Gossip in Dynamic Networks

Jan van Eijck
CWI & ILLC, Amsterdam

The 3rd Tsinghua Logic Colloquium, Beijing

October 21, 2015

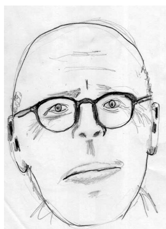
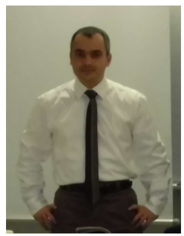
Abstract

A gossip protocol is a procedure for spreading secrets among a group of agents, using a connection graph. In this talk the problem of designing and analyzing gossip protocols is given a dynamic twist by assuming that when a call is established not only secrets are exchanged but also contact lists, i.e., links in the gossip graph. Thus, each call in the gossip graph changes both the graph and the distribution of secrets. In the talk, we give a full characterization for the class of dynamic gossip graphs where the Learn New Secrets protocol (make a call to an agent if you know the number but not the secret of that agent) is successful.

Abstract

A gossip protocol is a procedure for spreading secrets among a group of agents, using a connection graph. In this talk the problem of designing and analyzing gossip protocols is given a dynamic twist by assuming that when a call is established not only secrets are exchanged but also contact lists, i.e., links in the gossip graph. Thus, each call in the gossip graph changes both the graph and the distribution of secrets. In the talk, we give a full characterization for the class of dynamic gossip graphs where the Learn New Secrets protocol (make a call to an agent if you know the number but not the secret of that agent) is successful.

This is joint work with Hans van Ditmarsch, Pere Pardo, Rahim Ramezanian and François Schwarzentruber

Overview

- What are Gossip Protocols? Brief History
- Gossip in Totally Connected Networks
- Distributed Protocols
- The Dynamic Turn
- Examples of Gossip Graph Completion
- The Learn New Secrets Protocol
- Results
- Further Questions

An Abstract Perspective on Gossip Spreading

Key notions: gossip graph, gossip secret, gossip call, gossip protocol.

A gossip protocol is a protocol for spreading secrets among a group of agents, using a connection graph.

An Abstract Perspective on Gossip Spreading

Key notions: gossip graph, gossip secret, gossip call, gossip protocol.

A gossip protocol is a protocol for spreading secrets among a group of agents, using a connection graph.

- What do we assume about the graph?

An Abstract Perspective on Gossip Spreading

Key notions: gossip graph, gossip secret, gossip call, gossip protocol.

A gossip protocol is a protocol for spreading secrets among a group of agents, using a connection graph.

- What do we assume about the graph?
- What do we assume about the secrets?

An Abstract Perspective on Gossip Spreading

Key notions: gossip graph, gossip secret, gossip call, gossip protocol.

A gossip protocol is a protocol for spreading secrets among a group of agents, using a connection graph.

- What do we assume about the graph?
- What do we assume about the secrets?
- What do we assume about the protocol? In particular: is there a central authority, or is the protocol distributed?

Brief History of Gossip Analysis

Brief History of Gossip Analysis

- Gossip in totally connected graphs, central authority: 1970s [Tij71, Hur00]

Brief History of Gossip Analysis

- Gossip in totally connected graphs, central authority: 1970s [Tij71, Hur00]
- Gossip in various classes of graphs, central authority: 1980s [HHL88]

Brief History of Gossip Analysis

- Gossip in totally connected graphs, central authority: 1970s [Tij71, Hur00]
- Gossip in various classes of graphs, central authority: 1980s [HHL88]
- Connection to study of epidemics: early 21st century [EGKM04]

Brief History of Gossip Analysis

- Gossip in totally connected graphs, central authority: 1970s [Tij71, Hur00]
- Gossip in various classes of graphs, central authority: 1980s [HHL88]
- Connection to study of epidemics: early 21st century [EGKM04]
- Distributed gossip protocols: recent interest (2014, 2015): [AvDGvdH14, AGvdH15]. Also see: [WSvE10].

Brief History of Gossip Analysis

- Gossip in totally connected graphs, central authority: 1970s [Tij71, Hur00]
- Gossip in various classes of graphs, central authority: 1980s [HHL88]
- Connection to study of epidemics: early 21st century [EGKM04]
- Distributed gossip protocols: recent interest (2014, 2015): [AvDGvdH14, AGvdH15]. Also see: [WSvE10].
- Distributed gossip protocols: dynamic turn (this talk) [DvEP$^+$15a, DvEP$^+$15b]

Gossip in totally connected graphs, central authority

Assumptions: graph totally connected, during a call all secrets are exchanged.

Key question: find a minimal sequence of calls to achieve a state where all agents know all secrets. What are the lengths of these minimal sequences?

Gossip in totally connected graphs, central authority

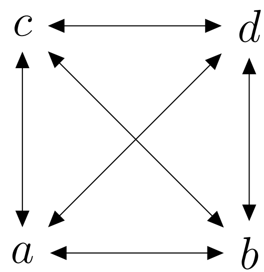
Assumptions: graph totally connected, during a call all secrets are exchanged.

Key question: find a minimal sequence of calls to achieve a state where all agents know all secrets. What are the lengths of these minimal sequences?

Fact: In a totally connected graph with $n > 3$ agents, $2n - 4$ calls are sufficient.

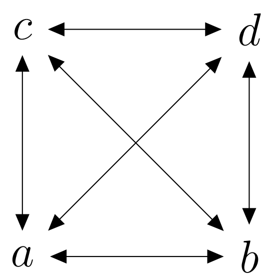
Gossip in Totally Connected Graphs: Example

Consider the totally connected graph with four agents, each with a unique secret.



Gossip in Totally Connected Graphs: Example

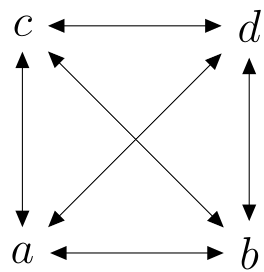
Consider the totally connected graph with four agents, each with a unique secret.



Consider calling sequence $ab; cd; ac; bd$. Four agents, four calls, and all secrets are shared.

Gossip in Totally Connected Graphs: Example

Consider the totally connected graph with four agents, each with a unique secret.



Consider calling sequence $ab; cd; ac; bd$. Four agents, four calls, and all secrets are shared.

Now suppose e is also present. Consider ae, next $ab; cd; ac; bd$, finally ae. Two extra calls are enough to accommodate one extra agent.

For n agents, $2n-4$ calls are enough

Let $n > 3$.

Basis: $n = 4$. We have seen that $4 = 8-4$ calls are enough.

Induction step: Assume for n agents $2n-4$ calls are enough. Next, add one extra agent x. Start with call from x to a, end with call from a to x, and all secrets are shared. This shows that $(2n-4)+2 = 2(n+1)-4$ calls are enough. $\square$

For n agents, $2n-4$ calls are enough

Let $n > 3$.

Basis: $n = 4$. We have seen that $4 = 8 - 4$ calls are enough.

Induction step: Assume for n agents $2n - 4$ calls are enough. Next, add one extra agent x. Start with call from x to a, end with call from a to x, and all secrets are shared. This shows that $(2n-4)+2 = 2(n+1)-4$ calls are enough. $\square$

It is a bit trickier to show that $2n-4$ calls are needed: see the original [Tij71], or [Hur00] and the references given there.

For n agents, $2n - 4$ calls are enough

Let $n > 3$.

Basis: $n = 4$. We have seen that $4 = 8 - 4$ calls are enough.

Induction step: Assume for n agents $2n - 4$ calls are enough. Next, add one extra agent x. Start with call from x to a, end with call from a to x, and all secrets are shared. This shows that $(2n - 4) + 2 = 2(n + 1) - 4$ calls are enough. $\square$

It is a bit trickier to show that $2n - 4$ calls are needed: see the original [Tij71], or [Hur00] and the references given there.

Note: in graphs that are weakly but not totally connected, the minimum number of calls to distribute all gossip may be larger than $2n - 4$ [HHL88].

Distributed Protocols

In distributed computing we look for procedures that do not need outside regulation.

Distributed Protocols

In distributed computing we look for procedures that do not need outside regulation.

Possible distributed protocol for gossip spreading:

Search For Secrets

While not every agent knows all secrets, randomly select a pair xy and let x call y.

Distributed Protocols, Dynamic Turn

Distributed Protocols, Dynamic Turn

- We drop the assumption that the graph of connections is complete from the start.

Distributed Protocols, Dynamic Turn

- We drop the assumption that the graph of connections is complete from the start.
- When a call is established not only secrets are exchanged but also contact lists (or: information about the graph).

Distributed Protocols, Dynamic Turn

- We drop the assumption that the graph of connections is complete from the start.
- When a call is established not only secrets are exchanged but also contact lists (or: information about the graph).
- Calls in the gossip graph are constrained by the current distribution of numbers, and each call changes both the graph and the distribution of secrets.

Distributed Protocols, Dynamic Turn

- We drop the assumption that the graph of connections is complete from the start.
- When a call is established not only secrets are exchanged but also contact lists (or: information about the graph).
- Calls in the gossip graph are constrained by the current distribution of numbers, and each call changes both the graph and the distribution of secrets.
- The network is given in distributed fashion: $(x, y) \in N$ iff y is in the contact list of x (think of contact lists in smartphones). These contact lists are exchanged (merged) when a call is made.

Representing Gossip Graphs

Representing Gossip Graphs

- Represent a gossip graph G as a triple (A, N, S).

Representing Gossip Graphs

- Represent a gossip graph G as a triple (A, N, S).
- A is the (finite) set of vertices or agents.

Representing Gossip Graphs

- Represent a gossip graph G as a triple (A, N, S).
- A is the (finite) set of vertices or agents.
- $N \subseteq A^2$ and $S \subseteq A^2$ are relations on A.

Representing Gossip Graphs

- Represent a gossip graph G as a triple (A, N, S).
- A is the (finite) set of vertices or agents.
- $N \subseteq A^2$ and $S \subseteq A^2$ are relations on A.
- Nxy expresses that x has a link to y (or: x does know the phone number of y).

Representing Gossip Graphs

- Represent a gossip graph G as a triple (A, N, S).
- A is the (finite) set of vertices or agents.
- $N \subseteq A^2$ and $S \subseteq A^2$ are relations on A.
- Nxy expresses that x has a link to y (or: x does know the phone number of y).
- Sxy expresses that x does know the secret of y.

Representing Gossip Graphs

- Represent a gossip graph G as a triple (A, N, S).
- A is the (finite) set of vertices or agents.
- $N \subseteq A^2$ and $S \subseteq A^2$ are relations on A.
- Nxy expresses that x has a link to y (or: x does know the phone number of y).
- Sxy expresses that x does know the secret of y.
- Alternatively, we can think of N and S as functions in $A \to \mathcal{P}A$, so that N_x is the set of agents whose numbers are known by x, and S_x is the set of agents whose secrets are known by x.

Calls and Their Effects

Represent a call from x to y as a tuple xy.

Calls and Their Effects

Represent a call from x to y as a tuple xy.

The call xy is possible in $G = (A, N, S)$ if Nxy.

Calls and Their Effects

Represent a call from x to y as a tuple xy.

The call xy is possible in $G = (A, N, S)$ if Nxy.

A call xy merges the secret lists and the contact lists of x and y. Let G^{xy} be the result of this merge in G.

Calls and Their Effects

Represent a call from x to y as a tuple xy.

The call xy is possible in $G = (A, N, S)$ if Nxy.

A call xy merges the secret lists and the contact lists of x and y. Let G^{xy} be the result of this merge in G.

If $G = (A, N, S)$ and $x, y \in A$, then $G^{xy} = (A, N', S')$ where

- N' is $N \cup \{(x, y), (y, x)\} \circ N$,
- S' is $S \cup \{(x, y), (y, x)\} \circ S$.

Calling Sequences, Accessible Secrets

Calling Sequences, Accessible Secrets

Calling Sequence A calling sequence σ is a finite list of calls.

Calling Sequences, Accessible Secrets

Calling Sequence A calling sequence σ is a finite list of calls.

Result after σ Let $G^\sigma = (A, N^\sigma, S^\sigma)$ be the graph that results after calling sequence σ.

Calling Sequences, Accessible Secrets

Calling Sequence A calling sequence σ is a finite list of calls.

Result after σ Let $G^{\sigma} = (A, N^{\sigma}, S^{\sigma})$ be the graph that results after calling sequence σ.

Possible calling sequence Define by recursion:

- ϵ is possible on any G,
- $\sigma; xy$ is possible on G iff σ is possible on G, and $N^{\sigma}xy$ holds.

Calling Sequences, Accessible Secrets

Calling Sequence A calling sequence σ is a finite list of calls.

Result after σ Let $G^{\sigma} = (A, N^{\sigma}, S^{\sigma})$ be the graph that results after calling sequence σ.

Possible calling sequence Define by recursion:

- ϵ is possible on any G,
- $\sigma; xy$ is possible on G iff σ is possible on G, and $N^{\sigma}xy$ holds.

Accessible secrets $G = (A, N, S)$ has accessible secrets if $I_A \subseteq S \subseteq N$, where $I_A = \{(a, a) \mid a \in A\}$.

Thus, G has accessible secrets iff every agent knows her own secret and moreover, if agent x knows the secret of y, x also knows the number of y.

Example

$$
\begin{tikzcd}
a : \{a\} \arrow[rr] & & b : \{b\} \arrow[dl] \\
& c : \{c\} &
\end{tikzcd}
$$

Example

$$
\begin{tikzcd}
a : \{a\} \arrow[rr] & & b : \{b\} \arrow[dl] \\
& c : \{c\} &
\end{tikzcd}
$$

$$\Rightarrow ab \Rightarrow$$

Example

$$
\begin{tikzcd}
a : \{a\} \arrow[rr] & & b : \{b\} \arrow[dl] \\
& c : \{c\} &
\end{tikzcd}
$$

$$\Rightarrow ab \Rightarrow$$

$$
\begin{tikzcd}
a : \{a, b\} \arrow[rr, leftrightarrow] \arrow[dr] & & b : \{a, b\} \arrow[dl] \\
& c : \{c\} &
\end{tikzcd}
$$

Example

$$
\begin{tikzcd}
a : \{a\} \arrow[rr] & & b : \{b\} \arrow[dl] \\
& c : \{c\} &
\end{tikzcd}
$$

$$\Rightarrow ab \Rightarrow$$

$$
\begin{tikzcd}
a : \{a, b\} \arrow[rr, leftrightarrow] \arrow[dr] & & b : \{a, b\} \arrow[dl] \\
& c : \{c\} &
\end{tikzcd}
$$

$$\Rightarrow bc \Rightarrow$$

Example

$$\begin{tikzcd} a : \{a\} \arrow[rr] & & b : \{b\} \arrow[dl] \\ & c : \{c\} & \end{tikzcd}$$

$$\Rightarrow ab \Rightarrow$$

$$\begin{tikzcd} a : \{a, b\} \arrow[rr, leftrightarrow] \arrow[dr] & & b : \{a, b\} \arrow[dl] \\ & c : \{c\} & \end{tikzcd}$$

$$\Rightarrow bc \Rightarrow$$

$$
\begin{tikzcd}
a : \{a, b\} \arrow[rr, leftrightarrow] \arrow[dr, leftrightarrow] & & b : \{a, b, c\} \arrow[dl, leftrightarrow] \\
& c : \{a, b, c\} &
\end{tikzcd}
$$

$$
\begin{tikzcd}
a : \{a, b\} \arrow[rr, leftrightarrow] \arrow[dr, leftrightarrow] & & b : \{a, b, c\} \arrow[dl, leftrightarrow] \\
& c : \{a, b, c\} &
\end{tikzcd}
$$

$$\Rightarrow ac \Rightarrow$$

$$
\begin{tikzcd}
a : \{a, b\} \arrow[rr, leftrightarrow] \arrow[dr, leftrightarrow] & & b : \{a, b, c\} \arrow[dl, leftrightarrow] \\
& c : \{a, b, c\} &
\end{tikzcd}
$$

$$\Rightarrow ac \Rightarrow$$

$$
\begin{tikzcd}
a : \{a, b, c\} \arrow[rr, leftrightarrow] \arrow[dr, leftrightarrow] & & b : \{a, b, c\} \arrow[dl, leftrightarrow] \\
& c : \{a, b, c\} &
\end{tikzcd}
$$

Example Revisited: Calls in Different Order

$$
\begin{tikzcd}
a : \{a\} \arrow[rr] & & b : \{b\} \arrow[dl] \\
& c : \{c\} &
\end{tikzcd}
\quad \Rightarrow bc \Rightarrow \quad
\begin{tikzcd}
a : \{a\} \arrow[rr] & & b : \{b, c\} \arrow[dl, leftrightarrow] \\
& c : \{b, c\} &
\end{tikzcd}
$$

Example Revisited: Calls in Different Order

$$\begin{tikzcd} a : \{a\} \arrow[rr] & & b : \{b\} \arrow[dl] \\ & c : \{c\} & \end{tikzcd} \quad \Rightarrow bc \Rightarrow \quad \begin{tikzcd} a : \{a\} \arrow[rr] & & b : \{b, c\} \arrow[dl, leftrightarrow] \\ & c : \{b, c\} & \end{tikzcd}$$

$$\Rightarrow ab \Rightarrow \quad \begin{tikzcd} a : \{a, b, c\} \arrow[rr, leftrightarrow] \arrow[dr] & & b : \{a, b, c\} \arrow[dl, leftrightarrow] \\ & c : \{b, c\} & \end{tikzcd}$$

Example Revisited: Calls in Different Order

$$\begin{tikzcd} a:\{a\} \arrow[rr] & & b:\{b\} \arrow[dl] \\ & c:\{c\} & \end{tikzcd} \quad \Rightarrow bc \Rightarrow \quad \begin{tikzcd} a:\{a\} \arrow[rr] & & b:\{b,c\} \arrow[dl, leftrightarrow] \\ & c:\{b,c\} & \end{tikzcd}$$

$$\Rightarrow ab \Rightarrow \quad \begin{tikzcd} a:\{a,b,c\} \arrow[rr, leftrightarrow] \arrow[dr] & & b:\{a,b,c\} \arrow[dl, leftrightarrow] \\ & c:\{b,c\} & \end{tikzcd}$$

$$\Rightarrow ac \Rightarrow \quad \begin{tikzcd} a:\{a,b,c\} \arrow[rr, leftrightarrow] \arrow[dr, leftrightarrow] & & b:\{a,b,c\} \arrow[dl, leftrightarrow] \\ & c:\{a,b,c\} & \end{tikzcd}$$

Old Questions, New Answers

In some cases, the dynamics can speed up the calling.

Old Questions, New Answers

In some cases, the dynamics can speed up the calling.

Consider a circle with five agents

$$a \longrightarrow b \longrightarrow c \longrightarrow d \longrightarrow e \longrightarrow a$$

Old Questions, New Answers

In some cases, the dynamics can speed up the calling.

Consider a circle with five agents

$$a \longrightarrow b \longrightarrow c \longrightarrow d \longrightarrow e \longrightarrow a$$

This needs $2n-3 = 7$ calls before everyone knows all secrets [HHL88]. A sequence that works is $ab; cd; ea; de; ea; ab; bc$ (check at your leasure).

Old Questions, New Answers

In some cases, the dynamics can speed up the calling.

Consider a circle with five agents

$$a \longrightarrow b \longrightarrow c \longrightarrow d \longrightarrow e \longrightarrow a$$

This needs $2n-3 = 7$ calls before everyone knows all secrets [HHL88]. A sequence that works is $ab; cd; ea; de; ea; ab; bc$ (check at your leasure).

In our dynamic approach 6 calls are sufficient: $ab; cd; ea; de; ac; bc$.

Old Questions, New Answers

In some cases, the dynamics can speed up the calling.

Consider a circle with five agents

$$a \longrightarrow b \longrightarrow c \longrightarrow d \longrightarrow e \longrightarrow a$$

This needs $2n-3 = 7$ calls before everyone knows all secrets [HHL88]. A sequence that works is $ab; cd; ea; de; ea; ab; bc$ (check at your leasure).

In our dynamic approach 6 calls are sufficient: $ab; cd; ea; de; ac; bc$.

Note: the fifth call ac in this sequence only works because a has learned the contact information about c from b.

Old Questions, New Answers

In some cases, the dynamics can speed up the calling.

Consider a circle with five agents

$$a \longrightarrow b \longrightarrow c \longrightarrow d \longrightarrow e \longrightarrow a$$

This needs $2n-3 = 7$ calls before everyone knows all secrets [HHL88]. A sequence that works is $ab; cd; ea; de; ea; ab; bc$ (check at your leasure).

In our dynamic approach 6 calls are sufficient: $ab; cd; ea; de; ac; bc$.

Note: the fifth call ac in this sequence only works because a has learned the contact information about c from b.

This shows that old questions about minimum lengths of calling sequences can receive new answers in this dynamic setting.

Gossip and Weakly Connected Components

Proposition 1 *Let* $G = (A, N, S)$*, and let* σ *be a possible calling sequence for* G*. Then* $N^{\sigma} \subseteq (N \cup N^{-1})^{*}$.

Intuitively, this says that gossip can only spread within the weakly connected components of G.

Gossip and Weakly Connected Components

Proposition 1 *Let* $G = (A, N, S)$, *and let* σ *be a possible calling sequence for* G. *Then* $N^\sigma \subseteq (N \cup N^{-1})^*$.

Intuitively, this says that gossip can only spread within the weakly connected components of G.

A gossip graph $G = (A, N, S)$ is **weakly connected** if for all $x, y \in A$ there is an $N \cup N^{-1}$-path from x to y.

Gossip and Weakly Connected Components

Proposition 1 *Let $G = (A, N, S)$, and let σ be a possible calling sequence for G. Then $N^\sigma \subseteq (N \cup N^{-1})^*$.*

Intuitively, this says that gossip can only spread within the weakly connected components of G.

A gossip graph $G = (A, N, S)$ is **weakly connected** if for all $x, y \in A$ there is an $N \cup N^{-1}$-path from x to y.

Theorem 2 *If σ is a possible calling sequence for $G = (A, N, S)$, then G is weakly connected iff G^σ is weakly connected.*

Intuition: by proposition 1, gossip cannot create weak connectedness.

Gossip Graph Completion

A gossip graph $G = (A, N, S)$ is complete if it holds for all $x \in A$ that $S_x = A$. That is, a gossip graph is complete if all agents know all secrets.

Gossip Graph Completion

A gossip graph $G = (A, N, S)$ is complete if it holds for all $x \in A$ that $S_x = A$. That is, a gossip graph is complete if all agents know all secrets.

Key question 1: find distributed protocols that always create complete graphs.

Precondition: the graph has to be weakly connected to start with, by Theorem 2.

Gossip Graph Completion

A gossip graph $G = (A, N, S)$ is complete if it holds for all $x \in A$ that $S_x = A$. That is, a gossip graph is complete if all agents know all secrets.

Key question 1: find distributed protocols that always create complete graphs.

Precondition: the graph has to be weakly connected to start with, by Theorem 2.

Key question 2: given some distributed protocol P, what is the class of graphs that can be completed by P?

Search for Secrets as a Distributed Protocol

Search For Secrets

While not every agent knows all secrets, randomly select a pair xy such that Nxy and let x call y.

This will complete any weakly connected graph, but it is not efficient.

Learn New Secrets

The following protocol is studied in [AvDGvdH14, AGvdH15] in the context of totally connected graphs.

> **Learn New Secret Protocol (LNS)**
>
> While not every agent is an expert, let an agent x that is not an expert randomly choose an agent y from the list of agents for which Nxy but not Sxy, and perform the call xy.

LNS-permitted and LNS-stuck Sequences

LNS-permitted calling sequences:

- ϵ is LNS-permitted on any G,
- $\sigma; xy$ is LNS-permitted on G iff σ is LNS-permitted on G and xy is LNS-permitted on G^{σ}.

LNS-permitted and LNS-stuck Sequences

LNS-permitted calling sequences:

- ϵ is LNS-permitted on any G,
- $\sigma; xy$ is LNS-permitted on G iff σ is LNS-permitted on G and xy is LNS-permitted on G^σ.

A calling sequence σ is **LNS-stuck** on G if σ is LNS-permitted on G, G^σ is not complete, and no call is LNS-permitted on G^σ.

Example of LNS-stuck Sequence

$$\begin{tikzcd}
a : \{a\} \arrow[rr] & & b : \{b\} \arrow[dl] \\
& c : \{c\} &
\end{tikzcd}
\quad \Rightarrow bc \Rightarrow \quad
\begin{tikzcd}
a : \{a\} \arrow[rr] & & b : \{b, c\} \arrow[dl, leftrightarrow] \\
& c : \{b, c\} &
\end{tikzcd}$$

Example of LNS-stuck Sequence

$$\begin{tikzcd}
a:\{a\} \arrow[rr] & & b:\{b\} \arrow[dl] \\
& c:\{c\} &
\end{tikzcd}
\quad \Rightarrow bc \Rightarrow \quad
\begin{tikzcd}
a:\{a\} \arrow[rr] & & b:\{b,c\} \arrow[dl, leftrightarrow] \\
& c:\{b,c\} &
\end{tikzcd}$$

$$\Rightarrow ab \Rightarrow \quad
\begin{tikzcd}
a:\{a,b,c\} \arrow[rr, leftrightarrow] \arrow[dr] & & b:\{a,b,c\} \arrow[dl, leftrightarrow] \\
& c:\{b,c\} &
\end{tikzcd}$$

Example of LNS-stuck Sequence

$$
\begin{tikzcd}
a:\{a\} \arrow[rr] & & b:\{b\} \arrow[dl] \\
& c:\{c\} &
\end{tikzcd}
\quad \Rightarrow bc \Rightarrow \quad
\begin{tikzcd}
a:\{a\} \arrow[rr] & & b:\{b,c\} \arrow[dl, leftrightarrow] \\
& c:\{b,c\} &
\end{tikzcd}
$$

$$
\Rightarrow ab \Rightarrow \quad
\begin{tikzcd}
a:\{a,b,c\} \arrow[rr, leftrightarrow] \arrow[dr] & & b:\{a,b,c\} \arrow[dl, leftrightarrow] \\
& c:\{b,c\} &
\end{tikzcd}
$$

$$\Rightarrow ?? \Rightarrow \text{STUCK}$$

Example of LNS-stuck Sequence

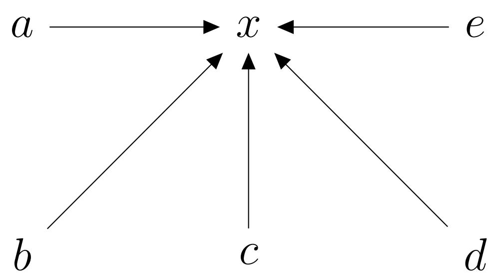
$$\begin{tikzcd} a:\{a\} \arrow[rr] & & b:\{b\} \arrow[dl] \\ & c:\{c\} & \end{tikzcd} \Rightarrow bc \Rightarrow \begin{tikzcd} a:\{a\} \arrow[rr] & & b:\{b,c\} \arrow[dl, leftrightarrow] \\ & c:\{b,c\} & \end{tikzcd}$$

$$\Rightarrow ab \Rightarrow \begin{tikzcd} a:\{a,b,c\} \arrow[rr, leftrightarrow] \arrow[dr] & & b:\{a,b,c\} \arrow[dl, leftrightarrow] \\ & c:\{b,c\} & \end{tikzcd}$$

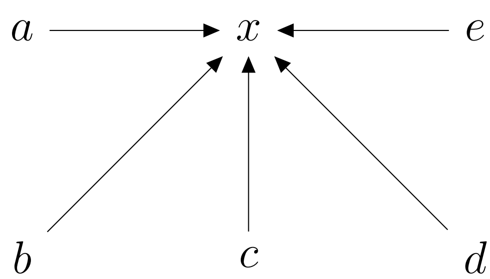
$$\Rightarrow ?? \Rightarrow \text{STUCK}$$

Still, the sequence $ab; bc; ca$ is LNS permitted.

Example Graph that Cannot be Completed by LNS



Example Graph that Cannot be Completed by LNS



Whoever starts the calling sequence, it can not be LNS completed.

The reason is that x, the spider in the web, never learns enough about the network structure to be able to make a useful call (x only learns contact info of agents whose secret x also learns.

The LNS-permitted sequences are all the permutations of $ax; bx; cx; dx; ex$, and they all get stuck.

Graphs Where LNS is Successful

The LNS protocol is successful on G if either G is complete, or there is an LNS-permitted call xy, and after any LNS-permitted call xy the LNS protocol is successful on G^{xy}.

It follows that LNS is successful on G iff every sequence of LNS-permitted calls σ results in a graph G^{σ} that is complete, or is such that there is an LNS-permitted call, and after any LNS-permitted call xy, LNS is successful on $G^{\sigma;xy}$.

> **LNS gossip graph algorithm**
>
> Search for an LNS-stuck calling sequence in depth-first fashion, and declare success if no such calling sequence can be found [EG15a].

LNS-maximal Sequences

A calling sequence σ for G is LNS-maximal if σ is LNS-permitted for G, and no calls are LNS-permitted in G^σ.

LNS-maximal Sequences

A calling sequence σ for G is LNS-maximal if σ is LNS-permitted for G, and no calls are LNS-permitted in G^σ.

Proposition 3 *If σ is an LNS-maximal calling sequence for G, and G has accessible secrets, then $S^\sigma = N^\sigma$.*

LNS-maximal Sequences

A calling sequence σ for G is LNS-maximal if σ is LNS-permitted for G, and no calls are LNS-permitted in G^{σ}.

Proposition 3 *If σ is an LNS-maximal calling sequence for G, and G has accessible secrets, then $S^{\sigma} = N^{\sigma}$.*

Proposition 4 *If σ is an LNS-maximal calling sequence for G, and G satisfies $I_A = S \subseteq N$, then $S^{\sigma} \circ N^{*} = S^{\sigma}$.*

Terminal Points, Skin, Sun Graphs

A **terminal point** in $G = (A, N, S)$ is a point x for which $N_x \subseteq \{x\}$. That is, a terminal point is an agent that knows at most her own number.

Terminal Points, Skin, Sun Graphs

A **terminal point** in $G = (A, N, S)$ is a point x for which $N_x \subseteq \{x\}$. That is, a terminal point is an agent that knows at most her own number.

The **skin** of a graph $G = (A, N, S)$ is the set $\{x \in A \mid N_x \subseteq \{x\}\}$ (the set of terminal points).

Terminal Points, Skin, Sun Graphs

A **terminal point** in $G = (A, N, S)$ is a point x for which $N_x \subseteq \{x\}$. That is, a terminal point is an agent that knows at most her own number.

The **skin** of a graph $G = (A, N, S)$ is the set $\{x \in A \mid N_x \subseteq \{x\}\}$ (the set of terminal points).

Let $s(G)$ be the result of skinning graph G, i.e. removing all terminal points from G. That is, $s(G) = (B, N', S')$ where

$$B = \{x \in A \mid N_x - \{x\} \neq \emptyset\}, N' = N \cap B^2, S' = S \cap B^2.$$

Note that skinning a graph is not a closure operation: there are graphs with $s(s(G)) \neq s(G)$.

Terminal Points, Skin, Sun Graphs

A **terminal point** in $G = (A, N, S)$ is a point x for which $N_x \subseteq \{x\}$. That is, a terminal point is an agent that knows at most her own number.

The **skin** of a graph $G = (A, N, S)$ is the set $\{x \in A \mid N_x \subseteq \{x\}\}$ (the set of terminal points).

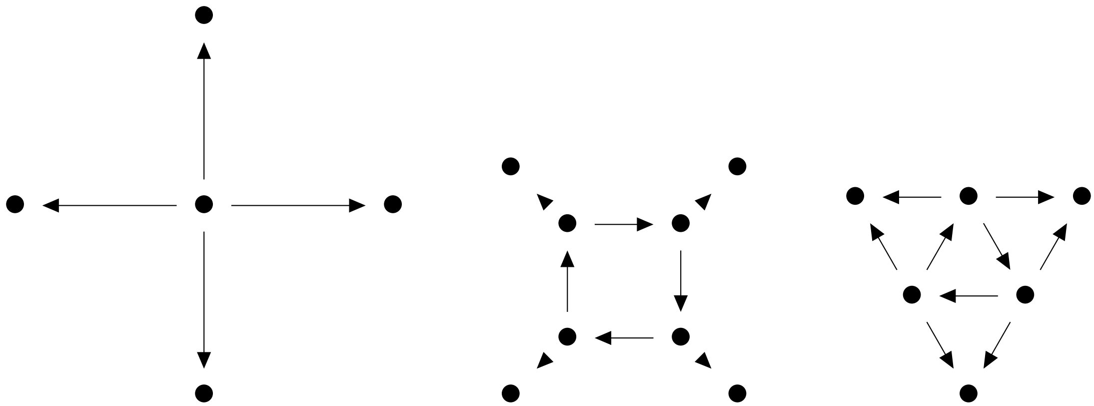
Let $s(G)$ be the result of skinning graph G, i.e. removing all terminal points from G. That is, $s(G) = (B, N', S')$ where

$$B = \{x \in A \mid N_x - \{x\} \neq \emptyset\}, N' = N \cap B^2, S' = S \cap B^2.$$

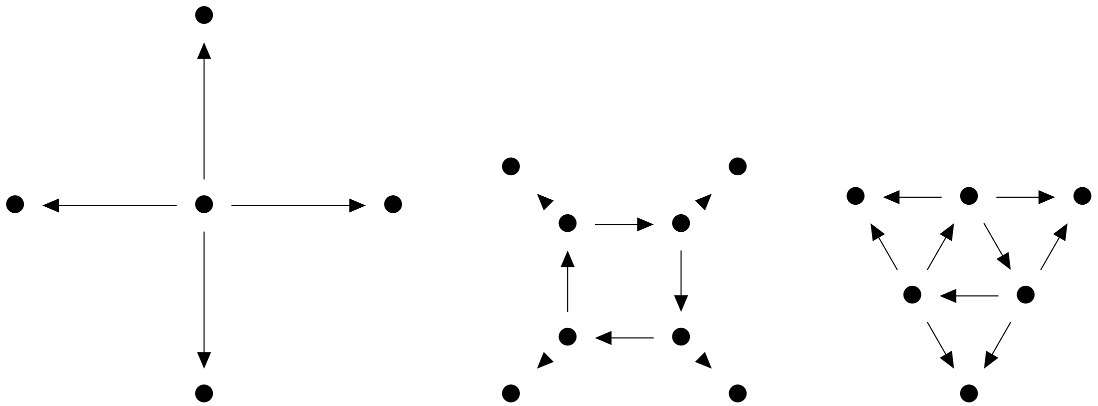
Note that skinning a graph is not a closure operation: there are graphs with $s(s(G)) \neq s(G)$.

Call a graph $G = (A, N, S)$ a **sun** if $S = I_A \subseteq N$, N is weakly connected on G, and N is strongly connected on $s(G)$.

Examples of Sun Graphs



Examples of Sun Graphs



Theorem 5 *The LNS protocol is successful for any sun G.*

Strongly Connected Components of a Graph

Let $\sim$ be the relation on $G = (A, N, S)$ given by $x \sim y$ iff there is an N-path from x to y and there is an N-path from y to x. Then $\sim$ is an equivalence relation, and a cell in the partition induced by $\sim$ is called a strongly connected component of G. Use $[x]_\sim$ for the strongly connected component of G that contains x.

Strongly Connected Components of a Graph

Let $\sim$ be the relation on $G = (A, N, S)$ given by $x \sim y$ iff there is an N-path from x to y and there is an N-path from y to x. Then $\sim$ is an equivalence relation, and a cell in the partition induced by $\sim$ is called a strongly connected component of G. Use $[x]_\sim$ for the strongly connected component of G that contains x.

Notice that a gossip graph G is strongly connected iff $\sim$ is universal on G.

Strongly Connected Components of a Graph

Let $\sim$ be the relation on $G = (A, N, S)$ given by $x \sim y$ iff there is an N-path from x to y and there is an N-path from y to x. Then $\sim$ is an equivalence relation, and a cell in the partition induced by $\sim$ is called a strongly connected component of G. Use $[x]_\sim$ for the strongly connected component of G that contains x.

Notice that a gossip graph G is strongly connected iff $\sim$ is universal on G.

If $G = (A, N, S)$ is a gossip graph and σ is a possible calling sequence for G, then we use $[x]^\sigma_\sim$ for the strongly connected component of G^σ that contains x.

Strongly Connected Components of a Graph

Let $\sim$ be the relation on $G = (A, N, S)$ given by $x \sim y$ iff there is an N-path from x to y and there is an N-path from y to x. Then $\sim$ is an equivalence relation, and a cell in the partition induced by $\sim$ is called a strongly connected component of G. Use $[x]_\sim$ for the strongly connected component of G that contains x.

Notice that a gossip graph G is strongly connected iff $\sim$ is universal on G.

If $G = (A, N, S)$ is a gossip graph and σ is a possible calling sequence for G, then we use $[x]^\sigma_\sim$ for the strongly connected component of G^σ that contains x.

If $G = (A, N, S)$, then the relation $\hat{N}$ on A is defined by means of:

$$\hat{N}xy \text{ iff } [x]_\sim \neq [y]_\sim \wedge \exists x' \in [x]_\sim \exists y' \in [y]_\sim : Nx'y'.$$

Strongly Connected Components of a Graph

Let $\sim$ be the relation on $G = (A, N, S)$ given by $x \sim y$ iff there is an N-path from x to y and there is an N-path from y to x. Then $\sim$ is an equivalence relation, and a cell in the partition induced by $\sim$ is called a strongly connected component of G. Use $[x]_\sim$ for the strongly connected component of G that contains x.

Notice that a gossip graph G is strongly connected iff $\sim$ is universal on G.

If $G = (A, N, S)$ is a gossip graph and σ is a possible calling sequence for G, then we use $[x]^\sigma_\sim$ for the strongly connected component of G^σ that contains x.

If $G = (A, N, S)$, then the relation $\hat{N}$ on A is defined by means of:

$$\hat{N}xy \text{ iff } [x]_\sim \neq [y]_\sim \wedge \exists x' \in [x]_\sim \exists y' \in [y]_\sim : Nx'y'.$$

Theorem 6 *If* $G = (A, N, S)$ *is a gossip graph with the property that* $s(G)$ *is connected but not strongly connected, then there are* $x, y \in A$ *such that*

1. $\hat{N}xy$*,*
2. *for all* $u \in A$ *with* $\hat{N}xu$ *and* N^*uy *it holds that* $[u]_\sim = [y]_\sim$*,*
3. *if* $[y]_\sim = \{y\}$ *then there is a* $z \in A$ *that is terminal in* G *with* Nyz*.*

Theorem 6 *If $G = (A, N, S)$ is a gossip graph with the property that $s(G)$ is connected but not strongly connected, then there are $x, y \in A$ such that*

1. $\hat{N}xy$,
2. *for all $u \in A$ with $\hat{N}xu$ and N^*uy it holds that $[u]_\sim = [y]_\sim$,*
3. *if $[y]_\sim = \{y\}$ then there is a $z \in A$ that is terminal in G with Nyz.*

Theorem 7 *For any gossip graph $G = (A, N, S)$ with $S = I_A \subseteq N$ and the property that $s(G)$ is connected but not strongly connected there is an LNS-permitted calling sequence σ such that G^σ is not complete, but no calls are LNS-permitted in G^σ.*

Theorem 6 *If $G = (A, N, S)$ is a gossip graph with the property that $s(G)$ is connected but not strongly connected, then there are $x, y \in A$ such that*

1. $\hat{N}xy$,
2. *for all $u \in A$ with $\hat{N}xu$ and N^*uy it holds that $[u]_\sim = [y]_\sim$,*
3. *if $[y]_\sim = \{y\}$ then there is a $z \in A$ that is terminal in G with Nyz.*

Theorem 7 *For any gossip graph $G = (A, N, S)$ with $S = I_A \subseteq N$ and the property that $s(G)$ is connected but not strongly connected there is an LNS-permitted calling sequence σ such that G^σ is not complete, but no calls are LNS-permitted in G^σ.*

Theorem 8 *For any connected graph $G = (A, N, S)$ with $I_A = S \subseteq N$ the following holds: $s(G)$ is strongly connected iff the LNS protocol is successful for G.*

LNS-impossible Graphs

Question Characterize the graphs where **no** CNS sequence is successful.

Answer See [DvEP+15a].

Note: the conditions for this are quite strong. Spider in the web above is an example.

Other Protocols: HYN

Help Your Neighbour (HYN) [Her15]

Everyone who ever contacted you becomes your neighbour. Every time you learn new secrets, you incur an obligation to inform them …

Theorem 9 *HYN completes any weakly connected graph. [Her15]*

Further Questions

Further Questions

- Give a survey of results about minimal calling sequences for dynamic networks with various properties, in the spirit of [HHL88].

Further Questions

- Give a survey of results about minimal calling sequences for dynamic networks with various properties, in the spirit of [HHL88].
- [DvEP+15a] investigates a number of different distributed protocols. Give a systematic classification.

Further Questions

- Give a survey of results about minimal calling sequences for dynamic networks with various properties, in the spirit of [HHL88].
- [DvEP+15a] investigates a number of different distributed protocols. Give a systematic classification.
- In order to implement knowledge bases gossip protocols (“tell the secret of x to y if you hold it for possible that y does not yet know”) one needs a notion of knowledge [Eij15, Efe15].

Further Questions

- Give a survey of results about minimal calling sequences for dynamic networks with various properties, in the spirit of [HHL88].
- [DvEP+15a] investigates a number of different distributed protocols. Give a systematic classification.
- In order to implement knowledge bases gossip protocols ("tell the secret of x to y if you hold it for possible that y does not yet know") one needs a notion of knowledge [Eij15, Efe15].
- Further features from actual life could be imported. An important one that comes to mind is caller blocking. Which blocking patterns on which graphs can be overcome by which protocols?

Further Questions

- Give a survey of results about minimal calling sequences for dynamic networks with various properties, in the spirit of [HHL88].
- [DvEP+15a] investigates a number of different distributed protocols. Give a systematic classification.
- In order to implement knowledge bases gossip protocols ("tell the secret of x to y if you hold it for possible that y does not yet know") one needs a notion of knowledge [Eij15, Efe15].
- Further features from actual life could be imported. An important one that comes to mind is caller blocking. Which blocking patterns on which graphs can be overcome by which protocols?
- In [EG15b] we try to harness PDL as a logic for dynamic gossip. Details will be given by Malvin in the next talk.


2015
Mark Zuckerberg


facebook, a great place to
spread gossip

References

[AGvdH15] Krzyzstof R. Apt, Davide Grossi, and Wiebe van der Hoek. Epistemic protocols for distributed gossiping. In Proceedings of TARK 2015, 2015.

[AvDGvdH14] M. Attamah, H. van Ditmarsch, D. Grossi, and W. van der Hoek. Knowledge and gossip. In Proc. of 21st ECAI, pages 21–26. IOS Press, 2014.

[DvEP+15a] Hans Ditmarsch, Jan van Eijck, Pere Pardo, Rahim Ramezanian, and François Schwarzentruber. Dynamic gossip. Work in Progress, October 2015.

[DvEP+15b] Hans Ditmarsch, Jan van Eijck, Pere Pardo, Rahim Ramezanian, and François Schwarzentruber. Gossip in dynamic networks. Under Submission, 2015.

[Efe15] Zarina Efendijeva. Comparisons of definitions of knowledge for epistemic gossip protocols. Master's thesis, Software Engineering, University of Amsterdam, August 2015.

[EG15a] Jan van Eijck and Malvin Gattinger. Gossip. Technical report, CWI, Amsterdam, 2015.

[EG15b] Jan van Eijck and Malvin Gattinger. Propositional dynamic gossip logic. Technical report, CWI, Amsterdam, 2015.

[EGKM04] Patrick Th. Eugster, Rachid Guerraoui, Anne-Marie Kermarrec, and Laurent Massoulié. Epidemic information dissemination in distributed systems. IEEE Computer, 37(5):60–67, 2004.

[Eij15] Jan van Eijck. Accessibilities. Technical report, CWI, Amsterdam, July 2015.

[Her15] Clément Herouard. More gossip in dynamic networks. Technical report, CWI, Amsterdam, August 2015.

[HHL88] Sandra Mitchell Hedetniemi, Stephen T. Hedetniemi, and Arthur L. Liestman. A survey of gossiping and broadcasting in communication networks. Networks, 18(4):319–349, 1988.

[Hur00] C. A. J. Hurkens. Spreading gossip efficiently. NAW, 5(1):208–210, 2000.

[Tij71] R. Tijdeman. On a telephone problem. Nieuw Archief voor Wiskunde, 3(19):188–192, 1971.

[WSvE10] Yanjing Wang, Floor Sietsma, and Jan van Eijck. Logic of information flow on communication channels (extended abstract). In Proceedings of AAMAS-2010, the 9th International Conference on Autonomous Agents and Multiagent Systems, pages 1447–1448, Toronto, May 2010.