

Monitoring Churn in Wireless Networks



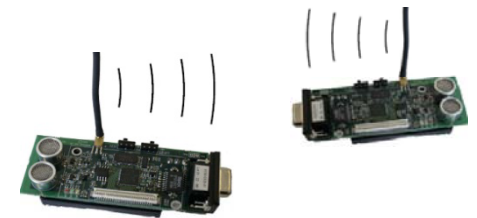
*Stephan Holzer
Yvonne Anne Pignolet
Jasmin Smula
Roger Wattenhofer*

Motivation / Intro



Network of sensor nodes:

- measuring certain properties of their environment
- wireless, communicating on several channels
- battery powered



Nodes might fail / nodes may be added

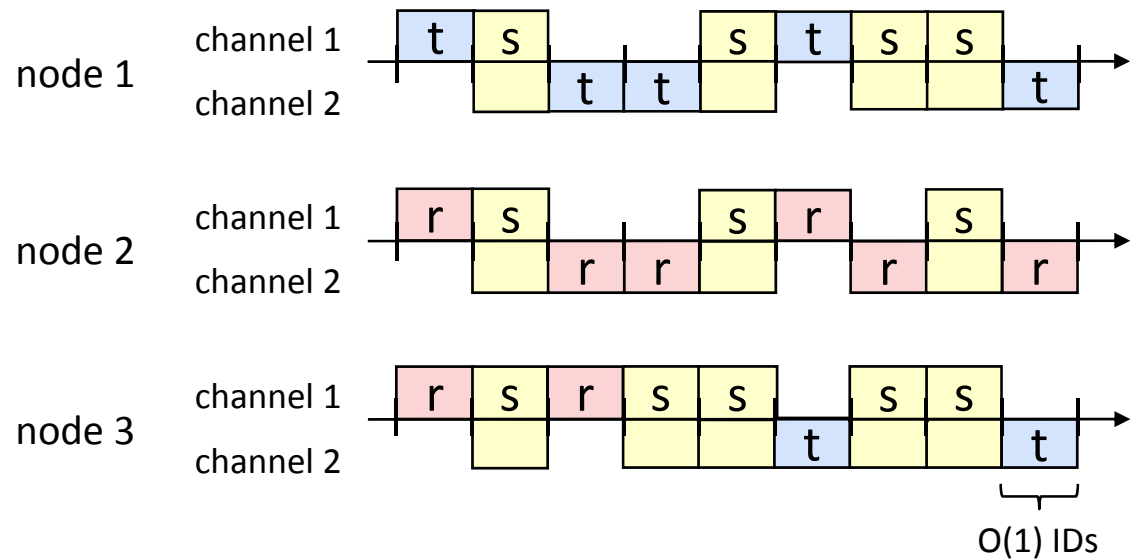
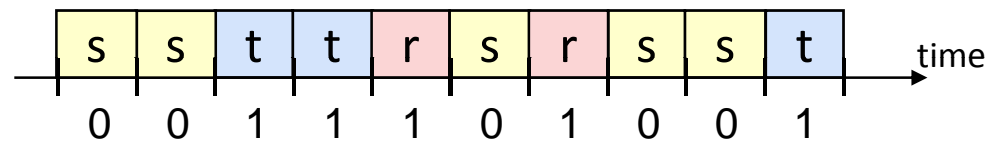
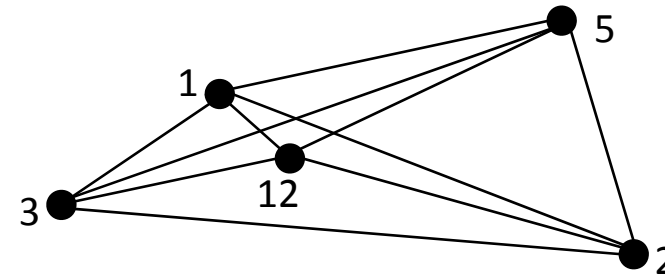
All nodes should be aware of all present nodes

- with small delay
- with little energy consumption
- using few channels for communication

Model



- n nodes with IDs
- single-hop
- synchronized time slots
- transmit / receive / sleep
- energy 1 / 1 / 0
- local computations free
- k channels
- no collision detection
- bounded message size



Model cont'd

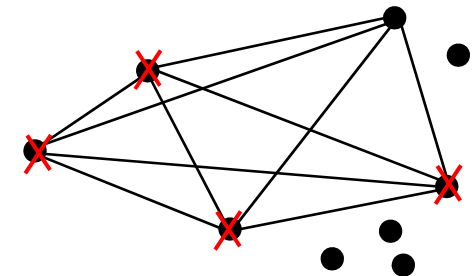
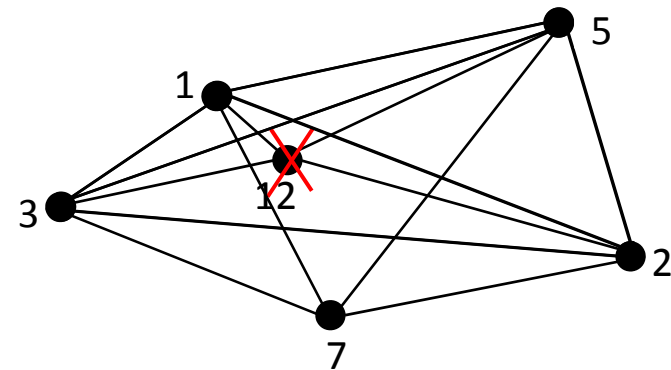


Nodes may join or crash at any time

churn = joins and crashes

burst = large number of joins and crashes in short time

Adversary:
May let nodes crash or join in order to make an algorithm fail



Goal



Every node must know the IDs of all nodes currently in the network

| | | | | | |
|---|---|---|---|---|----------------|
| 1 | 2 | 3 | 5 | 7 | 12 |
| ● | ● | ● | ● | ● | ● X |

Simple Lower Bounds



What time / energy is at least necessary in this model?

- every node can only receive one message per time slot containing at most a constant number of IDs

→ on average only constant rate of churn tolerable per time slot

→ $\Omega(b)$ time slots necessary to learn about b joins / crashes

→ every node needs $\Omega(b)$ energy units to learn about b joins / crashes

Results



Our Monitoring Algorithm:

- tolerates churn bursts in any order of magnitude
- is deterministic except for detection of joining nodes
- handles asymptotically maximum average rate of churn tolerable in this model
- after each burst of size b it takes
 - $O(b + \log n)$ time slots and
 - $O(b + \log n)$ energy per nodeuntil all nodes have updated their ID table (optimal up to additive logarithmic term)
- needs $\Theta(n/\log n)$ channels

Results cont'd



Our Monitoring Algorithm:

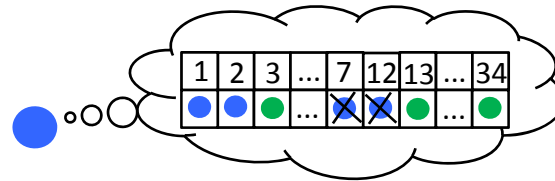
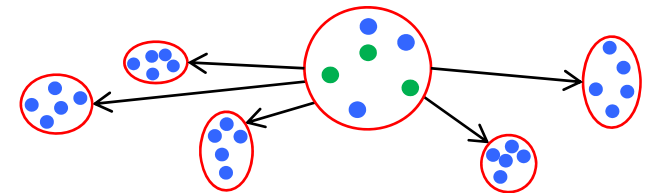
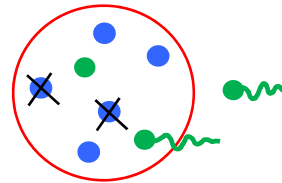
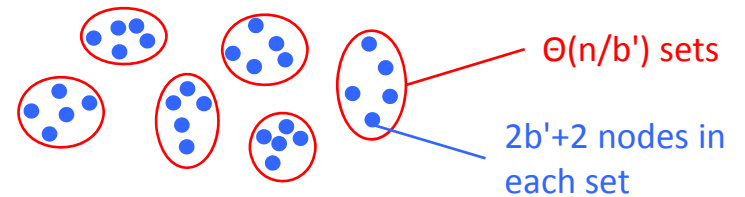
- can get by with less than $\Theta(n/\log n)$ channels:

- k channels available \rightarrow time $O\left(b + \frac{n}{k} \log\left(\frac{b}{\log n}\right)\right)$.

Monitoring Algorithm



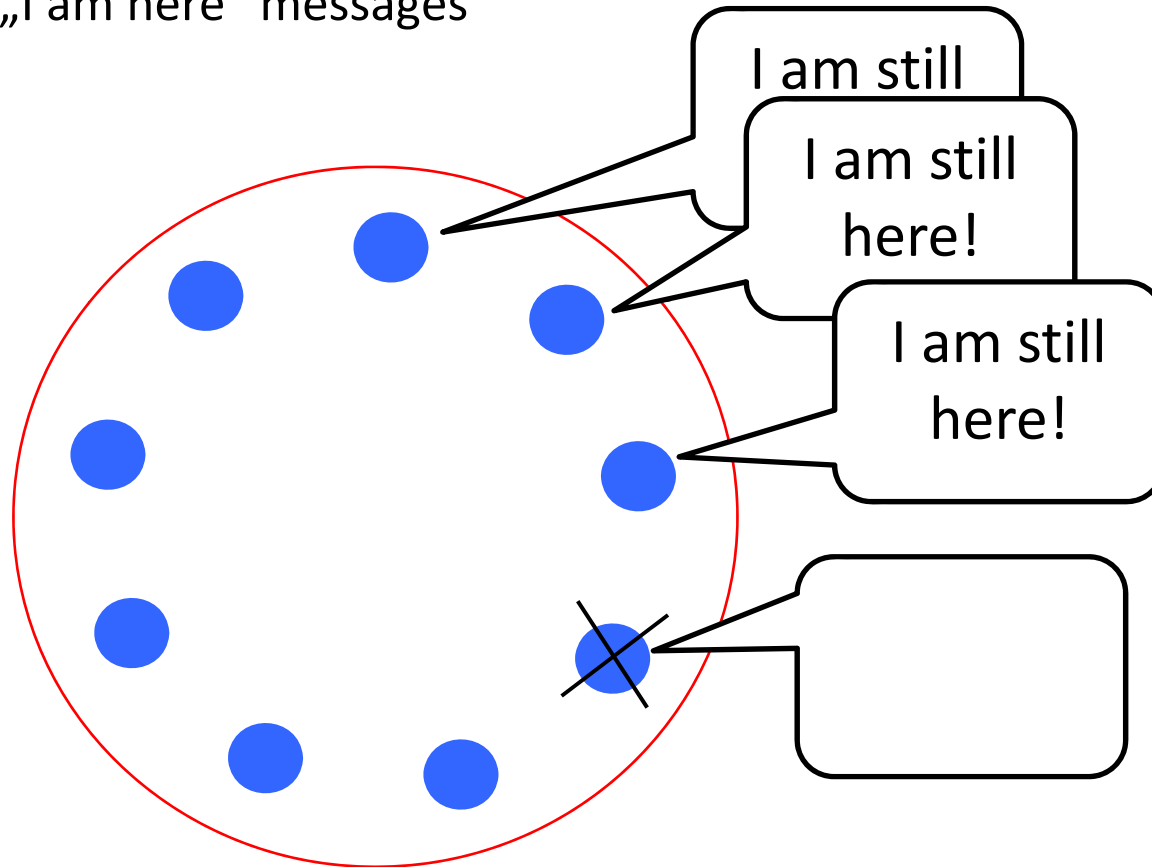
- burst size is assumed to be $b' = \log n$
- nodes partitioned into $n/(2b'+2)-1$ sets
- each set detects crashed and joined nodes on its own channel
- disseminate information to all nodes
- all nodes update ID table
- double b' if algo did not work



Crash Detection



- nodes send „I am here“ messages

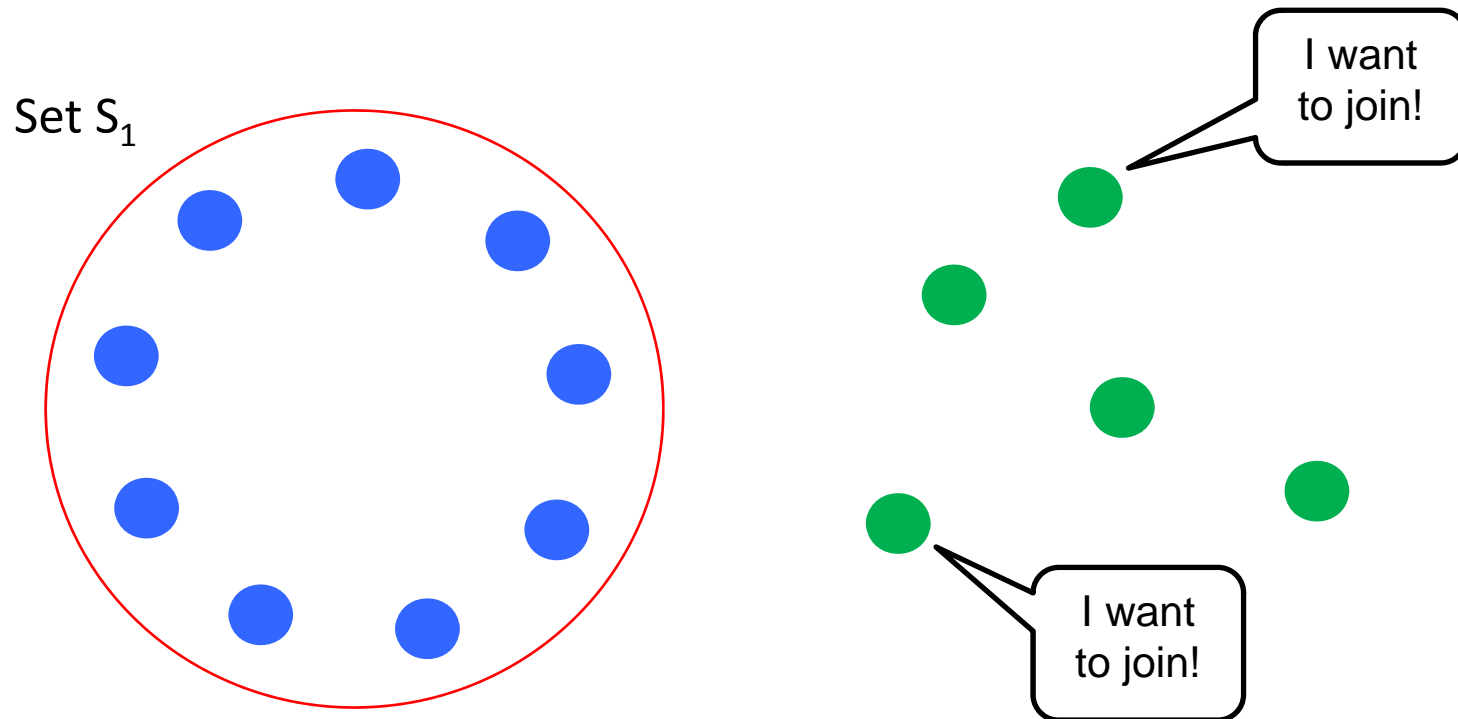


- $\min(2b'+2, n)$ time slots necessary

Join Detection



- joiners send join requests to with S_1 with probability $1/b'$

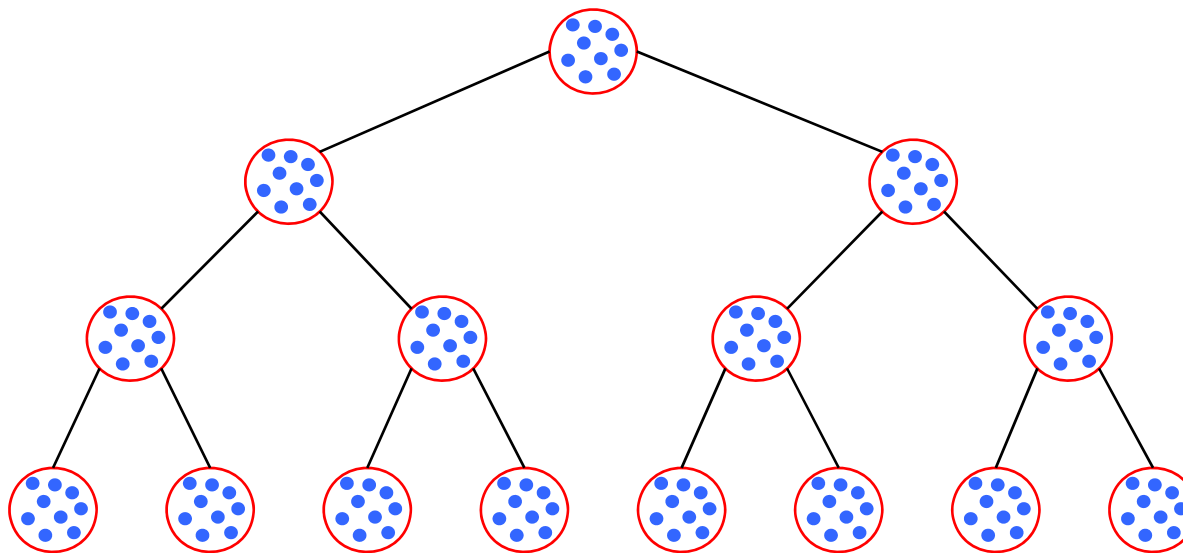


- b' in $\Omega(b) \rightarrow$ in constant number of rounds at least 1 joiner

Information Dissemination



- every set becomes vertex of balanced binary tree

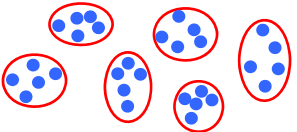
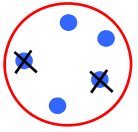
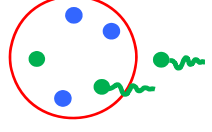
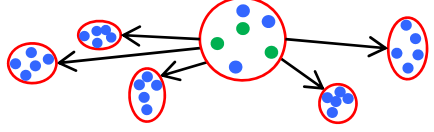
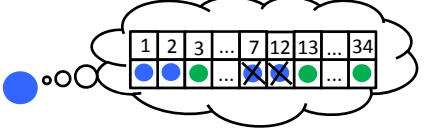


depth $\log n$

- every set forwards information on node v with smallest ID first
- information on v disseminated after $O(\log n)$ time slots
- all information disseminated after $O(\log n + b')$ time slots

Monitoring Algorithm



| Step | Time |
|--|------------------|
| <ul style="list-style-type: none"> $b' = \log n$ | $O(1)$ |
| <ul style="list-style-type: none"> partitioning  | $O(1)$ |
| <ul style="list-style-type: none"> crash detection  | $O(\min(b', n))$ |
| <ul style="list-style-type: none"> join detection  | $O(b' + \log n)$ |
| <ul style="list-style-type: none"> dissemination  | $O(b' + \log n)$ |
| <ul style="list-style-type: none"> update ID table  | $O(1)$ |
| <ul style="list-style-type: none"> double b' | $O(1)$ |

$\log(b/\log n)$
 times
 \Downarrow
 runtime
 $O(b + \log n)$

What if critical nodes crash?

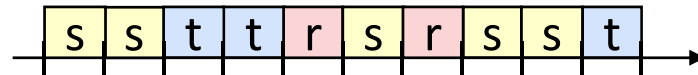


- in each set node which is responsible for communication with other sets
= representative
- all other nodes replacements
- replacements take over if representative does not send
- delay of at most b'
- still runtime of $O(b' + \log n)$ per round

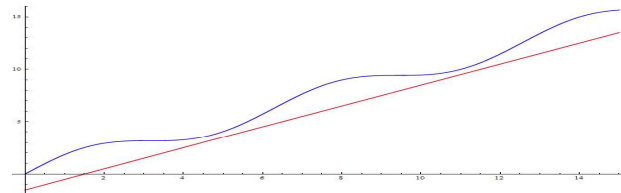
Conclusions & Future Work



- Model



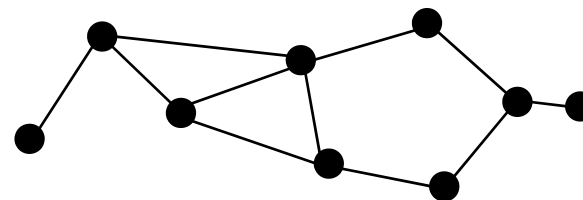
- Lower Bounds



- Monitoring Algorithm



- Future Work: Multi-hop



Thank You!

Questions & Comments?



*Stephan Holzer
Yvonne Anne Pignolet
Jasmin Smula
Roger Wattenhofer*