A BP-like Distributed Algorithm for Weighted Average Consensus

Zhaorong Zhang[1], Kan Xie[2], Qianqian Cai[2] and Minyue Fu[1]

Abstract— **A novel distributed algorithm designed for weighted average consensus is presented. This algorithm is developed based on the well-known Belief Propagation (BP) algorithm for statistical learning. For a connected network without loops, the algorithm will converge in finite number of iteration and produces a correct average consensus value for every node in the network. For a network with loops, we convert the weighted average problem into an optimization problem with a relaxation factor whose solution approaches to the weighted average with appropriate relaxation factor. A modified algorithm is also proposed for loopy networks and consensus will be reached asymptotically with low computation complexity and fast convergence rate.**

I. Introduction

Distributed average consensus algorithms are highly preferred and have great potential to be applied in statistical learning and distributed signal processing fields. Most of the existing algorithms are based on the Laplacian matrix approach [5]–[15], which performs distributed averaging iteratively using a stochastic matrix with averaging weights. However, correct average consensus can be reached only asymptotically, and the convergence rate is slow, especially for large networks.

In this paper, a novel distributed algorithm for a group of connected agents (nodes) to reach weighted average consensus is to be presented. The distributed algorithm is generalized from the celebrated Gaussian Belief Propagation algorithm. Belief Propagation algorithm, also known as Pearl's BP [1], was designed to compute the marginal probability densities of variables associated with random variables in a large-scale system with a sparse structure. Since its initial conception in 1980s, BP has received increasingly attention in both applications and theoretical studies of its convergence properties. The BP algorithm has been applied in a lot of areas such as statistical learning, communications, estimation, and control systems. It is known in both theoretical studies and simulations that the algorithm has remarkable convergence properties. Hence, we try to develop a new weighted average consensus algorithm by taking the advantages of the BP algorithm.

We connect the weighted average consensus problem with a particular BP algorithm called Gaussian BP designed for studying Gaussian distributions. The weighted average consensus problem turns out to equate to the computation of marginal means of a joint Gaussian distribution in the setting of Gaussian BP. The convergence condition for loopy Gaussian BP has been a main concern of a great number of researchers. There are several significant results towards the understanding of Gaussian BP algorithm. In particular, Weiss and Freeman proved in [2] that Gaussian BP will derive the correct marginal means asymptotically under the condition of the pairwise information matrices being diagonally dominant. In this paper, the assumption of generalised diagonally dominant, which is the relaxed convergence condition put forward by [3], will be used. Similar to Gaussian BP, the proposed algorithm is fully distributed and no central controller or processor is required. In each iteration, information exchange (or message passing) appeared only among neighbouring nodes. For acyclic networks (networks without loops), it takes only d iterations for for each node to acquire the correct average consensus, where d is the diameter of the network graph. When applied to loopy networks, we consider a relaxed average consensus problem whose optimal solution approximates the consensus solution with arbitrary precision by choosing a scaling parameter. We derive a modified average consensus algorithm and show that it has guaranteed exponential convergence to the optimal solution, when applied to a loopy graph directly.

II. Problem Description

Consider an undirected network $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where the node set is $\mathcal{V} = \{1, 2, \ldots, n\}$ and the edge set is $\mathcal{E} = \{(i, j) : i, j \in \mathcal{V}\}$. Each node $i \in V$ is associated with a known variable y_i whose weight is $w_i > 0$. Hence, the weighted average of all of the y_i is,

$$\bar{x} = \frac{\sum_{i\in\mathcal{V}} w_i y_i}{\sum_{i\in\mathcal{V}} w_i}. \tag{1}$$

where the weighted average is denoted by $\bar{x}$, the set of neighbors of node i is represented by $\mathcal{N}_i$ and the cardinality is denoted by $|\mathcal{N}_i|$.

The objective of distributed weighted averaging is to derive an estimate of $\bar{x}$, which is denoted by x_i and will eventually converge to $\bar{x}$ after a number of iterations, on every node $i \in V$ by an iterative algorithm. Notice that when $w_i = 1$ for every node, weighted average $\bar{x}$ will become the standard average of all these $y_i, i = \{1, 2, \ldots, n\}$. A graph is said to be connected if and only if that a path which starts from any node $i \in V$ and will end up with an arbitrary node $j \in V$ can always be found.

[1]School of Electrical Engineering and Computer Science, University of Newcastle, Callaghan, 2308, NSW, Australia.

[2]School of Automation, Guangdong University of Technology and Guangdong Key Laboratory of IoT Information Technology, Guangzhou 510006, China.

This work was supported by the National Natural Science Foundation of China (Grant numbers 61633014 and U1701264).

E-mails: zhaorong.zhang@uon.edu.au; kanxiegdut@gmail.com; qianqian@outlook.com; minyue.fu@newcastle.edu.au.

Denote the diameter of a graph as d which is the length of the longest path in the graph. The loop is a path that starts and ends up at the same node i passes through at least one node $j \neq i$. An acyclic graph means that there are no loops in the connected graph. A loopy graph is defined to be a graph that contains at least one loop.

The following constraints are imposed on the algorithm's complexities of communication, computation and storage to call it *distributed*:

1) Local information exchange: Each node i is allowed to exchange information with each $j \in \mathcal{N}_i$ only once per iteration.
2) Local computation: Each node i's computational load is limited to be at most $O(|\mathcal{N}_i|)$ per iteration.
3) Local storage: Each node i's storage is limited to be at most $O(|\mathcal{N}_i|)$ over all iterations.

Lemma 1. *Under the constraint on local information exchange, a connected undirected graph $\mathcal{G}$ needs a minimum of d iterations to achieve average consensus, where d is the diameter of $\mathcal{G}$.*

Proof: Take nodes i and j to be d hops away from each other. The information at node i must be propagated to node j for node j to accurately compute $\bar{x}$. By the constraint on local information exchange, this process takes at least d iterations.

III. Distributed Algorithm for Average Consensus

In this section, we present the proposed distributed algorithm for average consensus and offer its main property on acyclic graphs.

A. Distributed Algorithm

Let $x_{i\to j}(k)$ denote the information passed from node i to node j at time k, which represents a *scaled estimate* of the average $\bar{x}$ known to node i without using information from node j. Also denote by $s_{i\to j}(k)$ the *scale* passed from node i to node j at time k, which represents the weighted number of nodes used to compute $x_{i\to j}(k)$. We also define two temporary internal variables in node i, $\tilde{s}_i(k)$ and $\tilde{x}_i(k)$. Algorithm 1 is the proposed distributed algorithm for weighted average consensus.

Algorithm 1 (Distributed Algorithm for Average Consensus)

Initialization: For each node i, do: For each $j \in \mathcal{N}_i$, set $x_{i\to j}(0) = y_i, s_{i\to j}(0) = w_i$ and transmit them to node j.

Main loop: At iteration $k = 1, 2, \cdots$, for each node i, compute

$$\tilde{s}_i(k) = w_i + \sum_{j\in\mathcal{N}_i} s_{j\to i}(k-1) \tag{2}$$

$$\tilde{x}_i(k) = w_i y_i + \sum_{j\in\mathcal{N}_i} s_{j\to i}(k-1) x_{j\to i}(k-1) \tag{3}$$

$$\hat{x}_i(k) = \frac{\tilde{x}_i(k)}{\tilde{s}_i(k)}, \tag{4}$$

then for each $j \in \mathcal{N}_i$, compute

$$s_{i\to j}(k) = \tilde{s}_i(k) - s_{j\to i}(k-1) \tag{5}$$

$$x_{i\to j}(k) = \frac{\tilde{x}_i(k) - s_{j\to i}(k-1)x_{j\to i}(k-1)}{\tilde{s}_i(k) - s_{j\to i}(k-1)} \tag{6}$$

and transmit them to node j.

Theorem 1. *Suppose $\mathcal{G}$ is undirected and acyclic with diameter d. Then,*

$$\tilde{s}_i(k) = w_i + \sum_{m\in\mathcal{V}_i(k)} w_m \tag{7}$$

$$\tilde{x}_i(k) = w_i y_i + \sum_{m\in\mathcal{V}_i(k)} w_m x_m \tag{8}$$

for $k = 1, 2, \ldots, d$, where $\mathcal{V}_i(k)$ is the set of nodes in $\mathcal{G}$ at most k hops away from node i (not including node i). Consequently, average consensus is achieved by Algorithm 1 after d iterations, i.e.,

$$\hat{x}_i(k) = \bar{x}, \quad \forall\ k \geq d,\ i \in \mathcal{V}. \tag{9}$$

Sketch of Proof: For any edge $(i,j) \in \mathcal{E}$, we analyze the convergence of $s_{i\to j}(k)$ and $x_{i\to j}(k)$. To do so, we build two disjoint subgraphs $\mathcal{G}_i$ (containing node i) and $\mathcal{G}_j$ (containing node j) of $\mathcal{G}$ by removing (i,j). Notice that $\mathcal{G}_i$ and $\mathcal{G}_j$ are disjoint due to the acyclic nature of $\mathcal{G}$. Also, it is clear that $\mathcal{G}$ is formed by merging $\mathcal{G}_i$, $\mathcal{G}_j$ and edge (i,j). Denote by $\mathcal{W}_i$ (reps. $\mathcal{W}_j$) the set of nodes in $\mathcal{G}_i$ (reps. $\mathcal{G}_j$). We see from (2)-(6) that $s_{i\to j}(k)$ and $x_{i\to j}(k)$ are constructed using the information in $\mathcal{G}_i$ only because $s_{j\to i}(k-1)$ and $x_{j\to i}(k-1)$ (representing the information flow from node j to node i) get removed in each iteration. That is, (2)-(5) can be rewritten as

$$s_{i\to j}(k) = w_i + \sum_{m\in\mathcal{N}_i\backslash\{j\}} s_{m\to i}(k-1)$$

$$x_{i\to j}(k) = w_i y_i + \sum_{m\in\mathcal{N}_i\backslash\{j\}} s_{m\to i}(k-1)x_{m\to i}(k-1)$$

which shows that the information $(s_{j\to i}(k-1), x_{j\to i}(k-1))$ is not used in computing $(s_{i\to j}(k), x_{j\to i}(k))$. Similarly, $s_{j\to i}(k)$ and $x_{j\to i}(k)$ are constructed using the information in $\mathcal{G}_j$ only.

Recall that at Initialization ($k = 0$), $s_{i\to j}(0) = w_i$ is set. Consider the case of $k = 1$. From (2) and (5), we see that $s_{i\to j}(1)$ contains all $s_{m\to i}(0)$ for all the neighbouring nodes m, except node j. Following the definition of $\mathcal{V}_i(k)$, we have $\mathcal{V}_i(1) = \mathcal{N}_i$. It follows that

$$s_{i\to j}(1) = w_i + \sum_{m\in\mathcal{N}_i\backslash\{j\}} s_{m\to i}(0) = w_i + \sum_{m\in\mathcal{V}_i(1)\backslash\mathcal{W}_j} w_m.$$

Repeating the above process for $k = 2, 3, \ldots$, we can get, for any k,

$$s_{i\to j}(k) = w_i + \sum_{m\in\mathcal{V}_i(k)\backslash\mathcal{W}_j} w_m.$$

Next, consider $\tilde{s}_i(k)$ and $\tilde{x}_i(k)$. Notice from (5) that $\tilde{s}_i(k) = s_{i\to j}(k) + s_{j\to i}(k-1)$. Also notice that node j

is 1 hop away from node i, meaning that all the nodes in $\mathcal{G}_j$ which are $k-1$ hops away from node j are k hops away from node i, when considering $\mathcal{G}$. That is,

$$(\mathcal{V}_i(k)\backslash\mathcal{W}_j)\cup(\mathcal{V}_j(k-1)\backslash\mathcal{W}_i)\cup\{j\}=\mathcal{V}_i(k).$$

It follows that

$$\tilde{s}_i(k)=w_i+\sum_{m\in\mathcal{V}_i(k)}w_m,$$

which is (7). The equation (8) is shown in the same way. It follows that $\tilde{x}_i(d)$ is the weighted sum of all the nodes in $\mathcal{G}$ and $\tilde{s}_i(d)$ is the sum of their weights, resulting in $\hat{x}_i(d)=\bar{x}$ for each node i. Proceeding the analysis above further with $k>d$, we can show that $\hat{x}_i(k)$ is unchanged. The details are omitted due to space limit.

IV. Distributed Average Consensus for Loopy Graphs

In this section, we modify the distributed average consensus algorithm, Algorithm 1, to make it suitable for direct application to loopy graphs. This is done by relaxing the strict requirement that all the nodes must reach exact consensus. More precisely, for a given undirected and connected graph $\mathcal{G}=\{\mathcal{V},\mathcal{E}\}$ as before (but cyclic), we consider the following optimization problem:

$$\min_x\sum_{i\in\mathcal{V}}w_i\|x_i-y_i\|^2+\gamma\sum_{(i,j)\in\mathcal{E}}\|x_i-x_j\|^2, \tag{10}$$

where w_i and y_i are as given before, $x=\text{col}\{x_1,x_2,\dots,x_n\}$ and $\gamma>0$ is a large weighting (or penalty) parameter to be tuned. It is intuitive to see that as $\gamma\to\infty$, all the x_i will become the same due to the connectedness of $\mathcal{G}$, and the solution to (10) will become the solution to

$$\bar{x}=\arg\min_{x_0}\sum_{i\in\mathcal{V}}w_i\|x_0-y_i\|^2,$$

which can be easily verified to be identical to (1). The relaxation mentioned above is to solve (10) instead of (1) for a sufficiently large γ.

We first give its centralized optimal solution below. Denote $y=\text{col}\{y_1,y_2,\dots,y_n\}$, $W=\text{diag}\{w_1,w_2,\dots,w_n\}$, and define the $n\times n$ Laplacian matrix $L=[\ell_{ij}]$ as

$$\ell_{ij}=\begin{cases}|\mathcal{N}_i|, & i=j\\ -1, & i\neq j,(i,j)\in\mathcal{E}\\ 0, & i\neq j,(i,j)\notin\mathcal{E}\end{cases}$$

Lemma 2. *The optimal solution to (10) is given by*

$$x^\star=(\gamma L+W)^{-1}Wy. \tag{11}$$

Proof: It is easy to verify that the objective function in (10) can be rewritten as $(x-y)^TW(x-y)+x^T(\gamma L)x$. Differentiating it with respect to x and setting it to zero gives

$$W(x-y)+\gamma Lx=0.$$

Solving it gives the solution (11). This solution is optimal because L is symmetric and positive semi-definite, ensuring that $\gamma L+W$ is symmetric and positive definite, which in turn ensuring that the objective function is strictly convex.

The modified Algorithm 1 is presented in Algorithm 2. For notational convenience, we denote $f(w)=\gamma w/(\gamma+w)$. We first present a result on Algorithm 2 for acyclic graphs, similar to Theorem 1 on Algorithm 1.

Algorithm 2 (Modified Algorithm for Average Consensus)

```
Initialization: For each node $i$, do: For each $j\in\mathcal{N}_i$, set $x_{i\to j}(0)=y_i, s_{i\to j}(0)=f(w_i)$ and transmit them to node $j$.
Main loop: At iteration $k=1,2,\cdots$, for each node $i$, compute
    $\tilde{s}_i(k)=w_i+\sum_{j\in\mathcal{N}_i}s_{j\to i}(k-1)$    (12)
    $\tilde{x}_i(k)=w_iy_i+\sum_{j\in\mathcal{N}_i}s_{j\to i}(k-1)x_{j\to i}(k-1)$    (13)
    $\hat{x}_i(k)=\frac{\tilde{x}_i(k)}{\tilde{s}_i(k)}$,    (14)
then for each $j\in\mathcal{N}_i$, compute
    $s_{i\to j}(k)=f(\tilde{s}_i(k)-s_{j\to i}(k-1))$    (15)
    $x_{i\to j}(k)=\frac{\tilde{x}_i(k)-s_{j\to i}(k-1)x_{j\to i}(k-1)}{\tilde{s}_i(k)-s_{j\to i}(k-1)}$    (16)
and transmit them to node $j$.
```

Theorem 2. *Suppose that the graph $\mathcal{G}$ is undirected, connected and acyclic with diameter d. Then, running Algorithm 2 yields that*

$$\hat{x}(k)=x^\star,\ \ \forall k\geq d \tag{17}$$

where $\hat{x}(k)=\text{col}\{\hat{x}_1(k),\hat{x}_2(k),\dots,\hat{x}_n(k)\}$ and $x^\star$ is in (11).

Sketch of Proof: The proof follows the same general idea as in the proof of Theorem 1. But because the problem dealt with in (11) is no longer a simple weighted average consensus (due to the fact that $f(w)=\gamma w/(\gamma+w)$ is slightly different from w), it is notationally more cumbersome to keep track of the variables $s_{i\to j}(k)$ and $x_{i\to j}(k)$. The details are omitted due to space limit.

Next, we show that Algorithm 2 indeed applies to loopy graphs directly and still enjoys the convergence of $\hat{x}_i(k)\to x_i^\star$ as $k\to\infty$, for all $i\in\mathcal{V}$.

First, we point out an obvious fact of (15) that $s_{i\to j}(k)<\gamma$ because $f(w)=\gamma w/(\gamma+w)<\gamma$, i.e., $s_{i\to j}(k)$ is uniformly bounded.

To study the convergence of Algorithm 2 for loopy graphs, we construct an *unwrapped tree* with *depth* $t>0$ for a loopy graph $\mathcal{G}$ [2]. Take an arbitrary node, say node 1, to be the root and then iterate the following procedure t times:

- Find all leaves of the tree (start with the root);
- For each leaf, find all the nodes in the loopy graph that neighbor this leaf node, except its parent node in the

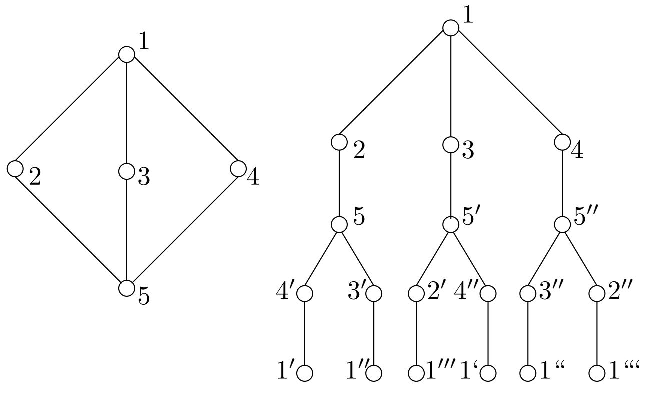

Fig. 1. Left: A loopy graph. Right: The unwrapped tree around node 1 with 4 layers ($t = 4$)

tree, and add all these node as the children to this leaf node.

The variables and weights for each node in the unwrapped tree are copied from the corresponding nodes in the loopy graph. It is clear that taking each node as root node will generate a different unwrapped tree. Fig. 1 shows the unwrapped tree around node 1 for the loopy graph. Note, for example, that nodes $1', 1'', 1''', 1‘, 1“, 1“‘$ all carry the same values y_1 and w_1.

List the nodes in the unwrapped tree in *breadth first* order, by starting from the root node, followed by the first layer, then the second layer, etc. Denote the unwrapped quantities using $\breve{}$ and denote the unwrapped tree as $\breve{\mathcal{G}}_t$. Then, as shown in [2], the unwrapped quantities and the original ones are related through a matrix O:

$$\breve{y} = Oy; \quad \breve{w} = Ow \tag{18}$$

Each row of O contains only 1 and the rest are all 0.

Consider the optimization problem (10) for $\breve{\mathcal{G}}_t$ (with unwrapped quantities). By Lemma 2, its optimal solution is given by

$$\breve{x}^\star = (\gamma\breve{L} + \breve{W})^{-1}\breve{W}\breve{y}, \tag{19}$$

where $\breve{L}$ is the Laplacian matrix for $\breve{\mathcal{G}}_t$.

We introduce two *key* lemmas below.

Lemma 3. *The optimal quantity $\breve{x}_1^\star$ (the first element of (19)) for $\breve{\mathcal{G}}_t$ coincides with $\hat{x}_1(t)$ (the result of (14) for node 1 after running Algorithm 2 on $\mathcal{G}$ for t iterations), i.e.,*

$$\hat{x}_1(t) = \breve{x}_1^\star \tag{20}$$

Proof omitted due to space limit.

Lemma 4. *The following equations hold for the depth-t unwrapped graph $\breve{\mathcal{G}}_t$ of $\mathcal{G}$:*

$$OW = \breve{W}O, \tag{21}$$

$$OL = \breve{L}O + E, \tag{22}$$

$$\breve{x}_1^\star = x_1^\star + e_1^T(\gamma\breve{L} + \breve{W})^{-1}Ex^\star. \tag{23}$$

In the above, e_1 is the column vector with 1 in the first element and zero everywhere else, and E is an error matrix with zero in all rows except for the last L_t rows, where L_t is the number of its depth-t leaf nodes in $\breve{\mathcal{G}}_t$.

Proof omitted due to space limit.

Define the error

$$z_1^\star = \breve{x}_1^\star - x_1^\star. \tag{24}$$

and denote $r = \gamma\breve{W}^{-1}Ex^\star$. Note that $\breve{W}$ is diagonal, hence only the last L elements of r are nonzero, corresponding to the depth-t leaf nodes. Then, (23) can be rewritten as

$$z_1^\star = e_1^T(\gamma\breve{L} + \breve{W})^{-1}\breve{W}r. \tag{25}$$

Next, we give the crucial result that $z_1^\star \to 0$ exponentially fast as the depth $t \to \infty$.

Lemma 5.

$$\|z_1^\star\| \leq \eta^{t-1}r_{\max}, \tag{26}$$

where $0 < \eta < 1$ is given by

$$\eta = \max_{i\in\mathcal{V}} \frac{(|\mathcal{N}_i| - 1)\gamma}{w_i + (|\mathcal{N}_i| - 1)\gamma}. \tag{27}$$

Proof omitted due to space limit.

Combining the results above, we obtain our main result on the convergence of Algorithm 2.

Theorem 3. *Suppose the graph $\mathcal{G}$ is undirected and connected. Then, running Algorithm 2 for $k \geq 1$ iterations yields that*

$$\|\hat{x}_i(k) - x_i^\star\| \leq \eta^{k-1}r_{\max}, \quad \forall\ i \in \mathcal{V},\ k = 0, 1, 2, \ldots, \tag{28}$$

where $0 < \eta < 1$ is given by (27) and $r_{\max}$ is a constant independent of k.

Proof: Taking any node $i \in \mathcal{V}$ as the root node and form the unwrapped tree $\breve{\mathcal{G}}_k$ around node i. Using Lemma 5 by treating node i as node 1, we get that $\|z_i^\star\| = \|\breve{x}_i^\star - x_i^\star\| \leq \eta^{k-1}r_{\max}$. Invoking Theorem 2 on the tree $\breve{\mathcal{G}}_k$, we get $\breve{x}_i^\star = \hat{x}_i(k)$ for the original graph $\mathcal{G}$, thus (28) holds.

V. Illustrative Examples

To verify the performance of the modified distributed algorithm, there are three examples being applied to in this section. The first one is a 13-node acyclic graph in Fig. 2 whose diameter is 5. The second example is a 9-node loopy graph in Fig. 5. The final one is a random 1000-node loopy graph in Fig. 8. As mentioned in the last section, the weights of each node $w_i = 1, i = 1, 2, \ldots, 13$, the relaxation factor $\gamma = 1000$ and $y_i = i, i = 1, 2, \ldots, 13$. For comparison, two distributed iterative methods and simulation results of which have also been presented. The distributed iteration of the first Laplacian matrix based algorithm [4] is as followed,

$$\hat{x}(k+1) = \hat{x}(k) - \alpha L\hat{x}(k), \quad \hat{x}(0) = y, \tag{29}$$

It has been proved in [4] that with $\alpha = 1/\lambda_{\max}(L)$ where $\lambda_{\max}$ is the maximal eigenvalue of Laplacian matrix L, the convergence rate appears to be the fastest.

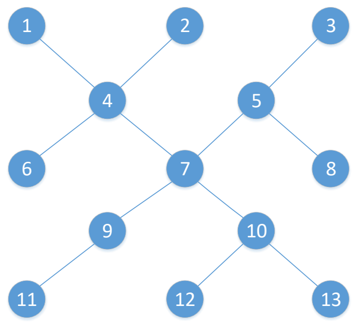

Fig. 2. A 13-node Acyclic Graph

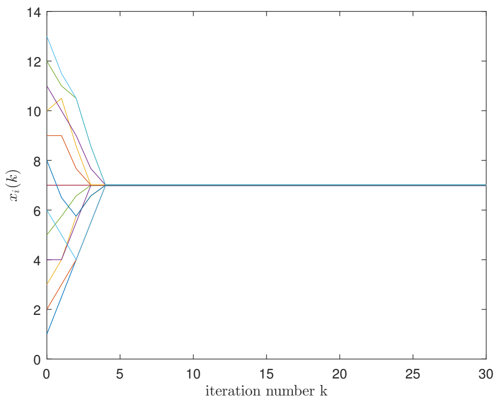

Fig. 3. Convergence of the Proposed Algorithm in an Acyclic Graph

Notice that the optimal solution to (11) can be regarded as the solution to the linear matrix equation

$$(\gamma L + W)x^{\star} = Wy. \tag{30}$$

For the example in figure 3, it can be seen from the simulation result that our proposed distributed algorithm converges after 5 iterations, which is the diameter of figure 2. By contrast, the method by [4] needs to take approximately 100 iterations to reach average consensus in figure 4, which is considerably slower.

For the example in figure 5, Algorithm 2 reaches consensus after around 15 iterations in figure 6, whereas in figure 7 it takes approximately 25 iterations to converge for [4].

For the example in figure 8, Algorithm 2 converges roughly after 100, as shown in figure 9. This is significantly faster than [4], shown in figure 10.

VI. Conclusions

With the inspiration of Gaussian BP algorithm, a novel distributed algorithm for nodes to reach weighted average consensus in large-scale systems has been presented. The algorithm owns all of the premium properties of the BP algorithm. Firstly, in acyclic graphs, the algorithm is able to produce correct estimate of weighted average at each node after d iterations, which is the graph diameter. Next, using a

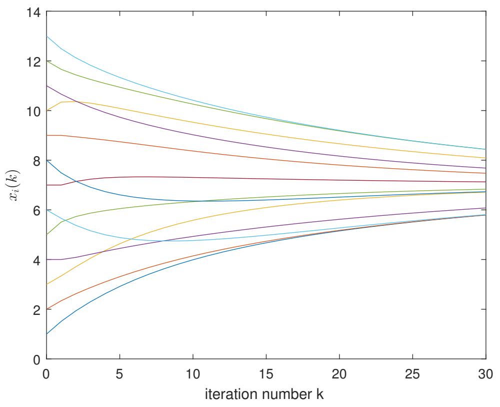

Fig. 4. Convergence of the Algorithm [4] in an Acyclic Graph

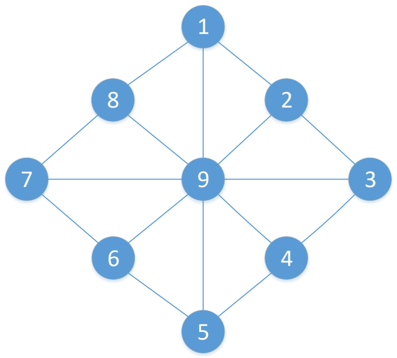

Fig. 5. A 9-node Loopy Graph

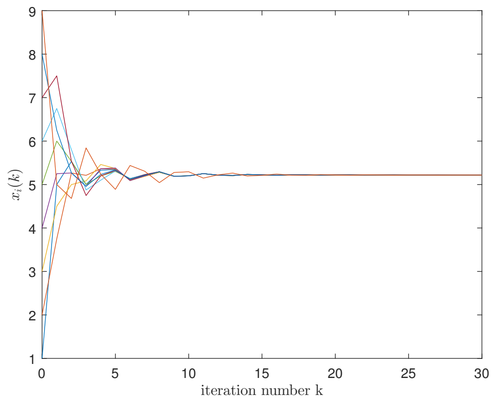

Fig. 6. Convergence of the Algorithm 2 in a 9-node Loopy Graph

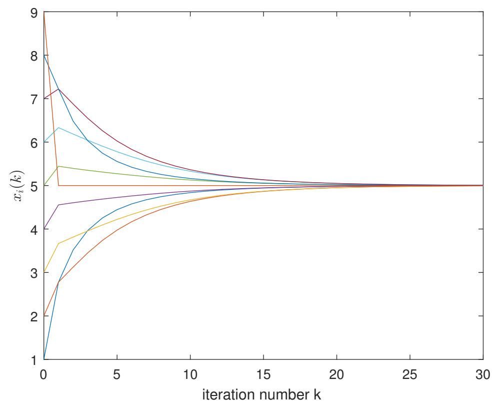

Fig. 7. Convergence of the Algorithm [4] in a 9-node Loopy Graph

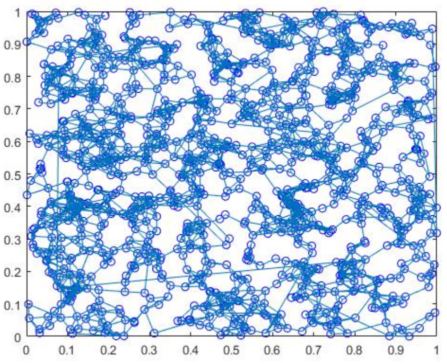

Fig. 8. A 1000-node Loopy Graph

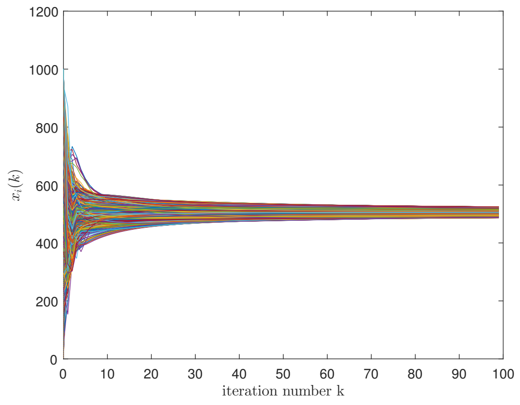

Fig. 9. Convergence of the Algorithm 2 in a 1000-node Loopy Graph

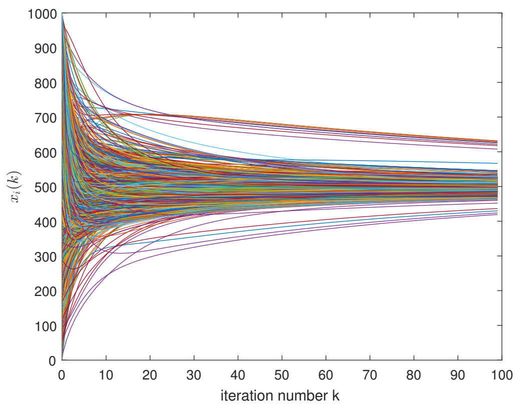

Fig. 10. Convergence of the Algorithm [4] in a 1000-node Loopy Graph

modified algorithm, nodes will reach weighted average consensus asymptotically in loopy graphs. Additionally, when applied to loopy networks, the exponential convergence rate of our algorithm has also been illustrated via simulation. We believe this novel distributed algorithm has the potential in contributing to distributed estimation and machine learning fields.

References

[1] J. Pearl, "Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference", Morgan Kaufmann, San Francisco, 1988.

[2] Y. Weiss, W.T.Freeman, "Correctness of Belief Propagation in Gaussian Graphical Models of Arbitrary Topology", Mitsubishi Electric Research Laboratories, 1999.

[3] D. M. Malioutov, J. Johnson, and A. Wilsky, "Walk-sums and Belief Propagation in Gaussian Graphical Models," J. Mach. Learn. Res., vol.7, no. 1, pp. 2031-2064, 2006.

[4] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," Systems and Control Letters, 53: 65-78, 2004.

[5] S. Mou, J. Liu, and A. S. Morse, "A Distributed Algorithm for Solving a Linear Algebraic Equation," IEEE Trans. on Autom. Control, vol. 60, no. 11, pp. 2863-2878, Nov. 2015.

[6] S. Sundaram and C. N. Hadjicostis, "Finite-time distributed consensus in graphs with time-invariant topologies," American Control Conf., 2007.

[7] G. Shi, B. Li, M. Johansson and K. H. Johansson, "Finite-time convergent Gossiping," IEEE Transactiosn on Networking, to appear.

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

[9] Q. Hui, W. M. Haddad and S. P. Bhat, "Finite-time semistability and consensus for nonlinear dynamical networks," IEEE Trans. Autom. Control, vol. 53, no. 8, pp. 1887-1900, 2008.

[10] S. Li,H. Du and X. Lin, "Finite-time consensus algorithm for multi-agent systems with double-integrator dynamics", Automatica, vol. 47, iss. 8,pp. 1706-1712, Aug. 2011.

[11] L. Xiao, S. Boyd and S. Kim, "Distributed average consensus with least-mean-square deviation", Journ. of Paral. and Distr. Compu., vol. 67, iss. 1, pp. 33-46, Jan. 2007.

[12] P. Blimana, G. Ferrari-Trecate, "Average consensus problems in networks of agents with delayed communications", Automatica, vol. 44, iss. 8, pp. 1985-1995, Aug. 2008.

[13] P. Lin, Y. Jia, "Average consensus in networks of multi-agents with both switching topology and coupling time-delay", Physica A: Stati. Mecha. and its Appli., vol. 387, iss. 1, pp. 303-313, Jan. 2008.

[14] T. Li, M. Fu, L. Xie and J. Zhang, "Distributed Consensus with Limited Communication Data Rate," IEEE Transactions on Automatic Control, vol. 56, no. 2, pp. 279-292, 2011.

[15] S. Mou, J. Liu and A. S. Morse, "A distributed algorithm for solving a linear algebraic equation," IEEE Transactions on Automatic Control, vol. 60, no. 11, pp. 2863-2878, 2015.