# Average whenever you meet: Opportunistic protocols for community detection

**Luca Becchetti**
Sapienza Università di Roma, Italy
becchetti@dis.uniroma1.it

**Andrea Clementi**
Università di Roma "Tor Vergata", Italy
clementi@mat.uniroma2.it

**Pasin Manurangsi**
U.C. Berkeley, California, USA
pasin@berkeley.edu

**Emanuele Natale**
Simons Institute and MPII, Germany
enatale@mpi-inf.mpg.de

**Francesco Pasquale**
Università di Roma "Tor Vergata", Italy
pasquale@mat.uniroma2.it

**Prasad Raghavendra**
U.C. Berkeley, California, USA
raghavendra@berkeley.edu

**Luca Trevisan**
Simons Institute and U.C. Berkeley, California, USA
luca@berkeley.edu

## Abstract

Consider the following asynchronous, opportunistic communication model over a graph $G$: in each round, one edge is activated uniformly and independently at random and (only) its two endpoints can exchange messages and perform local computations. Under this model, we study the following random process: *The first time a vertex is an endpoint of an active edge, it chooses a random number, say $\pm 1$ with probability $1/2$; then, in each round, the two endpoints of the currently active edge update their values to their average.*

We provide a rigorous analysis of the above process showing that, if $G$ exhibits a two-community structure (for example, two expanders connected by a sparse cut), the values held by the nodes will collectively reflect the underlying community structure over a suitable phase of the above process. Our analysis requires new concentration bounds on the product of certain random matrices that are technically challenging and possibly of independent interest.

We then exploit our analysis to design the first opportunistic protocols that approximately recover community structure using only logarithmic (or polylogarithmic, depending on the sparsity of the cut) work per node.

**2012 ACM Subject Classification** Theory of computation → Distributed algorithms

**Keywords and phrases** Community Detection, Random Processes, Spectral Analysis.

**Digital Object Identifier** 10.4230/LIPIcs.ESA.2018.7

**Funding** This material is based upon work supported by the National Science Foundation under Grants No. 1540685 and No. 1655215 and by the University of "Tor Vergata" under research programme "Mission: Sustainability" project ISIDE (grant no. E81I18000110005).

## 1 Introduction

**The Averaging Protocol.** Consider the following, elementary distributed process on an undirected graph $G = (V, E)$ with $|V| = n$ nodes and $|E| = m$ edges. Each node $v$ holds a real number $x_v$ (which we call the *state* of node $v$); at each time step, one random edge $\{u, v\}$ becomes active and its endpoints $u$ and $v$ update their states to their average.

Viewed as a protocol, the above process is consistent with asynchronous, opportunistic communication models, such as those considered in [1] for *population protocols*; here, in every round, one edge is activated uniformly and independently at random and (only) its two endpoints can exchange messages and perform local computations in that round. We further assume no global clock is available (nodes can at most count the number of local activations) and that the network is *anonymous*, i.e., nodes are not aware of theirs or their neighbors' identities and all nodes run the same process at all times.

The long-term behavior of the process outlined above is well-understood: assuming $G$ to be connected, for each initial global state $\mathbf{x} \in \mathbb{R}^V$ the system converges to a global state in which all nodes share a common value, namely, the average of their initial states. A variant of an argument of Boyd et al. [4] shows that convergence time is equivalent to the the mixing time of a lazy random walk on the graph, namely $\mathcal{O}\left(\frac{1}{\lambda_2} n \log n\right)$, where $\lambda_2$ is the second smallest eigenvalue of the normalized Laplacian of $G$.

**Distributed Community Detection.** Suppose now that $G$ is *well-clustered*, i.e. it exhibits a *community structure* which in the simplest case consists of two equal-sized expanders, connected by a sparse cut: this structure arises, for instance, when the graph is sampled from the popular *stochastic block model* $\mathcal{G}_{n,p,q}$ for $p \gg q$ and $p \geqslant \log n/n$ [6, 7, 10]. If we let the averaging process unfold on such a graph, for example starting from an initial $\pm 1$ random global state, one might reasonably expect a faster, transient convergence toward some local average within each community, accompanied by a slower, global convergence toward the average taken over the entire graph. If, as is likely the case, a gap exists between the local averages of the two communities, the global state during the transient phase would reflect the graph's underlying community structure. This intuition suggests the main questions we address in this paper: Is there *a phase in which the global state carries information about community structure*? If so, *how strong is the corresponding "signal"*? Finally, *can nodes leverage local history to recover this information*?

**Our Results: Highlights.** We show that, if $G$ exhibits a two-community structure (for example, two expanders connected by a sparse cut), the values held by the nodes will collectively reflect the underlying community structure over a suitable phase of the above process, allowing efficient and effective recovery in important cases.

In more detail, we first provide a first moment analysis showing that, for a large class of almost-regular clustered graphs that includes the *stochastic block model*, the expected values held by all but a negligible fraction of the nodes eventually reflect the underlying cut signal. We prove this property emerges after a "mixing" period of length $\mathcal{O}(n \log n)$.

We further provide a second moment analysis for a more restricted class of regular clustered graphs that includes the *regular stochastic block model* [3, 5, 11]. Since nodes do not share a common clock, it is not immediate to translate the above results into distributed clustering protocols. To this purpose, we show that concentration holds over a long time window and most nodes are able to select a local time within this window. So, most nodes can efficiently and locally identify their community of reference over a suitable time window. Even for the above class of regular graphs, our second moment analysis requires new concentration bounds on the product of certain random matrices that are technically challenging and possibly of independent interest.

This results in the first opportunistic protocols that approximately recover community structure. For clustered graphs with sparse (i.e. size $o(m)$) cut, we devise a first protocol, using the sign of the nodes' state as local clustering criterion (see Algorithm 2), that converges in $\mathcal{O}(n \log n)$ time and has only polylogarithmic work per node (see Theorem 9 for a formal statement). So, the protocol can be much faster than the global mixing time of the corresponding process and, moreover, the work per node does not depend on the node degree, thus resulting very efficient in the case of dense graphs. For clustered graphs with dense cut (i.e. size $\Theta(m)$), the cut "signal" is much harder to recover: we derive a more complex second moment analysis leading us to a weighted version of the averaging process, equipped with a clustering criterion based on the fluctuations of the nodes' state. This second protocol (see Algorithm 3) converges within $\mathcal{O}(n \log n + n/\lambda_2)$ rounds and has work per node $\mathcal{O}(\text{polylog}\,(n) + 1/\lambda_2)$ (see Theorem 10, Corollaries 5.2 and 5.3 for formal statements).

**Comparison to Previous Work.** We here discuss only strongly-related work (see the full-version [2] for a more detailed description of previous results. The idea of using averaging local rules to perform distributed community detection is not new: In [3], Becchetti et al. consider a deterministic dynamics in which, at every round, each node updates its local state to the average of its neighbors. The authors show that this results in a fast clustering algorithm with provable accuracy on a wide class of almost-regular graphs that includes the stochastic block model. We remark that the algorithm in [3] (only) works in a *synchronous, parallel communication model* where every node exchanges data with all its neighbors in each round. This implies considerable work and communication costs, especially when the graph is dense. It turns out that, in $d$-regular, well-clustered graphs, the algorithm in [3] requires overall communication cost $\Theta(nd\,\text{polylog}\,(n))$ and work per-node $\Theta(d\,\text{polylog}\,(n))$. On the other hand, each step of the process in [3] is described by the same matrix and its deterministic evolution unfolds according to the power of this matrix applied to the initial state. In contrast, the averaging process we consider in this paper is considerably harder to analyze than the one in [3], since each step is described by a random, possibly different averaging matrix. Differently from [3], our goal here is the design of simple, lightweight protocols for fully-decentralized community detection which fit the asynchronous, opportunistic communication model, in which a (random) link activation represents an opportunistic meeting that the endpoints can exploit to exchange one-to-one messages. More specifically, by "lightweight" we mean protocols that require minimalistic assumptions as to network capabilities, while performing their task with minimal work, storage and communication per node (at most logarithmic or polylogarithmic in our case). In this respect, any clustering strategies (like the one in [12]) which construct (and then work over) some static, sparse subgraph of the underlying graph are unfeasible in the opportunistic model we consider here. This restrictive setting is motivated by network scenarios in which individual agents need to autonomously and locally uncover underlying, implicit communities of which they are members. This has widespread applicability, for example in communication systems where lightweight data can be locally shared via wireless opportunistic meetings when agents come within close range [13].

**Roadmap of the paper.** After presenting some preliminaries in Section 2, the first moment analysis for almost-regular graphs is given in Section 3. The second moment analysis for regular graphs is given in Section 4 while, in Section 5, we describe our protocols for community detection and give the main bounds on their performances. Due to space constraints, most technical results are given in the full-version of the paper [2].

## 2 Preliminaries

**Averaging process.** In general, we consider the weighted version of the averaging process described in the introduction: In each round, one edge of the graph is sampled uniformly at random and the two endpoints of the sampled edge execute Algorithm 1.

```
AVERAGING($\delta$)   (for a node $u$ that is one of the two endpoints of an active edge)
Initialization: If it is the first time $u$ is active, then pick $\mathbf{x}_u \in \{-1,+1\}$ u.a.r.
Update: Send $\mathbf{x}_u$ to the other endpoint and set $\mathbf{x}_u := (1-\delta)\mathbf{x}_u + \delta r$, where $r$ is the
  value received from the other endpoint.
```

**Algorithm 1:** Updating rule for a node $u$ of an active edge, where $\delta \in (0,1)$ is the parameter measuring the weight given to the neighbor's value

**Graphs and their spectra.** For a connected graph $G = (V,E)$ with $n$ nodes, $m$ edges and adjacency matrix $A$, let $0 = \lambda_1 \leqslant \cdots \leqslant \lambda_n$ be the eigenvalues of the normalized Laplacian $\mathcal{L} = I - D^{-1/2}AD^{-1/2}$, where $D$ is the diagonal matrix with the degrees of the nodes. We consider the following graph classes.

▶ **Definition 1.** An $(n,d,\beta)$*-almost-regular graph* is a connected, non-bipartite graph $G = (V,E)$ with $n$ nodes such that every node has degree $d \pm \beta d$. An $(n,d,b)$*-clustered regular graph*, where $n$ is even and $d$ and $b$ are two positive integers with $2b < d < n$, is a graph $G = ((V_1,V_2),E)$ over node set $V = V_1 \cup V_2$, with $|V_1| = |V_2| = n/2$ and such that: (i) every node has degree $d$ and (ii) every node in $V_1$ has $b$ neighbors in $V_2$ and every node in $V_2$ has $b$ neighbors in $V_1$.

It is easy to see that the indicator vector $\boldsymbol{\chi} \in \{-1,+1\}$ of the cut $(V_1,V_2)$ is an eigenvector of $\mathcal{L}$ with eigenvalue $\frac{2b}{d}$, whenever the graph is clustered regular. If we further assume that $\lambda_3 > \frac{2b}{d}$, then $\boldsymbol{\chi}$ is an eigenvector of $\lambda_2$.

**Block reconstruction.** We next discuss what it means to recover the "underlying community structure" in a distributed setting, a notion that can come in stronger or weaker flavors [6, 11, 8, 9]. Ideally, we would like the protocol to reach a state in which, at least with high probability, each node can use a simple rule to assign itself one of two possible labels, so that labelling within each community is consistent and nodes in different communities are assigned different labels. Achieving this corresponds to *exact (block) reconstruction*. The next best guarantee is *weak (block) reconstruction*.

▶ **Definition 2** (Weak Reconstruction)**.** A function $f : V \to \{\pm 1\}$ is said to be an $\varepsilon$*-weak reconstruction* of $G$ if subsets $W_1 \subseteq V_1$ and $W_2 \subseteq V_2$ exist, each of size at least $(1-\varepsilon)n/2$, such that $f(W_1) \cap f(W_2) = \emptyset$.

We introduce a third notion, which we call *community-sensitive labeling* (CSL for short): in this case, there is a predicate that can be applied to pairs of labels so that, for all but a small fraction of outliers, the labels of any two nodes within the same community satisfy the predicate, whereas the converse occurs when they belong to different communities[1]. In this paper, informally speaking, nodes are labelled with binary signatures of logarithmic length, while two labels satisfy the predicate whenever their Hamming distance is below a certain threshold. This introduces a notion of similarity between nodes of the graph, with labels behaving like profiles that reflect community membership[2]. Note that this weaker notion of community-detection allows nodes to locally tell "friends" in their community from "foes" in the other community, which is the main application of distributed community detection in the opportunistic setting we consider here.

Let $\Delta(\mathbf{x}, \mathbf{y})$ denote the *Hamming distance* between two binary strings $\mathbf{x}$ and $\mathbf{y}$.

▶ **Definition 3** (Community-sensitive labeling)**.** Let $G = (V, E)$ be a graph, let $(V_1, V_2)$ be a partition of $V$ and let $\gamma \in (0, 1]$. For some $\ell \in \mathbb{N}$, a function $\mathbf{h} : V_1 \cup V_2 \to \{0, 1\}^\ell$ is a *$\gamma$-community-sensitive labeling* for $(V_1, V_2)$ if a subset $\tilde{V} \subseteq V$ with size $|\tilde{V}| \geqslant (1 - \gamma)|V|$ and two constants $0 \leqslant c_1 < c_2 \leqslant 1$ exist, such that for all $u, v \in \tilde{V}$ it holds that: $\Delta(\mathbf{h}_u, \mathbf{h}_v) \leqslant c_1 \ell$ if $i_u = i_v$, and $\Delta(\mathbf{h}_u, \mathbf{h}_v) \geqslant c_2 \ell$, otherwise, where $i_u = 1$ if $u \in V_1$ and $i_u = 2$ if $u \in V_2$.

## 3 First moment analysis

We analyze the expected behaviour of Algorithm Averaging(1/2) on an almost-regular graph $G$. The evolution of the resulting process can be formally described by the recursion $\mathbf{x}^{(t+1)} = W_t \cdot \mathbf{x}^{(t)}$, where $W_t = (W_t(i, j))$ is the random matrix that defines the updates of the values at round $t$, i.e.,

$$W_t(i,j) = \begin{cases} 0 & \text{if } i \neq j \text{ and } \{i,j\} \text{ is not sampled (at round } t\text{)}, \\ 1/2 & \begin{array}{l}\text{if } i = j \text{ and an edge with endpoint } i \text{ is sampled} \\ \text{or } i \neq j \text{ and edge } \{i,j\} \text{ is sampled,}\end{array} \\ 1 & \text{if } i = j \text{ and } i \text{ is not an endpoint of sampled edge.} \end{cases} \tag{1}$$

and the initial random vector $\mathbf{x}^{(0)}$ is uniformly distributed in $\{-1, 1\}^n$.[3] Note that, consequently, $\mathbf{x}^{(t+1)} = W_t \cdots W_1 \mathbf{x}^{(0)}$, with the $W_i$'s independently and identically distributed. Simple calculus shows that the expectation of the random matrices $\{W_t : t \geqslant 0\}$ can be expressed as

$$\overline{W} := \mathbb{E}[W_t] = I - \frac{1}{2m} L\,, \tag{2}$$

where $L = D - A$ is the Laplacian matrix of $G$. Matrix $\overline{W}$ is thus symmetric and doubly-stochastic. We denote its eigenvalues as $\bar{\lambda}_1, \ldots, \bar{\lambda}_n$, with $1 = \bar{\lambda}_1 \geqslant \bar{\lambda}_2 \geqslant \cdots \bar{\lambda}_n \geqslant -1$.

Our first contribution is an analysis of the expected evolution of the averaging process over $(n, d, \beta)$-almost regular graphs that possess a hidden and balanced partition of the

[1] Note that a weak reconstruction protocol entails a community-sensitive labeling. In this case, the predicate is true if two labels are the same.
[2] Hence the phrase *community-sensitive Labeling* we use to refer to our approach.
[3] Notice that, since each node chooses value $\pm 1$ with probability $1/2$ the first time it is active, by using the principle of deferred decisions we can assume there exists an "initial" random vector $\mathbf{x}^{(0)}$ uniformly distributed in $\{-1, +1\}^n$.

nodes with the following properties: (i) The cut separating the two communities contains $o(m)$ edges; (ii) the subgraphs induced by the two communities are expanders, i.e., the gap $\lambda_3 - \lambda_2$ is constant. The above conditions on the underlying graph are satisfied, for instance, by graphs sampled from the stochastic block model[4] $\mathcal{G}_{n,p,q}$ for $q = o(p)$ and $p \geqslant \log n/n$. Our analysis proves the following results.

▶ **Theorem 4.** *Let $G = (V, E)$ be an $(n, d, \beta)$-almost regular graph $G = (V, E)$ with a balanced partition $V = (V_1, V_2)$ and such that: (i) The cut $E(V_1, V_2)$ is sparse, i.e., $m_{1,2} = |E(V_1, V_2)| = o(m)$; (ii) $\lambda_3 - \lambda_2 = \Omega(1)$. If nodes of $G$ execute Protocol* Averaging *then, with constant probability w.r.t. the initial random vector $\mathbf{x}^{(0)} \in \{-1, 1\}^n$, after $\Theta(n \log n)$ rounds the following holds for all but $o(n)$ nodes:*
*(i) The expected value of a node $u$ increases or decreases depending on the community it belongs to, i.e., $\mathbf{sgn}\left(\mathbb{E}\left[\mathbf{x}_u^{(t-1)} \,|\, \mathbf{x}^{(0)}\right] - \mathbb{E}\left[\mathbf{x}_u^{(t)} \,|\, \mathbf{x}^{(0)}\right]\right) = \mathbf{sgn}\left(\boldsymbol{\chi}_u\right)$;*
*(ii) Over a suitable time window of length $\Omega(n \log n)$, the sign of the expected value of a node $u$ reflects the community $u$ belongs to, i.e., $\mathbf{sgn}\left(\mathbb{E}\left[\mathbf{x}_u^{(t)} \,|\, \mathbf{x}^{(0)}\right]\right) = \mathbf{sgn}\left(\alpha_2 \boldsymbol{\chi}_u\right)$, for some $\alpha_2 = \alpha_2(\mathbf{x}^{(0)})$.*

We note that these results suggest two different local criteria for community-sensitive labeling: (i) According to the first one, every node uses the sign of its own state within the aforementioned time window to set the generic component of its binary label (in fact, we run independent copies of the averaging process to get binary labels of logarithmic size - see Protocol Sign-Labeling in Section 5.1). (ii) According to the second criterion, every node uses the signs of fluctuations of its own value along consecutive rounds to set the generic component of its binary label (see Protocol Jump-Labeling in Section 5.2).

The above results describe the "expected" behaviour of the averaging process over a large class of well-clustered graphs, at the same time showing that our approach might lead to efficient, opportunistic protocols for block reconstruction. Yet, designing and analyzing protocols with provable, probabilistic guarantees, requires addressing the following questions: i) Do realizations of the averaging process approximately follow its expected behavior with high, or even constant, probability? ii) If this is the case, how can nodes locally and asynchronously recover the cut signal, let alone guess the "right" global time window? The first issue is addressed in Section 4, while the second one is addressed in Section 5, which presents our main algorithmic results for community detection.

## 4 Second Moment Analysis

Recall from Section 3 that $\mathbf{x}^{(t)}$ depends on the product of $t$ identically distributed random matrices. Not much is known about concentration of such products, but we are able to accurately characterize the class of regular clustered graphs. We point out that many of the technical results and tools we develop to this purpose apply to far more general settings than the regular case and may be of independent interest. In more detail, we are able to provide accurate concentration bounds on the norm of $\mathbf{x}^{(t)}$'s projection onto the subspace spanned by the first and second eigenvector of $\overline{W}$ for a class of regular clustered graphs that includes the *regular stochastic block model* [3, 5, 11]. These bounds are derived separately for two different regimes, defined by the sparseness of the cut separating the two communities. Assuming good inner expansion within each community, the first concentration result applies

[4] See the full-version [2] for the definition of $\mathcal{G}_{n,p,q}$ and for more details about our results for $\mathcal{G}_{n,p,q}$.

for cuts of size $o(m/\log^2 n)$ and it is given in Subsectioni 4.1 while, for the case of cuts of size up to $\alpha m$ for any $\alpha < 1$, the obtained concentration results are described in Subsection 4.2.

## 4.1 Second moment analysis for sparse cuts

We next provide a second moment analysis of the AVERAGING($\delta$) with $\delta = 1/2$ on the class of $(n, d, b)$-*clustered regular graphs* when the cut between the two communities is relatively sparse, i.e., for $\lambda_2 = 2b/d = o(\lambda_3/\log n)$. This analysis is consistent with the "expected" clustering behaviour of the dynamics explored in the previous section and highlights clustering properties that emerge well before global mixing time, as we show in Section 5.1.

Restriction to $(n, d, b)$-*clustered regular graphs* simplifies the analysis of the AVERAGING dynamics. When $G$ is regular, $\overline{W}$ defined in (2) can be written as $\overline{W} = \left(1 - \frac{1}{n}\right) I + \frac{1}{n} P = I - \frac{1}{n}\mathcal{L}$. This obviously implies that $\overline{W}$ and $\mathcal{L}$ share the same eigenvectors, while every eigenvalue $\lambda_i$ of $\mathcal{L}$ corresponds to an eigenvalue $\bar{\lambda}_i = 1 - \lambda_i/n$ of $\overline{W}$. For $(n, d, b)$-*clustered regular graphs*, these facts further imply $\bar{\lambda}_2 = 1 - \lambda_2/n = 1 - 2b/dn$ whenever $\lambda_3 > \frac{2b}{d}$ while, very importantly, the partition indicator vector $\boldsymbol{\chi}$ turns out to be the eigenvector of $\overline{W}$ corresponding to $\bar{\lambda}_2$ (see (2)). On the other hand, even in this restricted setting, our second moment analysis requires new, non-standard concentration results for the product of random matrices that apply to far more general settings and may be of independent interest.

For the sake of readability, in the remainder we denote $\mathbf{x}^{(t)}$'s projection onto $\mathbf{1}$ by $\mathbf{x}_{\|}$ and we use $\mathbf{y}^{(t)}$ to denote its component in the eigenspace of the second eigenvalue of $\overline{W}$ (i.e., $\boldsymbol{\chi}$).[5] Finally, we use $\mathbf{z}^{(t)}$ to denote $\mathbf{x}^{(t)}$'s projection onto the subspace orthogonal to $\mathbf{1}$ and $\boldsymbol{\chi}$. We thus have:

$$\mathbf{x}^{(t)} = \mathbf{x}_{\|} + \mathbf{y}^{(t)} + \mathbf{z}^{(t)}. \tag{3}$$

Our analysis of the process induced by AVERAGING(1/2) provides the following bound, whose proof can be found in the full-version [2].

▶ **Theorem 5** (Second moment analysis). *Let $G$ be an $(n, d, b)$-clustered regular graph with $\lambda_2 = \frac{2b}{d} = o(\lambda_3/\log n)$. Then, for every $\frac{3n}{\lambda_3}\log n \leqslant t \leqslant \frac{n}{4\lambda_2}$ it holds that*

$$\mathbb{E}\left[\left\|\mathbf{y}^{(t)} + \mathbf{z}^{(t)} - \mathbf{y}^{(0)}\right\|^2\right] \leqslant \frac{3\lambda_2 t}{n}.$$

We prove Theorem 5 by bounding and tracking the lengths of the projections of $\mathbf{x}^{(t)}$ onto the eigenspace of $\lambda_2$ and onto the space orthogonal to $\mathbf{1}$ and $\boldsymbol{\chi}$, i.e. $\|\mathbf{y}^{(t)}\|^2$ and $\|\mathbf{z}^{(t)}\|^2$. We here remark that the only part using the regularity of the graph is the derivation of the upper bound on $\mathbb{E}\left[\|\mathbf{y}^{(t+1)}\|^2\right]$, in particular its second addend. This term arises from an expression involving the Laplacian of $G$, which is far from simple in general, but that very nicely simplifies in the regular case. We suspect that increasingly weaker bounds should be achievable as the graph deviates from regularity.

Theorem 5 gives an upper bound on the squared norm of the difference of the state vector at step $t$ with the state vector at step 0. Intuitively, this will allow us to conclude that, for most vertices, $\mathbf{x}_v^{(t)} \approx \mathbf{x}_{\|,v} + \mathbf{y}_v^{(0)}$ over a time window of size $\Omega(n \log n)$. More formally, Corollary 4.1 below shows how such a *global* bound can be used to derive *pointwise* bounds on the values of the nodes.

[5] Note that $\mathbf{x}_{\|}$ is time-invariant.

▶ **Definition 6.** A node $v$ is *$\varepsilon$-good* at time $t$ if $(\mathbf{x}_v^{(t)} - (\mathbf{x}_{\|,v} + \mathbf{y}_v^{(0)}))^2 \leqslant \frac{\varepsilon^2}{n}\|\mathbf{y}^{(0)}\|^2$, it is *$\varepsilon$-bad* otherwise. We also define $B_t = \{u : u \text{ is } \varepsilon\text{-bad at time } t\}$.

▶ Corollary 4.1. Assume $3\frac{n}{\lambda_3}\log n \leqslant t \leqslant 3c\frac{n}{\lambda_3}\log n$ for any absolute constant $c \geqslant 1$ and $\lambda_2/\lambda_3 \leqslant \varepsilon^4/(4c\log n)$:

$$\mathbb{P}\left[|B_t| > \varepsilon n \,|\, \mathbf{x}^{(0)} = \mathbf{x}\right] \leqslant \varepsilon. \tag{4}$$

The next lemma strengthens the result above, giving a bound on the number of nodes that are good over a relatively large time-window. This is the key-property that we leverage to analyse the asynchronous protocol Sign-Labeling. The main idea of its proof is to first show that with probability strictly larger than $1-\varepsilon$, the number of $\varepsilon$-good nodes is at least $n \cdot (1-\varepsilon/\log n)$ in every round $t \in [t_1, 2t_1]$. Theorem 5 already ensures this to be true in any given time step within a suitable window, but simply taking a union bound will not work, since we have $n\log n$ time steps and only a $1-\varepsilon$ probability of observing the desired outcome in each of them. We will instead argue about the possible magnitude of the change in $\|\mathbf{y}^{(t)} + \mathbf{z}^{(t)} - \mathbf{y}^{(0)}\|^2$ over time, assuming this quantity is small at time $6\frac{n}{\lambda_3}\log n$. We will then show that our argument implies that, with probability $1-\varepsilon$, at least $n - \varepsilon n$ nodes remain $\varepsilon$-good over the entire window $[6\frac{n}{\lambda_3}\log n, 12\frac{n}{\lambda_3}\log n]$.

▶ **Lemma 7** (Non-ephemeral good nodes). *Let $\varepsilon > 0$ be an arbitrarily small value, let $G$ be an $(n,d,b)$-clustered regular graph with $\frac{\lambda_2}{\lambda_3} \leqslant \frac{\lambda_3\varepsilon^4}{c\log^2 n}$, for a large enough costant $c$. If we execute* Averaging(1/2) *on $G$, it holds that*
$\mathbb{P}\left[|B_t| \leqslant 3\varepsilon \cdot n\,, \forall t \,:\, 6\frac{n}{\lambda_3}\log n \leqslant t \leqslant 12\frac{n}{\lambda_3}\log n\right] \geqslant 1-\varepsilon.$

## 4.2 Second moment analysis for dense cuts

In this section, we extend our study to the lazy averaging algorithm Averaging($\delta$) where $\delta < 1/2$. Similar to the previous section, we assume that the underlying graph $G$ is an $(n,d,b)$-clustered regular graph and $\lambda_3 > \lambda_2 = 2b/d$. However, this new analysis works even for large (constant) $\lambda_2$, in contrast to that in Section 4.1 which only works for small $\lambda_2 \ll 1/\log^2 n$. Informally speaking, we show that, for an appropriate value of $\delta$ and any $t$ such that $\Omega(n\log n) \leqslant t \leqslant \mathcal{O}(n^2)$, with large probability, the vector $\mathbf{y}^{(t)} + \mathbf{z}^{(t)}$ is almost parallel to $\chi$, i.e., $\|\mathbf{z}^{(t)}\|$ is much smaller than $\|\mathbf{y}^{(t)}\|$. A more precise statement is given below as Theorem 8. Note that, for brevity, we write $\mathcal{E}$ here to denote the sequence $\{(u_t,v_t)\}_{t\in\mathbb{N}}$ of the edges chosen by the protocol.

▶ **Theorem 8.** *For any sufficiently large $n \in \mathbb{N}$, any[6] $\delta \in (0, 0.8(\lambda_3-\lambda_2))$ and any $t \in \left[\Omega\left(\frac{n}{\delta(\lambda_3-\lambda_2)}\log(n/\delta)\right), \mathcal{O}\left(\frac{n^2}{\delta(\lambda_3-\lambda_2)}\left(\frac{d(\lambda_3-\lambda_2)}{\delta b}\right)^{2/3}\right)\right]$, we have*
$\Pr_{\mathbf{x}^{(0)},\mathcal{E}}\left[\|\mathbf{z}^{(t)}\|^2 \leqslant \sqrt{\frac{\delta b}{d(\lambda_3-\lambda_2)}}\|\mathbf{y}^{(t)}\|^2\right] \geqslant 1 - \mathcal{O}\left(\sqrt[3]{\frac{\delta b}{d(\lambda_3-\lambda_2)}} + \frac{1}{\sqrt{n}}\right).$

Theorem 8 should be compared to Theorem 5: both assert that $\|\mathbf{y}^{(t)}\|$ is much larger than $\|\mathbf{z}^{(t)}\|$, but Theorem 8 works even when $\lambda_2$ is quite large whereas Theorem 5 only holds for $\lambda_2 \ll 1/\log^2 n$. While the parameter dependencies in Theorem 8 may look confusing at first, there are mainly two cases that are interesting here. First, for any error parameter $\varepsilon$, we can pick $\delta$ depending only on $\varepsilon$ and $\lambda_3 - \lambda_2$ in such a way that Theorem 8 implies that, with probability $1-\varepsilon$, $\|\mathbf{z}^{(t)}\|^2$ is at most $\varepsilon\|\mathbf{y}^{(t)}\|^2$, as stated below.

[6] Here 0.8 is arbitrary and can be changed to any constant less than 1. However, we pick an absolute constant here to avoid introducing another parameter to our theorem.

▶ Corollary 4.2. For any constant $\varepsilon > 0$ and for any $\lambda_3 > \lambda_2$, there exists $\delta$ depending only on $\varepsilon$ and $\lambda_3 - \lambda_2$ such that, for any sufficiently large $n$ and for any $t \in [\Omega_{\varepsilon,\lambda_3-\lambda_2}(n \log n), \mathcal{O}(n^2)]$, we have $\Pr_{\mathbf{x}^{(0)},\mathcal{E}} \left[ \|\mathbf{z}^{(t)}\|^2 \leqslant \varepsilon \|\mathbf{y}^{(t)}\|^2 \right] \geqslant 1 - \varepsilon$.

Another interesting case is when $\delta = 1/2$ (i.e., we consider the basic averaging protocol). Recalling that $\lambda_2 = 2b/d$, observe that $\lambda_2$ appears in both the bound on $\|\mathbf{z}^{(t)}\|^2$ and the error probability. Hence, we can derive a similar lemma as the one above, but with $\lambda_2$ depending on $\varepsilon$ instead of $\delta$:

▶ Corollary 4.3. Fix $\delta = 1/2$. For any constant $\varepsilon > 0$, any[7] $\lambda_3 > 0.7$, any sufficiently small $\lambda_2$ depending only on $\varepsilon$, any sufficiently large $n$ and any $t \in [\Omega_\varepsilon(n \log n), \mathcal{O}(n^2)]$, we have $\Pr_{\mathbf{x}^{(0)},\mathcal{E}} \left[ \|\mathbf{z}^{(t)}\|^2 \leqslant \varepsilon \|\mathbf{y}^{(t)}\|^2 \right] \geqslant 1 - \varepsilon$.

## 5 Distributed Community Detection

### 5.1 The Sign-Labeling protocol for sparse cuts

In the case of sparse cuts (i.e. of size $o(m/\log^2 n)$), the obtained bound on the variance of non-ephemeral nodes (see Lemma 7) holds over a time window that essentially equals the one "suggested" by our first moment analysis. Hence, we next propose a simple, lightweight opportunistic protocol that provides community-sensitive labeling for graphs that exhibit a relatively sparse cut.

The algorithm, denoted as Sign-Labeling (see Algorithm 2), adds a simple *labeling rule* to the Averaging(1/2) process: Each node keeps track of the number of times it is activated. Upon its $T$-th activation, for a suitable $T = \Theta(\log n)$, the node uses the sign of its current value as a binary label. The above local strategy is applied to $\ell$ independent runs of Averaging(1/2), so that every node is eventually assigned a binary signature of length $\ell$.

```
Sign-Labeling(T,ℓ) (for a node u that is one of the two endpoints of an active edge)
Component selection: Jointly with the other endpoint choose a component $j \in [\ell]$
  u.a.r.
Initialization and update: Run one step of Averaging (1/2) for component $j$.
Labeling: If this is the $T$-th activation of component $j$: set $\mathbf{h}_u^{sign}(j) = \mathbf{sgn}(\mathbf{x}_u(j))$.
```
**Algorithm 2:** Sign-Labeling algorithm.

Roughly, Lemma 7 implies that over a suitable time window of size $\Theta(n \log n)$, for all nodes $u$ but a fraction $\mathcal{O}(\varepsilon/\log n)$, we have $\mathbf{sgn}(\mathbf{x}_u^{(t)}) = \mathbf{sgn}(\mathbf{x}_{\|,u} + \mathbf{y}_u^{(0)})$. Recalling that $\mathbf{x}_\|$ and $\mathbf{y}^{(0)}$ respectively are $\mathbf{x}^{(0)}$'s projections along $\boldsymbol{\chi}/\sqrt{n}$ and $\mathbf{1}/\sqrt{n}$, this immediately implies that, with probability $1 - \varepsilon$ and up to a fraction $\varepsilon$ of the nodes, $\mathbf{sgn}(\mathbf{x}_u^{(t)}) = \mathbf{sgn}(\mathbf{x}_v^{(t)})$, whenever $u$ and $v$ belong to the same community and $t$ falls within the aforementioned window. As to the latter condition, we prove that each node labels itself within the right window with probability at least $1 - 1/n$.[8] Moreover, $\mathbf{sgn}(\mathbf{x}_{\|,u} + \mathbf{y}_u^{(0)}))) = \mathbf{sgn}(\boldsymbol{\chi}_u)$, whenever $\mathbf{y}_u^{(0)}$ exceeds $\mathbf{x}_{\|,u}$ in modulus, which occurs with probability $1/2 - o(1)$ from the (independent) Rademacher initialization. As a consequence, if we run $\ell$ suitably independent copies of the

[7] 0.7 here can be replaced by any constant larger than 0.5.

[8] It may be worth noting that $\mathbf{sgn}(\mathbf{x}_u^{(t)}) = \mathbf{sgn}(\mathbf{x}_v^{(t)})$ for $u$ and $v$ belonging to the same community does not imply $\mathbf{sgn}(\mathbf{x}_u^{(t)}) \neq \mathbf{sgn}(\mathbf{x}_v^{(t)})$ when they don't.

process, the following will happen for all but a fraction $\mathcal{O}(\varepsilon)$ of the nodes: the signatures of two nodes belonging to the same community will agree on $\ell - o(1)$ bits, whereas those of two nodes belonging to different communities will disagree on $\Omega(\ell)$ bits, i.e., our algorithm returns a community-sensitive labeling of the graph, as stated in the following theorem and corollary.

▶ **Theorem 9** (Community-sensitive labeling). *Let $\varepsilon > 0$ be an arbitrarily small value, let $G$ be an $(n, d, b)$-clustered regular graph with $\frac{\lambda_2}{\lambda_3} \leqslant \frac{\lambda_3 \varepsilon^4}{c \log^2 n}$, for a large enough constant $c$. Then, protocol* Sign-Labeling *$(T, \ell)$ with $T = (8/\lambda_3)\log n$ and $\ell = 10\varepsilon^{-1}\log n$ performs a $\gamma$-community-sensitive labeling of $G$ according to Definition 3 with $c_1 = 4\varepsilon$, $c_2 = 1/6$ and $\gamma = 6\varepsilon$, w.h.p. The convergence time is $\mathcal{O}(n\ell \log n/\lambda_3)$ and the work per node is $\mathcal{O}(\ell \log n/\lambda_3)$, w.h.p.*

Notice that, according to the hypothesis of Theorem 9, in order to set local parameters $T$ and $\ell$, nodes should know parameters $\varepsilon$ and $\lambda_3$ (in addition to a polynomial upper bound on the number of the nodes). However, it easy to restate it in a slightly restricted form that does not require such assumptions on what nodes know about the underlying graph.

▶ Corollary 5.1. Protocol Sign-Labeling $(80 \log n, 600 \log n)$ performs a $(1/10)$-community-sensitive labeling, according to Definition 3 with $c_1 = 1/15$ and $c_2 = 1/6$, of any $(n, d, b)$-clustered regular graph $G$ with $\lambda_3 \geqslant 1/10$ and $\lambda_2 \leqslant 1/(c \log^2 n)$ for a large enough constant $c$.

Observe that the "good" time-window begins after $\mathcal{O}(n \log n)$ rounds: So, if the underlying graph has dense communities and a sparse cut, nodes can collectively compute an accurate labeling *before* the global mixing time of the graph. For instance, if the cut is $\mathcal{O}(m/n^\gamma)$, for some constant $\gamma < 2$, our protocol is polynomially faster than the global mixing time. Importantly enough, the costs of our first protocol do not depend on the cardinality of the edge set $E$.

## 5.2 The Jump-Labeling protocol for dense cuts

The bound on the variance that allows us to adopt the sign-based criterion above does not hold when the cut is not sparse, i.e., whenever it is $\omega(m/\log^2 n)$. For such dense cuts, we use a different bound on the variance of nodes' values given in Theorem 8, which starts to hold after the global mixing time of the underlying graph and over a time window of length $\Theta(n^2)$. In this case, the specific form of the concentration bound leads to adoption of the second clustering criterion suggested by our first moment analysis, i.e., the one based on monotonicity of the values of non-ephemeral nodes. To this aim, we consider a "lazy" version of the averaging process equipped with a local clustering criterion, whereby nodes use the signs of fluctuations of their own values along consecutive rounds to label themselves (see Algorithm 3).

Here $\delta \in [0, 1]$ and $\tau^{\mathrm{s}}, \tilde{\tau}^{\mathrm{s}}, \tau^{\mathrm{e}}, \tilde{\tau}^{\mathrm{e}} \in \mathbb{N}$ are parameters that will be chosen later. Intuitively, protocol Jump-Labeling exploits the expected monotonicity in the behaviour of $\mathbf{sgn}(\mathbf{x}_u^{(t)} - \mathbf{x}^{(t-1)})$ highlighted in Section 3. Though this property does not hold for a single realization of the averaging process in general, the results of Section 4.2 allow us to show that the sign of $\mathbf{x}^{(\tau_u^e)} - \mathbf{x}^{(\tau_u^s)}$ reflects $u$'s community membership for most vertices with probability $1 - o(1)$ (i.e., the algorithm achieves weak reconstruction) when $\tau_u^s$ and $\tau_u^e$ are randomly chosen within a suitable interval. This is the intuition behind the main result of this section which is formalized below.

```
JUMP-LABELING($\delta, \tau^{s}, \tilde{\tau}^{s}, \tau^{e}, \tilde{\tau}^{e}$)
(for a node $u$ that is one of the two endpoints of an active edge)
Initialization: The first time it is activated, $u$ chooses $\tau_u^{s}, \tau_u^{e} \in \mathbb{N}$ independently
  uniformly at random from $[\tau^{s}, \tilde{\tau}^{s}]$ and $[\tau^{e}, \tilde{\tau}^{e}]$ respectively. Moreover, let $\tau_u = 0$.
Update (and AVERAGING's initialization): Run one step of AVERAGING($\delta$).
Labeling: If $\tau_u = \tau_u^{s}$, then set $x_u^{s} = x_u$. If $\tau_u = \tau_u^{e}$, then label $\mathbf{h}_u^{jump} = \mathbf{sgn}(x_u^{s} - x_u)$.
```

**Algorithm 3:** JUMP-LABELING Here, $\tau_u$ is a local counter keeping track of the number of times $u$ was an endpoint of an active edge, while $x_u$ is $u$'s current value.

▶ **Theorem 10.** *Let $n$ be any sufficiently large even positive integer. For any $0 < \delta < 0.8(\lambda_3 - \lambda_2)$, there exist $\tau^s, \tilde{\tau}^s, \tau^e, \tilde{\tau}^e \in \mathbb{N}$ such that, after $\mathcal{O}\left(\frac{n}{\delta(\lambda_3-\lambda_2)} \log(n/\delta) + \frac{nd}{b\delta}\right)$ rounds of protocol* JUMP-LABELING*$(\delta, \tau^s, \tilde{\tau}^s, \tau^e, \tilde{\tau}^e)$, every node labels its cluster and this labelling is a $\left(\sqrt[8]{\frac{\delta b}{d(\lambda_3-\lambda_2)}} + \sqrt[4]{\frac{1}{\log n}}\right)$-weak reconstruction of $G$, with probability at least $1 - \mathcal{O}\left(\sqrt[8]{\frac{\delta b}{d(\lambda_3-\lambda_2)}} + \sqrt[4]{\frac{1}{\log n}}\right)$. The convergence time of this algorithm is $\Omega_\delta\left(n\left(\log n + \frac{d}{b}\right)\right)$.*

**Proof of Theorem 10: an informal overview.** Since our discussion here will involve both local times and global times, let us define the following notation to facilitate the discussion: for each vertex $u \in V$, let $T_u : \mathbb{N} \to \mathbb{N}$ be a function that maps the local time of $u$ to the global time, i.e., $T_u(\tau) \triangleq \min\{t \in \mathbb{N} \mid |\{i \leqslant t \mid u \in \{u_i, v_i\}\}| \geqslant \tau\}$ where $(\{u_i, v_i\})_{i \in \mathbb{N}}$ is the sequence of active edges.

We let $a_y(t) \in \mathbb{R}$ be such that $\mathbf{y}^{(t)} = a_y(t) \cdot (\chi/\sqrt{n})$. Let us also assume without loss of generality that $a_y(0) \geqslant 0$. Observe first that our concentration result (Corollaries 4.2 and 4.3) implies the following: for any $t$ such that $\Omega(n \log n) \leqslant t \leqslant \mathcal{O}(n^2)$, with large probability, $\chi_u(\mathbf{x}_u^{(t)} - \mathbf{x}_{||,u})$ is roughly $\mathbb{E}_{\mathcal{E}}\, a_y(t)/n$ for most vertices $u \in V$; let us call these vertices *good for time t*. Imagine for a moment that we change the protocol in such a way that each $u$ has access to the global time $t$ and $u$ assigns $\mathbf{h}_u^{jump} = \mathbf{sgn}(\mathbf{x}_u^{(t^{e})} - \mathbf{x}_u^{(t^{s})})$ for some $t^{s}, t^{e} \in [\Omega(n \log n), \mathcal{O}(n^2)]$ that do not depend on $u$. If $t^{e} - t^{s}$ is large enough, then $\mathbb{E}_{\mathcal{E}}\, a_y(t^{s}) \gg \mathbb{E}_{\mathcal{E}}\, a_y(t^{e})$. This means that, if a vertex $u \in V$ is good at both times $t^{s}$ and $t^{e}$, then we have that $\chi_u(\mathbf{x}_u^{(t^{s})} - \mathbf{x}_{||,u}) \approx \mathbb{E}_{\mathcal{E}}\, a_y(t^{s})/n \gg \mathbb{E}_{\mathcal{E}}\, a_y(t^{e})/n \approx \chi_u(\mathbf{x}_u^{(t^{e})} - \mathbf{x}_{||,u})$. Note that when $\chi_u \cdot \mathbf{x}_u^{(t^{s})} > \chi_u \cdot \mathbf{x}_u^{(t^{e})}$, we have $\mathbf{h}_u^{jump} = \chi_u$. From this and from almost all vertices are good at both times $t^{s}$ and $t^{e}$, $\mathbf{h}^{jump}$ is indeed a good weak reconstruction for the graph!

The problem of the modified protocol above is of course that, in our settings, each vertex does not know the global time $t$. Perhaps the simplest approach to imitate the above algorithm in this regime is to fix $\tau^{s}, \tau^{e} \in [\Omega(\log n), \mathcal{O}(n)]$ and, for each $u \in V$, proceed as in JUMP-LABELING except with $\tau_u^{s} = \tau^{s}$ and $\tau_u^{e} = \tau^{e}$. In other words, $u$ assigns $\mathbf{h}_u^{jump} = \mathbf{sgn}(\mathbf{x}_u^{(T_u(\tau^{s}))} - \mathbf{x}_u^{(T_u(\tau^{e}))})$. The problem about this approach is that, while we know that $\mathbb{E}_{\mathcal{E}}\, T_u(\tau^{s}) = 0.5n\tau^{s}$ and $\mathbb{E}_{\mathcal{E}}\, T_u(\tau^{e}) = 0.5n\tau^{e}$, the actual values of $T_u(\tau^{s})$ and $T_u(\tau^{e})$ differ quite a bit from their means, i.e., on average they will be $\Omega(n\sqrt{\log n})$ of away their mean. Since our concentration result only says that, at each time $t$, we expect 99% of the vertices to be good, it is unclear how this can rule out the following extreme case: for many $u \in V$, $T_u(\tau^{s})$ or $T_u(\tau^{e})$ is a time step at which $u$ is bad. This case results in $\mathbf{h}^{jump}$ not being a good weak reconstruction of $V$.

The above issue motivates us to arrive at our eventual algorithm, in which $\tau_u^{s}$ and $\tau_u^{e}$ are not fixed to be the same for every $u$, but instead each $u$ pick these values randomly from specified intervals $[\tau^{s}, \tilde{\tau}^{s}]$ and $[\tau^{e}, \tilde{\tau}^{e}]$. To demonstrate why this overcomes the above problem, let us focus on the interval $[\tau^{s}, \tilde{\tau}^{s}]$. While $T_u(\tau^{s})$ and $T_u(\tilde{\tau}^{s})$ can still differ from their means, the interval $[T_u(\tau^{s}), T_u(\tilde{\tau}^{s})]$ still, with large probability, overlaps with most of

$[0.5n\tau^{\mathrm{s}}, 0.5n\tilde{\tau}^{\mathrm{s}}]$ if $\tilde{\tau}^{\mathrm{s}} - \tau^{\mathrm{s}}$ is sufficiently large. Now, if $T_u(\tau+1) - T_u(\tau)$ are the same for all $\tau \in [\tau^{\mathrm{s}}, \tilde{\tau}^{\mathrm{s}}]$, then the distribution of $\mathbf{x}_u^{(T_u(\tau^{\mathrm{s}}))}$ is very close to $\mathbf{x}_u^{(t_u^{\mathrm{s}})}$ if we pick $t_u^{\mathrm{s}}$ randomly from $[0.5n\tau^{\mathrm{s}}, 0.5n\tilde{\tau}^{\mathrm{s}}]$. From the usual global time step argument, it is easy to see that the latter distribution results in most $u$ being good at time $t_u^{\mathrm{s}}$. Of course, $T_u(\tau+1) - T_u(\tau)$ will not be the same for all $\tau \in [\tau^{\mathrm{s}}, \tilde{\tau}^{\mathrm{s}}]$, but we will be able to argue that, for almost all such $\tau$, $T_u(\tau+1) - T_u(\tau)$ is not too small, which is sufficient for our purpose. ◀

We remark that the $nd/b$ dependency in the running time is necessary. If we start with a good state where $\mathbf{x}^{(0)} = \mathbf{z}^{(0)} = 0$, then the values on one side of the partition are all $a_y(0)$ and the values on the other side are $-a_y(0)$. It is easy to see that, after $o(nd/b)$ steps of our protocol, $1 - o(1)$ fraction of the values remain the same. For these nodes, it is impossible to determine which cluster they are in and, hence, no good reconstruction can be achieved.

Similarly to our concentration results in Subsection 4.2, let us demonstrate the use of Theorem 10 to the two interesting cases. First, let us start with the case where $\lambda_3 - \lambda_2$ is constant.

▶ Corollary 5.2. For any constant $\varepsilon > 0$ and for any $\lambda_3, \lambda_2$, there exists $\delta$ depending only on $\varepsilon$ and $\lambda_3 - \lambda_2$ such that, for any sufficiently large $n$, there exists $\tau^{\mathrm{s}}, \tilde{\tau}^{\mathrm{s}}, \tau^{\mathrm{e}}, \tilde{\tau}^{\mathrm{e}} \in \mathbb{N}$ such that, with probability $1 - \varepsilon$, after $\mathcal{O}_{\varepsilon, \lambda_3 - \lambda_2}\left(n \log n + \frac{n}{\lambda_2}\right)$ rounds of JUMP-LABELING$(\delta, \tau^{\mathrm{s}}, \tilde{\tau}^{\mathrm{s}}, \tau^{\mathrm{e}}, \tilde{\tau}^{\mathrm{e}})$, every node labels its cluster and this labelling is a $\varepsilon$-weak reconstruction of $G$.

As in Subsection 4.2, we can consider the (non-lazy) averaging protocol and view $\lambda_2$ instead as a parameter. On this front, we arrive at the following reconstruction guarantee.

▶ Corollary 5.3. Fix $\delta = 1/2$. For any constant $\varepsilon > 0$, any $\lambda_3 > 0.7$, any sufficiently small $\lambda_2$ depending only on $\varepsilon$, any sufficiently large $n$, there exists $\tau^{\mathrm{s}}, \tilde{\tau}^{\mathrm{s}}, \tau^{\mathrm{e}}, \tilde{\tau}^{\mathrm{e}} \in \mathbb{N}$ such that, with probability $1 - \varepsilon$, after $\mathcal{O}_{\varepsilon}\left(n \log n + \frac{n}{\lambda_2}\right)$ rounds of JUMP-LABELING$(\delta, \tau^{\mathrm{s}}, \tilde{\tau}^{\mathrm{s}}, \tau^{\mathrm{e}}, \tilde{\tau}^{\mathrm{e}})$, the nodes' labelling is a $\varepsilon$-weak reconstruction of $G$.

While the weak reconstruction in the above claims is guaranteed only with arbitrarily-large constant probability, we can boost this success probability considering the same approach we used in Subsection 5.1.

Indeed, we first run $\ell = \Theta_\varepsilon(\log n)$ copies of JUMP-LABELING where, similarly to Algorithm 2, "running $\ell$ copies" of JUMP-LABELING means that each node keeps $\ell$ copies of the states of JUMP-LABELING and, when an edge $\{u, v\}$ is activated, $u$ and $v$ jointly sample a random $j \in [\ell]$ and run the $j$-th copy of JUMP-LABELING. In the previous section, we have seen that Lemma 7 and the repetition approach above allowed us to get a good community-sensitive labeling, w.h.p. (not a good weak-reconstruction). Interestingly enough, the somewhat stronger concentration results used in this section allow us to "add" a simple *majority* rule on the top of the $\ell$ components and get a "good" single-bit label, as described below. When all $\ell$ components of a node $u$ have been set, node $u$ sets $\mathbf{h}_u^{jump} = \text{MAJORITY}_{j \in [\ell]}(\mathbf{h}_u^{jump}(i))$ where $\mathbf{h}_u^{jump}(j)$ is the binary label of $u$ from the $j$-th copy of the protocol. Observe that the weak reconstruction guarantee of JUMP-LABELING shown earlier implies that the expected number of mislabelings of each copy is at most $2\varepsilon n$, i.e., $\mathbb{E}[\{u \in V \mid |\mathbf{h}_u^{jump}(i) \neq \chi_u|\}] \leqslant 2\varepsilon n$. Now, since the number of mislabelings of each copy is independent, the total number of mislabelings is at most $\mathcal{O}(\varepsilon n \ell)$, w.h.p. However, if the eventual label of $u$ is incorrect, it must contributes to mislabeling across at least $\ell/2$ copies. As a result, there are at most $\mathcal{O}(\varepsilon n)$ mislabelings in the new protocol, w.h.p.

▶ Corollary 5.4. For any constant $\varepsilon > 0$ and $\lambda_3 > \lambda_2$, there is a protocol that yields an $\varepsilon$-weak reconstruction of $G$ , w.h.p. The convergence time is $\Theta_{\varepsilon, \lambda_3 - \lambda_2}\left(n\left(\log^2 n + \frac{\log n}{\lambda_2}\right)\right)$ rounds, while the work per node is $\mathcal{O}_{\varepsilon, \lambda_3 - \lambda_2}\left(\log^2 n + \frac{\log n}{\lambda_2}\right)$.

We finally remark that, for the dense-cut case we focus on in this section (i.e. $\lambda_2 = 2b/d = \Theta(1)$), the fraction of outliers turns out to be a constant we can made arbitrarily small. If we relax the condition to $\lambda_2 = o(1)$, then this fraction can be made $o(1)$, accordingly.

## References

1 Dana Angluin, James Aspnes, David Eisenstat, and Eric Ruppert. The computational power of population protocols. *Distributed Computing*, 20(4):279–304, 2007.

2 Luca Becchetti, Andrea Clementi, Emanuele Natale, Francesco Pasquale, Prasad Raghavendra, and Luca Trevisan. Average whenever you meet: Opportunistic protocols for community detection. *CoRR*, abs/1703.05045, 2017. URL: `http://arxiv.org/abs/1703.05045`.

3 Luca Becchetti, Andrea Clementi, Emanuele Natale, Francesco Pasquale, and Luca Trevisan. Find your place: Simple distributed algorithms for community detection. In *Proc. of the 28th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA'17)*, pages 940–959. SIAM, 2017. `doi:10.1137/1.9781611974782.59`.

4 Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized Gossip Algorithms. *IEEE/ACM Transactions on Networking*, 14:2508–2530, 2006. URL: `http://dx.doi.org/10.1109/TIT.2006.874516`, `doi:10.1109/TIT.2006.874516`.

5 Gerandy Brito, Ioana Dumitriu, Shirshendu Ganguly, Christopher Hoffman, and Linh V. Tran. Recovery and Rigidity in a Regular Stochastic Block Model. In *Proc. of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 371–390. ACM, 2015.

6 Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84(6):066106, 2011.

7 Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

8 Laurent Massoulié. Community Detection Thresholds and the Weak Ramanujan Property. In *Proc. of the ACM Symposium on Theory of Computing (STOC)*, pages 694–703. ACM, 2014.

9 Elchanan Mossel, Joe Neeman, and Allan Sly. A proof of the block model threshold conjecture. *Combinatorica*, pages 1–44, 2013.

10 Elchanan Mossel, Joe Neeman, and Allan Sly. Belief propagation, robust reconstruction and optimal recovery of block models. In *Conference on Learning Theory*, pages 356–370, 2014.

11 Elchanan Mossel, Joe Neeman, and Allan Sly. Reconstruction and estimation in the planted partition model. *Probability Theory and Related Fields*, 162(3-4):431–461, 2015.

12 He Sun and Luca Zanetti. Distributed Graph Clustering and Sparsification. *CoRR*, abs/1711.01262, 2017. URL: `http://arxiv.org/abs/1711.01262`.

13 Matthew J. Williams, Roger M. Whitaker, and Stuart M. Allen. Decentralised detection of periodic encounter communities in opportunistic networks. *Ad Hoc Networks*, 10(8):1544–1556, 2012.