Random Graphs and their Use in Peer-to-Peer Networks

Kathlén Kohn

Faculty of Computer Science, Electrical Engineering and Mathematics
University of Paderborn

November 28, 2013

Table of Contents

Graph Transformations

Definitions

Graph Transformations

Definitions

Definition (Simple Digraph)

A simple digraph $G = (V, E)$ is defined by a node set $V = \{v_1, \ldots, v_n\}$ and a set of directed edges $E \subseteq \{(u, v) \mid u, v \in V, u \neq v\}$.

Graph Transformations

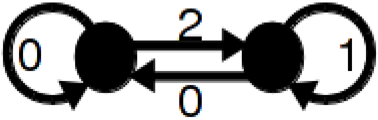
Definitions

Definition (Simple Digraph)

A simple digraph $G = (V, E)$ is defined by a node set $V = \{v_1, \ldots, v_n\}$ and a set of directed edges $E \subseteq \{(u, v) \mid u, v \in V, u \neq v\}$.

Definition (Multi-Digraph)

A multi-digraph $G = (V, E, \#)$ is defined by a node set $V = \{v_1, \ldots, v_n\}$ and a set of directed edges $E = \{(u, v) \mid u, v \in V\}$ with multiplicities given by $\# : E \rightarrow \mathbb{N}_0$.



Graph Transformations

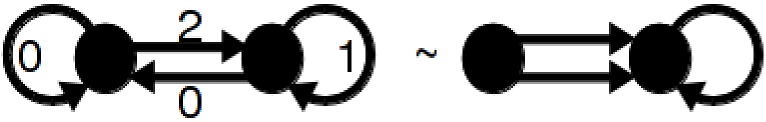
Definitions

Definition (Simple Digraph)

A simple digraph $G = (V, E)$ is defined by a node set $V = \{v_1, \dots, v_n\}$ and a set of directed edges $E \subseteq \{(u, v) \mid u, v \in V, u \neq v\}$.

Definition (Multi-Digraph)

A multi-digraph $G = (V, E, \#)$ is defined by a node set $V = \{v_1, \dots, v_n\}$ and a set of directed edges $E = \{(u, v) \mid u, v \in V\}$ with multiplicities given by $\# : E \to \mathbb{N}_0$.



Graph Transformations

Definitions

Definition (Graph Transformation)

Let $\mathcal{G} \subseteq \{G \mid G \text{ is a multi-digraph with } n \text{ nodes}\}$. A graph transformation is a random transition $\tau : \mathcal{G} \rightsquigarrow \mathcal{G}$ such that

$$\forall G \in \mathcal{G} : \sum_{G' \in \mathcal{G}} \Pr\left(\tau(G) = G'\right) = 1.$$

Graph Transformations

Definitions

Definition (Graph Transformation)

Let $\mathcal{G} \subseteq \{G \mid G \text{ is a multi-digraph with } n \text{ nodes}\}$. A graph transformation is a random transition $\tau : \mathcal{G} \rightsquigarrow \mathcal{G}$ such that

$$\forall G \in \mathcal{G} : \sum_{G' \in \mathcal{G}} \Pr\left(\tau(G) = G'\right) = 1.$$

If $|\mathcal{G}| < \infty$, τ defines a Markov chain, where the set of states is $\mathcal{G}$ and the transition matrix is $T \in \mathbb{R}^{|\mathcal{G}| \times |\mathcal{G}|}$ with $t_{G,G'} = \Pr(\tau(G) = G')$.

Graph Transformations

Requirements

Requirements for graph transformations used in peer-to-peer networks:

Graph Transformations

Requirements

Requirements for graph transformations used in peer-to-peer networks:

- Soundness: $\forall G \in \mathcal{G} : \tau(G) \in \mathcal{G}$

Graph Transformations

Requirements

Requirements for graph transformations used in peer-to-peer networks:

- Soundness: $\forall G \in \mathcal{G} : \tau(G) \in \mathcal{G}$
- Generality: $\forall G, G' \in \mathcal{G} : \lim_{k \to \infty} \Pr\left(\tau^k(G) = G'\right) > 0$

Graph Transformations

Requirements

Requirements for graph transformations used in peer-to-peer networks:

- Soundness: $\forall G \in \mathcal{G} : \tau(G) \in \mathcal{G}$
- Generality: $\forall G, G' \in \mathcal{G} : \lim_{k \to \infty} \Pr\left(\tau^k(G) = G'\right) > 0$
  - Uniform generality: $\forall G, G' \in \mathcal{G} : \lim_{k \to \infty} \Pr\left(\tau^k(G) = G'\right) = \frac{1}{|\mathcal{G}|}$

Graph Transformations

Requirements

Requirements for graph transformations used in peer-to-peer networks:

- Soundness: $\forall G \in \mathcal{G} : \tau(G) \in \mathcal{G}$
- Generality: $\forall G, G' \in \mathcal{G} : \lim_{k \to \infty} \Pr\left(\tau^k(G) = G'\right) > 0$
  - Uniform generality: $\forall G, G' \in \mathcal{G} : \lim_{k \to \infty} \Pr\left(\tau^k(G) = G'\right) = \frac{1}{|\mathcal{G}|}$
- Feasibility: τ can be described by a simple routine with a straightforward implementation in a distributed maintained network.

Graph Transformations

Requirements

Requirements for graph transformations used in peer-to-peer networks:

- Soundness: $\forall G \in \mathcal{G} : \tau(G) \in \mathcal{G}$
- Generality: $\forall G, G' \in \mathcal{G} : \lim_{k\to\infty} \Pr\left(\tau^k(G) = G'\right) > 0$
  - Uniform generality: $\forall G, G' \in \mathcal{G} : \lim_{k\to\infty} \Pr\left(\tau^k(G) = G'\right) = \frac{1}{|\mathcal{G}|}$
- Feasibility: τ can be described by a simple routine with a straightforward implementation in a distributed maintained network.
- Convergence rate: After a small number of transitions a good approximation of the ultimate distribution on $\mathcal{G}$ is achieved.

Graph Transformations

First Examples

Graph Transformations

First Examples

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}.$

$G = (V, E, \#)$ is d-out-regular $\Leftrightarrow \forall u \in V : \sum_{v \in V} \#((u, v)) = d$

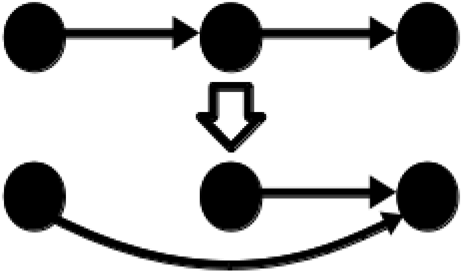
Graph Transformations

First Examples

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

$G = (V, E, \#)$ is d-out-regular $\Leftrightarrow \forall u \in V : \sum_{v \in V} \#((u, v)) = d$

Pointer-Push:



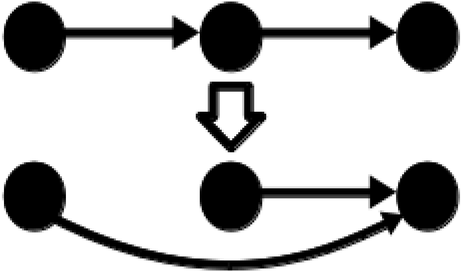
Graph Transformations

First Examples

Let $\mathcal{G}_u := \left\{ G \,\middle|\, \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

$G = (V, E, \#)$ is d-out-regular $\Leftrightarrow \forall u \in V : \sum_{v \in V} \#((u, v)) = d$

Pointer-Push:



- is sound
- is feasible
- is not general

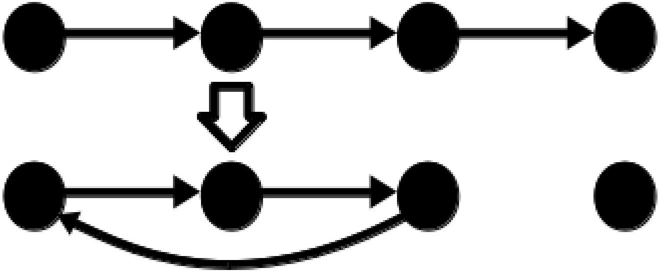
Graph Transformations

First Examples

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{c} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}.$

$G = (V, E, \#)$ is d-out-regular $\Leftrightarrow \forall u \in V : \sum_{v \in V} \#((u,v)) = d$

Pointer-Pull:



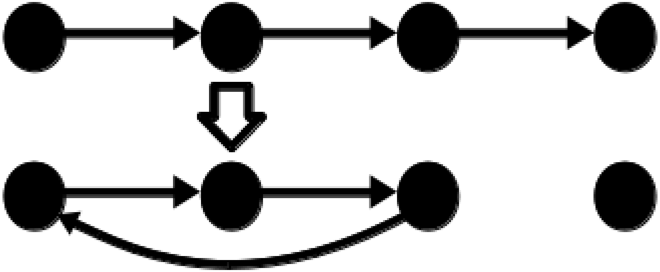
Graph Transformations

First Examples

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

$G = (V, E, \#)$ is d-out-regular $\Leftrightarrow \forall u \in V : \sum_{v \in V} \#((u, v)) = d$

Pointer-Pull:



- is not sound

Pointer-Push&Pull

Unlabeled Digraphs

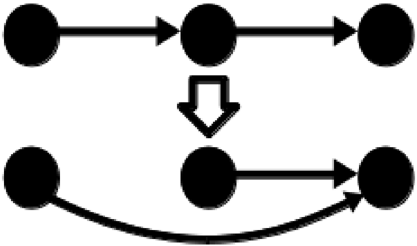
Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}.$

Pointer-Push&Pull

Unlabeled Digraphs

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}.$

Pointer-Push&Pull:

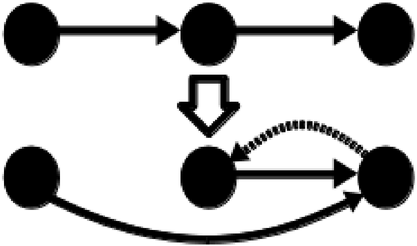


Pointer-Push&Pull

Unlabeled Digraphs

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

Pointer-Push&Pull:

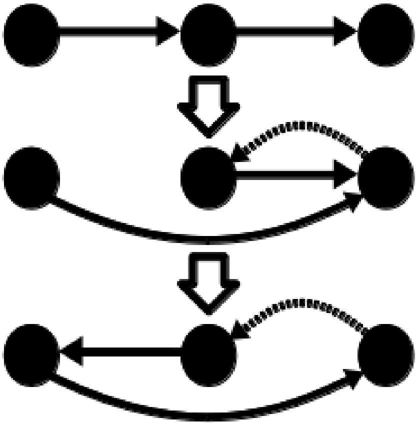


Pointer-Push&Pull

Unlabeled Digraphs

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}.$

Pointer-Push&Pull:

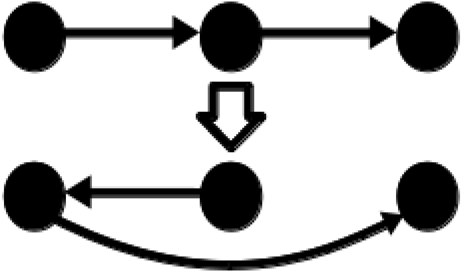


Pointer-Push&Pull

Unlabeled Digraphs

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}.$

Pointer-Push&Pull:

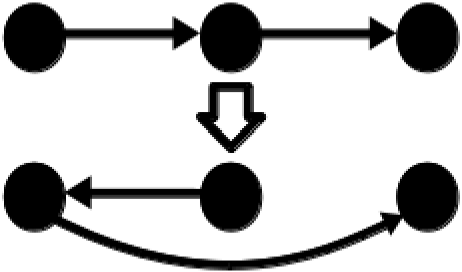


Pointer-Push&Pull

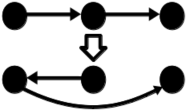
Unlabeled Digraphs

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

Pointer-Push&Pull:



- is sound
- is feasible
- is general
- is uniform general

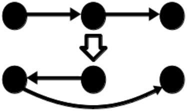


Pointer-Push&Pull

Unlabeled Digraphs

Let $G = (V, E, \#)$, $u \in V$. $N^+(u) := \{v \in V \mid \#((u, v)) > 0\}$.

Pointer-Push&Pull



Unlabeled Digraphs

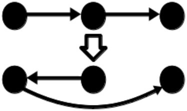
Let $G = (V, E, \#)$, $u \in V$. $N^+(u) := \{v \in V \mid \#((u,v)) > 0\}$.

Algorithm 2 Unlabeled Pointer-Push&Pull: $\tau_u : \mathcal{G}_u \rightsquigarrow \mathcal{G}_u$

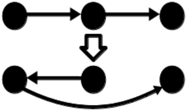
$v_1 \xleftarrow{R} V$
if random event with probability $\frac{|N^+(v_1)|}{d}$ occurs **then**
    $v_2 \xleftarrow{R} N^+(v_1)$
    if random event with probability $\frac{|N^+(v_2)|}{d}$ occurs **then**
        $v_3 \xleftarrow{R} N^+(v_2)$
        $\#((v_1, v_2)) := \#((v_1, v_2)) - 1$
        $\#((v_2, v_3)) := \#((v_2, v_3)) - 1$
        $\#((v_2, v_1)) := \#((v_2, v_1)) + 1$
        $\#((v_1, v_3)) := \#((v_1, v_3)) + 1$

Pointer-Push&Pull

Unlabeled Digraphs



Let $\mathcal{G}_u := \left\{ G \,\middle|\, \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}.$



Pointer-Push&Pull

Unlabeled Digraphs

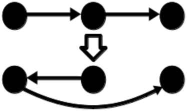
Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}.$

Lemma

$\forall G, G' \in \mathcal{G}_u : \Pr\left(\tau_u(G) = G'\right) = \Pr\left(\tau_u(G') = G\right).$

Pointer-Push&Pull

Unlabeled Digraphs



Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}.$

Lemma

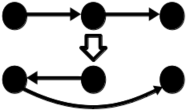
$\forall G, G' \in \mathcal{G}_u : \Pr(\tau_u(G) = G') = \Pr(\tau_u(G') = G).$

Proof.

Suppose G' is reached from G using the path (v_i, v_j, v_k).

Pointer-Push&Pull

Unlabeled Digraphs



Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}.$

Lemma

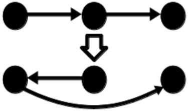
$\forall G, G' \in \mathcal{G}_u : \Pr(\tau_u(G) = G') = \Pr(\tau_u(G') = G).$

Proof.

Suppose G' is reached from G using the path (v_i, v_j, v_k).
$\Rightarrow$ G can be reached from G' exactly with (v_j, v_i, v_k).

Pointer-Push&Pull

Unlabeled Digraphs



Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

Lemma

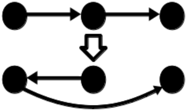
$\forall G, G' \in \mathcal{G}_u : \Pr(\tau_u(G) = G') = \Pr(\tau_u(G') = G)$.

Proof.

Suppose G' is reached from G using the path (v_i, v_j, v_k).

$\Rightarrow$ G can be reached from G' exactly with (v_j, v_i, v_k).

$\Rightarrow \Pr(\tau_u(G) = G') = \frac{1}{n} \cdot \frac{1}{d} \cdot \frac{1}{d} = \Pr(\tau_u(G') = G)$ $\square$



Pointer-Push&Pull

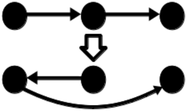
Unlabeled Digraphs

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

Lemma

Let $G, G' \in \mathcal{G}_u$. G' can be reached from G with at most $10nd$ Pointer-Push&Pull operations.

Pointer-Push&Pull



Unlabeled Digraphs

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

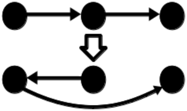
Lemma

Let $G, G' \in \mathcal{G}_u$. G' can be reached from G with at most $10nd$ Pointer-Push&Pull operations.

Proof.

Let $G = (V, E, \#)$ with $V = \{v_1, \ldots, v_n\}$.

Pointer-Push&Pull



Unlabeled Digraphs

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

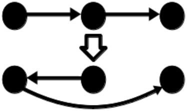
Lemma

Let $G, G' \in \mathcal{G}_u$. G' can be reached from G with at most $10nd$ Pointer-Push&Pull operations.

Proof.

Let $G = (V, E, \#)$ with $V = \{v_1, \ldots, v_n\}$.
Define $G_c := (V, E, \#_c)$ with:

$$\forall u \in V : \#_c\left((u, v_1)\right) = d,$$
$$\forall u \in V, v \in V \setminus \{v_1\} : \#_c\left((u, v)\right) = 0.$$



Pointer-Push&Pull

Unlabeled Digraphs

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

Lemma

Let $G, G' \in \mathcal{G}_u$. G' can be reached from G with at most $10nd$ Pointer-Push&Pull operations.

Proof.

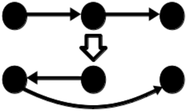
Let $G = (V, E, \#)$ with $V = \{v_1, \ldots, v_n\}$.
Define $G_c := (V, E, \#_c)$ with:

$$\forall u \in V : \#_c\left((u, v_1)\right) = d,$$
$$\forall u \in V, v \in V \setminus \{v_1\} : \#_c\left((u, v)\right) = 0.$$

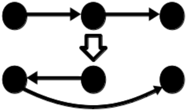
$\Rightarrow$ To show: G_c can be reached from G with at most $5nd$ Pointer-Push&Pull operations. □

Pointer-Push&Pull

Unlabeled Digraphs



Case 1: $\exists j \in \{2, \ldots, n\} : \#\left((v_1, v_j)\right) > 0$

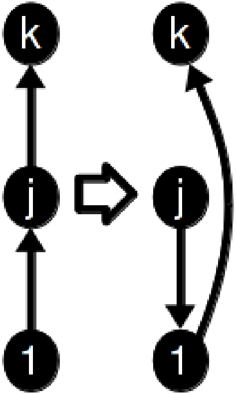


Pointer-Push&Pull

Unlabeled Digraphs

Case 1: $\exists j \in \{2, \ldots, n\} : \#((v_1, v_j)) > 0$

Case 1.1: $j \neq k \neq 1$

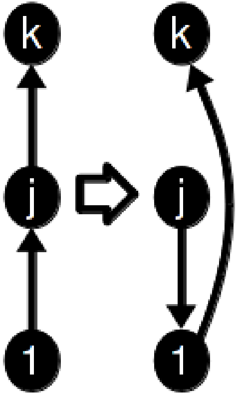


Pointer-Push&Pull

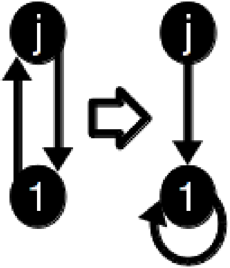
Unlabeled Digraphs

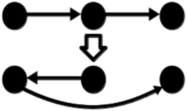
Case 1: $\exists j \in \{2, \ldots, n\} : \#\left((v_1, v_j)\right) > 0$

Case 1.1: $j \neq k \neq 1$



Case 1.2:



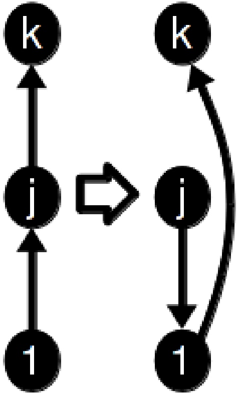


Pointer-Push&Pull

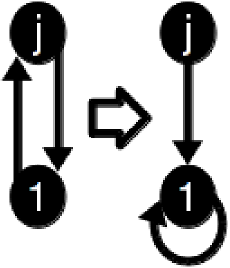
Unlabeled Digraphs

Case 1: $\exists j \in \{2, \ldots, n\} : \#\left((v_1, v_j)\right) > 0$

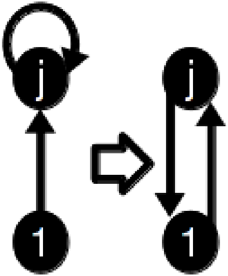
Case 1.1: $j \neq k \neq 1$

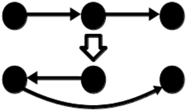


Case 1.2:



Case 1.3:





Pointer-Push&Pull

Unlabeled Digraphs

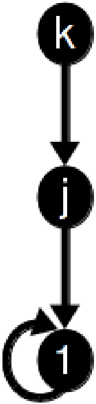
Case 2: $\#((v_1, v_1)) = d, \forall j \in \{2, \ldots, n\} : \#((v_1, v_j)) = 0$

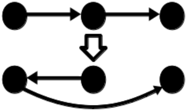
Pointer-Push&Pull

Case 2: $\#((v_1, v_1)) = d, \forall j \in \{2, \ldots, n\} : \#((v_1, v_j)) = 0$

Case 2.1:

$1 \neq j \neq k \neq 1$





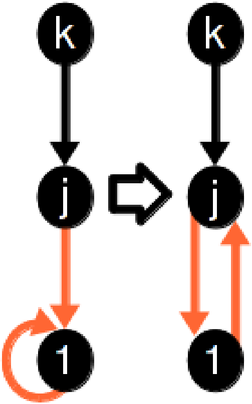
Pointer-Push&Pull

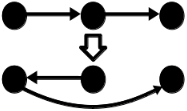
Unlabeled Digraphs

Case 2: $\#\left((v_1, v_1)\right) = d, \forall j \in \{2, \ldots, n\} : \#\left((v_1, v_j)\right) = 0$

Case 2.1:

$1 \neq j \neq k \neq 1$





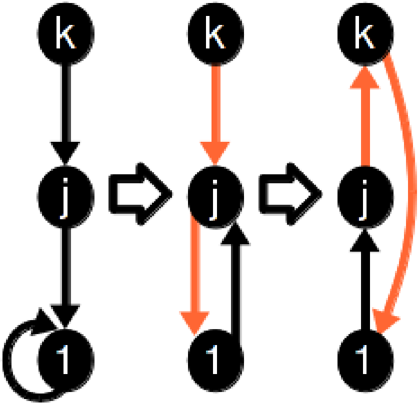
Pointer-Push&Pull

Unlabeled Digraphs

Case 2: $\#((v_1, v_1)) = d, \forall j \in \{2, \ldots, n\} : \#((v_1, v_j)) = 0$

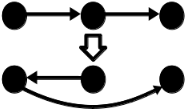
Case 2.1:

$1 \neq j \neq k \neq 1$



Pointer-Push&Pull

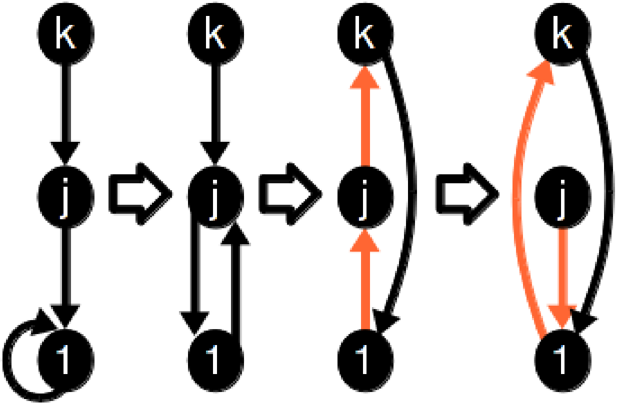
Unlabeled Digraphs



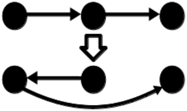
Case 2: $\#((v_1, v_1)) = d, \forall j \in \{2, \ldots, n\} : \#((v_1, v_j)) = 0$

Case 2.1:

$1 \neq j \neq k \neq 1$



Pointer-Push&Pull

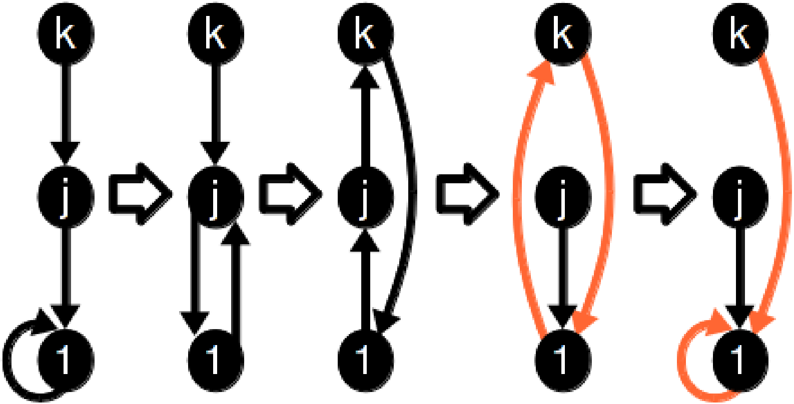


Unlabeled Digraphs

Case 2: $\#((v_1, v_1)) = d, \forall j \in \{2, \ldots, n\} : \#((v_1, v_j)) = 0$

Case 2.1:

$1 \neq j \neq k \neq 1$



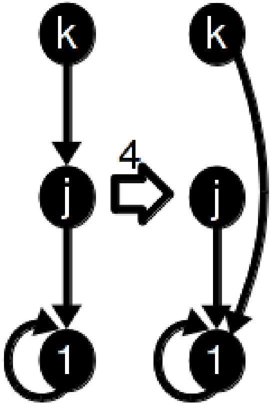
Pointer-Push&Pull

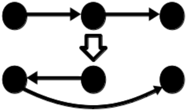
Unlabeled Digraphs

Case 2: $\#\left((v_1, v_1)\right) = d, \forall j \in \{2, \ldots, n\} : \#\left((v_1, v_j)\right) = 0$

Case 2.1:

$1 \neq j \neq k \neq 1$





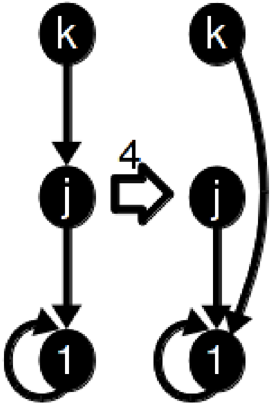
Pointer-Push&Pull

Unlabeled Digraphs

Case 2: $\#((v_1, v_1)) = d, \forall j \in \{2, \ldots, n\} : \#((v_1, v_j)) = 0$

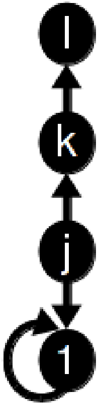
Case 2.1:

$1 \neq j \neq k \neq 1$



Case 2.2:

$1 \neq j \neq k \neq 1 \neq l \neq j$



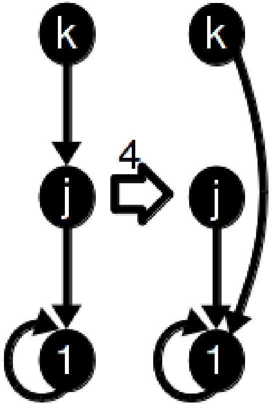
Pointer-Push&Pull

Unlabeled Digraphs

Case 2: $\#((v_1, v_1)) = d, \forall j \in \{2, \ldots, n\} : \#((v_1, v_j)) = 0$

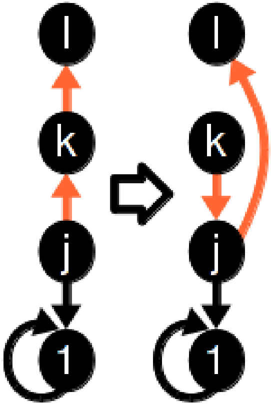
Case 2.1:

$1 \neq j \neq k \neq 1$



Case 2.2:

$1 \neq j \neq k \neq 1 \neq l \neq j$



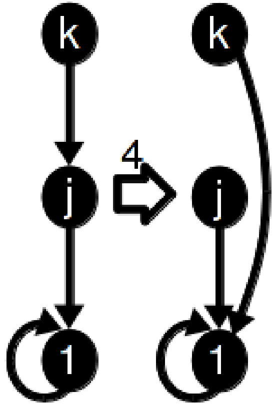
Pointer-Push&Pull

Unlabeled Digraphs

Case 2: $\#((v_1, v_1)) = d, \forall j \in \{2, \ldots, n\} : \#((v_1, v_j)) = 0$

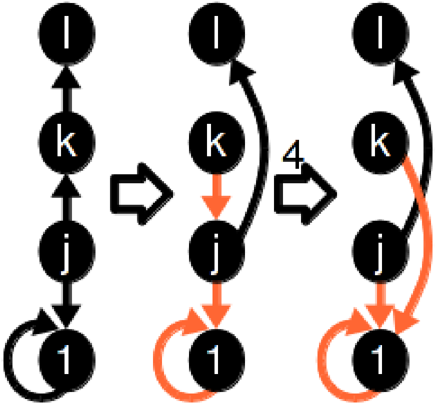
Case 2.1:

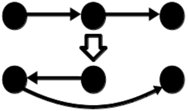
$1 \neq j \neq k \neq 1$



Case 2.2:

$1 \neq j \neq k \neq 1 \neq l \neq j$





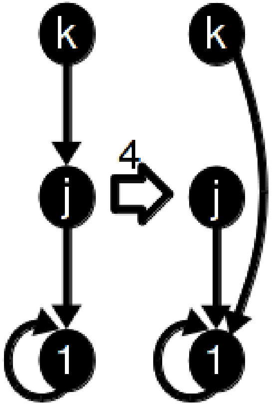
Pointer-Push&Pull

Unlabeled Digraphs

Case 2: $\#((v_1, v_1)) = d, \forall j \in \{2, \ldots, n\} : \#((v_1, v_j)) = 0$

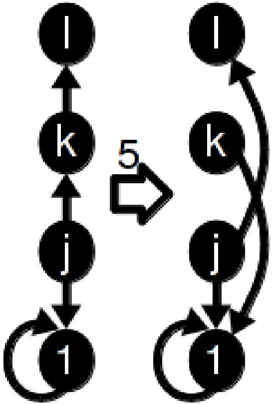
Case 2.1:

$1 \neq j \neq k \neq 1$



Case 2.2:

$1 \neq j \neq k \neq 1 \neq l \neq j$



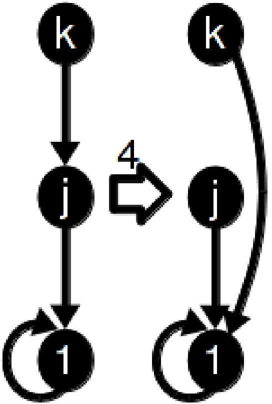
Pointer-Push&Pull

Unlabeled Digraphs

Case 2: $\#((v_1, v_1)) = d, \forall j \in \{2, \dots, n\} : \#((v_1, v_j)) = 0$

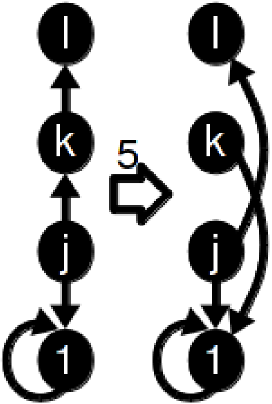
Case 2.1:

$1 \neq j \neq k \neq 1$



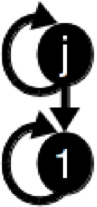
Case 2.2:

$1 \neq j \neq k \neq 1 \neq l \neq j$



Case 2.3:

$1 \neq j$



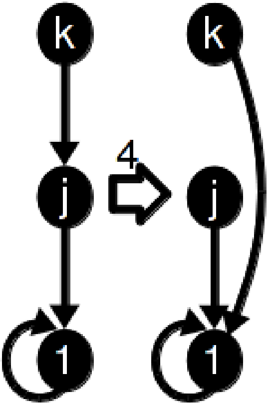
Pointer-Push&Pull

Unlabeled Digraphs

Case 2: $\#((v_1, v_1)) = d, \forall j \in \{2, \dots, n\} : \#((v_1, v_j)) = 0$

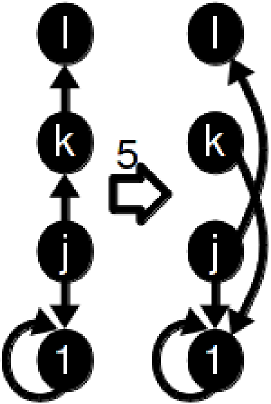
Case 2.1:

$1 \neq j \neq k \neq 1$



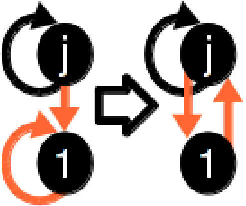
Case 2.2:

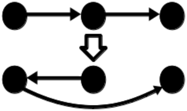
$1 \neq j \neq k \neq 1 \neq l \neq j$



Case 2.3:

$1 \neq j$





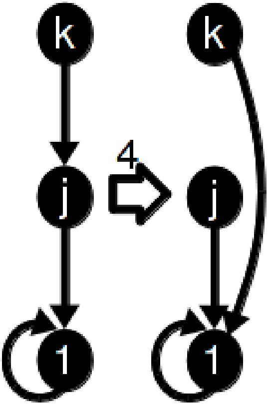
Pointer-Push&Pull

Unlabeled Digraphs

Case 2: $\#((v_1, v_1)) = d, \forall j \in \{2, \ldots, n\} : \#((v_1, v_j)) = 0$

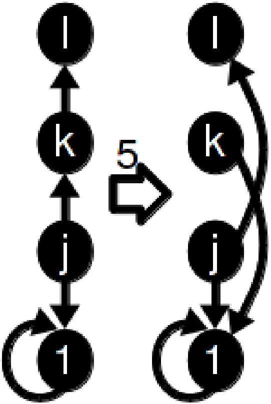
Case 2.1:

$1 \neq j \neq k \neq 1$



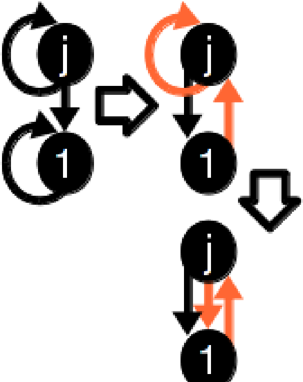
Case 2.2:

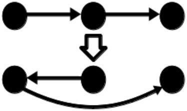
$1 \neq j \neq k \neq 1 \neq l \neq j$



Case 2.3:

$1 \neq j$





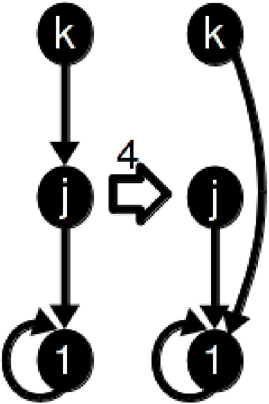
Pointer-Push&Pull

Unlabeled Digraphs

Case 2: $\#((v_1, v_1)) = d, \forall j \in \{2, \ldots, n\} : \#((v_1, v_j)) = 0$

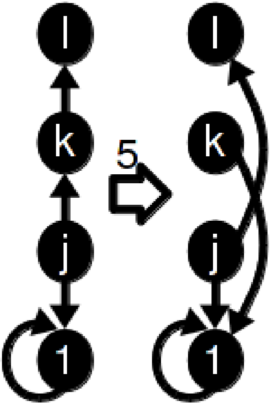
Case 2.1:

$1 \neq j \neq k \neq 1$



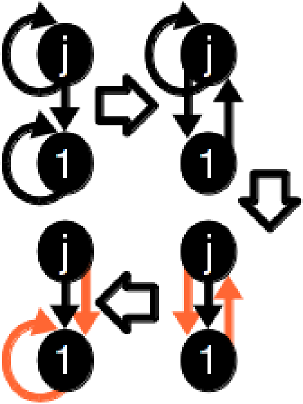
Case 2.2:

$1 \neq j \neq k \neq 1 \neq l \neq j$



Case 2.3:

$1 \neq j$



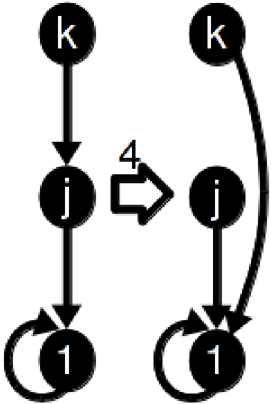
Pointer-Push&Pull

Unlabeled Digraphs

Case 2: $\#\left((v_1, v_1)\right) = d, \forall j \in \{2, \ldots, n\} : \#\left((v_1, v_j)\right) = 0$

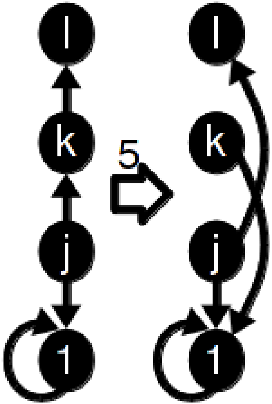
Case 2.1:

$1 \neq j \neq k \neq 1$



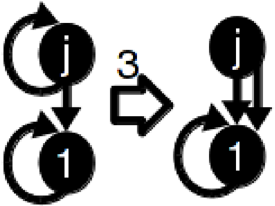
Case 2.2:

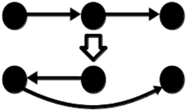
$1 \neq j \neq k \neq 1 \neq l \neq j$



Case 2.3:

$1 \neq j$





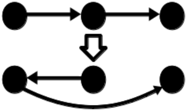
Pointer-Push&Pull

Unlabeled Digraphs

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}.$

Theorem

$\forall G, G' \in \mathcal{G}_u : \lim_{k\to\infty} \Pr\left(\tau_u^k(G) = G'\right) = \frac{1}{|\mathcal{G}_u|}$



Pointer-Push&Pull

Unlabeled Digraphs

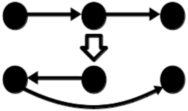
Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

Theorem

$$\forall G, G' \in \mathcal{G}_u : \lim_{k\to\infty} \Pr\left(\tau_u^k(G) = G'\right) = \frac{1}{|\mathcal{G}_u|}$$

Proof.

Follows from properties of corresponding Markov chain with transition matrix T:



Pointer-Push&Pull

Unlabeled Digraphs

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

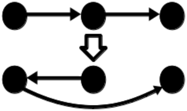
Theorem

$$\forall G, G' \in \mathcal{G}_u : \lim_{k \to \infty} \Pr\left(\tau_u^k(G) = G'\right) = \frac{1}{|\mathcal{G}_u|}$$

Proof.

Follows from properties of corresponding Markov chain with transition matrix T:

- T is symmetric
  $\Rightarrow \left(\frac{1}{|\mathcal{G}_u|}, \ldots, \frac{1}{|\mathcal{G}_u|}\right) T = \left(\frac{1}{|\mathcal{G}_u|}, \ldots, \frac{1}{|\mathcal{G}_u|}\right)$ is stationary distribution



Pointer-Push&Pull

Unlabeled Digraphs

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

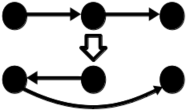
Theorem

$\forall G, G' \in \mathcal{G}_u : \lim_{k \to \infty} \Pr\left(\tau_u^k(G) = G'\right) = \frac{1}{|\mathcal{G}_u|}$

Proof.

Follows from properties of corresponding Markov chain with transition matrix T:

- T is symmetric
  $\Rightarrow \left(\frac{1}{|\mathcal{G}_u|}, \ldots, \frac{1}{|\mathcal{G}_u|}\right) T = \left(\frac{1}{|\mathcal{G}_u|}, \ldots, \frac{1}{|\mathcal{G}_u|}\right)$ is stationary distribution
- Markov chain is irreducible



Pointer-Push&Pull

Unlabeled Digraphs

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}.$

Theorem

$$\forall G, G' \in \mathcal{G}_u : \lim_{k\to\infty} \Pr\left(\tau_u^k(G) = G'\right) = \frac{1}{|\mathcal{G}_u|}$$

Proof.

Follows from properties of corresponding Markov chain with transition matrix T:

- T is symmetric
  $\Rightarrow \left(\frac{1}{|\mathcal{G}_u|}, \dots, \frac{1}{|\mathcal{G}_u|}\right) T = \left(\frac{1}{|\mathcal{G}_u|}, \dots, \frac{1}{|\mathcal{G}_u|}\right)$ is stationary distribution
- Markov chain is irreducible
- T has some non-zero diagonal entries
  $\Rightarrow$ Markov chain is aperiodic

Pointer-Push&Pull

Edge Labeled Digraphs

Pointer-Push&Pull

Edge Labeled Digraphs

Definition (Edge Labeled Multi-Digraph)

An edge labeled d-out-regular multi-digraph $G = (V, E)$ is defined by a node set $V = \{v_1, \ldots, v_n\}$ and a set of directed edges $E \subseteq \{(u, v, i) \mid u, v \in V, i \in \{1, \ldots, d\}\}$ with:

$$\forall u \in V \, \forall i \in \{1, \ldots, d\} \, \exists! N^+(u, i) \in V : (u, N^+(u, i), i) \in E.$$

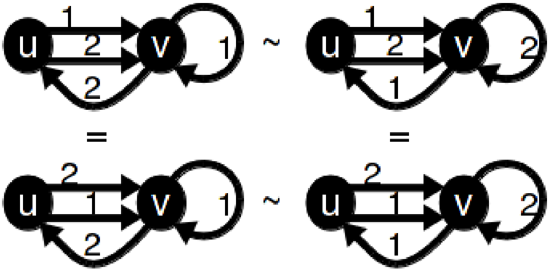
Pointer-Push&Pull

Edge Labeled Digraphs

Definition (Edge Labeled Multi-Digraph)

An edge labeled d-out-regular multi-digraph $G = (V, E)$ is defined by a node set $V = \{v_1, \ldots, v_n\}$ and a set of directed edges $E \subseteq \{(u, v, i) \mid u, v \in V, i \in \{1, \ldots, d\}\}$ with:

$$\forall u \in V \, \forall i \in \{1, \ldots, d\} \, \exists! N^+(u, i) \in V : (u, N^+(u, i), i) \in E.$$



Pointer-Push&Pull

Edge Labeled Digraphs

Definition (Equivalence Class)

Let G be an unlabeled d-out-regular multi-digraph. $[G]$ denotes the set of all edge labeled d-out-regular multi-digraphs describing G when omitting the edge labels.

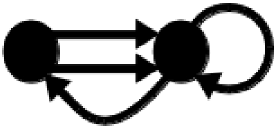
Pointer-Push&Pull

Edge Labeled Digraphs

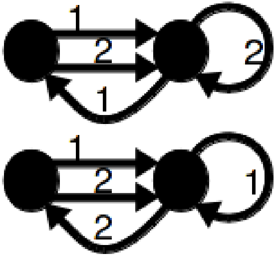
Definition (Equivalence Class)

Let G be an unlabeled d-out-regular multi-digraph. $[G]$ denotes the set of all edge labeled d-out-regular multi-digraphs describing G when omitting the edge labels.

G



$[G]$



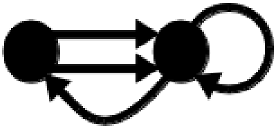
Pointer-Push&Pull

Edge Labeled Digraphs

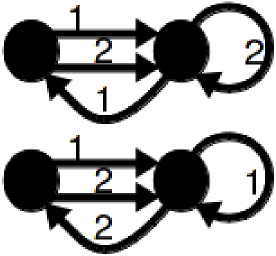
Definition (Equivalence Class)

Let G be an unlabeled d-out-regular multi-digraph. $[G]$ denotes the set of all edge labeled d-out-regular multi-digraphs describing G when omitting the edge labels.

G



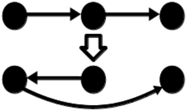
$[G]$



$\Rightarrow$ The lower the number of multi-edges in G, the larger is $|[G]|$.

Pointer-Push&Pull

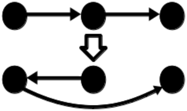
Edge Labeled Digraphs



Let $\mathcal{G}_l := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is an edge labeled weakly-connected} \\ d\text{-out-regular multi-digraph with } n \text{ nodes} \end{array} \right\}$.

Pointer-Push&Pull

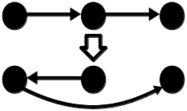
Edge Labeled Digraphs



Let $\mathcal{G}_l := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is an edge labeled weakly-connected} \\ d\text{-out-regular multi-digraph with } n \text{ nodes} \end{array} \right\}$.

Algorithm 4 Labeled Pointer-Push&Pull: $\tau_l : \mathcal{G}_l \rightsquigarrow \mathcal{G}_l$

$v_1 \stackrel{R}{\leftarrow} V$
$i \stackrel{R}{\leftarrow} \{1, \ldots, d\}$
$v_2 := N^+(v_1, i)$
$j \stackrel{R}{\leftarrow} \{1, \ldots, d\}$
$v_3 := N^+(v_2, j)$
$E := (E \setminus \{(v_1, v_2, i), (v_2, v_3, j)\}) \cup \{(v_2, v_1, j), (v_1, v_3, i)\}$



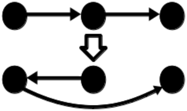
Pointer-Push&Pull

Edge Labeled Digraphs

Let $\mathcal{G}_I := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is an edge labeled weakly-connected} \\ d\text{-out-regular multi-digraph with } n \text{ nodes} \end{array} \right\}$.

As before: τ_I is

- sound
- feasible
- general
- uniform general



Pointer-Push&Pull

Edge Labeled Digraphs

Let $\mathcal{G}_l := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is an edge labeled weakly-connected} \\ d\text{-out-regular multi-digraph with } n \text{ nodes} \end{array} \right\}$.

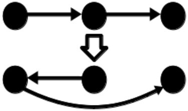
As before: τ_l is

- sound
- feasible
- general
- uniform general

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

Theorem

$$\forall G, G' \in \mathcal{G}_u : \lim_{k\to\infty} \Pr\left(\tau_l^k(G) = G'\right) = \frac{|[G']|}{|\mathcal{G}_l|}.$$



Pointer-Push&Pull

Edge Labeled Digraphs

Let $\mathcal{G}_l := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is an edge labeled weakly-connected} \\ d\text{-out-regular multi-digraph with } n \text{ nodes} \end{array} \right\}$.

As before: τ_l is

- sound
- feasible
- general
- uniform general

Let $\mathcal{G}_u := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is a weakly-connected } d\text{-out-regular} \\ \text{multi-digraph with } n \text{ nodes} \end{array} \right\}$.

Theorem

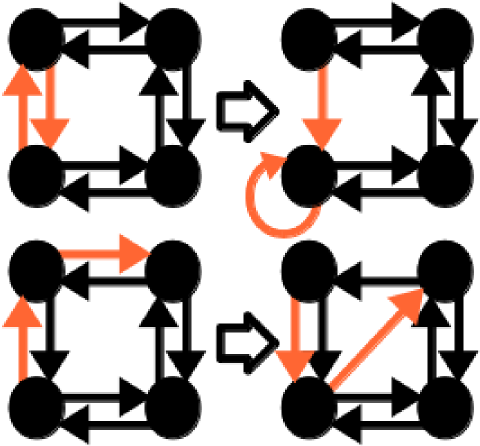
$\forall G, G' \in \mathcal{G}_u : \lim_{k\to\infty} \Pr\left(\tau_l^k(G) = G'\right) = \frac{|[G']|}{|\mathcal{G}_l|}$.

$\Rightarrow$ A particular simple digraph is more probable than a particular multi-digraph.

Simple Graphs

Simple Graphs

Pointer-Push&Pull cannot be restricted to simple graphs:



Simple Graphs

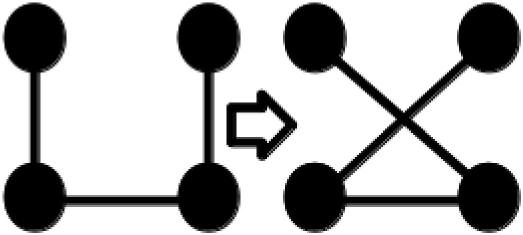
Let $\mathcal{G}_s := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is an undirected connected } d\text{-regular} \\ \text{simple graph with } n \text{ nodes} \end{array} \right\}.$

1-Flipper:

Simple Graphs

Let $\mathcal{G}_s := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is an undirected connected } d\text{-regular} \\ \text{simple graph with } n \text{ nodes} \end{array} \right\}$.

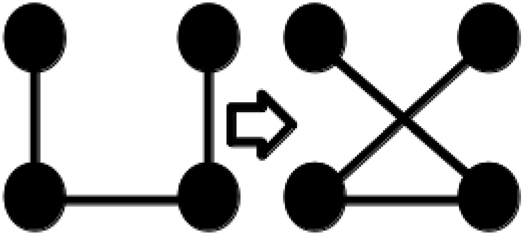
1-Flipper:



Simple Graphs

Let $\mathcal{G}_s := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is an undirected connected } d\text{-regular} \\ \text{simple graph with } n \text{ nodes} \end{array} \right\}.$

1-Flipper:

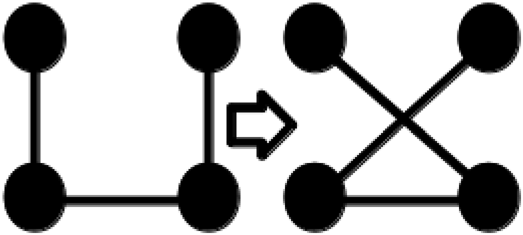


- is sound
- is feasible
- is general
- is uniform general

Simple Graphs

Let $\mathcal{G}_s := \left\{ G \;\middle|\; \begin{array}{l} G \text{ is an undirected connected } d\text{-regular} \\ \text{simple graph with } n \text{ nodes} \end{array} \right\}.$

1-Flipper:



- is sound
- is feasible
- is general
- is uniform general

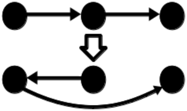
- Four peers have to participate actively
- Digraphs are sufficient in practice

Peer-to-Peer Networks

Implementation of Pointer-Push&Pull

Peer-to-Peer Networks

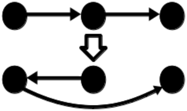
Implementation of Pointer-Push&Pull



1. v_1 requests a random neighbor from v_2
2. v_2 replaces v_3 by v_1 in neighborhood list and sends ID of v_3 to v_1
3. v_1 receives ID of v_3 from v_2 and replaces v_2 by v_3 in neighborhood list

Peer-to-Peer Networks

Implementation of Pointer-Push&Pull



1. v_1 requests a random neighbor from v_2
2. v_2 replaces v_3 by v_1 in neighborhood list and sends ID of v_3 to v_1
3. v_1 receives ID of v_3 from v_2 and replaces v_2 by v_3 in neighborhood list

$\Rightarrow$ only two network operations
$\Rightarrow$ no additional overhead to periodical neighborhood verification

Peer-to-Peer Networks

Advantage of Pointer-Push&Pull

Peer-to-Peer Networks

Advantage of Pointer-Push&Pull

Pointer-Push&Pull leads (with high probability) to random graphs with

- constant and small out-degree
- logarithmic diameter
- high connectivity

Peer-to-Peer Networks

Advantage of Pointer-Push&Pull

Pointer-Push&Pull leads (with high probability) to random graphs with

- constant and small out-degree
- logarithmic diameter
- high connectivity

Open Problems:

- Convergence rate
  - $O(n \log n)$ supposed
  - Simulations indicate quick convergence
- Similar operation for simple digraphs

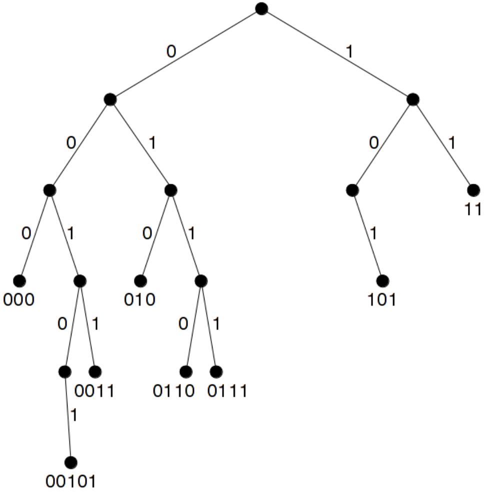
Peer-to-Peer Networks

Example: 3nuts

Peer-to-Peer Networks

Example: 3nuts

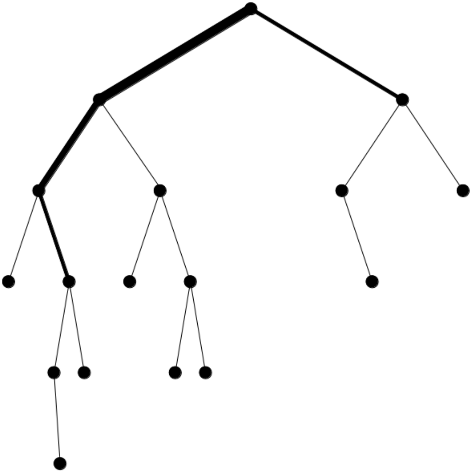
Data tree:
Prefix tree of
data identities



Peer-to-Peer Networks

Example: 3nuts

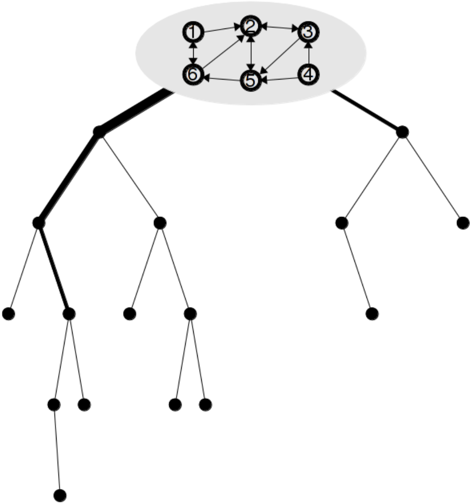
Network tree:



Peer-to-Peer Networks

Example: 3nuts

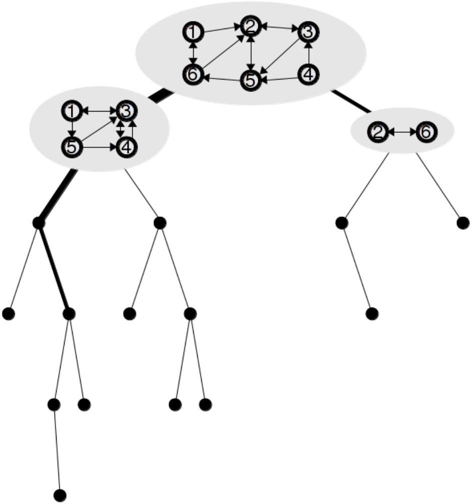
Network tree:



Peer-to-Peer Networks

Example: 3nuts

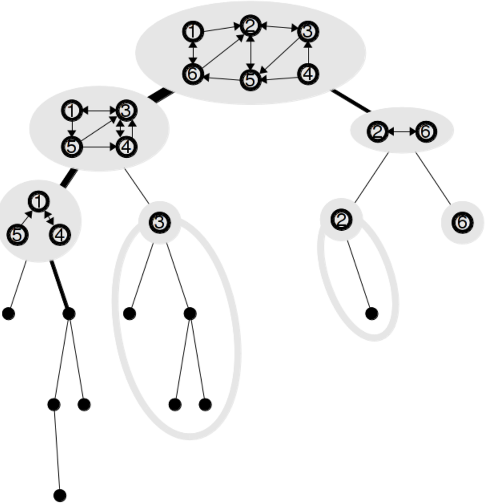
Network tree:



Peer-to-Peer Networks

Example: 3nuts

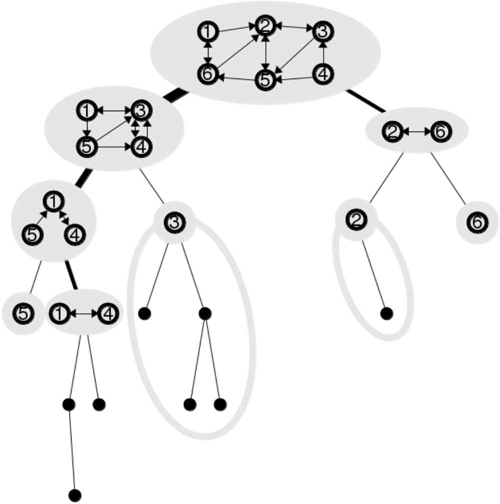
Network tree:



Peer-to-Peer Networks

Example: 3nuts

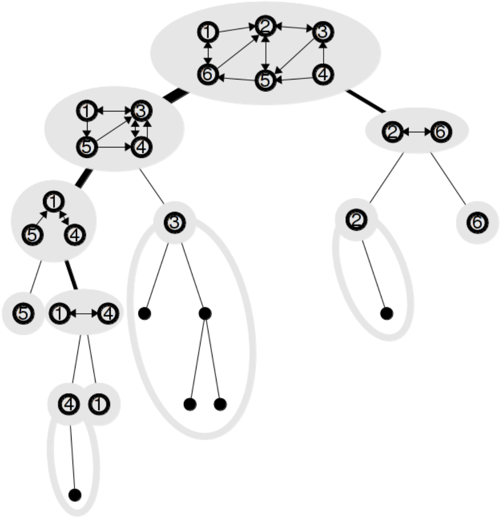
Network tree:



Peer-to-Peer Networks

Example: 3nuts

Network tree:



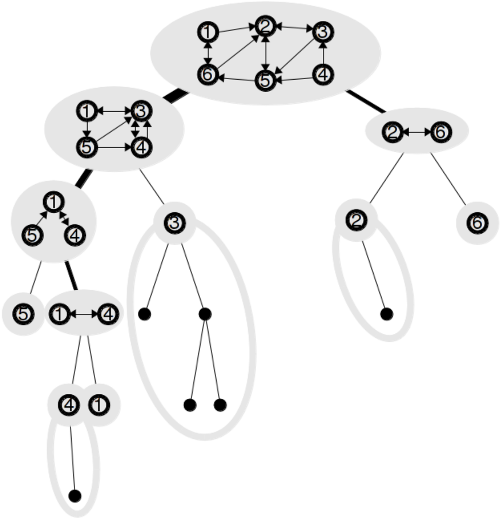
Peer-to-Peer Networks

Example: 3nuts

Network tree:

For each random network a peer has to save:

- random neighbors



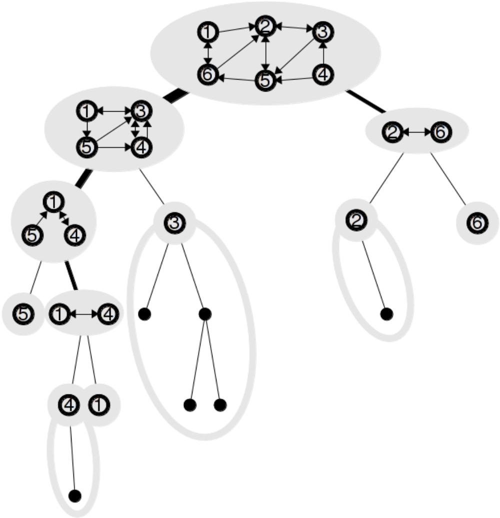
Peer-to-Peer Networks

Example: 3nuts

Network tree:

For each random network a peer has to save:

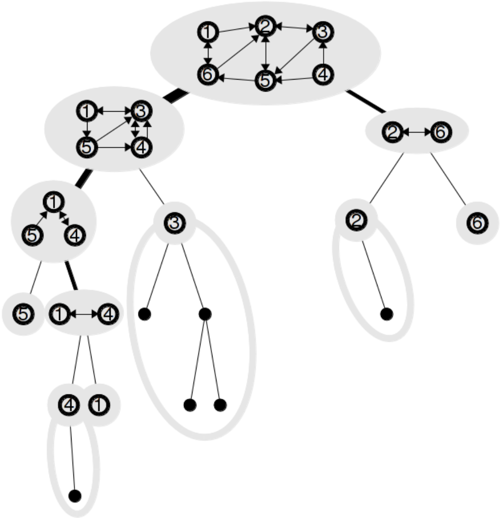
- random neighbors
- branch links to each child
  - random branch links
  - local branch links



Peer-to-Peer Networks

Example: 3nuts

Network tree:



For each random network a peer has to save:

- random neighbors
- branch links to each child
  - random branch links
  - local branch links
- responsible peers

Peer-to-Peer Networks

Example: 3nuts

Role of Pointer-Push&Pull:

Peer-to-Peer Networks

Example: 3nuts

Role of Pointer-Push&Pull:

- maintain truly random networks
  $\Rightarrow$ robustness

Peer-to-Peer Networks

Example: 3nuts

Role of Pointer-Push&Pull:

- maintain truly random networks
  $\Rightarrow$ robustness
- spread information among peers, e.g. tree structure, weights

Peer-to-Peer Networks

Example: 3nuts

Role of Pointer-Push&Pull:

- maintain truly random networks
  $\Rightarrow$ robustness
- spread information among peers, e.g. tree structure, weights
- update random branch links and guarantee them to be truly random

Peer-to-Peer Networks

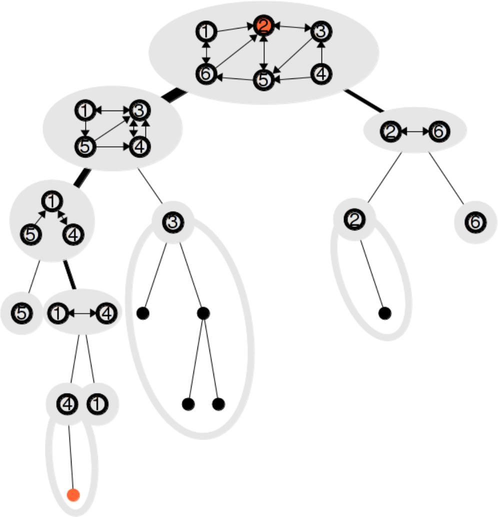
Example: 3nuts

Role of Pointer-Push&Pull:

- maintain truly random networks
  $\Rightarrow$ robustness
- spread information among peers, e.g. tree structure, weights
- update random branch links and guarantee them to be truly random
- measure round trip times and find good local branch links

Peer-to-Peer Networks

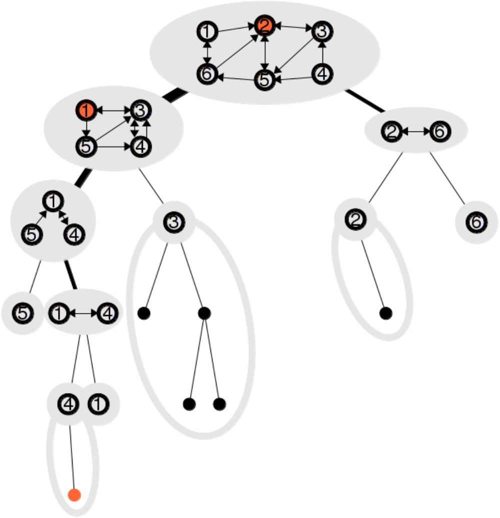
Example: 3nuts



Routing:

Peer-to-Peer Networks

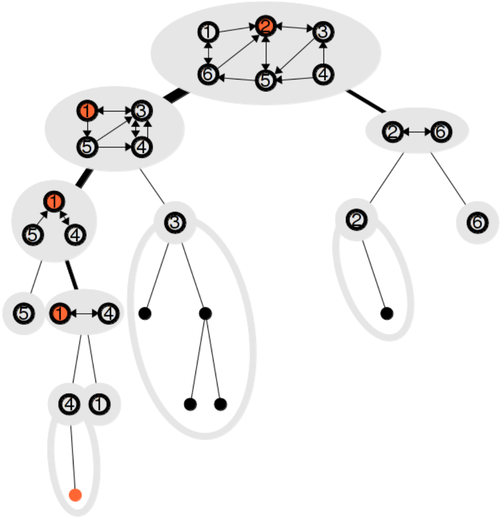
Example: 3nuts



Routing:

Peer-to-Peer Networks

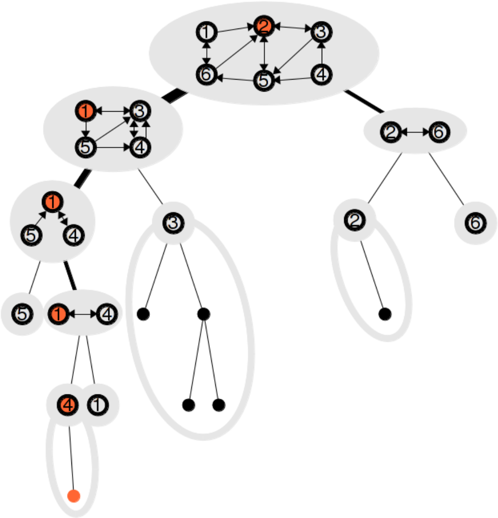
Example: 3nuts



Routing:

Peer-to-Peer Networks

Example: 3nuts



Routing:

Peer-to-Peer Networks

Example: 3nuts

Routing:

- Use random branch links
  $\Rightarrow$ Number of hops in 3nuts with n peers is in $O(\log n)$ with high probability
- Use local branch links
  $\Rightarrow$ Experimental evaluation shows that this can benefit routing