Robust and Efficient Computation in Dynamic Networks with Heavy Churn

John Augustine

Collaborators & Publications

Tejas Kulkarni

Anisur Rahaman Molla

Ehab Morsy

Gopal Pandurangan

Peter Robinson

Scott Roche

Sumathi Sivasubramaniam

Eli Upfal

SODA 2012

SPAA 2013

PODC 2013

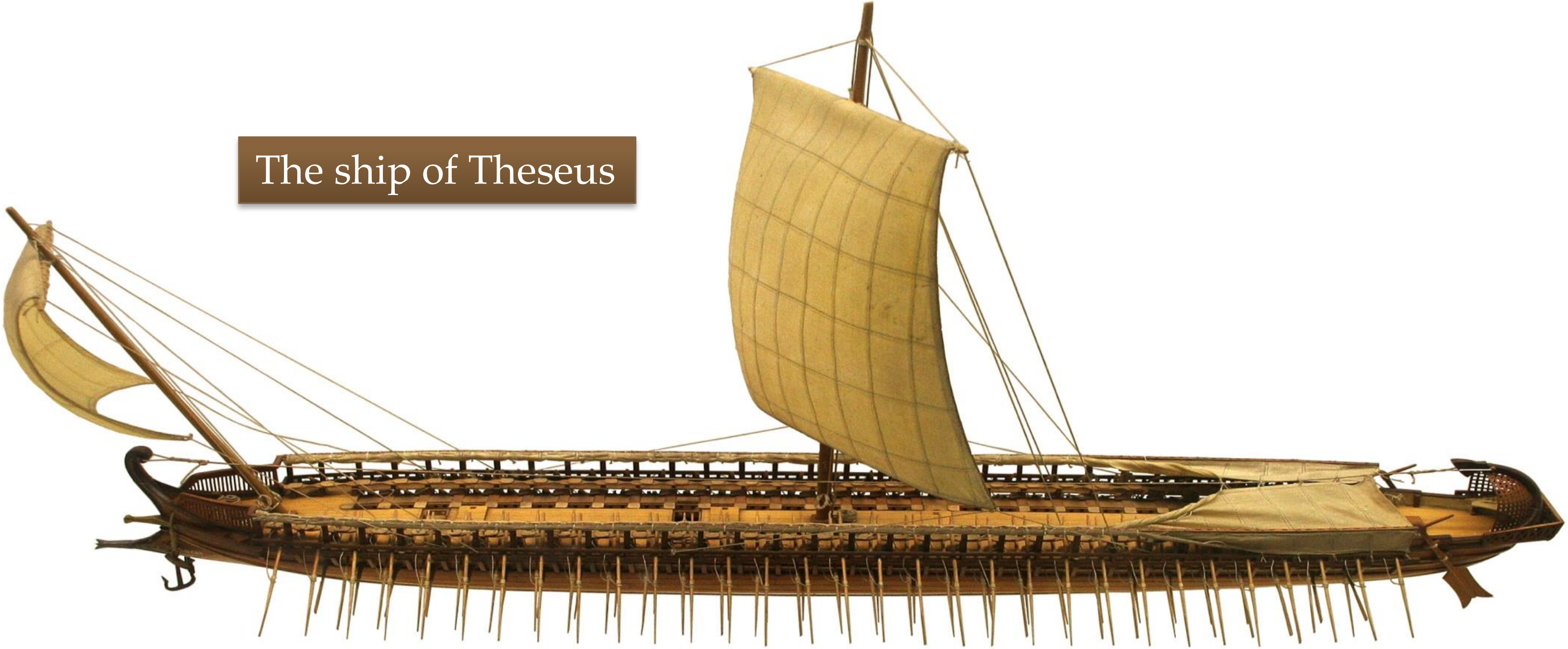
DISC 2015

IPDPS 2015

FOCS 2015

SIGACT News 47(1) 2016

The ship of Theseus

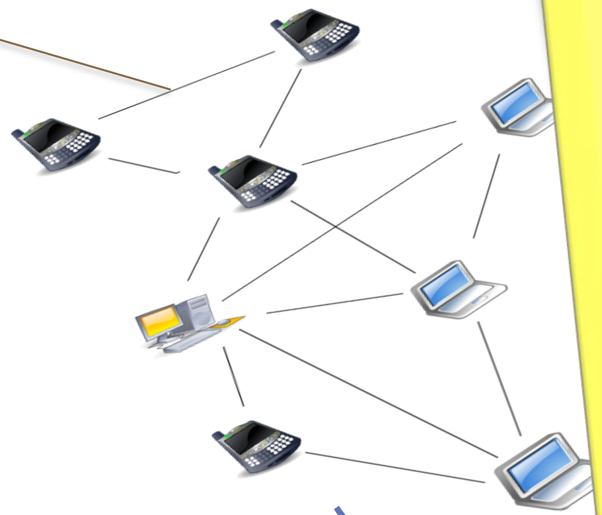


cell type	turnover time
small intestine epithelium	2-4 days
stomach	2-9 days
blood Neutrophils	1-5 days
white blood cells Eosinophils	2-5 days
gastrointestinal colon crypt cells	3-4 days
cervix	6 days
lungs alveoli	8 days
tongue taste buds (rat)	10 days
platelets	10 days
bone osteoclasts	2 weeks
intestine Paneth cells	20 days
skin epidermis cells	10-30 days
pancreas beta cells (rat)	20-50 days
blood B cells (mouse)	4-7 weeks
trachea	1-2 months
hematopoietic stem cells	2 months
sperm (male gametes)	2 months
bone osteoblasts	3 months
red blood cells	4 months
liver hepatocyte cells	0.5-1 year
fat cells	8 years
cardiomyocytes	0.5-10% per year

Source: http://book.bionumbers.org/how-quickly-do-different-cells-in-the-body-replace-themselves/

Peer-to-Peer Networks

The Overlay provides organization



- Highly dynamic
- 50% churn every hour
- Network size stable

- Traditional distributed algorithms don't work.
- Need new models and algorithms

The Internet substrate provides communicat

Pieces of the P2P Puzzle

Extensive literature…

Overlay Maintenance

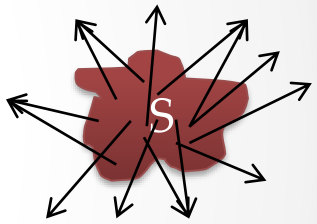
How to maintain a well-connected network despite heavy churn?

Good Expansion

→

Expander Graphs

Let $G = (V, E)$ be a graph on n nodes. Then G is an expander if, for *every* subset $S \subseteq V$ such that $|S| \leq n/2$:



$$\frac{|E(S, V - S)|}{|S|} \geq \alpha$$

for some constant $\alpha > 0$, where $E(S, \bar{S})$ is the number of edges with one endpoint in S and the other endpoint not in S.

In many applications, we want a sparse expander, i.e., the degree of each node is upper bounded by a constant.

Why Expanders?

A (regular degree) expander graph (on n nodes) has many desirable properties:

- Diameter is $O(\log n)$ — hence short paths between any two nodes.
- Highly resilient to adversarial deletions: Deleting even ϵn nodes (for some constant ϵ) will leave a $O(n)$ size connected graph which is also an expander! Thus an expander topology is very robust.
- Random walks mix very fast, i.e., in $O(\log n)$ steps, a random walk starting from any arbitrary node reaches essentially a random destination.

Main Problem: How to maintain an expander graph in a distributed network under heavy (adversarial) churn?

Constructing an Expander

Given n nodes, the following is a simple way to construct an $O(\log n)$-degree expander.

- Create random edges: For each node v, select $O(\log n)$ random nodes and make them neighbours of v.
- The above random graph is an expander with high probability (whp).

Challenges:

- Works for static networks, but what happens under adversarial churn ?
- How to maintain a constant degree expander graph ?

Related Works

A very quick rundown

Related Works

There have been a number of attempts to design algorithms to repair or maintain an expander network under disruption:

- Building a P2P network that can tolerate linear churn under a stochastic adversary. [Pandurangan, Raghavan, and Upfal, FOCS 2001]
- Distributed algorithm to construct a random expander graph, limited number of insertions or deletions. [Law and Siu, INFOCOM 2003]
- P2P network that can tolerate $O(\log n)$ adversarial churn rate. [Kuhn, Schmid, Wattenhofer, Distributed Computing, 2010]
- Repair a single node deletion/insertion in $O(\log n)$ time using $O(\log n)$ messages. [Pandurangan, Robinson, and Trehan, IPDPS 2014]

Related Works

The most similar work is a paper by [Cooper, Klasing, Radzik, Theoretical Computer Science, 2008]:

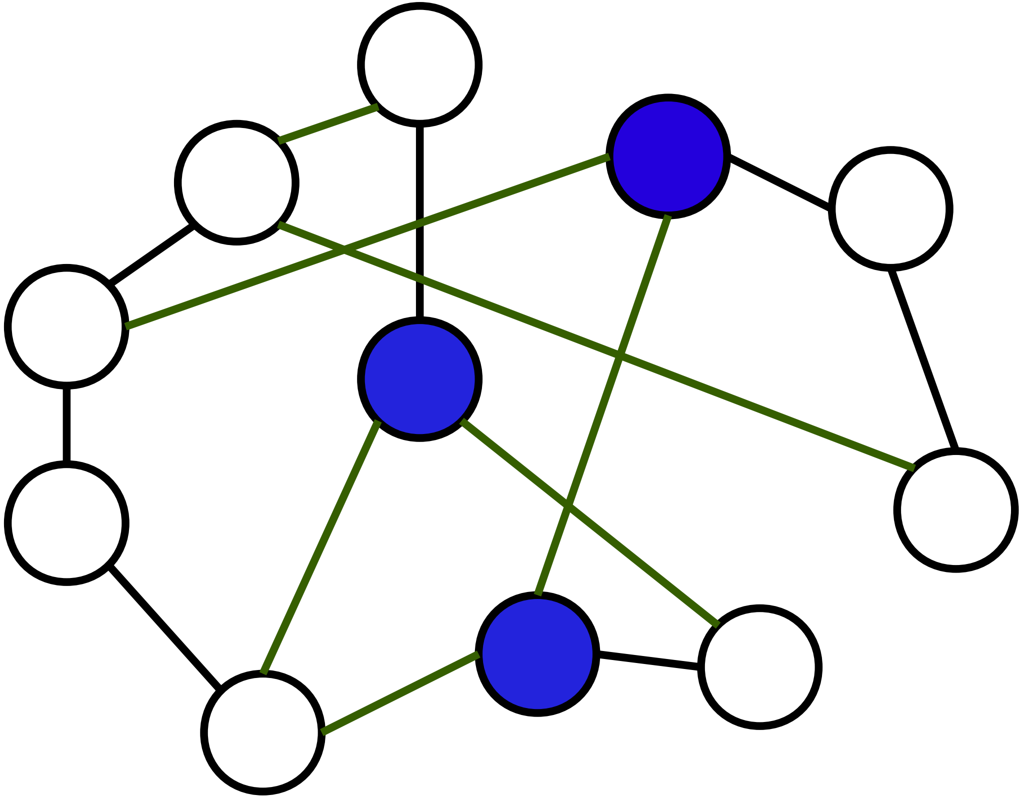
- Maintain well-connected, small-diameter graph under adversarial modification.
- Similar to our approach, uses random walks to provide a source of randomness in edge creation.
- Limitation: For more than small amount of churn, only connectivity is guaranteed.
- Limitation: Cannot handle high, continuous churn.

Related Works

- Recent work by Drees, Gmyr, and Scheideler (SPAA 2016) maintains overlay networks under high churn AND DoS attacks.
- Also related to distributed data structures like skip graphs (Aspnes and Shah, ACM TALG 2007) and their variants.
- A fairly large DHT literature dating back from late 90's.

The Model

Designed to allow the protocol to create and maintain an expander overlay network in an adversarial setting.



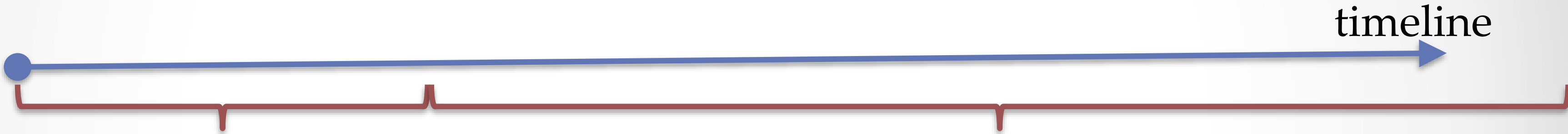
Adversarial Setting

Oblivious

[—, Pandurangan, Robinson, Roche, and Upfal; *FOCS* 2015]

- Adversary gives a "rudimentary" graph sequence $H_1, H_2, \ldots$, where $H_i = (V_i, E_i^H)$ is the graph in round i.
- Nodes have unique IDs and come with Δ ports.
- The number of nodes that change in a round is called the churn rate.
- The churn rate can be very high: up to $O(n/\text{polylog}(n))$ per round.
- Network size remains (essentially) constant.

Adversary’s Rules



<u>Bootstrap Phase</u>

- First O(log n) rounds
- No churn
- Network is an expander

<u>MaintenancePhase</u>

- Churn allowed, but
- New nodes connected to existing node
- Degree bound Δ respected.

Communication Model

- Synchronous model: computation/communication proceeds in a sequence of "handshake" rounds.
- Each edge can carry polylog(n) bits in each round.
- Direct Communication: If u knows v's ID (e.g., IP address) it can send a message.
- Overlay Communication when u and v are neighbours in the overlay network.
- Sparse network: Each node can only communicate with a constant number of nodes in any round.

Edge Creation

Suppose u knows the ID (IP address) of v and wants to establish edge (u, v). Then, . . .

- Node u sends an edge request to v.
- Node v can either:
  - Return an "accept" message $\Longrightarrow$ edge created, or
  - Ignore $\Longrightarrow$ edge not created.

Note: Both nodes must have spare ports.

Building an Expander

High-level Idea: Build a random graph in a distributed fashion.

Theorem 1. *Suppose that G is a graph on n nodes.*

- *Each node v initiates the creation of $\delta \in \Theta(1)$ edges.*
- *The other endpoint is drawn from a subset S, where $|S| \geq 0.8n$, with (almost) uniform probability distribution $\Theta(1/n)$.*

Then G is an expander with constant expansion $\alpha > 0$ with high probability.

Obtained via Random Walks

The above generalizes the static construction idea

→

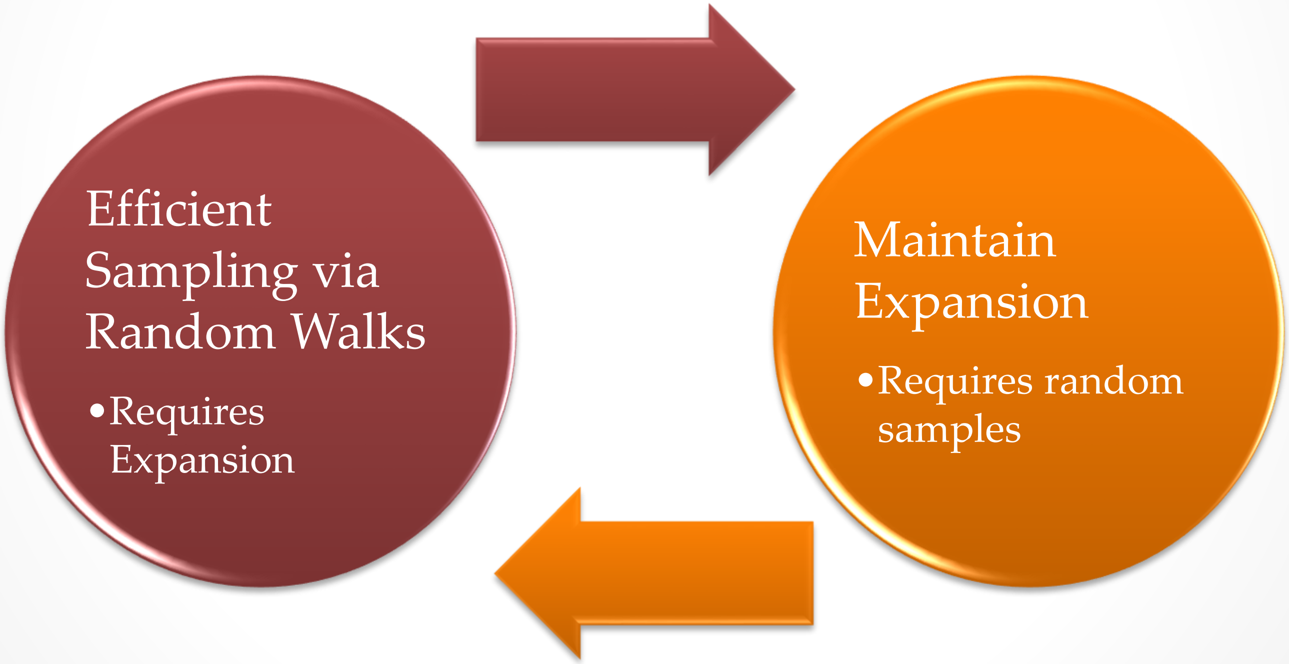
Random Walks Engine

- In every round, each node v generates a polylogarithmic (in n) number of tokens.
- Each token contains the origin address (v).
- Each token walks for $\tau \in \Theta(\log n)$ rounds (i.e., mixing time).
- After τ steps, stops at some node w.
- Placed in w's **token buffer** — mature (mixed) tokens. (Replaces oldest tokens in buffer.)
- A mature token can be used to create a random edge to v.

Sampling Lemma

- The Sampling Lemma: Most random walks mix well even in a highly dynamic adversarial network provided large expander subgraph.
- We show that the random walk sampling will generate near-uniform random samples from a large-sized subset of the nodes.

High Level Idea



Algorithmic Ingredients

Random Walks subroutine

- Runs in the background. Provides steady stream of samples.

Reconnect Operating Mode

- A node is churned in or loses edges or receives inadequate number of tokens. →

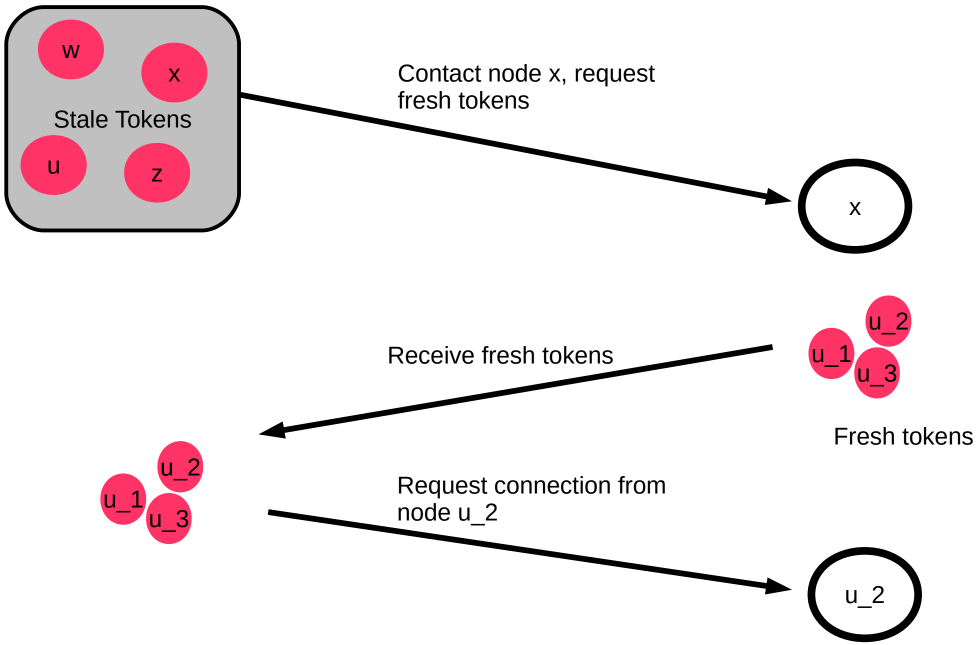
Normal Operating Mode

- All other times.

Challenges

- **(Re)Connect**: How will a new (or isolated node) (re)connect? Borrow and use mature tokens.
- **Out of mature tokens**: Use stale tokens to get mature tokens. $\longrightarrow$
- **Expansion Decay**: How to ensure expansion despite adversarial effort? Count the number of mature tokens received.

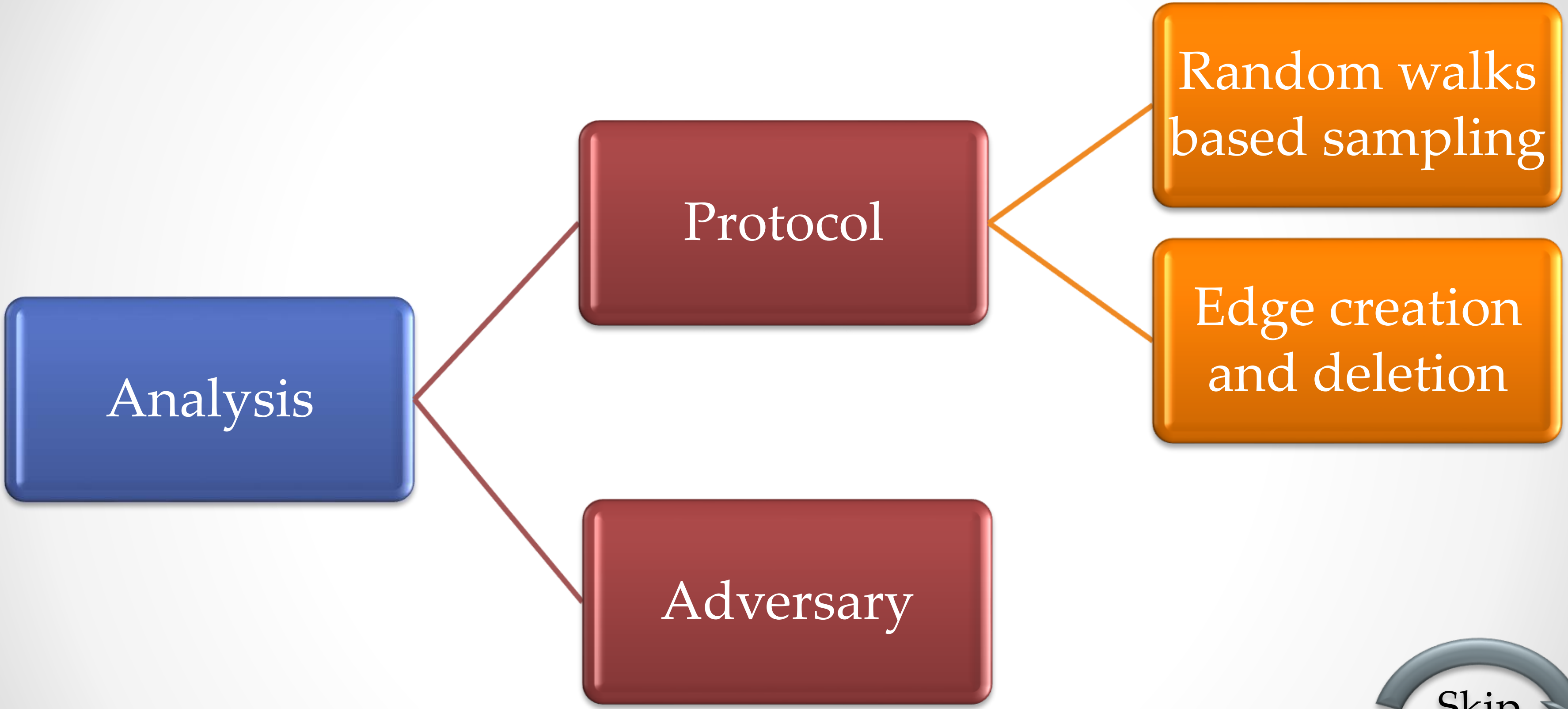
Illustration



Our Guarantees

Our maintenance guarantees that:

- No node has more than Δ degree.
- No node remains disconnected from the network for more than $O(\log n)$ rounds with high probability (i.e., with probability at least $1 - 1/n$).
- In every round, whp there is a large component of the network of size $n - o(n)$ whose induced subgraph has constant expansion $\alpha > 0$.


Analysis
Protocol
Random walks based sampling
Edge creation and deletion
Adversary

Skip Proof

Proof Idea:

- Difficult to work with the graph process produced by the protocol/adversary interaction G_i: Highly dynamic and not regular.
- Analyze a new graph process — $\bar{G}_i$ — that is regular (i.e., all nodes have Δ degree with no missing edges) and there is no churn.
  - Copy state of churned out nodes into churned in nodes.
  - Construct *ghost edges* (which are adversarially determined).

Proof Idea

- In $\bar{G}_i$ tokens mix well and have near-uniform origins.

- In G_i, if a node receives a lot of real tokens, then we show that they are likely to be well-mixed with near-uniform origins.

- The near-uniform distribution of tokens in G_i, is shown by appealing to the distribution of real tokens (not ghost tokens) in $\bar{G}_i$.

- Adversarial deletion of $\Theta(n/\text{polylog} n)$ nodes and edges from $\bar{G}_i$, leaves a large expander subgraph in G_i.

Now What???

What can we do with the P2P system that maintains an expander overlay?

Dynamic Network Model

[—, Pandurangan, Robinson, and Upfal; *SODA* 2012]

- Dynamic network is modelled as a graph process: $G_1, G_2, \ldots$, where $G_i = (V_i, E_i)$ is the graph in round i.
- An adversary controls the topology of every graph G_i, including the set of nodes, with the restriction that each G_i is an expander graph.
- The number of nodes that change in a round is called the churn rate.
- The churn rate can be very high: up to $O(n/\text{polylog}(n))$ per round.
- Network size remains (essentially) constant.

Could be higher

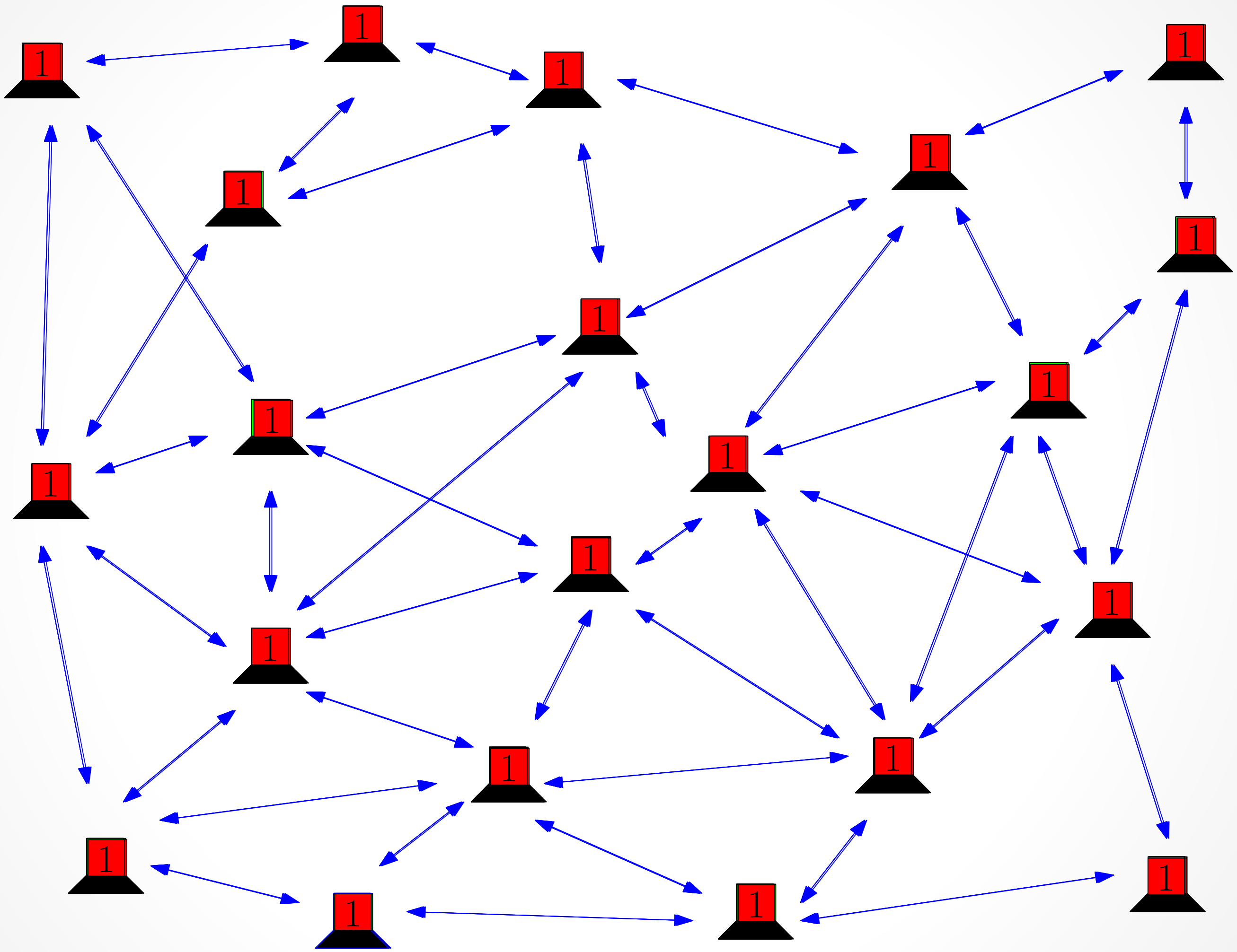
Communication Model

- Synchronous model: computation/communication proceeds in a sequence of rounds.
- Each edge can carry polylog(n) bits in each round.
- Direct Communication: If u knows v's ID (e.g., IP address) it can send a message.
- Overlay Communication when u and v are neighbours in the overlay network.
- Sparse network: Each node can only communicate with a constant number of nodes in any round.

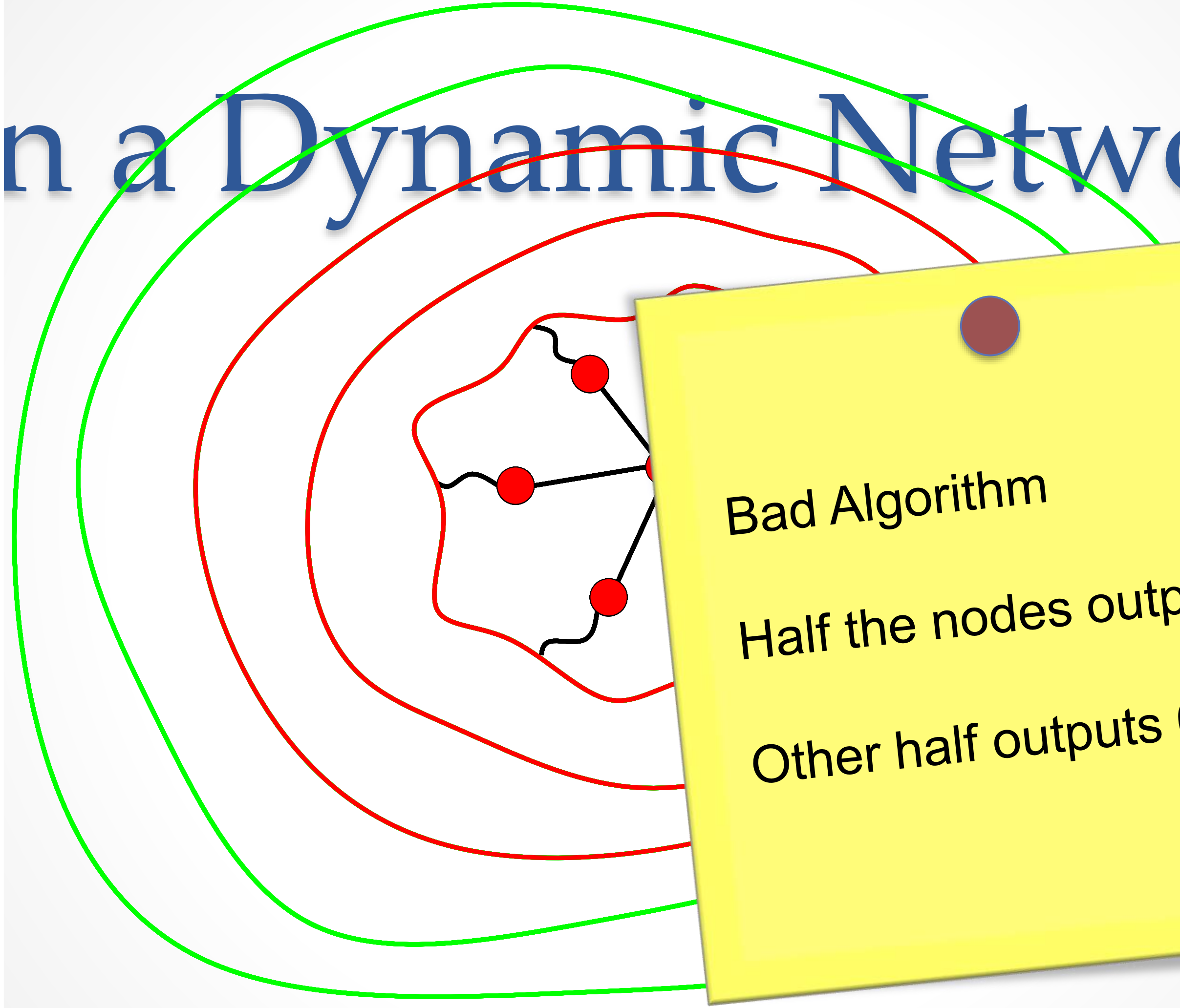
Not Exploited

Almost Everywhere Agreement

Every node (in round 1) starts with an input bit value.
Most nodes in the network must "agree"
on a valid input bit
within O(polylog n) rounds.


1

In a Dynamic Network



Bad Algorithm

Half the nodes output 1

Other half outputs 0

Information Spreading

The dynamic distance from a node $u \in V^r$ to a node v starting at round r is the number of rounds it takes for flooded messages from u to reach v starting at round r.

The influence set of u after R rounds starting at round r is the set of nodes in V^{r+R} whose dynamic distance from u starting at round r is at most R.

Note: The influence set is defined as a subset of V^{r+R}.

Information Spreading

We establish that A large fraction of the nodes can "influence" a (common) large fraction of nodes in $O(\log n)$ rounds.

1. Any reasonably sized fraction of the nodes (i.e., $\geq \beta n$ nodes) influence a large fraction of the nodes (i.e., $(1-\beta)n$ nodes) in constant rounds.

2. Given any reasonably sized fraction of the nodes U (i.e., $\geq \beta n$ nodes), there is a node $u \in U$ that influences $(1-\beta)n$ nodes in $O(\log n)$ rounds.

3. There is a large fraction of the nodes (i.e., $(1-\beta)n$ nodes) that influence a common large fraction of the nodes (i.e., $(1-\beta)n$ nodes) in $O(\log n)$ rounds.

Algorithmic Tools

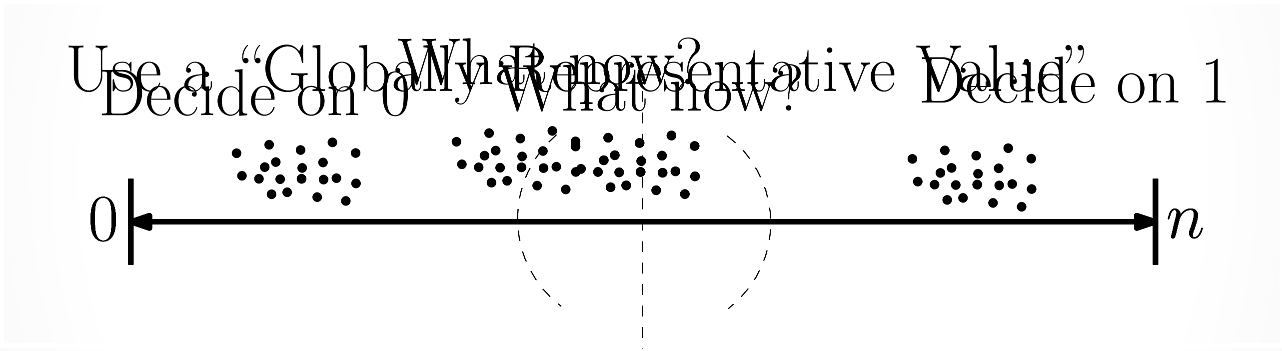
Globally Representative Value: A large fraction of the nodes (in unison) choose a value held by some node

- Why does this not suffice in the first place?

Support Estimation: Count the number of nodes currently proposing 1.

- Each node estimates
- Guarantee: large fraction of the nodes estimate within a small margin of error

Intuition Behind Solution



Storing and Retrieving

Towards a Dynamic Hash Table (DHT)

Towards a DHT

First steps towards solving a DHT (storage and retrieval of one item)

- $\longrightarrow$ Despite high levels of churn and edge dynamism
- $\longrightarrow$ Using scalable techniques
  (random walks — useful for sampling nodes, a fundamental primitive)
- $\longrightarrow$ With rigorous proof
- $\longrightarrow$ Against an oblivious adversary.

The Model

Dynamic Network Model

[—, Molla, Morsy, Pandurangan, Robinson, and Upfal; *SPAA* 2013]

- Dynamic network is modelled as a graph process: $G_1, G_2, \ldots$, where $G_i = (V_i, E_i)$ is the graph in round i.
- An adversary controls the topology of every graph G_i, including the set of nodes, with the restriction that each G_i is an expander graph.
- The number of nodes that change in a round is called the churn rate.
- The churn rate can be very high: up to $O(n/\text{polylog}(n))$ per round.
- Network size remains (essentially) constant.

Communication Model

- Synchronous model: computation/communication proceeds in a sequence of rounds.
- Each edge can carry polylog(n) bits in each round.
- Direct Communication: If u knows v's ID (e.g., IP address) it can send a message.
- Overlay Communication when u and v are neighbours in the overlay network.
- Sparse network: Each node can only communicate with a constant number of nodes in any round.

Crucial

Problem Definition

- Given a data item (as a <key, value> pair)
- Store (in *O(log n) rounds*)
- Maintain the item in the network for *poly(n)* rounds.
- Overhead of *o(n)* stored bits. $\boxed{\longrightarrow \tilde{O}\sqrt{n} \text{ in our case!}}$
- Most (n-o(n)) nodes can Retrieve when required in *O(log n)* rounds.
- With high probability

Some First Attempts

- Store the item in an arbitrary node
- Store the item in a random node

Recall Random Walks

- Store the item redundantly in multiple random nodes.
- Store the item redundantly in multiple random nodes and move it around.

Generalizing…

- Consider any task that takes time
  - Can't entrust to single node.

- **Solution**:
  - **Create** a committee of $\Theta(\log n)$ random nodes.
  - **Entrust** task to committee.
  - **Re-elect** new random members every $\Theta(\log n)$ rounds.

- **Guarantee**: Task stays alive for $poly(n)$ rounds (whp).

Back to Storing an Item

- Just entrust the task to a committee
  - All members in the committee hold a copy.
- Problem: How to find the committee members?
  - Store pointers in roughly $\sqrt{n}$ random nodes.
- Retrieving
  - Again entrust task to committee
  - Committee searches $\sqrt{n}$ random nodes and find item (by birthday paradox).

Conclusion

Distributed Hash Table

Fundamental Problems (Agreement, LE, etc.)

Overlay Maintenance

Other churn environments

Thank you