Mixer

Efficient Many-to-All Broadcast in Dynamic Wireless Mesh Networks

Carsten Herrmann, Fabian Mager, Marco Zimmerling
Networked Embedded Systems Lab, TU Dresden, Germany

Why Many-to-All Communication?

Why Many-to-All

universal:

- can serve any possible traffic pattern (one-to-one, one-to-many, all-to-all, etc.)

fundamental for a growing number of emerging applications and services:

- coordination and distributed control in Cyber-Physical Systems
  - factory automation
  - collaborative agents, drone swarms
  - (I)IoT edge
- programming abstractions (based on data replication)
- fault tolerance mechanisms (based on state replication)
- over-the-air programming / updates, ...

Source: U.S.DOT

Requirements for Many-to-All

- fast (10…500 ms end-to-end)
- reliable
- support for dynamic mesh topologies
- support for adequate message sizes (tens of bytes)
- energy efficient (weight, cost)

[1] Akerberg et al., Future research challenges in wireless sensor and actuator networks targeting industrial automation, IEEE INDIN 2011

Current Solutions

- Multi-Sink Routing [1]:
  - degenerates under high network dynamics
- Sequential Flooding (S-Glossy) [2]:
  - suboptimal scaling $O(M \cdot T)$
- Pipelined Flooding [3]:
  - not universal (only one-to-all)
  - infeasible under high network dynamics

[1] e.g. Mottola et al., MUSTER: Adaptive Energy-Aware Multisink Routing in Wireless Sensor Networks, IEEE Transactions on Mobile Computing 2011
[2] e.g. Ferrari et al., Efficient network flooding and time synchronization with Glossy, ACM/IEEE IPSN 2011
[3] e.g. Du et al., When Pipelines Meet Fountain: Fast Data Dissemination in Wireless Sensor Networks, ACM SenSys 2015

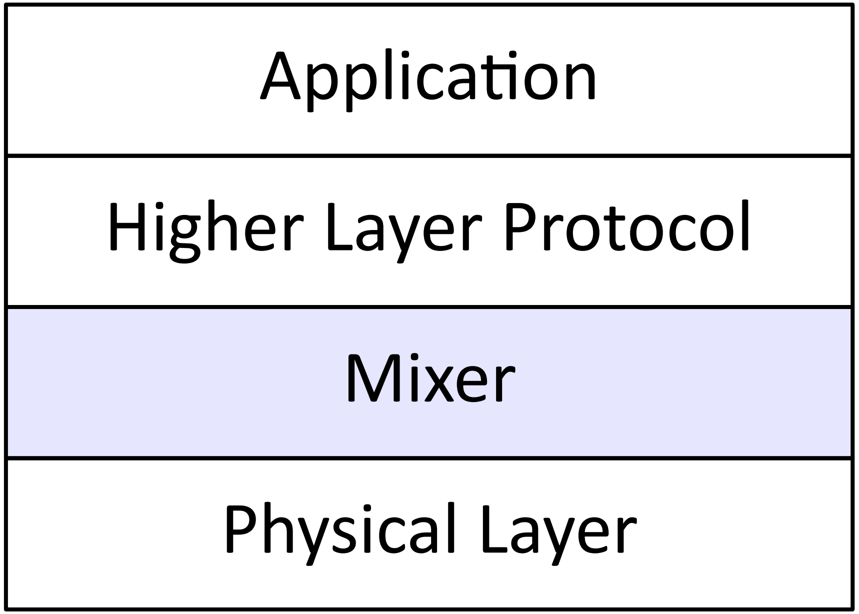
Our Contribution

Mixer, a new many-to-all broadcast primitive for dynamic wireless mesh networks

- significantly outperforms prior many-to-all solutions, approaches order-optimal scaling O(M + T)
- provides nearly perfect reliability despite significant network dynamics
- supports the full spectrum from one-to-all to all-to-all communication

Our Contribution

Mixer, a new many-to-all broadcast primitive for dynamic wireless mesh networks



Application

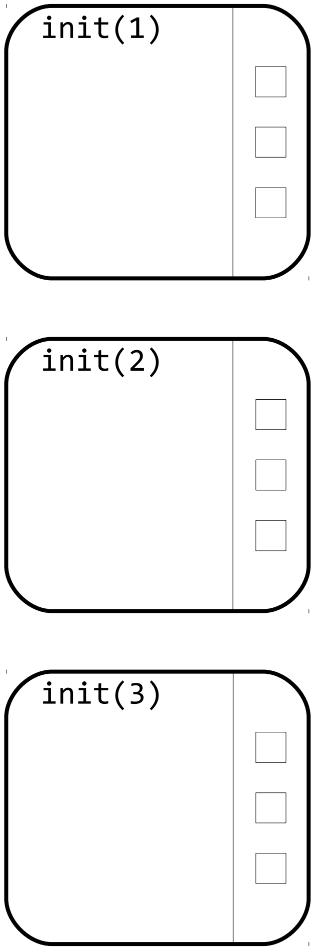
Higher Layer Protocol

Mixer

Physical Layer

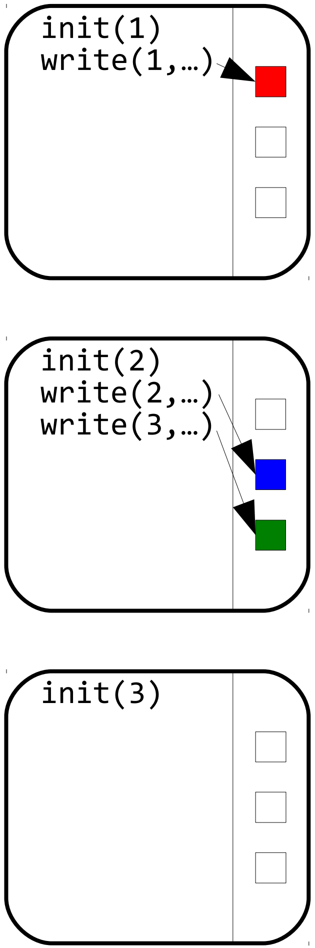
Our Contribution

```
mixer_init(node_id)
mixer_write(index, *msg, size)
mixer_arm(mode)
mixer_start()
mixer_read(index)
```



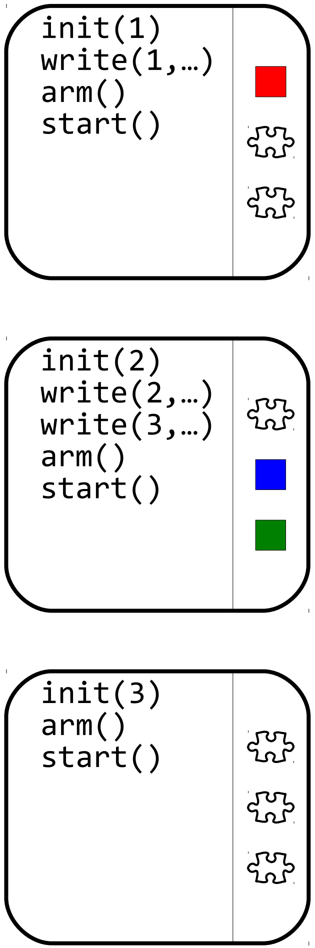
Our Contribution

```
mixer_init(node_id)
mixer_write(index, *msg, size)
mixer_arm(mode)
mixer_start()
mixer_read(index)
```



Our Contribution

```
mixer_init(node_id)
mixer_write(index, *msg, size)
mixer_arm(mode)
mixer_start()
mixer_read(index)
```



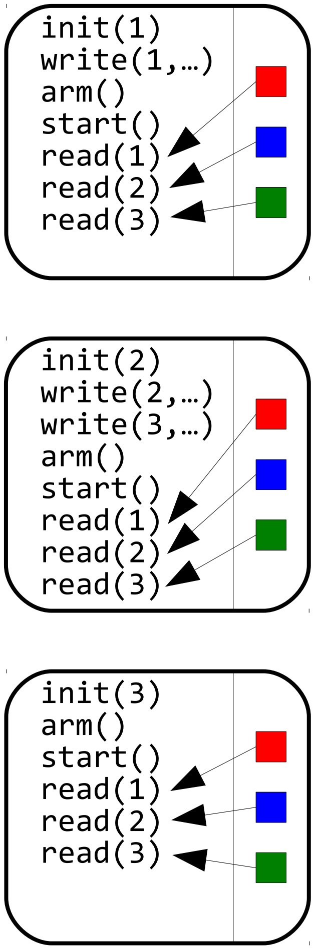
```
init(1)
write(1,…)
arm()
start()
```

```
init(2)
write(2,…)
write(3,…)
arm()
start()
```

```
init(3)
arm()
start()
```

Our Contribution

```
mixer_init(node_id)
mixer_write(index, *msg, size)
mixer_arm(mode)
mixer_start()
mixer_read(index)
```



```
init(1)
write(1,…)
arm()
start()
read(1)
read(2)
read(3)
```

```
init(2)
write(2,…)
write(3,…)
arm()
start()
read(1)
read(2)
read(3)
```

```
init(3)
arm()
start()
read(1)
read(2)
read(3)
```

Ingredients of Mixer

Ingredients of Mixer

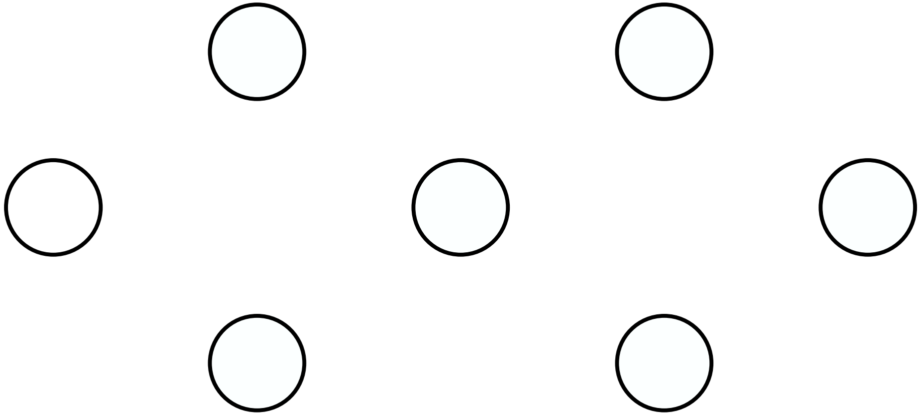
Key Concepts

- Random Linear Network Coding (RLNC)
  → overlay floods
- Synchronous Transmissions
  → enable capture and spatial reuse

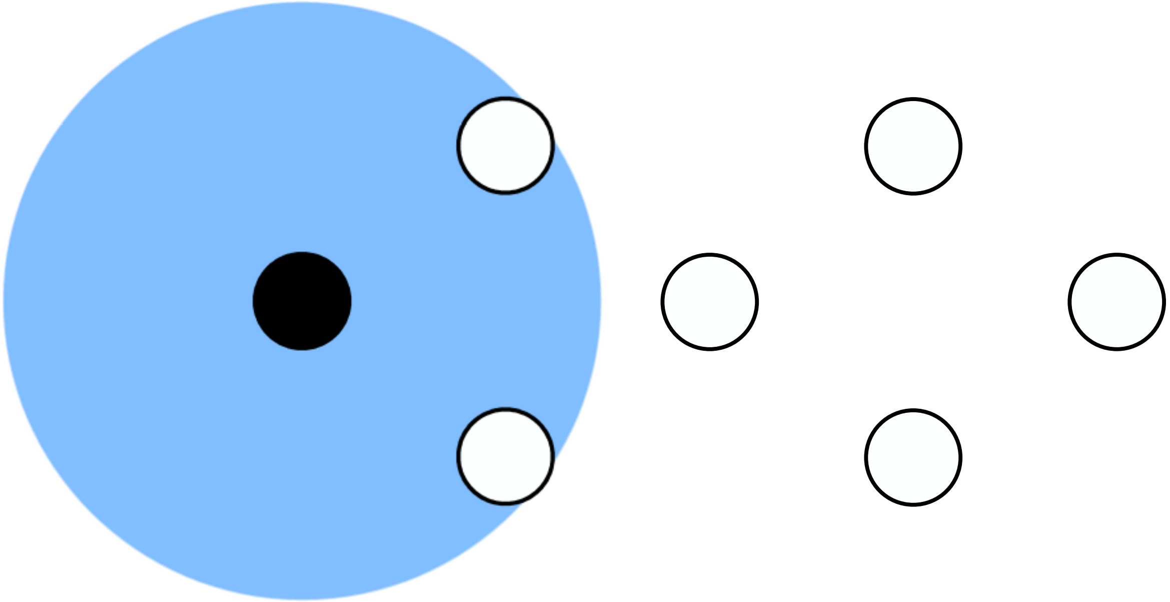
Linear Network Coding

task: disseminate 3 messages	Sequential Flooding (S-Glossy)	Linear Network Coding (example)
slot 1	`1 0 0 | 21`	`1 0 0 | 21`
...	`1 0 0 | 21`	`1 1 0 | 43`
	`0 1 0 | 22`	`0 1 1 | 45`
	`0 1 0 | 22`	`0 0 1 | 23`
...	`0 0 1 | 23`	-
slot 6	`0 0 1 | 23`	-
robustness	1 packet	1 packet
cost	6 slots	4 slots + computations

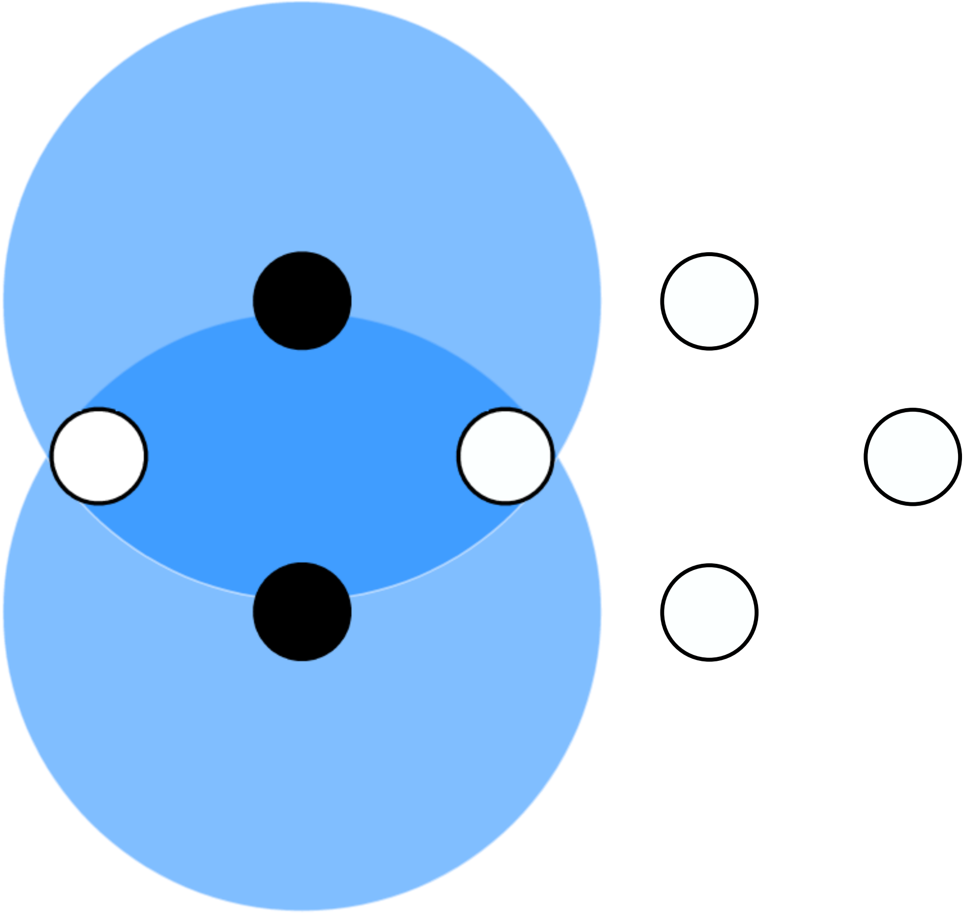
Sequential Flooding



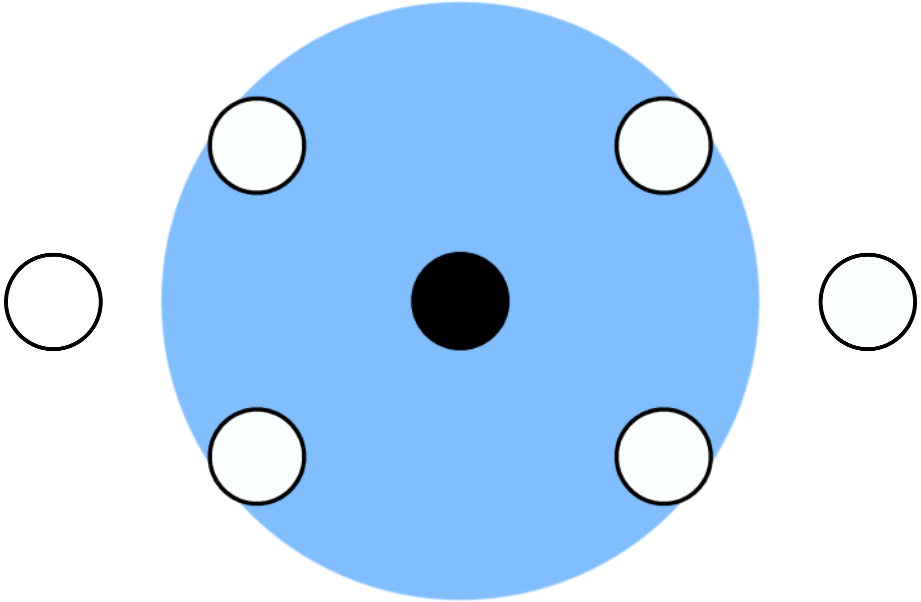
Sequential Flooding



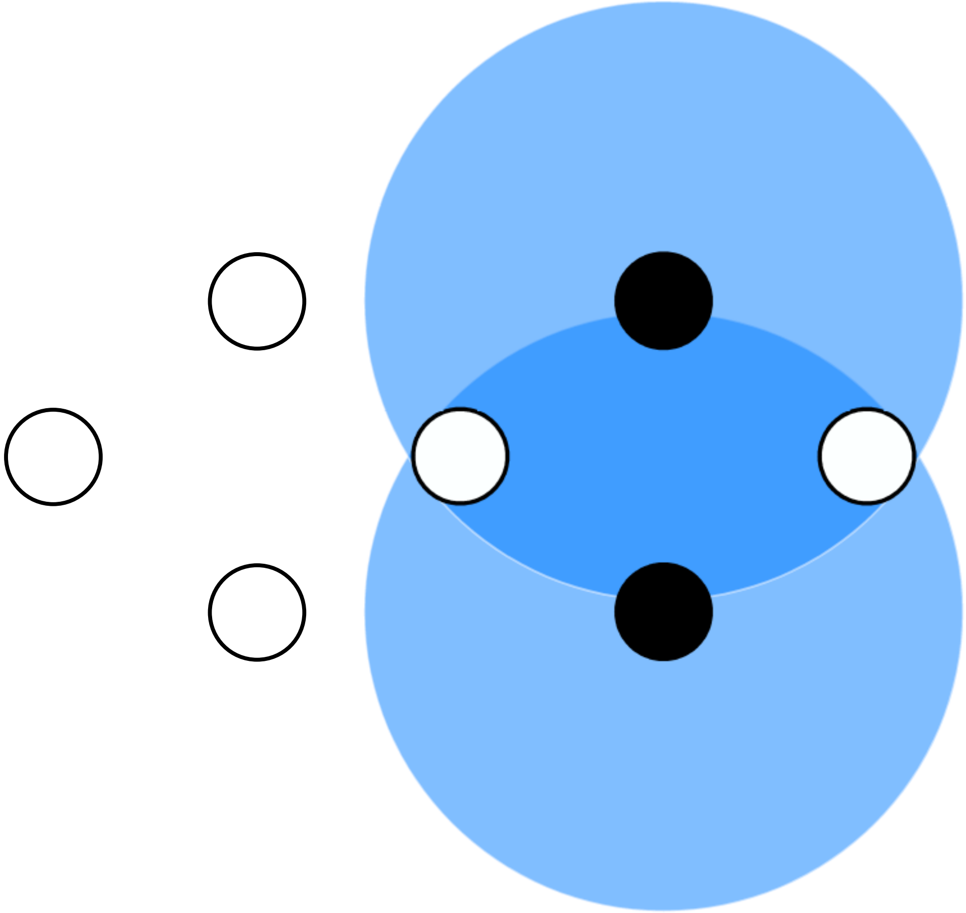
Sequential Flooding



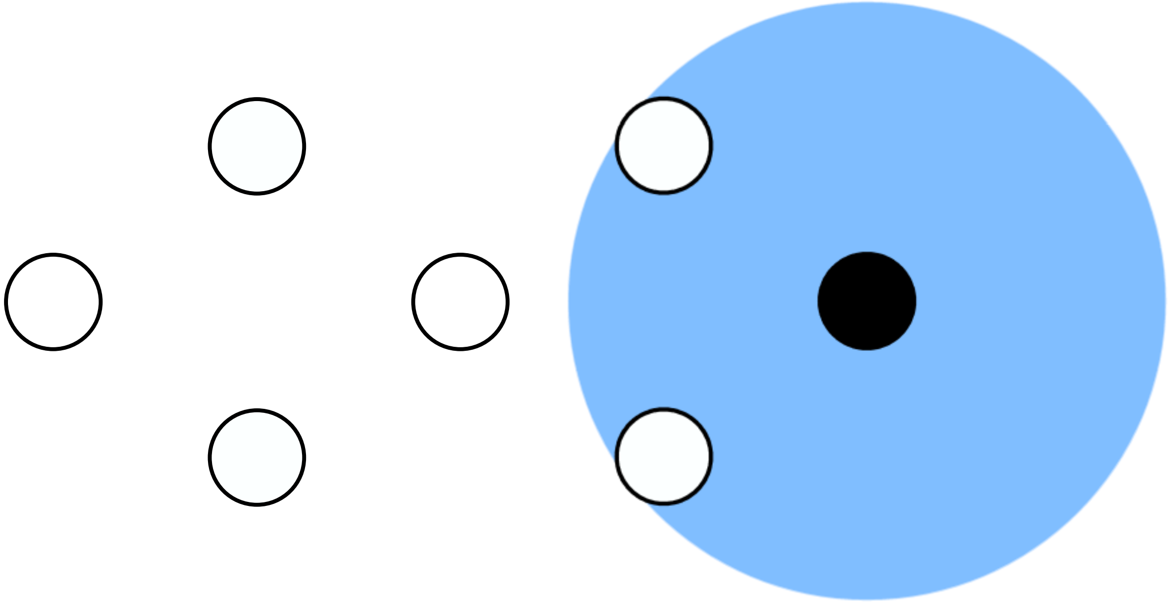
Sequential Flooding



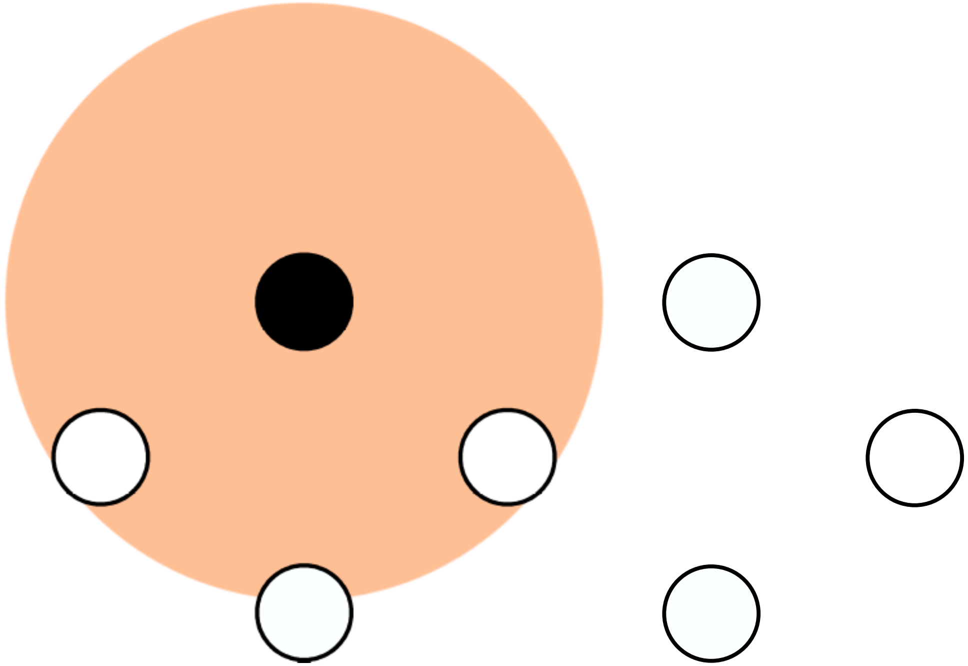
Sequential Flooding



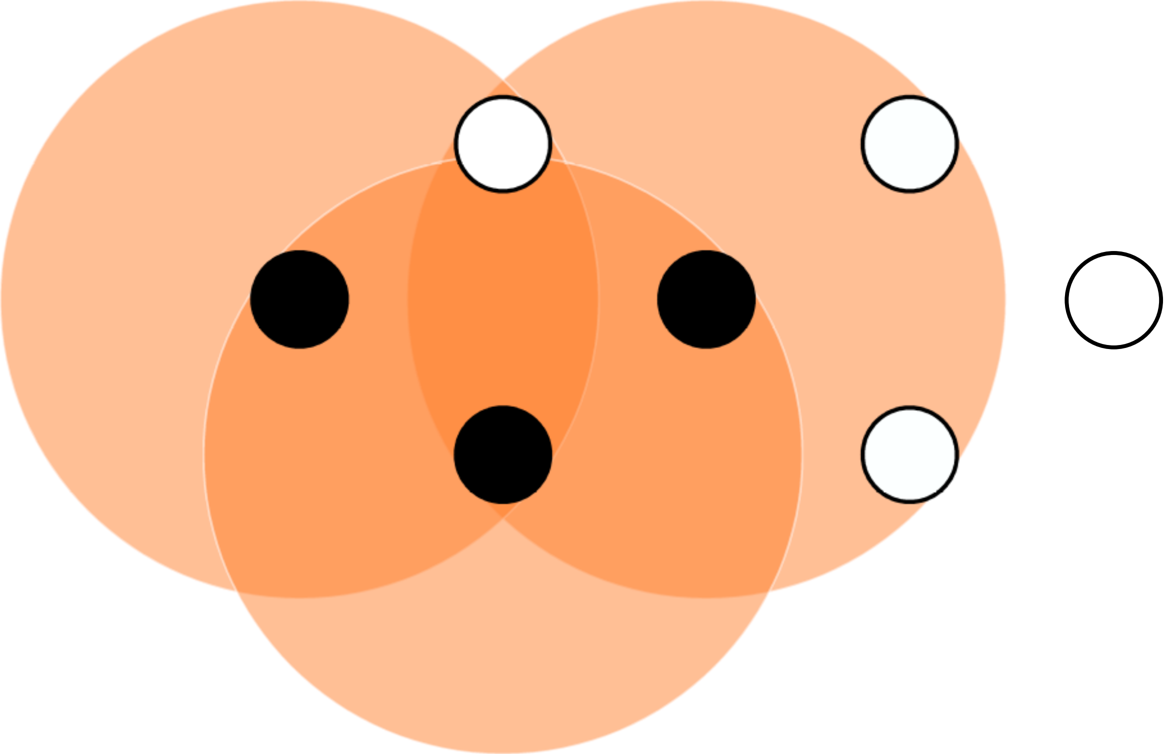
Sequential Flooding



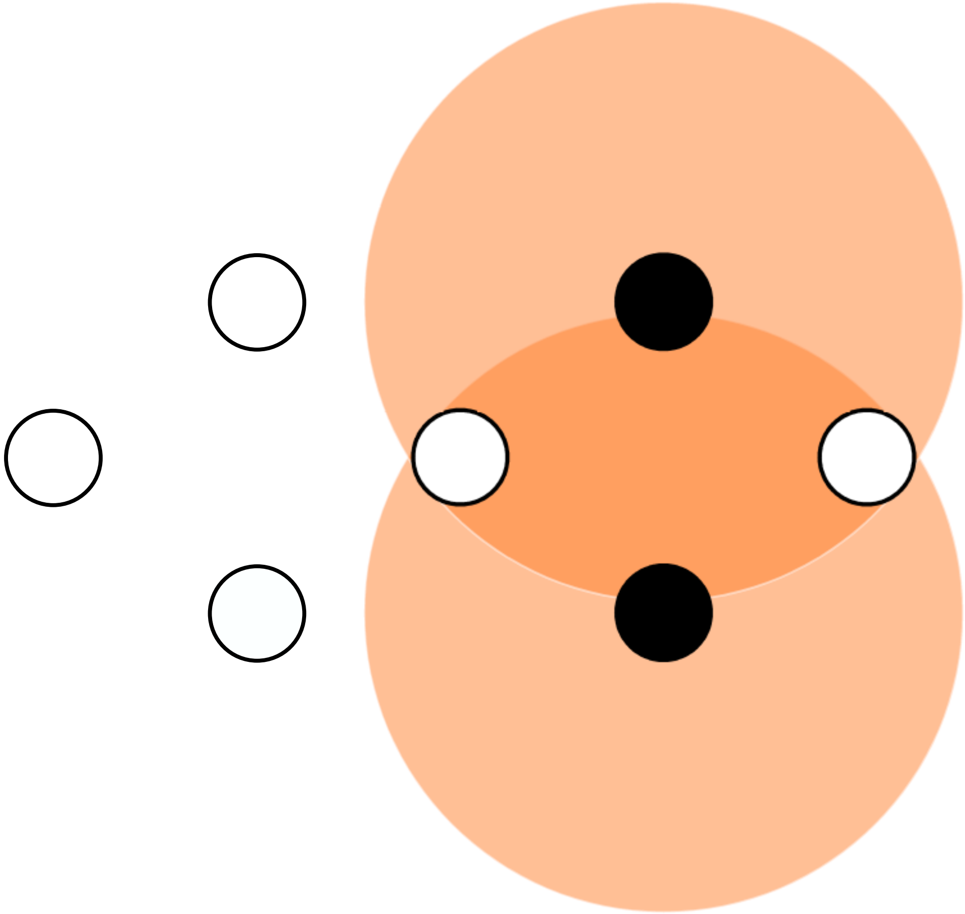
Sequential Flooding



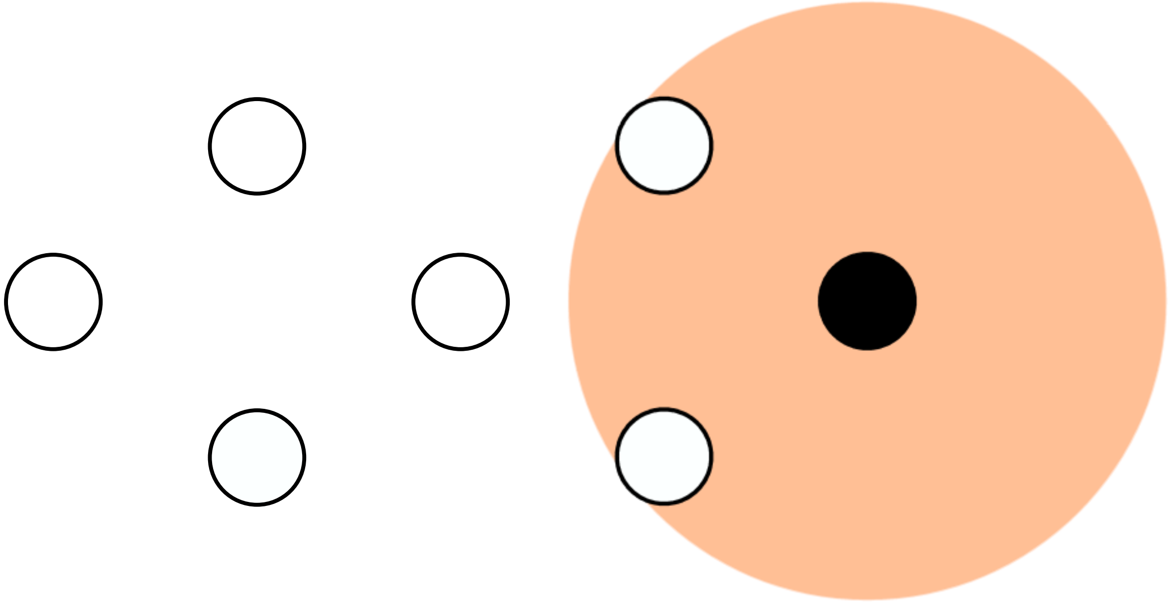
Sequential Flooding



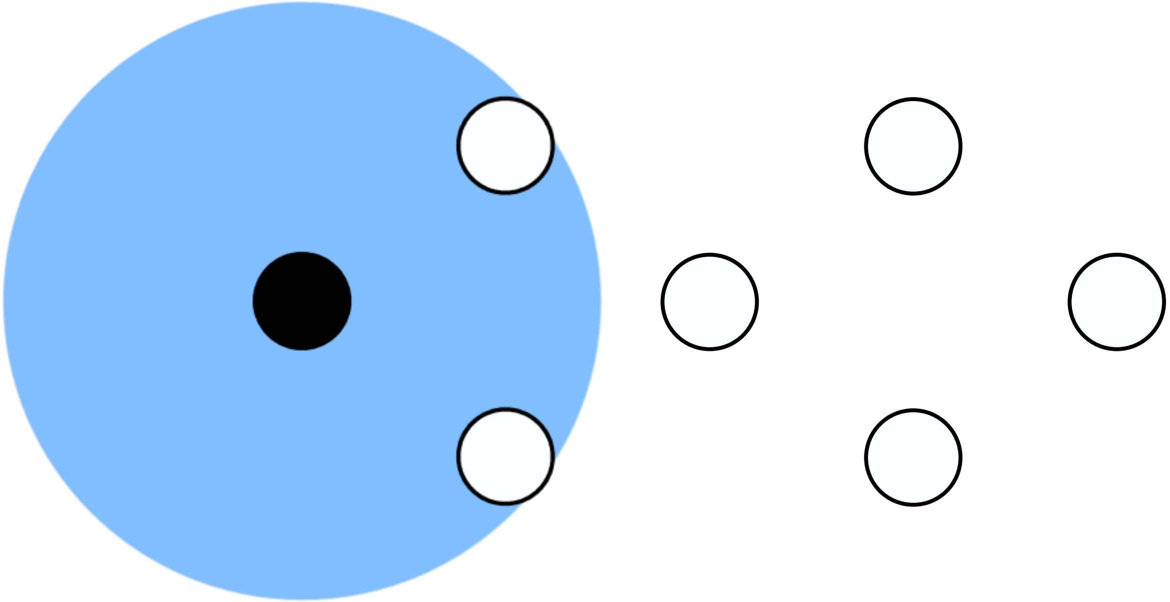
Sequential Flooding



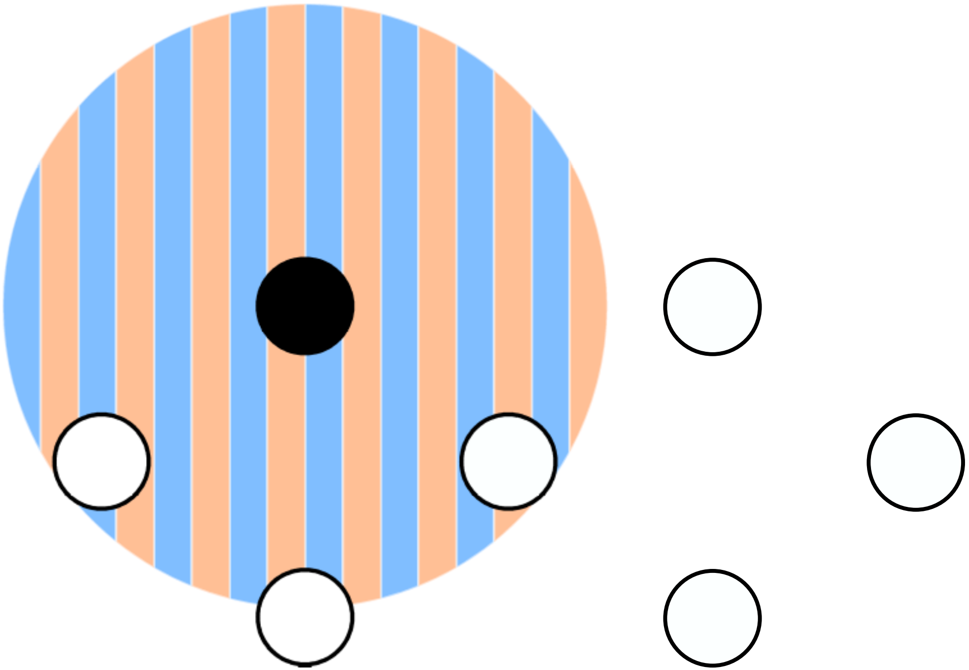
Sequential Flooding



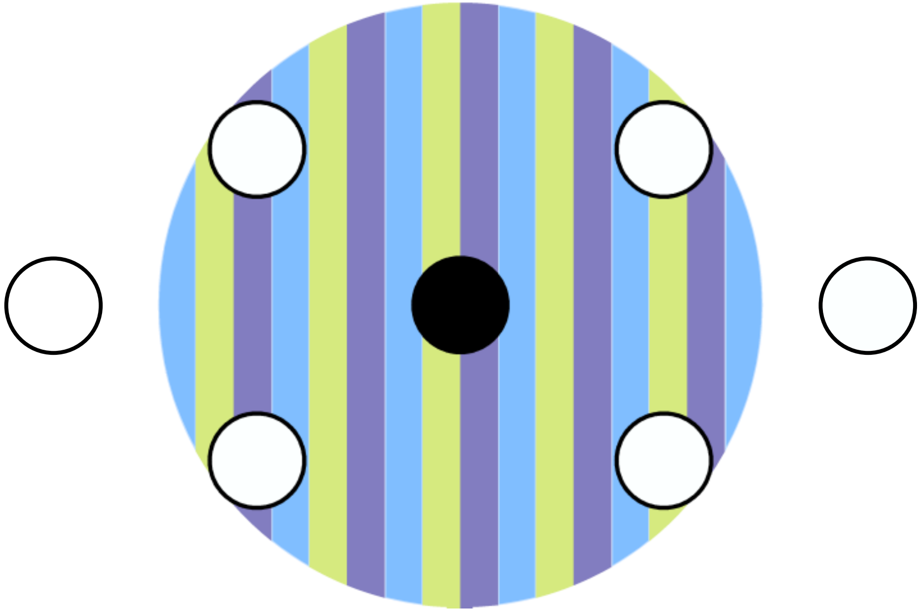
Mixer



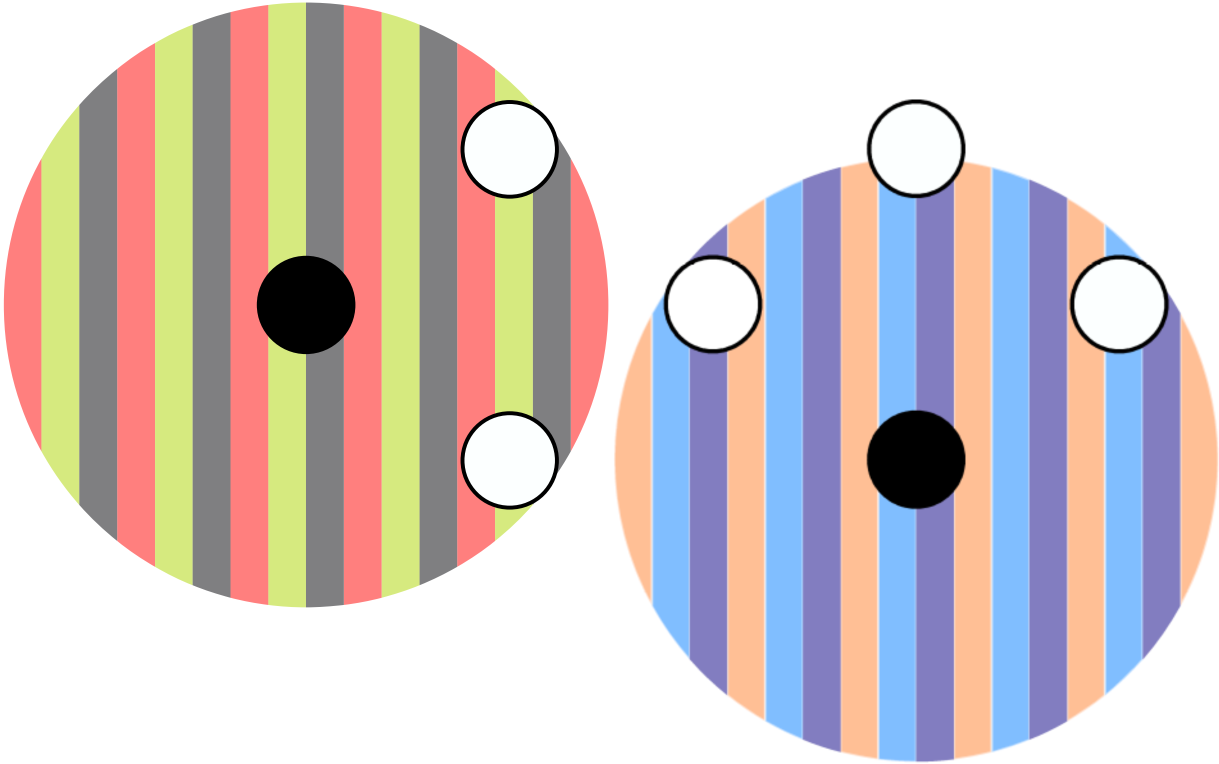
Mixer



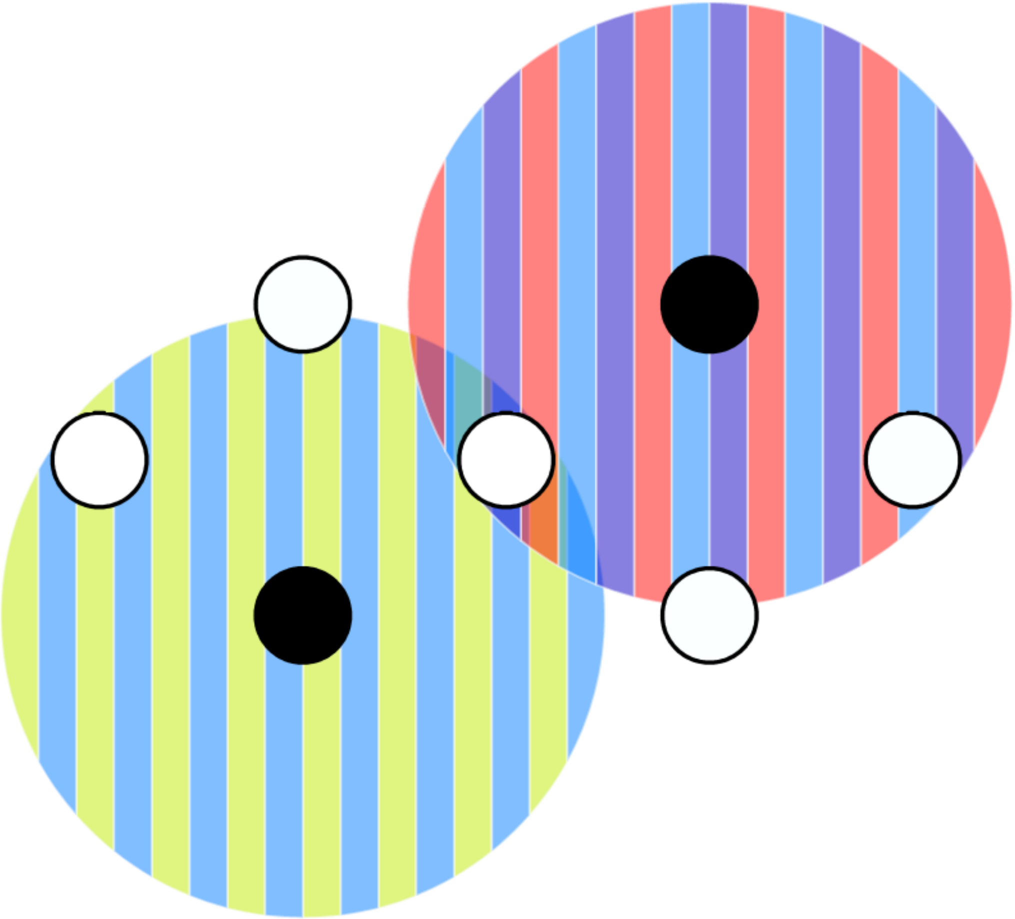
Mixer



Mixer



Mixer



Ingredients of Mixer

Key Concepts

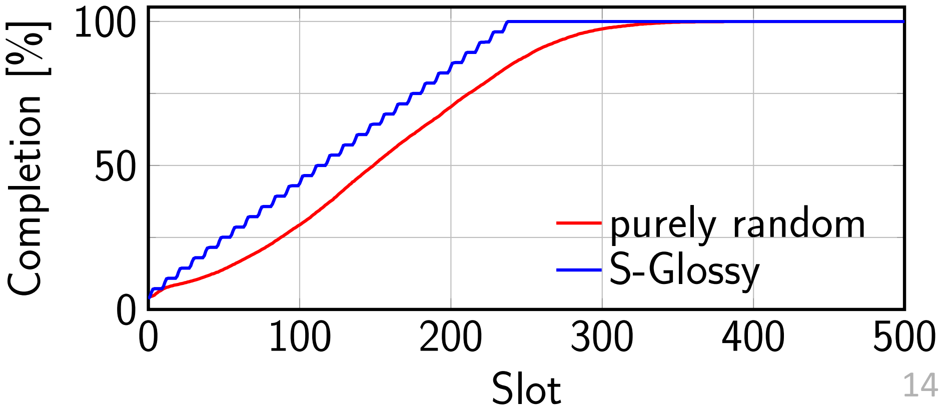
- Random Linear Network Coding (RLNC) → overlay floods
- Synchronous Transmissions → enable capture and spatial reuse

Challenges

- When should a node send or listen?
- What should a node send (combine)?

Efficient Architecture

- transport layer with sideload feature
- deliberate scheduling of (matrix) operations



Ingredients of Mixer

Key Concepts

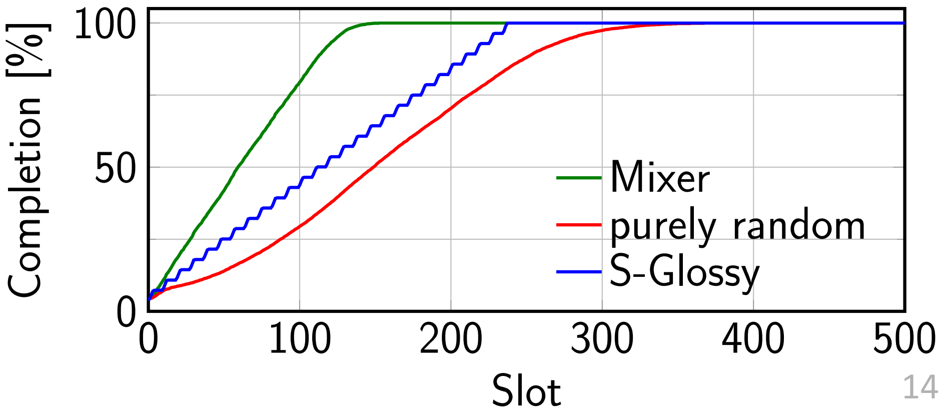
- Random Linear Network Coding (RLNC)
  → overlay floods
- Synchronous Transmissions
  → enable capture and spatial reuse

Efficient Architecture

- transport layer with sideload feature
- deliberate scheduling of (matrix) operations

Smart Policies / Features ("Spices")

- Semi-Coordinated Transmissions
- Explicit Innovation Forwarding
- Knowledge-based Startup Behavior
- Active Requests
- Smart Shutdown



Mixer in Action

…

Evaluation

Setup

- implementation on TelosB
  - MSP430, 4 MHz, 10KB RAM, IEEE 802.15.4
  - ARM port is in progress…
- extensive tests on
  - FlockLab (ETH Zürich, 27 nodes, 4 hops)
  - Indriya (NU Singapore, 94 nodes, 8 hops)

Experiments

Reliability

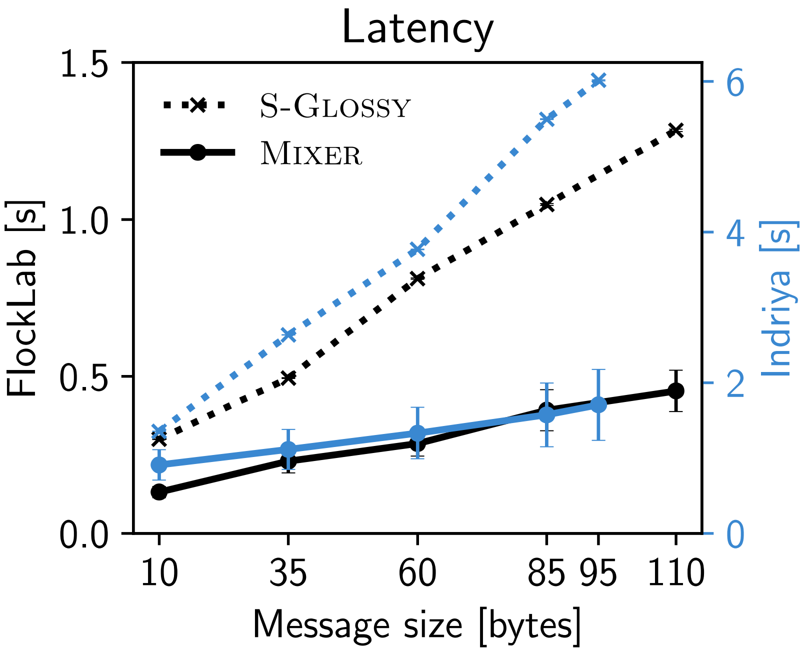
100%

Mixer delivered all messages
in all FlockLab and Indriya experiments

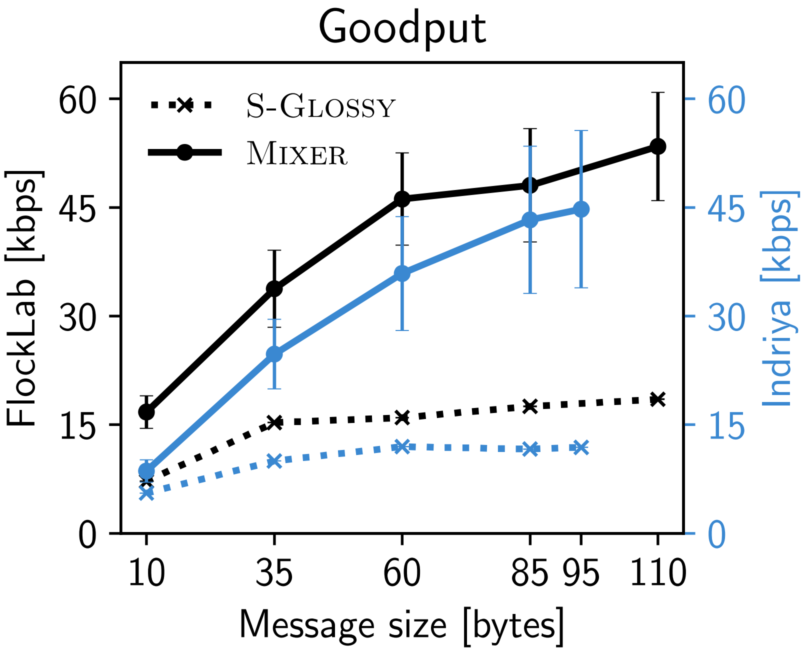
Experiments

- general performance: impact of
  - **message size**

Performance all-to-all



Latency



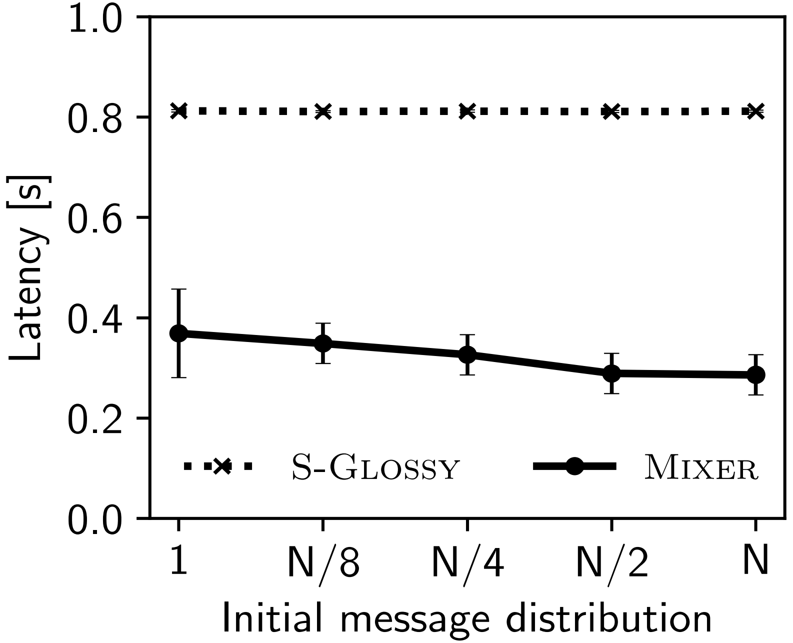
Goodput

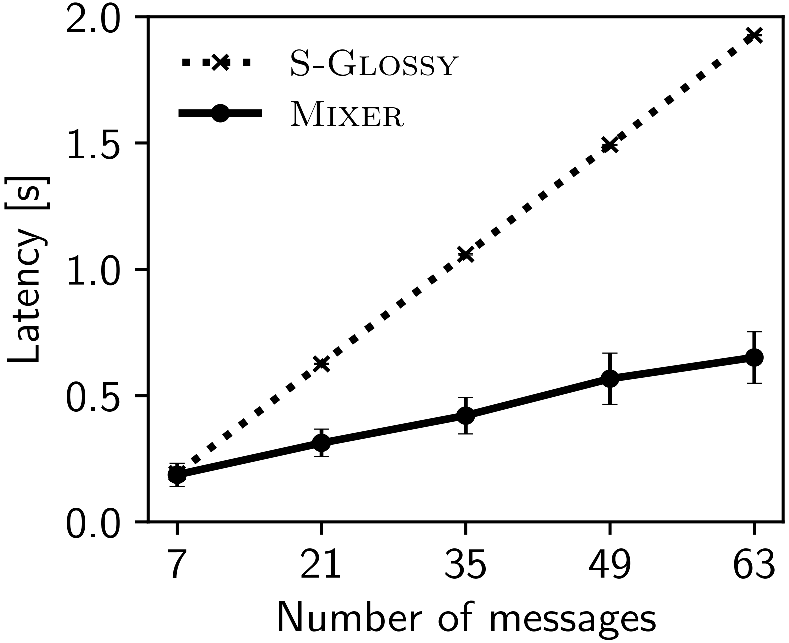
Mixer outperforms S-Glossy by up to **3.8x**

Experiments

- general performance: impact of
  - message size
  - **number of messages**
  - **initial message distribution**

Performance many-to-all





Mixer is versatile and scales well

Experiments

- general performance: impact of
  - message size
  - number of messages
  - initial message distribution
- impact of node failures
- impact of node mobility
- potential of faster CPUs and physical layers

Conclusion

Conclusion

Mixer, a many-to-all broadcast primitive

- designed for dynamic wireless mesh networks
- embeds **RLNC and synchronous transmissions** into an efficient architecture and adds smart policies to make the combination **„spicy“**
- supports any initial message distribution, i.e., complete to-all to all-to-all communication patterns

Visit https://mixer.nes-lab.org

→ source code, tutorial projects, documentation, …

→ TelosB (MSP430) available right now, ARM coming soon