Fifty-fourth Annual Allerton Conference
Allerton House, UIUC, Illinois, USA
September 27 - 30, 2016

Edge Conductance Estimation Using MCMC

Ashish Bora*, Vivek S. Borkar*, Dinesh Garg†, Rajesh Sundaresan‡
*Department of Electrical Engineering
Indian Institute of Technology Bombay
Mumbai, India
†IBM India Research Lab
Bengaluru, India
‡Department of Electrical Communication Engineering
Robert Bosch Centre for Cyber Physical Systems
Indian Institute of Science
Bangalore, India

***Abstract*—We propose an iterative and distributed Markov Chain Monte Carlo scheme for estimation of effective edge conductances in a graph. A sample complexity analysis is provided. The theoretical guarantees on the performance of the proposed algorithm are weak compared to those of existing algorithms. But numerical experiments suggest that the algorithm might still be effective while offering the advantages of low per iterate computation and memory requirements.**

I. Introduction

Given a graph $G = (V, E)$, with node set V and edge set E, the effective resistance between two nodes i and j is the potential difference that results when a unit current is injected at i and extracted at j. Effective conductance is the inverse of the effective resistance.

Effective resistance has been used as a robust measure of *distance in social network graphs* ([1], [2]) because it is less sensitive to edge or node insertions and deletions than the usual shortest path distance on such graphs. The sum of effective resistances across all pairs has been used as a measure of network robustness [3]. This sum also equals a *network criticality parameter*, defined as a *betweenness*[1] of some node in the graph, normalized by that node's degree [4]. Effective resistances are also used in *graph sparsification* [5]. The procedure for sparsification is to sample, with replacement, an edge with probability in proportion to its effective resistance and include it in the subgraph with an appropriate weight (to ensure unbiasedness of the edge weights in the sparsified subgraph). If an edge is sampled multiple times, its weights are summed.

An elegant method to estimate effective resistances was proposed by [5]. It is well-known that the nodes of the graph can be embedded in Euclidean s pace of dimension $|E|$ so that the resulting pair-wise distances encode the effective resistances (see p. 1921 of [5]). The embedding depends on the edge-node adjacency matrix and the Laplacian of the graph. A low dimensional random projection of these vectors, one that approximately preserves pair-wise distances to within a factor $(1 \pm \epsilon)$ with high probability, is then identified. This was done in the context of cut-preserving and spectrum-preserving graph sparsification, and high probability guarantees on the approximate pair-wise distances suffice for high probability guarantees on the approximations of cuts or the spectrum. The main drawback with this approach is that it is centralized – the low dimensional projection is computed based on information about the entire graph.

Our goal in this paper is to propose effective conductance and effective resistance estimation algorithms based on random walks on the graph. A (simple) random walk on the graph picks, from the current position, one of the neighbors with equal probability. Our idea is very simple. The probability that this random walk starting at node i visits node j before returning to node i equals the effective conductance of (i, j) normalized by the degree of node i. We will estimate the conductance via the number of node i to node i paths that contain node j in a long walk. Such algorithms are local – the next step of the walk depends only on the local neighborhood at the current node. They take fewer computations per step, use lesser memory, and are often amenable to parallel implementation. Incremental as these algorithms are, their performances improve with the number of iterations, and one does not have to select or decide the accuracy parameter in advance. They are also amenable to adaptation and find application in situations where either the graph is evolving or information on the graph becomes available only in a streaming fashion.

We will focus on estimation of effective conductances and effective resistances of only the edges of the graph. But the extension to other pairs of nodes is straightforward, and we will indicate why in the final section. We will further assume that the given graph is finite, undirected, unweighted, and the associated Markov chain is aperiodic. Extensions to weighted graphs are straightforward and will not be discussed here.

The current theoretical guarantee on the number of computations for probably approximately correct estimation of effective resistances, those that we report in this paper, are admittedly inferior to the results of [5]. But simulation results

[1]This is called random walk betweenness of a node j: the sum over all pairs of nodes i and k of the expected number of times j is visited in a simple random walk starting at i and ending at k. The betweenness of a node j normalized by that node's degree turns out to be independent of the chosen node j and can therefore be viewed as a graph property.

978-1-5090-4550-1/16/$31.00 ©2016 IEEE

provide some evidence that the algorithm might yet be effective while offering the advantages of parallelizability and low per iterate computation and memory requirements.

A. Basic definitions and notations

Let $G = (V, E)$ be an undirected, unweighted, connected, finite graph. Let $m = |E|$ and $n = |V|$. Define ∂_i as the neighborhood of node i, i.e., $\partial_i = \{j | (i, j) \in E\}$. Let $d_i = |\partial_i|$ be the degree of node $i \in V$. Let $d_{max} = \max_{i \in V} d_i$ and $d_{min} = \min_{i \in V} d_i$ be the maximum and minimum degrees, respectively. For a pair of nodes i and j, let $d_{ij} := \min\{d_i, d_j\}$ and $D_{ij} := \max\{d_i, d_j\}$.

Consider the discrete-time Markov chain associated with the random walk on the graph: from the current position, say node i, the walk moves to one of the neighbors of node i with equal probability. Denote this Markov chain by $(X_t, t \geq 1)$, where X_t is the state of the Markov chain (location of the walk) at time t. X_1 is the initial state. Let $\mathcal{L}(X_t)$ denote the distribution of X_t. We shall assume that the Markov chain is irreducible and aperiodic, and so it has a unique stationary distribution which we shall denote by π. It is easy to see that the Markov chain is reversible, and from the detailed balance equation, we have $\pi_i = d_i/(2m),\ i \in V$.

The time taken by the Markov chain to get "close" to the stationary distribution is the mixing time of the Markov chain. To define this precisely, let us consider the total variational distance between two distributions P and Q on the alphabet V:

$$d_{TV}(P, Q) := \sup_{A \subseteq V} |P(A) - Q(A)|.$$

The mixing time t_{mix} of this Markov chain is defined as

$$t_{mix} := \min \left\{ t \geq 1 \mid \max_{i \in V} d_{TV}(\mathcal{L}(X_t | X_1 = i), \pi) \leq \frac{1}{4} \right\}.$$

Let us define $d_{avg} = \sum_{i \in V} d_i \pi_i = \sum_{i \in V} d_i^2/2m$ to be the average degree with respect to the stationary measure. For every $(i, j) \in E$, let G_{ij} be the effective conductance between the nodes $i \in V$ and $j \in V$ and define effective resistance to be $R_{ij} := 1/G_{ij}$. Let us define p_{ij} to be the probability that the random walk starting at node i visits node j before returning to node i. The key fact that underlies our conductance estimation algorithm is that

$$p_{ij} = G_{ij}/d_i. \tag{1}$$

See Prop. 2.3 and Th. 4.1 of [6] for a proof of (1).

B. Related work

Reference [5] describes a procedure that outputs approximate effective edge resistances. The main idea of their work is based on the observation that the effective resistance can be computed as follows. Each node is associated with a suitable vector in $\mathbb{R}^m$. This vector depends on the edge-node adjacency matrix, the weights of the edges, and the associated Laplacian of the graph. The effective resistance between two nodes is then the squared distance between the two associated vectors in $\mathbb{R}^m$. They then use the Johnson-Lindenstrauss lemma [7] to project these vectors on a low dimensional space while approximately preserving distances. The projected vectors are approximated efficiently using a linear solver.

For any $\epsilon > 0$, their algorithm outputs effective resistances to within a factor $(1 \pm \epsilon)$ with probability atleast $1 - 1/n$. Their algorithm requires $\widetilde{O}(m/\epsilon^2)$ time; the $\widetilde{O}(\cdot)$ notation ignores poly-logarithmic factors.

II. The conductance estimation algorithm and its performance

For any two nodes $i, j \in V$, as indicated in (1), the probability p_{ij} that a simple random walk starting at node i visits node j before returning to node i is given by $p_{ij} = G_{ij}/d_i$. This suggests a natural Monte Carlo strategy to estimate G_{ij} by counting the number of visits to node j before returning to node i.

Algorithm : VisitBeforeReturn

---

1) Input T, $G = (V, E)$.
2) For each $i \in V$, initialize $N_i = 0$.
3) For each $(i, j) \in E$, initialize $\widehat{p}_{ij} = \widetilde{p}_{ij} = 0$.
4) Sample initial node X_1 from the stationary distribution π.
5) For $t = 1$ to T,
   Let $i = X_t$
   a) For each j in ∂_i:
      (i) $\widehat{p}_{ij} \leftarrow (\widehat{p}_{ij} N_i + \widetilde{p}_{ij})/(N_i + 1)$
      (ii) $\widetilde{p}_{ij} \leftarrow 0$
      (iii) $\widetilde{p}_{ji} \leftarrow 1$
   b) $N_i \leftarrow N_i + 1$
   c) Jump to a neighbor of the current node as identified by the walk.
6) For every $(i, j) \in E$, output

$$\widehat{G}_{ij} = \max\left(1, \frac{d_i}{2}\widehat{p}_{ij} + \frac{d_j}{2}\widehat{p}_{ji}\right).$$

Interpretation of the variables and the logic behind the above algorithm are as follows.

1) $\widetilde{p}_{ij}$ denotes the success or failure of visiting node j in an instance of a return path from node i to node i of the random walk.
2) N_i is the number of times node i was visited.
3) $\widehat{p}_{ij}$, which is just the average of $\widetilde{p}_{ij}$, is a running estimate of p_{ij}.
4) In the long run, every visit to node i marks the end of a return path. Thus, on visiting node i, we can update $\widehat{p}_{ij}$ using $\widetilde{p}_{ij}$.
5) In the long run, we would have visited all nodes at least once (with probability 1 because the chain is finite and irreducible). Fix an arbitrary node j (a neighbor of node i). A visit to node i at time t is a part of a cycle that originated at node j prior to time t and a subsequent return to node j after time t (with probability 1, because

of positive recurrence). Thus a visit to node i can be used to update $\widetilde{p}_{ji}$.

6) Since, $G_{ij} = d_i p_{ij} = d_j p_{ji} = (d_i/2)p_{ij} + (d_j/2)p_{ji}$, and since $G_{ij} \geq 1$, we may estimate $\widehat{G}_{ij}$ as in step 6 of the algorithm.

The theoretical guarantee we have been able to show for the above algorithm is the following.

Theorem 1 (Performance of **VisitBeforeReturn***):* Fix an edge $(i,j) \in E$. For any $0 < \epsilon < d_{ij}^2/(4m)$ and $0 < \delta < 1/2$,

$$T = \widetilde{O}\left(D_{ij} \cdot \max\{m, D_{ij} t_{mix}\} \cdot \frac{1}{\epsilon^2} \log \frac{1}{\delta}\right)$$

steps suffice to ensure that the output $\widehat{G}_{ij}$ of the algorithm **VisitBeforeReturn** satisfies

$$\mathbb{P}(|\widehat{G}_{ij} - G_{ij}| \geq \epsilon) \leq \delta.$$

If the algorithm is run for T steps, it requires $O(d_{avg}T)$ computation steps on the average (worst case $O(d_{max}T)$ computations), and uses $O(m \log T)$ space. □

For definitions, see Section I-A. See Appendix A for a proof.

We now highlight the positive aspects and the limitations of our algorithm.

One can turn our algorithm for estimating effective conductances into one for estimating effective resistances with relative error guarantees. It is these, estimates of effective resistances with relative error guarantees, that are used for graph sparsification in [5]. Let $\widehat{R}_{ij} = 1/\widehat{G}_{ij}$. Then we have the following for the relative error on the effective resistances:

$$\begin{aligned}
\mathbb{P}\left(\frac{|\widehat{R}_{ij} - R_{ij}|}{R_{ij}} \geq \epsilon\right) &= \mathbb{P}\left(\left|\frac{\widehat{R}_{ij}}{R_{ij}} - 1\right| \geq \epsilon\right) \\
&= \mathbb{P}\left(\left|\frac{G_{ij}}{\widehat{G}_{ij}} - 1\right| \geq \epsilon\right) \\
&= \mathbb{P}(|G_{ij} - \widehat{G}_{ij}| \geq \epsilon \widehat{G}_{ij}) \\
&\leq \mathbb{P}(|\widehat{G}_{ij} - G_{ij}| \geq \epsilon),
\end{aligned}$$

where the last inequality follows because $\widehat{G}_{ij} \geq 1$. This implies that the T steps, as specified in Theorem 1, are sufficient to ensure that the *relative error* on the effective resistance exceeds ϵ with probability at most δ.

By applying the union bound over edges, we also get that our algorithm takes

$$T = \widetilde{O}\left(d_{max} \cdot \max\{m, d_{max} t_{mix}\} \cdot \frac{1}{\epsilon^2} \log \frac{1}{\delta}\right)$$

steps to ensure that

$$\mathbb{P}\left(\max_{(i,j)\in E} |\widehat{G}_{ij} - G_{ij}| \geq \epsilon\right) \leq \delta.$$

Let us now discuss the limitations. Assume $\delta = 1/4$. One of the hypotheses of the theorem is the very restrictive assumption of $\epsilon < d_{ij}^2/(4m)$. This assumption arises because our algorithm is based on concentration of the number of returns to a node (say i), and this is available only if π_i is sufficiently large. A consequence of this condition on ϵ is that the running time is at least $O(m^3)$. One might as well invert the Laplacian matrix and compute the resistances exactly. In contrast to the theoretical guarantee on our algorithm, the algorithm of [5] requires only $\widetilde{O}(m/\epsilon^2)$ steps.

Suppose that the graph is dense so that $d_{min} = O(\sqrt{m})$. This is the situation when the problem of graph sparsification is most relevant. The assumption on ϵ now allows ϵ to be a small constant. But the number of steps for a theoretical guarantee on our algorithm is still too large: $\widetilde{O}(m^{3/2}/\epsilon^2)$.

If d_{max} and t_{mix} are $\widetilde{O}(1)$, and if one can remove the restriction $\epsilon < d_{ij}^2/(2m)$, then we would have an $\widetilde{O}(m/\epsilon^2)$ MCMC algorithm to ensure that

$$\mathbb{P}\left(\max_{(i,j)\in E} |\widehat{G}_{ij} - G_{ij}| \geq \epsilon\right) \leq \delta.$$

It would then be comparable with the algorithm of [5] with the advantage of decentralization, etc., that come with random walk algorithms. Examples of graphs that have d_{max} and t_{mix} to be $\widetilde{O}(1)$ are expanders such as Erdős-Rényi graphs $G(n,p)$ with $p = \log n/n$ or $p = c/n$, with G taken to be the giant component.

The question of whether the stringent restriction on ϵ can be removed is left as an open question. In the next section, we present some promising simulation results that highlight the performance of the MCMC approach.

Our algorithm suggests the following approach for graph sparsification. Given an $\epsilon > 0$, for an edge (i,j), if ϵ satisfies the condition with respect to i or j, then delete or keep the edge accordingly to whether the estimate of the conductance is high or low, respectively. Our algorithm provides a reliable estimate of the conductance of such edges. If ϵ is small with respect to π_i and π_j, then simply keep edge (i,j). Theorem 1 does not provide a guarantee on the performance of our algorithm, but since $G_{ij} = 2m\pi_i p_{ij}$ is likely to be low anyway (because π_i is), keep the edge. The performance of this MCMC-based sparsification needs further study.

III. Simulation Experiments

A. Convergence

To test the performance and the convergence of the algorithm, simulation experiments were performed as follows.

Given a graph G, it is known [5] that the effective conductance between any two nodes $i, j \in V$ is given by

$$G_{ij} = \left((\chi_i - \chi_j)^t L^+ (\chi_i - \chi_j)\right)^{-1}, \tag{2}$$

where χ_i is the i-th standard unit vector, and L^+ is the pseudoinverse of the Laplacian of the graph. The Laplacian L is given by $L = D - A$, where A is the adjacency matrix, i.e., $a_{ij} = 1$ if $(i,j) \in E$ and 0 otherwise and D is the diagonal matrix of degrees $D = \text{diag}(d_1, d_2, \ldots, d_n)$. Exact computation of (2) is expensive, but we do this on sample graphs for evaluation of our algorithm.

Let us denote by $\widehat{G}(t)$, the conductance estimate vector at the t^{th} step of the random walk. The estimates were compared

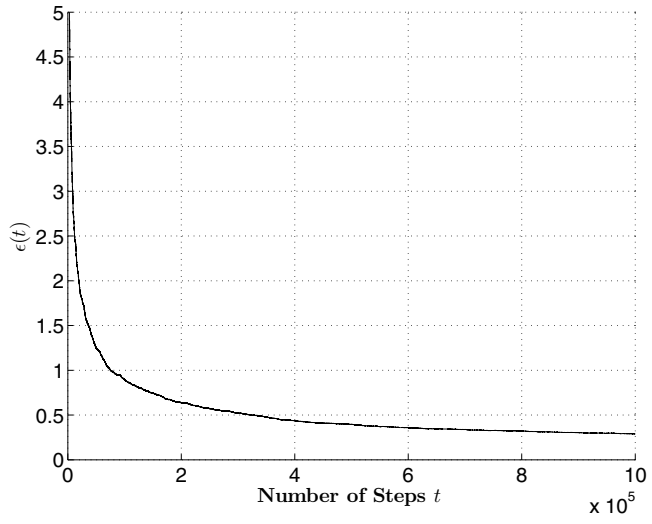

Fig. 1. $\epsilon(t)$ vs t

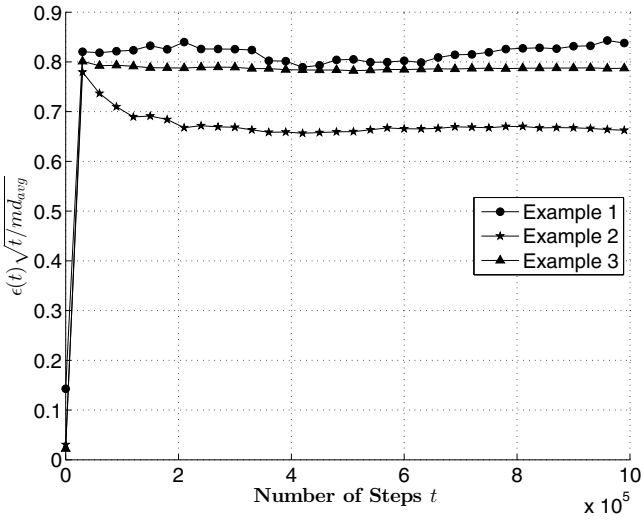

Fig. 3. $\epsilon(t)\sqrt{t/md_{avg}}$ vs t

to the true conductances as follows: the L^1-norm of the difference normalized by the number of edges,

$$\epsilon(t) = \frac{1}{m} \sum_{(i,j)\in E} \left| \widehat{G}_{ij}(t) - G_{ij} \right|, \tag{3}$$

was computed and plotted versus time t to demonstrate convergence.

We created three example graphs to test the convergence properties of our algorithm. These are described below.

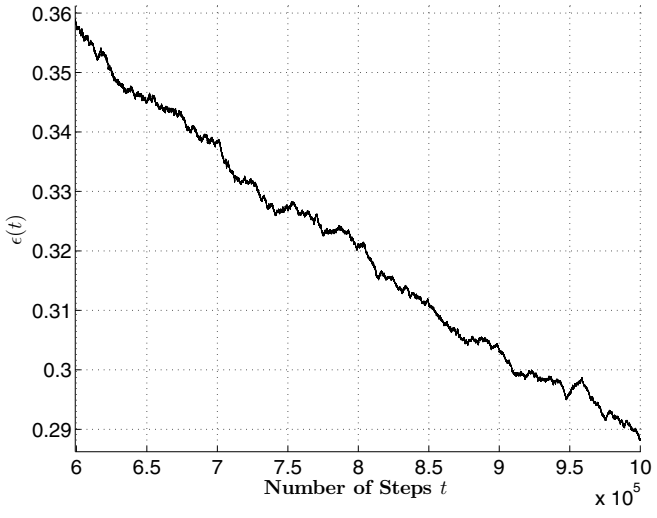

Fig. 2. An expanded version of Fig. 1.

Example 1: A dense Erdős-Rényi graph: A random Erdős-Rényi graph was created using the `random_graph` subroutine from [8] with $p = 0.5$. The number of nodes n was 100. Connectedness of the graph was ensured before simulation.

Example 2: A collaboration network: This is a large graph from the Stanford Network Analysis Platform [9] and is a component of the collaboration network of researchers working on General Relativity and Quantum Cosmology, and is derived from the e-print Arxiv GR-QC. Two individuals are neighbors on the graph if they coauthored a paper. The graph is based on papers in the period from January 1993 to April 2003 (124 months). Additionally, the graph was pre-processed to assign a weight of 1 for every edge. The largest connected component was then chosen. The resulting graph had $n = 4158$ nodes and $m = 13425$ edges. The degree distribution was not uniform. There were 3422 nodes with degree 1, but only 414 nodes with degree 2, 155 nodes with degree 3, 47 nodes with degree 4, 50 nodes with degree 5, and so on.

Example 3: An expander with $O(\log n)$ average degree: A random Erdős-Rényi graph $G(n,p)$ with $p = \log(n)/n$ was generated and G was taken to be the giant component. We used the `random_graph` subroutine once again from [8] with $n = 4000$ and $p = 50/4000$. It is known that G is an expander with high probability.

The number of steps of the random walk T was set to $1,000,000$. In Figure 1 we plot $\epsilon(t)$ as a function of t for Example 1. In Figure 2, we blow up the latter part and see that $\epsilon(t)$ continues to decrease with time, albeit slowly. The behavior is similar for the other two graphs.

From Theorem 1, if we can ignore the condition on ϵ, $T = \widetilde{O}(1/\epsilon^2)$ would suffice for approximating edge conductances on a fixed graph. We therefore anticipate that the error ϵ is of the order $1/\sqrt{T}$. A closer look suggests the following. If t_{mix} and d_{max} are $\widetilde{O}(1)$, and we can ignore the condition on ϵ, then $T = \widetilde{O}(md_{max}/\epsilon^2)$ steps suffice, which suggests $\epsilon(t) \approx \sqrt{md_{max}/t}$. A plot of $\epsilon(t)\sqrt{t}/\sqrt{md_{max}}$ versus t should roughly be a constant. This is corroborated in Figure 3. We used d_{avg} instead of d_{max} in Figure 3 because the plots for the three examples turn out to be closer to each other in addition to being roughly constant.

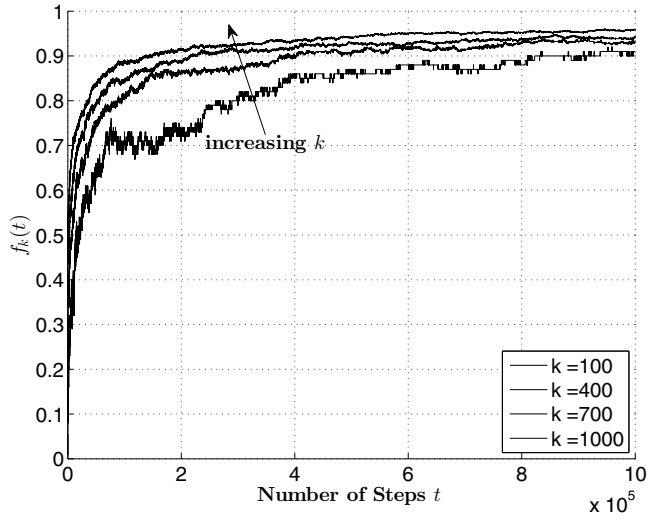

Fig. 4. Fraction of top-k largest conductance edges correctly identified at time t for Example 1

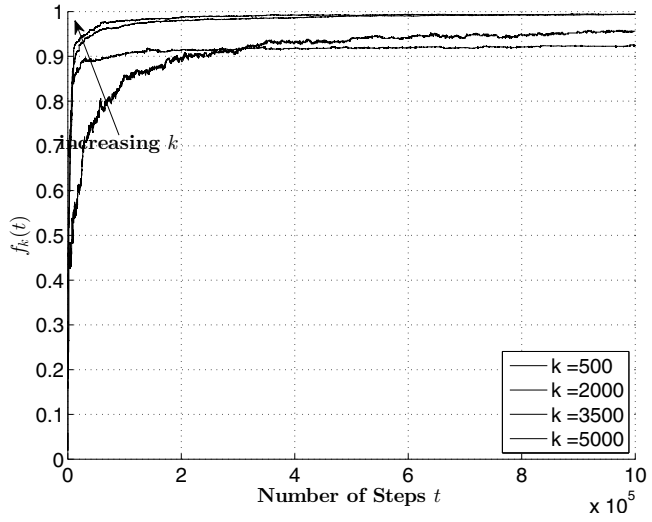

Fig. 5. Fraction of top-k largest conductance edges correctly identified at time t for Example 2

B. Top-k edge-conductance set estimation

We study the performance of our algorithm in recovering the top-k largest conductance edges. Notice that this problem is different from conductance estimation, because we are now interested in only ranking the edges in the decreasing order of their conductances. Ordinal estimation problems can usually be solved much faster than their corresponding cardinal estimation versions.

In Figures 4-6, we plot the fraction $f_k(t)$ of the top-k largest conductance edges that were correctly identified as a function of the number of steps t taken in the random walk. There is one plot for each example. In each plot, there is one curve for each indicated k.

As expected, the plots suggest that the estimated set of top-k largest conductance edges improves with number of steps.

IV. Conclusion

Conductances are useful quantities that capture important aspects of a network's structure. They are used in several important and practical applications. Many were highlighted in Section I. A particularly useful application is the use of approximate edge conductances in graph sparsification. In this paper, we proposed a MCMC based conduction estimation algorithm for edge conductance estimation. It is based on the idea that an edge's effective conductance G_{ij} equals $d_i p_{ij}$, where p_{ij} is the probability that a random walk starting at node i visits node j before returning to node i. While we focused on estimation of conductances of edges, our algorithm can be easily adapted for estimating the conductance value across any pair of nodes: maintain and update the variable $\widehat{p}_{ij}$ and $\widetilde{p}_{ij}$ for every pair (or desired pairs) of nodes at each step. Using concentration analysis, we obtained sample complexities and provided corresponding probabilistic and approximation guarantees. The theoretical performance guarantees are weak compared to existing centralized algorithms. But numerical experiments showed optimistic results – the convergence of estimated conductances to true conductances is at the rate predicted if one removes the restrictive assumption in Theorem 1 on ϵ. We also observed that the evolution of top-k conductance set improves with the number of iterations. Thus the scheme is well suited for ordinal inference.

Our algorithm is amenable to distributed implementation when only effective conductances of edges are required. Each node i keeps track of the number of visits of the random walk to itself. When node i is visited, it updates counters (one for each of its neighbors j) indicating that i was visited (in a j-to-j walk). It also gathers information from each neighbor j on whether j was visited in the just concluded i-to-i walk. This information is obtained via a local communication. Finally, the next node of the walk is also determined via locally

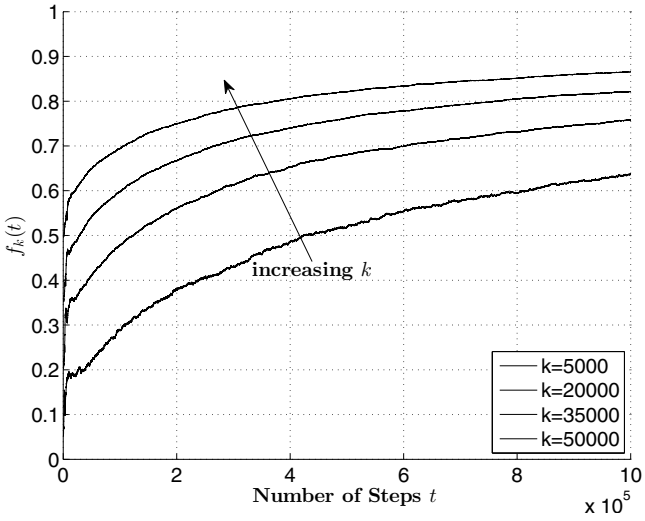

Fig. 6. Fraction of top-k largest conductance edges correctly identified at time t for Example 3

generated randomness. If effective conductances between far-off nodes is desired, the communication is however no longer local. Our algorithm is also amenable to parallelization. One could execute multiple such random walks in parallel, each independent of the others, and then average them to get sharper estimates.

APPENDIX

PROOF OF PERFORMANCE OF THE CONDUCTANCE ESTIMATION ALGORITHM

In this Appendix, we prove Theorem 1. The total number of visits to node i is N_i. Let $\widetilde{p}_{ij}^k$ be the indicator that node j was visited between the $(k-1)^{st}$ and k^{th} visit to node i. The zeroth visit is assumed to have taken place prior to time 0. From step 5-(a)-(i) of the algorithm **VisitBeforeReturn**, we have

$$\widehat{p}_{ij} = \frac{\sum_{k=1}^{N_i} \widetilde{p}_{ij}^k}{N_i}. \tag{4}$$

The main idea is to obtain a concentration inequality for the denominator and then use it to get a concentration inequality for $\widehat{p}_{ij}$.

We begin by stating a result from [10], which is essentially McDiarmid's inequality for Markov chains.

Lemma 2 (Cor. 2.11 of [10].): Let $(X_t, t \geq 1)$ be the random walk on G. Let its mixing time be t_{mix}. Assume that X_1 has the stationary distribution. Let $X = (X_1, \ldots, X_T) \in V^T$ be the first T steps in this random walk. Suppose that for some $c \in R_+^T$, a function $f : V^T \to \mathbb{R}$ satisfies the following (Lipschitz) condition,

$$f(x) - f(y) \leq \sum_{t=1}^{T} c_t \mathbb{I}[x_t \neq y_t] \quad \forall x, y. \tag{5}$$

Then for any $\epsilon \geq 0$, we have

$$\mathbb{P}(|f(X) - \mathbb{E}f(X)| \geq \epsilon) \leq 2\exp\left(\frac{-2\epsilon^2}{9\|c\|^2 t_{mix}}\right).$$

□

We refer the reader to [10] for a proof. We now obtain a concentration inequality for N_i, the denominator of (4).

Lemma 3: Let $(X_t, t \geq 1)$ be the random walk on G. Let its mixing time be t_{mix}. Assume that X_1 has the stationary distribution π. For $i \in V$, let $N_i := \sum_{t=1}^{T} \mathbb{I}[X_t = i]$ be the number of times node i is visited by this random walk upto time T. Then, for any $i \in V$, any $\epsilon \geq 0$,

$$\mathbb{P}\left(\left|\frac{N_i}{T} - \pi_i\right| \geq \epsilon\right) \leq 2\exp\left(\frac{-2T\epsilon^2}{9t_{mix}}\right). \tag{6}$$

□

Proof: Fix a node i. Given $X = (X_1, \ldots, X_T)$, define $f(X) := N_i/T$. Since X_1 has the stationary distribution π, we have

$$\mathbb{E}(f(X)) = \frac{\mathbb{E}(N_i)}{T} = \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}(\mathbb{I}[X_t = i]) = \frac{1}{T}\sum_{t=1}^{T} \pi_i = \pi_i.$$

Furthermore, for any $x, y \in V^T$, we have

$$\begin{aligned} f(x) - f(y) &= \frac{1}{T}\sum_{t=1}^{T} (\mathbb{I}[x_t = i] - \mathbb{I}[y_t = i]) \\ &\leq \frac{1}{T}\sum_{t=1}^{T} (\mathbb{I}[x_t \neq y_t]), \end{aligned}$$

and so $f(\cdot)$ satisfies (5) with $c_t = 1/T$. By Lemma 2, we get (6). ■

Our next step is to get the concentration for (4).

Lemma 4: Let $X = (X_1, \ldots, X_T)$ be T steps of the random walk on G. Let X_1 have the stationary distribution π. For $(i, j) \in E$, let $\widehat{p}_{ij}$ be as defined in (4) and let p_{ij} be the probability that a random walk starting at node i visits node j before returning to node i. Then, for any $(i, j) \in E$, for any $0 < \epsilon \leq d_{ij}/(4m)$, and for $T > 1/\epsilon$, we have

$$\begin{aligned} &\mathbb{P}(|\widehat{p}_{ij} - p_{ij}| \geq \epsilon) \\ &\quad \leq 2\exp\left(\frac{-2T\epsilon^2}{9t_{mix}}\right) + 8\epsilon T \exp\left(\frac{-\epsilon^2 T \pi_i}{4}\right). \end{aligned} \tag{7}$$

□

Proof: Define $X_k := \widetilde{p}_{ij}^k - p_{ij}$ and $S_K := \sum_{k=1}^{K} X_k$. Since $|X_k| \leq 1$, and X_k's are i.i.d., using Hoeffding's inequality [11], we have

$$\mathbb{P}\left(\left|\frac{S_K}{K}\right| \geq \epsilon\right) \leq 2\exp\left(-\frac{K\epsilon^2}{2}\right), \quad \forall\ K \geq 1.$$

We therefore have

$$\begin{aligned} &\mathbb{P}\left(|\widehat{p}_{ij} - p_{ij}| \geq \epsilon\right) \\ &= \mathbb{P}\left(\frac{1}{N_i}\left|\sum_{k=1}^{N_i} \left(\widetilde{p}_{ij}^k - p_{ij}\right)\right| \geq \epsilon\right) \\ &= \mathbb{P}\left(\left|\frac{S_{N_i}}{N_i}\right| \geq \epsilon\right) \\ &= \mathbb{P}\left(\left|\frac{S_{N_i}}{N_i}\right| \geq \epsilon, \left|\frac{N_i}{T} - \pi_i\right| \geq \epsilon\right) \\ &\quad + \mathbb{P}\left(\left|\frac{S_{N_i}}{N_i}\right| \geq \epsilon, \left|\frac{N_i}{T} - \pi_i\right| < \epsilon\right) \\ &\leq \mathbb{P}\left(\left|\frac{N_i}{T} - \pi_i\right| \geq \epsilon\right) \\ &\quad + \sum_{k=\lceil T(\pi_i - \epsilon)\rceil}^{\lfloor T(\pi_i + \epsilon)\rfloor} \mathbb{P}\left(\left|\frac{S_k}{k}\right| \geq \epsilon, N_i = k\right). \end{aligned} \tag{8}$$

Since $\pi_i = d_i/(2m) > 2\epsilon$, we have $T(\pi_i - \epsilon) \geq \pi_i/2$. Focusing on the second term in (8), for $\lceil T(\pi_i - \epsilon)\rceil \leq k \leq$

$\lfloor T(\pi_i+\epsilon)\rfloor$, we have

$$\begin{aligned}
&\mathbb{P}\left(\left|\frac{S_k}{k}\right|\geq\epsilon, N_i=k\right)\\
&\leq \mathbb{P}\left(\left|\frac{S_k}{k}\right|\geq\epsilon\right)\\
&\leq \max_{\lceil T(\pi_i-\epsilon)\rceil\leq k\leq\lfloor T(\pi_i+\epsilon)\rfloor}\mathbb{P}\left(\left|\frac{S_k}{k}\right|\geq\epsilon\right)\\
&\leq 2\exp\left(-\frac{\epsilon^2}{2}\lceil T(\pi_i-\epsilon)\rceil\right)\\
&\leq 2\exp\left(-\frac{\epsilon^2}{2}T(\pi_i-\epsilon)\right)\\
&\leq 2\exp\left(-\frac{\epsilon^2}{4}T\pi_i\right).
\end{aligned}$$

Plugging this upper bound into (8), we get that (8) is upper bounded by

$$\mathbb{P}\left(\left|\frac{N_i}{T}-\pi_i\right|\geq\epsilon\right)+2\lceil\epsilon T\rceil\times 2\exp\left(-\frac{\epsilon^2}{4}T\pi_i\right).$$

Further, since $\epsilon T>1$, we have $\lceil\epsilon T\rceil\leq 2\epsilon T$. This and Lemma 3 (to bound the first term) yield the desired result. ∎

We are now ready to finish the proof of Theorem 1.

Proof of Theorem 1: The conductance estimate is given by

$$\widehat{G}_{ij}=\max\left(1,\frac{d_i}{2}\widehat{p}_{ij}+\frac{d_j}{2}\widehat{p}_{ji}\right).$$

Subtracting G_{ij}, and using $G_{ij}\geq 1$, we get the following upper bound on the absolute value of the error:

$$|\widehat{G}_{ij}-G_{ij}|\leq\frac{d_i}{2}|\widehat{p}_{ij}-p_{ij}|+\frac{d_j}{2}|\widehat{p}_{ji}-p_{ji}|.$$

Thus,

$$\begin{aligned}
&\mathbb{P}(|\widehat{G}_{ij}-G_{ij}|\geq\epsilon)\\
&\leq \mathbb{P}\left(\frac{d_i}{2}|\widehat{p}_{ij}-p_{ij}|+\frac{d_j}{2}|\widehat{p}_{ji}-p_{ji}|\geq\epsilon\right)\\
&\leq \mathbb{P}\left(|\widehat{p}_{ij}-p_{ij}|\geq\frac{\epsilon}{d_i}\right)+\mathbb{P}\left(|\widehat{p}_{ji}-p_{ji}|\geq\frac{\epsilon}{d_j}\right), \quad (9)
\end{aligned}$$

where the last inequality follows by the union bound. Using (7) with ϵ/d_i replacing ϵ, assuming $\epsilon/d_i<d_i/(4m)$ and $T>d_i/\epsilon$, and by noting that $\pi_i=d_i/2m$, we get

$$\begin{aligned}
&\mathbb{P}\left(|\widehat{p}_{ij}-p_{ij}|\geq\frac{\epsilon}{d_i}\right)\\
&\leq 2\exp\left(\frac{-2T\epsilon^2}{9d_i^2t_{mix}}\right)+\frac{8\epsilon T}{d_i}\exp\left(-\frac{\epsilon^2T}{8md_i}\right). \quad (10)
\end{aligned}$$

A similar bound applies for the second term in (9) after assuming $\epsilon/d_j<d_j/(4m)$ and $T>d_j/\epsilon$.

Observe that one obtains $c_2Te^{-c_1T}\leq\delta/4$ whenever $T=(2/c_1)\log(4c_2/(\delta c_1))$ or larger. Choosing c_1 and c_2 appropriately for the second term in (10), we note that

$$T_1^i:=\frac{16md_i}{\epsilon^2}\log\left(\frac{256m}{\epsilon\delta}\right)$$

steps suffice to ensure that the second term in (10) is less than $\delta/4$. To ensure that the first term in (10) is less than $\delta/4$, we see that

$$T_2^i:=\frac{9d_i^2t_{mix}}{2\epsilon^2}\log\frac{8}{\delta}$$

steps suffice. Combining all the requirements and the assumptions on T, we have that

$$\begin{aligned}
T &= \max\left\{\frac{d_i}{\epsilon},\frac{d_j}{\epsilon},T_1^i,T_2^i,T_1^j,T_2^j\right\}\\
&= \widetilde{O}\left(D_{ij}\cdot\max\{m,D_{ij}t_{mix}\}\cdot\frac{1}{\epsilon^2}\log\frac{1}{\delta}\right)
\end{aligned}$$

steps suffice to ensure that the right-hand side of (9) is less than or equal to δ.

At every time step of the algorithm, if $X_t=i$, $O(d_i)$ computations are performed. The next step of the random walk requires at most $O(\log d_i)$ number of Bernoulli$(1/2)$ random variables generation, each of which we assume takes $O(1)$ steps. Taking both into consideration, in the long run, the averaged total computation is $O(d_{avg}T)$ (worst case is $O(d_{max}T)$).

We need to store the variables N_i, $N_i\widehat{p}_{ij}$, $\widetilde{p}_{ij}$, and $\widehat{G}_{ij}$ for each $i\in V$ and $j\in\partial_i$. Exact representation of each these requires $O(\log T)$ storage space. Thus the overall space requirement is $O(m\log T)$. ∎

ACKNOWLEDGMENTS

This work was supported by the Indo-French Centre for the Promotion of Advanced Research under grant No. 5100-ITA, by an IBM SUR grant, and by the Robert Bosch Foundation for Cyber Physical Systems.

REFERENCES

[1] A. Firat, S. Chatterjee, and M. Yilmaz, "Genetic clustering of social networks using random walks," *Computational Statistics & Data Analysis*, vol. 51, no. 12, pp. 6285–6294, 2007.

[2] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, "Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation," *Knowledge and Data Engineering, IEEE transactions on*, vol. 19, no. 3, pp. 355–369, 2007.

[3] W. Ellens, F. Spieksma, P. Van Mieghem, A. Jamakovic, and R. Kooij, "Effective graph resistance," *Linear algebra and its applications*, vol. 435, no. 10, pp. 2491–2506, 2011.

[4] A. Tizghadam and A. Leon-Garcia, "On robust traffic engineering in transport networks," in *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*. IEEE, 2008, pp. 1–6.

[5] D. A. Spielman and N. Srivastava, "Graph sparsification by effective resistances," *SIAM Journal on Computing*, vol. 40, no. 6, pp. 1913–1926, 2011.

[6] L. Lovász, "Random walks on graphs: A survey," *Combinatorics, Paul Erdös is eighty*, vol. 2, no. 1, pp. 1–46, 1993.

[7] W. B. Johnson and J. Lindenstrauss, "Extensions of lipschitz mappings into a hilbert space," *Contemporary mathematics*, vol. 26, no. 189-206, p. 1, 1984.

[8] G. Bounova and O. de Weck, "Overview of metrics and their correlation patterns for multiple-metric topology analysis on heterogeneous graph ensembles," *Physical Review E*, vol. 85, no. 1, p. 016117, 2012.

[9] J. Leskovec and R. Sosic, "SNAP: A general purpose network analysis and graph mining library in C++," 2014.

[10] D. Paulin, "Concentration inequalities for markov chains by marton couplings and spectral methods," *arXiv preprint, arXiv:1212.2015*, 2012.

[11] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *Journal of the American statistical association*, vol. 58, no. 301, pp. 13–30, 1963.