DECENTRALIZED COMPUTATION OF EFFECTIVE RESISTANCES AND ACCELERATION OF CONSENSUS ALGORITHMS

Necdet Serhat Aybat[*]

Department of IME
Pennsylvania State University
University Park, PA, USA
nsa10@psu.edu

Mert Gürbüzbalaban[†]

Department of MSIS
Rutgers Business School
Piscataway, NJ, USA
mg1366@rutgers.edu

ABSTRACT

The effective resistance between a pair of nodes in a weighted undirected graph is defined as the potential difference induced between them when a unit current is injected at the first node and extracted at the second node, treating edge weights as the conductance values of edges. The effective resistance is a key quantity of interest in many applications and fields including solving linear systems, Markov Chains and continuous-time averaging networks. We develop an efficient linearly convergent distributed algorithm for computing effective resistances and demonstrate its performance through numerical studies. We also apply our algorithm to the consensus problem where the aim is to compute the average of node values in a distributed manner. We show that the distributed algorithm we developed for effective resistances can be used to accelerate the convergence of the classical consensus iterations considerably by a factor depending on the network structure.

Index Terms— Effective resistance, graph, distributed optimization, consensus, Laplacian matrix, Kaczmarz method

1. INTRODUCTION

Let $\mathcal{G} = (\mathcal{N}, \mathcal{E}, w)$ be an undirected, weighted and connected graph defined by the set of nodes (agents) $\mathcal{N} = \{1, \ldots, n\}$, the set of edges $\mathcal{E} \subset \mathcal{N} \times \mathcal{N}$, and the edge weights $w_{ij} > 0$ for $(i, j) \in \mathcal{E}$. Since $\mathcal{G}$ is undirected, we assume that both (i, j) and (j, i) refer to the same edge when it exists, and for all $(i, j) \in \mathcal{E}$, we set $w_{ji} = w_{ij}$. Identifying the weighted graph $\mathcal{G}$ as an electrical network in which each edge (i, j) corresponds to a branch of conductance w_{ij}, the effective resistance R_{ij} between a pair of nodes i and j is defined as the voltage potential difference induced between them when a unit current is injected at i and extracted at j.

The effective resistance, also known as the resistance distance, is a key quantity of interest to compute in many applications and algoritmic questions over graphs. It defines a metric on graphs providing bounds on its conductance [1, 2]. Furthermore, it is closely associated with the hitting time and commute time for a random walk[1] on the graph G such that the probability of a transition from i to $j^* \in \mathcal{N}_i$ is $w_{ij^*} / \sum_{j \in \mathcal{N}_i} w_{ij}$ where $\mathcal{N}_i \triangleq \{j \in \mathcal{N} : (i, j) \in \mathcal{E}\}$ denotes the set of neighboring nodes of $i \in \mathcal{N}$; therefore, it arises naturally for studying random walks over graphs and their mixing time properties [3, 4, 5], continuous-time averaging networks including consensus problems in distributed optimization [3]. Other prominent applications include distributed control and estimation [6], solving symmetric diagonally dominant (SDD) linear systems [7], deriving complexity bounds in the Asymmetric Traveling Salesman Problem (ATSP) [8], design and control of communication networks [9, 10] and spectral sparsification of graphs [11].

There exist *centralized* algorithms for computing or approximating $\{R_{ij}\}_{i \neq j}$ accurately which require global communication beyond local communication among the neighboring agents [7, 12]. They are based on computing or approximating the entries of the pseudoinverse $\mathcal{L}^+$ of the Laplacian matrix, based on the identity $R_{ij} = \mathcal{L}^+_{ii} + \mathcal{L}^+_{jj} - 2\mathcal{L}^+_{ij}$ [7]. However, such centralized algorithms are *impractical* or *infeasible* for several key applications in multi-agent systems where only local communications between the neighboring agents are allowed; this motivates the development of distributed algorithms for computing effective resistances, which are used in solving many optimization and estimation problems over graphs. Prominent examples include, least square regression and more general estimation problems over graphs, formation control of moving agents with noisy measurements and stability of multi-vehicle swarms [6].

To our knowledge, there has been no systematic study of distributed algorithms for computing effective resistances. In this work, we discuss how existing algorithms in the distributed optimization literature for solving linear systems can be adapted to solve this problem. First, we show that a naive implementation of consensus optimization methods, e.g., the EXTRA algorithm [13] is inefficient in terms of the convergence and communication requirements. Second, we propose

[*]Research of N. S. Aybat was partially supported by NSF grants CMMI-1400217 and CMMI-1635106, and ARO grant W911NF-17-1-0298.

[†]Research of M. Gürbüzbalaban was partially supported by the NSF grant DMS-1723085.

[1]The hitting time H_{ij} is the expected number of steps of a random walk starting from i until it first visit j. The commute time C_{ij} is the expected number of steps required to go from i to j and back again.

a variant of the Kaczmarz method and show that it is linearly convergent while being efficient in terms of total number of local communications carried out. Third, we demonstrate the performance of our algorithms on numerical examples. In particular, numerical experiments suggest finite convergence of our algorithms which is of independent interest. Finally, we apply our results to the consensus problem [14] where the aim is to compute the average of values assigned to each node in a distributed manner. Specifically, we propose a variant of the classical *asynchronous* consensus protocol and show that we can accelerate the convergence considerably by a factor depending on the underlying network. The main idea is to use the distributed algorithm we developed for effective resistances to design a weight matrix which can help pass the information among neighbors more *effectively* – an alternative approach in [15] also builds on modifying the weights depending on the degree of the neighbors. Since the consensus iterations are the building block of many existing core distributed optimization algorithms such as the distributed subgradient, distributed proximal gradient and ADMM methods; we believe that our method and framework have far-reaching potential for accelerating many other distributed algorithms in addition to consensus algorithms, and this will be the subject of future work.

Outline. In Section 2, we introduce our algorithm for computing effective resistances. In Section 3, we provide numerical results; finally, in Section 4, we give a summary of our results and discuss future work.

Notation. Let $d_i \triangleq |\mathcal{N}_i|$ denote the degree of $i \in \mathcal{N}$, and $m \triangleq |\mathcal{E}|$. Throughout the paper, $\mathcal{L} \in \mathbb{R}^{|\mathcal{N}|\times|\mathcal{N}|}$ denotes the weighted Laplacian of $\mathcal{G}$, i.e., $\mathcal{L}_{ii} = \sum_{j\in\mathcal{N}_i} w_{ij}$, $\mathcal{L}_{ij} = -w_{ij}$ if $j \in \mathcal{N}_i$, and equal to 0 otherwise. The set $\mathbb{S}^n$ denotes the set of $n \times n$ real symmetric matrices. We use the notation $Z = [z_i]_{i=1}^n$ where z_i's are either the columns or rows of the matrix Z depending on the context. $\mathbf{1}$ is the column vector with all entries equal to 1, and $\mathbf{I}$ is the identity matrix.

2. METHODOLOGY

Clearly, $\mathcal{L}$ is symmetric and positive semidefinite; and since $\mathcal{G}$ is connected, the nullspace of $\mathcal{L}$ is spanned by $\mathbf{1}$. In particular, consider the eigenvalue decomposition $\mathcal{L} = \sum_{i=1}^n \lambda_i u_i u_i^\top$; we have $0 = \lambda_1 < \lambda_2 \leq \ldots \leq \lambda_n$ and $u_1 = \frac{1}{\sqrt{n}}\mathbf{1}$. Recall that we would like to compute $\mathcal{L}^\dagger = \sum_{i=2}^n \frac{1}{\lambda_i} u_i u_i^\top$ in a decentralized way. First, we are going to describe a naive way to solve this problem which would converge with a linear rate, but require storing and communicating $n \times n$ matrices among the neighboring nodes. Next, we discuss that $\mathcal{L}^\dagger$ can be computed in a distributed way using the (randomized) Kaczmarz (RK) method with significantly less communication burden.

2.1. A consensus-based naive method for computing $\mathcal{L}^\dagger$:

Let $\theta \geq \lambda_2$ and define $\bar{\mathcal{L}} \triangleq \mathcal{L} + \frac{\theta}{n}\mathbf{1}\mathbf{1}^\top$, i.e., $\bar{\mathcal{L}} = \theta u_1 u_1^\top + \sum_{i=2}^n \lambda_i u_i u_i^\top$; hence, $\bar{\mathcal{L}}^{-1} = \mathcal{L}^\dagger + \frac{1}{\theta} u_1 u_1^\top$. To compute $\bar{\mathcal{L}}^{-1}$, consider solving $(P) : \min_{X\in\mathbb{S}^n} f(X) \triangleq \frac{1}{2}\left\|\bar{\mathcal{L}}X - \mathbf{I}\right\|_F^2$. Note that f is strongly convex with modulus λ_2 since $\theta \geq \lambda_2$; moreover, such θ can be chosen easily in certain cases. For instance, for unweighted $\mathcal{G}$, i.e., $w_{ij} = 1$ for $(i,j) \in \mathcal{E}$, it is known that $\lambda_2 \leq \min_{i\in\mathcal{N}} d_i$; hence, θ could be chosen after running a min-consensus algorithm over $\mathcal{G}$. To solve (P) in a decentralized manner, we will exploit connectivity of $\mathcal{G}$. Let $\bar{\ell}_i \in \mathbb{R}^n$ be a column vector for $i \in \mathcal{N}$ such that $\bar{\mathcal{L}} = [(\bar{\ell}_i)^\top]_{i\in\mathcal{N}}$, i.e., $(\bar{\ell}_i)^\top$ denotes the i-th row of $\bar{\mathcal{L}}$. (P) can be equivalently written as follows:

$$(P') : \min_{X_i\in\mathbb{S}^n,\ i\in\mathcal{N}} \left\{ \sum_{i\in\mathcal{N}} \left\|X_i\bar{\ell}_i - e_i\right\|_2^2 :\ X_i = X_j\ \forall\,(i,j)\in\mathcal{E} \right\},$$

where e_i denotes the i-th standard basis vector of $\mathbb{R}^n$. Although this problem is not strongly convex in $[X_i]_{i\in\mathcal{N}}$, there is a way to regularize the objective $\bar{f}([X_i]_{i\in\mathcal{N}}) \triangleq \sum_{i\in\mathcal{N}} \left\|X_i\bar{\ell}_i - e_i\right\|_2^2$ to make it strongly convex. Indeed, it can be shown that for $\alpha > 0$ sufficiently large, $\bar{f}_\alpha \triangleq \bar{f} + \alpha r$ is strongly convex in $[X_i]_{i\in\mathcal{N}}$, where $r([X_i]_{i\in\mathcal{N}}) \triangleq \sum_{(i,j)\in\mathcal{E}} \|X_i - X_j\|_F^2$; and one can equivalently consider $\min\{\bar{f}_\alpha([X_i]_{i\in\mathcal{N}}) :\ X_i = X_j\ (i,j)\in\mathcal{E}\}$. In particular, the algorithm EXTRA in [13] exploits a similar restricted strong convexity argument and achieves a linear convergence rate for the iterate sequence. That said, the communication overhead is the main problem with this approach of solving (P'). In fact, at each iteration k, each node $i \in \mathcal{N}$ communicates its local estimate X_i^k to its neighbors $\mathcal{N}_i$; thus, each iteration of these consensus based methods would require $\mathcal{O}(2|\mathcal{E}|n^2)$ real variable communications in total, e.g., EXTRA. Next, we discuss the distributed implementation of the RK method to compute $\mathcal{L}^\dagger$, which would prove itself as a more communication efficient and practical method.

2.2. Distributed Kaczmarz method for computing $\mathcal{L}^\dagger$:

Consider a *consistent* system $Ax = b$, where $A = [a_i^\top]_{i=1}^m \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. Suppose A has no rows with all zeros, and let $x^* = \operatorname{argmin}\{\|x\|_2 :\ Ax = b\}$. In [16], it is shown that x^* can be computed using a randomized Kaczmarz method. In particular, it follows from the results in [16] that starting from $x^0 \in \mathbf{Null}(A)$, the method displayed in Algorithm 1 produces $\{x^k\}_{k\geq 1}$ such that $\mathbb{E}[\|x^k - x^*\|_2^2] \leq \rho^k \|x^0 - x^*\|_2^2$ for $k \geq 0$ with $\rho \triangleq 1 - \lambda_{\min}^+(A^\top H A)$ where $\lambda_{\min}^+(\cdot)$ denotes the smallest positive eigenvalue and $H = \sum_{i=1}^m p_i \frac{1}{\|a_i\|_2^2} e_i e_i^\top$; furthermore, $1 - \frac{1}{\mathbf{rank}(A)} \leq \rho < 1$. Note that fixing $p_i = \|a_i\|_2^2 / \|A\|_F^2$ gives us the randomized Kaczmarz in [17, 18].

Algorithm 1: RK($\{p_i\}_{i=1}^m$) – Randomized Kaczmarz

Initialization: $x^0 \in \mathbf{Null}(A)$
for $k \geq 0$ **do**
  Pick $i \in \{1,\ldots,m\}$ with probability p_i
  $x^{k+1} \leftarrow x^k - \frac{1}{\|a_i\|^2}(a_i^\top x^k - b_i)a_i$

Note $\mathcal{L}\mathcal{L}^\dagger = \sum_{i=2}^n u_i u_i^\top$ and $\mathbf{I} = \sum_{i=1}^n u_i u_i^\top$; hence, $\mathcal{L}\mathcal{L}^\dagger = \mathbf{I} - u_1 u_1^\top = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$. Although the solution set

$\{X \in \mathbb{S}^n : \mathcal{L}X = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^\top\}$ has infinitely many elements, it is well-known that $\mathcal{L}^\dagger$ is the unique solution to

$$\mathcal{L}^\dagger = \underset{X \in \mathbb{S}^n}{\operatorname{argmin}} \{\|X\|_F : \mathcal{L}X = B\}, \tag{1}$$

where $B \triangleq \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$. Let $x^l, b^l \in \mathbb{R}^n$ for $l \in \mathcal{N}$ be column vectors such that $X = [x^l]_{l\in\mathcal{N}}$ and $B = [b^l]_{l\in\mathcal{N}}$, i.e., $b^l = e_l - \frac{1}{n}\mathbf{1}$. Note n columns of $\mathcal{L}^\dagger$ can be computed in parallel:

$$x^l_* \triangleq \underset{x \in \mathbb{R}^n}{\operatorname{argmin}} \{\|x\|_2 : \mathcal{L}x = b^l\}, \quad l \in \mathcal{N}, \tag{2}$$

i.e., $\mathcal{L}^\dagger = [x^l_*]_{l\in\mathcal{N}}$. Since $\mathcal{L}^\dagger\mathbf{1} = \mathbf{0}$, $x^n_* = -\sum_{l=1}^{n-1} x^l_*$. Thus, one does not need to solve for all $l \in \mathcal{N}$; it suffices to compute $\{x^l_*\}_{l\in\mathcal{N}\setminus\{n\}}$ and calculate x^n_* from these.

Let $\{x^{l,k}\}_{k\geq 1}$ be the sequence generated when RK implemented on (2) for $l \in \mathcal{N} \setminus \{n\}$. In Algorithm 2, we summarized the distributed nature of RK steps assuming that each $i \in \mathcal{N}$ has an exponential clock with rate $r_i > 0$, and when its clock ticks, the node i wakes up and communicates with its neighbors $j \in \mathcal{N}_i$ on $\mathcal{G}$. More precisely, consider the resulting superposition of these point processes, and let $\{t_k\}_{k\in\mathbb{Z}_+}$ be the times such that one of the clocks ticks; hence, for all $k \geq 0$, the node that wakes up at time t_k is node i with probability $p_i = r_i / \sum_{j\in\mathcal{N}} r_i$, i.e., $\{t_k\}_{k\geq 0}$ denotes the arrival times of a Poisson process with rate $\sum_{j\in\mathcal{N}} r_i$.

Algorithm 2: D-RK($\{r_i\}_{i\in\mathcal{N}}$) – Decentralized RK

Initialization: $x_i^{l,0} \leftarrow 0$ for $l \in \mathcal{N} \setminus \{n\}$ and $i \in \mathcal{N}$
for $k \geq 0$ **do**
  At time t_k, $i \in \mathcal{N}$ wakes up w.p. $p_i = \frac{r_i}{\sum_{j\in\mathcal{N}} r_i}$
  for $l \in \mathcal{N} \setminus \{n\}$ **do**
    Node i requests and receives $x_j^{l,k}$ from $j \in \mathcal{N}_i$
    Node i computes and sends $q_i^{l,k}$ to all $j \in \mathcal{N}_i$
    $q_i^{l,k} = \frac{1}{\sum_{j\in\mathcal{N}_i\cup\{i\}} \mathcal{L}_{ij}^2}(\sum_{j\in\mathcal{N}_i\cup\{i\}} \mathcal{L}_{ij}x_j^{l,k} - b_i^l)$
    Each $j \in \mathcal{N}_i \cup \{i\}$ updates $x_j^{l,k+1} \leftarrow x_j^{l,k} - \mathcal{L}_{ij}q_i^{l,k}$

For $k \geq 0$, let $X^k \triangleq [x^{l,k}]_{l\in\mathcal{N}}$ be the concatenation of D-RK sequence, where $x^{n,k} \triangleq -\sum_{l=1}^{n-1} x^{l,k}$, and define $S = \mathbf{diag}(s)$ such that $s_i \triangleq \sum_{j\in\mathcal{N}_i\cup\{i\}} \mathcal{L}_{ij}^2$ for $i \in \mathcal{N}$. According to [16, 17], for $r_i = s_i$, we get $H = \frac{1}{\|\mathcal{L}\|_F^2}\mathbf{I}$, and this implies linear convergence of $\{X^k\}_{k\geq 0}$ to $\mathcal{L}^\dagger$ with rate $\rho = 1 - \left(\frac{\lambda^+_{\min}(\mathcal{L})}{\|\mathcal{L}\|_F}\right)^2$, i.e., $\mathbb{E}[\left\|X^k - \mathcal{L}^\dagger\right\|_F^2] \leq \rho^k \left\|\mathcal{L}^\dagger\right\|_F^2$ for $k \geq 0$. Moreover, for each $i \in \mathcal{N}$, when node i wakes up, D-RK requires $2d_i(n-1)$ communications – each communication i sends/receives a real variable to/from a neighboring node in $\mathcal{N}_i$; hence, at each iteration, i.e., at each time a node wakes up, the expected number of communication per iteration is $N = \sum_{i\in\mathcal{N}} 2p_i d_i(n-1) \leq 2d_{\max}(n-1)$. In particular, for unweighted graphs, i.e., $w_{ij} = 1$ for $(i,j) \in \mathcal{E}$, we have $p_i = \frac{d_i(d_i+1)}{2m+\sum_{j\in\mathcal{N}} d_j^2}$ for $i \in \mathcal{N}$.

Next, instead of (1), consider implementing D-RK on a normalized system $S^{-1/2}\mathcal{L}X = S^{-1/2}B$ to obtain better convergence rate in practice – i-th equation in this normalized system can be computed locally at $i \in \mathcal{N}$. For this system, where all the rows have unit norm, one can set $r_i = r$ for some $r > 0$ for all $i \in \mathcal{N}$ – hence, nodes wake up with uniform probability, i.e., $p_i = \frac{1}{n}$ for $i \in \mathcal{N}$; for this choice of equal clock rates, $H = \frac{1}{n}\mathbf{I}$ and $\{X^k\}_k$ converges linearly to $\mathcal{L}^\dagger$ with rate $\rho_S \triangleq 1 - \frac{1}{n}\lambda^+_{\min}(\mathcal{L}S^{-1}\mathcal{L})$. Moreover, the expected number of communication per iteration is $N = 4m\frac{n-1}{n} \leq 4m$. In all experiments on small world random networks – see the definition in the numerical section, D-RK implemented on the normalized system worked much better than directly implementing it on (1) (see Fig. 1). We conjecture that for certain family of random graphs,

$$\frac{1}{n}\lambda^+_{\min}(\mathcal{L}S^{-1}\mathcal{L}) \geq \left(\frac{\lambda^+_{\min}(\mathcal{L})}{\|\mathcal{L}\|_F}\right)^2 \tag{3}$$

holds with high probability which would directly imply that $\rho_S \leq \rho$, i.e., D-RK on the normalized system would be faster.

3. NUMERICAL EXPERIMENTS

In this chapter, first we provide numerical experiments to show that $\{R_{ij}\}_{(i,j)\in\mathcal{E}}$ can be computed very efficiently in a decentralized fashion, and second, we demonstrate the benefits of using effective resistances in consensus algorithms.

3.1. Decentralized computation of $\mathcal{L}^\dagger$

We tested D-RK and its normalized version on unweighted small-world type communication networks, and we compared these randomized methods with deterministic (cyclic) Kaczmarz method. Given positive integers n, m such that $m \geq n$, let $E \in \mathbb{S}^n$ denote the adjacency matrix of the small-wold network parameterized by (n, m) such that $E_{i,i+1} = 1$ for $i = 1, \ldots, n-1$ and $E_{1,n} = 1$, and the other $m - n$ entries are chosen uniformly at random among the remaining upper diagonal elements of E and set to 1. We considered $n \in \{10, 20\}$ and for each n, we chose m such that the edge density, $2m/(n^2 - n)$, is 0.4 or 0.8. For each scenario, we plot the average of $\log\log(1 + \left\|X^k - \mathcal{L}^\dagger\right\|_F / \left\|\mathcal{L}^\dagger\right\|_F)$ over 100 sample paths versus k. The results show that the randomized algorithms are slower than their deterministic counterpart; this is the price to pay for asynchronous computations. D-RK applied to the normalized system was also faster than the standard D-RK, i.e., numerically we see $\rho_S < \rho$ as suggested by the inequality (3). We also observed finite convergence on every sample path numerically – the finite number of iterations required for convergence depended on the sample path chosen; hence, averaging iterates over sample paths led to the smooth curves reported in Fig. 1.

3.2. Consensus exploiting effective resistances

Let $y^0 \in \mathbb{R}^n$ be a vector such that the i-th component represents the initial value at node i, and let $\bar{y} \triangleq \sum_{i=1}^n y_i^0/n$ be the average. In consensus algorithms, the aim is to compute $\bar{y}$ at each node in a distributed manner. As in Section 2, we assume that each $i \in \mathcal{N}$ has an exponential clock with rate $r_i > 0$; however, now, we assume that when its clock ticks at time t_k, the node i wakes up and picks *one* of its neighbors $j \in \mathcal{N}_i$

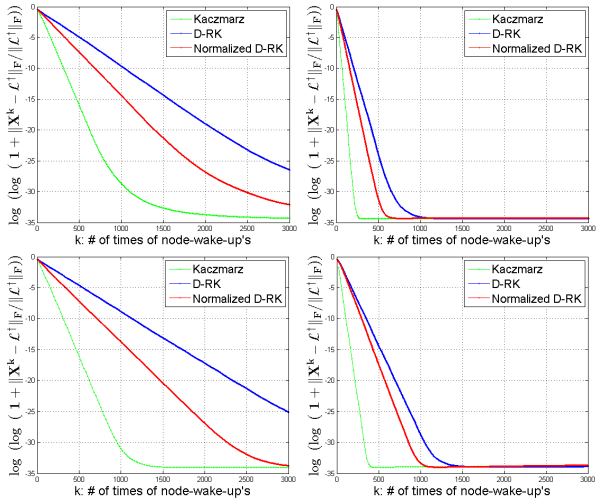

Fig. 1. Performance of D-RK and normalized D-RK on small-world $\mathcal{G}$: **top, left:** $(n, m) = (10, 18)$, **top, right:** $(n, m) = (10, 36)$, **bottom, left:** $(n, m) = (20, 76)$, **top, right:** $(n, m) = (20, 152)$.

with probability $p_{ij} \in (0, 1)$, i.e., $\sum_{j\in\mathcal{N}_i} p_{ij} = 1$. Next, nodes i and j exchange their local variables y_i^k and y_j^k. We assume that each node $i \in \mathcal{N}$ knows $\{R_{ij}\}_{j\in\mathcal{N}}$. We will be comparing two different consensus protocols, where in both protocols nodes operate as in Algorithm 3 but with different $\{p_i\}_{i\in\mathcal{N}}$ and $\{p_{ij}\}_{j\in\mathcal{N}_i}$ for $i \in \mathcal{N}$.

Algorithm 3: Randomized Gossiping

Initialization: $y^0 = [y_1^0, y_2^0, \ldots, y_n^0]^\top \in \mathbb{R}^n$
for $k \geq 0$ **do**
  At time t_k, $i \in \mathcal{N}$ wakes up w.p. p_i
  Picks $j \in \mathcal{N}_i$ randomly w.p. p_{ij}
  $y_i^{k+1} \leftarrow \frac{y_i^k + y_j^k}{2}$, $y_j^{k+1} \leftarrow \frac{y_i^k + y_j^k}{2}$.

Classic Randomized Gossiping: At each iteration k, each edge $(i, j) \in \mathcal{E}$ has equal probability of being activated. If an edge (i, j) is activated at iteration k the nodes take average of their decision variables y_i^k and y_j^k. This algorithm admits an asynchronous implementation – see, e.g., [14]. In our node-wake-up based asynchronous setting, the same behavior can be achieved if each node i wakes up with equal probability $p_i = \frac{1}{n}$, i.e., using uniform clock rates $r_i = r > 0$ for $i \in \mathcal{N}$, and node i picks (i, j) w.p. $p_{ij} = \frac{1}{d_i}$ for all $j \in \mathcal{N}_i$.

Randomized Gossiping with Effective Resistances: This algorithm is similar to classical randomized gossiping, with the only difference that edges are sampled with non-uniform probabilities proportional to effective resistances $\{R_{ij}\}_{(i,j)\in\mathcal{N}}$. In our node-wake-up based asynchronous setting, the same behavior can be achieved if each node i wakes up with probability $p_i = \frac{\sum_{j\in\mathcal{N}_i} R_{ij}}{2\sum_{(i,j)\in\mathcal{E}} R_{ij}}$, i.e., setting clock rate $r_i = \sum_{j\in\mathcal{N}_i} R_{ij}$ for $i \in \mathcal{N}$, and node i picks (i, j) w.p. $p_{ij} = \frac{R_{ij}}{\sum_{j\in\mathcal{N}_i} R_{ij}}$ for all $j \in \mathcal{N}_i$.

We compare the performance of both protocols over an unweighted barbell graph $K_n - K_n$ with $2n$ nodes. Such a graph is illustrated in Fig. 3. In our experiment, we set $n = 20$. Let $\mathcal{N}_R = \{1, \ldots, 20\}$ and $\mathcal{N}_L = \{21, \ldots, 40\}$ represent the node sets in right and left lobes (the subgraph of K_n on the right and left) of the barbell graph. To initialize y^0, we sample y_i^0 from $\mathcal{N}(100, 1)$ for $i \in \mathcal{N}_L$ and y_i^0 from $\mathcal{N}(0, 1)$ for $i \in \mathcal{N}_R$ – this way both lobes have significantly different local means. On the left of Fig. 3, we plot $\log\log(1 + \left\|y^k - \bar{y}\mathbf{1}\right\|_2 / |\bar{y}|)$; and on the right, we plot $\frac{1}{20}\sum_{i\in\mathcal{N}_L} y_i^k$ and $\frac{1}{20}\sum_{i\in\mathcal{N}_R} y_i^k$ vs k for both protocols. The results show that randomized gossiping with effective resistances is much faster.

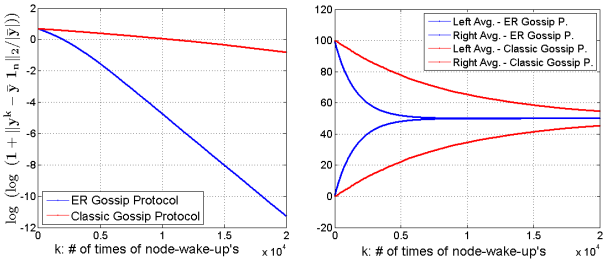

Fig. 2. Performance of classic vs effective resistance based gossiping on barbell $K_{20} - K_{20}$: **left:** Relative error vs k, **right:** Average of left and right lobes vs k for both protocols.

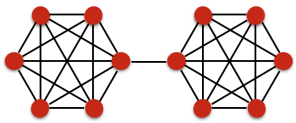

Fig. 3. Barbell graph $K_n - K_n$ with 12 nodes

4. CONCLUSIONS AND FUTURE WORK

In this work, we developed a distributed algorithm for computing effective resistances over an undirected graph $\mathcal{G}$. Our method builds on an efficient, distributed and asynchronous implementation of the Kaczmarz method for solving linear Laplacian systems $\mathcal{L}x = b$. We also presented an application of our algorithm to the consensus problem.

As part of our future work, we will investigate the finite convergence properties of this, suggested by the experiments. We will also study the inequality (3) further which was satisfied for a wide class of random graph models in our tests. Finally, we will investigate the applications of effective resistances to a wide class of distributed optimization algorithms which contain consensus-like iterations including distributed proximal-gradient algorithm (DPGA) and ADMM. In particular, one could design the communication matrix W for the DPGA-W method in [19] using effective resistances by setting $W_{ij} = -R_{ij}$ for $(i, j) \in \mathcal{E}$ and $W_{ii} = -\sum_{j\in\mathcal{N}_i} R_{ij}$. Similarly, it would be interesting to design the communication matrix in ADMM [20] using effective resistances for improving its performance over for optimization problems defined over ill-conditioned graphs.

5. REFERENCES

[1] D. J. Klein, "Resistance-distance sum rules," *Croatica chemica acta*, vol. 75, no. 2, pp. 633–649, 2002.

[2] D. J. Klein and M. Randić, "Resistance distance," *Journal of Mathematical Chemistry*, vol. 12, no. 1, pp. 81–95, 1993.

[3] A. Ghosh, S. Boyd, and A. Saberi, "Minimizing effective resistance of a graph," *SIAM review*, vol. 50, no. 1, pp. 37–66, 2008.

[4] D. Aldous and J. A. Fill, "Reversible markov chains and random walks on graphs," 2014, Unfinished monograph, available at: http://www.stat.berkeley.edu/∼aldous/RWG/book.html.

[5] P. G. Doyle and J. L. Snell, *Random walks and electric networks*, Mathematical Association of America,, 1984.

[6] P. Barooah and J. P. Hespanha, "Graph effective resistance and distributed control: Spectral properties and applications," in *Proceedings of the 45th IEEE Conference on Decision and Control*, Dec 2006, pp. 3479–3485.

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

[8] N. Anari and S. O. Gharan, "Effective-resistance-reducing flows, spectrally thin trees, and asymmetric tsp," in *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, Oct 2015, pp. 20–39.

[9] A. Tizghadam and A. Leon-Garcia, "Betweenness centrality and resistance distance in communication networks," *IEEE Network*, vol. 24, no. 6, pp. 10–16, November 2010.

[10] A. Jadbabaie, "On geographic routing without location information," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*. IEEE, 2004, vol. 5, pp. 4764–4769.

[11] M. Kapralov and R. Panigrahy, "Spectral sparsification via random spanners," in *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*. ACM, 2012, pp. 393–398.

[12] R. B. Bapat, I. Gutmana, and W. Xiao, "A simple method for computing resistance distance," *Zeitschrift für Naturforschung A*, vol. 58, no. 9-10, pp. 494–498, 2003.

[13] W. Shi, Q. Ling, G. Wu, and W. Yin, "Extra: An exact first-order algorithm for decentralized consensus optimization," *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 944–966, 2015.

[14] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE/ACM Transactions on Networking (TON)*, vol. 14, no. SI, pp. 2508–2530, 2006.

[15] Alex Olshevsky, "Linear time average consensus on fixed graphs and implications for decentralized optimization and multi-agent control," *arXiv preprint arXiv:1411.4186*, 2016.

[16] R. M. Gower and P. Richtárik, "Stochastic dual ascent for solving linear systems," *arXiv preprint arXiv:1512.06890*, 2015.

[17] T. Strohmer and R. Vershynin, "A randomized kaczmarz algorithm with exponential convergence," *Journal of Fourier Analysis and Applications*, vol. 15, no. 2, pp. 262–278, 2009.

[18] A. Zouzias and Nikolaos M. Freris, "Randomized extended Kaczmarz for solving least squares," *SIAM Journal on Matrix Analysis and Applications*, vol. 34, no. 2, pp. 773–793, 2013.

[19] N. S. Aybat, Z. Wang, T. Lin, and S. Ma, "Distributed Linearized Alternating Direction Method of Multipliers for Composite Convex Consensus Optimization," *arXiv preprint arXiv:1512.08122, accepted to IEEE Transactions on Automatic Control*, Dec. 2015.

[20] S. Boyd, N. Parikh, Er. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.