arXiv:1309.6129v1 [cs.DS] 24 Sep 2013

Partition-Merge: Distributed Inference and Modularity Optimization

Vincent Blondel VINCENT.BLONDEL@UCLOUVAIN.BE
Universite Catholique de Louvain, Belgium

Kyomin Jung KJUNG@SNU.AC.KR
Seoul National University, South Korea

Pushmeet Kohli PKOHLI@MICROSOFT.COM
Microsoft Research, Cambridge, UK

Devavrat Shah DEVAVRAT@MIT.EDU
MIT, Cambridge, USA

Editor: Editor

Abstract

This paper presents a novel meta algorithm, Partition-Merge (PM), which takes existing centralized algorithms for graph computation and makes them distributed and faster. In a nutshell, PM divides the graph into small subgraphs using our novel randomized partitioning scheme, runs the centralized algorithm on each partition separately, and then *stitches* the resulting solutions to produce a global solution. We demonstrate the efficiency of the PM algorithm on two popular problems: computation of Maximum A Posteriori (MAP) assignment in an arbitrary pairwise Markov Random Field (MRF), and modularity optimization for community detection. We show that the resulting distributed algorithms for these problems essentially run in time linear in the number of nodes in the graph, and perform as well – or even better – than the original centralized algorithm as long as the graph has geometric structures[1]. More precisely, if the centralized algorithm is a $\mathcal{C}-$factor approximation with constant $\mathcal{C} \geq 1$, the resulting distributed algorithm is a $(\mathcal{C}+\delta)$-factor approximation for any small $\delta > 0$; but if the centralized algorithm is a non-constant (e.g. logarithmic) factor approximation, then the resulting distributed algorithm becomes a constant factor approximation. For general graphs, we compute explicit bounds on the loss of performance of the resulting distributed algorithm with respect to the centralized algorithm.

Keywords: Graphical model, Approximate MAP, Modularity Optimization, Partition

1. Introduction

Graphical representation for data has become central to modern large-scale data processing applications. For many of these applications, large-scale data computation boils down to solving problems defined over massive graphs. While the theory of centralized algorithms for graph problems is getting reasonably well developed, their distributed (as well as parallel) counterparts are still poorly understood and remain very active areas of current investigation. Moreover, the emerging cloud-computation architecture is making the need of distributed solutions even more pressing.

Summary of results. In this paper, we take an important step towards this challenge. Specifically, we present a meta algorithm, Partition-Merge (PM), that makes existing centralized (exact or ap-

1. Roughly speaking, a graph has geometric structures, or polynomial growth property, when the number of nodes within distance r of any given node grows no faster than a polynomial function of r.

proximate) algorithms for graph computation distributed and faster without loss of performance, and in some cases, even improving performance. The PM meta algorithm is based on our novel partitioning of the graph into small disjoint subgraphs. In a nutshell, PM partitions the graph into small subgraphs, runs the centralized algorithm on each partition separately (which can be done in distributed or parallel manner); and finally *stitches* the resulting solutions to produce a global solution. We apply the PM algorithm to two representative class of problems: the MAP computation in a pairwise MRF and modularity optimization based graph clustering.

The paper establishes that for any graph that satisfies the polynomial growth property, the resulting distributed PM based implementation of the original centralized algorithm is a $(\mathcal{C} + \delta)$-approximation algorithm whenever the centralized algorithm is a $\mathcal{C}$-approximation algorithm for some constant $\mathcal{C} \geq 1$. In this expression, δ is a small number that depends on a tuneable parameter of the algorithm that affects the size of the induced subgraphs in the partition; the larger the subgraph size, the smaller the δ. More generally, if the centralized algorithm is an $\alpha(n)$-approximation (with $\alpha(n) = o(n)$) for a graph of size n, the resulting distributed algorithm becomes a constant factor approximation for graphs with geometric structure! The computational complexity of the algorithm scales linearly in n. Thus, our meta algorithm can make centralized algorithms, faster, distributed and improve its performance.

The algorithm applies to any graph structure, but strong guarantees on performance, as stated above, require geometric structure. However, it is indeed possible to explicitly evaluate the loss of performance induced by the distributed implementation compared to the centralized algorithm as stated in Section 4.2.

A cautionary remark is in order. Indeed, by no means, this algorithm means to answer all problems in distributed computation. Specifically, for dense graph, this algorithm is likely to exhibit poor performances and definitely such graph structure would require a very different approach. Our meta algorithm requires that the underlying graph problem is *decomposable* or *Markovian* in a sense. Not all problems have this structure and these problem therefore require different way to think about them.

Related Work and Our contributions

The results of this paper, on one hand, are related to a long list of works on designing distributed algorithms for decomposable problems. On the other hand, the applications of our method to MAP inference in pairwise MRFs and clustering relate our work to a large number of results in these two respective problems. We will only be able to discuss very closely related work here.

We start with the most closely related work on the use of graph partitioning for distributed algorithm design. Such an approach is quite old; see, e.g., Awerbuch et al. (1989); Klein et al. (1993) and Peleg (2000) for a detailed account of the approach until 2000. More recently, such decompositions have found wide variety of applications including local-property testing Hassidim et al. (2009). All such decompositions are useful for *homogeneous* problems, e.g. for finding maximum-*size* matching or independent set rather than the *heterogenous* maximum-*weight* variants of it. To overcome this limitation, a different (somewhat stronger) notion of decomposition was introduced by Jung and Shah Jung and Shah (2007) for minor-excluded graphs that built upon Klein et al. (1993). All of these results effectively partition the graph into small subgraphs and then solve the problem inside each small subgraph using exact (dynamic programming) algorithms. While this results in a $(1 + \epsilon)$-approximation algorithm for any $\epsilon > 0$ with computation scaling essentially linearly in the graph size (n), the computation constant depends super-exponentially in $1/\epsilon$. Therefore, even with $\epsilon = 0.1$, the algorithms become unmanageable in practice.

As the main contribution of this paper, we first propose a novel graph decomposition scheme for graphs with geometry or polynomial growth structure. Then we establish that by utilizing this decomposition scheme along with *any* centralized algorithm (instead of dynamic programming) for solving the problem inside the partition leads to performance comparable (or better) to that of the centralized algorithm for graph with polynomial growth. Then the resulting distributed algorithm becomes very fast in practice, unlike the dynamic programming approach, if the centralized algorithm inside the partition runs fast. Similar guarantees can be obtained for minor-excluded graphs as well using the scheme utilized in Jung and Shah (2007). As mentioned earlier, the result is established for both MAP in pair-wise MRF and modularity optimization based clustering.

MAP Inference. Computing the exact Maximum a Posteriori (MAP) solution in a general probabilistic model is an NP-hard problem. A number of algorithmic approaches have been developed to obtain approximate solutions for these problems. Most of these methods work by making 'local updates' to the assignment of the variables. Starting from an initial solution, the algorithms search the space of all possible local changes that can be made to the current solution (also called move space), and choose the best amongst them.

One such algorithm (which has been rediscovered multiple times) is called Iterated Conditional Modes or ICM for short. Its local update involves selecting (randomly or deterministically) a variable of the problem. Keeping the values of all other variables fixed, the value of the selected variable is chosen which results in a solution with the maximum probability. This process is repeated by selecting other variables until the probability cannot be increased further. The local step of the algorithm can be seen as performing inference in the smallest decomposed subgraph possible.

Another family of methods are related to max-product belief propagation (cf. Pearl (1988) and Yedidia et al. (2000)). In recent years a sequence of results suggest that there is an intimate relation between the max-product algorithm and a natural linear programming relaxation – for example, see Wainwright et al. (2005); Bayati et al. (2005, 2008); Huang and Jebara (2007); Sanghavi et al. (2007). Many of these methods can be seen as making local updates to partitions of the dual problem Sontag and Jaakkola (2009); Tarlow et al. (2011).

We also note that Swendsen-Wang algorithm (SW)Swendsen and Wang (1987), a local flipping algorithm, has a philosophy similar to ours in that it repeats a process of randomly partitioning the graph, and computing an assignment. However, the graph partitioning of SW is fundamentally different from ours and there is no known guarantee for the error bound of SW.

In summary, all the approaches thus far with provable guarantees for local update based algorithm are primarily for linear or more generally convex optimization setup.

Modularity Optimization for Clustering. The notion of modularity optimization was introduced by Newmann Newman (2006) to identify the communities or clusters in a network structure. Since then, it has become quite popular as a metric to find communities or clusters in variety of networked data cf. Blondel et al. (2008, 2010). The major challenge has been design of approximation algorithm for modularity optimization (which is computationally hard in general) that can operate in distributed manner and provide performance guarantees. Such algorithms with provable performance guarantees are known only for few cases, notably logarithmic approximation of DasGupta and Desai (2011) via a centralized solution.

Our contribution in the context of modularity optimization lies in showing that indeed it is a *decomposable* problem and therefore admits an distributed and fast approximation algorithm through our approach.

Organization. The rest of the paper is organized as follows. Section 2 describes problem statement and preliminaries. Section 3 describes our main algorithms, and Section 4 presents analyses of our algorithms. Section 6 and Section 6 provides the proofs of our main theorems, and Section 7 presents the conclusion.

2. Setup

Graphs. Our interest is in processing networked data represented through an undirected graph $G = (V, E)$ with $n = |V|$ vertices and E being the edge set. Let $m = |E|$ be the number of edges. Graphs can be classified structurally in many different ways: trees, planar, minor-excluded, geometric, expanding, and so on. We shall establish results for graphs with geometric structure or polynomial growth which we define next. A graph $G = (V, E)$ induces a natural 'graph metric' on vertices V, denoted by $\mathbf{d_G} : V \times V \to \mathbb{R}_+$ with $\mathbf{d_G}(i, j)$ given by the length of the shortest path between i and j; defined as ∞ if there is no path between them.

Definition 1 (Graph with Polynomial Growth)**.** *We say that a graph G (or a collection of graphs) has polynomial growth of degree (or growth rate) ρ, if for any $i \in V$ and $r \in \mathbb{N}$,*

$$|\mathbf{B}_G(i, r)| \le C \cdot r^\rho,$$

where $C > 0$ is a universal constant and $\mathbf{B}_G(i, r) = \{j \in V | \mathbf{d}_G(i, j) < r\}$.

Note that interesting values of C, ρ are integral between $\{0, 1, \dots, n\}$, and it is easy to verify in $O(mn)$ time. Therefore we will assume knowledge of C, ρ for algorithm design. A large class of graph model naturally fall into the graphs with polynomial growth. To begin with, the standard d-dimensional regular grid graphs have polynomial growth rate d. More generally, in recent years in the context of computational geometry and metric embedding, the graphs with finite doubling dimensions have become popular object of study Gupta et al. (2003). It can be checked that a graph with doubling dimension ρ is also a graph with polynomial growth rate ρ. Finally, the popular geometric graph model where nodes are placed arbitrarily in some Euclidean space with some minimum distance separation, and two nodes have an edge between them if they are within certain finite distance, has finite polynomial growth rate Gummadi et al. (2009).

Pair-wise graphical model and MAP. For a pair-wise Markov Random Filed (MRF) model defined on a graph $G = (V, E)$, each vertex $i \in V$ is associated with a random variable X_i which we shall assume to be taking value from a finite alphabet Σ; the edge $(i, j) \in E$ represents a form of 'dependence' between X_i and X_j. More precisely, the joint distribution is given by

$$\mathbb{P}\big(\mathbf{X} = \mathbf{x}\big) \propto \prod_{i \in V} \phi_i(x_i) \cdot \prod_{(i,j) \in E} \psi_{ij}(x_i, x_j) \tag{1}$$

where $\phi_i : \Sigma \to \mathbb{R}_+$ and $\psi_{ij} : \Sigma^2 \to \mathbb{R}_+$ are called node and edge potential functions[2]. The question of interest is to find the maximum a posteriori (MAP) assignment $\mathbf{x}^* \in \Sigma^n$, i.e.

$$\mathbf{x}^* \in \arg\max_{\mathbf{x} \in \Sigma^n} \mathbb{P}[\mathbf{X} = \mathbf{x}].$$

2. For simplicity of the analysis we assume strict positivity of ϕ_i's and ψ_{ij}'s.

Equivalently, from the optimization point of view, we wish to find an optimal assignment of the problem

$$\text{maximize} \quad \mathcal{H}(\mathbf{x}) \quad \text{over} \quad \mathbf{x} \in \Sigma^n, \quad \text{where}$$
$$\mathcal{H}(\mathbf{x}) = \sum_{i \in V} \ln \phi_i(x_i) + \sum_{(i,j) \in E} \ln \psi_{ij}(x_i, x_j).$$

For completeness and simplicity of exposition, we assume that the function $\mathcal{H}$ is finite valued over Σ^n. However, results of this paper extend for hard constrained problems such as the *hardcore* or *independent set* model. We call an algorithm α approximation for $\alpha \geq 1$ if it always produces assignment $\widehat{\mathbf{x}}$ such that

$$\frac{1}{\alpha}\mathcal{H}(\mathbf{x}^*) \leq \mathcal{H}(\widehat{\mathbf{x}}) \leq \mathcal{H}(\mathbf{x}^*).$$

Social data and clustering/community. Alternatively, in a social setting, vertices of graph G can represents individuals and edges represent some form of interaction between them. For example, consider a cultural show organized by students at a university with various acts. Let there be n students in total who have participated in one or more acts. Place an edge between two students if they participated in at least one act together. Then the resulting graph represents interaction between students in terms of acting together.

Based on this observed network, the goal is to identify the set of all acts performed and its 'core' participants. The true answer, which classifies each student/node into the acts in which s/he performed would lead to partitions of nodes in which a node may belong to multiple partitions. Our interest is in identifying disjoint partitions which would, in this example, roughly mean identification of 'core' members of acts.

In general, to select a disjoint partition of V given G, it is not clear what is the appropriate criteria. Newman Newman (2006) proposed the notion of modularity as a criteria. The intuition behind it is that a cluster or community should be as distinct as possible from being 'random'. Modularity of a partition of nodes is defined as the fraction of the edges that fall within the disjoint partitions minus the expected such fraction if edges were distributed at random with the same node degree sequences. Formally, the modularity of a subset $S \subset V$ is defined as

$$M(S) = \sum_{i,j \in S} \Big(A_{ij} - \frac{d_i d_j}{2m}\Big), \tag{2}$$

where $A_{ij} = 1$ iff $(i,j) \in E$ and 0 otherwise, $d_i = |\{k \in V : (i,k) \in E\}|$ is the degree of node $i \in V$, and $m = |E|$ represents the total number of edges in G. More generally, the modularity of a partition of V, $V = S_1 \cup \cdots \cup S_\ell$ for some $1 \leq \ell \leq n$ with $S_i \cap S_j = \emptyset$ for $i \neq j$, is given by

$$\mathcal{M}(S_1, \ldots, S_\ell) = \frac{1}{2m}\Big(\sum_{i=1}^{\ell} M(S_i)\Big). \tag{3}$$

The modularity optimization approach Newman (2006) proposes to identify the community structure as the disjoint partitions of V that maximizes the total modularity, defined as per (3), among all possible disjoint partitions of V with ties broken arbitrarily. The resulting clustering of nodes is the desired answer.

We shall think of clustering as assigning *colors* to nodes. Specifically, given a coloring $\chi : V \to \{1, \ldots, n\}$, two nodes i and j are part of the same cluster (partition) iff $\chi(i) = \chi(j)$. With this notation, any clustering of V can be represented by some such coloring χ and vice versa. Therefore, modularity optimization is equivalent to finding a coloring χ such that its modularity $\mathcal{M}(\chi)$ is maximized, where

$$\mathcal{M}(\chi) \;= \frac{1}{2m} \sum_{i,j \in V} \mathbf{1}_{\{\chi(i)=\chi(j)\}} \Big(A_{ij} - \frac{d_i d_j}{2m} \Big).$$

Here $\mathbf{1}_{\{\cdot\}}$ is the indicator function with $\mathbf{1}_{\{\text{true}\}} = 1$ and $\mathbf{1}_{\{\text{false}\}} = 0$. Let χ^* be a clustering that maximizes the modularity. Then, as before, an algorithm will be said α-approximate if it produces $\widehat{\chi}$ such that

$$\frac{1}{\alpha}\mathcal{M}(\chi^*) \leq \mathcal{M}(\widehat{\chi}) \; \leq \mathcal{M}(\chi^*). \tag{4}$$

3. Partition-Merge Algorithm

We describe a parametric meta-algorithm for solving the MAP inference and modularity optimization. The meta-algorithm uses two parameters; a large constant $K \geq 1$ and a small real number $\varepsilon \in (0, 1)$ to produce a partition of $V = V_1 \cup \cdots \cup V_p$ so that each partition $V_j, 1 \leq j \leq p$ is small. We will specify the values of K and ε in Section 4. The meta-algorithm uses an existing centralized algorithm to solve the original problem on each of these partitioned sub-graphs $G_j = (V_j, E_j)$ independently where $E_j = (V_j \times V_j) \cap E$. The result assignment leads to a candidate solution for the problem on entire graph. As we establish in Section 4, this becomes a pretty good solution. Next, we describe the algorithm in detail.

Step 1. Partition. We wish to create a partition of $V = V_1 \cup \cdots \cup V_p$ for some p with $V_i \cap V_j = \emptyset$ for $i \neq j$ so that the number of edges crossing partitions are small. The algorithm for such partitioning is iterative. Initially, no node is part of any partition. Order the n nodes arbitrarily, say $i_1, \ldots, i_n$. In iteration $k \leq n$, choose node i_k as the pivot. If i_k belongs to $\cup_{\ell=1}^{k-1} V_\ell$, then set $V_k = \emptyset$, and move to the next iteration if $k < n$ or else the algorithm concludes. If $i_k \notin \cup_{\ell=1}^{k-1} V_\ell$, choose a radius $R_k \leq K$ at random with distribution

$$\mathbb{P}\Big(R_k = \ell\Big) = \begin{cases} \varepsilon(1-\varepsilon)^{\ell-1} & \text{for } 1 \leq \ell < K \\ (1-\varepsilon)^{K-1}, & \text{for } \ell = K. \end{cases} \tag{5}$$

Let V_k be set of all nodes in V that are within distance R_k of i_k, but that are not part of $V_1 \cup \cdots \cup V_{k-1}$. Since we execute this step only if $i_k \notin \cup_{\ell=1}^{k-1} V_\ell$ and $R_k \geq 1$, V_k will be non-empty. At the end of the n iterations, we have a partition of V with at most n non-empty partitions. Let the non-empty partitions of V be denoted as $V = V_1 \cup \cdots \cup V_p$ for some $p \leq n$. A caricature of an iteration is described in Figure 1.

Step 2. Merge (solving the problem). Given the partition $V = V_1 \cup \cdots \cup V_p$, consider the graphs $G_k = (V_k, E_k)$ with $E_k = (V_k \times V_k) \cap E$ for $1 \leq k \leq p$. We shall apply a centralized algorithm for each of these graph $G_1, \ldots, G_k$ separately. Specifically, let $\mathcal{A}$ be an algorithm for MAP or for clustering: the algorithm may be exact (e.g. one solving problem by exhaustive search over all possible options, or dynamic programming) or it may be an approximation algorithm (e.g. α-approximate for any graph). We apply $\mathcal{A}$ for each subgraph separately.

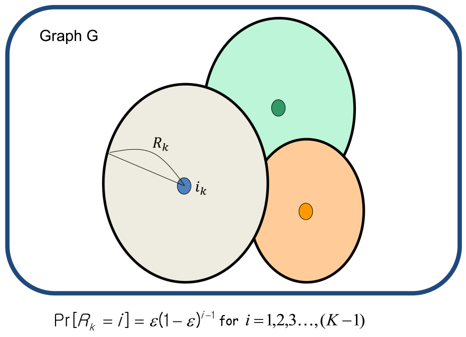
Graph G
R_k
i_k
$\Pr[R_k = i] = \varepsilon(1-\varepsilon)^{i-1}$ for $i = 1,2,3\ldots,(K-1)$

Figure 1: A pictorial description of an iteration of the graph partitioning.

- For MAP inference, this results in an assignment to all variables since in each partition each node is assigned some value and collectively all nodes are covered by the partition. Declare thus resulting global assignment, say $\widehat{\mathbf{x}}$ as the solution for MAP.
- For modularity optimization, nodes in each partition V_j are clustered. We declare the union of all such clusters across partitions as the global clustering. Thus two nodes in different partitions are always in different clusters; two nodes in the same partition are in different clusters if the centralized algorithm applied to that partition clusters them differently.

Computation cost. The computation cost of the partitioning scheme scales linearly in the number of edges in the graph. The computation cost of solving the problem in each of the components $G_1, \ldots, G_p$ depends on component sizes and on how the computation cost of algorithm $\mathcal{A}$ scales with the size. In particular, if the maximum degree of any node in G is bounded, say by d, then each partition has at most d^K nodes. Then the overall cost is $O(Q(d^K)n)$ where $Q(\ell)$ is the computation cost of $\mathcal{A}$ for any graph with ℓ vertices.

4. Main results

4.1 Graphs with polynomial growth

We state sharp results for graphs with polynomial growth. We state results for MAP inference and for modularity optimization under the same theorem statement to avoid repetition. The proofs, however, will have some differences.

Theorem 1. *Let the graph $G = (V, E)$ have polynomial growth with degree $\rho \geq 1$ and constant $C \geq 1$. Then, for a given $\delta \in (0, 1)$, select parameters*

$$\begin{aligned} K = K(\rho, C, \delta) &= \frac{8\rho}{\varepsilon} \log\left(\frac{8\rho}{\varepsilon}\right) + \frac{4}{\varepsilon} \log C + \frac{4}{\varepsilon} \log \frac{1}{\varepsilon} + 2, \\ \varepsilon = \varepsilon(\rho, C, \delta) &= \begin{cases} \frac{\delta}{2C2^{\rho}}, & \textit{for MAP} \\ \frac{\delta}{4(2C-1)}, & \textit{for modularity optimization.} \end{cases} \end{aligned} \tag{6}$$

Then, the following holds for the meta algorithm described in Section 3.

(a) *If $\mathcal{A}$ solves the problem (MAP or modularity optimization) exactly, then the solution produced by the algorithm $\widehat{\mathbf{x}}$ and $\widehat{\chi}$ for MAP and modularity optimization respectively are such that*

$$(1-\delta)\mathcal{H}(\mathbf{x}^*) \leq \mathbb{E}[\mathcal{H}(\widehat{\mathbf{x}})] \leq \mathcal{H}(\mathbf{x}^*)$$
$$(1-\delta)\mathcal{M}(\chi^*) \leq \mathbb{E}[\mathcal{M}(\widehat{\chi})] \leq \mathcal{M}(\chi^*). \tag{7}$$

(b) *If $\mathcal{A}$ is $\alpha(n) \geq 1$ approximation algorithm for graphs with n nodes, then*

$$\left(\frac{1}{\alpha(\tilde{K})} - \delta\right)\mathcal{H}(\mathbf{x}^*) \leq \mathbb{E}[\mathcal{H}(\widehat{\mathbf{x}})] \leq \mathcal{H}(\mathbf{x}^*),$$
$$\frac{(1-\delta)}{\alpha(\tilde{K})}\mathcal{M}(\chi^*) \leq \mathbb{E}[\mathcal{M}(\widehat{\chi})] \leq \mathcal{M}(\chi^*), \tag{8}$$

where $\tilde{K} = CK^\rho$.

4.2 General graph

The theorem in the previous section was for graphs with polynomial growth. We now state results for general graph. Our result tells us how to evaluate the 'error bound' on solutions produced by the algorithm for any instantiation of randomness. The result is stated below for both MAP and modularity optimization. The 'error function' depends on the problem.

Theorem 2. *Given an arbitrary graph $G = (V, E)$ and our algorithm operating on it with parameters $K \geq 1$, $\varepsilon \in (0, 1)$ using a known procedure $\mathcal{A}$, the following holds:*

(a) *If $\mathcal{A}$ solves the problem (MAP or modularity optimization) exactly, then the solution produced by the algorithm $\widehat{\mathbf{x}}$ and $\widehat{\chi}$ for MAP and modularity optimization respectively are such that (with $\mathcal{B} = E \backslash \cup_{k=1}^{p} E_k$),*

$$\mathcal{H}(\widehat{\mathbf{x}}) \geq \mathcal{H}(\mathbf{x}^*) - \sum_{(i,j)\in\mathcal{B}} \left(\psi_{ij}^U - \psi_{ij}^L\right),$$
$$\mathcal{M}(\widehat{\chi}) \geq \mathcal{M}(\chi^*) - \frac{|\mathcal{B}|}{2m}. \tag{9}$$

(b) *If $\mathcal{A}$ is instead a $\alpha(n)$-approximation for graphs of size n, then*

$$\mathcal{H}(\widehat{\mathbf{x}}) \geq \frac{1}{\alpha(\tilde{K})}\Big(\mathcal{H}(\mathbf{x}^*) - \sum_{(i,j)\in\mathcal{B}} \left(\psi_{ij}^U - \psi_{ij}^L\right)\Big)$$
$$\mathcal{M}(\tilde{\chi}) \geq \frac{1}{\alpha(\tilde{K})}\Big(\mathcal{M}(\chi^*) - \frac{|\mathcal{B}|}{2m}\Big), \tag{10}$$

where $\tilde{K}$ is the maximum number of nodes that are within K hops of any single node in V.

In the expression above, $\psi_{ij}^U \triangleq \max_{\sigma,\sigma'\in\Sigma} \ln \psi_{ij}(\sigma, \sigma')$, and $\psi_{ij}^L \triangleq \min_{\sigma,\sigma'\in\Sigma} \ln \psi_{ij}(\sigma, \sigma')$.

4.3 Discussion of results

Here we dissect implications of the above stated theorems. To start with, Theorem 1(a) suggests that when graphs have polynomial growth, there exists a Randomized Polynomial Time Approximation Scheme (PTAS) for MAP computation and modularity optimization that has computation time scaling linearly with n.

The dependence on ρ and δ is rather stringent and therefore, even for moderately small δ, it may not be possible to utilize existing computers to implement brute-force/dynamic programming procedure. The Theorem 1(b) suggests that, if instead of using exact procedure for each partition, when an approximation algorithm is used, the resulting solution almost retains its approximation guarantees: if $\alpha(n)$ is a constant, then the resulting approximation guarantee is essentially the same constant; if $\alpha(n)$ increases with n (e.g. $\log n$), then the resulting algorithm provides a constant factor approximation ! In either case, even if the approximation algorithm has superlinear computation time in the number of nodes (e.g. semi-definite programming), then our algorithm provides a way to achieve similar performance but in linear time for polynomially growing graphs.

The algorithm, for general graph, produces a solution for which we have approximation guarantees. Specifically, the error scales with the fraction of edges across partitions that are induced by our partitioning procedure. This error depends on parameters K, ε utilized by our partitioning procedure. For graph with polynomial growth, we provide recommendations on what the values should be for these parameters. However, for general graph we do not have precise recommendations. Instead, one may try various values of $K \in \{1, \ldots, n\}$ and $\varepsilon \in (0, 1)$ and then choose the best solution. Indeed, a reasonable way to implement such procedure would be to take values of K that are 2^k for $k \in \{0, \ldots, \log n\}$ and ε chosen at regular interval with granularity that an implementor is comfortable with (the smaller the granularity, the better).

5. Proofs of Theorems 1, 2: MAP inference

In this Section, we first prove Theorem 1, and Theorem 2 for MAP inference.

Bound on $|E \setminus \cup_{k=1}^{p} E_k|$. We first state the following Lemma which shows the essential property of the partition scheme. Lemma 1 will be used in the proofs of Theorems 1, 2 both for MAP and modularity optimization. The proof of Lemma 1 is stated at the end of this Section.

Lemma 1. *Given $G = (V, E)$ with polynomial growth of rate $\rho \geq 1$ and associated constant $C \geq 1$, by choosing $K = K(\rho, C, \delta)$ and $\varepsilon = \varepsilon(\rho, C, \delta) = \frac{\delta}{4(2C-1)}$, the partition scheme satisfies that for any edge $e \in E$,*

$$\mathbb{P}(e \in \mathcal{B}) \leq 2\varepsilon. \tag{11}$$

Lower bound on $\mathcal{H}(\mathbf{x}^*)$. Here we provide a lower bound on $\mathcal{H}^* = \mathcal{H}(\mathbf{x}^*)$ that will be useful to obtain multiplicative approximation property.

Lemma 2. *Let $\mathcal{H}^* = \max_{\mathbf{x} \in \Sigma^n} \mathcal{H}(\mathbf{x})$ denote the maximum value of $\mathcal{H}$ for a given pair-wise MRF on a graph G. If G has maximum vertex degree d^*, then*

$$(d^* + 1)\mathcal{H}(\mathbf{x}^*) \geq \sum_{(i,j) \in E} \left(\psi_{ij}^U - \psi_{ij}^L\right). \tag{12}$$

Proof. Assign weight $w_{ij} = \psi_{ij}^U$ to an edge $(i,j) \in E$. Since graph G has maximum vertex degree d^*, by Vizing's theorem there exists an edge-coloring of the graph using at most d^*+1 colors. Edges with the same color form a matching of the G. A standard application of Pigeon-hole's principle implies that there is a color with weight at least $\frac{1}{d^*+1}(\sum_{(i,j)\in E} w_{ij})$. Let $M \subset E$ denote these set of edges. Then

$$\sum_{(i,j)\in M} \psi_{ij}^U \geq \frac{1}{d^*+1}\left(\sum_{(i,j)\in E} \psi_{ij}^U\right).$$

Now, consider an assignment $\mathbf{x}^M$ as follows: for each $(i,j) \in M$ set $(x_i^M, x_j^M) = \arg\max_{(x,x')\in\Sigma^2} \psi_{ij}(x,x')$; for remaining $i \in V$, set x_i^M to some value in Σ arbitrarily. Note that for above assignment to be possible, we have used matching property of M. Therefore, we have

$$\begin{aligned}
\mathcal{H}(\mathbf{x}^M) &= \sum_{i\in V}\phi_i(x_i^M) + \sum_{(i,j)\in E} \psi_{ij}(x_i^M, x_j^M) \\
&= \sum_{i\in V}\phi_i(x_i^M) + \sum_{(i,j)\in E\setminus M} \psi_{ij}(x_i^M, x_j^M) + \sum_{(i,j)\in M} \psi_{ij}(x_i^M, x_j^M) \\
&\overset{(a)}{\geq} \sum_{(i,j)\in M} \psi_{ij}(x_i^M, x_j^M) \\
&= \sum_{(i,j)\in M} \psi_{ij}^U \\
&\geq \frac{1}{d^*+1}\left[\sum_{(i,j)\in E} \psi_{ij}^U\right]. \qquad (13)
\end{aligned}$$

Here (a) follows because ψ_{ij}, ϕ_i are non-negative valued functions. Since $\mathcal{H}(\mathbf{x}^*) \geq \mathcal{H}(\mathbf{x}^M)$ and $\psi_{ij}^L \geq 0$ for all $(i,j) \in E$, we prove Lemma 2. □

Decomposition of $\mathcal{H}^*$. Here we show that by maximizing $\mathcal{H}(\cdot)$ on a partition of V separately and combining the assignments, the resulting $\widehat{\mathbf{x}}$ has $\mathcal{H}(\cdot)$ value as good as that of MAP with penalty in terms of the edges across partitions.

Lemma 3. *For a given MRF defined on G, the algorithm the partition scheme produces output* $\widehat{\mathbf{x}}$ *such that*

$$\mathcal{H}(\widehat{\mathbf{x}}) \geq \mathcal{H}(\mathbf{x}^*) - \left(\sum_{(i,j)\in\mathcal{B}} (\psi_{ij}^U - \psi_{ij}^L)\right),$$

where $\mathcal{B} = E \setminus \cup_{k=1}^K E_k$, $\psi_{ij}^U \triangleq \max_{\sigma,\sigma'\in\Sigma} \ln \psi_{ij}(\sigma,\sigma')$, *and* $\psi_{ij}^L \triangleq \min_{\sigma,\sigma'\in\Sigma} \ln \psi_{ij}(\sigma,\sigma')$.

Proof. Let $\mathbf{x}^*$ be a MAP assignment of the MRF $\mathbf{X}$ defined on G. Given an assignment $\mathbf{x} \in \Sigma^{|V|}$ defined on a graph $G = (V,E)$ and a subgraph $S = (W,E')$ of G, let an assignment $\mathbf{x}' \in \Sigma^{|W|}$ be called *a restriction of* $\mathbf{x}$ *to* S if $\mathbf{x}'(v) = \mathbf{x}(v)$ for all $v \in W$. Let $S_1, \ldots, S_K$ be the connected components of $G' = (V, E - \mathcal{B})$, and let $\mathbf{x}_k^*$ be the restriction of $\mathbf{x}^*$ to the component S_k. Let $\mathbf{X}_k$ be the restriction of the MRF $\mathbf{X}$ to $G_k = (S_k, E_k)$, where $E_k = \{(u,w) \in E | u,w \in S_k\}$.

For $\mathbf{x}_k \in \Sigma^{|S_k|}$, define

$$\mathcal{H}_k(\mathbf{x}_k) = \sum_{i \in S_k} \phi_i(x_i) + \sum_{(i,j) \in E_k} \psi_{ij}(x_i, x_j).$$

Let $\widehat{\mathbf{x}}$ be the output of the partition scheme, and let $\widehat{\mathbf{x}}_k$ be the restriction of $\widehat{\mathbf{x}}$ to the component S_k. Note that since $\widehat{\mathbf{x}}_k$ is a MAP assignment of $\mathcal{H}_k(\cdot)$ by the definition of our algorithm, for all $k = 1, 2, \ldots K$,

$$\mathcal{H}_k(\widehat{\mathbf{x}}_k) \geq \mathcal{H}_k(\mathbf{x}_k^*). \tag{14}$$

Now, we have

$$\begin{aligned} \mathcal{H}(\widehat{\mathbf{x}}) - \mathcal{H}(\mathbf{x}^*) &= \sum_{k=1}^{K} [\mathcal{H}_k(\widehat{\mathbf{x}}_k) - \mathcal{H}_k(\mathbf{x}_k^*)] + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}(\widehat{x}_i, \widehat{x}_j) - \psi_{ij}(x_i^*, x_j^*) \\ &\overset{(a)}{\geq} \sum_{k=1}^{K} [\mathcal{H}_k(\widehat{\mathbf{x}}_k) - \mathcal{H}_k(\mathbf{x}_k^*)] - \sum_{(i,j) \in \mathcal{B}} (\psi_{ij}^U - \psi_{ij}^L) \\ &\overset{(b)}{\geq} - \sum_{(i,j) \in \mathcal{B}} (\psi_{ij}^U - \psi_{ij}^L). \end{aligned} \tag{15}$$

Here (a) follows from the definitions of ψ_{ij}^U and ψ_{ij}^L, and (b) follows from (14). This completes the proof of Lemma 3. □

Completing Proof of Theorem 1(a). Recall that the maximum vertex degree d^* of G is less than $2^\rho C$ by the definition of polynomially growing graph. Remind our definition $\varepsilon = \frac{\delta}{2C2^\rho}$ for MAP inference. Now we have that

$$\begin{aligned} \mathbb{E}[\mathcal{H}(\widehat{\mathbf{x}})] &\overset{(a)}{\geq} \mathcal{H}(\mathbf{x}^*) - \mathbb{E}\left[\sum_{(i,j) \in \mathcal{B}} \left(\psi_{ij}^U - \psi_{ij}^L\right)\right] && (16) \\ &\overset{(b)}{\geq} \mathcal{H}(\mathbf{x}^*) - 2\varepsilon \left(\sum_{(i,j) \in E} \left(\psi_{ij}^U - \psi_{ij}^L\right)\right) && (17) \\ &\overset{(c)}{\geq} \mathcal{H}(\mathbf{x}^*) \left(1 - 2\varepsilon(d^* + 1)\right) && (18) \\ &\overset{(d)}{\geq} (1 - \delta)\mathcal{H}(\mathbf{x}^*). && (19) \end{aligned}$$

Here (a) follows from Lemma 3, (b) follows from Lemma 1, (c) from Lemma 2, and (d) follows from the definition of ε for MAP inference. This completes the proof of Theorem 1(a) for MAP inference.

Completing Proof of Theorem 1(b). Suppose that we use an approximation procedure $\mathcal{A}$ to produce an approximate MAP assignment $\widehat{\mathbf{x}}_k$ on each partition S_k in our algorithm. Let $\mathcal{A}$ be such that the assignment produced satisfies that $\mathcal{H}_k(\widehat{\mathbf{x}}_k)$ has value at least $1/\alpha(n)$ times the maximum $\mathcal{H}_k(\cdot)$

value for any graph of size n. Now since $\mathcal{A}$ is applied to each partition separately, the approximation is within $\alpha(\tilde{K})$ where $\tilde{K} = CK^{\rho}$ is the bound on the number of nodes in each partition.

$$\mathcal{H}(\widehat{\mathbf{x}}_k) \geq \frac{1}{\alpha(\tilde{K})}\mathcal{M}(\mathbf{x}_k^*). \tag{20}$$

By the same proof of Lemma 3 together with (20), we have that

$$\mathbb{E}[\mathcal{H}(\widehat{\mathbf{x}})] \geq \frac{1}{\alpha(\tilde{K})}\mathcal{H}(\mathbf{x}^*) - \mathbb{E}\left[\sum_{(i,j)\in\mathcal{B}} \left(\psi_{ij}^U - \psi_{ij}^L\right)\right]. \tag{21}$$

Hence we have that

$$\begin{aligned}
\mathbb{E}[\mathcal{H}(\widehat{\mathbf{x}})] &\geq \frac{1}{\alpha(\tilde{K})}\mathcal{H}(\mathbf{x}^*) - \mathbb{E}\left[\sum_{(i,j)\in\mathcal{B}} \left(\psi_{ij}^U - \psi_{ij}^L\right)\right] &(22)\\
&\overset{(a)}{\geq} \frac{1}{\alpha(\tilde{K})}\mathcal{H}(\mathbf{x}^*) - 2\varepsilon\left(\sum_{(i,j)\in E} \left(\psi_{ij}^U - \psi_{ij}^L\right)\right) &(23)\\
&\overset{(b)}{\geq} \mathcal{H}(\mathbf{x}^*)\left(\frac{1}{\alpha(\tilde{K})} - 2\varepsilon(d^*+1)\right) &(24)\\
&\overset{(c)}{\geq} \left(\frac{1}{\alpha(\tilde{K})} - \delta\right)\mathcal{H}(\mathbf{x}^*). &(25)
\end{aligned}$$

Here (a) follows from Lemma 1, (b) follows from Lemma 2, and (c) from the definition of ε for MAP inference. This completes the proof of Theorem 1(b) for MAP inference.

Completing Proof of Theorem 2. The same arguments as in the proof Theorem 1 together with Lemma 3 completes the proof of Theorem 2 for MAP inference.

Proof of Lemma 1. Now we prove Lemma 1. First, we consider property of the partition scheme applied to a generic metric space $\mathcal{G} = (V, \mathbf{d_G})$, where V is the set of points over which metric $\mathbf{d_G}$ is defined. We state the result below for any metric space (rather than restricted to a graph) as it's necessary to carry out appropriate induction based proof. Note that the algorithm the partition scheme can be applied to any metric space (not just graph as well) as it only utilizes the property of metric in it's definition. The edge set E of metric space $\mathcal{G}$ is precisely the set of all vertices that are within distance 1 of each other.

Proposition 1. *Consider a metric space $\mathcal{G} = (V, \mathbf{d_G})$ defined over n point set V, i.e. $|V| = n$. Let $\mathcal{B} = E \setminus \cup_{k=1}^{p} E_k$ be the boundary set of the partition scheme applied to $\mathcal{G}$. Then, for any $e \in E$,*

$$\mathbb{P}[e \in \mathcal{B}] \leq \varepsilon + P_K \cdot |\mathbf{B}(e, K)|,$$

where $\mathbf{B}(e, K) = \mathbf{B}_{G}(e, K)$ is the union of the two balls of radius K in $\mathcal{G}$ with respect to the $\mathbf{d_G}$ centered around the two end vertices of e, and $P_K = (1-\varepsilon)^{K-1}$.

Proof. The proof is by induction on the number of points n. When $n = 1$, the algorithm chooses only point as u_0 in the initial iteration and hence no edge can be part of the output set $\mathcal{B}$. That is, for any edge, say e,

$$\mathbb{P}[e \in \mathcal{B}] = 0 \le \varepsilon + P_K|\mathbf{B}(e, K)|.$$

Thus, we have verified the base case for induction ($n = 1$).

As induction hypothesis, suppose that the Proposition 1 is true for any graph with n nodes with $n < N$ for some $N \ge 2$. As the induction step, we wish to establish Proposition 1 for any $\mathcal{G} = (V, \mathbf{d_G})$ with $|V| = N$. For this, consider any $v \in V$. Now consider the last iteration of the the partition scheme applied to $\mathcal{G}$. The algorithm picks $i_1 \in V$ uniformly at random in the first iteration. Given e, depending on the choice of i_1 we consider three different cases (or events). We will show that in these three cases,

$$\mathbb{P}[e \in \mathcal{B}] \le \varepsilon + P_K|\mathbf{B}(e, K)|$$

holds.

Case 1. Suppose i_1 is such that $\mathbf{d_G}(i_1, e) < K$, where the distance of a point and an edge of $\mathcal{G}$ is defined as a minimum distance from the point to one of the two end-points of the edge. Call this event E_1. Further, depending on choice of random number R_1, define the following events

$$E_{11} = \{\mathbf{d_G}(i_1, e) < R_1\},\ E_{12} = \{\mathbf{d_G}(i_1, e) = R_1\},\ \text{and } E_{13} = \{\mathbf{d_G}(i_1, e) > R_1\}.$$

By the definition of the partition scheme, when E_{11} happens, e can never be a part of $\mathcal{B}$. When E_{12} happens, e is definitely a part of $\mathcal{B}$. When E_{13} happens, it is said to be left as an element of the set $\mathcal{W}_1$. This new vertex set $\mathcal{W}_1$ has points less than N. The original metric $\mathbf{d_G}$ is still considered as the metric on the points[3] of $\mathcal{W}_1$. By its definition, the partition scheme excluding the first iteration is the same as the partition scheme applied to $(\mathcal{W}_1, \mathbf{d_G})$. Therefore, we can invoke induction hypothesis which implies that if event E_{13} happens then the probability of $v \in \mathcal{B}$ is bounded above by $\varepsilon + P_K \cdot |\mathbf{B}(e, K)|$, where $\mathbf{B}(e, K)$ is the ball with respect to $(\mathcal{W}_1, \mathbf{d_G})$ which has no more than the number of points in the ball $\mathbf{B}(e, K)$ defined with respect to the original metric space $\mathcal{G}$. Finally, let us relate the $\mathbb{P}[E_{11}|E_2]$ with $\mathbb{P}[E_{12}|E_1]$. Suppose $\mathbf{d_G}(i_1, e) = \ell < K$. By the definition of probability distribution of $\mathbf{Q}$, we have

$$\mathbb{P}[E_{12}|E_1] = \varepsilon(1-\varepsilon)^{\ell-1}, \tag{26}$$

$$\begin{aligned}\mathbb{P}[E_{11}|E_1] &= (1-\varepsilon)^{K-1} + \sum_{j=\ell+1}^{K-1} \varepsilon(1-\varepsilon)^{j-1} \\ &= (1-\varepsilon)^{\ell}.\end{aligned} \tag{27}$$

That is,

$$\mathbb{P}[E_{12}|E_1] = \frac{\varepsilon}{1-\varepsilon}\mathbb{P}[E_{11}|E_1].$$

3. Note the following subtle but crucial point. We are not changing the metric $\mathbf{d_G}$ after we remove points from the original set of points.

Let $q \overset{\triangle}{=} \mathbb{P}[E_{11}|E_1]$. Then,

$$\begin{aligned}\mathbb{P}[e \in \mathcal{B}|E_1] &= \mathbb{P}[e \in \mathcal{B}|E_{11} \cap E_1]\mathbb{P}[E_{11}|E_1] + \mathbb{P}[e \in \mathcal{B}|E_{12} \cap E_1]\mathbb{P}[E_{12}|E_1] \\ &\quad + \mathbb{P}[e \in \mathcal{B}|E_{13} \cap E_1]\mathbb{P}[E_{13}|E_1] \\ &\leq 0 \times q + 1 \times \frac{\varepsilon q}{1-\varepsilon} + (\varepsilon + P_K|\mathbf{B}(e,K)|)\left(1 - \frac{q}{1-\varepsilon}\right) \\ &= \varepsilon + P_K|\mathbf{B}(e,K)| + \frac{q}{1-\varepsilon}(\varepsilon - \varepsilon - P_K|\mathbf{B}(e,K)|) \\ &= \varepsilon + P_K|\mathbf{B}(e,K)| - \frac{qP_K|\mathbf{B}(e,K)|}{1-\varepsilon} \\ &\leq \varepsilon + P_K|\mathbf{B}(e,K)|. \end{aligned} \tag{28}$$

Case 2. Now, suppose $i_1 \in V$ is such that $\mathbf{d_G}(i_1, e) = K$. We will call this event E_2. Further, define the event $E_{21} = \{R_1 = K\}$. Due to the independence of selection of R_1, $\mathbb{P}[E_{21}|E_2] = P_K$. Under the event $E_{21} \cap E_2$, $e \in \mathcal{B}$ with probability 1. Therefore,

$$\begin{aligned}\mathbb{P}[e \in \mathcal{B}|E_2] &= \mathbb{P}[e \in \mathcal{B}|E_{21} \cap E_2]\mathbb{P}[E_{21}|E_2] + \mathbb{P}[e \in \mathcal{B}|E_{21}^c \cap E_2]\mathbb{P}[E_{21}^c|E_2] \\ &= 1 \times P_K + \mathbb{P}[e \in \mathcal{B}|E_{21}^c \cap E_2](1 - P_K). \end{aligned} \tag{29}$$

Under the event $E_{21}^c \cap E_2$, we have $e \in \mathcal{W}_1$, and the remaining metric space $(\mathcal{W}_1, \mathbf{d_G})$. This metric space has $< N$ points. Further, the ball of radius K around e with respect to this new metric space has at most $|\mathbf{B}(e,K)| - 1$ points (this ball is with respect to the original metric space $\mathcal{G}$ on N points). Now we can invoke the induction hypothesis for this new metric space to obtain

$$\mathbb{P}[e \in \mathcal{B}|E_{21}^c \cap E_2] \quad \leq \quad \varepsilon + P_K \cdot (|\mathbf{B}(e,K)| - 1). \tag{30}$$

From (29) and (30), we have

$$\begin{aligned}\mathbb{P}[e \in \mathcal{B}|E_3] &\leq P_K + (1 - P_K)(\varepsilon + P_K \cdot (|\mathbf{B}(e,K)| - 1)) \\ &= \varepsilon(1 - P_K) + P_K|\mathbf{B}(e,K)| + P_K^2(1 - |\mathbf{B}(e,K)|) \\ &\leq \varepsilon + P_K|\mathbf{B}(e,K)|. \end{aligned}$$

In above, we have used the fact that $|\mathbf{B}(e,K)| \geq 1$ (or else, the bound was trivial to begin with).

Case 3. Finally, let E_3 be the event that $\mathbf{d_G}(i_1, e) > K$. Then, at the end of the first iteration of the algorithm, we again have the remaining metric space $(\mathcal{W}_1, \mathbf{d_G})$ such that $|\mathcal{W}_1| < N$. Hence, as before, by induction hypothesis we have

$$\mathbb{P}[e \in \mathcal{B}|E_3] \leq \varepsilon + P_K|\mathbf{B}(e,K)|.$$

Now, the three cases are exhaustive and disjoint. That is, $\cup_{i=1}^3 E_i$ is the universe. Based on the above discussion, we obtain the following.

$$\begin{aligned}\mathbb{P}[e \in \mathcal{B}] &= \sum_{i=1}^{3} \mathbb{P}[e \in \mathcal{B}|E_i]\mathbb{P}[E_i] \\ &\leq \left(\max_{i=1}^{3} \mathbb{P}[e \in \mathcal{B}|E_i]\right)\left(\sum_{i=1}^{3} \mathbb{P}[E_i]\right) \\ &\leq \varepsilon + P_K \cdot |\mathbf{B}(e,K)|. \end{aligned} \tag{31}$$

This completes the proof of Proposition 1. □

Now, we will use Proposition 1 to complete the proof of Lemma 1. The definition of growth rate implies that,

$$|\mathbf{B}(e,K)| \leq C \cdot K^{\rho}.$$

From the definition $P_K = (1-\varepsilon)^{K-1}$, we have

$$P_K \, |\mathbf{B}(e,K)| \leq C(1-\varepsilon)^{K-1} K^{\rho}.$$

Therefore, to show Lemma 1, it is sufficient to show that our definition of K satisfies the following Lemma.

Lemma 4. *We have that*

$$C(1-\varepsilon)^{K-1} K^{\rho} \leq \varepsilon.$$

Proof. We will show the following equivalent inequality.

$$(K-1)\log(1-\varepsilon)^{-1} \geq \rho \log K + \log C + \log \frac{1}{\varepsilon}. \tag{32}$$

First, note that for all $\varepsilon \in (0,1)$,

$$\log(1-\varepsilon)^{-1} \geq \log(1+\varepsilon) \geq \frac{\varepsilon}{2}.$$

Hence to prove (32), it is sufficient to show that

$$K \geq \frac{2\rho}{\varepsilon}\log K + \frac{2}{\varepsilon}\log C + \frac{2}{\varepsilon}\log\frac{1}{\varepsilon} + 1. \tag{33}$$

Recall that

$$K = K(\varepsilon,\rho) = \frac{8\rho}{\varepsilon}\log\left(\frac{8\rho}{\varepsilon}\right) + \frac{4}{\varepsilon}\log C + \frac{4}{\varepsilon}\log\frac{1}{\varepsilon} + 2.$$

From the definition of K, we will show that

$$\frac{K}{2} \geq \frac{2\rho}{\varepsilon}\log K$$

and

$$\frac{K}{2} \geq \frac{2}{\varepsilon}\log C + \frac{2}{\varepsilon}\log\frac{1}{\varepsilon} + 1,$$

which will prove (33). The following is straightforward:

$$\frac{K}{2} \geq \frac{2}{\varepsilon}\log C + \frac{2}{\varepsilon}\log\frac{1}{\varepsilon} + 1. \tag{34}$$

Now, let $\widehat{K} = \frac{8\rho}{\varepsilon}\log\left(\frac{8\rho}{\varepsilon}\right)$. Then

$$\frac{\widehat{K}}{2} = \frac{4\rho}{\varepsilon}\log\left(\frac{8\rho}{\varepsilon}\right) \geq \frac{2\rho}{\varepsilon}\left(\log\left(\frac{8\rho}{\varepsilon}\right) + \log\log\left(\frac{8\rho}{\varepsilon}\right)\right) = \frac{2\rho}{\varepsilon}\log\widehat{K}.$$

That is, $\frac{\widehat{K}}{2} - \frac{2\rho}{\varepsilon}\log\widehat{K} \geq 0$. Since the function $\phi(x) = \frac{x}{2} - \frac{2\rho}{\varepsilon}\log x$ is an increasing function of x when $x \geq \frac{4\rho}{\varepsilon}$, and from the fact that $K \geq \widehat{K} \geq \frac{4\rho}{\varepsilon}$, we have

$$\frac{K}{2} \geq \frac{2\rho}{\varepsilon}\log K. \tag{35}$$

From (34) and (35), we have (33), which completes the proof of Lemma 4. □

6. Proofs of Theorems 1, 2: Modularity optimization

In this Section, we prove Theorem 1, and Theorem 2 for modularity optimization.

Lower bound on $\mathcal{M}^*$. Here we provide a lower bound on $\mathcal{M}^*$ that will be useful to obtain multiplicative approximation property.

Lemma 5. *Let $\mathcal{M}^* = \max_{\chi} \mathcal{M}(\chi)$ denote the maximum value of modularity for graph G. Then,*

$$\mathcal{M}^* \geq \frac{1}{2(2C-1)}\Big(1 - \frac{C^2}{2m}\Big).$$

Proof. Since the graph has polynomial growth with degree ρ and associated constant C, it follows that the number of nodes within one hop of any node $i \in V$ (i.e. its immediate neighbors) is at most C. That is, $d_i \leq C$ for all $i \in V$. Given this bound, it follows that there exists a matching of size at least $m/(2C-1)$ in G. Given such a matching, consider the following clustering (coloring). Each edge in the matching represent a community of size 2, while all the nodes that are unmatched lead to community of size 1. By definition, the individual (unmatched) nodes contribute 0 to the modularity. The nodes that are part of the two node communities, each contribute at least $\frac{1}{2m}\big(1 - \frac{C^2}{2m}\big)$ since vertex degree of each node is bounded above by C. Since there are $m/(2C-1)$ edges in the matching, it follows that the net modularity of such community assignment is at least $\frac{1}{2(2C-1)}\big(1 - \frac{C^2}{2m}\big)$. This completes the proof of Lemma 5 (Similar result, with tighter constant, follows from Han (2008)). □

Decomposition of $\mathcal{M}^*$. Here we show that by maximizing modularity on a partition of V separately, the resulting clustering has modularity as good as that of optimal partitioning with penalty in terms of the edges across partitions. To that end, let $V = V_1 \cup \cdots \cup V_p$ be a partition of V, i.e. $V_i \cap V_j = \emptyset$ for $i \neq j$. Let $G_k = (V_k, E_k)$, where $E_k = (V_k \times V_k) \cap E$, denote the subgraph of G for $1 \leq k \leq p$. Let χ^k be a coloring (clustering) of G_k with maximum modularity. Let χ^* be a coloring of G with maximum modularity ($\mathcal{M}^*$) and let $\chi^{*,k}$ be the restriction of χ^* to G_k. Let $\widehat{\chi}$ denote the clustering of G obtained by taking union of clusterings $\chi^1, \ldots, \chi^p$. Then we claim the following.

Lemma 6. *For any partition $V = V_1 \cup \cdots \cup V_p$,*

$$\mathcal{M}(\widehat{\chi}) \ \geq \mathcal{M}(\chi^*) - \frac{1}{2m}|E \setminus \cup_{k=1}^p E_k|.$$

Proof. Consider the following:

$$\begin{aligned}
2m\, \mathcal{M}(\widehat{\chi}) &= \sum_{i,j \in V} \mathbf{1}_{\{\widehat{\chi}(i)=\widehat{\chi}(j)\}}\Big(A_{ij} - \frac{d_i d_j}{2m}\Big) \overset{(a)}{=} \sum_{k=1}^{p} \sum_{i,j \in V_k} \mathbf{1}_{\{\chi^k(i)=\chi^k(j)\}}\Big(A_{ij} - \frac{d_i d_j}{2m}\Big) \\
&\overset{(b)}{\geq} \sum_{k=1}^{p} \sum_{i,j \in V_k} \mathbf{1}_{\{\chi^{*,k}(i)=\chi^{*,k}(j)\}}\Big(A_{ij} - \frac{d_i d_j}{2m}\Big) \\
&= \sum_{i,j \in V} \mathbf{1}_{\{\chi^*(i)=\chi^*(j)\}}\Big(A_{ij} - \frac{d_i d_j}{2m}\Big) - \sum_{(i,j) \in V^2 \setminus \cup_{k=1}^p V_k^2} \mathbf{1}_{\{\chi^*(i)=\chi^*(j)\}}\Big(A_{ij} - \frac{d_i d_j}{2m}\Big) \qquad (36) \\
&\geq \sum_{i,j \in V} \mathbf{1}_{\{\chi^*(i)=\chi^*(j)\}}\Big(A_{ij} - \frac{d_i d_j}{2m}\Big) - |E \setminus \cup_{k=1}^p E_k|, \qquad (37)
\end{aligned}$$

where the last inequality follows because the term inside the summation in (36) is positive only if $A_{ij} = 1$, i.e. $(i, j) \in E$ or else it is negative. Therefore, for the purpose of lower bound, we only need to worry about $(i, j) \in E$ such that $(i, j) \notin \cup_{k=1}^{p} V_k \times V_k$. This is precisely equal to $E \backslash \cup_{k=1}^{p} E_k$. The (a) follows because $\widehat{\chi}$, by definition, assigns nodes in V^i and V^j for $i \neq j$ to different clusters. The (b) follows because χ^k has maximum modularity in G_k and hence it is at least as large (in terms of modularity) as that of the $\chi^{*,k}$, the restriction of χ^* to G_k. This completes the proof of Lemma 6 since the first term in (37) is precisely $2m\mathcal{M}(\chi^*) = 2m\mathcal{M}^*$. $\square$

Approximation factor for $\mathcal{M}(\widehat{\chi})$. Let $\beta = |E \backslash \cup_{k=1}^{p} E_k|/m$ denote the fraction of edges that are across partitions for a given partition $V = V_1 \cup \cdots \cup V_p$. Then, from Lemmas 5 and 6, it follows that for $m \geq C^2$,

$$\mathcal{M}(\widehat{\chi}) \geq \mathcal{M}(\chi^*)\Big(1 - \frac{\beta}{2\mathcal{M}(\chi^*)}\Big) \geq \mathcal{M}(\chi^*)\Big(1 - 2(2C-1)\beta\Big). \tag{38}$$

Therefore, if $2(2C-1)\beta \leq \delta$, then $\mathcal{M}(\widehat{\chi})$ is at least $\mathcal{M}^* \cdot (1-\delta)$. Now from Lemma 1 and the linearity of expectation, we have

$$\mathbb{E}\big[|E \backslash \cup_{k=1}^{p} E_k|\big] \leq \frac{\delta}{2(2C-1)} m. \tag{39}$$

Completing Proof of Theorem 1(a). When $\mathcal{A}$ produces exact solution to the modularity optimization for each partition, the resulting solution of our algorithm is $\widehat{\chi}$. Therefore, from (38) and (39), it follows that

$$\mathbb{E}[\mathcal{M}(\widehat{\chi})] \geq \mathcal{M}(\chi^*)(1-\delta). \tag{40}$$

Completing Proof of Theorem 1(b). Suppose we use an approximation procedure $\mathcal{A}$ to produce clustering on each partition in our algorithm. Let $\mathcal{A}$ be such that the clustering produced has modularity at least $1/\alpha(n)$ times the optimal modularity for any graph of size n. Now since $\mathcal{A}$ is applied to each partition separately, the approximation is within $\alpha(\tilde{K})$ where $\tilde{K} = CK^\rho$ is the bound on the number of nodes in each partition. Let $\tilde{\chi}^1, \ldots, \tilde{\chi}^p$ be the clustering (coloring) produced by $\mathcal{A}$ on graphs $G_1, \ldots, G_p$. Then by the approximation property of $\mathcal{A}$, we have

$$\mathcal{M}(\tilde{\chi}^k) \geq \frac{1}{\alpha(\tilde{K})}\mathcal{M}(\chi^k). \tag{41}$$

Therefore, for the overall clustering $\tilde{\chi}$ obtained as union of $\tilde{\chi}^1, \ldots, \tilde{\chi}^p$, we have

$$\mathcal{M}(\tilde{\chi}) = \sum_{k=1}^{p} \mathcal{M}(\tilde{\chi}^k) \geq \frac{1}{\alpha(\tilde{K})} \sum_{k=1}^{p} \mathcal{M}(\chi^k) = \frac{1}{\alpha(\tilde{K})}\mathcal{M}(\widehat{\chi}). \tag{42}$$

Since $\mathbb{E}[\mathcal{M}(\widehat{\chi})]$ is at least $(1-\delta)\mathcal{M}^*$, it follows that $\mathbb{E}[\mathcal{M}(\tilde{\chi})] \geq \frac{(1-\delta)}{\alpha(\tilde{K})}\mathcal{M}^*$.

Completing Proof of Theorem 2. Lemma 6 directly proves Theorem 2(a), and the same arguments as in the proof Theorem 1(b) completes the proof of Theorem 2(b).

7. Conclusion

In recent years, it has become increasingly important to design distributed high-performance graph computation algorithms that can deal with large-scale networked data in a cloud-like distributed computation architecture. Inspired by this, in this paper, we have introduced Partition-Merge, a simple meta-algorithm, that takes an existing centralized algorithm and produces a distributed implementation. The resulting distributed implementation, with the underlying graph having polynomial growth property, runs in essentially linear time and is as good as, and sometimes even better than the centralized algorithm.

The algorithm is applicable to any graph in general, and its computation time as well as performance guarantees depend on the underlying graph structure – interestingly enough, we have evaluated the performance guarantees for any graph. We strongly believe that such an algorithmic approach would be of great value for developing large-scale cloud-based graph computation facilities.

Acknowledgments

Part of this work appeared in the preliminary version Jung et al. (2009). This work is supported in parts by Army Research Office under MURI Award 58153-MA-MUR, and in part by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2012032786).

References

B. Awerbuch, M. Luby, A.V. Goldberg, and S.A. Plotkin. Network decomposition and locality in distributed computation. In *Foundations of Computer Science (FOCS)*. IEEE, 1989.

M. Bayati, D. Shah, and M. Sharma. Maximum weight matching via max-product belief propagation. In *IEEE ISIT*, 2005.

M. Bayati, D. Shah, and M. Sharma. Max-Product for Maximum Weight Matching: Convergence, Correctness, and LP Duality. *IEEE Transactions on Information Theory*, 54(3):1241–1251, 2008.

V. Blondel, G. Krings, and I. Thomas. Regions and borders of mobile telephony in belgium and in the brussels metropolitan zone. *Brussels Studies*, 42(4), 2010.

V.D. Blondel, J.L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008:P10008, 2008.

B. DasGupta and D. Desai. On the complexity of newman's community finding approach for biological and social networks. *Arxiv preprint arXiv:1102.0969*, 2011.

R. Gummadi, K. Jung, D. Shah, and R. Sreenivas. Computing the capacity region of a wireless network. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2009.

A. Gupta, R. Krauthgamer, and J.R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *Foundations of Computer Science (FOCS)*, 2003.

Y. Han. Matching for graphs of bounded degree. *Frontiers in Algorithmics*, pages 171–173, 2008.

A. Hassidim, J.A. Kelner, H.N. Nguyen, and K. Onak. Local graph partitions for approximation and testing. In *Foundations of Computer Science (FOCS)*, 2009.

B. Huang and T. Jebara. Loopy belief propagation for bipartite maximum weight b-matching. *Artificial Intelligence and Statistics (AISTATS)*, 2007.

K. Jung and D. Shah. Local algorithms for approximate inference in minor-excluded graphs. In *Annual Conference on Neural Information Processing Systems (NIPS)*, 2007.

K. Jung, P. Kohli, and D. Shah. Local rules for global map: When do they work? *Advances in Neural Information Processing Systems (NIPS)*, 22:871–879, 2009.

P. Klein, S.A. Plotkin, and S. Rao. Excluded minors, network decomposition, and multicommodity flow. In *ACM symposium on Theory of computing (STOC)*, 1993.

M.E.J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577, 2006.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA: Morgan Kaufmann, 1988.

D. Peleg. *Distributed computing: a locality-sensitive approach*, volume 5. Society for Industrial Mathematics, 2000.

S. Sanghavi, D. Shah, and A. Willsky. Message-passing for Maximum Weight Independent Set. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

D. Sontag and T. Jaakkola. Tree block coordinate descent for map in graphical models. *Journal of Machine Learning Research - Proceedings Track*, 5:544–551, 2009.

R. Swendsen and J. Wang. Nonuniversal critical dynamics in monte carlo simulations. *Phys. Rev. Letter.*, 58:86–88, 1987.

D.l Tarlow, D. Batra, P. Kohli, and V. Kolmogorov. Dynamic tree block coordinate ascent. In *ICML*, 2011.

M. J. Wainwright, T. Jaakkola, and A. S. Willsky. Map estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Transactions on Information Theory*, 2005.

J. Yedidia, W. Freeman, and Y. Weiss. Generalized belief propagation. *Mitsubishi Elect. Res. Lab., TR-2000-26*, 2000.