Systems & Control Letters 95 (2016) 46–52

Contents lists available at ScienceDirect

Systems & Control Letters

journal homepage: www.elsevier.com/locate/sysconle

Reprint of "A distributed algorithm for efficiently solving linear equations and its applications (Special Issue JCW)"[☆,☆☆]

S. Mou[a,*], Z. Lin[b], L. Wang[c], D. Fullmer[c], A.S. Morse[c]

[a] School of Aeronautics and Astronautics, Purdue University, United States
[b] College of Electrical Engineering, Zhejiang University, China
[c] Department of Electrical Engineering, Yale University, United States

ARTICLE INFO

Article history:
Available online 12 March 2016

Keywords:
Distributed algorithms
Least-square solution
Network localization

ABSTRACT

A distributed algorithm is proposed for solving a linear algebraic equation $Ax = b$ over a multi-agent network, where $A \in \mathbb{R}^{\bar{n}\times n}$ and the equation has a unique solution $x^* \in \mathbb{R}^n$. Each agent knows only a subset of the rows of $\begin{bmatrix} A & b \end{bmatrix}$, controls a state vector $x_i(t)$ of size smaller than n and is able to receive information from its nearby neighbors. Neighbor relations are characterized by time-dependent directed graphs. It is shown that for a large class of time-varying networks, the proposed algorithm enables each agent to recursively update its own state by only using its neighbors' states such that all $x_i(t)$ converge exponentially fast to a specific part of x^* of interest to agent i. Applications of the proposed algorithm include solving the least square solution problem and the network localization problem.

1. Motivations

A natural idea for solving large systems of linear algebraic equations is to decompose them into smaller ones which can then be solved by a multi-agent network. Since autonomous agents in a network are usually physically separated from each other, each agent can only communicate with its nearby neighbors. There are typically communication constraints on the information flow across a multi-agent network. This consequently precludes centralized processing and gives rise to distributed algorithms for solving linear equations $Ax = b$ simultaneously by m agents [1–4], in which each agent i knows $\begin{bmatrix} A_i & b_i \end{bmatrix}$ of the partitioned matrix

$$\begin{bmatrix} A & b \end{bmatrix} = \begin{bmatrix} A_1 & b_1 \\ A_2 & b_2 \\ \vdots & \vdots \\ A_m & b_m \end{bmatrix}, \quad A \in \mathbb{R}^{\bar{n}\times n} \tag{1}$$

current estimates of the equation's solution x^* generated by its neighbors, and nothing more. Rather than go through the intermediate step of reformulating the problem of solving linear equations, an optimization problem [5–9] or a constrained consensus problem [10], the algorithm in [3] has tackled the problem directly based on a so-called "agreement principle", which, on the one hand limits each agent's state update to satisfy its own private equation and on the other causes all agents' states to reach a consensus. Implementing these algorithms requires each agent to transmit at each time step a vector of dimension at least n to each of its current neighbors. In certain applications, to be described in this paper, it is not necessary for each agent to determine all of the entries of x^*, especially when the matrix A have zero columns. There is the possibility that each agent may be able to determine its own favorite part of the entries of x^* by transmitting estimates of dimensions smaller than n. The aim of this paper is to present such an algorithm for solving the linear equation which requires less communication between agents than the algorithm in [3] by exploiting the special structure of A.

DOI of original article: http://dx.doi.org/10.1016/j.sysconle.2016.02.010.

☆ The research of A.S. Morse, L. Wang and D. Fullmer is supported by the US Air Force Office of Scientific Research and the National Science Foundation. The research of Z. Lin is supported by the National Natural Science Foundation of China under Grant 61273113.

☆☆ This article is a reprint of a previously published article. For citation purposes, please use the original publication details [Systems and Control Letters, 91 (2016), pp. 21–27].

* Corresponding author.
E-mail addresses: mous@purdue.edu (S. Mou), linz@zju.edu.cn (Z. Lin), lili.wang@yale.edu (L. Wang), daniel.fullmer@yale.edu (D. Fullmer), as.morse@yale.edu (A.S. Morse).

http://dx.doi.org/10.1016/j.sysconle.2016.05.011
0167-6911/

The rest of the paper is organized as follows: We formulate the problem of interest in Section 2 and present an algorithm to solve it in Section 3. Analysis to prove the effectiveness of the proposed algorithm is provided in Section 4. Two application examples are discussed in Section 5 and we conclude in Section 6. Proof for all propositions and lemmas is given in the Appendix.

Notation. Let $\mathbf{m} = \{1, 2, \ldots, m\}$. Let $\mathbf{1}_m$ denote the $m \times 1$ vector with all elements equal to 1. Let I_m denote the $m \times m$ identity matrix. Let I denote the identity matrix of a proper size. Let M' and $|M|$ denote the transpose and the 2-norm of the matrix M, respectively. Throughout the paper, we let col $\{\cdots\}$ denote a column stack of all elements in it and let col $\{A_i,\ i \in \mathcal{R}\}$, where $\mathcal{R}$ is a set of positive integers, denote a stack of A_i with the index in a top-down ascending order. For two sets $\mathcal{R}_1$ and $\mathcal{R}_2$, we let $\mathcal{R}_1 \cap \mathcal{R}_2$ and $\mathcal{R}_1 \cup \mathcal{R}_2$ denote their intersection and union, respectively. Let $\mathcal{R}_1/\mathcal{R}_2$ denote the set by deleting all the elements belonging to $\mathcal{R}_2$ from $\mathcal{R}_1$. Let diag $\{A_1, A_2, \ldots, A_m\}$ represent a block diagonal matrix with A_k as its kth diagonal block. Let $\mathcal{G}_{sa}$ denote the set of all directed graphs with m vertices which have self-arcs at all vertices. Let $\gamma(M)$ denote the graph of a square matrix $M \in \mathbb{R}^{m\times m}$ with nonnegative entries, which is an m vertex directed graph defined so that (j, i) is an arc from j to i in the graph just in case the ijth entry of M is non-zero.

2. Problem formulation

Given a linear equation $Ax = b$, where $A \in \mathbb{R}^{\bar{n}\times n}$ and $b \in \mathbb{R}^{\bar{n}}$ can be partitioned as in (1), we suppose each A_i admits a block partition of the form $A_i = \begin{bmatrix} A_{i1} & A_{i2} & \cdots & A_{im} \end{bmatrix}$, $i \in \mathbf{m}$, where some of the $A_{ij} \in \mathbb{R}^{\bar{n}_i\times n_j}$ may be zero matrices. To represent the pattern of non-zero blocks in A, we let $\mathcal{R}_i$ and $\mathcal{C}_i$ denote the set of positive integers such that $A_{ij} \neq 0$ for each $j \in \mathcal{R}_i$ and $A_{ji} \neq 0$ for each $j \in \mathcal{C}_i$, respectively, $i \in \mathbf{m}$. For example, if

$$A = \begin{bmatrix} A_{11} & A_{12} & 0 \\ A_{21} & 0 & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}, \tag{2}$$

one has $\mathcal{R}_1 = \{1, 2\}$ and $\mathcal{C}_1 = \{1, 2, 3\}$. Without loss of generality, we assume that A is partitioned such that there are no zero block rows or columns. That is, $\mathcal{R}_i \neq \emptyset$ and $\mathcal{C}_i \neq \emptyset$ for all $i \in \mathbf{m}$. The existence of zero blocks in A_i leads to the following factorization

$$A_i = \bar{A}_i R_i. \tag{3}$$

Here $\bar{A}_i \in \mathbb{R}^{\bar{n}_i\times m_i}$ with $m_i = \sum_{k\in\mathcal{R}_i} n_k$ is resulted from deleting zero blocks from A_i,

$$R_i = \text{col}\,\{E_k, k \in \mathcal{R}_i\} \in \mathbb{R}^{m_i\times n}$$

with

$$E_k = \begin{bmatrix} 0_{n_k\times \sum_{i}^{k-1} n_i} & I_{n_k\times n_k} & 0_{n_k\times \sum_{i=k+1}^{m} n_i} \end{bmatrix} \in \mathbb{R}^{n_k\times n}, \quad k \in \mathbf{m}.$$

Consider a network of m agents that are able to receive information from their "neighbors". Here by a neighbor of agent i is meant any agent within agent i's reception range. We write $\mathcal{N}_i(t)$ for the labels of agent i's neighbors at time t and we always take agent i as a neighbor of itself, that is, $i \in \mathcal{N}_i(t)$. Neighbor relations can be conveniently characterized by a directed graph $\mathbb{N}(t)$ with m vertices and a set of arcs defined so that there is an arc in $\mathbb{N}(t)$ from vertex j to i just in case that agent j is a neighbor of agent i.

Let x^* denote the unique solution to $Ax = b$. Corresponding to the pattern of zero-blocks in A, we partition $x^* =$ col $\{x_1^*, x_2^*, \ldots, x_m^*\}$ with $x_i^* \in \mathbb{R}^{n_i}$. Let $y_i = R_i x^*$, that is,

$$y_i = \text{col}\,\{x_k^*, k \in \mathcal{R}_i\}, \quad i \in \mathbf{m}. \tag{4}$$

Then y_i is a part of x^*, which may become agent i's specific interest in certain situations. This leads us the following problem:

Problem 1. Suppose each agent i controls a state vector $x_i(t) \in \mathbb{R}^{m_i}$ to satisfy its own private function $\bar{A}_i x_i(t) = b_i$. Devise a local rule for each agent to update its own state only using state vectors from its neighbors such that all $x_i(t)$ converge exponentially fast to the constant vector y_i.

3. The algorithm

In this section, we will present a distributed algorithm to solve Problem 1.

Suppose time is discrete in that t takes values in $\{1, 2, \ldots\}$ and all agents operate synchronously. At $t = 1$, each agent i picks $x_i(1)$ to satisfy $\bar{A}_i x_i(1) = b_i$. For $t \geq 1$, we restrict the updating of $x_i(t)$ to iterations of the form

$$x_i(t+1) = x_i(t) + K_i u_i(t)$$

where the columns of K_i form a basis for the kernel of $\bar{A}_i$. Then no matter what $u_i(t)$ is, each $x_i(t)$ will obviously satisfy $\bar{A}_i x_i(t) = b_i$. If additionally there exists a constant vector x^* such that

$$x_i(t) = R_i x^*, \quad i \in \mathbf{m}, \tag{5}$$

one by $A_i = \bar{A}_i R_i$ has $A_i x^* = b_i$ for all $i \in \mathbf{m}$. Then $Ax^* = b$.

From the definitions of E_k and R_i, one notes that the kth substate of $R_i' x_i(t)$ is $E_k R_i' x_i(t)$. To show that there exists a constant vector x^* satisfying (5), one only needs to show that there exists a constant vector x^* such that

$$E_k R_i' x_i(t) = E_k x^*, \quad k \in \mathcal{R}_i,\ i \in \mathbf{m}. \tag{6}$$

To accomplish this it suffices to show

$$E_k R_i' x_i(t) = E_k R_j' x_j(t), \quad j \in \mathcal{N}_i(t) \cap \mathcal{C}_k,\ k \in \mathcal{R}_i,\ i \in \mathbf{m}. \tag{7}$$

This is clearly a consensus problem which can be solved by choosing $x_i(t+1)$ such that

$$E_k R_i' x_i(t+1) = \sum_{j\in\mathcal{N}_i(t)\cap\mathcal{C}_k} w_{ijk}(t) E_k R_j' x_j(t), \quad k \in \mathcal{R}_i$$

where the $w_{ijk}(t)$ are called *weights* chosen to be such that $w_{ijk}(t) \geq 0$ for $t \geq 1$ and for all $i, j, k \in \mathbf{m}$; $w_{ijk}(t) \neq 0$ if and only if $k \in \mathcal{R}_i \cap \mathcal{R}_j$; and

$$\sum_{j\in\mathcal{N}_i(t)\cap\mathcal{C}_k} w_{ijk}(t) = 1, \quad k \in \mathcal{R}_i,\ i \in \mathbf{m}. \tag{8}$$

Inspired by the consensus literature [11–15], we choose $u_i(t)$ to minimize the square of the 2-norm of

$$\text{col}\left\{E_k R_i' x_i(t+1) - \sum_{j\in\mathcal{N}_i(t)\cap\mathcal{C}_k} w_{ijk}(t) E_k R_j' x_j(t),\ k \in \mathcal{R}_i\right\}. \tag{9}$$

From $R_i =$ col $\{E_k, k \in \mathcal{R}_i\}$ and $R_i R_i' = I$, one has (9) is equivalent to

$$x_i(t+1) - v_i(t) \tag{10}$$

where

$$v_i(t) = \text{col}\left\{\sum_{j\in\mathcal{N}_i(t)\cap\mathcal{C}_k} w_{ijk}(t) E_k R_j' x_j(t),\ k \in \mathcal{R}_i\right\}. \tag{11}$$

Choosing $u_i(t)$ to minimize the square of the 2-norm of (10) leads to an update for agent i in the form of

$$x_i(t+1) = x_i(t) - P_i\,(x_i(t) - v_i(t)) \tag{12}$$

where P_i denotes the orthogonal projection matrix to the kernel of $\bar{A}_i$.