# Ant routing for the Lightning Network (with C. Grunspan)

Ricardo Pérez-Marco

@rperezmarco
webusers.imj-prg.fr/∼ricardo.perez-marco

CNRS, IMJ-PRG, Univ. Paris 7

Building on Bitcoin

Lisbon

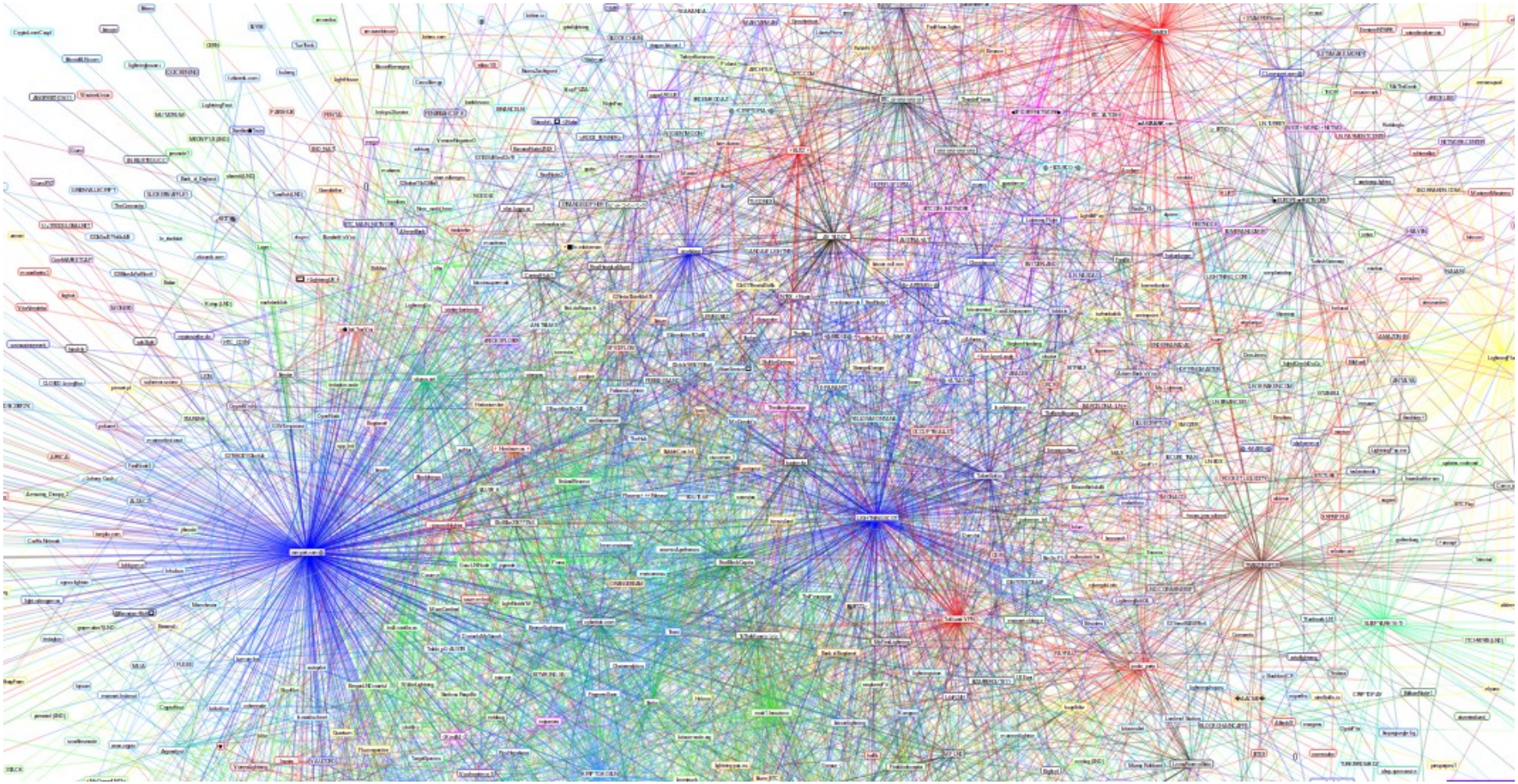
ArXiv:1807.00151 (July 2018)

# Ant routing for the Lightning Network

# Ant routing for the Lightning Network

# The Lightning Network.

# The Lightning Network.



# Liberté, Égalité, Diversité, Vérificabilité

# Liberté, Égalité, Diversité, Vérificabilité

Necessary conditions for decentralization:

# Liberté, Égalité, Diversité, Vérificabilité

Necessary conditions for decentralization:

- Open and affordable access to the network. Liberté

# Liberté, Égalité, Diversité, Vérificabilité

Necessary conditions for decentralization:

- Open and affordable access to the network. Liberté
- Nodes can process the information they want. Liberté

# Liberté, Égalité, Diversité, Vérificabilité

Necessary conditions for decentralization:

- Open and affordable access to the network. Liberté
- Nodes can process the information they want. Liberté
- Nodes have access to the same information. Égalité

# Liberté, Égalité, Diversité, Vérificabilité

Necessary conditions for decentralization:

- Open and affordable access to the network. Liberté
- Nodes can process the information they want. Liberté
- Nodes have access to the same information. Égalité
- Nodes have the same power. Égalité

# Liberté, Égalité, Diversité, Vérificabilité

Necessary conditions for decentralization:

- Open and affordable access to the network. Liberté
- Nodes can process the information they want. Liberté
- Nodes have access to the same information. Égalité
- Nodes have the same power. Égalité
- Nodes follow the same protocol. Égalité

# Liberté, Égalité, Diversité, Vérificabilité

Necessary conditions for decentralization:

- Open and affordable access to the network. Liberté
- Nodes can process the information they want. Liberté
- Nodes have access to the same information. Égalité
- Nodes have the same power. Égalité
- Nodes follow the same protocol. Égalité
- Ownership of nodes is well distributed. Diversité

# Liberté, Égalité, Diversité, Vérificabilité

Necessary conditions for decentralization:

- Open and affordable access to the network. Liberté
- Nodes can process the information they want. Liberté
- Nodes have access to the same information. Égalité
- Nodes have the same power. Égalité
- Nodes follow the same protocol. Égalité
- Ownership of nodes is well distributed. Diversité
- Richly connected (thousands nodes). Diversité

# Liberté, Égalité, Diversité, Vérificabilité

Necessary conditions for decentralization:

- Open and affordable access to the network. Liberté
- Nodes can process the information they want. Liberté
- Nodes have access to the same information. Égalité
- Nodes have the same power. Égalité
- Nodes follow the same protocol. Égalité
- Ownership of nodes is well distributed. Diversité
- Richly connected (thousands nodes). Diversité
- Randomly connected. Diversité

# Liberté, Égalité, Diversité, Vérificabilité

Necessary conditions for decentralization:

- Open and affordable access to the network. Liberté
- Nodes can process the information they want. Liberté
- Nodes have access to the same information. Égalité
- Nodes have the same power. Égalité
- Nodes follow the same protocol. Égalité
- Ownership of nodes is well distributed. Diversité
- Richly connected (thousands nodes). Diversité
- Randomly connected. Diversité
- All nodes check the information shared. Vérificabilité

# Liberté, Égalité, Diversité, Vérificabilité

Necessary conditions for decentralization:

- Open and affordable access to the network. Liberté
- Nodes can process the information they want. Liberté
- Nodes have access to the same information. Égalité
- Nodes have the same power. Égalité
- Nodes follow the same protocol. Égalité
- Ownership of nodes is well distributed. Diversité
- Richly connected (thousands nodes). Diversité
- Randomly connected. Diversité
- All nodes check the information shared. Vérificabilité
- Nobody trusts anyone, everyone verifies. Vérificabilité

# Payment channels

# Payment channels

- Payment channels: Allow off-chain transactions.

# Payment channels

- Payment channels: Allow off-chain transactions.
- One Initial Commitment Transaction and one Settlement Transaction are the only on-chain transactions.

# Payment channels

- Payment channels: Allow off-chain transactions.
- One Initial Commitment Transaction and one Settlement Transaction are the only on-chain transactions.
- Unidirectional or bidirectional payment channels.

# Payment channels

- Payment channels: Allow off-chain transactions.
- One Initial Commitment Transaction and one Settlement Transaction are the only on-chain transactions.
- Unidirectional or bidirectional payment channels.

**Constraints and properties**

# Payment channels

- Payment channels: Allow off-chain transactions.
- One Initial Commitment Transaction and one Settlement Transaction are the only on-chain transactions.
- Unidirectional or bidirectional payment channels.

**Constraints and properties**

- Maximal volume.

# Payment channels

- Payment channels: Allow off-chain transactions.
- One Initial Commitment Transaction and one Settlement Transaction are the only on-chain transactions.
- Unidirectional or bidirectional payment channels.

**Constraints and properties**

- Maximal volume.
- Instantaneous and anonymous (unrecorded) transactions.

# Payment channels

- Payment channels: Allow off-chain transactions.
- One Initial Commitment Transaction and one Settlement Transaction are the only on-chain transactions.
- Unidirectional or bidirectional payment channels.

**Constraints and properties**

- Maximal volume.
- Instantaneous and anonymous (unrecorded) transactions.
- Payment channels are composable (transitive property).

# Payment channels

- Payment channels: Allow off-chain transactions.
- One Initial Commitment Transaction and one Settlement Transaction are the only on-chain transactions.
- Unidirectional or bidirectional payment channels.

**Constraints and properties**

- Maximal volume.
- Instantaneous and anonymous (unrecorded) transactions.
- Payment channels are composable (transitive property).

If Alice and Bob have a payment channel, and Bob and Charles have another, then Alice can pay Charles through Bob.

# Payment channels

- Payment channels: Allow off-chain transactions.
- One Initial Commitment Transaction and one Settlement Transaction are the only on-chain transactions.
- Unidirectional or bidirectional payment channels.

**Constraints and properties**

- Maximal volume.
- Instantaneous and anonymous (unrecorded) transactions.
- Payment channels are composable (transitive property).

If Alice and Bob have a payment channel, and Bob and Charles have another, then Alice can pay Charles through Bob.

- Fee incentive for intermediaries.

# Payment channels

- Payment channels: Allow off-chain transactions.
- One Initial Commitment Transaction and one Settlement Transaction are the only on-chain transactions.
- Unidirectional or bidirectional payment channels.

**Constraints and properties**

- Maximal volume.
- Instantaneous and anonymous (unrecorded) transactions.
- Payment channels are composable (transitive property).

If Alice and Bob have a payment channel, and Bob and Charles have another, then Alice can pay Charles through Bob.

- Fee incentive for intermediaries.

# LN set up

# LN set up

(Nakamoto, Hearn, Spilman, Decker, Wattenhofer, Dryja, Poon, Prihodko, Ostrovskyi, Sahno, Zhigulin, Russell, Osuntokun,...)

# LN set up

(Nakamoto, Hearn, Spilman, Decker, Wattenhofer, Dryja, Poon, Prihodko, Ostrovskyi, Sahno, Zhigulin, Russell, Osuntokun,...)

- LN network of payment channels (weighted oriented graph).

# LN set up

(Nakamoto, Hearn, Spilman, Decker, Wattenhofer, Dryja, Poon, Prihodko, Ostrovskyi, Sahno, Zhigulin, Russell, Osuntokun,...)

- LN network of payment channels (weighted oriented graph).
- Decentralized network: Rich and randomly connected, etc

# LN set up

(Nakamoto, Hearn, Spilman, Decker, Wattenhofer, Dryja, Poon, Prihodko, Ostrovskyi, Sahno, Zhigulin, Russell, Osuntokun,...)

- LN network of payment channels (weighted oriented graph).
- Decentralized network: Rich and randomly connected, etc
- On top of the LN network we have a richer and fast communication network.

# LN set up

(Nakamoto, Hearn, Spilman, Decker, Wattenhofer, Dryja, Poon, Prihodko, Ostrovskyi, Sahno, Zhigulin, Russell, Osuntokun,...)

- LN network of payment channels (weighted oriented graph).
- Decentralized network: Rich and randomly connected, etc
- On top of the LN network we have a richer and fast communication network.
- Nodes reserve a mempool space for routing purposes.

# LN set up

(Nakamoto, Hearn, Spilman, Decker, Wattenhofer, Dryja, Poon, Prihodko, Ostrovskyi, Sahno, Zhigulin, Russell, Osuntokun,...)

- LN network of payment channels (weighted oriented graph).
- Decentralized network: Rich and randomly connected, etc
- On top of the LN network we have a richer and fast communication network.
- Nodes reserve a mempool space for routing purposes.

Main problem: Decentralized payment path finding algorithm.

# LN set up

(Nakamoto, Hearn, Spilman, Decker, Wattenhofer, Dryja, Poon, Prihodko, Ostrovskyi, Sahno, Zhigulin, Russell, Osuntokun,...)

- LN network of payment channels (weighted oriented graph).
- Decentralized network: Rich and randomly connected, etc
- On top of the LN network we have a richer and fast communication network.
- Nodes reserve a mempool space for routing purposes.

Main problem: Decentralized payment path finding algorithm.

# Problems and difficulties

# Problems and difficulties

- Global knowledge of the geometry of the network is a vector of attack.

# Problems and difficulties

- Global knowledge of the geometry of the network is a vector of attack.
- Solutions based on “beacon nodes” with rich routing tables.

# Problems and difficulties

- Global knowledge of the geometry of the network is a vector of attack.
- Solutions based on “beacon nodes” with rich routing tables.
- Beacon nodes or supernodes violate decentralization.

# Problems and difficulties

- Global knowledge of the geometry of the network is a vector of attack.
- Solutions based on “beacon nodes” with rich routing tables.
- Beacon nodes or supernodes violate decentralization.

**Some hints:**

# Problems and difficulties

- Global knowledge of the geometry of the network is a vector of attack.
- Solutions based on “beacon nodes” with rich routing tables.
- Beacon nodes or supernodes violate decentralization.

**Some hints:**

- Bitcoin network does not use routing tables. Information (transactions) are propagated to the whole network.

# Problems and difficulties

- Global knowledge of the geometry of the network is a vector of attack.
- Solutions based on “beacon nodes” with rich routing tables.
- Beacon nodes or supernodes violate decentralization.

**Some hints:**

- Bitcoin network does not use routing tables. Information (transactions) are propagated to the whole network.
- Ant path finding algorithms are efficient and highly decentralized.

# Problems and difficulties

- Global knowledge of the geometry of the network is a vector of attack.
- Solutions based on “beacon nodes” with rich routing tables.
- Beacon nodes or supernodes violate decentralization.

**Some hints:**

- Bitcoin network does not use routing tables. Information (transactions) are propagated to the whole network.
- Ant path finding algorithms are efficient and highly decentralized.

# Ant paths

# Ant paths

# Goss et al. (1989)

# Goss et al. (1989)

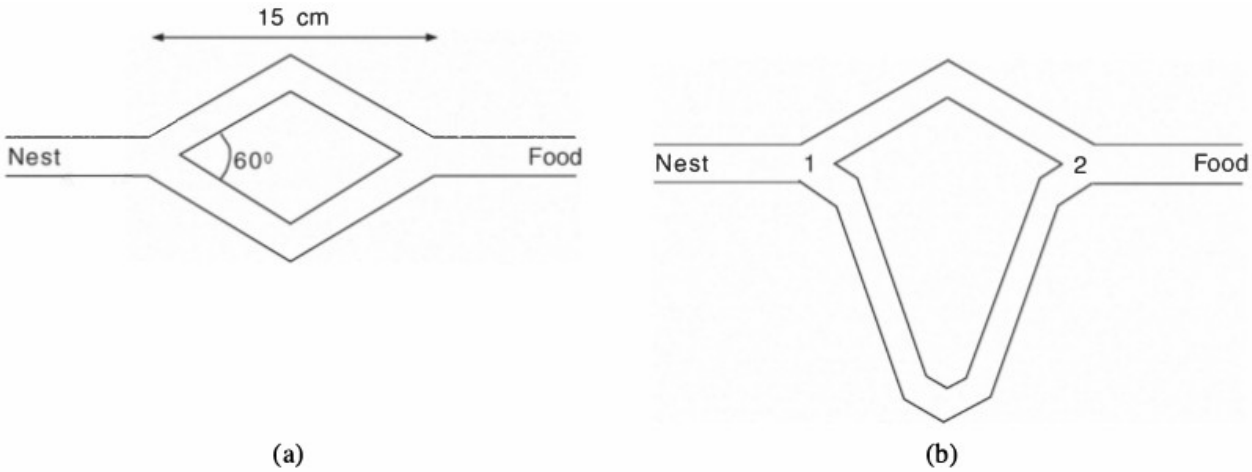
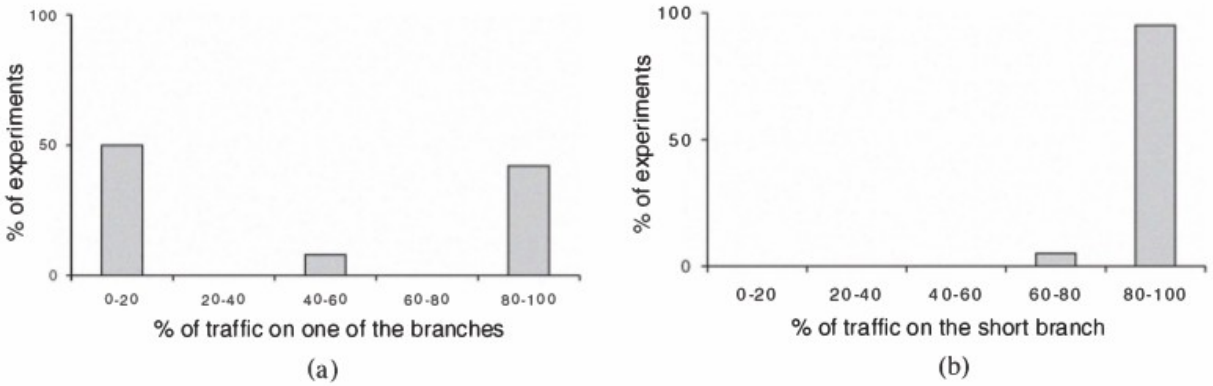
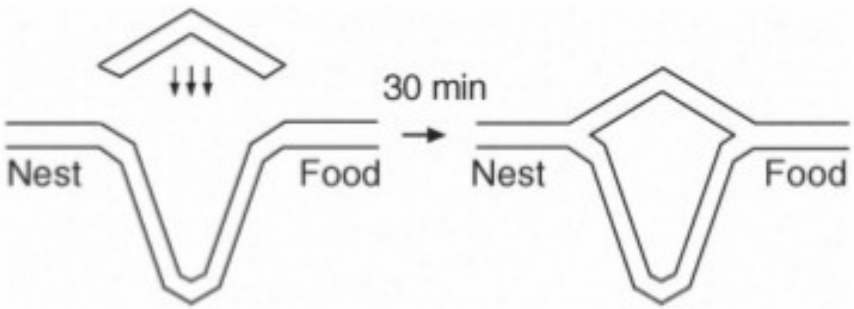


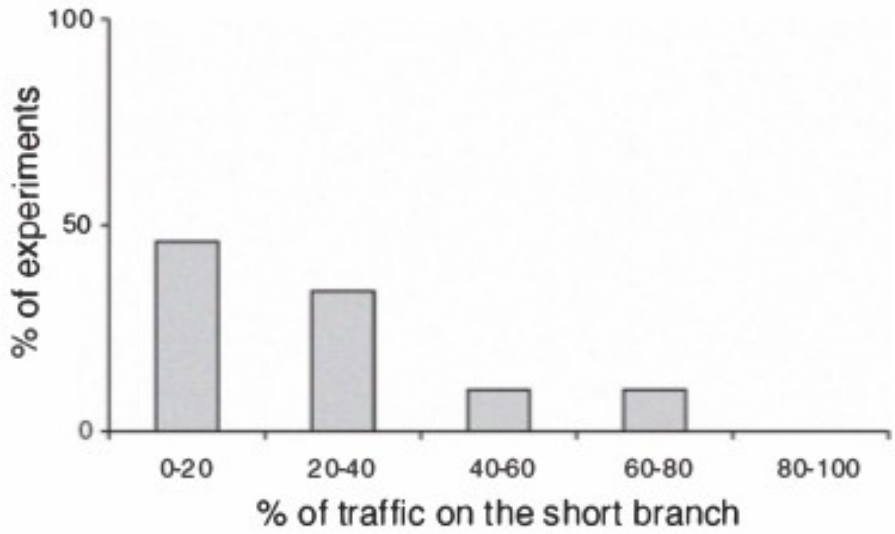

**Figure 1.1**
Experimental setup for the double bridge experiment. (a) Branches have equal length. (b) Branches have different length. Modified from Goss et al. (1989).



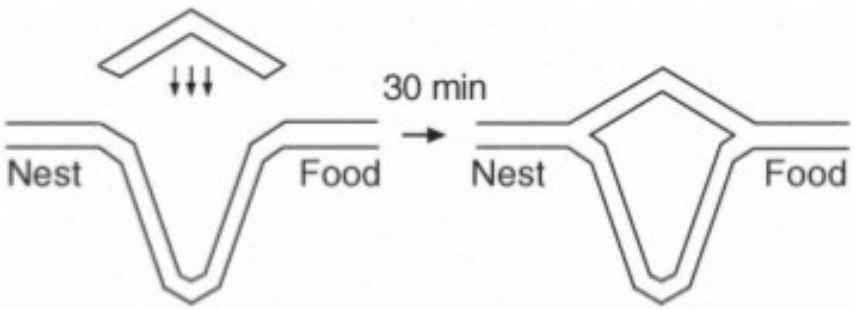
# Goss et al. (1989)

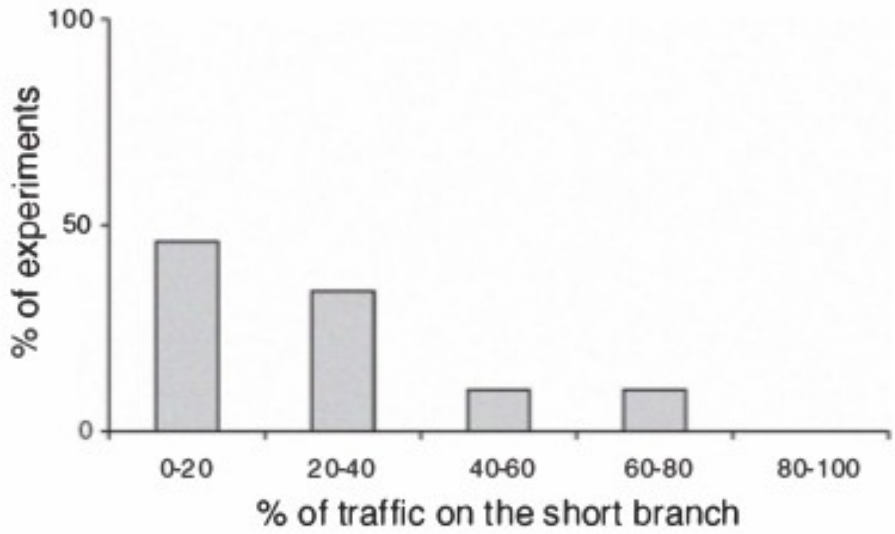
# Goss et al. (1989)



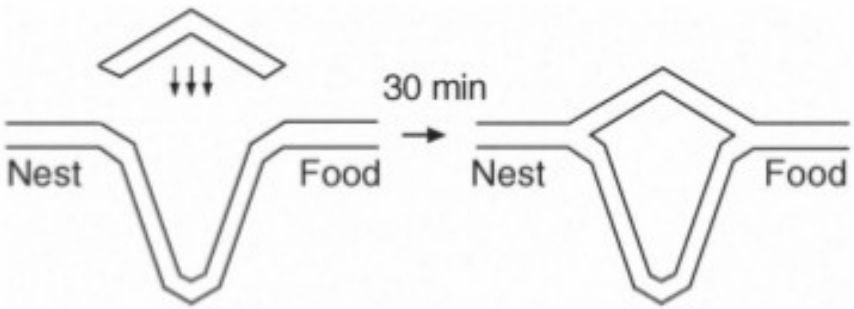
# Goss et al. (1989)

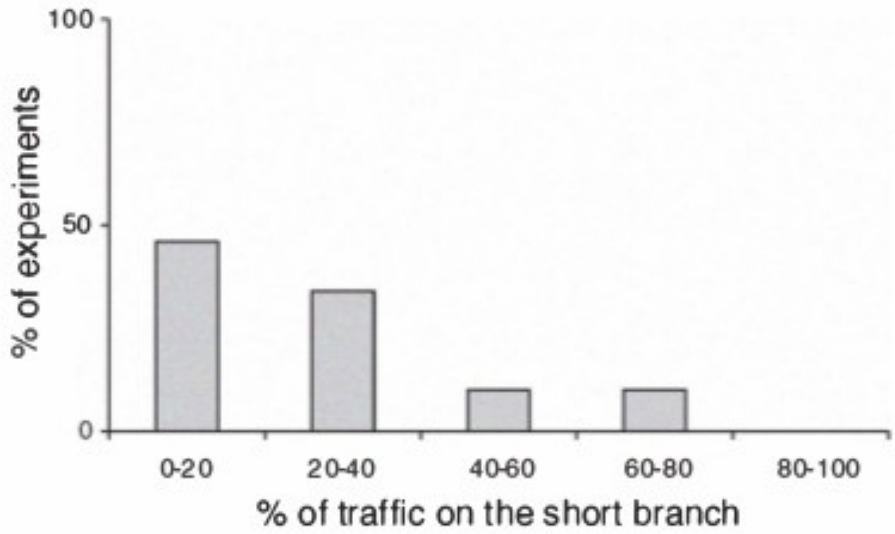



- Marking paths with pheromones.

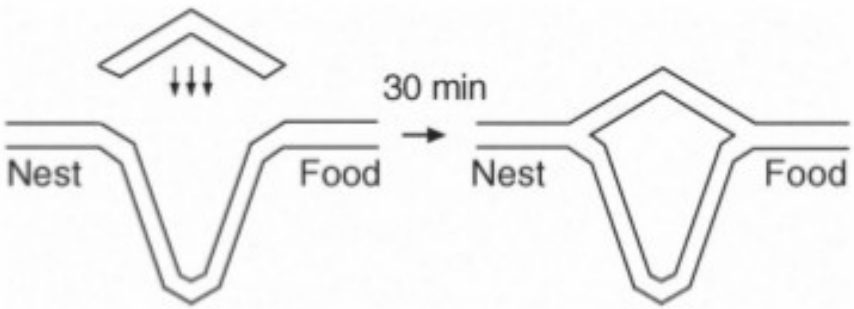
# Goss et al. (1989)

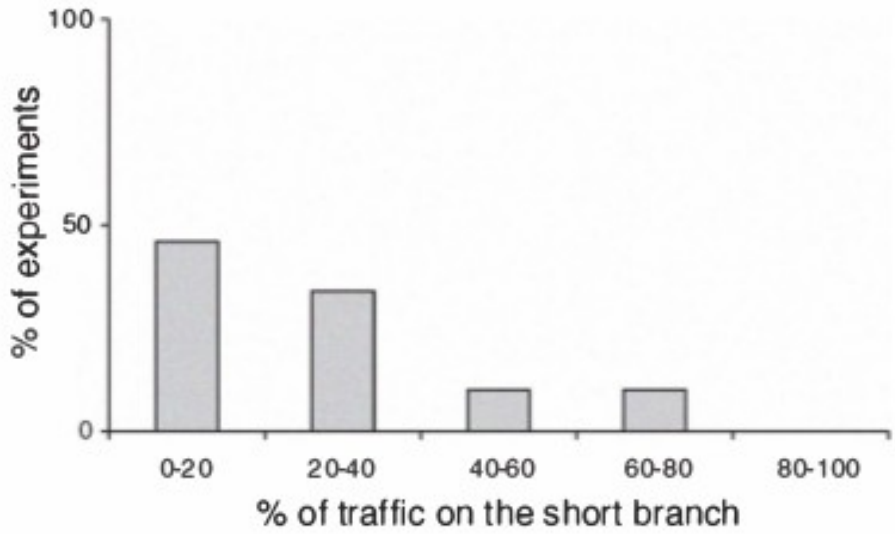



- Marking paths with pheromones.
- Reinforcing paths with pheromones.

# Goss et al. (1989)




- Marking paths with pheromones.
- Reinforcing paths with pheromones.

# Ariadne's thread

# Ariadne's thread

# Pheromone seeds

# Pheromone seeds

- Alice wants to pay Bob. They agree on a common random number $R$.

# Pheromone seeds

- Alice wants to pay Bob. They agree on a common random number $R$.
- Alice's pheromone seed $S(A) = 0^\frown R$.

# Pheromone seeds

- Alice wants to pay Bob. They agree on a common random number $R$.
- Alice's pheromone seed $S(A) = 0^\frown R$.
- Bob's pheromone seed $S(B) = 1^\frown R$.

# Pheromone seeds

- Alice wants to pay Bob. They agree on a common random number $R$.
- Alice's pheromone seed $S(A) = 0^\frown R$.
- Bob's pheromone seed $S(B) = 1^\frown R$.
- Derived seed: If $S = X^\frown R$, the derived seed is $S' = R$.

# Pheromone seeds

- Alice wants to pay Bob. They agree on a common random number $R$.
- Alice's pheromone seed $S(A) = 0^\frown R$.
- Bob's pheromone seed $S(B) = 1^\frown R$.
- Derived seed: If $S = X^\frown R$, the derived seed is $S' = R$.
- Conjugate seed: If $S = 0^\frown R$ (resp. $S = 1^\frown R$), the conjugate seed is $\bar{S} = 1^\frown R$ (resp. $\bar{S} = 0^\frown R$).

# Pheromone seeds

- Alice wants to pay Bob. They agree on a common random number $R$.
- Alice's pheromone seed $S(A) = 0^\frown R$.
- Bob's pheromone seed $S(B) = 1^\frown R$.
- Derived seed: If $S = X^\frown R$, the derived seed is $S' = R$.
- Conjugate seed: If $S = 0^\frown R$ (resp. $S = 1^\frown R$), the conjugate seed is $\bar{S} = 1^\frown R$ (resp. $\bar{S} = 0^\frown R$).
- Alice, resp. Bob, propagates $S(A)$, resp. $S(B)$, to neighbors.

# Pheromone seeds

- Alice wants to pay Bob. They agree on a common random number $R$.
- Alice's pheromone seed $S(A) = 0^\frown R$.
- Bob's pheromone seed $S(B) = 1^\frown R$.
- Derived seed: If $S = X^\frown R$, the derived seed is $S' = R$.
- Conjugate seed: If $S = 0^\frown R$ (resp. $S = 1^\frown R$), the conjugate seed is $\bar{S} = 1^\frown R$ (resp. $\bar{S} = 0^\frown R$).
- Alice, resp. Bob, propagates $S(A)$, resp. $S(B)$, to neighbors.

# Propagation and matching

# Propagation and matching

- A node that receives a pheromone seed $S$ notes from which neighbor it arrived and checks if $S$ or $\bar{S}$ was received before.

# Propagation and matching

- A node that receives a pheromone seed $S$ notes from which neighbor it arrived and checks if $S$ or $\bar{S}$ was received before.
- If none was received, it stores $S$ in the mempool and propagates to other neighbors.

# Propagation and matching

- A node that receives a pheromone seed $S$ notes from which neighbor it arrived and checks if $S$ or $\bar{S}$ was received before.
- If none was received, it stores $S$ in the mempool and propagates to other neighbors.
- If $S$ was already in the mempool but not $\bar{S}$, nothing else needs to be done.

# Propagation and matching

- A node that receives a pheromone seed $S$ notes from which neighbor it arrived and checks if $S$ or $\bar{S}$ was received before.
- If none was received, it stores $S$ in the mempool and propagates to other neighbors.
- If $S$ was already in the mempool but not $\bar{S}$, nothing else needs to be done.
- If $S$ is not in the mempool but $\bar{S}$ is, then a matching occurs. The node constructs the matched seed $S_m = 0^\frown S(A)$ and propagates it to the neighbors that send the pheromone seeds $S(A)$ and $S(B)$.

# Propagation and matching

- A node that receives a pheromone seed $S$ notes from which neighbor it arrived and checks if $S$ or $\bar{S}$ was received before.
- If none was received, it stores $S$ in the mempool and propagates to other neighbors.
- If $S$ was already in the mempool but not $\bar{S}$, nothing else needs to be done.
- If $S$ is not in the mempool but $\bar{S}$ is, then a matching occurs. The node constructs the matched seed $S_m = 0^\frown S(A)$ and propagates it to the neighbors that send the pheromone seeds $S(A)$ and $S(B)$.
- If both $S$ and $\bar{S}$ were already in the mempool then the matching occurred earlier and nothing needs to be done.

# Propagation and matching

- A node that receives a pheromone seed $S$ notes from which neighbor it arrived and checks if $S$ or $\bar{S}$ was received before.
- If none was received, it stores $S$ in the mempool and propagates to other neighbors.
- If $S$ was already in the mempool but not $\bar{S}$, nothing else needs to be done.
- If $S$ is not in the mempool but $\bar{S}$ is, then a matching occurs. The node constructs the matched seed $S_m = 0^\frown S(A)$ and propagates it to the neighbors that send the pheromone seeds $S(A)$ and $S(B)$.
- If both $S$ and $\bar{S}$ were already in the mempool then the matching occurred earlier and nothing needs to be done.

# Confirmation and payment

# Confirmation and payment

- Alice waits for several matched seed to arrive, and chooses one and constructs the confirmed seed $S_c = 0^\frown S_m$.

# Confirmation and payment

- Alice waits for several matched seed to arrive, and chooses one and constructs the confirmed seed $S_c = 0^\frown S_m$.
- Alice propagates the confirmed seed to the neighbor that send her the matched seed and waits for Bob the confirmation of the path.

# Confirmation and payment

- Alice waits for several matched seed to arrive, and chooses one and constructs the confirmed seed $S_c = 0^\frown S_m$.
- Alice propagates the confirmed seed to the neighbor that send her the matched seed and waits for Bob the confirmation of the path.
- Nodes that receive a confirmed seed propagate it back.

# Confirmation and payment

- Alice waits for several matched seed to arrive, and chooses one and constructs the confirmed seed $S_c = 0^\frown S_m$.
- Alice propagates the confirmed seed to the neighbor that send her the matched seed and waits for Bob the confirmation of the path.
- Nodes that receive a confirmed seed propagate it back.
- Once Bob receives the confirmed seed, he signals it to Alice and the payment is initiated through that path.

# Confirmation and payment

- Alice waits for several matched seed to arrive, and chooses one and constructs the confirmed seed $S_c = 0^\frown S_m$.
- Alice propagates the confirmed seed to the neighbor that send her the matched seed and waits for Bob the confirmation of the path.
- Nodes that receive a confirmed seed propagate it back.
- Once Bob receives the confirmed seed, he signals it to Alice and the payment is initiated through that path.

# Amount and fees

# Amount and fees

- Alice and Bob enrich the pheromone seed by adding:
  - An amount field.

# Amount and fees

- Alice and Bob enrich the pheromone seed by adding:
  - An amount field.
  - A maximal fee field.

# Amount and fees

- Alice and Bob enrich the pheromone seed by adding:
  - An amount field.
  - A maximal fee field.
  - A current fee field initialized to 0.

# Amount and fees

- Alice and Bob enrich the pheromone seed by adding:
  - An amount field.
  - A maximal fee field.
  - A current fee field initialized to 0.

# Amount and fees

- Alice and Bob enrich the pheromone seed by adding:
  - An amount field.
  - A maximal fee field.
  - A current fee field initialized to 0.
- Nodes only propagate pheromone seeds with an amount compatible with the volume of the payment channels.

# Amount and fees

- Alice and Bob enrich the pheromone seed by adding:
  - An amount field.
  - A maximal fee field.
  - A current fee field initialized to 0.
- Nodes only propagate pheromone seeds with an amount compatible with the volume of the payment channels.
- Nodes increase the current fee field with their fee.

# Amount and fees

- Alice and Bob enrich the pheromone seed by adding:
  - An amount field.
  - A maximal fee field.
  - A current fee field initialized to 0.
- Nodes only propagate pheromone seeds with an amount compatible with the volume of the payment channels.
- Nodes increase the current fee field with their fee.
- The node matching conjugate pheromone seeds updated the amount of the fee field adding his fee to both fee amounts, and checks that is lower than the maximal fee.

# Amount and fees

- Alice and Bob enrich the pheromone seed by adding:
  - An amount field.
  - A maximal fee field.
  - A current fee field initialized to 0.
- Nodes only propagate pheromone seeds with an amount compatible with the volume of the payment channels.
- Nodes increase the current fee field with their fee.
- The node matching conjugate pheromone seeds updated the amount of the fee field adding his fee to both fee amounts, and checks that is lower than the maximal fee.

# Mempool management

# Mempool management

- Nodes only store non-confirmed information for a few seconds. Each node choses his treshold time $\tau_0$.

# Mempool management

- Nodes only store non-confirmed information for a few seconds. Each node choses his treshold time $\tau_0$.
- After a period $\tau_0$ nodes erase the data.

# Mempool management

- Nodes only store non-confirmed information for a few seconds. Each node choses his treshold time $\tau_0$.
- After a period $\tau_0$ nodes erase the data.
- Alice chooses her own waiting time $\tau_1$ and selection algorithm for selecting matched seeds (minimum fee after the period $\tau_1$ is the obvious).

# Mempool management

- Nodes only store non-confirmed information for a few seconds. Each node choses his treshold time $\tau_0$.
- After a period $\tau_0$ nodes erase the data.
- Alice chooses her own waiting time $\tau_1$ and selection algorithm for selecting matched seeds (minimum fee after the period $\tau_1$ is the obvious).
- Seeds of about 30 Bytes are probably acceptable.

# Mempool management

- Nodes only store non-confirmed information for a few seconds. Each node choses his treshold time $\tau_0$.
- After a period $\tau_0$ nodes erase the data.
- Alice chooses her own waiting time $\tau_1$ and selection algorithm for selecting matched seeds (minimum fee after the period $\tau_1$ is the obvious).
- Seeds of about 30 Bytes are probably acceptable.
- $\tau_0$ of about 2 sec seem realistic depending on the speed of communications.

# Mempool management

- Nodes only store non-confirmed information for a few seconds. Each node choses his treshold time $\tau_0$.
- After a period $\tau_0$ nodes erase the data.
- Alice chooses her own waiting time $\tau_1$ and selection algorithm for selecting matched seeds (minimum fee after the period $\tau_1$ is the obvious).
- Seeds of about 30 Bytes are probably acceptable.
- $\tau_0$ of about 2 sec seem realistic depending on the speed of communications.
- A mempool space of a few Mb should be enough to process thousands of Tx per second.

# Mempool management

- Nodes only store non-confirmed information for a few seconds. Each node choses his treshold time $\tau_0$.
- After a period $\tau_0$ nodes erase the data.
- Alice chooses her own waiting time $\tau_1$ and selection algorithm for selecting matched seeds (minimum fee after the period $\tau_1$ is the obvious).
- Seeds of about 30 Bytes are probably acceptable.
- $\tau_0$ of about 2 sec seem realistic depending on the speed of communications.
- A mempool space of a few Mb should be enough to process thousands of Tx per second.

# Self-improvement features

# Self-improvement features

- The topology of the network is dynamical.

# Self-improvement features

- The topology of the network is dynamical.
- Nodes can store historical performance of neighbors and compute some weighting.

# Self-improvement features

- The topology of the network is dynamical.
- Nodes can store historical performance of neighbors and compute some weighting.
- Comparison of historical weighting with short term one allows to adjust to topology changes in the network.

# Self-improvement features

- The topology of the network is dynamical.
- Nodes can store historical performance of neighbors and compute some weighting.
- Comparison of historical weighting with short term one allows to adjust to topology changes in the network.
- Best analysis will increase traffic and profitability.

# Self-improvement features

- The topology of the network is dynamical.
- Nodes can store historical performance of neighbors and compute some weighting.
- Comparison of historical weighting with short term one allows to adjust to topology changes in the network.
- Best analysis will increase traffic and profitability.

# Numerical simulations

# Numerical simulations

- Warning! The paper ArXiv:1807.00151 is only a first draft.

# Numerical simulations

- Warning! The paper ArXiv:1807.00151 is only a first draft.
- Numerical simulations are necessary to evaluate scalability, and resources.

# Numerical simulations

- Warning! The paper ArXiv:1807.00151 is only a first draft.
- Numerical simulations are necessary to evaluate scalability, and resources.
- Numerical simulations and dynamical study are necessary to find out the best self-improvement parametrization.

# Numerical simulations

- Warning! The paper ArXiv:1807.00151 is only a first draft.
- Numerical simulations are necessary to evaluate scalability, and resources.
- Numerical simulations and dynamical study are necessary to find out the best self-improvement parametrization.
- Communication speed seems to be the main bottleneck.

# Numerical simulations

- Warning! The paper ArXiv:1807.00151 is only a first draft.
- Numerical simulations are necessary to evaluate scalability, and resources.
- Numerical simulations and dynamical study are necessary to find out the best self-improvement parametrization.
- Communication speed seems to be the main bottleneck.

# Properties

# Properties

- No routing tables. No information about the network topology.

# Properties

- No routing tables. No information about the network topology.
- Other nodes don't know about transactions between Alice and Bob. (Anonimity)

# Properties

- No routing tables. No information about the network topology.
- Other nodes don't know about transactions between Alice and Bob. (Anonimity)
- Equal role for all nodes. (Decentralization)

# Properties

- No routing tables. No information about the network topology.
- Other nodes don't know about transactions between Alice and Bob. (Anonimity)
- Equal role for all nodes. (Decentralization)
- Scalability (pending numerical simulations).

# Properties

- No routing tables. No information about the network topology.
- Other nodes don't know about transactions between Alice and Bob. (Anonimity)
- Equal role for all nodes. (Decentralization)
- Scalability (pending numerical simulations).
- Self-improvement features.

# Properties

- No routing tables. No information about the network topology.
- Other nodes don't know about transactions between Alice and Bob. (Anonimity)
- Equal role for all nodes. (Decentralization)
- Scalability (pending numerical simulations).
- Self-improvement features.
- Resilience to failure of part of the network. (Anti-fragility)

# Properties

- No routing tables. No information about the network topology.
- Other nodes don't know about transactions between Alice and Bob. (Anonimity)
- Equal role for all nodes. (Decentralization)
- Scalability (pending numerical simulations).
- Self-improvement features.
- Resilience to failure of part of the network. (Anti-fragility)

# Thank for your attention!