Random Walks on Dynamic Graphs

John Augustine
IIT Madras

Joint works with
A. R. Molla, E. Morsy, G. Pandurangan, P. Robinson, and E. Upfal

June 28, 2016

Peer-To-Peer Networks — a backdrop.

Peer-to-Peer Networks

Prevailing Definition

A network of peer nodes

mostly decentralized, but *some* central control

Some real world examples

Skype, BitTorrent, Cloudmark, CrashPlan, Symform, etc . . .

Gnutella is somewhat decentralized,

but employs ultrapeers (target for malicious agents) and flooding

Our Aspiration

A scalable decentralized network.

must take a life of its own

but not at the cost of efficiency

Peer-to-Peer Networks

Prevailing Definition

A network of peer nodes

mostly decentralized, but *some* central control

Some real world examples

Skype, BitTorrent, Cloudmark, CrashPlan, Symform, etc ...

Gnutella is somewhat decentralized,

but employs ultrapeers (target for malicious agents) and flooding

Our Aspiration

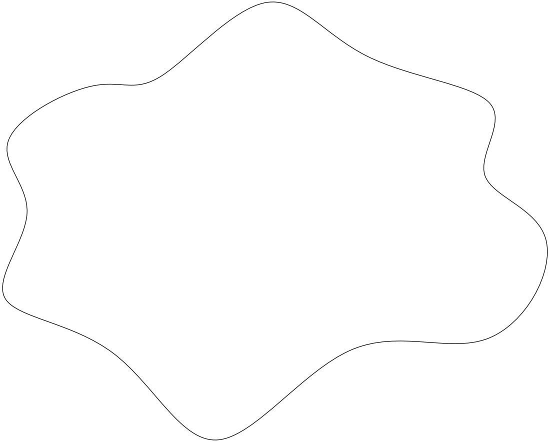
A scalable decentralized network.

must take a life of its own

but not at the cost of efficiency

Peer-to-Peer Networks

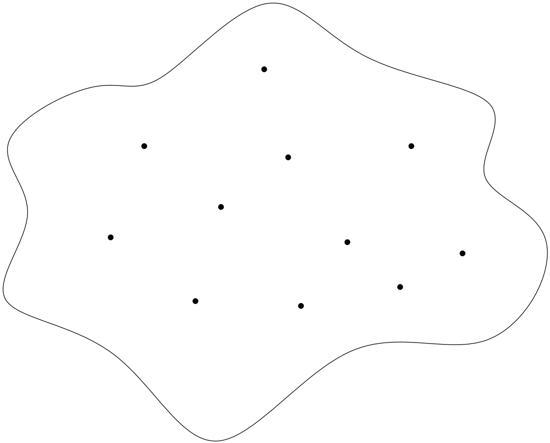
— architecture



Underlying Internet(Complete Connectivity)

Peer-to-Peer Networks

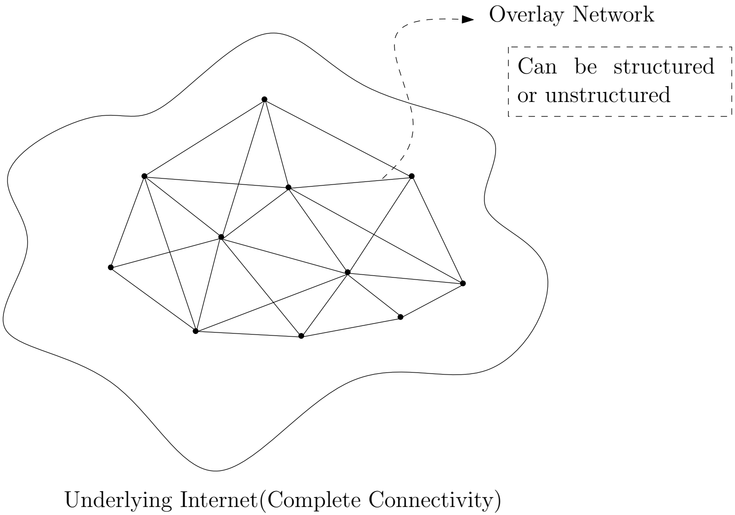
— architecture



Underlying Internet(Complete Connectivity)

Peer-to-Peer Networks

— architecture



Peer-to-Peer Networks

— Key challenge

- The network is highly dynamic

  → The network experiences heavy churn (up to 50% new nodes every hour)

  → Overlay edges are created and destroyed all the time

Lots of applications and papers, but need more rigorous guarantees.

Peer-to-Peer Networks

— Key challenge

- The network is highly dynamic
  - → The network experiences heavy churn (up to 50% new nodes every hour)
  - → Overlay edges are created and destroyed all the time

Lots of applications and papers, but need more rigorous guarantees.

Peer-to-Peer Networks

— our goal

First steps towards solving a fundamental problem (data storage and retrieval)

⟶ Despite high levels of dynamism (churn *and* edge dynamism)

⟶ Using scalable techniques
(random walks — useful for sampling nodes, a fundamental primitive)

⟶ With rigorous proof

⟶ Against and oblivious adversary.

Peer-to-Peer Networks

— our goal

First steps towards solving a fundamental problem (data storage and retrieval)

$\longrightarrow$ Despite high levels of dynamism (churn *and* edge dynamism)

$\longrightarrow$ Using scalable techniques
(random walks — useful for sampling nodes, a fundamental primitive)

$\longrightarrow$ With rigorous proof

$\longrightarrow$ Against and oblivious adversary.

Peer-to-Peer Networks

— our goal

First steps towards solving a fundamental problem (data storage and retrieval)

⟶ Despite high levels of dynamism (churn *and* edge dynamism)

⟶ Using scalable techniques
(random walks — useful for sampling nodes, a fundamental primitive)

⟶ With rigorous proof

⟶ Against and oblivious adversary.

Peer-to-Peer Networks

— our goal

First steps towards solving a fundamental problem (data storage and retrieval)

$\longrightarrow$ Despite high levels of dynamism (churn *and* edge dynamism)

$\longrightarrow$ Using scalable techniques
(random walks — useful for sampling nodes, a fundamental primitive)

$\longrightarrow$ With rigorous proof

$\longrightarrow$ Against and oblivious adversary.

Peer-to-Peer Networks

— our goal

First steps towards solving a fundamental problem (data storage and retrieval)

⟶ Despite high levels of dynamism (churn *and* edge dynamism)

⟶ Using scalable techniques
(random walks — useful for sampling nodes, a fundamental primitive)

⟶ With rigorous proof

⟶ Against and oblivious adversary.

Dynamic Networks.

Dynamic Networks

Edge Dynamic Networks.

Avin, Koucky, and Lotker (2008)
Kuhn, Lynch, Oshman (2010)

- Nodes fixed
- Edges changed arbitrarily by an adversary
- Various assumptions on connectivity

Edge Dynamic Networks with Churn.

with Pandurangan, Robinson, and Upfal (2012)

- Nodes can be churned in/out.
- Network changed arbitrarily by an adversary
- Stable network size.
- Stronger assumption on connectivity

Distributed Networks With Churn (DNC)

Our Model

— the setting

Synchronous. All nodes follow the same clock. In each round $r = 1, 2, \ldots$

- Each node sends messages to its neighbours
- Each node receives messages from its neighbours
- Nodes perform local computations

Adversarial Dynamism. An oblivious adversary (knows algorithm, but not the coin toss outcomes) designs churn and edge dynamics

$$\mathcal{G} = (G^0, G^1, \ldots, G^r, \ldots)$$

Our Model

— the setting

Synchronous. All nodes follow the same clock. In each round $r = 1, 2, \ldots$

- Each node sends messages to its neighbours
- Each node receives messages from its neighbours
- Nodes perform local computations

Adversarial Dynamism. An oblivious adversary (knows algorithm, but not the coin toss outcomes) designs churn and edge dynamics

$$\mathcal{G} = (G^0, G^1, \ldots, G^r, \ldots)$$

Our Model

— the setting

Unique ID and single lifetime. Each node comes in once and leaves at most once.

Churn. Up to $n/\text{polylog } n$ nodes leave/join the network per round.

Stable Network Size. Number of nodes n unchanged over time (In each round: $\#$ churned out $= \#$ churned in)

Edge Dynamics. Topology can change round to round arbitrarily

Our Model

— the setting

Unique ID and single lifetime. Each node comes in once and leaves at most once.

Churn. Up to n/polylog n nodes leave/join the network per round.

Stable Network Size. Number of nodes n unchanged over time (In each round: # churned out = # churned in)

Edge Dynamics. Topology can change round to round arbitrarily

Our Model

— high connectivity assumption

Each $G^r = (V^r, E^r)$ is a d-regular α-expander graph.

A graph $G = (V, E)$ is an α-expander if

For every $S \subset V$ such that $|S| \leq |V|/2$,

$$|\Gamma(S)| \geq \alpha \cdot |S|,$$

where $\Gamma(S) = \{u \in V \setminus S \mid \exists s \in S \text{ such that } (s, u) \in E\}$.

- Common assumption in fault tolerant peer-to-peer networks
- Prior dynamic network models also make such assumptions
- New techniques to maintain expansion (FOCS 2015, with Pandurangan, Robinson, Roche, & Upfal)

Our Model

— high connectivity assumption

Each $G^r = (V^r, E^r)$ is a d-regular α-expander graph.

A graph $G = (V, E)$ is an α-expander if

For every $S \subset V$ such that $|S| \leq |V|/2$,

$$|\Gamma(S)| \geq \alpha \cdot |S|,$$

where $\Gamma(S) = \{u \in V \setminus S \mid \exists s \in S \text{ such that } (s, u) \in E\}$.

- Common assumption in fault tolerant peer-to-peer networks
- Prior dynamic network models also make such assumptions
- New techniques to maintain expansion (FOCS 2015, with Pandurangan, Robinson, Roche, & Upfal)

Our Model

— high connectivity assumption

Each $G^r = (V^r, E^r)$ is a d-regular α-expander graph.

A graph $G = (V, E)$ is an α-expander if

For every $S \subset V$ such that $|S| \leq |V|/2$,

$$|\Gamma(S)| \geq \alpha \cdot |S|,$$

where $\Gamma(S) = \{u \in V \setminus S \mid \exists s \in S \text{ such that } (s, u) \in E\}$.

- Common assumption in fault tolerant peer-to-peer networks
- Prior dynamic network models also make such assumptions
- New techniques to maintain expansion (FOCS 2015, with Pandurangan, Robinson, Roche, & Upfal)

Our Model

— the setting

Message Passing Communication via messages through edges.

CONGEST At most $O(\text{polylog}(n))$ bits per round per recipient.

Direct Communication in P2P When recipient address is known and not churned out (not guaranteed)

Our Model

— the setting

Message Passing Communication via messages through edges.

CONGEST At most $O(\text{polylog}(n))$ bits per round per recipient.

Direct Communication in P2P When recipient address is known and not churned out (not guaranteed)

Our Model

— the setting

Message Passing Communication via messages through edges.

CONGEST At most $O(\text{polylog}(n))$ bits per round per recipient.

Direct Communication in P2P When recipient address is known and not churned out (not guaranteed)

Our Results

Consider a single data item (think $\langle \text{key}, \text{value} \rangle$ pair) generated by some node

Storage & Maintenance

- *Store* $O(\log n)$ copies
- *Maintain* the data in the network for polynomial in n number of rounds

Search

- *Construct* a search facilitating structure of size $\tilde{O}(\sqrt{n})$
- *Most* nodes (at least $n - o(n)$) can
- *Search* for a particular key in $O(\log n)$ time

(Results hold with high probability)

A Long Term Task in a Short Lived Life

— the committee

Issue: Suppose a node is entrusted with a task.
The node may be churned out before task completed.

Solution: Form a committee of $\Theta(\log n)$ *random* nodes.

- Small enough, low communication cost
- Easy to behave in unison
- Hard for oblivious adversary to disrupt

A Long Term Task in a Short Lived Life

— the committee

Issue: Suppose a node is entrusted with a task.
The node may be churned out before task completed.

Solution: Form a committee of $\Theta(\log n)$ *random* nodes.

- Small enough, low communication cost
- Easy to behave in unison
- Hard for oblivious adversary to disrupt

Creating a Committee

Context. A node u has some "task" to perform.

1: Node u sends an invitation (along with list of invitees) to $\Theta(\log n)$ random nodes.

2: Nodes that receive the invitation (if alive) connect with the other invitees and form a committee.

Committee Maintenance

— the algorithm

Why maintain? With $n/\text{polylog}(n)$ churn, the committee will be decimated in $O(\text{polylog}(n))$ round.

Every $\Theta(\log n)$ **rounds**

1: The committee elects a leader ℓ

2: The node ℓ elects a new committee

3: The task is handed over to the new committee

4: The old committee disbands itself after the new committee can fully take over.

Committee Maintenance

— the analysis

A Good Committee

Good if at least $\Omega(\log n)$ nodes.

Theorem

With high probability, a committee will be good for a number of rounds that is polynomial in n.

Committee Landmarks

— useful in storing and maintaining data

{Everytime a new committee is (re)formed, the following is executed.}

Each node in the new committee spawns a pair of landmarks by selecting two samples and passes all committee ids and initiates level $\leftarrow 0$.

Each new landmark spawns another pair in turn and sends committee ids and incremented level. Repeat until $\tilde{O}(\sqrt{n})$ landmarks are created.

Storing and Maintaining Data

To store an identifiable data item,

1. node u creates a committee and the member store the data
2. Landmarks are created
3. Data is passed on to new committee members chosen for maintenance.

Committee Landmarks

— useful in storing and maintaining data

{Everytime a new committee is (re)formed, the following is executed.}

Each node in the new committee spawns a pair of landmarks by selecting two samples and passes all committee ids and initiates level $\leftarrow 0$.

Each new landmark spawns another pair in turn and sends committee ids and incremented level. Repeat until $\tilde{O}(\sqrt{n})$ landmarks are created.

Storing and Maintaining Data

To store an identifiable data item,

1. node u creates a committee and the member store the data
2. Landmarks are created
3. Data is passed on to new committee members chosen for maintenance.

Searching For Identifiable Data

— using the birthday paradox

1. Create a committee and entrust with task of finding data
2. The committee creates landmarks (similar to before)
3. When these landmarks collide with landmarks of searched data, the data can be retrieved.

Theorem

At any round r,

- *There are $n - o(n)$ nodes that can store a data item,*
- *The data item can be maintained for a polynomial in n number of rounds, and*
- *The data item can be searched by $n - o(n)$ nodes in $\tilde{O}(\log n)$ rounds.*

Searching For Identifiable Data

— using the birthday paradox

1. Create a committee and entrust with task of finding data
2. The committee creates landmarks (similar to before)
3. When these landmarks collide with landmarks of searched data, the data can be retrieved.

Theorem

At any round r,

- *There are $n - o(n)$ nodes that can store a data item,*
- *The data item can be maintained for a polynomial in n number of rounds, and*
- *The data item can be searched by $n - o(n)$ nodes in $\tilde{O}(\log n)$ rounds.*

Technical Contribution

Flooding is a heavy weight operation.

Random walks are

1. Scalable
2. Useful in sampling
3. Edge dynamism $\surd$ (good expansion needed)
4. Churn $\surd$ (but only up to $n/\text{polylog}(n)$ nodes per round)

Technical Contribution

Flooding is a heavy weight operation.

Random walks are

1. Scalable
2. Useful in sampling
3. Edge dynamism $\surd$ (good expansion needed)
4. Churn $\surd$ (but only up to $n/\text{polylog}(n)$ nodes per round)

Technical Contribution

Flooding is a heavy weight operation.

Random walks are

1. Scalable
2. Useful in sampling
3. Edge dynamism $\surd$ (good expansion needed)
4. Churn $\surd$ (but only up to $n/\text{polylog}(n)$ nodes per round)

How to find Random Nodes?

for round $r = 1, 2, \ldots$ at every node v **do**
    Initiate $\Theta(\log n)$ random walk tokens
    $\longrightarrow$ with node v's id and
    $\longrightarrow$ a timer set for $\Theta(\log n)$ rounds
    Forward every unexpired random walk tokens
    Consume expired random walks (right away) as node samples
end for

Theorem (paraphrased)

At every time step (after initial bootstrap), most nodes will get $\Theta(\log n)$ node samples every round chosen uniformly at random from most nodes.

How to find Random Nodes?

1: **for** round $r = 1, 2, \ldots$ at every node v **do**

2: **Initiate** $\Theta(\log n)$ random walk tokens

$\longrightarrow$ with node v's id and

$\longrightarrow$ a timer set for $\Theta(\log n)$ rounds

3: **Forward** every unexpired random walk tokens

4: **Consume** expired random walks (right away) as node samples

5: **end for**

Theorem (paraphrased)

At every time step (after initial bootstrap), most nodes will get $\Theta(\log n)$ node samples every round chosen uniformly at random from most nodes.

How to find Random Nodes?

```
for round r = 1, 2, ... at every node v do
    Initiate Θ(log n) random walk tokens
      ⟶ with node v's id and
      ⟶ a timer set for Θ(log n) rounds
    Forward every unexpired random walk tokens
    Consume expired random walks (right away) as node samples
end for
```

Theorem (paraphrased)

At every time step (after initial bootstrap), most nodes will get $\Theta(\log n)$ node samples every round chosen uniformly at random from most nodes.

How to find Random Nodes?

1: **for** round $r = 1, 2, \ldots$ at every node v **do**

2: **Initiate** $\Theta(\log n)$ random walk tokens

$\longrightarrow$ with node v's id and

$\longrightarrow$ a timer set for $\Theta(\log n)$ rounds

3: **Forward** every unexpired random walk tokens

4: **Consume** expired random walks (right away) as node samples

5: **end for**

Theorem (paraphrased)

At every time step (after initial bootstrap), most nodes will get $\Theta(\log n)$ node samples every round chosen uniformly at random from most nodes.

How to find Random Nodes?

for round $r = 1, 2, \ldots$ at every node v **do**

  Initiate $\Theta(\log n)$ random walk tokens

  $\longrightarrow$ with node v's id and

  $\longrightarrow$ a timer set for $\Theta(\log n)$ rounds

  Forward every unexpired random walk tokens

  Consume expired random walks (right away) as node samples

end for

Theorem (paraphrased)

At every time step (after initial bootstrap), most nodes will get $\Theta(\log n)$ node samples every round chosen uniformly at random from most nodes.

How to find Random Nodes?

1: **for** round $r = 1, 2, \ldots$ at every node v **do**
2: **Initiate** $\Theta(\log n)$ random walk tokens
$\longrightarrow$ with node v's id and
$\longrightarrow$ a timer set for $\Theta(\log n)$ rounds
3: **Forward** every unexpired random walk tokens
4: **Consume** expired random walks (right away) as node samples
5: **end for**

Theorem (paraphrased)

At every time step (after initial bootstrap), most nodes will get $\Theta(\log n)$ node samples every round chosen uniformly at random from most nodes.

Random Walks Theory on Static Graphs

— the basics

Consider a d-regular graph $G = (V, E)$ and
let a token start at $x \in V$ and make a random walk
choosing neighbours uniformly and independently at random.

$\pi_x(t) \triangleq$ probability distribution vector after t steps. ($\lim_{t\to\infty} \pi_x(t) = \frac{1}{n}\mathbf{1}$.)

$\tau_\epsilon^x \triangleq \min\{t : ||\pi_x(t) - \frac{1}{n}\mathbf{1}||_\infty \leq \epsilon\}$

The mixing time $\tau \triangleq \max_x \tau_\epsilon^x$, where $\epsilon \in O(1/n)$.

Well known that for expander graphs (with constant expansion)

$$\tau \in O(\log n)$$

Random Walks Mix Well in Edge Dynamic Graphs

— Avin *et al.* (ICALP 2008) and Das Sarma *et al.* (DISC 2012).

Random walks work similarly in edge dynamic graphs.

Theorem

For any d-regular connected non-bipartite edge dynamic graph $\mathcal{G}$, the dynamic mixing time of a simple random walk on $\mathcal{G}$ is bounded by $O(\log n)$.

Alternatively, a random walk that starts at s Will be in any d with probability in $\Theta(1/n)$ (say $[1/2n, 3/2n]$) after $\Theta(\log n)$ rounds.

Random Walks Mix Well in Edge Dynamic Graphs

— Avin *et al.* (ICALP 2008) and Das Sarma *et al.* (DISC 2012).

Random walks work similarly in edge dynamic graphs.

Theorem

For any d-regular connected non-bipartite edge dynamic graph $\mathcal{G}$, the dynamic mixing time of a simple random walk on $\mathcal{G}$ is bounded by $O(\log n)$.

Alternatively, a random walk that starts at s Will be in any d with probability in $\Theta(1/n)$ (say $[1/2n, 3/2n]$) after $\Theta(\log n)$ rounds.

What about churn?

— Our Concern

$$\mathcal{G} = (G^0, G^1, \ldots, G^r, \ldots)$$

Now with churn at most $O\left(\frac{n}{\log^{1+\delta} n}\right)$, where $\delta > 0$

- Random token may not survive
- Even if it survives
  - $\rightarrow$ Does the notion of mixing time make sense?
  - $\rightarrow$ Can it become biased?

We show that random walks starting from most nodes "mix"

What about churn?

— Our Concern

$$\mathcal{G} = (G^0, G^1, \ldots, G^r, \ldots)$$

Now with churn at most $O\left(\frac{n}{\log^{1+\delta} n}\right)$, where $\delta > 0$

- Random token may not survive
- Even if it survives
  - $\rightarrow$ Does the notion of mixing time make sense?
  - $\rightarrow$ Can it become biased?

We show that random walks starting from most nodes "mix"

The Dynamic Sampling Theorem

$\pi(\mathcal{G}, s, d, t) \triangleq \Pr[$a r.w starting at $s \in V^0$ ends at d in G^t in round $t]$.

Recall $\tau \in O(\log n)$ is mixing time of expander graphs

Theorem

Suppose churn is at most $O\left(n/\log^{1+\delta} n\right)$ for any fixed $\delta > 0$.

Then, $\exists$ a set $\textsc{Core} \subset V^0 \cap V^{2\tau}$ of cardinality $n - o(n)$

such that

For any $s \in \textsc{Core}$ and $d \in \textsc{Core}$,

$$\pi(\mathcal{G}, s, d, 2\tau) \in \Theta(1/n).$$

Skip Past Proof

The Dynamic Sampling Theorem

$\pi(\mathcal{G}, s, d, t) \triangleq \Pr[$a r.w starting at $s \in V^0$ ends at d in G^t in round $t]$.

Recall $\tau \in O(\log n)$ is mixing time of expander graphs

Theorem

Suppose churn is at most $O\left(n/\log^{1+\delta} n\right)$ for any fixed $\delta > 0$.

Then, $\exists$ a set $\textsc{Core} \subset V^0 \cap V^{2\tau}$ of cardinality $n - o(n)$

such that

For any $s \in \textsc{Core}$ and $d \in \textsc{Core}$,

$$\pi(\mathcal{G}, s, d, 2\tau) \in \Theta(1/n).$$

Skip Past Proof

The Dynamic Sampling Theorem

$\pi(\mathcal{G}, s, d, t) \triangleq \Pr[\text{a r.w starting at } s \in V^0 \text{ ends at } d \text{ in } G^t \text{ in round } t]$.

Recall $\tau \in O(\log n)$ is mixing time of expander graphs

Theorem

Suppose churn is at most $O\left(n/\log^{1+\delta} n\right)$ for any fixed $\delta > 0$.

Then, $\exists$ a set $\textsc{Core} \subset V^0 \cap V^{2\tau}$ of cardinality $n - o(n)$

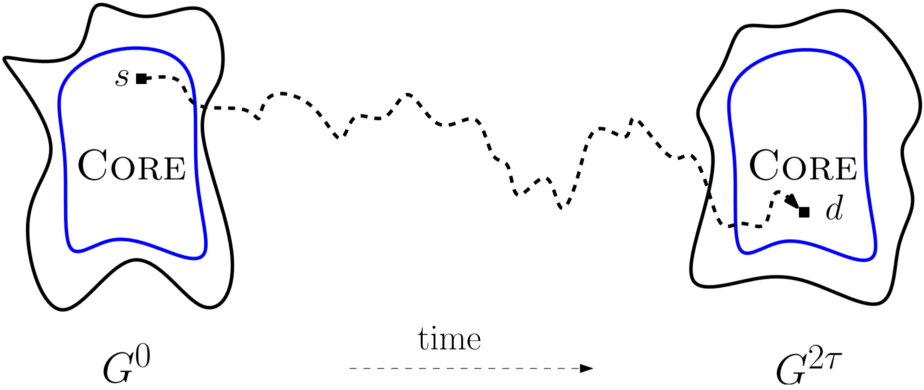
such that

For any $s \in \textsc{Core}$ and $d \in \textsc{Core}$,

$$\pi(\mathcal{G}, s, d, 2\tau) \in \Theta(1/n).$$

Skip Past Proof

The Dynamic Sampling Theorem
— illustrated



Skip Past Proof

Proof of the Dynamic Sampling Theorem

A random walks preserving graph $\bar{\mathcal{G}} = (\bar{G}^0, \bar{G}^1, \ldots)$

Each $\bar{G}^r = G^r$ except that

State of each node churned out is copied onto a unique node that is churned in.

The upshot

Random walks do not die in $\bar{\mathcal{G}}$.

From Das Sarma *et al.*, mixing time of $\bar{\mathcal{G}}$ is $O(\log n)$.

How does it help?

We can simulate the random walks in $\bar{\mathcal{G}}$ and adjust the results to fit $\mathcal{G}$.

Proof of the Dynamic Sampling Theorem

A random walks preserving graph $\bar{\mathcal{G}} = (\bar{G}^0, \bar{G}^1, \ldots)$

Each $\bar{G}^r = G^r$ except that

State of each node churned out is copied onto a

unique node that is churned in.

The upshot

Random walks do not die in $\bar{\mathcal{G}}$.

From Das Sarma *et al.*, mixing time of $\bar{\mathcal{G}}$ is $O(\log n)$.

How does it help?

We can simulate the random walks in $\bar{\mathcal{G}}$ and adjust the results to fit $\mathcal{G}$.

Proof of the Dynamic Sampling Theorem

A random walks preserving graph $\bar{\mathcal{G}} = (\bar{G}^0, \bar{G}^1, \ldots)$

Each $\bar{G}^r = G^r$ except that

State of each node churned out is copied onto a unique node that is churned in.

The upshot

Random walks do not die in $\bar{\mathcal{G}}$.

From Das Sarma *et al.*, mixing time of $\bar{\mathcal{G}}$ is $O(\log n)$.

How does it help?

We can simulate the random walks in $\bar{\mathcal{G}}$ and adjust the results to fit $\mathcal{G}$.

Proof of the Dynamic Sampling Theorem

A random walks preserving graph $\bar{\mathcal{G}} = (\bar{G}^0, \bar{G}^1, \ldots)$

Each $\bar{G}^r = G^r$ except that

State of each node churned out is copied onto a unique node that is churned in.

The upshot

Random walks do not die in $\bar{\mathcal{G}}$.

From Das Sarma *et al.*, mixing time of $\bar{\mathcal{G}}$ is $O(\log n)$.

How does it help?

We can simulate the random walks in $\bar{\mathcal{G}}$ and adjust the results to fit $\mathcal{G}$.

Proof of the Dynamic Sampling Theorem

$\pi_*(\mathcal{G}, s, \tau) \triangleq$ Prob. that r.w from s in G^0 is churned out before round τ

Lemma

Let churn be at most $O\left(n/\log^{1+\delta} n\right)$ *and*

$$S \triangleq \left\{ s \in V^0 \mid \pi_*(\mathcal{G}, s, \tau) \leq \frac{1}{\log^{\delta/2} n} \right\}.$$

Then,

$$|S| \geq n - o(n).$$

Proof.

Start one random walk from each node and let them walk for τ rounds.

$$\frac{1}{\log^{\delta/2} n} |V^0 \setminus S| \leq \text{Exp. } \# \text{ that die} \leq O\left(\frac{n}{\log^{\delta} n}\right)$$

□

Proof of the Dynamic Sampling Theorem

— a crucial lemma

Recall: $S \triangleq \left\{ s \in V^0 \mid \pi_*(\mathcal{G}, s, \tau) \leq \frac{1}{\log^{\delta/2} n} \right\}$ and we know $|S| \geq n - o(n)$.

Lemma (crucial)

For every $s \in S$, $\exists\ D(s) \subset V^\tau$ with $|D(s)| \geq n - o(n)$

such that

$\forall d \in D(s)$, $\pi(\mathcal{G}, s, d, \tau) \in \Theta(1/n)$.

Proof of the Dynamic Sampling Theorem

— proof of the crucial lemma

Easy part: $\pi(\mathcal{G}, s, d, \tau) \leq \pi(\bar{\mathcal{G}}, s, d, \tau) \leq 3/2n$ for <u>any</u> $d \in V^\tau$.

Hard part: S. T. $|D(s)| \geq n - o(n)$, where

$$D(s) \triangleq \{d \in V^\tau : \pi(\mathcal{G}, s, d, \tau) \geq 1/4n\}.$$

Equivalently S.T. $\hat{D} \triangleq V^\tau \setminus D(s)$ has cardinality at most $o(n)$.

$$\sum_{d \in V^\tau} (\pi(\bar{\mathcal{G}}, s, d, \tau) - \pi(\mathcal{G}, s, d, \tau)) = \pi_*(\mathcal{G}, s, \tau) \leq 1/\log^{\delta/2} n$$

$$\sum_{d \in \hat{D}} (\pi(\bar{\mathcal{G}}, s, d, \tau) - \pi(\mathcal{G}, s, d, \tau)) \leq 1/\log^{\delta/2} n$$

$$\sum_{d \in \hat{D}} (1/2n - 1/4n) = |\hat{D}|(1/4n) \leq 1/\log^{\delta/2} n$$

Thus, $|\hat{D}| \leq \frac{4n}{\log^{\delta/2} n} \in o(n)$.

Proof of the Dynamic Sampling Theorem

— proof of the crucial lemma

Easy part: $\pi(\mathcal{G}, s, d, \tau) \le \pi(\bar{\mathcal{G}}, s, d, \tau) \le 3/2n$ for <u>any</u> $d \in V^\tau$.

Hard part: S. T. $|D(s)| \ge n - o(n)$, where

$$D(s) \triangleq \{d \in V^\tau : \pi(\mathcal{G}, s, d, \tau) \ge 1/4n\}.$$

Equivalently S.T. $\hat{D} \triangleq V^\tau \setminus D(s)$ has cardinality at most $o(n)$.

$$\sum_{d \in V^\tau} (\pi(\bar{\mathcal{G}}, s, d, \tau) - \pi(\mathcal{G}, s, d, \tau)) = \pi_*(\mathcal{G}, s, \tau) \le 1/\log^{\delta/2} n$$

$$\sum_{d \in \hat{D}} (\pi(\bar{\mathcal{G}}, s, d, \tau) - \pi(\mathcal{G}, s, d, \tau)) \le 1/\log^{\delta/2} n$$

$$\sum_{d \in \hat{D}} (1/2n - 1/4n) = |\hat{D}|(1/4n) \le 1/\log^{\delta/2} n$$

Thus, $|\hat{D}| \le \frac{4n}{\log^{\delta/2} n} \in o(n)$.

Proof of the Dynamic Sampling Theorem

— proof of the crucial lemma

Easy part: $\pi(\mathcal{G}, s, d, \tau) \leq \pi(\bar{\mathcal{G}}, s, d, \tau) \leq 3/2n$ for <u>any</u> $d \in V^\tau$.

Hard part: S. T. $|D(s)| \geq n - o(n)$, where

$$D(s) \triangleq \{d \in V^\tau : \pi(\mathcal{G}, s, d, \tau) \geq 1/4n\}.$$

Equivalently S.T. $\hat{D} \triangleq V^\tau \setminus D(s)$ has cardinality at most $o(n)$.

$$\sum_{d \in V^\tau} (\pi(\bar{\mathcal{G}}, s, d, \tau) - \pi(\mathcal{G}, s, d, \tau)) = \pi_*(\mathcal{G}, s, \tau) \leq 1/\log^{\delta/2} n$$

$$\sum_{d \in \hat{D}} (\pi(\bar{\mathcal{G}}, s, d, \tau) - \pi(\mathcal{G}, s, d, \tau)) \leq 1/\log^{\delta/2} n$$

$$\sum_{d \in \hat{D}} (1/2n - 1/4n) = |\hat{D}|(1/4n) \leq 1/\log^{\delta/2} n$$

Thus, $|\hat{D}| \leq \frac{4n}{\log^{\delta/2} n} \in o(n)$. □

Proof of the Dynamic Sampling Theorem

— reversibility of random walks

Lemma (reverse of crucial)

With high probability,

$\exists D \subseteq V^\tau$ *of cardinality at least* $n - o(n)$ *such that,*

for any $d \in D$, $\exists\ S(d) \subseteq V^0$ *of cardinality at least* $n - o(n)$ *such that,*

a random walk that terminated in d

originated in a fixed $s \in S(d)$ *with probability in* $[1/4n, 3/2n]$.

Combining the crucial lemma and its reverse, we can prove the dynamic sampling theorem. □

Recall the Dynamic Sampling Theorem

Related Works

- Agreement despite Byzantine nodes (PODC 2013 with Pandurangan and Robinson)
- Leader election despite Byzantine nodes (DISC 2015 with Pandurangan and Robinson)
- Maintaining expansion (FOCS 2015 with Pandurangan, Robinson, Roche, and Upfal)

Open Problems

- A robust framework/family of random evolving graph processes

  Some graph processes known to me:

  - Nodes making a random walk on an underlying structure (say, 2D grid) and getting connected when on the same grid point.
  - Start with a positively weighted complete graph. Each time step, pick a random edge and randomly increment or decrement weight.

  Some graph processes that seem natural

  - A generalization of $G(n, m)$ in which, at each time step, an existing edge is removed and a new edge is added — both chosen UAR.
  - Fixed infrastructure graph G. The graph process connects vertices that are not too far from each other on G. Inspiration is Prof. Parongama Sen's talk.
  - Randomly connect to neighbour's neighbour (with random edge deletions).

Thank you