Distributed Weight Selection in Consensus Protocols by Schatten Norm Minimization

Mahmoud El Chamie, Giovanni Neglia, Konstantin Avrachenkov

Distributed Weight Selection in Consensus Protocols by Schatten Norm Minimization

Mahmoud El Chamie, Giovanni Neglia, Konstantin Avrachenkov

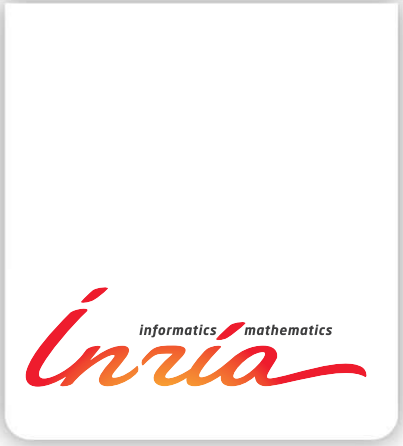
RESEARCH REPORT

N° 8078

October 2012

Project-Teams Maestro

ISSN 0249-6399 ISRN INRIA/RR--8078--FR+ENG



Distributed Weight Selection in Consensus Protocols by Schatten Norm Minimization

Mahmoud El Chamie*†, Giovanni Neglia‡, Konstantin Avrachenkov§

Project-Teams Maestro

Research Report n° 8078 — October 2012 — 23 pages

Abstract: In average consensus protocols, nodes in a network perform an iterative weighted average of their estimates and those of their neighbors. The protocol converges to the average of initial estimates of all nodes found in the network. The speed of convergence of average consensus protocols depends on the weights selected on links (to neighbors). We address in this paper how to select the weights in a given network in order to have a fast speed of convergence for these protocols. We approximate the problem of optimal weight selection by the minimization of the Schatten p-norm of a matrix with some constraints related to the connectivity of the underlying network. We then provide a totally distributed gradient method to solve the Schatten norm optimization problem. By tuning the parameter p in our proposed minimization, we can simply trade-off the quality of the solution (i.e. the speed of convergence) for communication/computation requirements (in terms of number of messages exchanged and volume of data processed). Simulation results show that our approach provides very good performance already for values of p that only needs limited information exchange. The weight optimization iterative procedure can also run in parallel with the consensus protocol and form a joint consensus–optimization procedure.

Key-words: average consensus, weight selection, Schatten norm, distributed gradient algorithms

* Inria Sophia Antipolis, France, Mahmoud.El_Chamie@inria.fr
† University of Nice Sophia Antipolis, France
‡ Inria Sophia Antipolis, France, Giovanni.Neglia@inria.fr
§ Inria Sophia Antipolis, France, K.Avrachenkov@sophia.inria.fr

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Une procédure distribuée de sélection de poids dans les protocoles de consensus par minimisation de la norme de Schatten

Résumé : Dans les protocoles de consensus, les nœuds d'un réseau calculent itérativement une moyenne pondérée de leurs mesures et celles de leurs voisins. Le protocole converge vers la moyenne des mesures initiales de tous les nœuds présents dans le réseau. La vitesse de convergence des protocoles de consensus dépend des poids sélectionnés sur les liens entre voisins. Nous abordons dans cet article la question suivante : comment choisir les poids dans un réseau donné afin d'avoir une plus grande vitesse de convergence du protocole? Nous approchons le problème de la sélection optimale de poids avec un problème de minimisation de la p-norme de Schatten. Ce dernier est résolu de manière totalement distribuée grâce à une mèthode du gradient. Selon la valeur du paramètre p, nous pouvons trouver un compromis entre la qualité de la solution (c'est-à-dire la vitesse de convergence) et les coût en termes de communication et calcul (e.g. nombre de messages échangés et volume de données traitées). Les résultats des simulations montrent que notre approche fournit une très bonne performance même avec un échange d'informations limité. La procédure d'optimisation des poids peut également se dérouler en simultané avec le protocole de consensus.

Mots-clés : consensus de moyenne, sélection de poids, la norme de Schatten, algorithmes distribués de gradient

Contents

1 Introduction

A network is formed of a group of nodes or agents that are interconnected by communication links that allow these nodes to share information and resources. Algorithms for efficient routing and efficient use of resources are proposed to save energy and speed up the processing. For small networks, it is possible for a central unit to be aware of all the components of the network and take decision for optimal usage of a resource on a global view basis. As networks expand, the central unit needs to handle a larger amount of data, and centralized optimization may become unfeasible especially when the network is dynamic. In fact, the optimal configuration needs to be computed whenever a link fails or there is any change in the network. Moreover, nodes may have some processing capabilities that are not used in the centralized optimization. With these points in mind, it becomes more convenient to perform distributed optimization relying on local computation at each node and local information exchange between neighbors. Such distributed approach is intrinsically able to adapt to local network changes. There has been recently a significant amount of research on distributed optimization in networks. New faster techniques ([1, 2, 3]) have been proposed for the traditional dual decomposition approach for separable problems that is well known in the network community since Kelly's seminal work on TCP ([4]). A completely different approach has been recently proposed in [5]: it combines a consensus protocol, that is used to distribute the computations among the agents, and a subgradient method for the minimization of a local objective. Finally a third approach relies on some intelligent random exploration of the possible solution space, e.g. using genetic algorithms ([6]) or the annealed Gibbs sampler ([7]).

In this paper we study distributed techniques to optimally select the weights of average consensus protocols (also referred to as ave-consensus protocols or algorithms). These protocols allow nodes in a network, each having a certain measurement or state value, to calculate the average across all the values in the network by communicating only with their neighbors. Consensus algorithms are used in various applications such as environmental monitoring of wireless sensor networks, and multi-vehicle cooperative control where a team of machines work together to accomplish some predefined goal. Machines moving in parallel for example must reach consensus on the speed and direction of their motion to prevent collisions. Although the average seems to be a simple function to compute, it is an essential element for performing much complex tasks including optimization, source localization, compression, and subspace tracking ([8, 9, 10]).

In the ave-consensus protocol ([11, 12]), each node first selects weights for each of its neighbors, then at each iteration the estimates are exchanged between neighbors and each node updates its own estimate by performing a weighted sum of the values received. Under quite general conditions the estimates asymptotically converge to the average across all the original values. The weights play an essential role to determine the speed of convergence of the ave-consensus. For this reason in this paper we study how to select the weights in a given network in order to have fast convergence independently from initial nodes' estimates. In [13], they refer to this problem as the fastest distributed linear averaging (FDLA) weight selection problem and they show it is equivalent to maximizing the spectral gap of the weight matrix W, that is a convex *non-smooth* function (then we talk about a convex non-smooth optimization problem) which can be solved offline by a centralized node using interior point methods. As we discussed above, this approach may not be convenient for large scale networks with a dynamic topology.

In this work, we propose to select the consensus weights as the values that minimize the Schatten p-norm of the weight matrix under some constraints imposed by the connectivity of the underlying network. We show that this new optimization problem can be considered an approximation of the original problem in [13] (the FDLA) and we reformulate our problem into an equivalent unconstrained, convex and *smooth* minimization, and then it can be solved by

the gradient method. More importantly, we show that in this case the gradient method can be efficiently distributed among the nodes. Let σ_W be the vector containing the singular values of the weight matrix W, then the Schatten p-norm of W is defined as $||W||_{\sigma p} = ||\sigma_W||_p$ where $||\sigma_W||_p$ is the usual p-norm of a vector. We describe a distributed gradient procedure to minimize the Schatten p-norm for an even integer p that requires each node to recover information from nodes that are up to $\frac{p}{2}$-hop distant. Then the order p of the Schatten norm is a tuning parameter that allows us to trade off the quality of the solution for the amount of communication/computation needed. In fact the larger p the more precise the approximation, but also the larger the amount of information nodes need to exchange and process. Our simulation results on random graphs show that our algorithm provides very good performance in comparison to other distributed weighted selection algorithms already for $p = 2$, i.e. when each node needs to collect information only from its direct neighbors. Finally, we show that nodes do not need to run our weight optimization algorithm *before* being able to start the consensus protocol to calculate the average value, but the two can run in parallel.

The paper is organized as follows: In section 2 we formulate the problem we are considering and we give the notation used across the paper. Section 3 presents the existing centralized and distributed approaches for solving the optimization problem. In section 4 we propose Schatten p-norm minimization as an approximation of the original problem and in 5 we propose an efficient decentralized algorithm to solve our approximated problem and we study its computation and communication costs. Section 6 is devoted to a comparison between the performance of our algorithm and that of other known weight selection algorithms on different graph topologies (random graphs). We also implement a parallel algorithm that performs the weight optimization simultaneously with running the consensus protocol. Section 7 summarizes the paper.

2 Problem Formulation

Consider a network of n nodes that can exchange messages between each other through communication links. Every node in this network has a certain measurement (e.g. a measurement for pressure or temperature), and we need each node to know the average of the initial measurements by following a distributed linear iteration approach. The network of nodes can be modeled as a graph $G = (V, E)$ where V is the set of vertices, labeled from 1 to n, and E is the set of edges, then $(i, j) \in E$ if nodes i and j are connected and can communicate (they are neighbors) and $|E| = m$. We label the edges from 1 to m. If link (i, j) has label l, we write $l \sim (i, j)$. Let also N_i be the neighborhood set of node i. All graphs in this report are considered to be *connected* and *undirected*. Let $x_i(0) \in \mathbb{R}$ be the initial value at node i. We are interested in computing the average

$$x_{ave} = (1/n) \sum_{i=1}^{n} x_i(0),$$

in a decentralized manner with nodes only communicating with their neighbors. The averaging is done on a fixed network having a global clock. When the clock ticks, all nodes in the system perform the iteration of the averaging protocol at the same time (this is the synchronous update assumption). At iteration $k + 1$, node i updates its state value x_i as follows:

$$x_i(k + 1) = w_{ii} x_i(k) + \sum_{j \in N_i} w_{ij} x_j(k), \tag{1}$$

where w_{ij} is the weight selected by node i for the value sent by its neighbor j and w_{ii} is the weight selected by node i for it own value. Usually two neighbors select the same weight for each

other, i.e. $w_{ij} = w_{ji}$. Also in this paper we consider this case. The matrix form equation is:

$$\mathbf{x}(k+1) = W\mathbf{x}(k), \tag{2}$$

where $\mathbf{x}(k)$ is the state vector of the system and W is the weight matrix. The main problem we are considering in this paper is how a node i can choose the weights w_{ij} for its neighbors so that the state vector $\mathbf{x}$ of the system converges fast to consensus. There are centralized and distributed ways for selection of W, but in order to explain them, we need first to provide some more notation. We denote by $\mathbf{W}_i$ the vector of weights used by node i (then the transpose of the i^{th} row of matrix W). We denote also by $\mathbf{w}$ the vector of dimensions $m \times 1$, whose l-th element w_l is the weight associated to link l, then if $l \sim (i,j)$ it holds $w_l = w_{ij} = w_{ji}$. A is the adjacency matrix of graph G, i.e. $a_{ij} = 1$ if $(i,j) \in E$ and $a_{ij} = 0$ otherwise. $\mathcal{C}_G$ is the set of all real $n \times n$ matrices M following graph G, i.e. $m_{ij} = 0$ if $(i,j) \notin E$. D is a diagonal matrix where d_{ii} (or simply d_i) is the degree of node i in the graph G. $\mathcal{I}$ is the $n \times m$ incidence matrix of the graph, such that for each $l \sim (i,j) \in E$ $\mathcal{I}_{il} = +1$ and $\mathcal{I}_{jl} = -1$ and the rest of the elements of the matrix are null. L is the laplacian matrix of the graph, so $L = D - A$. It can also be seen that $L = \mathcal{I}\mathcal{I}^T$. I is the $n \times n$ identity matrix. Given that W is real and symmetric, it has real eigenvalues (and then they can be ordered). We denote by λ_i the i-th largest eigenvalue of W, and by μ the largest eigenvalue in module non considering λ_1, i.e. $\mu = \max\{\lambda_2, -\lambda_n\}$. σ_i is the i-th largest singular value of a matrix. $\mathrm{Tr}(B)$ is the trace of the matrix B and $\rho(B)$ is its spectral radius. $||X||_{\sigma p}$ as mentioned before is the Schatten p-norm of a matrix X, i.e. $||X||_{\sigma p} = (\sum_i \sigma_i^p)^{1/p}$. Finally we use the symbol $\frac{\mathrm{d}}{\mathrm{d}X} f(X)$, where f is a differentiable scalar-valued function $f(X)$ with matrix argument $X \in \mathbb{R}^{m \times n}$, to denote the $n \times m$ matrix whose (i,j) entry is $\frac{\partial f(X)}{\partial x_{ji}}$. Table 1 summarizes the notation used in this paper.

2.1 Convergence Conditions

In [13] the following set of conditions is proven to be necessary and sufficient to guarantee convergence to consensus for any initial condition:

$$\mathbf{1}^T W = \mathbf{1}^T, \tag{3}$$

$$W\mathbf{1} = \mathbf{1}, \tag{4}$$

$$\rho(W - \frac{1}{n}\mathbf{1}\mathbf{1}^T) < 1, \tag{5}$$

We observe that the weights are not required to be non-negative. As we said we consider W to be symmetric, then the first two conditions are equivalent to each other and equivalent to the possibility to write the weight matrix as follows: $W = I - \mathcal{I} \times \mathrm{diag}(\mathbf{w}) \times \mathcal{I}^T$.

2.2 Fastest Consensus

The speed of convergence of the system given in (2) is governed by how fast W^k converges. Since W is real symmetric, it has real eigenvalues and it is diagonalizable. We can write W^k as follows ([14]):

$$W^k = \sum_i \lambda_i^k G_i, \tag{6}$$

where the matrices G_i's have the following properties: G_i is the projector onto the null-space of $W - \lambda_i I$ along the range of $W - \lambda_i I$, $\sum_i G_i = I$ and $G_i G_j = 0^{n \times n}$ $\forall i \neq j$. Conditions (3) and (5) imply that 1 is the largest eigenvalue of W in module and is simple. Then $\lambda_1 = 1$, $G_1 = 1/n\mathbf{1}\mathbf{1}^T$ and $|\lambda_i| < 1$ for $i > 1$. From the above representation of W^k, we can deduce two important facts:

Table 1: Notion

Symbol	Description	Dimension
G	network of nodes and links	-
V	set of nodes/vertices	$\|V\| = n$
E	set of links/edges	$\|E\| = m$
$\mathbf{x}(k)$	state vector of the system at iteration k	$n \times 1$
W	weight matrix	$n \times n$
$\mathbf{W}_i$	transpose of ith row of W	$n \times 1$
$\mathbf{w}$	vector of weights on links	$m \times 1$
$\text{diag}(\mathbf{v})$	diagonal matrix having the elements of the $n \times 1$ vector $\mathbf{v}$	$n \times n$
$\mathcal{C}_G$	set of $n \times n$ real matrices following G	-
D	degree diagonal matrix	$n \times n$
A	adjacency matrix of a graph	$n \times n$
$\mathcal{I}$	incidence matrix of a graph	$n \times m$
L	laplacian matrix $L = D - A = \mathcal{I}\mathcal{I}^T$	$n \times n$
λ_i	ith largest eigenvalue of W	scalar
Λ	eigenvalue diagonal matrix $\Lambda_{ii} = \lambda_i$	$n \times n$
σ_i	ith largest sigular value	scalar
μ	second largest eigenvalue in magnitude of W	scalar
$\rho(X)$	spectral radius of matrix X	scalar
$\text{Tr}(X)$	trace of the matrix X	scalar
$\|\|X\|\|_{\sigma p}$	Schatten p-norm of a matrix X	scalar
$\frac{\mathrm{d}}{\mathrm{d}X} f(X)$	Derivative of $f(X)$, $X \in \mathbb{R}^{m \times n}$, $f(X) \in \mathbb{R}$	$n \times m$
$P_S(.)$	Projection on a set $S \subset \mathbb{R}^m$	$m \times 1$

1. First we can check that W^k actually converges, in fact we have $\lim_{k\to\infty} \mathbf{x}(k) = \lim_{k\to\infty} W^k \mathbf{x}(0) = \frac{1}{n}\mathbf{1}\mathbf{1}^T x(0) = x_{ave}\mathbf{1}$ as expected.

2. Second, the speed of convergence of W^k is governed by the second largest eigenvalue in module, i.e. on $\mu = \max\{\lambda_2, -\lambda_n\}$. For obtaining the fastest convergence, nodes have to select weights that minimizes μ, or equivalently maximize the spectral gap[1] of W.

Then the problem of finding the weight matrix that guarantees the fastest convergence can be formalized as follows:

$$\begin{aligned} \text{minimize} \quad & \mu(W) \\ \text{subject to} \quad & W = W^T, \\ & W\mathbf{1} = \mathbf{1}, \\ & W \in \mathcal{C}_G, \end{aligned} \tag{7}$$

where the last constraint on the matrix W derives from the assumption that nodes can only communicate with their neighbors and then necessarily $w_{ij} = 0$ if $(i,j) \notin E$.

The above minimization problem is a convex one and the function $\mu(W)$ is non-smooth convex function. It is convex since when W is a symmetric matrix, we have $\mu(W) = ||W - G_1||_2$ which is a composition between an affine function and the matrix L-2 norm, and all matrix norms are convex functions. The function $\mu(W) = \rho(W - G_1)$ is non-smooth since the spectral

[1] The spectral gap is the difference between the largest eigenvalue in module and the second largest one in module. In this case it is equal to $1 - \mu$.

radius of a matrix is not differentiable at points where the eigenvalues coalesce [15]. The process of minimization itself in (7) tends to make them coalesce at the solution. Therefore, smooth optimization methods cannot be applied to (7). Moreover, the weight matrix solution of the optimization problem is not unique. For example it can be checked that for the network in Fig. 1, there are infinite weight values that can be assigned to the link $(2,3)$ that solves the optimization problem (7), including $w_{23} = 0$.[2]

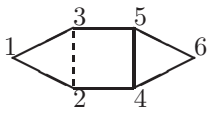


Figure 1: Network of 6 nodes.

We address in this paper the different ways to solve this minimization problem, either in a centralized program by formulating a semi-definite program or by using decentralized techniques such as using a sub-gradient method which is in practice difficult to implement distributedly because of the non-smoothness of the function μ. We then present a differentiable approximation of the problem, and we show how the new optimization problem can be implemented in a fully decentralized manner using gradient techniques. We then compare the approximated solution with the optimal one and other distributed weight selection algorithms such as the metropolis or the max degree ones.

3 Existing Algorithms for Fast Consensus

The function to minimize in the optimization (7) is a non-smooth convex function, so distributed implementation techniques are challenging. In this section, we will present two approaches proposed by Xiao and Boyd in [13] to solve the optimization problem, a centralized approach and a distributed one and a similar approach proposed in [10].

3.1 Centralized Solution

Xiao and Boyd in [13] have shown that problem (7) can be formulated as a Semi-Definite Program (SDP) that can be solved using interior point methods running in polynomial time (see [17]). The semi-definite program is the following:

$$\begin{aligned} &\text{minimize} && s \\ &\text{subject to} && -sI \preceq I - \mathcal{I} \times \text{diag}(\mathbf{w}) \times \mathcal{I}^T - G_1 \\ & && I - \mathcal{I} \times \text{diag}(\mathbf{w}) \times \mathcal{I}^T - G_1 \preceq sI, \end{aligned} \tag{8}$$

where s is an auxiliary real variable and $A \preceq B$ if and only if $B - A$ is positive semi-definite, and G_1 is the projection matrix as in 6.

The output of this program is the optimal weight vector $\mathbf{w}$ such that $w_l, l = 1...m$ is the weight selected for link l. The limit of such centralized approach to weight selection is also shown by

[2] Additionally, the remark above shows that adding an extra link in a graph (e.g. link $(2,3)$ in the Fig. 1), does not necessarily reduce the second largest eigenvalue of the optimal weight matrix, as it was erroneously conjectured in [16].

the fact that a popular solver as `CVX`, matlab software for disciplined convex programming [18], can only find the solution of (8) for networks with at most tens of thousands of links. Moreover any topology change will require to be notified to the central unity that should solve again the problem.

3.2 Distributed Solution

Contrary to the centralized approach, in a distributed solution all the nodes in the network contribute to calculate the solution of the optimization problem. The whole network then benefits from nodes' processing capabilities.

The authors of [13] present a sub-gradient method for selecting weights on links in a network by minimizing the unconstrained problem:

$$\text{minimize } r(\mathbf{w}) = \rho(I - \mathcal{I} \times \text{diag}(\mathbf{w}) \times \mathcal{I}^T - G_1).$$

Each link weight is updated according to the following sub-gradient iteration:

$$w_l^{(k+1)} = w_l^{(k)} - \gamma^{(k)} g_l^{(k)} / ||\mathbf{g}^{(k)}||, \tag{9}$$

where $w_l^{(k)}$ is the weight on link l at iteration k, $g_l^{(k)}$ is the l-th component of a subgradient $\mathbf{g}^{(k)}$ of the function to minimize calculated in $\mathbf{w}^{(k)}$ and $\gamma^{(k)}$ is the stepsize satisfying the following sufficient conditions for convergence, $\lim_{k\to\infty} \gamma^{(k)} = 0$ and $\sum_{k=1}^{\infty} \gamma^{(k)} = \infty$. The components of the sub-gradient can be calculated as follows:

- if $r(\mathbf{w}) = \lambda_2(W)$, then

  $$g_l = -(u_i - u_j)^2, \text{ if } l \sim (i,j),\ l = 1, ..., m$$

  where u_i is the i-th component of a unit eigenvector of the weight matrix $W(k)$ relative to the eigenvalue λ_2.

- if $r(\mathbf{w}) = -\lambda_n(W)$, then

  $$g_l = (u_i - u_j)^2, \text{ if } l \sim (i,j),\ l = 1, ..., m$$

  where u_i is the i-th component of a unit eigenvector of the weight matrix $W(k)$ relative to the eigenvalue λ_n.

We observe that the stepsize used in (9) is normalized by $||\mathbf{g}^{(k)}||$ which cannot be locally computed by each node. While this problem can probably be circumvented by a different choice of the stepsize (without loosing the convergence properties of (9)), there are other aspects that make problematic this distributed implementation. In fact this iterative procedure requires at every step to calculate $\lambda_2(W(k))$ and $\lambda_n(W(k))$, and determine one eigenvector for the one of these two eigenvalues that is the largest in module. For the solution to be really distributed also these quantities have to be calculated in a distributed way. This is not an easy task. There are some distributed iterative techniques ([19, 20, 21]) that converge asymptotically to the correct eigenvalue-eigenvector pair. Then each step of the optimization procedure requires itself the convergence of an iterative sub-procedure to calculate the two eigenvalue and the eigenvector with significant computation and communication costs. We remark in particular that at each step the sub-procedure has to run long enough to guarantee that the estimations are accurate enough to not jeopardize the convergence of the optimization procedure. Deciding when to

terminate the sub-procedure at each step may require itself another distributed mechanisms or the use of worst-case bounds on the errors.

A similar optimization problem but with some additional constraints is to find the fastest converging algorithm for randomized gossiping, and it has been studied in [10]. The authors provide a subgradient method that projects the variables violating the constraints back onto the feasible set. The projection can be done in a distributed way and the stepsize sequence can be calculated at each node. Nevertheless, the gradient of the cost function depends also in this case on eigenvalues and eigenvectors of the underlying graph, so its calculation incurs the same problems exposed above.

The main contribution of this paper is to consider a different cost function that approximates that in the original problem (7) (i.e. the module of the second largest eigenvalue) and to propose a distributed gradient method to obtain the optimal solution. The main difference in comparison to the approach described in this section—beside the obvious difference to deal with differentiable function rather than with non-smooth one— is that each component of the gradient can be expressed in a closed form and calculated by each node only on the basis of some local information, without the need to perform an iterative sub-procedure at each step.

4 Schatten Norm Minimization

We change the original minimization problem in (7) by considering a different cost function that is a monotonic function of the Schatten Norm. The minimization problem we propose is the following one:

$$\begin{array}{ll} \text{minimize} & f(W) = ||W||_{\sigma p}^{p} \\ \text{subject to} & W = W^T, \\ & W\mathbf{1} = \mathbf{1}, \\ & W \in \mathcal{C}_G, \end{array} \tag{10}$$

where p is an even positive integer. The following result establishes that (10) is a smooth convex optimization problem and also it provides an alternative expression of the cost function in terms of the trace of W^p. For this reason we refer to our problem also as *Trace Minimization* (TM).

Proposition 1. *$f(W) = ||W||_{\sigma p}^{p} = Tr(W^p)$ is a scalar-valued smooth convex function on its feasible domain when p is an even positive integer.*

Proof. We have $\mathrm{Tr}(W^p) = \sum_{i=1}^{n} \lambda_i^p$. Since W is symmetric, its non-zero singular values are the absolute values of its non-zero eigenvalues ([14]). Given that p is even, then $\sum_{i=1}^{n} \lambda_i^p = \sum_{i=1}^{n} \sigma_i^p$. Therefore, $\mathrm{Tr}(W^p) = ||W||_{\sigma p}^{p}$.

The Schatten norm $||W||_{\sigma p}$ is a nonnegative convex function, then f is convex because it is the composition of a non-decreasing convex function—function x^p where x is non-negative—and a convex function (see [22]).

The function is also differentiable and we have

$$\frac{\mathrm{d}}{\mathrm{d}W}\mathrm{Tr}(W^p) = pW^{p-1}, \tag{11}$$

(see [23, p. 411]). □

We now illustrate the relation between (10) and the optimization (7). The following lemmas will prepare the result:

Lemma 1. *For any symmetric weight matrix W whose rows (and columns) sum to 1 and with eigevalues $\lambda_1(W) \geq \lambda_2(W) \geq \ldots \lambda_n(W)$, there exists two integers $K_1 \in \{1, 2, \ldots n-1\}, K_2 \in \{0, 1, 2, \ldots n-1\}$ and a positive constant $\alpha < 1$ such that for any positive integer $p = 2q$ we have:*

$$1 + \tau(W)^p K_1 \leq Tr(W^p) \leq 1 + \tau(W)^p (K_1 + K_2 \alpha^p), \tag{12}$$

where

$$\tau(W) = \begin{cases} \rho(W) = \max\{\lambda_1(W), -\lambda_n(W)\} & \text{if } \rho(W) > 1, \\ \mu(W) = \max\{\lambda_2(W), -\lambda_n(W)\} & \text{if } \rho(W) \leq 1. \end{cases} \tag{13}$$

Proof. Let us consider the matrix W^2 and denote by $\nu_1, \nu_2, \ldots \nu_r$ its distinct eigenvalues ordered by the largest to the smallest and by $m_1, m_2, \ldots m_r$ their respective multiplicities. We observe that they are all non-negative and then they are also different in module. For convenience we consider $\nu_s = m_s = 0$ for $s > r$. We can then write:

$$\mathrm{Tr}(W^p) = \sum_{i=1}^{n} \lambda_i^p = \sum_{i=1}^{r} m_i \nu_i^q.$$

The matrix W^2 has 1 as an eigenvalue. Let us denote by j its position in the ordered sequence of distinct eigenvalues, i.e. $\nu_j = 1$. Then it holds:

$$\mathrm{Tr}(W^p) = 1 + (m_j - 1) + \sum_{i \neq j} m_i \nu_i^q.$$

If $\rho(W) = 1$ (i.e. 1 is the largest eigenvalue in module of W), then 1 is also the largest eigenvalue of W^2 ($\nu_1 = 1$). If $m_1 > 1$, then it has to be either $\lambda_2(W) = 1$ (the multiplicity of the eigenvalue 1 for W is larger than 1) or $\lambda_n(W) = -1$. In both cases $\tau(W) = \mu(W) = 1$,

$$\mathrm{Tr}(W^p) = 1 + (m_1 - 1) + \sum_{i>1} m_i \nu_i^q$$

and the result holds with $K_1 = m_1 - 1$, $K_2 = \sum_{i>1} m_i$ and $\alpha = \sqrt{\nu_2}$. If $m_1 = 1$, then $\nu_2 = \lambda_2^2$. We can write:

$$\mathrm{Tr}(W^p) = 1 + \nu_2^q \left(m_2 + \sum_{i>2} m_i \left(\frac{\nu_i}{\nu_2} \right)^q \right)$$

and the result holds with $K_1 = m_2$, $K_2 = \sum_{i>2} m_i$, and $\alpha = \sqrt{\nu_3/\nu_2}$.

If $\rho(W) > 1$, then $\nu_1 = \rho(W)^2 > 1$ and we can write:

$$\mathrm{Tr}(W^p) = 1 + \nu_1^q \left(m_1 + \sum_{\substack{i>1 \\ i \neq j}} m_i (\frac{\nu_i}{\nu_1})^q + (m_j - 1)(\frac{1}{\nu_1})^q \right).$$

Then the result holds with $\tau(W) = \sqrt{\nu_1} = \rho(W)$, $K_1 = m_1$, $K_2 = \sum_{i>1} m_i$, and $\alpha = \sqrt{\nu_2/\nu_1}$. □

Lemma 2. *Let us denote by $W_{(p)}$ the solution of the minimization problem* (10)*. If the graph of the network is strongly connected then $\tau\left(W_{(p)}\right) < 1$ for p sufficiently large.*

Proof. If the graph is strongly connected then there are multiple ways to assign the weights such that the convergence conditions (3)-(5) are satisfied. In particular the local degree method described in Sec. 6 is one of them. Let us denote by $W_{(LD)}$ its weight matrix. A consequence of the convergence conditions is that 1 is a simple eigenvalue of $W_{(LD)}$, and that all other eigenvalues are strictly less than one in magnitude (see [13]). It follows that $\tau\left(W_{(LD)}\right)$ in Lemma 1 is strictly smaller than one and that $\lim_{p\to\infty} \mathrm{Tr}\left(W_{(LD)}^p\right) = 1$. Then there exists a value p_0 such that for each $p > p_0$

$$\mathrm{Tr}\left(W_{(LD)}^p\right) < 2.$$

Let us consider the minimization problem (10) for a value $p > p_0$. $W_{(LD)}$ is a feasible solution for the problem, then

$$\mathrm{Tr}(W_{(p)}^p) \leq \mathrm{Tr}(W_{LD}^p) < 2.$$

Using this inequality and Lemma 1, we have:

$$1 + \tau\left(W_{(p)}\right)^p \leq 1 + \tau\left(W_{(p)}\right)^p K_1 \leq \mathrm{Tr}(W_{(p)}^p) < 2,$$

from which the thesis follows immediately. □

We are now ready to state our main result:

Proposition 2. *If the graph of the network is strongly connected, then the solution of the Schatten Norm minimization problem* (10) *satisfies the consensus protocol convergence conditions for p sufficiently large. Moreover when p diverges this minimization problem is equivalent to the minimization problem* (7) *(i.e. to minimize the second largest eigenvalue $\mu(W)$).*

Proof. The solution of problem (10), $W_{(p)}$ is necessarily symmetric and its rows sum to 1. From Lemma 2 it follows that for p sufficiently large $\tau\left(W_{(p)}\right) < 1$ then by the definition of $\tau(.)$ it has to be $\rho(W_{(p)}) = 1$ and $\mu(W_{(p)}) < 1$. Therefore $W_{(p)}$ satisfies all the three convergence conditions (3)-(5) and then the consensus protocol converges.

Now we observe that with respect to the variable weight matrix W, minimizing $\mathrm{Tr}(W^p)$ is equivalent to minimizing $(\mathrm{Tr}(W^p) - 1)^{1/p}$. From Eq. (12), it follows:

$$\tau(W)K_1^{\frac{1}{p}} \leq (\mathrm{Tr}(W^p) - 1)^{\frac{1}{p}} \leq \tau(W)(K_1 + K_2\alpha^p)^{\frac{1}{p}}.$$

K_1 is bounded between 1 and $n-1$ and K_2 is bounded between 0 and $n-1$, and $\alpha < 1$,then it holds:

$$\tau(W)K_1^{\frac{1}{p}} \leq (\mathrm{Tr}(W^p) - 1)^{\frac{1}{p}} \leq \tau(W)K^{\frac{1}{p}},$$

with $K = 2(n-1)$. For p large enough $\tau\left(W_{(p)}\right) = \mu(W_{(p)})$, then

$$\left|(\mathrm{Tr}(W_{(p)}^p) - 1)^{\frac{1}{p}} - \mu(W_{(p)})\right| \leq \mu(W_{(p)})\left(K^{\frac{1}{p}} - 1\right) \leq K^{\frac{1}{p}} - 1.$$

Then the difference of the two cost functions converges to zero when p diverges. In this sense, Schatten Norm minimization (10) can be considered an approximation for the original problem (7). □

Remark Comparing the results of Schatten Norm minimization (10) with the original problem (7), we observe that on some graphs the solution of problem (10) already for $p = 2$ gives the optimal solution of the main problem (7); this is for example the case for fully connected

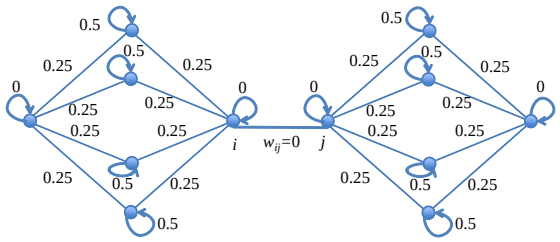


Figure 2: For this network the matrix solution of Schatten Norm minimization (10) with $p = 2$ does not guarantee convergence of average consensus, because $w_{i,j} = 0$.

graphs[3]. However, on some other graphs, it may give a weight matrix that does not guarantee the convergence of the consensus protocol because the second largest eigenvalue is larger than or equal to 1 (the other convergence conditions are intrinsecally satisfied). We have built a toy example, shown in Fig. 2, where this happens. The solution of (10) assigns weight 0 to the link $\{i, j\}$; $w_{ij} = 0$ separates the network into two disconnected subgraphs, so $\mu(W) = 1$ in this case. We know by Lemma 2 that this problem cannot occur for p large enough. In particular for the toy example the matrix solution for $p = 4$ already guarantees convergence. Moreover we observe that we have never observed this problem on the larger networks we have simulated.

Given that problem (10) is smooth and convex, it can be solved by interior point methods which would be a centralized solution. In the next section we are going to show a distributed algorithm to solve problem (10).

5 A Distributed Algorithm for Schatten Norm minimization

In this section we will show that the optimization problem (10) can be solved in a distributed way using gradient methods. By distributed algorithm we mean an algorithm where each node only needs to retrieve information from a limited neighborhood (possibly larger than N_i) in order to calculate the weights if its incident links. In what follows for the sake of simplicity we will sometimes describe nodes' operations in terms of the matrix W or of its rows $\mathbf{W}_i$ $(i = 1, 2, \dots n)$. For example one step of our procedure requires node i to "send the weight vector $\mathbf{W}_i$ to its neighbour." This may erroneously suggest that some global knowledge is required at each node. But due to the constraint $W \in \mathcal{C}_G$, any row i of the matrix W can only have values different from zero at column j where $j \in N_i$. Then node i only needs to transmit the weight of its incident links and its own weight w_{ii} in the form an adiacency list and does not even need to know the total number of nodes in the network (i.e. the length of the vector $\mathbf{W}_i$). Below we discuss in detail more complex operations, but the reader has been warned.

The constraint $W = W^T$ in the optimization requires any two neighbors i and j to choose the same weight on their common link $l \sim \{i, j\}$ i.e. $w_{ij} = w_{ji} = w_l$. The last condition $W\mathbf{1} = \mathbf{1}$ means that at every node i, the sum of all weights on its incident links plus its self-weight w_{ii} must be equal to one. This condition can be satisfied by an algorithm that first chooses weights on links, and then each node will choose the variable w_{ii} to satisfy this condition. Moreover

[3] This can be easily checked. In fact, for any convergent matrix it holds $\mu(W) \geq 0$ and $\mathrm{Tr}(W^2) \geq 1$. The matrix $\hat{W} = 1/n\mathbf{1}\mathbf{1}^T$ (corresponding to each link having the same weight $1/n$) has eigenvalues 1 and 0 with multiplicity 1 and $n - 1$ respectively. Then $\mu(\hat{W}) = 0$ and $\mathrm{Tr}(\hat{W}^2) = 1$. It follows that $\hat{W}$ minimizes both the cost function of problem (7) and (10).

these two constraints lead to the possibility to write W as follows: $W = I - \mathcal{I} \times diag(\mathbf{w}) \times \mathcal{I}^T$, where $\mathbf{w} \in \mathbb{R}^m$ is the vector of all the weight links w_l, $l = 1...m$. It follows that Schatten Norm minimization (10) is equivalent to the following unconstrained problem:

$$\text{minimize } h(\mathbf{w}) = Tr\left((I - \mathcal{I} \times \text{diag}(\mathbf{w}) \times \mathcal{I}^T)^p\right). \tag{14}$$

We will give a distributed algorithm to solve the Schatten Norm minimization (10) by applying gradient techniques to problem (14). Since the cost function to optimize is smooth and convex as we proved in Proposition 1, if the gradient technique converges to a stationary point, then it converges to the global optimum. The gradient method uses the simple iteration:

$$w_l^{(k+1)} = w_l^{(k)} - \gamma^{(k)} g_l^{(k)} \ \forall l = 1...m \ ,$$

where $\gamma^{(k)}$ is the stepsize at iteration k and $g_l^{(k)}$ is the l-th component of the gradient $\mathbf{g}^{(k)}$ of the function $h(\mathbf{w})$. At every iteration k, starting with a feasible solution for link weights, $w_l^{(k)}$, we calculate the gradient $g_l^{(k)}$ for every link, and then we obtaine a new weight value $w_l^{(k+1)}$. There are different conditions on the function $h(.)$ and on the stepsize sequence that can guarantee convergence (see for example [24]). In our case, the results that guarantee global convergence cannot be applied , because the function $h(.)$ does not satisfy some conditions (e.g. Lipschitz continuity), or because stepsize calculation would require some global knowledge (e.g. the value of the function $h(.)$ or the module of its gradient). We will then add a further constraint, looking for a solution in a set X, and we will consider the following projected gradient method:

$$\mathbf{w}^{(k+1)} = P_X\left(\mathbf{w}^{(k)} - \gamma^{(k)}\mathbf{g}^{(k)}\right),$$

where $P_X()$ is the projection on the set X. We will show that by a particular choice of X and $\gamma^{(k)}$ the method converges to the solution of the original problem. Moreover, all the calculations can be performed in a distributed way on the basis of local knowledge. In particular, we will show that:

- nodes incident to l are able to calculate $g_l^{(k)}$ using only information they can retrieve from their (possibly extended) neighborhood;
- the stepsize sequence $\gamma^{(k)}$ is determined a priori and then nodes do not need to evaluate the function h or any other global quantity to calculate it;
- the projection on set X can be reduced to projection by coordinate which is calculated locally at nodes.

We will start by g_l and show that it only depends on information local to nodes i and j incident to the link $l \sim \{i, j\}$, then we will discuss the choice of the stepsize $\gamma^{(k)}$ and of the projection set X.

5.1 Locally Computed Gradient

Consider the link $l \sim \{i,j\}$, then $w_l = w_{ij} = w_{ji}$ and $w_{ii} = 1 - \sum_{s \in N_i} w_{is}$. The gradient g_l of the function $h(\mathbf{w})$ can be calculated as follows:

$$\begin{aligned} g_l &= \frac{\mathrm{d}h(\mathbf{w})}{\mathrm{d}w_l} \\ &= \frac{\mathrm{d}f(I - \mathcal{I} \times \mathrm{diag}(\mathbf{w}) \times \mathcal{I}^T)}{\mathrm{d}w_l} \\ &= \frac{\partial f}{\partial w_{ij}}\frac{\mathrm{d}w_{ij}}{\mathrm{d}w_l} + \frac{\partial f}{\partial w_{ji}}\frac{\mathrm{d}w_{ji}}{\mathrm{d}w_l} + \frac{\partial f}{\partial w_{ii}}\frac{\mathrm{d}w_{ii}}{\mathrm{d}w_l} + \frac{\partial f}{\partial w_{jj}}\frac{\mathrm{d}w_{jj}}{\mathrm{d}w_l} \\ &= \frac{\partial f}{\partial w_{ij}} + \frac{\partial f}{\partial w_{ji}} - \frac{\partial f}{\partial w_{ii}} - \frac{\partial f}{\partial w_{jj}} \\ &= p\big((W^{p-1})_{ji} + (W^{p-1})_{ij} - (W^{p-1})_{ii} - (W^{p-1})_{jj}\big). \end{aligned} \tag{15}$$

In the last equality we used equation (11).

It is well know from graph theory that if we consider W to be the adjacency matrix of a weighted graph G, then $(W^s)_{ij}$ is a function of the weights on the edges of the $i-j$ walks (i.e. the walks from i to j) of length exactly s (in particular for an unweighted graph $(W^s)_{ij}$ is the number of distinct $i-j$ s-walks [25]). Since for a given p the gradient g_l, $l \sim \{i,j\}$, depends on the $\{ii, jj, ij, ji\}$ terms of the matrix W^{p-1}, g_l can be calculated locally by using only the weights of links and nodes at most $\frac{p}{2}$ hops away from i or j[4] . Practically speaking, at each step, nodes i and j need to contact all the nodes up to $p/2$ hops away in order to retrieve the current values of the weights on the links of these nodes and the values of weights on the nodes themselves. For example, when $p = 2$, then the minimization is the same as the minimization of the Frobenius norm of W since $Tr(W^2) = \sum_{i,j} w_{ij}^2 = ||W||_F^2$, and the gradient g_l can be calculated as $g_l = 2 \times (2W_{ij} - W_{ii} - W_{jj})$ which depends only on the weights of the vertices incident to that link and the weight of the link itself.

An advantage of our approach is that it provides a trade-off between locality and optimality. In fact, the larger the parameter p, the better the solution of problem (10) approximates the solution of problem (7), but at the same time the larger is the neighborhood from which each node needs to retrieve the information. For example, if $p = 2$, then g_l where $l \sim \{i,j\}$ only depends on the weights of subgraph induced by the two nodes i and j. For $p = 4$, the gradient g_l depends only on the weights found on the subgraph induced by the set of vertices $N_i \cup N_j$, then it is sufficient that nodes i and j exchange the weights of all their incident links.

5.2 Choice of Stepsize and Projection set

The global convergence of gradient methods (i.e. for any initial condition) has been proven under a variety of different hypotheses on the function h to minimize and on the step size sequence $\gamma^{(k)}$. In many cases the step size has to be adaptively selected on the basis of the value of the function or of its gradient at the current estimate, but this cannot be done in a distributed way. This leads us to look for convergence results where the step size sequence can be fixed ahead of time. Moreover the usual conditions, like Lipschitzianity or boundness of the gradient, are not satisfied by the function $h(.)$ over all the feasible set. For this reason we add another constraint to our original problem (14) by considering that the solution has to belong to a given convex and

[4] If a link or a node is more than $p/2$ hops away both from node i and node j, then it cannot belong to a $i-j$ walk of lenght p.

compact set X. Before further specifying how we choose the set X, we state our convergence result.

Proposition 3. *Given the following problem*

$$\begin{aligned} &\textit{minimize} && h(\mathbf{w}) = Tr\left((I - \mathcal{I} \times diag(\mathbf{w}) \times \mathcal{I}^T)^p\right), \\ &\textit{subject to} && \mathbf{w} \in X \end{aligned} \tag{16}$$

where $X \subseteq \mathbb{R}^m$ is a convex and compact set, if $\sum_k \gamma^{(k)} = \infty$ and $\sum_k \left(\gamma^{(k)}\right)^2 < \infty$, then the following iterative procedure converges to the minimum of h in X:

$$\mathbf{w}^{(k+1)} = P_X\left(\mathbf{w}^{(k)} - \gamma^{(k)}\mathbf{g}^{(k)}\right), \tag{17}$$

where $P_X(.)$ is the projection operator on the set X and $\mathbf{g}^{(k)}$ is the gradient of h evaluated in $\mathbf{w}^{(k)}$.

Proof. The function h is continuous on a compact set X, so it has a point of minimum. Moreover also the gradient $\mathbf{g}$ is continuous and then bounded on X. The result then follows from Proposition 8.2.6 in [26, pp. 480]. □

For example, $\gamma^{(k)} = a/(b+k)$ where $a > 0$ and $b \geq 0$ satisfies the step size condition in Proposition 3.

While the convergence is guaranteed for any set X convex and compact, we have two other requirements. First, it should be possible to calculate the projection P_X in a distributed way. Second, the set X should contain the solution of the optimization problem (14). About the first issue, we observe that if X is the cartesian product of real intervals, i.e. if $X = [a_1, b_1] \times [a_2, b_2] \times \ldots [a_m, b_m]$, then we have that the l-th component of the projection on X of a vector $\mathbf{y}$ is simply the projection of the l-th component of the vector on the interval $[a_l, b_l]$, i.e.:

$$[P_X(\mathbf{y})]_l = P_{[a_l,b_l]}(y_l) = \begin{cases} a_l & \text{if } y_l < a_l, \\ y_l & \text{if } a_l \leq y_l \leq b_l, \\ b_l & \text{if } b_l < y_l. \end{cases} \tag{18}$$

Then in this case Eq. (17) can be written component-wise as

$$w_l^{(k+1)} = P_{[a_l,b_l]}(w_l^{(k)} - \gamma^{(k)} g_l^{(k)}).$$

We have shown in the previous section that g_l can be calculated in a distributed way, then the iterative procedure can be distributed. About the second issue, we choose X in such a way that we include in the feasibility set all the weight matrices with spectral radius at most 1. The following lemma indicates us how to choose X.

Lemma 3. *Let W be a real and symmetric matrix where each row (and column) sums to 1, then the following holds,*

$$\rho(W) = 1 \implies \max_{i,j} |w_{ij}| \leq 1.$$

Proof. Since W is real and symmetric, then we can write W as follows

$$W = S\Lambda S^T,$$

where S is an orthonormal matrix ($S^TS = SS^T = I$), and Λ is a diagonal matrix having $\Lambda_{kk} = \lambda_k$ and λ_k is the k-th largest eigenvalue of W. Let $\mathbf{r}_k$ and $\mathbf{c}_k$ be the rows and columns of S respectively and $r_k^{(i)}$ be the i-th element of this vector. So,

$$W = \sum_k \lambda_k \mathbf{c}_k \mathbf{c}_k^T,$$

and

$$|w_{ij}| = |\sum_k \lambda_k c_k^{(i)} c_k^{(j)}| \tag{19}$$

$$\leq \sum_k |c_k^{(i)}||c_k^{(j)}| \tag{20}$$

$$= \sum_k |r_i^{(k)}||r_j^{(k)}| \tag{21}$$

$$\leq ||\mathbf{r}_i||_2 ||\mathbf{r}_j||_2 \tag{22}$$

$$= 1. \tag{23}$$

The transition from (19) to (20) is due to the fact $\rho(W) = 1$, the transition from (21) to (22) is due to Cauchy–Schwarz inequality. The transition from (22) to (23) is due to the fact that S is an orthonormal matrix. □

A consequence of Lemma 3 is that if we choose $X = [-1, 1]^m$ the weight vector of the matrix solution of problem (7) necessarily belongs to X (the weight matrix satisfies the convergence conditions). The same is true for the solution of problem (14) for p large enough because of Proposition 2. The following proposition summarizes our results.

Proposition 4. *If the graph of the network is strongly connected, then the following distributed algorithm converges to the solution of the Schatten norm minimization problem for p large enough:*

$$w_l^{(k+1)} = P_{[-1,1]}(w_l^{(k)} - \gamma^{(k)} g_l^{(k)}), \quad \forall l = 1, \ldots, m, \tag{24}$$

where $\sum_k \gamma^{(k)} = \infty$ *and* $\sum_k \left(\gamma^{(k)}\right)^2 < \infty$.

5.3 Complexity of the Algorithm

Our distributed algorithm for Schatten Norm minimization requires to calculate at every iteration, the stepsize $\gamma^{(k)}$, the gradient $g_l^{(k)}$ for every link, and a projection on the feasible set X. Its complexity is determined by the calculation of link gradient g_l, while the cost of the other operations is negligible. In what follows we are going to detail both computational costs incurred by each node (in terms of number of operations and memory required) and communication costs (in terms of volume of information to transmit) for the two values $p = 2$ and $p = 4$.

5.3.1 Complexity for $p = 2$

For $p = 2$, $g_l = 2 \times (2W_{ij} - W_{ii} - W_{jj})$, so taking into consideration that nodes are aware of their own weights (W_{ii}) and of the weights of the links they are incident to (W_{ij}), the only missing parameter in the equation is their neighbors self weight (W_{jj}). So after every iteration in the subgradient methods, nodes must broadcast their self weight to their neighbors. We can say that the computational complexity for $p = 2$ is negligible and the communication complexity is 1 message carrying a single real value (w_{ii}) per link, per node and per iteration.

5.3.2 Complexity for $p = 4$

For $p = 4$, the node must collect information from a larger neighborhood. The gradient at link $l \sim \{i, j\}$ is given by $g_l = 4\big((W^3)_{ij} + (W^3)_{ji} - (W^3)_{ii} - (W^3)_{jj}\big)$. Notice that g_l can be written as follows:

$$g_l = 4\big(\mathbf{W}_i^T W \mathbf{W}_j + \mathbf{W}_j^T W \mathbf{W}_i - \mathbf{W}_i^T W \mathbf{W}_i - \mathbf{W}_j^T W \mathbf{W}_j\big).$$

From the equation of g_l it seems like the node must be aware of all the weight matrix in order to calculate the 4 terms in the equation, however this is not true. As discussed in the previous section, each of the 4 terms can be calculated only locally from the weights within 2-hops from i or j. In fact, $\mathbf{W}_i^T W \mathbf{W}_j$ depends only on the weights of links covered by a walk with 3 jumps, starting from i we do the first jump to a neighbor of i, the second jump to a neighbor of j and finally the third jump must finish at j, then we cannot move farther than 2 hops from i. Then this term can be calculated at node i, i.e. every node s in N_i, sends its weight vector $\mathbf{W}_s$ to i. The same is true for the addend $\mathbf{W}_j^T W \mathbf{W}_i$. The term $\mathbf{W}_i^T W \mathbf{W}_i$ depends on the walks of length 3 starting and finishing in i, then node i can calculate it once it knows $\mathbf{W}_s$ for each s in N_i. Finally, the calculation of the term $\mathbf{W}_j^T W \mathbf{W}_j$ at node i requires more information about the links existing among the neighbors of node j. Instead of the transmission of this detailed information, we observe that node j can calculate the value $\mathbf{W}_j^T W \mathbf{W}_j$ (as node i can calculate $\mathbf{W}_i^T W \mathbf{W}_i$) and then can transmit directly the result of the calculation to node i. Therefore, the calculation of g_l by node i for every link l incident to i can be done in three steps:

1. Create the subgraph H containing the neighbors of i and the neighbors of its neighbors by sending ($\mathbf{W}_i$) and receiving the weight vectors ($\mathbf{W}_j$) from every neighbors j.
2. Calculate $\mathbf{W}_i^T W \mathbf{W}_i$ and broadcast it to the neighbors.
3. Calculate g_l.

We evaluate now both the computational and the communication complexity.

- Computation Complexity: Each node must store the subgraph H of its neighborhood. The number of nodes of H is $n_H \leq \Delta^2 + 1$, the number of links of H is $m_H \leq \Delta^2$ where Δ is the maximum degree in the network. Due to sparsity of matrix W, the calculation of the value $\mathbf{W}_i^T W \mathbf{W}_i$ requires $O(\Delta^3)$ multiplication operation without the use of any accelerating technique in matrix multiplication which —we believe— could further reduce the cost. So the total cost for calculating g_l is in the worst case $O(\Delta^3)$. Notice that the complexity for solving the SDP (8) is of order $O(m^3)$ where m is the number of links in the network. Therefore, on networks where $\Delta << m$, the gradient method would be computationally more efficient.
- Communication Complexity: The packets that need to be transmitted by nodes are due to steps 1 and 2. So the complexity would be two messages per link per gradient iteration. The first message carries at most Δ values (the weight vector $\mathbf{W}_i$) and the second message carries one real value ($\mathbf{W}_i^T W \mathbf{W}_i$).

6 Performance Evaluation

As we have discussed in Section 2.2, the speed of convergence of W^k is asymptotically determined by the second largest eigenvalue in module. For this reason $\mu(W)$ will be used as a performance metric to compare different weight selection algorithms for choosing the weight matrix W. We will start the evaluation of our algorithm by first presenting in Fig. 3 a comparison between

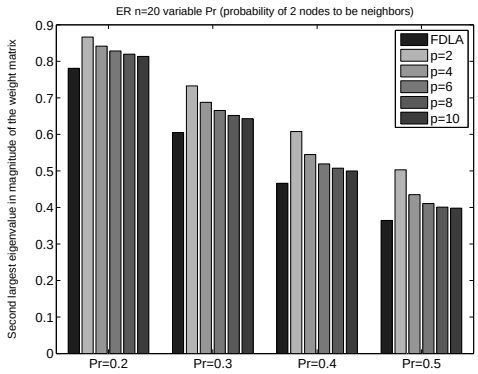


Figure 3: Performance comparison between FDLA solved by SDP and the Schatten Norm minimization of different values of p.

the optimal algorithm which is the fastest distributed linear averaging (FDLA) that minimizes μ solving the SDP problem (8) (we used the `CVX` solver, see [18]) and the different results of choosing the weight matrix W by our proposed Schatten p-Norm minimization formulation for different values of p. The values in the figure are averaged over 100 connected Erdos-Reyni graphs with 20 nodes each. An ER random graph is generated as follows: we start from a 20 nodes fully connected graph, and then every link is removed from the graph by a probability $(1 - Pr)$ and is left there with a probability Pr. The comparison is done for different probability values Pr. We see from the results that as we solve the trace minimization for higher order p, the weight matrix solution of problem (10) approaches that of (7) as we discussed in Proposition 2.

The weights obtained by solving the optimization for $p = 2$ and $p = 4$ are based on local information exchange only. We can compare the performance of our proposed weight selection algorithm with the following ones from the literature (see [27, 13]):

- max degree weights (MD):
  $w_l = \frac{1}{\Delta+1} \quad \forall l = 1...m.$
- local degree (metropolis) weights (LD):
  $w_l = \frac{1}{\max\{d_i,d_j\}+1}\ l \sim \{i,j\} \quad \forall l = 1, 2, \ldots m.$
- optimal constant weights (OC):
  $w_l = \frac{2}{\lambda_1(L)+\lambda_{n-1}(L)}\ \forall l = 1...m.$

where $\Delta = \max_i\{d_i\}$ is the maximum degree in the network and L is the Laplacian of the graph. The weight matrix can be then deduced from $\mathbf{w}$:

$$W = I - \mathcal{I} \times \mathrm{diag}(\mathbf{w}) \times \mathcal{I}^T.$$

6.1 Asymptotic Convergence Rate

We compare now the second largest eigenvalue of the weight matrix $\mu(W)$ of the different weight selection algorithms with our proposed algorithm for $p = 2$ and $p = 4$. The comparison is done on Erdos Renyi (ER) random graphs described earlier, and on Random Geometric Graphs (RGG) where n nodes are thrown uniformly at random on a unit square area, and any two nodes within a connectivity radius r are connected by a link. We have tested the performance for different

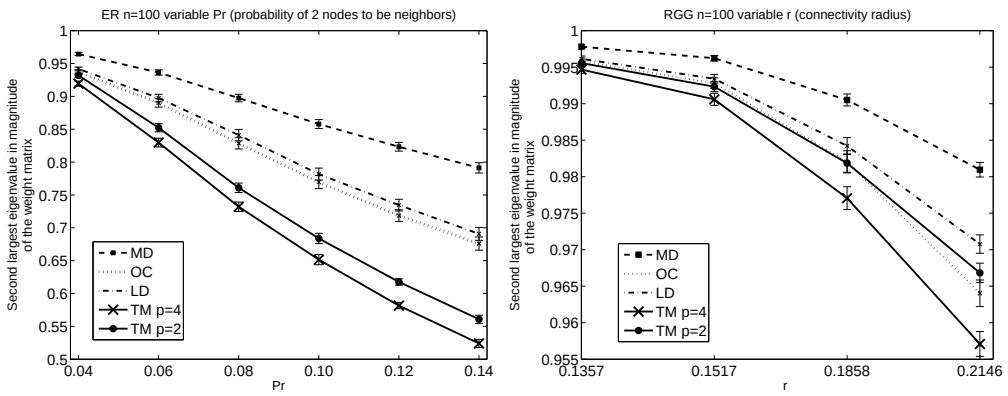


Figure 4: Performance comparison between Schatten Norm minimization (TM) for $p = 2$ and $p = 4$ with other weight selection algorithms on ER and RGG graphs.

connectivity radii. It is known that for a small connectivity radius, the nodes tend to form clusters.

Fig. 4 shows a comparison of performance between the Schatten Norm minimization (TM) of order $p = 2$ and $p = 4$ and other weight selection algorithms on ER and RGG graphs. For every probability Pr on ER and each radius r on RGG, we averaged across 100 connected graphs with 100 nodes each, and we give the 95% confidence interval for the different curves in the plot. The performance metric on ER graphs (in the left plot) and RGG (in the right plot) is the second largest eigenvalue $\mu(W)$. The smaller μ, the faster is the convergence of the algorithm, as our objective is to minimize this value. We see in Fig. 4 that TM for $p = 2$ and $p = 4$ outperforms other weight selection algorithms on ER by giving lower μ. Similarly on RGG the TM algorithm reaches faster convergence than the other known algorithms even when the graph is well connected (large connectivity radius). However, the larger the degrees of nodes, the higher the complexity of our algorithm. We note that the results are quite interesting because even with the Schatten norm minimization of the least complexity ($p = 2$), nodes were able to find in a distributed way a weight matrix that has faster speed of convergence than the OC algorithm that is considered as a global algorithm which is difficult to implement in a distributed way (but must be solved offline by a centralized unit).

6.2 Transient Performance and Interleaving Optimization and Averaging

The metric that we used so far for evaluation of the different algorithms is the second largest eigenvalue μ which characterizes the asymptotic convergence rate of average protocol (characterizes speed of convergence with respect to a worst case initial condition study). However, this asymptotic metric could mask some important speed differences on different initial distribution of estimates. For this reason, we consider in this section a random initial distribution of nodes' estimates and we define the convergence time to be the number of iterations needed for the error (the distance between the estimates and the actual average) to become smaller than a given threshold. More precisely, we define the normalized error $e(k)$ as

$$e(k) = \frac{||\mathbf{x}(k) - \bar{\mathbf{x}}||_2}{||\mathbf{x}(0) - \bar{\mathbf{x}}||_2},$$

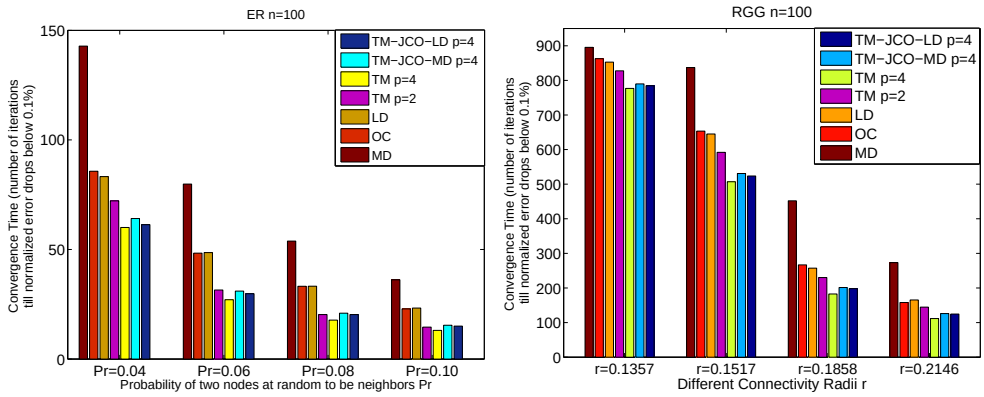


Figure 5: Convergence time of different weight selection algorithms on ER and RGG graphs. TM-JCO-LD $p = 4$ is the joint consensus-optimization algorithm initialized with the LD algorithm's weight matrix and the same for TM-JCO-MD $p = 4$ but initialized with the MD algorithm's one.

where $\bar{\mathbf{x}} = x_{ave}\mathbf{1}$, and the convergence time is the minimum number of iterations after which $e(k) < 0.001$ (note that $e(k)$ is non increasing).

Moreover, the distributed algorithm we described in solving the Schatten norm minimization do not interrupt the averaging protocol, on the contrary, the output of every iteration in the Schatten norm minimization is a feasible weight matrix (expected to have faster converging properties than the previous iteration weight matrix) which can be used freely in the averaging protocol. So we do not run our weight selection algorithm first and then the averaging protocol using the optimal weight matrix, but we interleave their respective iterations. We refer to this technique as the joint consensus–optimization (JCO) procedure. Then each node at every step will improve its own weight according to (15) and use the current weight values to perform the averaging (1). Weights can be initially set according to one of the other existing algorithms like LD or MD. The stepsize is chosen to be $\gamma^{(k)} = \frac{1}{p(1+k)}$ and can be checked that it satisfies the conditions stated in the proposition to guarantee convergence of the optimization. The simulations show that our weight selection algorithm outperforms the other algorithms also in this case. In particular, the two plots in Fig. 5 show the convergence time for various weight selection criteria on ER and RGG graphs respectively. For each of the network topology selected, we averaged the data in the simulation over 100 generated graphs, and for each of these graphs we averaged the convergence time of different algorithms shown in the figures over 20 random initial conditions (the initial conditions were the same for all algorithms). Notice that running at the same time the optimization with consensus gave good results in comparison to LD, MD, and even OC algorithms. We also notice, that the initial feasible solution plays a role in the convergence time of the gradient optimization. In particular, the joint consensus-optimization algorithm converges faster when it is initialized with the LD algorithm's weight matrix.

7 Conclusion

We have proposed in this paper an approximated solution for the fastest distributed linear averaging by solving a Schatten p-norm minimization with some constrains related to the connectivity in the network. We also gave an algorithm to solve this optimization. Unlike previous meth-

ods, our algorithm can be decentralized. Moreover, we show that the distributed algorithm for the Schatten norm minimization can run in parallel with the averaging protocol to have a joint consensus–optimization procedure. Through simulation, we show that this method produces results that are better than other common weight selection heuristics and comparable to the optimal ones which need to be calculated in a centralized way.

References

[1] E. Wei, A. Ozdaglar, and A. Jadbabaie, "A Distributed Newton Method for Network Utility Maximization, I: Algorithm," MIT LIDS, Tech. Rep. 2832, 2011, accepted to IEEE Transaction on Automatic Control.

[2] ——, "A Distributed Newton Method for Network Utility Maximization, I: Convergence," MIT LIDS, Tech. Rep. 2870, 2011, accepted to IEEE Transaction on Automatic Control.

[3] E. Ghadimi, M. Johansson, and I. Shames, "Accelerated gradient methods for networked optimization," in *American Control Conference (ACC), 2011*, 29 2011-july 1 2011, pp. 1668 –1673.

[4] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability," *The Journal of the Operational Research Society*, vol. 49, no. 3, pp. 237–252, 1998.

[5] A. Nedic and A. Ozdaglar, "On the rate of convergence of distributed subgradient methods for multi-agent optimization," in *Proceedings of IEEE CDC*, 2007, pp. 4711–4716.

[6] S. Alouf, G. Neglia, I. Carreras, D. Miorandi, and A. Fialho, "Fitting Genetic Algorithms to Distributed On-line Evolution of Network Protocols," *Elsevier Computer Networks*, vol. 54, no. 18, pp. 3402–3420, Dec. 2010.

[7] B. Kauffmann, F. Baccelli, A. Chaintreau, V. Mhatre, K. Papagiannaki, and C. Diot, "Measurement-Based Self Organization of Interfering 802.11 Wireless Access Networks," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, 2007, pp. 1451–1459.

[8] D. Rabideau, "Fast, rank adaptive subspace tracking and applications," *IEEE Transactions on Signal Processing,*, vol. 44, no. 9, pp. 2229 –2244, September 1996.

[9] W. Ren and R. W. Beard, *Distributed Consensus in Multi-vehicle Cooperative Control: Theory and Applications*, 1st ed. Springer Publishing Company, Incorporated, 2007.

[10] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized Gossip Algorithms," *IEEE Trans. Inf. Theory*, vol. 52, pp. 2508–2530, June 2006.

[11] R. Olfati-saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," in *Proc. of the IEEE*, Jan. 2007.

[12] W. Ren, R. Beard, and E. Atkins, "A survey of consensus problems in multi-agent coordination," in *American Control Conference, 2005. Proceedings of the 2005*, June 2005, pp. 1859 – 1864 vol. 3.

[13] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems and Control Letters*, vol. 53, pp. 65–78, 2004.

[14] C. D. Meyer, Ed., *Matrix analysis and applied linear algebra*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2000.

[15] M. K. Fan and B. Nekooie, "On minimizing the largest eigenvalue of a symmetric matrix," *Linear Algebra and its Applications*, vol. 214, pp. 225–246, 1995.

[16] R. Carli, "Topics on the average consensus problem," Ph.D. dissertation, Università di Padova - Dep. of Information Engineering, July 2008.

[17] Y. Nesterov and A. Nemirovskii, *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM, 1994.

[18] M. Grant and S. Boyd, "CVX: Matlab Software for Disciplined Convex Programming, version 1.21," April 2011.

[19] D. Kempe and F. McSherry, "A decentralized algorithm for spectral analysis," in *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*. ACM, 2004, pp. 561–568.

[20] M. Franceschelli, A. Gasparri, A. Giua, and C. Seatzu, "Decentralized Laplacian eigenvalues estimation for networked multi-agent systems," in *Proc. of the 48th IEEE CDC/CCC 2009*, Dec. 2009, pp. 2717 –2722.

[21] P. Yang, R. Freeman, G. Gordon, K. Lynch, S. Srinivasa, and R. Sukthankar, "Decentralized estimation and control of graph connectivity in mobile sensor networks," in *American Control Conference, 2008*, June 2008, pp. 2678 –2683.

[22] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, March 2004.

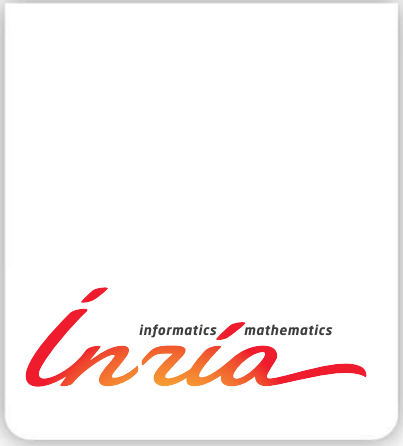
[23] D. Bernstein, *Matrix mathematics: theory, facts, and formulas*. Princeton University Press, 2009.

[24] B. Polyak, *Introduction to Optimization*. New York: Optimization Software, 1987.

[25] D. B. West, *Introduction to Graph Theory (2nd Edition)*. Prentice Hall, Aug. 2000.

[26] D. P. Bertsekas, A. Nedić, and A. E. Ozdaglar, *Convex Analysis and Optimization*. Athena Scientific, 2003.

[27] L. Xiao, S. Boyd, and S. jean Kim, "Distributed average consensus with least-mean-square deviation," *Journal of Parallel and Distributed Computing*, vol. 67, pp. 33–46, 2005.



RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399