is higher than the target.

2 Clusters and Conductance

Let $G = (V, E)$ be an undirected graph with $V = \{1, \ldots, n\}$. A cluster of $G$ is a subset of $V$ that is richly intra-connected but sparsely connected with the rest of the graph. The quality of a
cluster can be measured by its conductance, the ratio of the number of its external connections to the number of its total connections.

We let $d(i)$ denote the degree of vertex $i$. For $S \subseteq V$, we define $\mu(S) = \sum_{i \in S} d(i)$ (often called the volume of $S$). So, $\mu(V) = 2|E|$. Let $E(S, V - S)$ be the set of edges connecting a vertex in $S$ with a vertex in $V - S$. We define the conductance of a set of vertices $S$, written $\Phi(S)$ by

$$\Phi(S) \overset{\text{def}}{=} \frac{|E(S, V - S)|}{\min(\mu(S), \mu(V - S))}.$$ 

The conductance of $G$ is then given by

$$\Phi_G \overset{\text{def}}{=} \min_{S \subseteq V} \Phi(S).$$

We sometime refer to a subset $S$ of $V$ as a cut of $G$ and refer to $(S, V - S)$ as a partition of $G$. The balance of a cut $S$ or a partition $(S, V - S)$ is then equal to

$$\text{bal}(S) = \min(\mu(S), \mu(V - S))/\mu(V).$$

We call $S$ a sparsest cut of $G$ if $\Phi(S) = \Phi_G$ and $\mu(S)/\mu(V) \leq 1/2$.

In the construction of a partition of $G$, we will be concerned with vertex-induced subgraphs of $G$. However, when measuring the conductance and volumes of vertices in these vertex-induced subgraphs, we will continue to measure the volume according to the degrees of vertices in the original graph. For clarity, we define the conductance of a set $S$ in the subgraph induced by $A \subseteq V$ by

$$\Phi^G_A(S) \overset{\text{def}}{=} \frac{|E(S, A - S)|}{\min(\mu(S), \mu(A - S))},$$

and

$$\Phi^G_A \overset{\text{def}}{=} \min_{S \subseteq A} \Phi^G_A(S).$$

For convenience, we define $\Phi^G_A(\emptyset) = 1$ and, for $|A| = 1$, $\Phi^G_A = 1$.

For $A \subseteq V$, we let $G(A)$ denote the subgraph of $G$ induced by the vertices in $A$. We introduce the notation $G[A]$ to denote graph $G(A)$ to which self-loops have been added so that every vertex in $G[A]$ has the same degree as in $G$. Each self-loop adds 1 to the degree. We remark that if $G(A)$ is the subgraph of $G$ induced on the vertices in $A$, then

$$\Phi^G_A \leq \Phi_{G(A)}.$$ 

So, when we prove lower bounds on $\Phi^G_A$, we obtain lower bounds on $\Phi_{G(A)}$.

Clustering is an optimization problem: Given an undirected graph $G$ and a conductance parameter, find a cluster $C$ such that $\Phi(C) \leq \phi$, or determine no such cluster exists. The problem is NP-complete (see, for example [LR99] or [SS06]). But, approximation algorithms exist. Leighton and Rao [LR99] used linear programming to obtain $O(\log n)$-approximations of the sparsest cut. Arora, Rao and Vazirani [ARV04] improved this to $O(\sqrt{\log n})$ through semi-definite programming. Faster algorithms obtaining similar guarantees have been constructed by Arora, Hazan and Kale [AHK04], Khandekar, Rao and Vazirani [KRV06], Arora and Kale [AK07], and Orecchia, Schulman, Vazirani, and Vishnoi [OSV08].
2.1 The Algorithm Nibble

The algorithm Nibble works by approximately computing the distribution of a few steps of the random walk starting at a seed vertex \( v \). It is implicit in the analysis of the volume estimation algorithm of Lovász and Simonovits \[LS93\] that one can find a cut with small conductance from the distributions of the steps of the random walk starting at any vertex from which the walk does not mix rapidly. We will observe that a random vertex in a set of low conductance is probably such a vertex. We then extend the analysis of Lovász and Simonovits to show one can find a cut with small conductance from approximations of these distributions, and that these approximations can be computed quickly. In particular, we will truncate all small probabilities that appear in the distributions to 0. In this way, we reduce the work required to compute our approximations.

For the rest of this section, we will work with a graph \( G = (V,E) \) with \( n \) vertices and \( m \) edges, so that \( \mu(V) = 2m \). We will allow some of these edges to be self-loops. Except for the self-loops, which we allow to occur with multiplicities, the graph is assumed to be unweighted. We will let \( A \) be the adjacency matrix of this graph. That is,

\[
A(u,v) = \begin{cases} 
1 & \text{if } (u,v) \in E \text{ and } u \neq v \\
k & \text{if } u = v \text{ and this vertex has } k \text{ self-loops} \\
0 & \text{otherwise.}
\end{cases}
\]

We define the following two vectors supported on a set of vertices \( S \):

\[
\chi_S(u) = \begin{cases} 
1 & \text{for } u \in S, \\
0 & \text{otherwise,}
\end{cases}
\]

\[
\psi_S(u) = \begin{cases} 
d(u)/\mu(S) & \text{for } u \in S, \\
0 & \text{otherwise.}
\end{cases}
\]

We will consider the random walk that at each time step stays at the current vertex with probability \( 1/2 \), and otherwise moves to the endpoint of a random edge attached to the current vertex. Thus, self-loops increase the chance the walk stays at the current vertex. For example, if a vertex has 4 edges, one of which is a self-loop, then when the walk is at this vertex it has a 5/8 chance of staying at that vertex, and a 1/8 chance of moving to each of its 3 neighbors.

The matrix realizing this walk can be expressed by \( M = (AD^{-1} + I)/2 \), where \( d(i) \) is the degree of node \( i \), and \( D \) is the diagonal matrix with diagonal entries \( (d(1),\ldots,d(n)) \). Typically, a random walk starts at a node \( v \). In this case, the distribution of the random walk at time \( t \) evolves according to \( p_t = M^t \chi_v \).

We note that \( \psi_V \) is the steady-state distribution of the random walk, and that \( \psi_S \) is the restriction of that walk to the set \( S \).

We will use the truncation operation defined by

\[
[p]_\epsilon(u) = \begin{cases} 
p(u) & \text{if } p(u) \geq d(u)\epsilon, \\
0 & \text{otherwise.}
\end{cases}
\]
Our algorithm, \texttt{Nibble}, will generate the sequence of vectors starting at $\chi_v$ by the rules

$$q_t = \begin{cases} 
\chi_v & \text{if } t = 0, \\
Mr_{t-1} & \text{otherwise,}
\end{cases}$$

(1)

$$r_t = [q_t]_e.$$  

(2)

That is, at each time step, we will evolve the random walk one step from the current density, and then round every $q_t(u)$ that is less than $d(u)e$ to 0. Note that $q_t$ and $r_t$ are not necessarily probability vectors, as their components may sum to less than 1.

In the statement of the algorithm and its analysis, we will use the following notation. For a vector $p$, we let $S_j(p)$ be the set of $j$ vertices $u$ maximizing $p(u)/d(u)$, breaking ties lexicographically. That is, $S_j(p) = \{\pi(1), \ldots, \pi(j)\}$ where $\pi$ is the permutation such that

$$p(\pi(i))/d(\pi(i)) \geq p(\pi(i+1))/d(\pi(i+1))$$

for all $i$, and $\pi(i) < \pi(i+1)$ when these two ratios are equal. We then set

$$\lambda_j(p) = \mu(S_j(p)) = \sum_{u \in S_j(p)} d(u).$$

Note that $\lambda_n(p)$ always equals $2m$.

Following Lovász and Simonovits \cite{LS90}, we set

$$I(p, x) = \max_{w \in [0,1]^n} \sum_{u \in V} w(u)p(u).$$

(3)

This function $I(p, \cdot)$ is essentially the same as the function $h$ defined by Lovász and Simonovits—it only differs by a linear transformation.

We remark that for $x = \lambda_j(p)$, $I(p, x) = p(S_j(p))$, and that $I(p, x)$ is linear in $x$ between these points. Finally, we let $I_x(p, x)$ denote the partial derivative of $I(p, x)$ with respect to $x$, with the convention that for $x = \lambda_j(p)$,

$$I_x(p, x) = \lim_{\delta \to 0} I_x(p, x - \delta) = p(\pi(j))/d(\pi(j)),$$

where $\pi$ is the permutation specified above so that $\pi(j) = S_j(p) - S_{j-1}(p)$.

As $p(\pi(i))/d(\pi(i))$ is non-increasing, $I_x(p, x)$ is a non-increasing function in $x$ and $I(p, x)$ is a concave function in $x$.

During the course of our exposition, we will need to set many constants, which we collect here for convenience. For each, we provide a suitable value and indicate where it is first used in the paper.

| constant | value | where first used |
|----------|-------|-----------------|
| $c_1$    | 200   | (12)            |
| $c_2$    | 280   | (15)            |
| $c_3$    | 1800  | (16)            |
| $c_4$    | 140   | \texttt{Nibble}, line C.4 |
| $c_5$    | 20    | Definition 2.11 |
| $c_6$    | 60    | Definition 2.11 |
The following is an exhaustive list of the inequalities we require these constants to satisfy.

\[ c_2 \geq 2c_4 \quad (4) \]
\[ c_6 \geq 2c_5 \quad (5) \]
\[ c_3 \geq 8c_5 \quad (6) \]
\[ c_4 \geq 4c_5 \quad (7) \]
\[ \frac{1}{2c_6} - \frac{1}{c_3} - \frac{1}{2c_5c_6} \geq \frac{1}{c_4} \quad (8) \]
\[ \frac{1}{2c_5} \geq \frac{6}{5c_6} + \frac{1}{c_1} \quad (9) \]
\[ \frac{1}{5} \geq \frac{1}{c_5} + \frac{4c_6}{3c_3} + \frac{1}{2c_1} + \frac{1}{2c_2} \quad (10) \]

Given a \( \phi \), we set constants that will play a prominent role in our analysis:

\[ \ell \overset{\text{def}}{=} \lceil \log_2 (\mu (V) / 2) \rceil, \quad (11) \]
\[ t_1 \overset{\text{def}}{=} \left\lceil \frac{2}{\phi^2} \ln \left( c_1 (\ell + 2) \sqrt{\mu (V) / 2} \right) \right\rceil, \quad (12) \]
\[ t_h \overset{\text{def}}{=} ht_1, \text{ for } 0 \leq h \leq \ell + 1, \quad (13) \]
\[ t_{last} \overset{\text{def}}{=} (\ell + 1)t_1, \text{ and } \quad (14) \]
\[ f_1(\phi) \overset{\text{def}}{=} \frac{1}{c_2 (\ell + 2)t_{last}}. \quad (15) \]

Note that

\[ f_1(\phi) \geq \Omega \left( \frac{\phi^2}{\log^3 \mu (V)} \right). \]
\[ C = \text{Nibble}(G, v, \phi, b) \]
where \( v \) is a vertex
\( 0 < \phi < 1 \)
b is a positive integer.

1. Set
\[ \epsilon = 1/(c_3(\ell + 2)t_{last}2^b). \] (16)
2. Set \( q_0 = \chi_v \) and \( r_0 = [q_0]_\epsilon \).
3. For \( t = 1 \) to \( t_{last} \)
   (a) Set \( q_t = Mr_{t-1} \)
   (b) Set \( r_t = [q_t]_\epsilon \).
   (c) If there exists a \( j \) such that
       (C.1) \( \Phi(S_j(q_t)) \leq \phi \),
       (C.2) \( \lambda_j(q_t) \leq (5/6)\mu(V) \),
       (C.3) \( 2^b \leq \lambda_j(q_t) \), and
       (C.4) \( I_x(q_t, 2^b) \geq 1/c_4(\ell + 2)2^b \).
       then return \( C = S_j(q_t) \) and quit.
4. Return \( C = \emptyset \).

Condition (C.1) guarantees that the set \( C \) has low conductance. Condition (C.2) ensures that it does not contain too much volume, while condition (C.3) ensures that it does not contain too little. Condition (C.4) guarantees that many elements of \( C \) have large probability mass. While it would be more natural to define condition (C.4) as a constraint on \( I_x(q_t, \lambda_j(q_t)) \) instead of \( I_x(q_t, 2^b) \), our proof of correctness requires the latter.

In the rest of this section, we will prove the following theorem on the performance of \textbf{Nibble}.

\textbf{Theorem 2.1 (Nibble).} \textbf{Nibble} can be implemented so that on all inputs, it runs in time \( O(2^b(\log^b m)/\phi^4) \). Moreover, \textbf{Nibble} satisfies the following properties.

\begin{enumerate}
\item [(N.1)] When \( C = \text{Nibble}(G, v, \phi, b) \) is non-empty,
\[ \Phi(C) \leq \phi \quad \text{and} \quad \mu(C) \leq (5/6)\mu(V). \]
\item [(N.2)] Each set \( S \) satisfying
\[ \mu(S) \leq (2/3)\mu(V) \quad \text{and} \quad \Phi(S) \leq f_1(\phi) \]
has a subset \( S^g \) such that
\begin{enumerate}
\item [(N.2.a)] \( \mu(S^g) \geq \mu(S)/2 \), and
\item [(N.2.b)] \( v \in S^g \) and \( C = \text{Nibble}(G, v, \phi, b) \neq \emptyset \) imply \( \mu(C \cap S) \geq 2^b - 1 \).
\end{enumerate}
\item [(N.3)] The set \( S^g \) may be partitioned into subsets \( S^g_0, \ldots, S^g_\ell \) such that if \( v \in S^g_0 \), then the set \( C \) output by \textbf{Nibble}(\( G, v, \phi, b \)) will not be empty.
\end{enumerate}
2.2 Basic Inequalities about Random Walks

We first establish some basic inequalities that will be useful in our analysis. Readers who are eager to see the analysis of Nibble can first skip this subsection. Suppose $G = (V, E)$ is an undirected graph. Recall $M = (AD^{-1} + I)/2$, where $A$ is the adjacency matrix of $G$.

**Proposition 2.2** (Monotonicity of Mult by $M$). For all non-negative vectors $p$,

$$\|D^{-1}(Mp)\|_\infty \leq \|D^{-1}p\|_\infty.$$  

**Proof.** Applying the transformation $z = D^{-1}p$, we see that it is equivalent to show that for all $z$

$$\|D^{-1}MDz\|_\infty \leq \|z\|_\infty.$$  

To prove this, we note that $D^{-1}MD = D^{-1}(AD^{-1} + I)D/2 = M^T$, and the sum of the entries in each row of this matrix is 1. \hfill \Box 

**Definition 2.3.** For a set $S \subseteq V$, we define the matrix $D_S$ to be the diagonal matrix such that $D_S(u, u) = 1$ if $u \in S$ and 0 otherwise.

**Proposition 2.4.** For every $S \subseteq V$, all non-negative vectors $p$ and $q$, and every $t \geq 1$,

$$p^T(D_SM)^tq \leq p^T M^tq.$$  

**Proof.** For $t = 1$, we observe

$$p^T(M)q = p^T((D_S + D_S)M)q = p^T(D_SM)q + q^T(D_SM)q \geq p^T(D_SM)q,$$

as $p$, $q$, $D_S$, and $M$ are all non-negative. The proposition now follows by induction. \hfill \Box 

**Proposition 2.5** (Escaping Mass). For all $t \geq 0$ and for all $S \subset V$,

$$1^T(D_SM)^t\psi_S \geq 1 - t\Phi_V(S)/2.$$  

**Proof.** Note that $M\psi_S$ is the distribution after a single-step walk from a random vertex in $S$ and $1^T D_S(M\psi_S)$ is the probability that the walk stays inside $S$. Thus, $1^T(D_SM)^t\psi_S$ is the probability that a $t$-step walk starting from a random vertex in $S$ stays entirely in $S$.

We first prove by induction that for all $t \geq 0$,

$$\|D^{-1}(D_SM)^t\psi_S\|_\infty \leq 1/\mu(S).$$ (17)  

The base case, $t = 0$, follows from the fact that $\|D^{-1}\psi_S\|_\infty = 1/\mu(S)$. To complete the induction, observe that if $x$ is a non-negative vector such that $\|D^{-1}x\|_\infty \leq 1/\mu(S)$, then

$$\|D^{-1}(D_SM)x\|_\infty = \|D_SM D^{-1}Mx\|_\infty \leq \|D^{-1}Mx\|_\infty \leq \|D^{-1}M\|_\infty \leq 1/\mu(S),$$

where the second-to-last inequality follows from Proposition 2.2.

We will now prove that for all $t$,

$$1^T(D_SM)^t\psi_S - 1^T(D_SM)^{t+1}\psi_S \leq \Phi_V(S)/2,$$
from which the proposition follows, as $1^T \psi_S = 1$.

Observing that $1^T M = 1^T$, we compute

$$1^T (DSM)^t \psi_S - 1^T (DSM)^{t+1} \psi_S$$

$$= 1^T (I - DSM)(DSM)^t \psi_S$$

$$= 1^T (M - DSM)(DSM)^t \psi_S$$

$$= 1^T (I - DS)M(DSM)^t \psi_S$$

$$= \chi_S^T M(DSM)^t \psi_S$$

$$= (1/2) \chi_S^T (I + AD^{-1})(DSM)^t \psi_S$$

$$= (1/2) \chi_S^T (AD^{-1})(DSM)^t \psi_S$$

(as $\chi_S^T ID_S = 0$)

$$\leq (1/2) |E(S, V - S)| \|D^{-1}(DSM)^t \psi_S\|_\infty$$

$$\leq \frac{1}{2} \frac{|E(S, V - S)|}{\mu(S)}$$

(by inequality (17))

$$\leq \Phi_V(S)/2.$$

\[ \square \]

2.3 The Analysis of Nibble

Our analysis of Nibble consists of three main steps. First, we define the sets $S^g$ mentioned in Theorem 2.1 and establish property (N.2). We then refine the structure of $S^g$ to define sets $S^g_b$ and prove property (N.3). The sets $S^g$ and $S^g_b$ are defined in terms of the distributions of random walks from a vertex in $S$, without reference to the truncation we perform in the algorithm. We then analyze the impact of truncation used in Nibble and extend the theory of Lovász and Simonovits [LS93] to truncated random walks.

Step 1: $S^g$ and its properties

Definition 2.6 ($S^g$). For each set $S \subseteq V$, we define $S^g$ to be the set of nodes $v$ in $S$ such that for all $t \leq t_{last}$,

$$\chi_S^T M^t \chi_v \leq t_{last} \Phi(S).$$

Note that $\chi_S^T M^t \chi_v$ denotes the probability that a $t$-step random walk starting from $v$ terminates outside $S$. Roughly speaking, $S^g$ is the set of vertices $v \in S$ such that a random walk from $v$ it is reasonably likely to still be in $S$ after $t_{last}$ time steps. We will prove the following bound on the volume of $S^g$.

Lemma 2.7 (Volume of $S^g$),

$$\mu(S^g) \geq \mu(S)/2.$$

Proof. Let $S \subseteq V$, and let $D_S$ be the diagonal matrix such that $D_S(u, u) = 1$ if $u \in S$ and 0
otherwise. For \( t \geq 0 \),
\[
\chi_S^T M^t \chi_v = (1 - \chi_S)^T M^t \chi_v \\
= 1^T \chi_v - \chi_S^T M^t \chi_v \\
= 1 - \chi_S^T M^t \chi_v \\
\leq 1 - 1^T (D_S M)^t \chi_v, \quad \text{by Proposition 2.4} \\
\leq 1 - 1^T (D_S M)^{t_{\text{last}}} \chi_v,
\]
as \( 1^T (D_S M)^t \chi_v \) is a non-increasing function of \( t \). Define
\[
S' = \{ v : 1 - 1^T (D_S M)^{t_{\text{last}}} \chi_v \leq t_{\text{last}} \Phi(S) \}.
\]
So, \( S' \subseteq S^g \), and it suffices to prove that \( \mu(S') \geq \mu(S)/2 \).

Applying Proposition 2.5, we obtain
\[
t_{\text{last}} \Phi(S)/2 \geq 1 - 1^T (D_S M)^{t_{\text{last}}} \psi_S \\
= \sum_{v \in S} \frac{d(v)}{\mu(S)} (1 - 1^T (D_S M)^{t_{\text{last}}} \chi_v) \\
> \sum_{v \in S - S'} \frac{d(v)}{\mu(S)} t_{\text{last}} \Phi(S), \quad \text{by the definition of } S' \\
= \frac{\mu(S - S')}{\mu(S)} t_{\text{last}} \Phi(S).
\]
So, we may conclude
\[
\frac{\mu(S - S')}{\mu(S)} < \frac{1}{2},
\]
from which the lemma follows.

We now prove the following lemma, which says that if \texttt{Nibble} is started from any \( v \in S^g \) with parameter \( b \) and returns a non-empty set \( C \), then \( \mu(C \cap S) \geq 2^{b-1} \).

**Lemma 2.8 (N2).** Let \( S \subseteq V \) be a set of vertices such that \( \Phi(S) \leq f_1(\phi) \). If \texttt{Nibble} is run with parameter \( b \), is started at a \( v \in S^g \), and outputs a non-empty set \( C \), then \( \mu(C \cap S) \geq 2^{b-1} \).

**Proof.** For \( v \in S^g \), let \( q_t \) be given by (1) and (2). Then, for \( t \leq t_{\text{last}} \),
\[
\chi_S^T q_t \leq \chi_S^T M^t \chi_v \leq \Phi(S) t_{\text{last}} \leq f_1(\phi) t_{\text{last}} \leq \frac{1}{c_2(\ell + 2)},
\]
where the second inequality follows from the definition of \( S^g \).

Let \( t \) be the index of the step at which the set \( C \) is generated. Let \( j' \) be the least integer such that \( \lambda_{j'}(q_t) \geq 2^b \). Condition (C.3) implies \( j' \leq j \). As \( I_x \) is non-increasing in its second argument and constant between \( 2^b \) and \( \lambda_{j'}(q_t) \), Condition (C.4) guarantees that for all \( u \in S_{j'}(q_t) \),
\[
q_t(u)/d(u) \geq 1/c_4(\ell + 2) 2^b.
\]
Thus,\[
\mu \left( S \setminus (q_t) \cap S \right) = \sum_{u \in S \setminus (q_t) \cap S} d(u) \leq \sum_{u \in S \setminus (q_t) \cap S} c_4(\ell + 2)2^b q_t(u) \\
\leq c_4(\ell + 2)2^b (\chi_S q_t) \leq \frac{c_4(\ell + 2)2^b}{c_2(\ell + 2)} \leq 2^{b-1},
\]
by \[14\]. So, \( \mu \left( S \setminus (q_t) \cap S \right) \geq 2^{b-1} \), and, as \( j' \leq j \),
\[\mu \left( S \setminus (q_t) \cap S \right) \geq \mu \left( S \setminus (q_t) \cap S \right) \geq 2^{b-1}.\]
\[\square\]

Step 2: Refining \( S^q \)

Before defining the sets \( S^q \), we first recall some of the facts we can infer about the function \( I \) from the work of Lovász and Simonovits. These facts will motivate our definitions and analysis.

In the first part of the proof of Lemma 1.4 of [LS93], Lovász and Simonovits prove

**Lemma 2.9.** For every non-negative vector \( p \) and every \( x \),
\[I(Mp, x) \leq I(p, x).\]
(18)

For each \( p_t \), \( I(p_t, x) \) is a concave function that starts at \( (0, 0) \) and goes to \( (\mu(V), 1) \). Lemma 2.9 says that for each \( t \), the curve defined by \( I(p_{t+1}, x) \) lies below the curve defined by \( I(p_t, x) \). In particular,
\[\forall x, I(p_{t+1}, x) \leq I(p_t, x).\]
(19)

If none of the sets \( S_j(p_{t+1}) \) has conductance less than \( \phi \), then Lovász and Simonovits prove a bound on how far below \( I(p_t, x) \) the curve of \( I(p_{t+1}, x) \) must lie. The following Lemma is a special case of Lemma 1.4 of [LS93], restricted to points \( x \) of the form \( \lambda_j(Mp) \). Lovász and Simonovits [LS90] claim that the following is true for all \( x \), but point out in the journal version of their paper [LS93] that this claim was false. Fortunately, we do not need the stronger claim.

**Lemma 2.10.** For any non-negative vector \( p \), if \( \Phi(S_j(Mp)) \geq \phi \), then for \( x = \lambda_j(Mp) \),
\[I(Mp, x) \leq \frac{1}{2} \left( I(p, x - 2\phi \bar{x}) + I(p, x + 2\phi \bar{x}) \right),\]
where \( \bar{x} \) denotes \( \min(x, 2m - x) \).

The mistake in [LS90] is the assertion in the beginning of the proof that the inequality holds for all \( x \) if it holds for all \( x \) of form \( \lambda_j(Mp) \).

When this lemma applies, one may draw a chord across the curve of \( I(p_t, x) \) around \( x \) of width proportional to \( \phi \), and know that \( I(p_{t+1}, x) \) lies below. Thus, we know that if none of the sets \( S_j(p_t) \) has conductance less than \( \phi \), then the curve \( I(p_t, x) \) will approach a straight line. On the other hand, Proposition 2.5 will tell us that some point of \( I(p_{t_{last}}, x) \) lies well above this line (see Lemma 2.14).

We will now define the sets \( S^q \) for \( b = 1, \ldots, \ell \), simultaneously with two quantities—\( h_v \) and \( x_h \), where \( h_v \) is such that Nibble will stop between iterations \( t_{h_{v-1}} \) and \( t_{h_v} \) and \( x_{h_v} \) is the \( x \)-coordinate of a point that may be shown in Lemmas 2.14 and 2.17 to contradict the conclusion of Lemma 2.10 and thereby enable us to find a set of low conductance.
**Definition 2.11** \((x_h, h_v, \text{and } S^q_b)\). Given a \(v \in S^q\), let \(p_t = M^t\chi_v\). For \(0 \leq h \leq \ell + 1\), define 
\(x_h(v)\) to be the real number such that 
\[
I(p_{th}, x_h(v)) = \frac{h + 1}{c_5(\ell + 2)}.
\]
We write \(x_h\) instead of \(x_h(v)\) when \(v\) is clear from context. Define 
\(h_v = \begin{cases} 
\ell + 1 & \text{if } x_\ell(v) \geq 2m/c_5(\ell + 2), \\
\min \{h : x_h \leq 2x_{h-1}\} & \text{otherwise}. 
\end{cases}\)
We define 
\(S^q_0 = \{v : x_{h_v-1}(v) < 2\}\), 
and for \(b = 1, \ldots, \ell\), we define 
\(S^q_b = \{v : x_{h_v-1}(v) \in [2^b, 2^{b+1})\}\).

**Proposition 2.12.** The quantities \(h_v\) are well-defined and the sets \(S^q_b\) partition \(S^q\). Moreover, 
\(x_{h-1} < x_h\) for all \(h\).

**Proof.** It follows from the definition of \(I\) that for a probability vector \(p\) the slope of \(I(p, \cdot)\) is always less than 1, and so \(x_0 \geq 1/c_5(\ell + 2)\). If \(x_\ell < \mu(V)/c_6(\ell + 2)\), then 
\[
x_\ell/x_0 < \frac{\mu(V)c_5(\ell + 2)}{c_6(\ell + 2)} \leq \mu(V)/2, \quad \text{(by inequality (5))}
\]
so there is an integer \(h \leq \ell\) such that \(x_h \leq 2x_{h-1}\), and so the quantities \(h_v\) are well-defined.

To see that the sets \(S^q_b\) partition \(S^q\), it now suffices to observe that \(x_{h_v-1} < \mu(V) \leq 2^{\ell+1}\).

Finally, to show that \(x_{h-1} < x_h\), we apply Lemma 2.9 to show 
\[
I(p_{th}, x_{h-1}) \leq I(p_{th-1}, x_{h-1}) = \frac{h}{c_5(\ell + 2)}.
\]
As \(I(p_{th}, \cdot)\) is non-decreasing and 
\[
I(p_{th}, x_h) > \frac{h}{c_5(\ell + 2)},
\]
we can conclude that \(x_h > x_{h-1}\). \(\Box\)

**Step 3: Clustering and truncated random walks**

We now establish that vectors produced by the truncated random walk do not differ too much from those produced by the standard random walk.

**Lemma 2.13** (Low-impact Truncation). For all \(u \in V\) and \(t\), 
\[
p_t(u) \geq q_t(u) \geq r_t(u) \geq p_t(u) - t\epsilon d(u). \tag{20}
\]
For all \(t\) and \(x\), 
\[
I(p_t, x) \geq I(q_t, x) \geq I(r_t, x) \geq I(p_t, x) - \epsilon x t. \tag{21}
\]
Proof. The left-hand inequalities of (20) are trivial. To prove the right-hand inequality of (20), we consider \( p_t - [p_t]_e \), observe that by definition
\[
\left\| D^{-1} (p_t - [p_t]_e) \right\|_\infty \leq \epsilon,
\]
and then apply Proposition 2.2. Inequality (21) then follows from (3).

Lemma 2.14 (Lower bound on I). Let \( S \subseteq V \) be a set of vertices such that \( \mu(S) \leq (2/3) \mu(V) \) and \( \Phi(S) \leq f_1(\phi) \), and let \( v \) lie in \( S^g \). Define \( q_t \) by running 

1. If \( x_\ell(v) \geq 2m/c_6(\ell + 2) \), then
\[
I(q_{t+1}, (2/3)(2m)) \geq 1 - \frac{1}{c_2(\ell + 2)} - \frac{4c_6}{3c_3}
\]
(22)

2. Otherwise,
\[
I(q_{t+1}, x_{h_v}) \geq \frac{h_v + 1/2}{c_5(\ell + 2)}
\]
(23)

Proof. In the case \( x_\ell(v) \geq 2m/c_6(\ell + 2) \), we compute
\[
I(p_{t+1}, (2/3)(2m)) \geq I(p_{t+1}, \mu(S)) \geq \sum_{u \in S} p_{t+1}(u) \geq \lambda_S T p_{t+1} \geq 1 - t_{last} f_1(\phi), \quad \text{by the definition of } S^g
\]
by (3).

As \( 2^{b+1} > x_\ell \geq 2m/c_6(\ell + 2) \), we may use Lemma 2.13 to show
\[
I(q_{t+1}, (2/3)(2m)) \geq 1 - \frac{1}{c_2(\ell + 2)} - \epsilon(4m/3)t_{last} = 1 - \frac{1}{c_2(\ell + 2)} - \frac{4c_6}{3c_3},
\]
by (16).

If \( x_\ell(v) < 2m/c_6(\ell + 2) \), we compute
\[
I(q_{t+1}, x_{h_v}) \geq I(p_{t+1}, x_{h_v}) - \epsilon t_{last} x_{h_v} \quad \text{by Lemma 2.13}
\]
\[
= \frac{h_v + 1}{c_5(\ell + 2)} - \epsilon t_{last} x_{h_v}
\]
(24)
\[
\geq \frac{h_v + 1}{c_5(\ell + 2)} - \frac{x_{h_v}}{c_3(\ell + 2)2^b} \quad \text{as } x_{h_v} \leq 2x_{h_v-1}
\]
(25)
\[
\geq \frac{h_v + 1}{c_5(\ell + 2)} - \frac{2x_{h_v-1}}{c_3(\ell + 2)2^b} \quad \text{as } x_{h_v-1} < 2^{b+1}
\]
(26)
\[
\geq \frac{h_v + 1/2}{c_5(\ell + 2)}, \quad \text{by } (6).
\]
□
Lemma 2.15 (C.4). Let $S \subseteq V$ be a set of vertices such that $\mu(S) \leq (2/3)\mu(V)$ and $\Phi(S) \leq f_1(\phi)$, and let $v$ lie in $S^0$. If the Nibble is run with parameter $b$, then for all $t \in (t_{h_v-1}, t_{h_v}]$, condition (C.4) is satisfied.

Proof. We first consider the case in which $x_\ell < 2m/c_0(\ell + 2)$, which by definition implies $x_{h_v} \leq 2x_{h_v-1}$.

In this case, we have

$$I(q_t, x_{h_v-1}) \leq I(p_t, x_{h_v-1}) \leq I(p_{h_v-1}, x_{h_v-1}) = h_v/c_5(\ell + 2),$$

where the first inequality follows from Lemma 2.13 and the second follows from Lemma 2.9.

As $I_x(q_t, x)$ is non-increasing in $x$ and $x_{h_v-1} < x_{h_v} \leq 2x_{h_v-1}$, we have

$$I_x(q_t, x_{h_v-1}) \geq \frac{I(q_t, x_{h_v}) - I(q_t, x_{h_v-1})}{x_{h_v} - x_{h_v-1}} \geq \frac{I(q_{h_v}, x_{h_v}) - I(q_t, x_{h_v-1})}{x_{h_v} - x_{h_v-1}} \geq \frac{1}{2c_5(\ell + 2)x_{h_v-1}},$$

where the second inequality follows from Lemma 2.9 and the definition of $q_t$, and the last follows from (23).

If $x_{h_v-1} \geq 2$, then $b \geq 1$ and we have $2^b \leq x_{h_v-1} < 2^{b+1}$, and so

$$I_x(q_t, 2^b) \geq I_x(q_t, x_{h_v-1}) \geq \frac{1}{2c_5(\ell + 2)2^{b+1}},$$

and by (7) condition (C.4) is satisfied.

If $x_{h_v-1} < 2$, then $b = 0$, and so $I_x(q_t, 2^b) = I_x(q_t, x)$ for all $x < 1$, which implies

$$I_x(q_t, 2^b) \geq I_x(q_t, x_{h_v-1}) \geq \frac{1}{2c_5(\ell + 2)x_{h_v-1}} > \frac{1}{2c_5(\ell + 2)^b}$$

and condition (C.4) is satisfied.

If $x_\ell \geq 2m/c_0(\ell + 2)$, in which case $h_v = m + 1$ and $x_\ell < 2^{b+1}$, we apply Lemma 2.13 to show that for all $t \in (t_\ell, t_{\ell+1}]$,

$$I(q_t, 2m) \geq I(p_t, 2m) - 2m\eta_{last} = 1 - \frac{2m}{c_3(\ell + 2)2^b} \geq 1 - \frac{2m}{c_3(\ell + 2)x_\ell/2} \geq 1 - \frac{2c_0}{c_3}.$$

On the other hand,

$$I(q_t, x_\ell) \leq I(p_t, x_\ell) \leq I(p_{t_\ell}, x_\ell) < 1/c_5.$$

As $I_x(q_t, \cdot)$ is non-decreasing and $x_\ell \geq 2^b$, we have

$$I_x(q_t, 2^b) \geq I_x(q_t, x_\ell) \geq \frac{I(q_t, 2m) - I(q_t, x_\ell)}{2m - x_\ell} \geq \frac{1}{2m} \left( 1 - \frac{2c_0}{c_3} - \frac{1}{c_3} \right) \geq \frac{1}{c_0(\ell + 2)^{b+1}} \left( 1 - \frac{2c_0}{c_3} - \frac{1}{c_3} \right),$$

as $2^{b+1} > x_\ell \geq 2m/c_0(\ell + 2)$, and so by (8) condition (C.4) is satisfied.\qed

It remains to show that conditions (C.1–3) are met for some $t \in (t_{h_v-1}, t_{h_v}]$. We will do this by showing that if at least one of these conditions fail for every $j$ and every $t \in (t_{h_v-1}, t_{h_v}]$, then the curve $I(q_{h_v}, \cdot)$ will be too low, in violation of Lemma 2.14.
Lemma 2.16. If there exists a $\beta > 0$ and an $h \in [1, \ell + 1]$, such that for all $t \in (t_{h-1}, t_h]$ and for all $j$ either

1. $\Phi(S_j(q_t)) \geq \phi$,
2. $\lambda_j(q_t) > (5/6)2m$, or
3. $I(q_t, \lambda_j(q_t)) < \beta$,

then, for all $x$

$$I(q_{t_h}, x) < \beta + \frac{3x}{5m} + \sqrt{x} \left(1 - \frac{\phi^2}{2}\right)^{t_1}.$$  

Proof. We will prove by induction that the conditions of the lemma imply that for all $t \in [t_{h-1}, t_h]$ and all $x$,

$$I(q_t, x) < \beta + \frac{3x}{5m} + \sqrt{x} \left(1 - \frac{\phi^2}{2}\right)^{t-t_{h-1}}. \tag{24}$$

The base case is when $t = t_{h-1}$, in which case (24) is satisfied because

- For $1 \leq x \leq 2m - 1$, $I(q_t, x) \leq I(q_t, 2m) \leq 1 \leq \sqrt{x}$.
- For $0 \leq x \leq 1$, we have $I(q_t, x) \leq \sqrt{x}$ as both are 0 at $x = 0$, the right-hand term dominates at $x = 1$, the left-hand term is linear in this region, and the right-hand term is concave.
- For $2m - 1 \leq x \leq 2m$, we note that at $x = 2m$, $I(q_t, x) = 1 < 3x/5m$, and that we already know the right-hand term dominates at $x = 2m - 1$. The inequality then follows from the facts that left-hand term is linear in this region, and the right-hand term is concave.

Let

$$f(x) \overset{\text{def}}{=} \sqrt{x}.$$  

Lovász and Simonovits [LS90] observe that

$$\frac{1}{2} \left(f(x - 2\phi \sqrt{x}) + f(x + 2\phi \sqrt{x})\right) \leq f(x) \left(1 - \frac{\phi^2}{2}\right). \tag{25}$$

We now prove that (24) holds for $t$, assuming it holds for $t - 1$, by considering three cases. As the right-hand side is concave and the left-hand side is piecewise-linear between points of the form $\lambda_j(q_t)$, it suffices to prove the inequality at the points $\lambda_j(q_t)$. If $x = \lambda_j(q_t)$ and $I(q_t, x) \leq \beta$, then (24) holds trivially. Similarly, if $x = \lambda_j(q_t) > (5/6)2m$, then (24) holds trivially as well, as the left-hand side is at most 1, and the right hand side is at least 1. In the other cases, we have
\( \Phi(S_j(q_i)) \geq \phi \), in which case we may apply Lemma 2.10 to show that for \( x = \lambda_j(q_i) \)

\[
I(q_t, x) = I(Mr_{t-1}, x)
\]

\[
\leq \frac{1}{2} \left( I(r_{t-1}, x - 2\phi h) + I(r_{t-1}, x + 2\phi h) \right) \text{ by definition}
\]

\[
\leq \frac{1}{2} (I(q_{t-1}, x - 2\phi h) + I(q_{t-1}, x + 2\phi h))
\]

\[
< \frac{1}{2} \left[ \beta + \frac{3(x - 2\phi h)}{5m} + \sqrt{x - 2\phi h} \left( 1 - \frac{\phi^2}{2} \right)^{t-1} - t_{h-1} \right]
\]

\[
= \frac{3x}{5m} + \frac{1}{2} \sqrt{x - 2\phi h} + \sqrt{x + 2\phi h} \left( 1 - \frac{\phi^2}{2} \right)^{t-1} - t_{h-1}
\]

\[
\leq \beta + \frac{3x}{5m} + \sqrt{x} \left( 1 - \frac{\phi^2}{2} \right)^{t-h-1},
\]

by (26).

We now observe that \( t_1 \) has been chosen to ensure

\[
\sqrt{x} \left( 1 - \frac{\phi^2}{2} \right)^{t_1} < \frac{1}{c_1(\ell + 2)},
\]

Lemma 2.17 (C.1-3). Let \( S \) be a set of vertices such that \( \mu(S) \leq (2/3)(2m) \) and \( \Phi(S) \leq f_1(\phi) \), and let \( v \) lie in \( S_b^g \). If Nibble is run with parameter \( b \), then there exists a \( t \in (t_{h_v-1}, t_{h_v}] \) and a \( j \) for which conditions (C.1-3) are satisfied.

Proof. We first show that for \( t \in (t_{h_v-1}, t_{h_v}] \), (C.3) is implied by \( I(q_t, \lambda_j(q_i)) \geq h_v/c_5(\ell + 2) \). To see this, note that for \( b > 0 \)

\[
I(q_t, 2b) \leq I(q_t, x_{h_v-1}),
\]

\[
\leq I(p_t, x_{h_v-1}), \quad \text{as } x_{h_v-1} \geq 2b,
\]

\[
\leq I(p_{h_v-1}, x_{h_v-1}) \quad \text{by Lemma 2.13}
\]

\[
= \frac{h_v}{c_5(\ell + 2)}.
\]

So, \( I(q_t, \lambda_j(q_t)) \geq h_v/c_5(\ell + 2) \) implies \( \lambda_j(q_t) \geq 2b \), and we may prove the lemma by exhibiting a \( t \) and \( j \) for which (C.1), (C.2) and I(q_t, \lambda_j(q_i)) \geq h_v/c_5(\ell + 2) hold. On the other hand, if \( b = 0 \) then \( \lambda_j(q_t) \geq 1 = 2^b \) for all \( j \geq 1 \), so \( I(q_t, \lambda_j(q_i)) \geq h_v/c_5(\ell + 2) \) trivially implies \( j \geq 1 \) and therefore (C.3).

We will now finish the proof by contradiction: we show that if no such \( t \) and \( j \) exist, then the curve \( I(q_{h_v}, x) \) would be too low. If for all \( t \in (t_{h_v-1}, t_{h_v}] \) and all \( j \) one of (C.1), (C.2) or \( I(q_t, \lambda_j(q_i)) \geq h_v/c_5(\ell + 2) \) fails, then Lemma 2.16 tells us that for all \( x \)

\[
I(q_{h_v}, x) \leq \frac{h_v}{c_5(\ell + 2)} + \frac{3x}{5m} + \sqrt{x} \left( 1 - \frac{\phi^2}{2} \right)^{t_1} \leq \frac{h_v}{c_5(\ell + 2)} + \frac{3x}{5m} + \frac{1}{c_1(\ell + 2)},
\]
by inequality (26).

In the case \( x_\ell < 2m/c_6(\ell + 2) \), we obtain a contradiction by plugging in \( x = x_{hv} \) to find

\[
I(q_{hv}, x_{hv}) < \frac{1}{\ell + 2} \left( \frac{hv}{c_5} + \frac{6}{5c_6} + \frac{1}{c_1} \right)
\]

which by (9) contradicts (23).

In the case in that \( x_\ell \geq 2m/c_6(\ell + 2) \), and so \( hv = \ell + 1 \), we substitute \( x = (2/3)2m \) to obtain

\[
I(q_{\ell+1}, (2/3)2m) < \frac{4}{5} + \frac{1}{c_5} + \frac{1}{c_1(\ell + 2)},
\]

which by (10) contradicts (22).

\[\square\]

### 2.4 Proof of Theorem 2.1

Fact (N.1) follows from conditions (C.1) and (C.2) in the algorithm. Given a set \( S \) satisfying \( \mu(S) \leq (2/3)\mu(V) \), the lower bound on the volume of the set \( S^\ell \) is established in Lemma 2.7. If \( \Phi(S) \leq f_1(\phi) \) and \( v \in S^\ell \), then Lemmas 2.17 and 2.15 show that the algorithm will output a non-empty set. Finally, lemma 2.8 tells us that if \( \Phi(S) \leq f_1(\phi) \), \( v \in S^\ell \) and the algorithm outputs a non-empty set \( C \), then it satisfies \( \mu(C \cap S) \geq 2b - 1 \).

It remains to bound the running time of \texttt{Nibble}. The algorithm will run for \( t_{\text{last}} \) iterations. We will now show that with the correct implementation, each iteration takes time \( O((\log n)/\epsilon) \). Instead of performing a dense vector multiplication in step (3.a), the algorithm should keep track of the set of vertices \( u \) at which \( r_\ell(u) > 0 \). Call this set \( V_\ell \). The set \( V_\ell \) can be computed in time \( O(|V_\ell|) \) in step (3.b). Given knowledge of \( V_{\ell-1} \), the multiplication in step (3.a) can be performed in time proportional to

\[
\mu(V_{\ell-1}) = \sum_{u \in V_{\ell-1}} d(u) \leq \sum_{u \in V_{\ell-1}} r_\ell(u)/\epsilon \leq 1/\epsilon.
\]

Finally, the computation in step (3.c) might require sorting the vectors in \( V_\ell \) according to \( r_\ell \), which could take time at most \( O(|V_\ell| \log n) \). Thus, the run-time of \texttt{Nibble} is bounded by

\[
O \left( \frac{t_{\text{last}} \log n}{\epsilon} \right) = O \left( \frac{t_{\text{last}}^2 2^b \log^2 m}{\phi^4} \right).
\]

### 3 Nearly Linear-Time Graph Partitioning

In this section, we apply \texttt{Nibble} to design a partitioning algorithm \texttt{Partition}. This new algorithm runs in nearly linear-time. It computes an approximate sparsest cut with approximately optimal balance. In particular, we prove that there exists a constant \( \alpha > 0 \) such that for any graph \( G = (V, E) \) that has a cut \( S \) of sparsity \( \alpha \cdot \theta^2/\log^3 n \) and balance \( b \leq 1/2 \), with high probability, \texttt{Partition} finds a cut \( D \) with \( \Phi_V(D) \leq \theta \) and \( \text{bal}(D) \geq b/2 \). Actually, \texttt{Partition} satisfies an even stronger guarantee: with high probability either the cut it outputs is well balanced,

\[
\frac{1}{4} \mu(V) \leq \mu(D) \leq \frac{3}{4} \mu(V),
\]

\[\square\]
or touches most of the edges touching $S$,
\[ \mu(D \cap S) \geq \frac{1}{2}\mu(S). \]

The expected running time of \textbf{Partition} is $O(m \log^7 n/\phi^4)$. Thus, it can be used to quickly find crude cuts.

\textbf{Partition} calls \textbf{Nibble} via a routine called \textbf{Random Nibble} that calls \textbf{Nibble} with carefully chosen random parameters. \textbf{Random Nibble} has a very small expected running time, and is expected to remove a similarly small fraction of any set with small conductance.

### 3.1 Procedure \textbf{Random Nibble}

\begin{verbatim}
C = RandomNibble(G, \phi)

(1) Choose a vertex $v$ according to $\psi_V$.

(2) Choose a $b$ in $1, \ldots, \lceil \log m \rceil$ according to
\[ \Pr[b = i] = 2^{-i}/(1 - 2^{-\lceil \log m \rceil}). \]

(3) $C = \text{Nibble}(G, v, \phi, b)$.
\end{verbatim}

**Lemma 3.1 (Random Nibble).** Let $m$ be the number of edges in $G$. The expected running time of \textbf{Random Nibble} is $O \left( \log^7 m/\phi^4 \right)$. If the set $C$ output by \textbf{Random Nibble} is non-empty, it satisfies

(R.1) $\Phi_V(C) \leq \phi$, and

(R.2) $\mu(C) \leq (5/6)\mu(V)$.

Moreover, for every set $S$ satisfying
\[ \mu(S) \leq (2/3)\mu(V) \quad \text{and} \quad \Phi_V(S) \leq f_1(\phi), \]

(R.3) $E[\mu(C \cap S)] \geq \mu(S)/4\mu(V)$.

**Proof.** The expected running time of \textbf{Random Nibble} may be upper bounded by
\[ O \left( \sum_{i=1}^{\lceil \log m \rceil} \left( 2^{-i}/(1 - 2^{[\log m]}(2^i \log^6(m)/\phi^4) \right) = O \left( \log^7(m)/\phi^4 \right). \]

Parts (R.1) and (R.2) follow directly from part (N.1) of Theorem 2.1. To prove part (R.3), define $\alpha_b$ by
\[ \alpha_b = \frac{\mu(S^g_b)}{\mu(S^g)}. \]
So, \( \sum_b \alpha_b = 1 \). For each \( i \), the chance that \( v \) lands in \( S_i^g \) is \( \alpha_i \mu(S^g) / \mu(V) \). Moreover, the chance that \( b = i \) is at least \( 2^{-i} \). If \( v \) lands in \( S_i^g \), then by part (N.3) of Theorem 2.1, \( C \) satisfies 
\[
\mu(C \cap S) \geq 2^{i-1}.
\]
So,
\[
E[\mu(C \cap S)] \geq \sum_i 2^{-i} \alpha_i \mu(S^g) / \mu(V) 2^{i-1} = \sum_i (1/2) \alpha_i (\mu(S^g) / \mu(V)) = \mu(S^g) / 2 \mu(V) \geq \mu(S) / 4 \mu(V). \quad \text{(by part (N.2.a) of Theorem 2.1)}
\]

\[\square\]

3.2 Partition

We now define Partition and analyze its performance. First, define 
\[
f_2(\theta) \overset{\text{def}}{=} f_1(\theta/7)/2,
\]
and note 
\[
f_2(\theta) \geq \Omega\left(\frac{\theta^2}{\log^7 m}\right)
\]

\[
D = \text{Partition}(G, \theta, p), \text{ where } G \text{ is a graph, } \theta, p \in (0, 1).
\]

(0) Set \( W_0 = V, j = 0 \) and \( \phi = \theta/7 \).

(1) While \( j < 12m \lceil \lg(1/p) \rceil \) and \( \mu(W_j) \geq (3/4) \mu(V) \),

(a) Set \( j = j + 1 \).

(b) Set \( D_j = \text{RandomNibble}(G[W_{j-1}], \phi) \)

(c) Set \( W_j = W_{j-1} - D_j \).

(2) Set \( D = D_1 \cup \cdots \cup D_j \).

**Theorem 3.2 (Partition).** On input a graph with \( m \) edges, the expected running time of Partition is \( O\left( m \lg(1/p) \log^7 m / \theta^4 \right) \). Let \( D \) be the output of Partition\((G, \theta, p)\), where \( G \) is a graph and \( \theta, p \in (0, 1) \). Then

(P.1) \( \mu(D) \leq (7/8) \mu(V) \),

(P.2) If \( D \neq \emptyset \) then \( \Phi_V(D) \leq \theta \), and
(P.3) If $S$ is any set satisfying

$$
\mu(S) \leq \mu(V)/2 \quad \text{and} \quad \Phi_V(S) \leq f_2(\theta),
$$

then with probability at least $1 - p$, $\mu(D) \geq \mu(S)/2$.

In particular, with probability at least $1 - p$ either

(P.3.a) $\mu(D) \geq (1/4)\mu(V)$, or

(P.3.b) $\mu(S \cap D) \geq \mu(S)/2$.

Property (P.3) is a little unusual and deserves some explanation. It says that for every set $S$ of low conductance, with high probability either $D$ is a large fraction of $S$, or it is a large fraction of the entire graph. While we would like to pick just one of these properties and guarantee that it holds with high probability, this would be unreasonable: on one hand, there might be no big set $D$ of small conductance; and, on the other hand, even if $S$ is small the algorithm might cut out a large set $D$ that completely avoids $S$.

Proof of Theorem 3.2. The bound on the expected running time of Partition is immediate from the bound on the running time of RandomNibble.

Let $j_{\text{out}}$ be the iteration at which Partition stops, so that $D = D_1 \cup \cdots \cup D_{j_{\text{out}}}$. To prove (P.1), note that $\mu(W_{j_{\text{out}}-1}) \geq (3/4)\mu(V)$ and so $\mu(D_1 \cup \cdots \cup D_{j_{\text{out}}-1}) \leq (1/4)\mu(V)$. By part (R.2) of Lemma 3.1, $\mu(D_{j_{\text{out}}}) \leq (5/6)\mu(W_{j_{\text{out}}-1})$. So,

$$
\mu(D_1 \cup \cdots \cup D_{j_{\text{out}}}) \leq \mu(V) - \mu(W_{j_{\text{out}}-1}) + \frac{5}{6}\mu(W_{j_{\text{out}}-1}) = \mu(V) - \frac{1}{6}\mu(W_{j_{\text{out}}-1}) \leq \frac{7}{8}\mu(V).
$$

To establish (P.2), we first compute

$$
|E(D, V - D)| = \sum_{i=1}^{j_{\text{out}}} |E(D_i, V - D)| \leq \sum_{i=1}^{j_{\text{out}}} |E(D_i, W_{i-1} - D_i)| \leq \sum_{i=1}^{j_{\text{out}}} \phi\mu(D_i) \quad \text{(by (R.1))}
$$

$$
= \phi\mu(D).
$$

So, if $\mu(D) \leq \mu(V)/2$, then $\Phi_V(D) \leq \phi$. On the other hand, we established above that $\mu(D) \leq (7/8)\mu(V)$, from which it follows that

$$
\mu(V - D) \geq (1/8)\mu(V) \geq (8/7)(1/8)\mu(D) = (1/7)\mu(D).
$$

So,

$$
\Phi_V(D) = \frac{|E(D, V - D)|}{\min(\mu(D), \mu(V - D))} \leq 7\frac{|E(D, V - D)|}{\mu(D)} \leq 7\phi = \theta.
$$
Let \( j_{max} = 12m \lceil \lg(1/p) \rceil \). To prove part (P.3), let \( S \) satisfy (28) and consider what happens if we ignore the second condition in the while loop and run Partition for all potential \( j_{max} \) iterations, obtaining cuts \( D_1, \ldots, D_{12m\lceil \lg(1/p) \rceil} \). Let

\[
D^{\leq j} = \bigcup_{i \leq j} D_i.
\]

We will prove that if neither \( \mu(D^{\leq j} \cap U_0) \geq \frac{3}{4} \mu(V) \) nor \( \mu(S \cap D^{\leq j}) \geq \frac{1}{2} \mu(S) \)

hold at iteration \( k \), then with probability at least \( 1/2 \), one of these conditions will be satisfied by iteration \( k + 12m \). Thus, after all \( j_{max} \) iterations, one of conditions (P.3.a) or (P.3.b) will be satisfied with probability at least \( 1 - p \). If the algorithm runs for fewer iterations, then condition (P.3.a) is satisfied.

To simplify notation, let \( C_i = D_{k+i} \) and \( U_i = W_{k+i} \), for \( 0 \leq i \leq 12m \). Assume that

\[
\mu(U_0) \geq \frac{3}{4} \mu(V) \quad \text{and} \quad \mu(S \cap U_0) < \frac{1}{2} \mu(S).
\]

For \( 1 \leq i \leq 12m \), define the random variable

\[
X_i = \frac{\mu(C_i \cap S)}{\mu(U_0 \cap S)}.
\]

As each set \( C_i \) is a subset of \( U_0 \), and the \( C_i \) are mutually disjoint, we will always have

\[
\sum_{i=1}^{12m} X_i \leq 1.
\]

Define \( \beta \) to satisfy

\[
(1 - \beta) \mu(S \cap U_0) = \frac{1}{2} \mu(S),
\]

and note that this ensures \( 0 < \beta \leq 1/2 \). Moreover, if \( \sum X_i \geq \beta \), then \( \mu(S \cap D^{\leq k+12m}) \geq \mu(S) / 2 \) will hold.

Let \( E_j \) be the event

\[
\mu(U_j) < \frac{3}{4} \mu(V).
\]

We need to show that, with probability at least \( 1/2 \), either an event \( E_j \) holds, or \( \sum X_i \geq \beta \). To this end, we now show that if neither \( E_j \) nor \( \sum_{i \leq j} X_i \geq \beta \) holds, then \( E \{X_{j+1}\} \geq 1/8m \). If \( \sum_{i \leq j} X_i < \beta \), then

\[
\mu(S \cap U_j) = \mu(S \cap U_0) - \sum_{i \leq j} \mu(S \cap C_i) = \mu(S \cap U_0) \left( 1 - \sum_{i \leq j} X_i \right) > \mu(S \cap U_0) (1 - \beta) = \frac{1}{2} \mu(S).
\]

If \( E_j \) does not hold, then

\[
\mu(U_j - S \cap U_j) = \mu(U_j) - \mu(S \cap U_j) \geq \frac{3}{4} \mu(V) - \mu(S) \geq \frac{1}{4} \mu(V) \geq \frac{1}{2} \mu(S).
\]
So,
\[
\Phi_{U_j}^G (S \cap U_j) = \frac{|E(S \cap U_j, U_j - S \cap U_j)|}{\min (\mu(S \cap U_j), \mu(U_j - S \cap U_j))} \leq \frac{|E(S, V - S)|}{(1/2)\mu(S)} \leq 2\Phi_G (S) \leq 2f_2(\theta) = f_1(\phi).
\]

We also have \(\mu(S \cap U_j) \leq \mu(S) \leq (2/3)\mu(U_j)\), so the conditions of part (R.3) of Lemma 3.1 are satisfied and
\[
\mathbb{E} [X_{j+1}] \geq 1/4\mu(U_j) \geq 1/8m.
\]

Now, set
\[
Y_j = \begin{cases} 
1/8m & \text{if } \sum_{i < j} X_i \geq \beta, \text{ or if } E_{j-1} \\
X_j & \text{otherwise.}
\end{cases}
\]

So, for all \(j\) we have \(\mathbb{E} [Y_j] \geq 1/8m\), and so \(\mathbb{E} \left[ \sum_{j \leq 12m} Y_j \right] \geq 3/2\). On the other hand,
\[
\sum_{i \leq 12m} Y_j \leq \sum_{i} X_i + \frac{12m}{8m} \leq 5/2.
\]

So, with probability at least 1/2,
\[
\sum_{j \leq 12m} Y_j \geq 1/2 \geq \beta.
\]

This implies that with probability at least 1/2 either \(\sum_{i} X_i \geq \beta\) or some event \(E_j\) holds, which is what we needed to show.

\[\square\]

References

[ABC+07] Reid Andersen, Christian Borgs, Jennifer T. Chayes, John E. Hopcraft, Vahab S. Mirrokni, and Shang-Hua Teng. Local computation of pagerank contributions. In Anthony Bonato and Fan R. K. Chung, editors, WAW, volume 4863 of Lecture Notes in Computer Science, pages 150–165. Springer, 2007.

[ACL06] Reid Andersen, Fan Chung, and Kevin Lang. Local graph partitioning using pagerank vectors. Proceedings: 47th Annual Symposium on Foundations of Computer Science, pages 475–486, 2006.

[ACL07] Reid Andersen, Fan R. K. Chung, and Kevin J. Lang. Local partitioning for directed graphs using pagerank. In Anthony Bonato and Fan R. K. Chung, editors, WAW, volume 4863 of Lecture Notes in Computer Science, pages 166–178. Springer, 2007.

[AHK04] Sanjeev Arora, Elad Hazan, and Satyen Kale. \(O(\sqrt{\log n})\) approximation to Sparsest Cut in \(\tilde{O}(n^2)\) time. In FOCS: IEEE Symposium on Foundations of Computer Science (FOCS), pages 238–247, 2004.

[AK07] Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. In STOC ’07: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing, pages 227–236, New York, NY, USA, 2007. ACM.
[And08] Reid Andersen. A local algorithm for finding dense subgraphs. In Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008, pages 1003–1009. SIAM, 2008.

[ARV04] Sanjeev Arora, Satish Rao, and Umesh 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.

[Cor05] Intel Corporation. Morre’s law. ftp://download.intel.com/museum/MooresLaw/PrintedMaterials/MooresLaw2005.

[GS05] A. Gulli and A. Signorini. The indexable web is more than 11.5 billion pages. In WWW ’05: Special interest tracks and posters of the 14th international conference on World Wide Web, pages 902–903. ACM, 2005.

[KRV06] Rohit Khandekar, Satish Rao, and Umesh 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.

[KVV04] Ravi Kannan, Santosh Vempala, and Adrian Vetta. On clusterings: Good, bad and spectral. J. ACM, 51(3):497–515, 2004.

[LH08] Jure Leskovec and Eric Horvitz. Planetary-scale views on a large instant-messaging network. In WWW ’08: Proceeding of the 17th international conference on World Wide Web, pages 915–924. ACM, 2008.

[LR99] Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. Journal of the ACM, 46(6):787–832, November 1999.

[LS90] L. Lovasz and M. Simonovits. The mixing rate of Markov chains, an isoperimetric inequality, and computing the volume. In IEEE, editor, Proceedings: 31st Annual Symposium on Foundations of Computer Science: October 22–24, 1990, St. Louis, Missouri, volume 1, pages 346–354, 1109 Spring Street, Suite 300, Silver Spring, MD 20910, USA, 1990. IEEE Computer Society Press.

[LS93] Lovasz and Simonovits. Random walks in a convex body and an improved volume algorithm. RSA: Random Structures & Algorithms, 4:359–412, 1993.

[OSVV08] Lorenzo Orecchia, Leonard J. Schulman, Umesh V. Vazirani, and Nisheeth K. Vishnoi. On partitioning graphs via single commodity flows. In STOC ’08: Proceedings of the 40th annual ACM symposium on Theory of computing, pages 461–470, New York, NY, USA, 2008. ACM.

[PBMW98] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital
Library Technologies Project, Stanford University, Stanford, CA, USA, November 1998.

[SLM+02] Ashish Sharma, Xinlian Liu, Paul Miller, Aiichiro Nakano, Rajiv K. Kalia, Priya Vashishta, Wei Zhao, Timothy J. Campbell, and Andy Haas. Immersive and interactive exploration of billion-atom systems. In *VR ’02: Proceedings of the IEEE Virtual Reality Conference 2002*, page 217. IEEE Computer Society, 2002.

[SS06] Jirí Síma and Satu Elisa Schaeffer. On the NP-completeness of some graph cluster measures. In Jirí Wiedermann, Gerard Tel, Jaroslav Pokorný, Mária Bieliková, and Julius Stuller, editors, *SOFSEM*, volume 3831 of *Lecture Notes in Computer Science*, pages 530–537. Springer, 2006.

[ST03] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. available at http://arxiv.org/abs/cs.DS/0310051. An extended abstract appeared in *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, 81–90, 2003.

[ST08a] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *CoRR*, abs/cs/0607105, 2008. Available at http://www.arxiv.org/abs/cs.NA/0607105.

[ST08b] Daniel A. Spielman and Shang-Hua Teng. Spectral sparsification of graphs. *CoRR*, abs/0808.4134, 2008. Available at http://arxiv.org/abs/0808.4134.