Distributed Graph Clustering by Load Balancing

He Sun
University of Bristol
Bristol, UK
h.sun@bristol.ac.uk

Luca Zanetti
University of Bristol
Bristol, UK
luca.zanetti@bristol.ac.uk

ABSTRACT

Graph clustering is a fundamental computational problem with a number of applications in algorithm design, machine learning, data mining, and analysis of social networks. Over the past decades, researchers have proposed a number of algorithmic design methods for graph clustering. However, most of these methods are based on complicated spectral techniques or convex optimisation, and cannot be applied directly for clustering many networks that occur in practice, whose information is often collected on different sites. Designing a simple and distributed clustering algorithm is of great interest, and has wide applications for processing big datasets.

In this paper we present a simple and distributed algorithm for graph clustering: for a wide class of graphs that are characterised by a strong cluster-structure, our algorithm finishes in a poly-logarithmic number of rounds, and recovers a partition of the graph close to an optimal partition. The main component of our algorithm is an application of the *random matching model* of load balancing, which is a fundamental protocol in distributed computing and has been extensively studied in the past 20 years. Hence, our result highlights an intrinsic and interesting connection between graph clustering and load balancing.

At a technical level, we present a purely algebraic result characterising the *early behaviours* of load balancing processes for graphs exhibiting a cluster-structure. We believe that this result can be further applied to analyse other gossip processes, such as rumour spreading and averaging processes.

CCS CONCEPTS

• **Theory of computation → Graph algorithms analysis**; **Distributed algorithms**; **Random walks and Markov chains**;

KEYWORDS

graph clustering; load balancing; spectral graph theory

1 Introduction

Analysis of large-scale networks has brought significant advances to our understanding of complex systems. One of the most relevant features possessed by networks occurring in practice is a strong cluster-structure, i.e., an organisation of nodes into clusters such that nodes within the same cluster are highly connected in contrast to nodes from different clusters. Recovering this cluster-structure is the aim of graph clustering, which is an important research topic in many disciplines, including computer science, physics, biology, and sociology. For instance, graph clustering is widely used in finding communities in social networks, webpages dealing with similar topics, and proteins having the same specific function within the cell in protein-protein interaction networks [13]. Most algorithms for graph clustering, however, require advanced algorithm design techniques such as spectral methods, or convex optimisation, which may make the algorithms difficult to be implemented in the setting of big data, where graphs may be allocated in sites that are physically remote. For this reason, designing a simple and distributed algorithm is of great interest in practice, and has received considerable attention in recent years [6, 20, 31].

In this work we propose a simple and distributed graph clustering algorithm that is mainly based on the following classical load balancing process (random matching model): assume that there is an application running on a parallel network with n processors. Every processor has initially a certain amount of loads (jobs) and the processors are connected by an arbitrary graph G. A load balancing process in the random matching model consists of synchronous rounds: in each round a random matching of G is generated in a distributed way, and every two matched nodes average their loads evenly. This process continues until every node has almost the same amount of load. Despite its low communication cost (at most $\lfloor n/2 \rfloor$ edges are involved in each round for load distribution) and highly distributed properties (every node only contacts its neighbors in the entire process), load balancing has been proven to be very efficient [27], and has been widely used in various domains, including scheduling [30], hashing [23], routing [10], and numerical computation such as solving partial differential equations [32].

1.1 Structure of Clusters

Let $G = (V, E)$ be an undirected graph with n nodes. For any set S, let the conductance of S be

$$\phi_G(S) \triangleq \frac{|E(S, V \setminus S)|}{\mathrm{vol}(S)},$$

where $E(S, V \setminus S)$ is the set of edges between S and $V \setminus S$, and $\mathrm{vol}(S)$ is the number of edges with at least one endpoint in S. Intuitively, nodes in S form a cluster if $\phi_G(S)$ is small, i.e., there are few edges connecting the nodes of S to the nodes in $V \setminus S$. We call subsets of nodes (i.e. *clusters*) $A_1, \ldots, A_k$ a k*-way partition* of G if $A_i \cap A_j = \emptyset$ for different i and j, and $\bigcup_{i=1}^{k} A_i = V$. Moreover, we define the k*-way expansion constant* by

$$\rho(k) \triangleq \min_{\text{partition } A_1,\ldots,A_k} \max_{1 \leqslant i \leqslant k} \phi_G(A_i).$$

SPAA '17, July 24-26, 2017, Washington DC, USA

ACM ISBN 978-1-4503-4593-4/17/07…$15.00
https://doi.org/10.1145/3087556.3087569

Computing the exact value of $\rho(k)$ is coNP-hard, and a sequence of results show that $\rho(k)$ can be approximated by algebraic quantities relating to the matrices of G. For instance, Lee et al. [22] proved the following high-order Cheeger inequality:

$$\frac{1-\lambda_k}{2} \leqslant \rho(k) \leqslant O\left(k^2\right)\sqrt{1-\lambda_k}, \tag{1}$$

where $1 = \lambda_1 \geqslant \cdots \geqslant \lambda_n \geqslant -1$ are the eigenvalues of the random walk matrix of G. Based on (1), we know that a large gap between $(1-\lambda_{k+1})$ and $\rho(k)$ guarantees (i) existence of a k-way partition $S_1, \ldots S_k$ with bounded $\phi_G(S_i) \leqslant \rho(k)$, and (ii) any $(k+1)$-way partition $A_1, \ldots, A_{k+1}$ of G contains a subset A_i with significantly higher conductance $\rho(k+1) \geqslant (1-\lambda_{k+1})/2$ compared with $\rho(k)$. Peng et al. [25] formalise these observations by defining the parameter

$$\Upsilon \triangleq \frac{1-\lambda_{k+1}}{\rho(k)},$$

and shows that a suitable lower bound on the value of Υ implies that G has k well-defined clusters.

Throughout the rest of the paper, we assume that $S_1, \ldots, S_k$ is a k-way partition that achieves $\rho(k)$, and there is a known threshold $\beta > 0$ such that $|S_i| \geqslant \beta n$ for any $1 \leqslant i \leqslant k$, i.e., the clusters have balanced size. We say G is *well-clustered* if

$$\Upsilon = \omega\left(k^5 \frac{1}{\beta^3}\log^4\frac{1}{\beta}\log n\right). \tag{2}$$

Notice that (2) can be written as $\Upsilon = \omega(\log n)$ when the number of clusters k is a constant, and the sizes of all the clusters are almost balanced. For simplicity, we assume that G is a d-regular graph, and we will discuss in Section 4.5 how to generalise our result to almost-regular graphs, as long as the ratio between the maximum and minimum degree is upper bounded by a constant.

1.2 Our Results

We investigate the power of random matching model of load balancing, a widely studied process in distributed computing [4, 11, 15, 17, 26, 27]. We propose a high-dimensional version of this random matching model, and show that the proposed algorithm can be used for graph clustering. Our algorithm is decentralised, and very easy to implement. Moreover, our approach corresponds to a natural centralised algorithm for graph clustering, which is also new to the best of our knowledge. Our main result is summarised as follows:

Theorem 1.1. *There exists a distributed algorithm such that, for any well-clustered graph $G = (V, E)$ with n nodes and k clusters $S_1, \ldots, S_k$ that satisfies* (2)*, finishes in $T \triangleq \Theta\left(\frac{\log n}{1-\lambda_{k+1}}\right)$ rounds and, with constant probability, at the end of the algorithm the following statements hold:*

(1) *Each node v receives a label ℓ_v such that the total number of misclassified nodes is $o(n)$, i.e., there exists a permutation of the labels σ such that*

$$\left|\bigcup_{i=1}^{k}\{v | v \in S_i \text{ and } \ell_v \neq \sigma(i)\}\right| = o(n);$$

(2) *The total information exchanged among these n nodes, i.e., the message complexity, is $O(T \cdot n \cdot k \log k)$ words.*

An important scenario for graph clustering is the case where G consists of $k = \Theta(1)$ clusters $S_1, \ldots, S_k$, where $|S_i| = \Theta(n/k)$, every $G[S_i]$ is a spectral expander, and has conductance $\phi_G(S_i) = O(1/\mathrm{poly}\log n)$ for $i = 1, \ldots, k$. It is easy to verify that for such graph G our gap assumption (2) on Υ holds, and our algorithm finishes in $O(\log n)$ rounds with message complexity $O(n \log n)$. Moreover, the non-distributed version of our algorithm runs in $O(n \log n)$ time once we have an oracle which outputs a random neighbour of any node. That is, when the input graph is d-regular with $d = \omega(\log n)$, our algorithm runs in *sub-linear time.* This example shows a clear distinction between our algorithm and most other graph clustering algorithms, which usually require at least linear-time. Hence, the techniques presented in our paper might be of interest for designing algorithms for other models of computation as well, e.g., local algorithms, and algorithms for property testing.

1.3 Related Work

There is a large amount of literature on graph clustering, and our work is most closely related to efficient algorithms for graph clustering under different formulations of clusters. Oveis Gharan and Trevisan [24] formulate the notion of clusters with respect to the *inner* and *outer* conductance: a cluster S should have low outer conductance, and the conductance of the induced subgraph by S should be high. Under a assumption between λ_{k+1} and λ_k, they present a polynomial-time algorithm which finds a k-way partition $\{A_i\}_{i=1}^k$ that satisfies the inner- and outer-conductance condition. To ensure that every A_i has high inner conductance, they assume that $\lambda_{k+1} \geqslant \mathrm{poly}(k)\lambda_k^{1/4}$, which has a stronger polynomial dependency on λ_k.

Another line of research closely related to our result is the design of local algorithms for finding a subset of nodes of low conductance, e.g., [16, 29]. In particular, Allen-Zhu et al. [1] studies a cluster structure with a gap assumption similar to ours, and presents a local algorithm with better approximation guarantee than previously known algorithms under that gap assumption. However, there is substantial difference between our algorithm and most local algorithms [1, 16, 29] for the following reasons: (1) We need to run a local algorithm k times in order to find k clusters. However, as the output of each execution of a local algorithm only returns an *approximate* cluster, the approximation ratio of the final output cluster might not be guaranteed when the value of k is large. (2) For many instances, our algorithm requires only a poly-logarithmic number of rounds, while local algorithms run in time proportional to the volume of the output set. It is unclear how these algorithms could finish in a poly-logarithmic number of rounds, even if we were able to implement them in the distributed setting.

Recently, Becchetti et al. [3] studies a distributed process to partition an almost-regular graph into clusters, and their analysis focuses mostly on graphs generated randomly from stochastic block models. In contrast to ours, their algorithm requires each node to exchange information with all of its neighbours in each round, and has significantly higher communication cost for a dense graph.

We also notice that the distributed algorithm presented in Kempe et al. [21] for computing the top k eigenvectors of the adjacency matrix of a graph can be applied for graph clustering. Their algorithm is, however, much more involved than ours. Moreover, for an input graph G of n nodes, the number of rounds required in their algorithm is proportional to the mixing time of a random walk in G. For a graph consisting of multiple expanders connected by a few edges, their algorithm requires a polynomial number of rounds, while ours only requires a poly-logarithmic number of rounds.

Finally, we point out that our work is closely related to multiple random walks [2, 9, 12], other variants of load balancing processes [14], and runtime analysis of gossip algorithms [7, 8, 18, 19].

1.4 Organisation

The remaining part of the paper is organised as follows: Section 2 lists the notations used in the paper, and discusses necessary knowledge of load balancing. Section 3 presents our algorithm, and Section 4 gives a detailed analysis of our algorithm.

2 Preliminaries

2.1 Notations

Let $G = (V, E)$ be an undirected graph with n nodes and m edges. For any set $S, T \subseteq V$, we define $E(S, T)$ to be the set of edges between S and T, i.e., $E(S, T) \triangleq \{\{u, v\} | u \in S \text{ and } v \in T\}$. For two sets X and Y, the symmetric difference of X and Y is defined as $X \triangle Y \triangleq (X \setminus Y) \cup (Y \setminus X)$.

For any d-regular graph G, the random walk matrix of G is defined by $\mathbf{P} = (1/d) \cdot \mathbf{A}$, where $\mathbf{A}$ is the adjacency matrix of G defined by $\mathbf{A}_{u,v} = 1$ if $\{u, v\} \in E(G)$, and $\mathbf{A}_{u,v} = 0$ otherwise. For this matrix, we will denote its n eigenvalues with $\lambda_1 \geqslant \cdots \geqslant \lambda_n$, with their corresponding orthonormal eigenvectors $f_1, \ldots, f_n$.

For any set S of nodes, let $\chi_S \in \mathbb{R}^n$ be the normalised indicator vector of S, where $\chi_S(v) = 1/|S|$ if $v \in S$, and $\chi_S(v) = 0$ otherwise. In particular, we simply write χ_v instead of $\chi_{\{v\}}$ when the set $S = \{v\}$. The Euclidean norm of any vector $x \in \mathbb{R}^n$ is defined by $\|x\| \triangleq \sqrt{\sum_{i=1}^n x_i^2}$, and the spectral norm of any matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ is defined as

$$\|\mathbf{M}\| \triangleq \max_{x \in \mathbb{R}^n \setminus \{\mathbf{0}\}} \frac{\|\mathbf{M}x\|}{\|x\|}.$$

Given two symmetric $n \times n$ matrices A, B, we write $A \preceq B$ if $x^\intercal A x \leqslant x^\intercal B x$ holds for any $x \in \mathbb{R}^n$.

2.2 The Matching Model for Load Balancing

One of the key components used in our algorithm is the random matching model for load balancing [5, 26, 27], in which one generates a random matching in each round and every two matched nodes balance their loads evenly. There are several simple and distributed randomised protocols to generate such matching, and in the present paper we use the following protocol [5]: (1) every node is either *active* or *non-active* with probability 1/2; (2) every *active* node chooses one of its neighbours v uniformly at random; (3) every non-active node v chosen by exactly one of its neighbours is included in the matching together with its neighbour u.

We use a matching matrix $\mathbf{M}^{(t)} \in \mathbb{R}^{n \times n}$ to express the matching used in round t: for every matched nodes u and v, we have $\mathbf{M}^{(t)}_{u,u} \triangleq 1/2$, $\mathbf{M}^{(t)}_{v,v} \triangleq 1/2$, and $\mathbf{M}^{(t)}_{u,v} \triangleq 1/2$, $\mathbf{M}^{(t)}_{v,u} \triangleq 1/2$; if u is not matched, then $\mathbf{M}^{(t)}_{u,u} \triangleq 1$ and $\mathbf{M}^{(t)}_{u,v} \triangleq 0$ if $u \neq v$. With slight abuse of notation, we also use $\mathbf{M}^{(t)}$ to express the set of edges included in the matching in round t.

The following lemma describes the properties of matrix $\mathbf{M}^{(t)}$, and is originally proven in [5].

Lemma 2.1 ([5]). *Let $\bar{d} = \left(1 - \frac{1}{2d}\right)^{d-1}$. Then, the following statements hold for any $t \geqslant 1$:*

(1) $\mathbb{E}\left[\mathbf{M}^{(t)}\right] = \left(1 - \frac{\bar{d}}{4}\right)\mathbf{I} + \frac{\bar{d}}{4} \cdot \mathbf{P}$;

(2) $\mathbf{M}^{(t)}$ is a projection matrix, i.e., $\left(\mathbf{M}^{(t)}\right)^2 = \mathbf{M}^{(t)}$;

Proof. We start with the first statement. By definition, it holds for any edge $\{u, v\}, u \neq v$, that

$$\begin{aligned}
&\mathbb{P}\left[\, \{u, v\} \text{ is included in a matching in round } t \,\right] \\
&= 2 \cdot \mathbb{P}\left[\, u \text{ is active} \,\right] \cdot \mathbb{P}\left[\, v \text{ is non-active} \,\right] \\
&\quad \cdot \mathbb{P}\left[\, \{u, v\} \text{ is chosen as a matching} \,\right] \\
&= 2 \cdot \frac{1}{4} \cdot \frac{1}{d}\left(1 - \frac{1}{2d}\right)^{d-1} \\
&= \frac{1}{2} \cdot \frac{\bar{d}}{d}.
\end{aligned}$$

Hence, we have for any edge $\{u, v\}, u \neq v$, that

$$\begin{aligned}
\mathbb{E}\left[\mathbf{M}^{(t)}_{u,v}\right] &= \frac{1}{2} \cdot \mathbb{P}\left[\, \{u, v\} \text{ is included in a matching in round } t \,\right] \\
&= \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{\bar{d}}{d} = \frac{\bar{d}}{4} \cdot \mathbf{P}_{u,v}.
\end{aligned}$$

Similarly, we have for any vertex u that

$$\begin{aligned}
\mathbb{E}\left[\mathbf{M}^{(t)}_{u,u}\right] &= \frac{1}{2} \cdot \mathbb{P}\left[\, u \text{ is included in a matching in round } t \,\right] \\
&\quad + \mathbb{P}\left[\, u \text{ is not included in a matching} \,\right] \\
&= \frac{1}{2} \cdot \frac{\bar{d}}{2} + \left(1 - \frac{\bar{d}}{2}\right) = 1 - \frac{\bar{d}}{4}.
\end{aligned}$$

Combining these two equations gives us the first statement.

The second statement follows from the fact that, for any $x \in \mathbb{R}^n$, $\mathbf{M}^{(t)}x$ is the projection of x on the subspace

$$\left\{ y \in \mathbb{R}^n \mid y(u) = y(v) \text{ for any } \{u, v\} \in \mathbf{M}^{(t)} \right\}.$$

■

3 Algorithm

Now we present the distributed algorithm for graph clustering. For completeness, in Section 3.1 we will first present the detailed implementation of our algorithm in the distributed setting. In Section 3.2 we will discuss our algorithm in a more abstract way, and show its connection to load balancing processes.

3.1 Formal Description

At the initialisation step, every node v picks a random number from 1 to n^3, which is used as the identification of node v. It is easy to show that, with high probability, all the nodes pick different numbers. We assume that this holds in the remaining part of the paper, and use $\mathsf{ID}(v)$ to represent the ID of node v. Our algorithm consists of three procedures:

The Seeding Procedure: Every node v repeats the following experiment for

$$\bar{s} \triangleq \frac{3}{\beta} \ln \frac{1}{\beta}$$

trials, where in each trial node v becomes *active* with probability $1/n$. For every node v that has been active at least once, node v sets its initial state as $\mathsf{State}_v(0) = \{(\mathsf{ID}(v), 0)\}$. Every non-active node v sets $\mathsf{State}_v(0) = \emptyset$. For simplicity, we call $\mathsf{ID}(v)$ and x the *prefix* and *suffix* of vector $(\mathsf{ID}(v), x)$.

The Averaging Procedure: The averaging procedure proceeds for T rounds, where in each round t each node v computes its state $\mathsf{State}_v(t)$ through the following operations: (1) nodes apply the distributed algorithm described in Section 2.2 to generate a matching; (2) each node v computes the vector $\mathsf{State}_v(t)$ in round t as follows: If node v is not involved in any matching, then node v sets its state in round t as $\mathsf{State}_v(t) = \mathsf{State}_v(t-1)$. Otherwise node v is matched to node u in round t, and their states are computed based on the following rule, where both $\mathsf{State}_u(t)$ and $\mathsf{State}_v(t)$ are set to be empty initially:

- If there is $(\mathsf{ID}(w), x) \in \mathsf{State}_u(t-1)$ and $(\mathsf{ID}(w), y) \in \mathsf{State}_v(t-1)$ with the same prefix, then both of u and v adds the vector $(\mathsf{ID}(w), (x+y)/2)$ to $\mathsf{State}_u(t)$ and $\mathsf{State}_v(t)$.
- For any vector $(\mathsf{ID}(w), x) \in \mathsf{State}_u(t-1)$ that does not share a common prefix with any vector in $\mathsf{State}_v(t-1)$, both of u and v adds the vector $(\mathsf{ID}(w), x/2)$ to $\mathsf{State}_u(t)$ and $\mathsf{State}_v(t)$ respectively.
- For any vector $(\mathsf{ID}(w), y) \in \mathsf{State}_v(t-1)$ that does not share a common prefix with any vector in $\mathsf{State}_u(t-1)$, both of u and v adds the vector $(\mathsf{ID}(w), y/2)$ to $\mathsf{State}_u(t)$ and $\mathsf{State}_v(t)$ respectively.

The Query Procedure: The query procedure assigns every node v to a label ℓ_v, and any two nodes u, v belong to the same cluster if and only if $\ell_u = \ell_v$. Formally, based on $\mathsf{State}_v(T)$ node v uses

$$\ell_v = \min\left\{ \mathsf{ID}(w) \mid (\mathsf{ID}(w), x) \in \mathsf{State}_v(T) \bigwedge x \geqslant \frac{1}{\sqrt{2\beta n}} \right\}$$

as the label of the cluster it belongs to, and ℓ_v is set to be an arbitrary ID if there is no vector $(\mathsf{ID}(w), x) \in \mathsf{State}_v(T)$ satisfying $x \geqslant 1/\left(\sqrt{2\beta n}\right)$.

3.2 Connection to Multi-Dimensional Load Balancing

From the formal description above, it is easy to see that the prefix of any vector is only used to identify from which node the corresponding unit load is generated, and loads from the vectors with different prefix will not be balanced during the execution of the algorithm. Therefore, we can view our algorithm as a multi-dimensional load balancing process, which is described as follows.

The seeding procedure consists of $\bar{s} \triangleq (3/\beta)\ln(1/\beta)$ trials, where in each trial every node becomes *active* with probability $1/n$. For simplicity, we use s to denote the number of active nodes at the end of these $\bar{s}$ trials, and use $v_1, \cdots, v_s$ to denote these active nodes. Moreover, we introduce s vectors $x^{(0,1)}, \ldots, x^{(0,s)} \in \mathbb{R}^n$, where $x^{(0,i)} = \chi_{v_i}$ for any $1 \leqslant i \leqslant s$.

After that, the averaging procedure proceeds for T rounds, where in each round t the nodes apply the distributed algorithm described in Section 2.2 to generate a matching $\mathbf{M}^{(t)}$, and update the vectors $x^{(t,i)}$ as follows: if nodes u and v are matched in round t, then they simply average their load evenly, i.e.,

$$x^{(t,i)}(u) = x^{(t,i)}(v) = \frac{x^{(t-1,i)}(u) + x^{(t-1,i)}(v)}{2}, \qquad i = 1, \ldots, s;$$

otherwise, for every unmatched node u, node u simply sets

$$x^{(t,i)}(u) = x^{(t-1,i)}(u), \qquad i = 1, \ldots, s.$$

Notice that the evolution of these s load vectors can be described by $x^{(t,i)} = \mathbf{M}^{(t)} x^{(t-1,i)}$ for any $i = 1, \ldots, s$.

Finally, at the query procedure every node v checks its coordinates $x^{(T,1)}(v), \ldots, x^{(T,s)}(v)$, and uses

$$\ell_v = \min\left\{ i \mid x^{(T,i)}(v) \geqslant \frac{1}{\sqrt{2\beta n}} \right\}$$

as the label of the cluster it belongs to. If no such index i exists, the algorithm assigns node v an arbitrary label $\ell_v \in \{1, \ldots, s\}$.

As a side remark, notice that, compared with the standard load balancing process in which the configuration for any round is expressed by an n-dimensional vector, in our algorithm there are s vectors of dimension n in each round. However, in each round the *same* matching matrix is applied to update these s vectors. Notice that, since $\mathbb{E}[\, s \,] = \bar{s}$, the expected communication cost is low, i.e., $O(T \cdot \bar{s} \cdot n)$. Secondly, as an interesting feature, our algorithm does not need to know the exact number of clusters k, and a lower bound of β suffices for our algorithm. Thirdly, the number of rounds T required by our algorithm relates to an upper bound of the *local mixing time* of a cluster, i.e., the time required for a random walk to become mixed inside a cluster. In particular, a value of $T = \Theta\left(\frac{\log n}{1-\lambda_{k+1}}\right)$ suffices and this value is typically a poly-logarithmic function of n for most graphs exhibiting a strong cluster-structure.

4 Analysis

In this section we analyse the algorithm, and prove Theorem 1.1. Remember that the configuration of our algorithm is expressed by s vectors $x^{(t,1)}, \ldots, x^{(t,s)}$, and these vectors are updated with respect to the *same* matching matrix in each round. To elaborate the intuitions behind our analysis, we first look at the standard load balancing process (the 1-dimensional case), and use the symbols $y^{(t)} \in \mathbb{R}^n$ to express the load distribution in round t for the 1-dimensional load balancing process, where $y^{(0)} \triangleq \chi_u$ for some node u, and the load distribution in round $t+1$ is defined by

$$y^{(t+1)} \triangleq \mathbf{M}^{(t)} y^{(t)}. \tag{3}$$

It is well-known that the sequence $\left\{y^{(t)}\right\}_{t=1}^{\infty}$ converges to the stationary distribution of a random walk in G, i.e., the first eigenvector f_1 of $\mathbf{P}$ [5, 28], and $y^{(t)}$ is close to f_1 when t is the mixing time of a random walk in G [27]. Studying the early behaviour of load balancing processes, however, is more complicated, and we will show that the early behaviour of this process depends on the cluster-structure of G.

Our starting point is to study the load distribution $y^{(T)}$. Informally, our choice of T corresponds to the time when a random walk gets well mixed and the resulting distribution becomes stable in S_i, as long as the random walk always stays in S_i. This happens if a random walk starts from a *good node* in S_i so that it won't leave S_i quickly. We will prove that there are enough good nodes so that, if the load balancing process above starts with χ_u for a good node u, then $y^{(T)}$ is close to a linear combination of $\chi_{S_1}, \cdots, \chi_{S_k}$. This implies that $y^{(T)}(u)$ and $y^{(T)}(v)$ are approximately the same if u and v belong to the same cluster.

Generalising this argument, we study the multi-dimensional load balancing process and prove the following fact: if the load balancing process starts with s vectors $x^{(0,1)}, \ldots, x^{(0,s)}$, then two nodes u, v belong to the same cluster if the values node u maintains, i.e., $\left(x^{(T,1)}(u), \ldots, x^{(T,s)}(u)\right)$, are similar with the values node v maintains.

4.1 Proof Sketch

We first focus on the load balancing process for the 1-dimensional case, and study the changes in vectors $\left\{y^{(t)}\right\}_{t=1}^{\infty}$. We will prove that $y^{(T)}$ is close to the projection of the initial vector $y^{(0)}$ on the subspace spanned by $f_1, \ldots, f_k$. Formally, we denote by $\mathbf{Q}$ the projection matrix onto the subspace spanned by $f_1, \ldots, f_k$ of $\mathbf{P}$, and show the following result:

Lemma 4.1. *It holds for any $t \geqslant T$ and any constant $c > 0$ that*

$$\mathbb{E}\left[\left\|\mathbf{Q}y^{(0)} - y^{(t)}\right\|\right] \leqslant 2\sqrt{t \cdot (1-\lambda_k)}\left\|\mathbf{Q}y^{(0)}\right\| + o\left(n^{-c}\right),$$

where the expectation is over all possible random matchings chosen during the first t rounds.

To explain the statement above, notice that every sampled random matching matrix $\mathbf{M}^{(j)}$ in any round j satisfies

$$\mathbb{E}\left[\mathbf{M}^{(j)}\right] = \left(1 - \frac{\bar{d}}{4}\right)\mathbf{I} + \frac{\bar{d}}{4} \cdot \mathbf{P}$$

by Lemma 2.1, i.e., the expected behaviour of a single round load balancing is the same as a 1-step lazy random walk. Therefore, we can imagine that $y^{(T)}$ will be close to $\mathbf{Q}y^{(0)}$ in T rounds, as there is a gap between λ_k and λ_{k+1}, and the contributions of $f_1, \ldots, f_k$ towards $y^{(T)}$ will become dominant. Each sampled matrix $\mathbf{M}^{(j)}$ in each round j, however, can differ from $\mathbb{E}\left[\mathbf{M}^{(j)}\right]$ significantly, affecting the distribution of the load vectors in all subsequent rounds. Lemma 4.1 states that, although the above event could occur, in expectation $\left\|\mathbf{Q}y^{(0)} - y^{(t)}\right\|$ is small.

Remark 1. *Notice that the bound in Lemma 4.1 is* increasing *in t. This is due to the fact that, although the distribution of a random walk becomes stable inside a cluster in T rounds, after $t \gg T$ steps the distribution of such random walk will converge to the uniform distribution of the whole graph, and the error term will increase with respect to t.*

Next, we will show that when the underlying graph G is well-clustered, there is an orthonormal set $\{\widehat{\chi}_i\}_{i=1}^{k}$, each $\widehat{\chi}_i$ being in the span of $\{\chi_{S_1}, \ldots, \chi_{S_k}\}$, such that $\widehat{\chi}_i$ is close to f_i. Combining this with Lemma 4.1, we will prove that $\mathbf{Q}y^{(0)}$ is almost constant on each cluster.

Lemma 4.2. *For any $1 \leqslant i \leqslant k$ there exists $\widehat{\chi}_i$ in the span of $\{\chi_{S_1}, \ldots, \chi_{S_k}\}$, such that*

$$\|\widehat{\chi}_i - f_i\| \leqslant \mathcal{E} \triangleq \Theta\left(k\sqrt{\frac{k}{\Upsilon}}\right).$$

Moreover, $\{\widehat{\chi}_i\}_{i=1}^{k}$ form an orthonormal set.

Lemma 4.2 bounds the ℓ_2-distance between $\widehat{\chi}_i$ and f_i for $i = 1, \ldots, k$. We will next show that there are enough "good" nodes that have "small" contribution to $\sum_{i=1}^{k}\|\widehat{\chi}_i - f_i\|^2$. If we start the load balancing process at one of these good nodes, then the load distribution $y^{(T)}$ will be close to a vector that is constant on the coordinates corresponding to nodes in some cluster S_j, and 0 otherwise. Formally, for every node v, let

$$\alpha_v \triangleq \sqrt{\sum_{i=1}^{k}\left(f_i(v) - \widehat{\chi}_i(v)\right)^2} \tag{4}$$

be the contribution of node v to the total error $\sum_{i=1}^{k}\|\widehat{\chi}_i - f_i\|^2$ from Lemma 4.2. We call a node v *good* if

$$\alpha_v \leqslant k\mathcal{E}\sqrt{\frac{C \log n \log(1/\beta)}{\beta n}}$$

for some constant C and call v a *bad* node otherwise. The following lemma shows that, when staring the 1-dimensional load balancing process from a good node v in a cluster S_j, the expected distance between $y^{(T)}$ and χ_{S_j} can be bounded.

Lemma 4.3. *Let S_j be any cluster, and $v \in S_j$ be a good node. Starting the load balancing process for T rounds with the initial load vector $y^{(0)} = \chi_v$, we have that*

$$\mathbb{E}\left[\left\|y^{(T)} - \chi_{S_j}\right\|\right] = O\left(k \cdot \mathcal{E} \cdot \sqrt{\frac{\log n \cdot \log(1/\beta)}{\beta \cdot n}}\right).$$

Based on these lemmas, we are ready to prove Theorem 1.1.

Proof of Theorem 1.1. The seeding procedure consists of $\bar{s}$ trials, where in each trial a node is active with probability $1/n$. Hence, the total number of active nodes s satisfies $\mathbb{E}[s] = \bar{s}$ and, by Markov inequality, $s = O(\bar{s})$ with probability at least $1 - c$ for an arbitrary small constant $c > 0$. We assume this holds in the remaining part of the proof.

For any fixed cluster S_j, the probability that no node in S_j is active in any one of the $\bar{s}$ trials is at most

$$\prod_{v \in S_j}\left(1 - \frac{1}{n}\right)^{\bar{s}} \leqslant \prod_{v \in S_j} \mathrm{e}^{-\bar{s}/n} = \mathrm{e}^{-\bar{s}\sum_{v \in S_j} 1/n}$$
$$\leqslant \mathrm{e}^{-\bar{s}\beta} \leqslant \mathrm{e}^{-3\ln\beta^{-1}} \leqslant \mathrm{e}^{-3}/k$$

where we use the fact $1 - x \leqslant \mathrm{e}^{-x}$ for $x \leqslant 1$ in the first inequality, and the assumption that $\left|S_j\right| \geqslant \beta n, \beta \leqslant 1/k$. Applying a union bound, with probability at least $1 - \mathrm{e}^{-3}$ there is at least one active node in each cluster.

Let $\mathcal{I} = \{v_1, \ldots, v_s\}$ be the set of active nodes, and denote by $\mathcal{S}(v)$ the cluster to which node v belongs to. By the definition of

α_v and the fact $\sum_v \alpha_v^2 = k\mathcal{E}^2$, the number of bad nodes is at most

$$k\mathcal{E}^2 \cdot \left(k\mathcal{E}\sqrt{\frac{C\log n \log(1/\beta)}{\beta n}}\right)^{-2} = \frac{\beta n}{C \cdot k \log n \log(1/\beta)}$$

by the averaging argument. Hence, the probability that in any given trial a bad node is active is at most

$$\frac{1}{n} \cdot \frac{\beta n}{C \cdot k \log n \log(1/\beta)} = \frac{\beta}{C \cdot k \log n \log(1/\beta)},$$

and with constant probability all the active nodes are good. From now on we assume that this event occurs.

Now we apply Lemma 4.3 on each coordinate of the multi-dimensional load vector, and obtain

$$\mathbb{E}\left[\left\|x^{(T,i)} - \chi_{\mathcal{S}(v_i)}\right\|\right] = O\left(k \cdot \mathcal{E} \cdot \sqrt{\frac{\log n \cdot \log(1/\beta)}{\beta \cdot n}}\right)$$

for $i = 1, 2, \ldots, s$. By Markov inequality and the union bound, with constant probability it holds for all $i = 1, \ldots, s$ that

$$\left\|x^{(T,i)} - \chi_{\mathcal{S}(v_i)}\right\|^2 = O\left(\bar{s} \cdot k \cdot \mathcal{E} \cdot \sqrt{\frac{\log n \cdot \log(1/\beta)}{\beta \cdot n}}\right)^2. \quad (5)$$

To analyse the performance of the query procedure, notice that node v can be misclassified only if there is $i \in \{1, \ldots, s\}$ such that

$$\left|x^{(T,i)}(v) - \chi_{\mathcal{S}(v_i)}(v)\right|^2 \geqslant \frac{1}{2\beta n^2}.$$

By a simple averaging argument and assuming (5) holds, the number of misclassified nodes is at most

$$\begin{aligned}
&\sum_{i=1}^{s}\sum_{v\in V} \mathbb{1}\left\{\left|x^{(T,i)} - \chi_{\mathcal{S}(v_i)}(v)\right|^2 \geqslant \frac{1}{2\beta n^2}\right\} \\
&\leqslant \sum_{i=1}^{s} O\left(\bar{s} \cdot k \cdot \mathcal{E} \cdot \sqrt{\frac{\log n \cdot \log(1/\beta)}{\beta \cdot n}}\right)^2 \cdot 2\beta n^2 \\
&= O\left(\bar{s}^3 \cdot k^2 \cdot \mathcal{E}^2 \log\frac{1}{\beta}\log n\right) n \\
&= O\left(k^2 \cdot \mathcal{E}^2 \cdot \frac{1}{\beta^3}\log^4\frac{1}{\beta} \cdot \log n\right) n.
\end{aligned}$$

Combining this with the definition of $\mathcal{E}$ gives us that

$$\begin{aligned}
&\sum_{i=1}^{s}\sum_{v\in V} \mathbb{1}\left\{\left|x^{(T,i)} - \chi_{\mathcal{S}(v_i)}(v)\right|^2 \geqslant \frac{1}{2\beta n^2}\right\} \\
&= O\left(k^2 \cdot \mathcal{E}^2 \cdot \frac{n}{\beta^3}\log^4\frac{1}{\beta} \cdot \log n\right) \\
&= O\left(\frac{k^5}{\Upsilon} \cdot \frac{n}{\beta^3}\log^4\frac{1}{\beta} \cdot \log n\right) \\
&= o(n),
\end{aligned}$$

where the last equality holds by the assumption on Υ.

The total information exchanged follows from the fact that the algorithm finishes in T rounds, and in each round only matched nodes exchange the information of $O(k \log k)$ words. ∎

4.2 Proof of Lemma 4.1

Proof of Lemma 4.1. Without loss of generality, we denote by $\mathbf{Q}^\perp \triangleq \mathbf{I} - \mathbf{Q}$ the projection on the subspace spanned by the eigenvectors $f_{k+1}, \ldots, f_n$. Since $\mathbf{Q}y^{(t)}$ and $\mathbf{Q}^\perp y^{(t)}$ are orthogonal to each other, it holds that

$$\begin{aligned}
&\mathbb{E}\left[\left\|\mathbf{Q}y^{(0)} - y^{(t)}\right\|^2\right] \\
&= \mathbb{E}\left[\left\|\mathbf{Q}y^{(0)} - \left(\mathbf{Q} + \mathbf{Q}^\perp\right)y^{(t)}\right\|^2\right] \\
&= \mathbb{E}\left[\left\|\mathbf{Q}y^{(0)} - \mathbf{Q}y^{(t)}\right\|^2\right] + \mathbb{E}\left[\left\|\mathbf{Q}^\perp y^{(t)}\right\|^2\right]. \quad (6)
\end{aligned}$$

Proving that the first term in (6) is small corresponds to show that after $t \approx T$ rounds the contribution of the top k eigenvectors $f_1, \ldots, f_k$ to $y^{(t)}$ is dominant, while proving that the second term is small means that the contribution of the bottom k eigenvectors $f_{k+1}, \ldots, f_n$ to $y^{(t)}$ becomes negligible. This is what we would expect if at each round we were able to apply directly the expected matrix $\mathbb{E}\left[\mathbf{M}^{(t)}\right]$. We prove that in expectation these facts hold, although different matching matrices $\mathbf{M}^{(t)}$ are applied in different rounds.

Formally, we analyse the first term in (6) and have that

$$\begin{aligned}
&\mathbb{E}\left[\left\|\mathbf{Q}\left(y^{(0)} - y^{(t)}\right)\right\|^2\right] \\
&= \mathbb{E}\left[\sum_{i=1}^{k}\left\langle y^{(0)} - y^{(t)}, f_i\right\rangle^2\right] \\
&= \sum_{i=1}^{k}\mathbb{E}\left[\left(\left\langle y^{(0)}, f_i\right\rangle - \left\langle y^{(t)}, f_i\right\rangle\right)^2\right] \\
&= \sum_{i=1}^{k}\left(\left\langle y^{(0)}, f_i\right\rangle^2 + \mathbb{E}\left[\left\langle y^{(t)}, f_i\right\rangle^2\right] - 2\left\langle y^{(0)}, f_i\right\rangle\mathbb{E}\left[\left\langle y^{(t)}, f_i\right\rangle\right]\right) \\
&\leqslant \sum_{i=1}^{k}\left(2\left\langle y^{(0)}, f_i\right\rangle^2 - 2\left\langle y^{(0)}, f_i\right\rangle\mathbb{E}\left[\left\langle y^{(t)}, f_i\right\rangle\right]\right), \quad (7)
\end{aligned}$$

where the last inequality uses the fact that, for every t, $\mathbf{M}^{(t)}$ is a projection matrix with norm at most one, and therefore

$$\mathbb{E}\left[\left\langle y^{(t)}, f_i\right\rangle^2\right] \leqslant \left\langle y^{(0)}, f_i\right\rangle^2.$$

Also, since at every round t the picked matrix $\mathbf{M}^{(t)}$ is independent from previous matchings, it holds that

$$\begin{aligned}
\mathbb{E}\left[\left\langle y^{(t)}, f_i\right\rangle\right] &= y^{(0)\intercal}\mathbb{E}\left[\mathbf{M}^{(t)} \cdots \mathbf{M}^{(1)}\right] f_i \\
&= y^{(0)\intercal}\mathbb{E}\left[\mathbf{M}^{(0)}\right]^t f_i \\
&= \left(1 - \frac{\bar{d} - \bar{d} \cdot \lambda_i}{4}\right)^t \left\langle y^{(0)}, f_i\right\rangle, \quad (8)
\end{aligned}$$

Therefore, it holds that

$$\begin{aligned}
&\mathbb{E}\left[\left\|\mathbf{Q}y^{(0)} - \mathbf{Q}y^{(t)}\right\|^2\right] \\
&\leqslant \sum_{i=1}^{k}\left(2\left\langle y^{(0)}, f_i\right\rangle^2 - 2\left(1 - \frac{\bar{d} - \bar{d} \cdot \lambda_i}{4}\right)^t \left\langle y^{(0)}, f_i\right\rangle^2\right) \\
&\leqslant 2t \cdot (1 - \lambda_k)\left\|\mathbf{Q}y^{(0)}\right\|^2. \quad (9)
\end{aligned}$$

To bound the second term in (6), we study the total expected norm of $y^{(t)}$, and prove that, for any $\ell, t \geqslant 1$, it holds that

$$\mathbb{E}\left[\mathbf{M}^{(t)}\mathbf{P}^{\ell}\mathbf{M}^{(t)}\right] \preceq \left(1-\frac{\bar{d}}{8}\right)\mathbf{P}^{\ell}+\frac{\bar{d}}{8}\mathbf{P}^{\ell+1}. \tag{10}$$

To see this, we fix two nodes u, v. Then, the value of ${\mathbf{M}^{(t)}\mathbf{P}^{\ell}\mathbf{M}^{(t)}}_{u,v}$ depends on how nodes u and v are matched in round t:

Case 1: If both of u and v are not involved in the matching in round t, then ${\mathbf{M}^{(t)}\mathbf{P}^{\ell}\mathbf{M}^{(t)}}_{u,v} = \mathbf{P}^{\ell}_{u,v}$.

Case 2: If u is not involved in the matching but v is matched to a node $\sigma(v) \neq v$, then ${\mathbf{M}^{(t)}\mathbf{P}^{\ell}\mathbf{M}^{(t)}}_{u,v} = (1/2)\cdot\mathbf{P}^{\ell}_{u,v}+(1/2)\cdot\mathbf{P}^{\ell}_{u,\sigma(v)}$.

Case 3: Similarly, if u is matched to $\sigma(u) \neq u$ but v is not involved in the matching in round t, then ${\mathbf{M}^{(t)}\mathbf{P}^{\ell}\mathbf{M}^{(t)}}_{u,v} = (1/2)\cdot\mathbf{P}^{\ell}_{u,v} + (1/2)\cdot\mathbf{P}^{\ell}_{\sigma(u),v}$.

Case 4: If u is matched to $\sigma(u) \neq u$ and v to $\sigma(v) \neq v$, then ${\mathbf{M}^{(t)}\mathbf{P}^{\ell}\mathbf{M}^{(t)}}_{u,v} = (1/4)\cdot\left(\mathbf{P}^{\ell}_{u,v}+\mathbf{P}^{\ell}_{\sigma(u),v}+\mathbf{P}^{\ell}_{u,\sigma(v)}+\mathbf{P}^{\ell}_{\sigma(u),\sigma(v)}\right)$.

Notice that the exact value of $\mathbb{E}\left[{\mathbf{M}^{(t)}\mathbf{P}^{\ell}\mathbf{M}^{(t)}}_{u,v}\right]$ depends on how node v can be reached from node u from one or two matching edges in round t, as well as a walk of length ℓ. Hence, we can write

$$\mathbb{E}\left[{\mathbf{M}^{(t)}\mathbf{P}^{\ell}\mathbf{M}^{(t)}}_{u,v}\right] = \alpha_1\mathbf{P}^{\ell}_{u,v}+\alpha_2{\mathbf{P}^{\ell+1}}_{u,v}+\alpha_3{\mathbf{P}^{\ell+2}}_{u,v},$$

where $\alpha_1+\alpha_2+\alpha_3 = 1$. In particular, since the first case occurs with probability at most $(1-\bar{d}/4)$, it holds that $\alpha_1 \leqslant 1-\bar{d}/8$. Then, (10) follows from the fact that $\mathbf{P}^{\ell+2} \preceq \mathbf{P}^{\ell+1} \preceq \mathbf{P}^{\ell}$, and we have that

$$\begin{aligned}
&\mathbb{E}\left[\left\|y^{(t)}\right\|^2\right]\\
&= {y^{(0)}}^{\intercal}\,\mathbb{E}\left[\mathbf{M}^{(t)}\mathbf{M}^{(t-1)}\cdots\mathbf{M}^{(1)}\mathbf{M}^{(1)}\cdots\mathbf{M}^{(t-1)}\mathbf{M}^{(t)}\right]y^{(0)}\\
&\leqslant {y^{(0)}}^{\intercal}\left(\left(1-\frac{\bar{d}}{8}\right)\mathbf{I}+\frac{\bar{d}}{8}\cdot\mathbf{P}\right)^{t}y^{(0)}.
\end{aligned}\tag{11}$$

To bound $\mathbb{E}\left[\left\|\mathbf{Q}y^{(t)}\right\|^2\right]$, we use (8) and obtain that

$$\begin{aligned}
\mathbb{E}\left[\left\|\mathbf{Q}y^{(t)}\right\|^2\right] &= \sum_{i=1}^{k}\mathbb{E}\left[\left\langle y^{(t)}, f_i\right\rangle^2\right]\\
&\geqslant (1-2t(1-\lambda_k))\sum_{i=1}^{k}\left\langle y^{(0)}, f_i\right\rangle^2\\
&= (1-2t(1-\lambda_k))\left\|\mathbf{Q}y^{(0)}\right\|^2,
\end{aligned}\tag{12}$$

where the first inequality follows from the Jensen's inequality. Combining (11), (12) and the fact that $\left\langle \mathbf{Q}y^{(t)}, \mathbf{Q}^{\perp}y^{(t)}\right\rangle = 0$, we obtain that

$$\begin{aligned}
&\mathbb{E}\left[\left\|\mathbf{Q}^{\perp}y^{(t)}\right\|^2\right]\\
&= \mathbb{E}\left[\left\|y^{(t)}\right\|^2\right]-\mathbb{E}\left[\left\|\mathbf{Q}y^{(t)}\right\|^2\right]\\
&\leqslant {y^{(0)}}^{\intercal}\left(\left(1-\frac{\bar{d}}{8}\right)\mathbf{I}+\frac{\bar{d}}{8}\cdot\mathbf{P}\right)^{t}y^{(0)}-(1-2t(1-\lambda_k))\left\|\mathbf{Q}y^{(0)}\right\|^2\\
&\leqslant 2t(1-\lambda_k)\left\|\mathbf{Q}y^{(0)}\right\|^2+\left(1-\frac{\bar{d}}{8}+\frac{\bar{d}}{8}\lambda_{k+1}\right)^{t}\\
&\leqslant 2t(1-\lambda_k)\left\|\mathbf{Q}y^{(0)}\right\|^2+o(n^{-c}),
\end{aligned}\tag{13}$$

where (13) holds for a large constant $c > 0$ due to our choice of $t \geqslant T$.

Finally, combining (9) with (13) gives us that

$$\mathbb{E}\left[\left\|\mathbf{Q}y^{(0)}-y^{(t)}\right\|^2\right] \leqslant t\cdot(1-\lambda_k)\left\|\mathbf{Q}y^{(0)}\right\|^2+o\left(n^{-c}\right),$$

and Lemma 4.1 holds by applying the Jensen's inequality. ■

4.3 Proof of Lemma 4.2

To prove Lemma 4.2, we need the following lemma:

Lemma 4.4 ([25]). *Let $\{S_i\}_{i=1}^{k}$ be a k-way partition of G achieving $\rho(k)$, and let $\Upsilon = \Omega\left(k^2\right)$. Assume that $\widetilde{\chi}_i$ is the projection of f_i in the span of $\{\chi_{S_1},\ldots,\chi_{S_k}\}$. Then, it holds for any $1 \leqslant i \leqslant k$ that*

$$\|\widetilde{\chi}_i - f_i\| = O\left(\sqrt{\frac{k}{\Upsilon}}\right).$$

Proof of Lemma 4.2. Since $\{f_i\}_{i=1}^{k}$ is an orthonormal set, it holds by Lemma 4.4 that $\{\widetilde{\chi}_i\}_{i=1}^{k}$ are *almost* orthonormal. Hence, our task is to construct an orthonormal set $\{\widehat{\chi}_i\}_{i=1}^{k}$ based on $\{\widetilde{\chi}_i\}_{i=1}^{k}$, which can be achieved by applying the Gram-Schmidt orthonormalisation procedure. The error bound follows from the fact that

$$\left\langle\widetilde{\chi}_i,\widetilde{\chi}_j\right\rangle = O\left(\sqrt{\frac{k}{\Upsilon}}\right)$$

holds for $i \neq j$. ■

4.4 Proof of Lemma 4.3

Proof. We first show that χ_{S_i} is the projection of the initial load vector $y^{(0)} = \chi_v$ in the span of $\{\chi_{S_1},\ldots,\chi_{S_k}\}$. Since every $\widehat{\chi}_i$ $(1 \leqslant i \leqslant k)$ is a linear combination of vectors in $\{\chi_{S_i}\}_{i=1}^{k}$, and $\widehat{\chi}_1,\ldots,\widehat{\chi}_k$ are orthonormal by Lemma 4.2, we have that $\mathrm{span}\{\widehat{\chi}_1,\ldots,\widehat{\chi}_k\} = \mathrm{span}\{\chi_{S_1},\ldots,\chi_{S_k}\}$. Hence,

$$\begin{aligned}
\sum_{i=1}^{k}\left\langle\chi_v,\widehat{\chi}_i\right\rangle\widehat{\chi}_i &= \sum_{i=1}^{k}\left\langle\chi_v,\frac{\chi_{S_i}}{\|\chi_{S_i}\|}\right\rangle\frac{\chi_{S_i}}{\|\chi_{S_i}\|}\\
&= \left\langle\chi_v,\chi_{S_j}\right\rangle\frac{\chi_{S_j}}{\|\chi_{S_j}\|^2} = \chi_{S_j},
\end{aligned}\tag{14}$$

where the first equality holds by the fact that $\mathrm{span}\{\widehat{\chi}_1,\ldots,\widehat{\chi}_k\} = \mathrm{span}\{\chi_{S_1},\ldots,\chi_{S_k}\}$, the second equality holds since χ_v is orthogonal to every χ_{S_ℓ} with $\ell \neq j$, and the third equality holds by the fact that $\langle\chi_v,\chi_{S_j}\rangle = 1/|S_j| = \|\chi_{S_j}\|^2$.

Based on this, we bound the expected distance between $y^{(T)}$ and χ_{S_j}. By the triangle inequality, it holds that

$$\mathbb{E}\left[\left\|y^{(T)}-\chi_{S_j}\right\|\right] \leqslant \mathbb{E}\left[\left\|\mathbf{Q}\chi_v-y^{(T)}\right\|\right]+\left\|\mathbf{Q}\chi_v-\chi_{S_j}\right\|, \tag{15}$$

where the expectation is over all possible random matchings generated within the first T rounds. By Lemma 4.1, we have that

$$\mathbb{E}\left[\left\|\mathbf{Q}\chi_v-y^{(T)}\right\|\right] \leqslant 2\sqrt{T\cdot(1-\lambda_k)}\,\|\mathbf{Q}\chi_v\|+o\left(n^{-c}\right). \tag{16}$$

For the second term in the right hand side of (15), by the triangle inequality we have that

$$\begin{aligned}
&\left\|\mathbf{Q}\chi_v - \chi_{S_j}\right\| \\
&= \left\|\sum_{i=1}^{k}\langle\chi_v, f_i\rangle f_i - \sum_{i=1}^{k}\langle\chi_v, f_i\rangle\widehat{\chi}_i + \sum_{i=1}^{k}\langle\chi_v, f_i\rangle\widehat{\chi}_i - \sum_{i=1}^{k}\langle\chi_v, \widehat{\chi}_i\rangle\widehat{\chi}_i\right\| \\
&\leqslant \left\|\sum_{i=1}^{k}\langle\chi_v, f_i\rangle f_i - \sum_{i=1}^{k}\langle\chi_v, f_i\rangle\widehat{\chi}_i\right\| \\
&\qquad + \left\|\sum_{i=1}^{k}\langle\chi_v, f_i\rangle\widehat{\chi}_i - \sum_{i=1}^{k}\langle\chi_v, \widehat{\chi}_i\rangle\widehat{\chi}_i\right\| \qquad (17)
\end{aligned}$$

To bound the first term in (17), we have that

$$\begin{aligned}
\left\|\sum_{i=1}^{k}\langle\chi_v, f_i\rangle f_i - \sum_{i=1}^{k}\langle\chi_v, f_i\rangle\widehat{\chi}_i\right\| &\leqslant \sum_{i=1}^{k}|\langle\chi_v, f_i\rangle|\,\|f_i - \widehat{\chi}_i\| \\
&\leqslant \mathcal{E}\sum_{i=1}^{k}|\langle\chi_v, f_i\rangle| \leqslant k\mathcal{E}\,\|\mathbf{Q}\chi_v\| \qquad (18)
\end{aligned}$$

where the first line follows from the triangle inequality, the second follows by Lemma 4.2, and the last follows by the Cauchy-Schwarz inequality. To bound the second term in (17), we have that

$$\begin{aligned}
\left\|\sum_{i=1}^{k}\left(\langle\chi_v, f_i\rangle - \langle\chi_v, \widehat{\chi}_i\rangle\right)\widehat{\chi}_i\right\| &= \left\|\sum_{i=1}^{k}\left(f_i(v) - \widehat{\chi}_i(v)\right)\widehat{\chi}_i\right\| \\
&= \sqrt{\sum_{i=1}^{k}\left(f_i(v) - \widehat{\chi}_i(v)\right)^2\|\widehat{\chi}_i\|^2} \\
&= \alpha_v
\end{aligned}$$

where the second inequality follows from the orthonormality of $\{\widehat{\chi}_i\}_i$, and the third equality from the definition of α_v and, again, the orthonormality of $\{\widehat{\chi}_i\}_i$. Thus, we rewrite (17) as

$$\left\|\mathbf{Q}\chi_v - \chi_{S_j}\right\| \leqslant k\cdot\mathcal{E}\cdot\|\mathbf{Q}\chi_v\| + \alpha_v. \qquad (19)$$

Combining (15), (16) with (19), we have that

$$\begin{aligned}
\mathbb{E}\left[\left\|y^{(T)} - \chi_{S_j}\right\|\right] &\leqslant \left(\sqrt{T\cdot(1-\lambda_k)} + k\cdot\mathcal{E}\right)\|\mathbf{Q}\chi_v\| + \alpha_v \\
&= O\left(k\cdot\mathcal{E}\sqrt{\log n}\right)\|\mathbf{Q}\chi_v\| + \alpha_v, \qquad (20)
\end{aligned}$$

where the last equality follows by (1) and the fact that

$$\sqrt{T\cdot(1-\lambda_k)} = O\left(\sqrt{\frac{(1-\lambda_k)\log n}{1-\lambda_{k+1}}}\right) = O\left(k\cdot\mathcal{E}\sqrt{\log n}\right).$$

Hence, it suffices to bound $\|\mathbf{Q}\chi_v\|^2$. Direct calculation shows that

$$\begin{aligned}
\|\mathbf{Q}\chi_v\|^2 &= \sum_{i=1}^{k}\langle\chi_v, f_i\rangle^2 \\
&= \sum_{i=1}^{k}\langle\chi_v, \widetilde{\chi}_i - (\widetilde{\chi}_i - f_i)\rangle^2 \\
&= \sum_{i=1}^{k}\left(\langle\chi_v, \widetilde{\chi}_i\rangle - \langle\chi_v, \widetilde{\chi}_i - f_i\rangle\right)^2 \\
&\leqslant \sum_{i=1}^{k}2\left(\langle\chi_v, \widetilde{\chi}_i\rangle^2 + \langle\chi_v, \widetilde{\chi}_i - f_i\rangle^2\right) \qquad (21) \\
&= 2\left\|\chi_{S_j}\right\|^2 + 2\langle\chi_v, \widetilde{\chi}_i - f_i\rangle^2 \qquad (22) \\
&\leqslant 2\left\|\chi_{S_j}\right\|^2 + 2\alpha_v^2 \qquad (23)
\end{aligned}$$

where (21) follows from the inequality

$$(a-b)^2 \leqslant 2(a^2+b^2),$$

(22) follows from (14), and (23) follows from the definition of α_v. Hence, it holds that $\|\mathbf{Q}\chi_v\| = O\left(\left\|\chi_{S_j}\right\| + \alpha_v\right)$, and we can rewrite (20) as

$$\begin{aligned}
\mathbb{E}\left[\left\|y^{(T)} - \chi_{S_j}\right\|\right] &= O\left(k\cdot\mathcal{E}\sqrt{\log n}\cdot\left(\left\|\chi_{S_j}\right\| + \alpha_v\right)\right) + \alpha_v \\
&= O\left(k\cdot\mathcal{E}\sqrt{\log n}\cdot\left\|\chi_{S_j}\right\| + \alpha_v\right),
\end{aligned}$$

where the last equality follows from the assumption on Υ. Then the lemma follows from by the definition of α_v and the fact that

$$\left\|\chi_{S_j}\right\| = \frac{1}{\sqrt{|S_j|}} \leqslant \frac{1}{\sqrt{\beta n}}.$$

■

4.5 Analysis for Almost-Regular Graphs

Finally, we show that our algorithm and analysis can be easily modified to work for *almost-regular graphs*, i.e., the graphs for which the ratio between maximum degree $\Delta = \max_{v\in V}\{d_v\}$ and the minimum degree $\delta = \min_{v\in V}\{d_v\}$ is upper bounded by some constant. We also assume each node knows an upper bound $D \geqslant \Delta$ of the maximum degree such that $D/\delta = \Theta(\Delta/\delta)$. With these assumptions, we only need to slightly modify the seeding procedure, in which every node v sets to be active with probability $\frac{1}{2} + \frac{D-d_v}{2D}$, instead of $1/2$ for the case of regular graphs. The Averaging and Query procedures remain the same.

To show our algorithm and analysis holds for almost-regular graphs, we view the underlying almost-regular graph G as a D-regular graph $G^\star$, which is obtained from G by adding $D - d_v$ self-loops to each node v. Then, the conductance of any set S is almost the same in G and $G^\star$, since

$$\phi_{G^\star}(S) = \frac{|E_G(S, V\setminus S)|}{D\cdot|S|} = \Theta\left(\frac{|E_G(S, V\setminus S)|}{\mathrm{vol}(S)}\right) = \Theta\left(\phi_G(S)\right).$$

It is also easy to see that the $(k+1)$th eigenvalues of the random walk matrix of G and $G^\star$ differ by at most a constant factor, and therefore $G^\star$ is well-clustered. Hence, Theorem 1.1 holds for almost-regular graphs as well.

REFERENCES

[1] Zeyuan Allen-Zhu, Silvio Lattanzi, and Vahab S. Mirrokni. 2013. A local algorithm for finding well-connected clusters. In *30th International Conference on Machine Learning (ICML'13)*. 396–404.

[2] Noga Alon, Chen Avin, Michal Koucký, Gady Kozma, Zvi Lotker, and Mark R. Tuttle. 2008. Many random walks are faster than one. In *20th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA'08)*. 119–128.

[3] Luca Becchetti, Andrea Clementi, Emanuele Natale, Francesco Pasquale, and Luca Trevisan. 2017. Find your place: Simple distributed algorithms for community detection. In *28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'17)*. 940–959.

[4] Petra Berenbrink, Colin Cooper, Tom Friedetzky, Tobias Friedrich, and Thomas Sauerwald. 2015. Randomized diffusion for indivisible loads. *J. Comput. Syst. Sci.* 81, 1 (2015), 159–185.

[5] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. 2006. Randomized gossip algorithms. *IEEE Transactions on Information Theory* 52, 6 (2006).

[6] Jiecao Chen, He Sun, David P. Woodruff, and Qin Zhang. 2016. Communication-optimal distributed clustering. In *29th Advances in Neural Information Processing Systems (NIPS'16)*. 3720–3728.

[7] Flavio Chierichetti, Silvio Lattanzi, and Alessandro Panconesi. 2010. Almost tight bounds for rumour spreading with conductance. In *42nd Annual ACM Symposium on Theory of Computing (STOC'10)*. 399–408.

[8] Flavio Chierichetti, Silvio Lattanzi, and Alessandro Panconesi. 2010. Rumour spreading and graph conductance. In *21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'10)*. 1657–1663.

[9] Colin Cooper, Alan M. Frieze, and Tomasz Radzik. 2009. Multiple random walks in random regular graphs. *SIAM Journal on Discrete Mathematics* 23, 4 (2009), 1738–1761.

[10] George Cybenko. 1989. Dynamic load balancing for distributed memory multiprocessors. *J. Parallel and Distrib. Comput.* 7 (1989), 279–301.

[11] Robert Elsasser and Thomas Sauerwald. 2010. Discrete load balancing is (almost) as easy as continuous load balancing. In *29th Annual ACM-SIGOPT Principles of Distributed Computing (PODS'10)*. 346–354.

[12] Robert Elsasser and Thomas Sauerwald. 2011. Tight bounds for the cover time of multiple random walks. *Theoretical Computer Science* 412, 24 (2011), 2623–2641.

[13] Santo Fortunato. 2010. Community detection in graphs. *Physics Reports* 486, 3 (2010), 75–174.

[14] Tobias Friedrich, Martin Gairing, and Thomas Sauerwald. 2012. Quasirandom load balancing. *SIAM J. Comput.* 41, 4 (2012), 747–771.

[15] Tobias Friedrich and Thomas Sauerwald. 2009. Near-perfect load balancing by randomized rounding. In *41st Annual ACM Symposium on Theory of Computing (STOC'09)*. 121–130.

[16] Shayan Oveis Gharan and Luca Trevisan. 2012. Approximating the expansion profile and almost optimal local graph clustering. In *53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS'12)*. 187–196.

[17] Bhaskar Ghosh, S. Muthukrishnan, and Martin H. Schultz. 1996. First and second order diffusive methods for rapid, coarse, distributed load balancing. In *8th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA'96)*. 72–81.

[18] George Giakkoupis. 2011. Tight bounds for rumor spreading in graphs of a given conductance. In *28th International Symposium on Theoretical Aspects of Computer Science (STACS'11)*. 57–68.

[19] George Giakkoupis and Thomas Sauerwald. 2012. Rumor spreading and vertex expansion. In *23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'12)*. 1623–1641.

[20] Pan Hui, Eiko Yoneki, Shu Yan Chan, and Jon Crowcroft. 2007. Distributed community detection in delay tolerant networks. In *Proceedings of 2nd ACM/IEEE International Workshop on Mobility in the Evolving Internet Architecture*.

[21] David Kempe and Frank McSherry. 2004. A decentralized algorithm for spectral analysis. In *36th Annual ACM Symposium on Theory of Computing (STOC'04)*. 561–568.

[22] James R. Lee, Shayan Oveis Gharan, and Luca Trevisan. 2014. Multiway spectral partitioning and higher-order Cheeger inequalities. *Journal of the ACM* 61, 6 (2014), 37:1–37:30.

[23] Gurmeet Singh Manku. 2004. Balanced binary trees for ID management and load balance in distributed hash tables. In *23rd Annual ACM-SIGOPT Principles of Distributed Computing (PODS'04)*. 197–205.

[24] Shayan Oveis Gharan and Luca Trevisan. 2014. Partitioning into expanders. In *25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'14)*. 1256–1266.

[25] Richard Peng, He Sun, and Luca Zanetti. 2017. Partitioning well-clustered graphs: spectral clustering works! *SIAM J. Comput.* 46, 2 (2017), 710–743.

[26] Yuval Rabani, Alistair Sinclair, and Rolf Wanka. 1998. Local divergence of markov chains and the analysis of iterative load balancing schemes. In *39th Annual IEEE Symposium on Foundations of Computer Science (FOCS'98)*. 694–705.

[27] Thomas Sauerwald and He Sun. 2012. Tight bounds for randomized load balancing on arbitrary network topologies. In *53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS'12)*. 341–350.

[28] Devavrat Shah. 2009. Gossip Algorithms. *Foundations and Trends in Networking* 3, 1 (2009), 1–125.

[29] Daniel A. Spielman and Shang-Hua Teng. 2013. A local clustering algorithm for massive graphs and its application to nearly linear time graph partitioning. *SIAM J. Comput.* 42, 1 (2013), 1–26.

[30] Sonesh Surana, Brighten Godfrey, Karthik Lakshminarayanan, Richard M. Karp, and Ion Stoica. 2006. Load balancing in dynamic structured peer-to-peer systems. *Performance Evaluation* 63, 3 (2006), 217–240.

[31] Wenzhuo Yang and Huan Xu. 2015. A divide and conquer framework for distributed graph clustering. In *32nd International Conference on Machine Learning (ICML'15)*. 504–513.

[32] Dongliang Zhang, Changjun Jiang, and Shu Li. 2009. A fast adaptive load balancing method for parallel particle-based simulations. *Simulation Modelling Practice and Theory* 17, 6 (2009).