Joint 48th IEEE Conference on Decision and Control and
28th Chinese Control Conference
Shanghai, P.R. China, December 16-18, 2009

On Matrix Factorization and Finite-time Average-consensus

Chih-Kai Ko and Xiaojie Gao

***Abstract*— We study the finite-time average-consensus problem for arbitrary connected networks. Viewing this consensus problem as a factorization of $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$ by suitable families of matrices, we prove the existence of a finite factorization and provide tight bounds on the size of the minimal factorization by exhibiting finite-time average-consensus algorithms and bounding their runtimes. We also show that basic matrix theory yields insights into the structure of finite-time consensus algorithms.**

I. Introduction

In a consensus problem, a group of agents (or network nodes) try to reach agreement on a certain quantity of interest that depends on their states [1]. Consensus problems arise in diverse areas such as oscillator synchronization [2], [3], flocking behavior of swarms [4], rendezvous problems [5]–[7], multi-sensor data fusion [8], multi-vehicle formation control [9], satellite alignment [10], [11], distributed computation [12], and many more. When the objective is to agree upon the average, it is an *average-consensus* problem. A motivating example (from [13]) is a network of temperature sensors needing to average their readings to combat fluctuations in ambient temperature and sensor variations.

Many efficient algorithms exist under various settings, *e.g.* [13]–[17]. Although the majority of the proposed algorithms offer rapid convergence, very few offer guaranteed consensus in finite time. In this paper, we study algorithms that achieve average-consensus in finite time for arbitrarily connected networks. We adopt a dual view of finite matrix factorization motivated by the following lemma:

Lemma 1. *Let $W(0), W(1), \cdots, W(T-1) \in \mathbb{R}^{n\times n}$ be a finite sequence of T matrices, then $W(T-1)W(T-2)\cdots W(0) = \frac{1}{n}\mathbf{1}\mathbf{1}^T$ iff*

$$W(T-1)W(T-2)\cdots W(0)\; x(0) = \frac{1}{n}\mathbf{1}\mathbf{1}^T x(0) \qquad (1)$$

for all $x(0) \in \mathbb{R}^n$.

Proof. The "*only if*" direction is clear, so we show the "*if*" direction. Let e_i denote a unit-vector in $\mathbb{R}^n$ with 1 in the i-th coordinate. If we take $x(0) = e_i$, then equation (1) becomes $\frac{1}{n}\mathbf{1} = \prod_{t=0}^{T-1} W(t)\, e_i$. So the i-th column of $\prod_{t=0}^{T-1} W(t)$ must be $\frac{1}{n}\mathbf{1}$ for any i and the lemma follows. □

A. Background and Notation

In this paper, we adopt the following notation: $G = (V, E)$ denotes a connected undirected graph on n vertices with vertex set V and edge set $E \subseteq V \times V$. A *spanning tree* T is a subgraph that is a tree and contains all vertices of G. We assume basic familiarity with elementary graph algorithms and asymptotic notation $O(\cdot), \Omega(\cdot)$ and $\Theta(\cdot)$. Unless otherwise specified, all graphs discussed in this paper are connected and undirected. $\mathbb{R}, \mathbb{Q}$, and $\mathbb{N}$ denote the set of real, rational, and natural numbers, respectively. Boldface $\mathbf{1} \in \mathbb{R}^n$ is a vector of all 1's, $e_i \in \mathbb{R}^n$ is a unit vector with 1 in the i-th coordinate, $I \in \mathbb{R}^{n\times n}$ denotes the identity matrix. Superscript $^\mathsf{T}$ denotes matrix transpose. For a sequence of T matrices $W(t) \in \mathbb{R}^{n\times n}$, the product $W(T-1)W(T-2)\cdots W(0)$ is abbreviated as $\prod_{t=0}^{T-1} W(t)$.

B. Problem Statement

First, we need the notion of admissible matrices.

Definition 2. *Given a graph $G = (V, E)$, we say a matrix $W \in \mathbb{R}^{n\times n}$ is G-*admissible *if*

1) (Conservation) *W is left stochastic: $W_{ij} \geq 0$, $\mathbf{1}^T W = \mathbf{1}^T$; and*
2) (Connectivity) *For $i \neq j$, $W_{ij} = 0$ if $(i, j) \notin E$.*

With slight abuse of notations, we write $W \in G$ if W is G-admissible.

Loosely speaking, a G-admissible matrix performs averaging according to the topolgoy specified by G. In this work, we study the following problems:

- *(Existence)* Given G, does there exist a $T \in \mathbb{N}$ such that $\prod_{t=0}^{T-1} W(t) = \frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$, $W(t) \in G$. In other words, we study the existence of a ***finite*** G-admissible factorization of $n^{-1}\mathbf{1}\mathbf{1}^\mathsf{T}$.
- *(Algorithm)* How can we find a G-admissible factorization, if it exists.
- *(Minimality)* If it exists, what is the minimal such factorization? Denote such minimum by T^*_G.

Definition 3. *Given a set of matrices $\mathcal{S} \subseteq \mathbb{R}^{n\times n}$, define*

$$T^*_{\mathcal{S}} \triangleq \min\left\{ T : \exists W(t) \in \mathcal{S} \text{ with } \prod_{t=0}^{T-1} W(t) = \frac{1}{n}\mathbf{1}\mathbf{1}^T \right\}$$

when it exists. For convenience, when $\mathcal{S}$ is the set of G-admissible matrices, we write T^*_G.

Our problem is equivalent to a finite-time average-consensus problem. Given a graph $G = (V, E)$, imagine vertices V as nodes in a network connected according to E. For each node $i \in V$, let $x_i(t)$ denote the value of node i at time step t. Define $x(t) = [x_0(t), \cdots, x_{n-1}(t)]^\mathsf{T}$. Given any set of initial values $x(0)$, we are interested in a finite sequence of (averaging) operations, $W(t)$, that allows all

Chih-Kai Ko is with the Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125, USA cko@caltech.edu

Dr. Xiaojie Gao is an independent researcher.

nodes to reach average consensus, *i.e.* $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}x(0)$. Expressed as a linear dynamical system, we have

$$x(t+1) = W(t)x(t) \tag{2}$$

with $W(t) \in G$. The G-admissibility requirement limits our averaging operations to those that are consistent with the network topology. Ultimately, we desire a ***finite*** sequence of G-admissible matrices $W(0), \cdots, W(T-1)$ such that $x(T) = \prod_{t=0}^{T-1} W(t)\, x(0) = \frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}x(0)$ for all $x(0) \in \mathbb{R}^n$. Thus, T_G^* is the minimum consensus time.

As we shall see shortly, the set of G-admissible matrices may be too general in the context of network consensus problems. In addition to connectivity and conservation constraints, networked nodes may act synchronously or asynchronously, nodes may be power-constrained, and nodes may have different levels of knowledge/computation. Each of these restrictions further constrains the factors of $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$ to specific subsets of G-admissible matrices.

C. Contributions

Our contributions are as follows:

- We show that any connected graph admits a finite G-admissible factorization of $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$. Furthermore, $\max_G T_G^* = \Theta(n)$.
- When one restricts the factors to come from $S \cap G$, we prove that $\max_G T_{S\cap G}^* = \Theta(n)$. *See Section IV for definition of S.*
- When one further restricts the factors to come from from $S_1 \cap G$, we prove $\max_G T_{S_1\cap G}^* = \Theta(n^2)$. *See Section IV for definition of S_1.*
- We show that matrix theory provides insight on the structure of finite-time average-consensus algorithms.

Since our results are of a "centralized" nature, our results lowerbounds the runtime of any distributed finite-time average-consensus algorithm.

D. Organization

The rest of the paper is organized as follows: In Section II, we review the existing literature. In Section III we show that a G-admissible factorization of $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$ always exists. In Section IV, we study factorization by subsets of G-admissible matrices. In Section V, we discuss algorithm structures dictated by matrix properties. Finally, we close with potential research directions and concluding remarks in Section VI.

II. Related Work

We introduced the finite time average-consensus problem as a matrix factorization problem to emphasize that we require an *exact* average in *finite* time. Much of the existing work analyzes asymptotic properties of $\prod_t W(t)$ as $t \to \infty$, *e.g.* [1], [9], [13]. If we relax our exactness requirement and allow a randomized choice of product matrices, we can define the ϵ-average time of distribution $\mathcal{D}$ [13] as

$$T_{\text{ave}}(\epsilon, \mathcal{D}) \triangleq \sup_{x(0)} \inf \left\{ \mathbf{P}_{\mathcal{D}} \left(\frac{\|x(t) - \frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}\|}{\|x(0)\|} \geq \epsilon \right) \leq \epsilon \right\}$$

where $\epsilon > 0$ and $\mathcal{D}$ is a probability distribution on the set of G-admissible matrices, and $W(t)$ are drawn independently from $\mathcal{D}$. The choice of $\mathcal{D}$ reflects the behavior of different distributed consensus algorithms. For a trivial $\mathcal{D}$, *e.g.* pick a $W \in G$ with $W\mathbf{1} = \mathbf{1}$ and let $W(t) = W$ for all t; the ϵ-average time is governed by the second largest eigenvalue of W, *c.f.* [1], [9]. Optimization of $T_{\text{ave}}(\epsilon, W)$ over W can be written as a semidefinite program (SDP) [18], hence solved efficiently numerically. Tight bounds on $T_{\text{ave}}(\epsilon, \mathcal{D})$ when $\mathcal{D}$ correspond to synchronous and asynchronous distributed gossip algorithms can be found in [13]. For a more detailed overview of convergence behavior of consensus-type problems, we refer the reader to [1], [9], [13] and the references within.

Although exponentially-fast convergence is sufficient in many cases, it is sometimes desirable to achieve convergence in finite time. A number of authors have studied finite-time consensus in the framework of continuous-time systems: Cortés [19] employed nonsmooth gradient flows to design gradient-based coordination algorithms that achieve average-consensus in finite time. Using finite-time semistability theory, Hui et al [20] designed finite time consensus algorithms for a class of thermodynamically motivated dynamic network. Wang and Xiao [21] used finite-time Lyapunov functions to derive finite-time guarantees of specific coordination protocols.

In the discrete-time setting, Sundaram and Hadjicostis [22], [23] studied the finite-time consensus problem for discrete-time systems. By allowing sufficient computation power for the network nodes, [22] showed that nodes in certain linear time-invariant system can compute their averages after a finite number of linear iterations.

Our work is most closely related to [17] where Ko and Shi examined link scheduling on the complete graph to achieve finite-time average-consensus. They provided necessary and sufficient conditions for finite-time consensus and computed the minimum consensus time on the boolean hypercube. By working with a complete graph, one implicitly assumes that any two nodes in the network can communicate. In this paper, we generalize their results to provide communication schedules (*i.e.* consensus algorithms) for any arbitrary connected network.

III. G-admissible Factorization

To prove the existence of a finite G-admissible factorization of $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$, we present Algorithm 1. Starting from its leaves, the algorithm traverses up a spanning tree of G. Along the way, it gives all its value onto its parent and then removes itself. This process terminates when only a single vertex remains. At this point, the remain node contains the sum of all initial node values. The remainder of the algorithm traverses back down the tree while re-distributing the values to achieve average-consensus at termination.

To translate Algorithm 1 into a G-admissible factorization. Notice that line 6 corresponds to a G-admissible matrix W

Algorithm 1: GATHER-PROPAGATE

Input: Graph G, initial values x
Output: $x \leftarrow \frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}x$

$d \leftarrow$ vector of 1's indexed by $V(G)$
$T \leftarrow$ a spanning tree of G
while T *is not a single vertex* **do**
  Pick a leaf $v \in V(T)$
  Let $e = (u, v)$ be the edge attaching v to T
  $\begin{bmatrix} x_u \\ x_v \end{bmatrix} \leftarrow \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_u \\ x_v \end{bmatrix}$
  $d_u \leftarrow d_u + d_v$
  $T \leftarrow (V \backslash \{v\}, E \backslash \{e\})$
end
Let $u \leftarrow$ remaining vertex of T
PROPAGATE(G, u, d) `// See Algorithm 2`

Algorithm 2: PROPAGATE

Input: G, u, d
Output: $x \leftarrow \frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}x$

foreach *neighbor* v *of* u **do**
  $\begin{bmatrix} x_u \\ x_v \end{bmatrix} \leftarrow \frac{1}{d_u} \begin{bmatrix} d_u - d_v \\ d_v \end{bmatrix} x_u$
  $E \leftarrow E \backslash \{(u, v)\}$
  PROPAGATE(G, v, d)
  $d_u = d_u - d_v$
end

with

$$W_{ij} = \begin{cases} 1 & \text{if } i = j \neq v, \\ 1 & \text{if } i = u \text{ and } j = v, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Similarly, line 2 of Algorithm 2 corresponds to a G-admissible matrix W with

$$W_{ij} = \begin{cases} (d_u - d_v)/d_u & \text{if } i = j = u, \\ d_v/d_u & \text{if } i = v \text{ and } j = u, \\ 1 & \text{if } i = j \neq u, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

It is straightforward to construct a finite factorization of $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$using $2(n-1)$ G-admissible matrices: $n-1$ matrices of type (3) followed by $n-1$ matrices of type (4). Summarizing into a theorem:

Theorem 4. *For any connected graph $G = (V, E)$ on n vertices, there exists a finite G-admissible factorization of $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$. Furthermore, $T^*_G \leq 2(n-1)$ and Algorithm 1 exhibits such a factorization.*

To see that our upperbound is tight, we consider a path on n vertices: let $G = (V, E)$ with $V = \{0, \cdots, n-1\}$ and $(i, j) \in E$ iff $j = i + 1$. Fix the initial values $x(0)$ as

$$x_i(0) = \begin{cases} 1 & \text{if } i = 0, \\ 0 & \text{otherwise.} \end{cases}$$

Since all of the mass is contained in node-0, we require at least n averaging operations to distribute mass to node-$(n-1)$. Each operation (*i.e.* multiplication by a G-admissible matrix) can only propagate information by one additional node down the path. Thus,

Theorem 5. $\max_G T^*_G = \Theta(n)$

IV. FACTORIZATION UNDER ADDITIONAL CONSTRAINTS

In terms of network consensus, allowing factorization by G-admissible matrices may be too strong of a requirement. Before proceeding, we need some additional notation. Given a graph G on n vertices and a set $\mathcal{S} \subseteq \mathbb{R}^{n\times n}$ of matrices, we write $\mathcal{S} \cap G$ to mean $\{W \in \mathbb{R}^{n\times n} : W \in \mathcal{S} \text{ and } W \in G\}$.

Various existing consensus algorithms correspond to factoring $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$ using $\mathcal{S} \cap G$ with differing $\mathcal{S}$. For example, it may be desirable to factor $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$ using $W(t) \in \{W : W\mathbf{1} = \mathbf{1} \text{ and } W \in G\}$ so the average is a fix point of iteration (2).

Gossip-based asynchronous algorithms, *c.f.* [13], correspond to factorization using $W(t) \in S'_1 \cap G$ where

$$S'_1 \triangleq \left\{ I - \frac{(e_i - e_j)(e_i - e_j)^\mathsf{T}}{2} : 0 \leq i, j < n \right\}.$$

Each matrix in $S'_1 \cap G$ corresponds to the averaging of two neighboring node values. Boyd et al [13] studies the ϵ-average time of (2) when the $W(t)$'s are drawn independently and uniformly at random from $S'_1 \cap G$. In terms of finite-time consensus, [17] showed that

Theorem 6. *Given a connected graph G on n vertices, finite factorization of $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$with $S'_1 \cap G$ is possible only if $n = 2^m$ for some $m \in \mathbb{N}$. Furthermore, $T^*_{S'_1 \cap G} \geq m\,2^{m-1}$ and equality is achieved when G is the boolean m-hypercube.*

Proof. See Algorithm 1 of [17] for hypercube achievability, Theorem 2 of [17] for the lowerbound, and Theorem 8 of [17] for the necessary condition. □

Thus, for arbitrary G, the set $S'_1 \cap G$ is too restricting and we must look beyond $S'_1 \cap G$ if we desire a finite factorizations of $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$.

A. Pair-wise Weighted Averages

Consider the following generalization of S'_1:

$$S_1 \triangleq \left\{ I - \frac{(e_i - e_j)(e_i - e_j)^\mathsf{T}}{m} : 1 \leq m \in \mathbb{Q}; 0 \leq i, j < n \right\}$$

The set S_1 allows pair-wise weighted-averages and notice that $S'_1 \subset S_1$. Using Algorithm 3, we show that finite-time average-consensus is possible using only pair-wise weighted averages at each step. That is, $T^*_{S_1 \cap G} < \infty$.

Let us examine Algorithm 3. Let $x \in \mathbb{R}^n$ denote a vector of node values. Starting with a leaf node v (*i.e.* v has degree 1) of T, the algorithm performs a depth-first search (see Algorithm 4) that terminates with x_v achieving the average. Once v reaches the average, it can be removed from future consideration. The algorithm repeats this procedure leaf-by-leaf until all vertices have been examined. At which time, all nodes have reached average-consensus.

Algorithm 3: CONSENSUS

Input: Graph G, initial values x
Output: $x \leftarrow \frac{1}{n}\mathbf{1}\mathbf{1}^{\mathsf{T}}x$

1 $T \leftarrow$ a spanning tree of G
2 $d \leftarrow$ vector indexed by $V(T)$
3 **while** T *is not a single vertex* **do**
4 Initialize d to all 1's
5 Pick a node $v \in V(T)$ such that degree(v)=1
6 Designate v as the root of T
7 DFS(T, v, x, d) `// See Algorithm 4`
8 $T \leftarrow T \backslash \{v\}$
9 **end**

Algorithm 4: DFS

Input: Tree T, vertex v,vectors x, d
Output: $x_v \leftarrow |T|^{-1} \sum_{u \in T} x_u$

1 **if** v *has no children* **then**
2 **return**
3 **else**
4 **foreach** *child* u *of* v **do**
5 DFS(T, u, x, d)
6 $\begin{bmatrix} x_v \\ x_u \end{bmatrix} \leftarrow \begin{bmatrix} \frac{d_v}{d_v+d_u} & \frac{d_u}{d_v+d_u} \\ \frac{d_u}{d_v+d_u} & \frac{d_v}{d_v+d_u} \end{bmatrix} \begin{bmatrix} x_v \\ x_u \end{bmatrix}$
7 $d_v \leftarrow d_v + d_u$
8 **end**
9 **end**

It is straight forward to construct a sequence of G-admissible matrices from Line 6 of Algorithm 4. It's runtime is $O(n^2)$ since depth-first search executes in time $O(n)$ (see §22.3 of [24]) and we perform $n-1$ of them. Thus,

Lemma 7 (Upperbound)**.** *For any connected graph G with n vertices, $T^*_{S_1 \cap G} = O(n^2)$.*

As it turns out, we can't do better than $O(n^2)$ on the path:

Lemma 8 (Lowerbound)**.** *When $G = (V, E)$ is the path on n vertices,* i.e. *$V = \{0, 1, \cdots, n-1\}$ and $E = \{(i, i+1) : 0 \le i < n\}$, $T^*_{S_1 \cap G} = \Omega(n^2)$*

Proof. Consider the initial condition $x(0)$: $x_i(0) = 0$ if $i < n/2$ and $x_i(0) = 1$ if $i \ge n/2$. The total mass is $\mathbf{1}^{\mathsf{T}}x(0) = n/2$ and average-consensus is achieved when $x(T) = \frac{1}{2}\mathbf{1}$ for some T. In this proof, it is useful to view each matrix in $S_1 \cap G$ as a "use" of a particular edge in E. Using a mass-balancing flow argument, we show the need for $\Omega(n^2)$ edge uses.

For $0 < i < n/2$, there are $i+1$ nodes to the left of edge $(i, i+1)$, these nodes require $(i+1)/n$ fraction of the total mass. Since the total mass is $n/2$ and each use of an edge carries at most 1 unit of mass, we know that edge $(i, i+1)$ must be used at least $\lceil (i+1)/2 \rceil$ times. Summing over these edges yield

$$T^*_{S_1 \cap G} \ge \sum_{i=1}^{n/2-1} \left\lceil \frac{i+1}{2} \right\rceil \ge \frac{1}{2} \sum_{i=1}^{n/2-1} (i+1) = \Omega(n^2)$$

Algorithm 5: DFS

Input: Tree T, vertex v,vectors x, d
Output: $x_v \leftarrow |T|^{-1} \sum_{u \in T} x_u$

1 **if** v *has no children* **then**
2 **return**
3 **else**
4 **foreach** *child* u *of* v **do**
5 DFS(T, u, x, d)
6 **end**
7 $\{u_1, \cdots, u_l\} \leftarrow$ set of children of v
8 $D \leftarrow d_v + \sum_{j=1}^{l} d_{u_j}$
9 $\begin{bmatrix} x_v \\ x_{u_1} \\ \vdots \\ x_{u_l} \end{bmatrix} \leftarrow \begin{bmatrix} \frac{d_v}{D} & \frac{d_{u_1}}{D} & \cdots & \frac{d_{u_l}}{D} \\ \frac{d_{u_1}}{D} & \frac{D-d_{u_1}}{D} & & 0 \\ \vdots & & \ddots & \\ \frac{d_{u_l}}{D} & 0 & & \frac{D-d_{u_l}}{D} \end{bmatrix} \begin{bmatrix} x_v \\ x_{u_1} \\ \vdots \\ x_{u_l} \end{bmatrix}$
10 $d_v \leftarrow D$
11 **end**

To see that each edge can carry a flow of at most 1, observe that matrices in S_1 correspond to convex combinations of pairs of node values. Since initial values are $x_i(0) \in \{0, 1\}$, any sequence of convex combinations must keep the values in the closed interval $[0, 1]$, *i.e.* $0 \le x_i(t) \le 1$ for all t. □

Combining Lemmas 7 and 8, we have

Theorem 9. $\max_G T^*_{S_1 \cap G} = \Theta(n^2)$

B. Symmetric Weighted Averages

Let us now restrict ourselves even less and consider $S \triangleq \left\{W \in \mathbb{Q}^{n \times n} : W = W^{\mathsf{T}}\right\}$. Note that the matrices in S are doubly stochastic (*i.e.* $\mathbf{1}^{\mathsf{T}}W = \mathbf{1}^{\mathsf{T}}$ and $W\mathbf{1} = \mathbf{1}$). The motivation for S is to allow distribution of mass by symmetric weighted averages; yet disallow drastic aggregation steps such as line 6 of Algorithm 1. Such operations are often impossible under typical network node assumptions (*i.e.* topology awareness, computational limitations, distributed behavior... etc.). We will see that by using $S \cap G$ instead of $S_1 \cap G$, the consensus time is improved from $\Theta(n^2)$ to $\Theta(n)$.

As $S_1 \subset S$, we can still use Algorithm 3 to achieve consensus. Instead using the depth-first search in Algorithm 4, we modify it slightly (see Algorithm 5) to use fewer matrices.

Intuitively speaking, Algorithm 6 implements a pipelined version of Algorithm 3. We employ parallel consensus steps when they do not interfere with each other. For clarity, we use a simple example to illustrate the essence of the pipelined algorithm. Consider G as the path with $V = \{1, 2, 3, 4, 5\}$ and $E = \{a, b, c, d\}$ as shown in Figure 1. Suppose that line 5 of Algorithm 3 examines nodes $5, 4, 3, 2, 1$ in that order, then the sequence of pair-wise weighted averages corresponds to

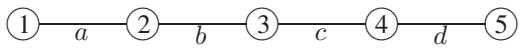

Fig. 1. A line graph.

the following sequence of edges:

	a	b	c	d						⑤
					a	b	c			④
								a	b	③
									a	①&②
Time:	1	2	3	4	5	6	7	8	9	10

Here, time runs left-to-right and each time column enumerates all edges used during that time slot. With Algorithm 3, each time slot only utilizes one edge. The right-most annotation indicates that a sequence of edges allowed a node to obtain the correct average. For example, after the edge sequence a, b, c, d in time steps 1-4, node ⑤ will have the correct global average. Since edges a and c are vertex-disjoint, the averaging on a will not effect the values of nodes in edge c. We can thus perform some averages in parallel and implement a pipelined fashion:

	a	b	c	d				⑤
			a	b	c			④
					a	b		③
							a	①&②
Time:	1	2	3	4	5	6	7	

Pipelining allow us to use multiple edges per time step (*e.g.* At time 4, edges b and d are used in parallel. At time 5, edges c and a are used in parallel). Once again, the right-most labels annotate the epochs dedicated to each node obtaining the global average (*e.g.* after edge sequence a, b, c in time 3-5, node ④ obtains the global average). The key innovation of Algorithm 6 is that it takes advantage of edges that can be used simultaneously.

Now we examine the inner workings of Algorithm 6. We begin by establishing a postordering $(r_1, r_2, \cdots, r_n)$ of vertices by a depth-first search from an arbitrary vertex. During the execution process, we keep track of ϕ, an indicator of whether a vertex's value is the average of its descendants:

$$\phi_v = \begin{cases} 1 & \text{if } v = \frac{1}{\deg(v)} \sum_{u \in \text{decedents(v)}} x_u \\ 0 & \text{otherwise} \end{cases}$$

The computations of x updates in each *while* loop (line 13) can be translated into a single matrix in $S \cap G$ as each vertex appears at most once in line 20 for each iteration of the loop. The number of *while* loops needed for r_1 to achieve the average is at most $n-1$. After r_i reaches the average, the number of *while* loops needed for r_{i+1} to achieve the average is the length of the path from r_i to r_{i+1}. Since the sequence $r_1, r_2, \cdots, r_n$ is a postordering of $V(T)$ by depth-first search, $\sum_{i=1}^{n-1}$ (length of path from r_i to r_{i+1}) $\leq 2n$. Therefore, the number of matrices in this factorization is $O(n)$.

Theorem 10 (Upperbound)**.** *For any connected graph G with n vertices, $T^*_{S \cap G} = O(n)$.*

Algorithm 6: CONSENSUS

Input: Graph G, initial values x
Output: $x \leftarrow \frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}x$

1 $T \leftarrow$ a spanning tree of G
2 $r_1, r_2, \cdots, r_n \leftarrow$ a postordering of $V(T)$ by depth-first search
3 $d \leftarrow$ vector of 1's indexed by $V(T)$
4 $\phi \leftarrow$ vector of 0's indexed by $V(T)$
5 **foreach** $v \in V(T)$ **do**
6 **if** *degree(v)=1* **then** $\phi_v = 1$
7 **end**
8 $i \leftarrow 1$
9 Let r_i be the root of T
10 $\phi_{r_i} = 0$
11 **while** $|V(T)| > 1$ **do**
12 $d' \leftarrow d$
13 $\phi' \leftarrow \phi$
14 **foreach** $v \in V(T)$ **do**
15 **if** $\phi_v = 0$ *and* $\forall$ *child* u *of* $v, \phi_u = 1$ **then**
16 $\{u_1, \cdots, u_l\} \leftarrow$ set of children of v
17 $D \leftarrow d_v + \sum_{j=1}^{l} d_{u_j}$

$$\begin{bmatrix} x_v \\ x_{u_1} \\ \vdots \\ x_{u_l} \end{bmatrix} \leftarrow \begin{bmatrix} \frac{d_v}{D} & \frac{d_{u_1}}{D} & \cdots & \frac{d_{u_l}}{D} \\ \frac{d_{u_1}}{D} & \frac{D-d_{u_1}}{D} & & 0 \\ \vdots & & \ddots & \\ \frac{d_{u_l}}{D} & 0 & & \frac{D-d_{u_l}}{D} \end{bmatrix} \begin{bmatrix} x_v \\ x_{u_1} \\ \vdots \\ x_{u_l} \end{bmatrix}$$

18 $d'_v \leftarrow D$
19 $\phi'_v \leftarrow 1$
20 **foreach** *child* u *of* v **do**
21 **if** $d_u \neq 1$ **then**
22 $d'_u \leftarrow 1$
23 $\phi'_u \leftarrow 0$
24 **end**
25 **end**
26 **end**
27 **end**
28 $d \leftarrow d'$
29 $\phi \leftarrow \phi'$
30 **if** $\phi_{r_i} = 1$ **then**
31 let $r_i \sim u_1 \sim \cdots \sim u_m \sim r_{i+1}$ be the path from r_i to r_{i+1} in T
32 **foreach** $0 < j \leq m$ **do**
33 $d_{u_j} \leftarrow 1$
34 $\phi_{u_j} \leftarrow 0$
35 **end**
36 **if** *degree(u_1)=1* **then**
37 $\phi_{u_1} \leftarrow 1$
38 **end**
39 $T \leftarrow T \backslash \{r_i\}$
40 $i \leftarrow i + 1$
41 Let r_i be the root of T
42 $\phi_{r_i} = 0$
43 **end**
44 **end**

Since $\Omega(n)$ is a lower bound on T_G^* for any G, we have:

Corollary 11. $\max_G T^*_{S\cap G} = \Theta(n)$.

V. Matrix Insights

Many of our finite factorization results have been derived constructively from algorithms. We now examine what basic matrix theory can tell us about the algorithmic structure. Let us consider factorization of $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$ with $W(t) \in S_1 \cap G$. Except for matrices in $S_1' \cap S_1$, all of the matrices in S_1 are non-singular. Thus, for $\det \prod_{t=0}^{T-1} W(t) = \det \frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$, we must have $W(t) \in S_1'$ for at least one t. In fact,

Theorem 12. *If a finite sequence of T matrices $W(0), \cdots, W(T-1)$ satisfies $\prod_{t=0}^{T-1} W(t) = \frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$ with $W(t) \in S_1 \cap G$; then, there exists a sequence of $n-1$ indices $\mathcal{I} = \{t_1, t_2, \cdots, t_{n-1}\}$ such that $W(t_i) \in S_1' \cap G$ for all $i \in \mathcal{I}$.*

Proof. First note that $\operatorname{rank} \mathrm{A} = \mathrm{n} - 1$ for $A \in S_1'$, $\operatorname{rank} \mathrm{B} = \mathrm{n}$ for $B \in S_1 \backslash S_1'$, and $\operatorname{rank} \mathbf{1}\mathbf{1}^\mathsf{T} = 1$. Since multiplication by a rank-$(n-1)$ matrix can decrease the rank of a matrix by at most one, we need $n-1$ such matrices to reach a rank of one. □

VI. Discussion and Extensions

All of the algorithms given thus far have assumed access to a spanning tree of G. In terms of finding a finite factorization of $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$, this is not a problem: we can compute a spanning tree in polynomial time using depth-first search. In terms of distributed consensus algorithms, this assumption corresponds to nodes knowing the network topology and behaving synchronously. Or, alternatively, assumed existence of a centralized scheduler (who knows the topology) that schedules the interactions in a consensus protocol.

We close with some possible future research directions: (1) In our analysis, we have assumed that network nodes are homogeneous and capable of performing weighted average operations. If nodes are inhomogeneous (*e.g.* a network of mobile phones and base stations) then their ability to compute weighted averages may differ. It is interesting to consider the implications of inhomogenous networks on finite-time average-consensus. (2) If nodes communicate wirelessly using directional antennas, then their topology is represented by a directed graph. Hence, the analysis of G-admissible factorizations of $\frac{1}{n}\mathbf{1}\mathbf{1}^\mathsf{T}$when G is a directed graph is a natural extension.

VII. Acknowledgment

The authors would like to thank Leonard Schulman, Lijun Chen, and the anonymous reviewers for helpful discussions and/or comments.

References

[1] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and Cooperation in Networked Multi-Agent Systems," *Proceedings of the IEEE, Special Issue on Networked Control Systems*, vol. 95, no. 1, pp. 215–233, 2007.

[2] V. Preciado and G. Verghese, "Synchronization in generalized erdos-renyi networks of nonlinear oscillators," *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, pp. 4628–4633, Dec. 2005.

[3] A. Papachristodoulou and A. Jadbabaie, "Synchronization in oscillator networks: Switching topologies and non-homogeneous delays," *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, pp. 5692–5697, Dec. 2005.

[4] W. Xi, X. Tan, and J. S. Baras, "A Stochastic Algorithm for Self-Organization of Autonomous Swarms," in *IEEE Conference on Decision and Control*, 2005.

[5] H. Ando, Y. Oasa, I. Suzuki, and M. Yamashita, "Distributed memoryless point convergence algorithm for mobile robots with limited visibility," *Robotics and Automation, IEEE Transactions on*, vol. 15, pp. 818–828, Oct 1999.

[6] A. S. M. J. Lin and B. D. O. Anderson, "The multi-agent rendezvous problem," *42nd IEEE Conference on Decision and Control, 2003*, p. 1508, Dec. 2003.

[7] S. Martínez, J. Cortés, and F. Bullo, "Motion coordination with distributed information," *IEEE Control Systems Magazine*, vol. 27, no. 4, pp. 75–88, 2007.

[8] D. P. Spanos, R. Olfati-Saber, and R. M. Murray, "Distributed Sensor Fusion Using Dynamic Consensus," in *IFAC World Congress*, 2005.

[9] J. A. Fax and R. M. Murray, "Information Flow and Cooperative Control of Vehicle Formations," *IEEE Trans. on Automatic Control*, vol. 49, pp. 1465–1476, September 2004.

[10] T. Arai, E. Pagello, and L. E. Parker, "Guest Editorial Advances in Multirobot Systems," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 655 – 661, 2002.

[11] P. Ogren, E. Fiorelli, and N. E. Leonard, "Cooperative Control of Mobile Sensor Networks: Adaptive Gradient Climbing in a Distributed Environment," *IEEE Trans. on Automatic Control*, vol. 49, pp. 1292–1302, August 2004.

[12] N. Lynch, *Distributed Algorithms*. San Mateo, CA: Morgan Kaufmann Publishers, 1996.

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

[14] A. D. D. Kempe and J. Gehrke, "Gossip-based computation of aggregate information," in *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Sciend (FOCS 03)*, vol. 8, p. 482, 2003.

[15] J.-Y. Chen, G. Pandurangan, and D. Xu, "Robust computation of aggregates in wireless sensor networks: distributed randomized algorithms and analysis," *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pp. 348–355, April 2005.

[16] R. Carli, F. Fagnani, A. Speranzon, and S. Zampieri, "Communication constraints in coordinated consensus problems," American Control Conference, June 2006.

[17] C.-K. Ko and L. Shi, "Scheduling for finite time consensus," in *Proceedings of the American Control Conference*, June 2009.

[18] L. Xiao and S. Boyd, "Fast Linear Iterations for Distributed Averaging," *Systems and Control Letters*, vol. 53, pp. 65–78, September 2004.

[19] J. Cortes, "Finite-time convergent gradient flows with applications to network consensus," *Automatica*, vol. 42, no. 11, pp. 1993 – 2000, 2006.

[20] Q. Hui, W. Haddad, and S. Bhat, "Finite-time semistability theory with applications to consensus protocols in dynamical networks," in *Proceedings of the American Control Conference*, July 2007.

[21] L. Wang and F. Xiao, "Finite-time consensus problems for networks of dynamic agents," *http://arxiv.org/pdf/math/0701724*.

[22] S. Sundaram and C. N. Hadjicostis., "Finite-Time Distributed Consensus in Graphs with Time-Invariant Topologies," in *The 26th American Control Conference, New York, NY*, 2007.

[23] S. Sundaram and C. N. Hadjicostis, "Distributed consensus and linear functional calculation in networks: an observability perspective," in *The 6th International Conference on Information Processing in Sensor Networks*, 2007.

[24] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. The MIT Press, 2nd ed., 2001.