

# DISTRIBUTED NETWORK DECOMPOSITION: A PROBABILISTIC GREEDY APPROACH

Yanbing Zhang and Huaiyu Dai  
 Department of Electrical and Computer Engineering  
 NC State University  
 Raleigh, NC USA  
 {yzhang, Huaiyu\_Dai}@ncsu.edu

**Abstract**—In this paper, we propose a novel distributed network decomposition algorithm with the aid of the factor graph model and the max-product algorithm, which aims to achieve minimum cut weight. Its effectiveness is testified for general graph partition as well as distributed inference in wireless networks. Our algorithm is fully distributed, simple in computation, and readily extensible, thus providing a potentially powerful, data-independent clustering scheme for a wide range of data processing and networking applications.

**Keywords** - Network decomposition; distributed clustering; factor graph; max-product algorithm

## I. INTRODUCTION

Identifying substructures in a large graph (network), also called clustering, graph partitioning or network decomposition, plays a key role in various applications, such as pattern recognition, parallel computation, and exploratory data analysis in scientific and medical study. While heuristic approaches abound [1], many of them suffer from lack of solid theoretical justification, raising concerns on their optimality and applicability to wider applications. One exception is the spectral clustering algorithm, which is shown to solve a relaxation of the NP-hard min-cut problem subject to cluster size balancing constraint [2][3]. Most applications of spectral clustering are centralized (both in terms of knowledge and computation), not feasible for large-scale networked systems. A partial solution in a distributed environment is provided in [4], where the first  $K$  principle eigenvectors of a symmetric weighted adjacency matrix are calculated, with poly( $K$ ) computation, communication and storage requirement per node per iteration. Depending on network instances and applications, such requirement may still be demanding (if  $K$  needs to scale with the network size).

Distributed clustering attracts much attention in wireless networks [5], but the approaches there give little consideration to the “semantics” or probabilistic dependence between the nodes. More recently, a distributed clustering algorithm, affinity propagation (AP), was proposed in [6] for data processing. As a distributed variant of the k-medoids clustering method, affinity propagation formulates message-passing rules between the data points, until a set of exemplars and corresponding clusters gradually emerge. Inspired by this approach, we propose a fully distributed network decomposition algorithm, whose objective is instead to solve the above mentioned NP-hard min-cut problem subject to cluster size balancing constraint. We exploit the factor graph model and the max-product algorithm to derive the message passing forms for our algorithm, which upon convergence produces some candidate solutions. Then we take a greedy approach to gradually approach the optimal solution.

The rest of this paper is organized as follows. Section II formulates the problem and briefly introduces the factor graph and the max-product algorithm. Our novel distributed clustering algorithm is discussed in Section III. Section IV presents some supporting simulation results. Finally, concluding remarks are given in Section V.

## II. PROBLEM FORMULATION

### A. Problem Statement

Consider an  $N$ -node undirected graph  $G = (V, E)$  with a weight

matrix  $W \in R^{N \times N}$ , where the nonnegative entry  $w_{ij}$  denotes a general weight on the edge  $(i, j) \in E$  (such as communication cost, distance or similarity metrics depending on specific applications).  $W$  does not need to be symmetric. A  $K$ -cluster decomposition of the network is a collection of disjoint vertex subsets  $P = \{C_1, \dots, C_K\}$  satisfying  $C_1 \cup \dots \cup C_K = V$ . The cut weight associated with the decomposition  $P$  is the sum of the edge weights connecting vertices in different clusters:

$$\Omega(P) = \sum_{\substack{(i,j) \in E, i \in C_i \\ j \in C_j, C_i \neq C_j}} w_{ij}. \quad (1)$$

A classical problem of network decomposition is to find a partition  $P$  that minimizes  $\Omega(P)$  under some regulations on the balancing of cluster sizes so as to avoid trivial solutions. In this work, we focus on the  $L$ -bounded min-cut decomposition, with no cluster in partition having more than  $L$  vertices.

Define an  $N \times K$  indicator matrix  $B = [b_{ik}]$  with the entry  $b_{ik}$  given by:

$$b_{ik} = \begin{cases} 1 & i \in C_k \\ 0 & i \notin C_k, \end{cases} \quad (2)$$

for  $i = 1, \dots, N$  and  $k = 1, \dots, K$ . Then the total weight  $\Omega(P)$  can be expressed in the form of

$$\Omega(P) = \mathbf{1}_N^T W \mathbf{1}_N - \text{trace}(B^T W B), \quad (3)$$

where  $\mathbf{1}_N$  is the  $N \times 1$  vector with all entries equal to one.

The problem of interest can be equivalently formulated as the following optimization problem:

$$\begin{aligned} & \text{maximize } \text{trace}(B^T W B) \\ & \text{subject to } b_{ik} \in \{0, 1\}, \forall i, k \\ & B \text{ column orthogonal} \\ & \|B\|_F = \sqrt{N} \\ & B^T \mathbf{1}_N \leq L \mathbf{1}_K, \end{aligned} \quad (4)$$

where the objective corresponds to the total intra-cluster weights,  $\|\cdot\|_F$  denotes the Frobenius norm, and the inequality in the last constraint is understood element-wise. Essentially the last three constraints respectively indicate that every vertex only belongs to one cluster, all vertices are clustered, and no cluster contains more than  $L$  vertices.

### B. Factor Graph and Max-Product Algorithm

A factor graph [7] is a bi-partite graph consisting of a set of  $N$  variable nodes  $\mathbf{X} = \{X_1, \dots, X_N\}$  and  $M$  function nodes  $\{f_1, \dots, f_M\}$ . For a variable node  $X_n$ ,  $\Gamma_n$  denotes the set of indices of its neighboring function nodes;  $\Gamma_m$  is similarly defined for a function node  $f_m$ , which only depends on  $X_{\Gamma_m}$ . The whole factor graph represents (perhaps a scaled version of) a global function which is the product of all its function nodes

$$F(\mathbf{X} = \mathbf{x}) = \prod_{m=1}^M f_m(X_{\Gamma_m}). \quad (5)$$

The max-product algorithm, a variant of the sum-product algorithm, involves the exchange of messages between variable nodes and function nodes to obtain the (pseudo)-mode of (5) [7]. The message sent from a variable node  $X_n$  to a function node  $f_m$  is a function of  $x_n$  and reflects the current inference (“belief”) about  $X_n$  given evidence from all its neighbors  $f_{\Gamma_n}$  except  $f_m$ , taking an element-wise product form as

$$v_{n \rightarrow m}(x_n) = \prod_{m' \in \Gamma_n \setminus m} \mu_{m' \rightarrow n}(x_n). \quad (6)$$

The message sent from a function node  $f_{m'}$  to variable node  $X_n$ ,  $\mu_{m' \rightarrow n}$ , takes the form

$$\mu_{m' \rightarrow n}(x_n) = \max_{X_{(\Gamma_{m'} \setminus n)}} f_{m'}(x_{\Gamma_{m'}}) \prod_{n' \in \Gamma_{m'} \setminus n} v_{n' \rightarrow m'}(x_{n'}), \quad (7)$$

which reflects the current belief of  $f_{m'}$  about  $X_n$  summarizing messages from all its other neighbors. Finally, the best configuration  $\mathbf{x}^*$  is formed by

$$\mathbf{x}_n^* = \arg \max_{x_n} \prod_{m \in \Gamma_n} \mu_{m \rightarrow n}(x_n) = \arg \max_{x_n} \mu_{m \rightarrow n}(x_n) v_{n \rightarrow m}(x_n). \quad (8)$$

### III. DISTRIBUTED CLUSTERING ALGORITHM

To implement the network decomposition problem (4) in a distributed way, we use alternate binary variables  $b_{ij} \in \{0,1\}$  to indicate the cluster assignment. For  $i, j = 1, \dots, N$ ,  $b_{ij} = 1$  means node  $i$  and node  $j$  are in the same cluster, and  $b_{ij} = 0$  otherwise ( $b_{jj}$  is always 1). An additional feature of this setup is that, in contrast to most existing clustering algorithms, no cluster heads are imposed; nodes self-organize into clusters satisfying the objective. If needed, cluster head election can be conducted at a later stage.

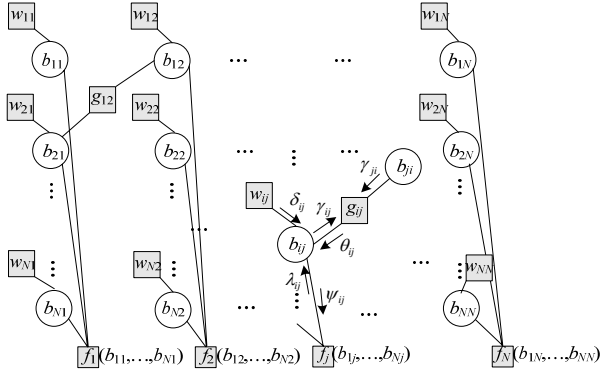


Figure 1. Factor graph structure for distributed clustering

A factor graph for our distributed clustering problem is constructed as shown in Figure 1, where each variable node  $b_{ij}$  is paired with a function node related to the corresponding edge weight, evaluated as  $\exp\{b_{ij}w_{ij}\}$ . We further add function nodes  $f_j(b_{1j}, \dots, b_{Nj})$ ,  $j = 1, \dots, N$  to address the last constraint in (4):

$$f_j(b_{1j}, \dots, b_{Nj}) = \begin{cases} 1 & \text{if } \sum_{i=1}^N b_{ij} \leq L, \\ 0 & \text{otherwise.} \end{cases} \quad (9)$$

We also impose function nodes  $g_{ij}(b_{ij}, b_{ji})$ ,  $i, j = 1, \dots, N$ , to guarantee that  $b_{ij}$  and  $b_{ji}$  always coincide, which, together with our clustering process below, ensures that each node belongs to one and only one cluster:

$$g_{ij}(b_{ij}, b_{ji}) = \begin{cases} 1 & \text{if } b_{ij} = b_{ji}, \\ 0 & \text{otherwise.} \end{cases} \quad (10)$$

This configuration leads to a global function

$$\begin{aligned} F(\{b_{ij}\}_{i,j=1}^N) &= \prod_{i=1}^N \prod_{j=1}^N e^{b_{ij}w_{ij}} \sqrt{g_{ij}(b_{ij}, b_{ji})} \prod_{j=1}^N f_j(b_{1j}, \dots, b_{Nj}) \\ &= \exp \left\{ \sum_{i=1}^N \sum_{j=1}^N b_{ij}w_{ij} + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \log g_{ij}(b_{ij}, b_{ji}) + \sum_{j=1}^N \log f_j(b_{1j}, \dots, b_{Nj}) \right\} \end{aligned} \quad (11)$$

where the first term in the exponent corresponds to our objective function in (4), while the second and third term are intended for excluding solutions violating the constraints. In the following, we exploit the max-product algorithm to derive message-passing rules that lead to the best inference of  $\{b_{ij}\}_{i,j=1}^N$ , maximizing (the exponent of) (11).

As shown in Figure 1, five types of messages are passed between variable nodes and function nodes, where messages outgoing from variable nodes are easy to obtain following (6):

$$\gamma_{ij}(b_{ij}) = \begin{cases} \delta_{ij}(0)\lambda_{ij}(0) & b_{ij} = 0, \\ \delta_{ij}(1)\lambda_{ij}(1) & b_{ij} = 1; \end{cases} \quad (12)$$

and

$$\psi_{ij}(b_{ij}) = \begin{cases} \delta_{ij}(0)\theta_{ij}(0) & b_{ij} = 0, \\ \delta_{ij}(1)\theta_{ij}(1) & b_{ij} = 1. \end{cases} \quad (13)$$

Since these messages are binary, they can be efficiently represented by the corresponding scalar ratios:

$$\gamma_{ij} \triangleq \frac{\gamma_{ij}(1)}{\gamma_{ij}(0)} = \frac{\delta_{ij}(1)\lambda_{ij}(1)}{\delta_{ij}(0)\lambda_{ij}(0)} \triangleq \delta_{ij}\lambda_{ij}, \quad (14)$$

$$\psi_{ij} \triangleq \frac{\psi_{ij}(1)}{\psi_{ij}(0)} = \frac{\delta_{ij}(1)\theta_{ij}(1)}{\delta_{ij}(0)\theta_{ij}(0)} \triangleq \delta_{ij}\theta_{ij}. \quad (15)$$

By (7), the message from the weight function node  $w_{ij}$  is simply  $\delta_{ij}(b_{ij}) = e^{b_{ij}w_{ij}}$ , normalized as  $\delta_{ij} \triangleq \delta_{ij}(1) / \delta_{ij}(0) = e^{w_{ij}}$ .

Messages outgoing from the function nodes  $g$  are given by:

$$\theta_{ij}(b_{ij}) = \max_{b_{ji}} g_{ij}(b_{ij}, b_{ji}) \gamma_{ji}(b_{ji}) = \begin{cases} \gamma_{ji}(0) & b_{ij} = 0, \\ \gamma_{ji}(1) & b_{ij} = 1. \end{cases} \quad (16)$$

So we have

$$\theta_{ij} \triangleq \frac{\theta_{ij}(1)}{\theta_{ij}(0)} = \frac{\gamma_{ji}(1)}{\gamma_{ji}(0)} = \gamma_{ji}. \quad (17)$$

For messages  $\lambda_{ij}$  outgoing from the function nodes  $f$ , we only need to consider the cases  $i \neq j$ . Consider incoming messages of  $f_j$ ,  $\{\psi_{i'j}\}$  such that  $i' \neq i \neq j$ , and sort them in the decreasing order as  $\psi_{i'j}^{(1)} \geq \psi_{i'j}^{(2)} \geq \dots \geq \psi_{i'j}^{(L-1)} \geq \dots$ . If  $\psi_{i'j}^{(L-1)} > 1$ ,

$$\lambda_{ij}(b_{ij}) = \max_{b_j} \dots \max_{b_{(i-1)j}} \max_{b_{(i+1)j}} \dots \max_{b_{Nj}}$$

$$\left[ f_j(b_{1j}, \dots, b_{(i-1)j}, b_{ij}, b_{(i+1)j}, \dots, b_{Nj}) \prod_{i': i' \neq i} \psi_{i'j}(b_{i'j}) \right]$$

$$= \begin{cases} \prod_{l=1}^{L-1} \psi_{i'j}^{(l)} \prod_{i': i' \neq i} \psi_{i'j}(0) & b_{ij} = 0, \\ \prod_{l=1}^{L-2} \psi_{i'j}^{(l)} \prod_{i': i' \neq i} \psi_{i'j}(0) & b_{ij} = 1. \end{cases} \quad (18)$$

On the other hand, if  $\psi_{i'j}^{(L-1)} \leq 1$ ,

$$\lambda_{ij}(b_{ij}) = \begin{cases} \prod_{l=1}^{L'} \psi_{i'j}^{(l)} \prod_{i': i' \neq i} \psi_{i'j}(0) & b_{ij} = 0, \\ \prod_{l=1}^{L'} \psi_{i'j}^{(l)} \prod_{i': i' \neq i} \psi_{i'j}(0) & b_{ij} = 1. \end{cases} \quad (19)$$

Combine (18) and (19), we get

$$\lambda_{ij} = \frac{\lambda_{ij}^{(1)}}{\lambda_{ij}^{(0)}} = \begin{cases} 1/\psi_{i'j}^{(L-1)} & \psi_{i'j}^{(L-1)} > 1, \\ 1 & \psi_{i'j}^{(L-1)} \leq 1. \end{cases} \quad (20)$$

and (c.f. (15), (17) and (14))

$$\psi_{ij} = \delta_{ij}\theta_{ij} = \delta_{ij}\gamma_{ji} = \delta_{ij}\delta_{ji}\lambda_{ji}. \quad (21)$$

Defining  $u_{ij} = \log \psi_{ij}$ ,  $v_{ij} = \log \lambda_{ij}$ , the above equations can be rewritten as

$$v_{ij} = \begin{cases} -u_{i'j}^{(L-1)} & u_{i'j}^{(L-1)} \geq 0 \\ 0 & u_{i'j}^{(L-1)} < 0 \end{cases} = \min\{0, -u_{i'j}^{(L-1)}\}, \quad (22)$$

$$u_{ij} = w_{ij} + w_{ji} + v_{ji} = \min\{w_{ij} + w_{ji}, w_{ij} + w_{ji} - u_{i'j}^{(L-1)}\}. \quad (23)$$

According to the max-product algorithm (see (8)),  $b_{ij}$  should be determined as  $(1 + \text{sgn}(u_{ij} + v_{ij}))/2$ . As there are typically multiple  $j$  such that  $b_{ij} = 1$ , to facilitate the clustering process, we take a greedy approach by letting node  $i$  choose

$$j^* = \arg \max_j \{u_{ij} + v_{ij}\} \quad (24)$$

as the one that is in the same cluster as itself, and send this result to all its neighbors. Finally, every node obtains all its neighbors' clustering information by exchanging its co-cluster neighbor selection.

After the above (first round) clustering process is completed, we can regard the generated clusters as "mega-nodes", and conduct the similar message passing operation between these clusters, to determine which clusters need to be further combined. Assume totally  $K_1$  clusters are formed in the previous round, we can similarly form a cluster-based factor graph (for those clusters whose sizes are less than  $L$ ) as in Figure 2:

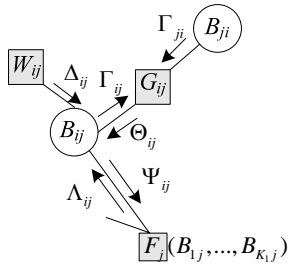


Figure 2. Factor graph structure of "mega-node" clustering

Here variables  $B_{ij} \in \{0,1\}$ ,  $i, j = 1, \dots, K_1$ , are cluster combination indicators:  $B_{ij} = 1$  indicates cluster  $i$  and cluster  $j$  should be combined, and  $B_{ij} = 0$  otherwise.  $W_{ij}$  is the sum of weights on all inter-cluster edges between Cluster  $i$  and  $j$ . Similarly, function  $G_{ij}(B_{ij}, B_{ji})$ ,  $i, j = 1, \dots, K_1$  enforces that  $B_{ij}$  and  $B_{ji}$  are always consistent. Function  $F_j(B_{1j}, \dots, B_{K_1j})$  guarantees that the size of the combined cluster is still less than  $L$ :

$$F_j(B_{1j}, \dots, B_{K_1j}) = \begin{cases} 1 & \text{if } \sum_{i=1}^{K_1} B_{ij} |C_i| \leq L \\ 0 & \text{otherwise,} \end{cases} \quad (25)$$

where  $|C_i|$  denotes the cardinality of cluster  $i$ .

Consider incoming messages of  $F_j$ ,  $\{\Psi_{i'j}\}$  such that  $i' \neq i \neq j$ , and sort them as  $\Psi_{i'j}^{(1)} \geq \Psi_{i'j}^{(2)} \geq \dots$ . Assume there is a number  $L'$  such that  $\Psi_{i'j}^{(L'-1)} > 1$  while  $\Psi_{i'j}^{(L')} \leq 1$ . Let  $U_{ij} = \log \Psi_{ij}$ ,  $V_{ij} = \log \Lambda_{ij}$ . Following the previous procedure,

we can get the counterparts of (22) and (23) as (details omitted in the interest of space)

$$V_{ij} = \begin{cases} -\sum_{l=L'}^{\bar{L}} U_{i'l} & \sum_{i'=1}^{L'} |C_{i'}| \geq L - |C_i|, \\ 0 & \sum_{i'=1}^{L'} |C_{i'}| < L - |C_i|, \end{cases} \quad (26)$$

$$U_{ij} = W_{ij} + W_{ji} + V_{ji}, \quad (27)$$

where  $\bar{L}$  and  $\bar{L}'$  are chosen such that

$$\begin{aligned} \sum_{l=1}^{\bar{L}} |C_l| \leq L, \quad \sum_{l=1}^{\bar{L}'} |C_l| > L, \quad \text{and} \\ \sum_{l=1}^{\bar{L}-1} |C_l| \leq L - |C_i|, \quad \sum_{l=1}^{\bar{L}'} |C_l| > L - |C_i|. \end{aligned} \quad (28)$$

Finally, if there exists  $U_{ij} + V_{ij} > 0$ , cluster  $i$  chooses

$$j^* = \arg \max_j \{U_{ij} + V_{ij}\}, \quad (29)$$

as the one that is combined with itself; otherwise, it keeps unchanged. This process is recursively carried out until no clusters can be combined.

#### IV. SIMULATION RESULTS

In this section, we present some numerical results to test the effectiveness of the proposed algorithm. Firstly, we compare our approach with some other clustering algorithms, including affinity propagation [6], HEED, a well-known distributed clustering scheme in wireless networks [10] (modified so that only node degree is considered and the metric residual energy is ignored), and a centralized semi-definite programming method which solves a relaxation of the equi-min-cut problem [8].

We consider a setting where nodes are randomly distributed on a unit plane. The weight on the edge between node  $i$  and  $j$  is set as  $w_{ij} = \exp(-d_{ij}/\sigma)$ , where  $d_{ij}$  is the Euclidean distance between  $i$  and  $j$ , and  $\sigma$  is the attenuation parameter, set to 10 in our simulations.

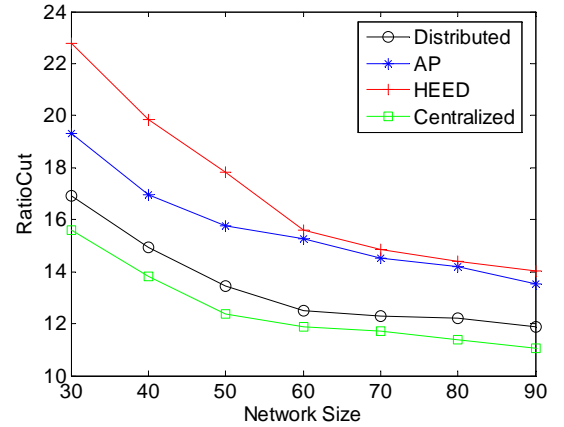


Figure 3. Ratio cut comparison with different network sizes

In Figure 3, we fix the number of clusters as 10, and compare the ratio cuts of the four approaches for different network sizes, which is defined as [2]

$$\text{RatioCut}(C_1, \dots, C_K) = \sum_{k=1}^K \text{cut}(C_k, \bar{C}_k) / |C_k|, \quad (30)$$

where  $\text{cut}(C_k, \bar{C}_k) = \sum_{(i,j) \in E, i \in C_k, j \notin C_k} w_{ij}$  is the cut weight of

cluster  $C_k$ . We can see as the network size increases, the performance of our distributed algorithm approaches that of the centralized one, and it consistently outperforms the other two. It is also observed that the computational time of our algorithm is compatible with HEED and AP, while that of the centralized method is tens and up to one hundred times longer. Note that

AP can only implicitly control the cluster size through one parameter, and it results in a worse performance.

Then, with the network size fixed to 60, we examine the performance for different numbers of clusters. A similar trend is observed in Figure 4.

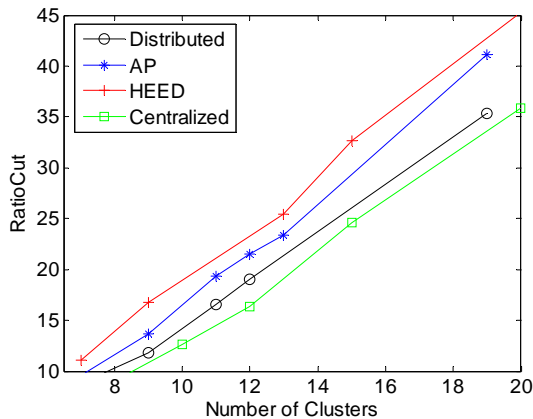


Figure 4. Ratio cut comparison with different cluster sizes

Distributed inference is the primary task of a wide range of wireless networking applications. For a large network, it is often advantageous to partition it into smaller clusters with small inter-cluster correlation, and exploit simple approximate methods for inter-cluster inference while use more complex and accurate approaches for intra-cluster inference. One example is the structured mean field (SMF) approach developed in our recent work [9]. Let us consider a Gaussian Markov random field estimation problem (see [9] for details), with 300 nodes randomly distributed in a unit plane. The estimation mean square error is used as the comparison metric, defined as

$$MSE = \|\hat{\mathbf{x}}_m - \hat{\mathbf{x}}\|_2 / \|\hat{\mathbf{x}}\|_2, \quad (31)$$

with  $\hat{\mathbf{x}}_m$  denoting the estimate at the  $m$ th iteration, and  $\hat{\mathbf{x}}$  being the optimal estimate (the MMSE solution).

Figure 5 compares the convergence rates of the SMF approach with our distributed clustering and HEED clustering. In SMF, belief propagation (BP) is employed for intra-cluster inference, while the naive mean field (MF) approach is employed for inter-cluster inference. Their performances on the entire network are also given as the two extremes for SMF, corresponding to the two cases with one cluster, and clusters of size 1, respectively. The results verify that BP converges fastest, while SMF can significantly speed up the convergence. A reasonable cluster size (8 in this setting) achieves performance very close to that of BP. It is observed that our distributed clustering scheme yields more accurate estimation than HEED.

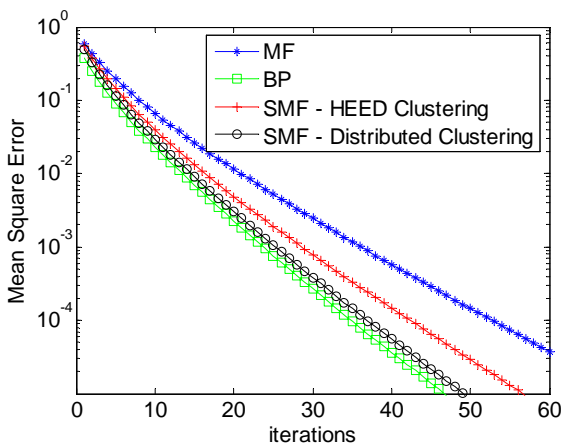


Figure 5. Performance of distributed clustering for structured inferences- Convergence Rate

Another important factor for wireless networks is the energy consumption. This comparison is illustrated in Figure 6, where we assume for simplicity that the communication energy is proportional to the message complexity, and the computation energy is neglected. Note that compared to MF, BP exchanges more messages per round but converges faster. It is interesting to observe that for this simulation setting SMF consumes the least communication energy to obtain the same estimation accuracy, which indicates its potential superiority in practice. And by balancing the cluster size, our proposed distributed clustering scheme also consumes less energy than HEED for inference.

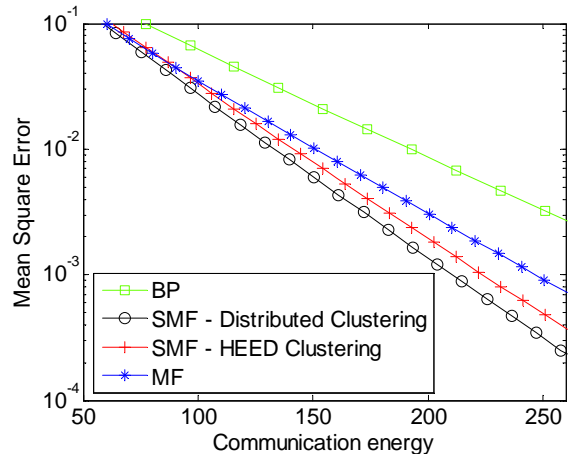


Figure 6. Performance of distributed clustering for structured inferences- Energy Consumption

## V. CONCLUSIONS

Motivated by the need of efficient distributed clustering in large-scale networking and data processing applications, we propose a simple and fully distributed network decomposition algorithm in this paper, which is general enough to admit wide applications. One future direction is to theoretically quantify the quality and efficiency of this clustering algorithm.

## REFERENCES

- [1] K. Schloegel, G. Karypis, and V. Kumar, *Graph partitioning for high performance scientific simulations*, CRPC Parallel Computing Handbook, Morgan Kaufmann, 2000.
- [2] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, 17(4), 2007.
- [3] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: analysis and an algorithm," *Advances in Neural Information Processing Systems 14*, MIT Press 2002.
- [4] D. Kempe and F. McSherry, "A decentralized algorithm for spectral analysis," *Proc. the 36<sup>th</sup> Annual ACM Symposium on Theory of Computing (STOC)*, 2004.
- [5] O. Younis, M. Krunz, and S. Ramasubramanian, "Node clustering in wireless sensor networks: Recent developments and deployment challenges," *IEEE Network*, Vol. 20, issue 3, pp. 20-25, May 2006.
- [6] B. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, vol. 315, pp. 972-976, 2007.
- [7] F.R. Kschischang, B.J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-Product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498-519, 2001.
- [8] E.P. Xing, M.I. Jordan and S. Russell, "Graph partition strategies for generalized mean field inference," *Proc. of 20th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 2004.
- [9] Y. Zhang and H. Dai, "Structured variational methods for distributed inference in wireless ad hoc and sensor networks," *2009 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Taipei, April 2009.
- [10] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid energy-efficient approach," *Proc. IEEE INFOCOM*, March 2004.