Brief Announcement: Maintaining a Bounded Degree Expander in Dynamic Peer-To-Peer Networks

Antonio Cruciani
Aalto University, Espoo, Finland

Abstract

We study the problem of maintaining robust and sparse overlay networks in fully distributed settings where nodes continuously join and leave the system. This scenario closely models real-world unstructured peer-to-peer networks, where maintaining a well-connected yet low-degree communication graph is crucial. We generalize a recent protocol by Becchetti et al. [SODA 2020] that relies on a simple randomized connection strategy to build an expander topology with high probability to a dynamic networks with churn setting. In this work, the network dynamism is governed by an oblivious adversary that controls which nodes join and leave the system in each round. The adversary has full knowledge of the system and unbounded computational power, but cannot see the random choices made by the protocol. Our analysis builds on the framework of Augustine et al. [FOCS 2015], and shows that our distributed algorithm maintains a constant-degree expander graph with high probability, despite a continuous adversarial churn with a rate of up to $\mathcal{O}(n/polylog(n))$ per round, where n is the stable network size. The protocol and proof techniques are not new, but together they resolve a specific open problem raised in prior work. The result is a simple, fully distributed, and churn-resilient protocol with provable guarantees that align with observed empirical behavior.

2012 ACM Subject Classification Theory of computation → Distributed algorithms; Networks → Peer-to-peer networks; Theory of computation → Expander graphs and randomness extractors

Keywords and phrases Peer-to-peer network, dynamic network, churn, distributed algorithm, randomized algorithm

Digital Object Identifier 10.4230/LIPIcs.DISC.2025.53

Related Version *Full Version*: `https://arxiv.org/abs/2506.17757` [6]

Funding *Antonio Cruciani*: This work was supported in part by the Research Council of Finland, Grant 363558.

1 Introduction

In this brief announcement, we study a simple *dynamics* that builds and maintains sparse and well connected graph despite an *adversarial churn* at each round.

As a motivating example, consider Peer-to-Peer (P2P) networks that are (for example) at the base of major blockchain systems such as Bitcoin and Ethereum in which the system provides decentralized communication and resilience against single points of failure. These overlays are governed by strict privacy rules restricting any leak of sensitive information about the peers. Compliance to such privacy rules essentially makes it difficult to employ any distributed protocol that explicitly uses standard message passing techniques. For example (say in the P2P Bitcoin Network), it is not possible to use random walks to "uniformly spread" the node IDs (in this case IP Addresses) on the network and provide peers with a fresh set of uniform IDs from which to sample new neighbors. Indeed, after an initial bootstrap phase in which the nodes rely on DNS seeds for node discovery, nodes running the Bitcoin-core implementation turn to a decentralized policy to rebuild their neighborhood

when their degree drops below the configured threshold. Each node has a minimum and a maximum number of neighbors that they must have (respectively 8 and 125, in the default configuration) and it locally stores a large list of IP addresses of active nodes. Every time the number of current neighbors of a node drops below the configured minimum value, it tries to find new connections with nodes sampled from its list. Such a list, is periodically shared to its neighbors and updated with the lists received by the neighbors. In the long run each node samples its neighbors from a list that represent a sufficiently random subset of nodes of the network. This sampling-based mechanism, despite being decentralized and oblivious to the global structure of the network, enables the construction of a sparse and robust overlay that induce good expansion properties. In particular, Becchetti et al. [5] formalized and analyzed a dynamic random graph model inspired by the Bitcoin protocol, showing that the resulting network converges rapidly to a constant-degree expander with high probability.

However, real-world P2P networks are inherently dynamic: nodes frequently join and leave the system over time. In this brief announcement, we address an open question posed by Becchetti et al. [5] by adapting their RAES protocol to a dynamic setting with adversarial churn. Our solution builds directly on the analytical framework developed by Augustine et al. [2], and shows that a RAES-style protocol can maintain expansion with high probability even under continuous adversarial churn. This work serves as a theoretical closure to earlier experimental studies of RAES variants under churn [8]. While the protocol and techniques are not original–both are adapted from prior work–the analysis formally confirms the empirical behavior observed in simulations. The result is a modest but rigorous endpoint to a research line focused on the resilience of randomized overlay networks.

2 Preliminaries

A *dynamic graph*[1] $\mathcal{G}$ is a family of graphs $\mathcal{G} = \{G_t = (V_t, E_t) : t \in \mathbb{N}\}$. We call G_t the *snapshot* of the dynamic graph at time t. For a set of nodes $S \subseteq V_t$, the *edge-expansion* of the snapshot G_t is defined as the minimum over all set of nodes $S \subseteq V_t$ whose size is $0 < |S| \leq n/2$ of the ratio between the number of edges leaving the set and its size.

The Dynamic Network with Churn Model. We consider a synchronous dynamic network controlled by an *oblivious* adversary. The adversary fixes a sequence of graphs $\mathcal{H} = (H_1, H_2, \dots, H_t, \dots)$ with each H_t defined on a vertex set V_t. The size of the vertex set is assumed to be stable. In other words, we have $|V_t| = n$ for all t. Furthermore, we assume that $|V_t \setminus V_{t+1}| = |V_{t+1} \setminus V_t| \in \mathcal{O}(n/\log^k n)$ where k is a constant determined in the analysis. The node set V_t is the set of nodes present in the network at round t. We assume that each node has a unique ID chosen from an ID space of size poly(n). The degree of each graph H_t is bounded by a constant δ. For the first $B = polylog(n)$ rounds, the adversary is *silent*, i.e., there is no churn, more precisely $H_1 = H_2 = \cdots = H_B$. Such an initial period is called *bootstrap phase*, and we can think of this phase as a period of stability during which the protocol builds and initial bounded degree expander graph. Subsequently, the network is said to be in the *maintenance phase* during which it can experience churn in the sense that a large number of nodes might join and leave dynamically at each step. During the maintenance phase, we require that any new node v that enters the network at round t must be connected in H_t to at least one pre-existing node v, i.e., if $u \in V_t \setminus V_{t-1}$, then u must be connected to some $v \in V_{t-1} \cap V_t$ in round t. The adversary must ensure that the degree

[1] In this paper we will use dynamic-graph and dynamic-network interchangeably.

bound δ is not violated. We only require that a new node is connected to a pre-existing node in its very first round upon entering the network. Secondly, after the bootstrap phase we do not require H_t to be connected. Indeed, the only required edges in H_t are the edges connecting new nodes to pre-existing ones. This is a standard requirement in the Dynamic Network with Churn (DNC) model (see, for example, [3, 2]) and prevents the adversary to create disconnected clusters. Our goal is to design a simple protocol that maintains a graph process $\mathcal{G} = \{G_t = (V_t, E_t)\}_{t \geq 1}$ over time such that each G_t for $t \geq B$, with high probability, has good expansion while maintaining a constant degree bound for each node between d and Δ such that $d < \Delta$ are two constants. We assume that, at time t, all the nodes have the ability of sampling from a uniform distribution over the vertex set at time t.

Remark. Although the assumption that a node can pick its neighbors uniformly at random among all nodes of the network is unrealistic in many scenarios, the edge creation processes in our model is reminiscent of the way some unstructured peer-to-peer networks such as the Bitcoin Network maintain a "random" topology.

The RAES Protocol. *RAES (Request a link, then Accept if Enough Space)* [5] is a random graph model defined by three parameters $n \in \mathbb{N}, d \in \{1, \ldots, n-1\}, c > 1$, in which each one of n nodes has degree at least d and at most $cd = \Delta$. The random graph is generated according to the discrete random process described in Algorithm 1. The process terminates when all nodes have degree in $[d, \Delta]$.

Algorithm 1 Overview of the RAES protocol.

```
1 Let G = (V, {∅}).
2 foreach t ≥ 0 do
3 |   Phase 1 (reconnection): Each node u with degree d(u) < d, picks d − d(u) new
  |     neighbors uniformly at random.
4 |   Phase 2 (degree adjustment): Each node u with degree d(u) > Δ selects
  |     d(u) − Δ neighbors u uniformly at random and drops the connection with them.
5 end
```

The RAES model can be seen as a simplified version of the network-formation process implemented in Bitcoin-core [9, 1]. The protocol converges in $\mathcal{O}(\log n)$ rounds to an expander graph with high probability [5]. However, this is not guaranteed to happen when nodes continuously join and leave the network.

3 Our Contribution

We address the problem of defining a variation of the RAES protocol that can build and maintain a dynamic expander graph despite an adversarial churn of $\mathcal{O}(n/\log^k n)$ at each round. Our construction relies on periodic randomized neighbor refreshing and pruning of high-degree nodes, and remains lightweight in terms of memory and messaging overhead. Our solution tackles one of the open problems in [5] by building directly on the framework developed by Augustine et al. [2].

The D-RAES Protocol. We now introduce D-RAES, a dynamic extension of the RAES protocol that is resilient to a continuous adversarial churn of up to $\mathcal{O}(n/\mathit{polylog}(n))$ nodes per round. The execution of the protocol is divided into two main phases: a bootstrap phase and

a maintenance phase. During the bootstrap phase, the goal is to construct a bounded-degree expander graph. In the maintenance phase, the protocol continually repairs and adapts the network in response to churn events. The bootstrap phase begins with a graph of n nodes and no edges. We run the RAES protocol (Algorithm 1) for $B = \mathcal{O}(\log n)$ rounds. As shown in [5], this results in an expander graph whp. (see Theorem 2.1 in [5]). After this phase, the adversary is allowed to churn nodes, and each node executes a maintenance cycle composed of three phases (Algorithm 2).

Algorithm 2 Overview of the D-RAES protocol.

Let G be the graph obtained after the Bootstrap Phase.
foreach $t > B$ **do**
  Phase 1 (refresh neighbors): With probability $1/\log^k n$ each node u with degree $d(u) \in [d, \Delta]$ drops all its neighbors.
  Phase 2 (reconnection): Each node u with degree $d(u) < d$, picks $d - d(u)$ new neighbors uniformly at random.
  Phase 3 (degree adjustment): Each node u with degree $d(u) > \Delta$ selects $d(u) - \Delta$ neighbors u uniformly at random and drops the connection with them.
end

The high level idea of the protocol is that each node in the network seeks to maintain a bounded degree graph by keeping a degree between the two parameters d and Δ with $d < \Delta$. This simple procedure mimics the behavior of the Bitcoin Network formation process [1]. We require that nodes in the network that were not churned by the adversary must refresh their neighbor list with probability $1/\mathit{polylog}(n)$. This is fundamental to avoid the oblivious adversary incrementally building bad structures that can gradually degrade and ultimately destroy the expansion of the graph. Thus, we require that on average, nodes must refresh their neighborhood every $\mathit{polylog}(n)$ rounds. Indeed, the authors in [7, 8] experimentally showed that refreshing long-lasting edges in the network might help the process to recover after a churn. The protocol uses a simple mechanism to understand when it is time to reconnect: when the number of neighbors falls below d it infers that is not well-connected anymore and it tries to find new random neighbors from the network. In addition, the protocol preserves bounded degrees by "pruning" neighbors of nodes that exceed Δ.

As mentioned above, our goal is to keep the network close to a random constant-degree graph at every round, w.h.p. Throughout, we fix the degree parameters d and Δ as constants with $d < \Delta$. Beyond the reconnection triggered by lost edges, in every round each node independently, with probability $\Theta(1/\mathit{polylog}(n))$, simply deletes all its edges and reconnects again. This refreshing step continuously creates new "random enough" edges in the network, which is useful in showing expansion properties in each round. Moreover, it is possible to show that if we do not include such a phase in the protocol (or an analogous step) it is impossible to maintain a constant degree expander graph under adversarial churn.

▶ **Theorem 1.** *Assume that after the bootstrap phase, the graph G_t is a constant-degree expander where every node has degree in $[d, \Delta]$, where $d < \Delta$ are suitable constants. Suppose that the maintenance protocol does not include any edge-refreshing phase. Then, there is an adversarial strategy that (1) maintains a constant degree graph in which each node has degree in $[d, \Delta]$ and (2) the graph is not anymore an expander graph.*

We show a high-level overview of the proof. The core idea is that there is an adversarial strategy that gradually erodes the well connected part of the graph while maintaining a bounded-degree graph. Without the edge-refresh phase (and thus using the RAES protocol

as it is in Algorithm 1) the adversary can: churn out some nodes from the graph and immediately insert $n/polylog(n)$ nodes by connecting these new nodes to a low-expansion region and avoid touching the core that keeps being an expander. However, over multiple rounds the well-connected part of the graph shrinks due to the churn while the weakly connected fringe grows. Since we do not have any mechanism to re-randomize the bad edges we have that the expansion collapses even though the graph is still a bounded-degree graph. Furthermore, we observe that a similar lower bound holds even when the adversary is oblivious, i.e., has no knowledge of the graph structure.

▶ **Observation 2.** *The lower bound in Theorem 1 holds under oblivious adversary.*

The graph produced after each iteration of Algorithm 2 (lines 2-6), is the result of the interaction between two entities: the oblivious adversary and Algorithm 2. To analyze the distributed protocol behavior under churn, we use the same approach of Augustine et al. [2] and we define a family of processes whose behavior is: (1) the adversary's move; (2) the actions by the nodes; and, (3) the random bits available to the process itself. As in [2], we refer to a member of this family of processes as the Υ-process.

Overview of the process. We present a high-level description of the process and intuitions behind its analysis and we refer to the full version of this work [6] for a complete description and analysis of the process. The main idea is that the process captures both, the distributed protocol's and the adversary's actions by first applying the adversarial churn to the graph and then steps (1-3) in Algorithm 2 at each round. We have that the Υ-process starts from a bounded degree expander graph that was created during the bootstrap phase. At each iteration it generates a new graph $G_t = (V_t, E_t)$, given the graph at $t-1$. The process ensures that nodes with a degree less than d and a life-span greater than $\Omega(\log n)$ will gain at least once a degree between d and Δ in $\mathcal{O}(\log n)$ rounds. Moreover, to argue that at each round we have a large expander subgraph we have to take into account that: (1) the (adversarial) churn is $\mathcal{O}(n/\log^k n)$, (2) the number of nodes that choose to refresh their neighborhood is $\mathcal{O}(n/\log^k n)$ whp., and (3) the number of "unlucky" nodes that after the reconnection phase had degree exactly d and that after the degree adjustment phase lost some neighbors (due to some highly connected node with degree greater than Δ dropping a random subset of their neighbors) is $\mathcal{O}(n/\log^k n)$ whp. By iteratively using the results by Bagchi et al. in [4], the fact that nodes sample from a uniform distribution over V_t and that no node can have degree less than d for $\Omega(\log n)$ consecutive rounds whp. we can argue that at the end of each iteration t, the Υ process produces a graph G_t that contains a large core with good expansion properties. This result implies that the D-RAES protocol maintains a graph with a $(n - o(n))$-sized constant expander subgraph in which all the nodes have degree between d and Δ. Our main result can be summarized as follows.

▶ **Theorem 3** (Main Theorem). *Let d and Δ be two suitable constants such that $d < \Delta$. Despite an adversarial churn rate of $\mathcal{O}(n/\log^k n)$ for any $k \geq 1$, the D-RAES protocol maintains a dynamic graph $\mathcal{G} = \{G_t = (V_t, E_t)\}_{t\geq 1}$ such that, with high probability, for at least n^c rounds (for any arbitrarily large constant $c \geq 1$), each snapshot G_t contains a connected subgraph C_t on $n - o(n)$ nodes that satisfies:*

- *each node in C_t has degree in $[d, \Delta]$;*
- *C_t has edge expansion of at least α, for some constant $\alpha > 0$.*

Our analysis and guarantees also hold under an adaptive adversary, provided that nodes always have access to uniform sampling over the current vertex set at each round. In this setting, although the adversary can observe the graph structure and churn history before

making decisions, it cannot predict or influence the outcome of random sampling. This implies that nodes can still establish fresh, well-distributed connections, and our edge-refreshing mechanism ensures that poorly connected regions do not persist. As a result, the same arguments used under the oblivious adversary extend to the adaptive case. However, we emphasize that this holds only under perfect uniform sampling. If uniform sampling is replaced with random walk-based sampling (as in [2]), an adaptive adversary can bias walk trajectories, making the analysis significantly more delicate and outside the scope of this work.

4 Conclusion

We presented a fully distributed and lightweight protocol for maintaining a constant-degree expander graph despite a continuous adversarial churn of up to $\mathcal{O}(n/polylog(n))$ nodes per round. Despite this strong adversary, our protocol ensures that the resulting overlay graph remains an expander with high probability over time. Moreover, this work demonstrates that robustness and good connectivity can be achieved under far weaker assumptions, without relying on global knowledge or heavy coordination among nodes. Our analysis suggests that, if nodes in the network periodically refresh their neighborhood, the Bitcoin Network creation Protocol can maintain a good expander graph that preserves a good edge expansion against heavy adversarial churn by simply executing a local dynamics in which nodes do not need any information about the overall network structure. Finally, our result closes an open problem in [5] where the authors suggested to study their protocol under churn settings.

References

1 Bitcoin core p2p network. URL: `https://en.wikipedia.org/wiki/Bitcoin_Core`.

2 John Augustine, Gopal Pandurangan, Peter Robinson, Scott T. Roche, and Eli Upfal. Enabling robust and efficient distributed computation in dynamic peer-to-peer networks. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*. IEEE Computer Society, 2015. `doi:10.1109/FOCS.2015.29`.

3 John Augustine, Gopal Pandurangan, Peter Robinson, and Eli Upfal. Towards robust and efficient computation in dynamic peer-to-peer networks. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*. SIAM, 2012. `doi:10.1137/1.9781611973099.47`.

4 Amitabha Bagchi, Ankur Bhargava, Amitabh Chaudhary, David Eppstein, and Christian Scheideler. The effect of faults on network expansion. *Theory Comput. Syst.*, 2006. `doi:10.1007/S00224-006-1349-0`.

5 Luca Becchetti, Andrea Clementi, Emanuele Natale, Francesco Pasquale, and Luca Trevisan. Finding a bounded-degree expander inside a dense one. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*. SIAM, 2020. `doi:10.1137/1.9781611975994.80`.

6 Antonio Cruciani. Maintaining a bounded degree expander in dynamic peer-to-peer networks, 2025. `doi:10.48550/arXiv.2506.17757`.

7 Antonio Cruciani and Francesco Pasquale. Brief announcement: Dynamic graph models for the bitcoin P2P network: Simulation analysis for expansion and flooding time. In *Stabilization, Safety, and Security of Distributed Systems - 24th International Symposium, SSS 2022, Clermont-Ferrand, France, November 15-17, 2022, Proceedings*, Lecture Notes in Computer Science. Springer, 2022. `doi:10.1007/978-3-031-21017-4_23`.

8 Antonio Cruciani and Francesco Pasquale. Dynamic graph models inspired by the bitcoin network-formation process. In *24th International Conference on Distributed Computing*

and Networking, ICDCN 2023, Kharagpur, India, January 4-7, 2023. ACM, 2023. doi:10.1145/3571306.3571398.

9 Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2009. URL: http://www.bitcoin.org/bitcoin.pdf.