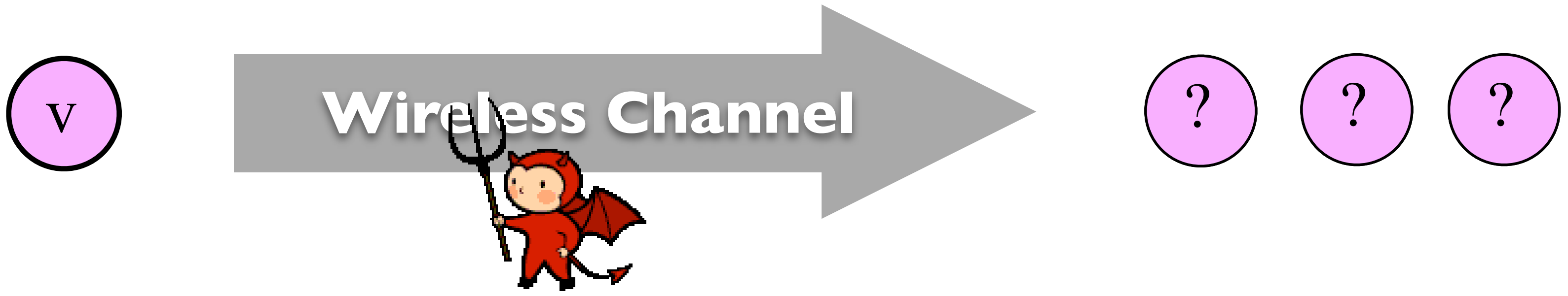
Gossiping in a Multi-Channel Radio Network

Shlomi Dolev Seth Gilbert Rachid Guerraoui Calvin Newport

Malicious Adversary + Radio = Dangerous

Malicious Adversary + Radio = Dangerous



How to circumvent the adversary?

Assume adversary cannot jam the channel.

- No collisions, no spoofing (e.g., [Koo 04]).

Assume adversary is probabilistic.

- Byzantine transmission fault occurs with probability $p < 1$ (e.g., [Pelc, Peleg 05]).

Assume adversary has limited broadcast power.

- Adversary can only broadcast β times (e.g., [Gilbert, Guerraoui, Newport 06]).

Goal for Today

Remove restrictions on adversary:

1. Adversary can *jam* the channel.
2. *Worst-case* adversary (not probabilistic).
3. *No bound* on adversarial broadcasts.

Multi-Channel Radio Network

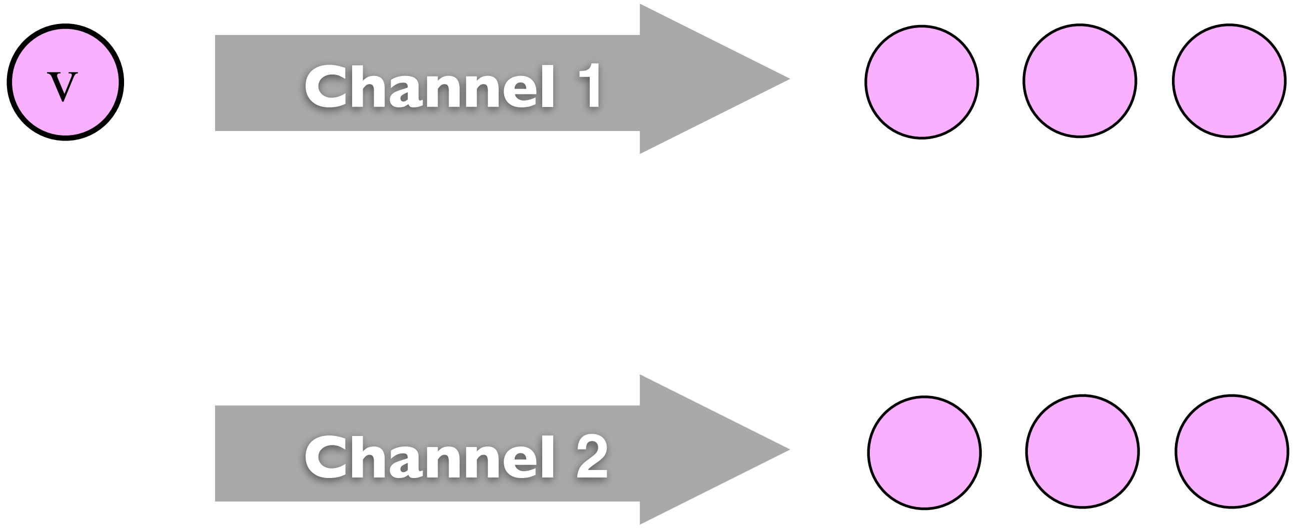
Basic Model:

- *n* nodes
- *c* channels, synchronous
- Each can *transmit* or *receive* on 1 channel per round.
- Only 1 can transmit per channel per round.

- 1 adversary
- Adversary can disrupt 1 channel per round.
- If adversary disrupts channel, *nothing* is received.

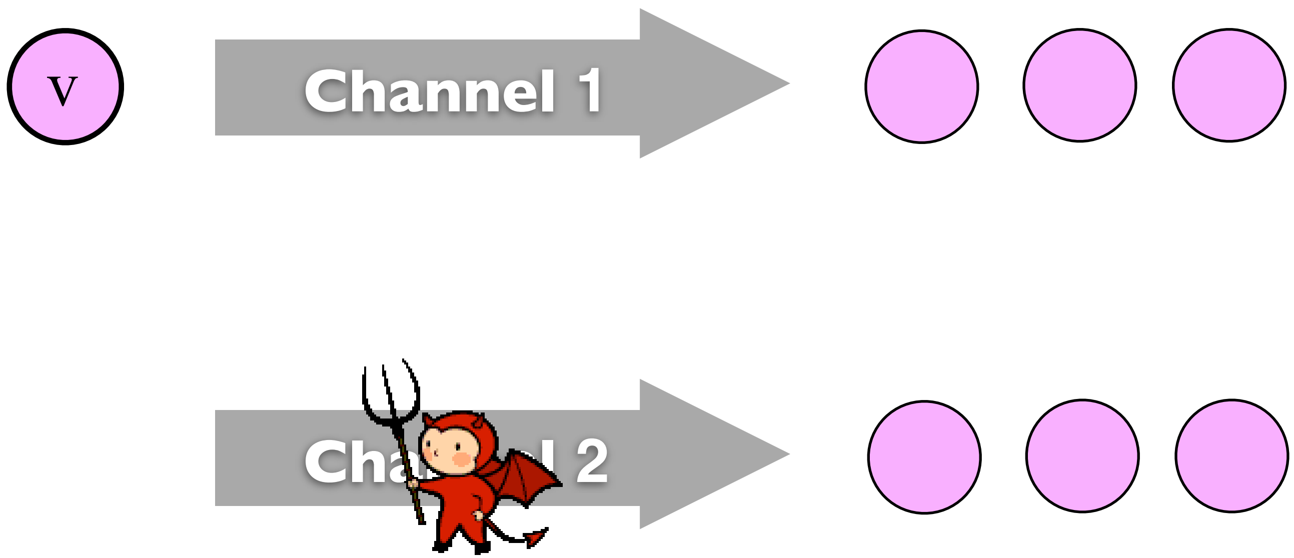
Multi-Channel Radio Network

Example: $c = 2, \ n = 7$



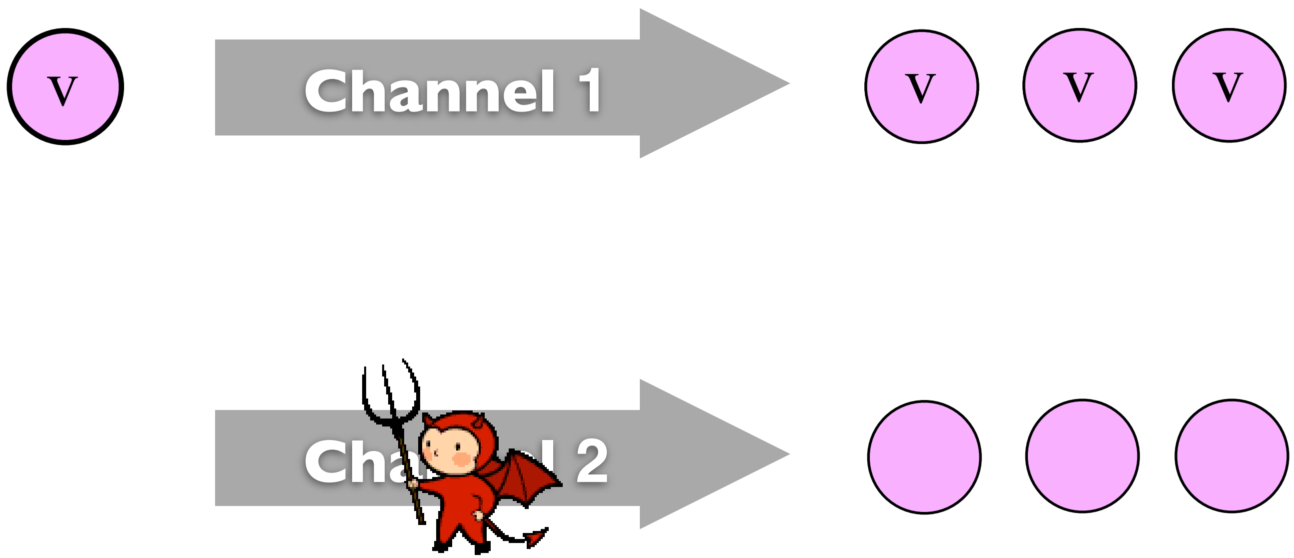
Multi-Channel Radio Network

Example: $c = 2, \; n = 7$



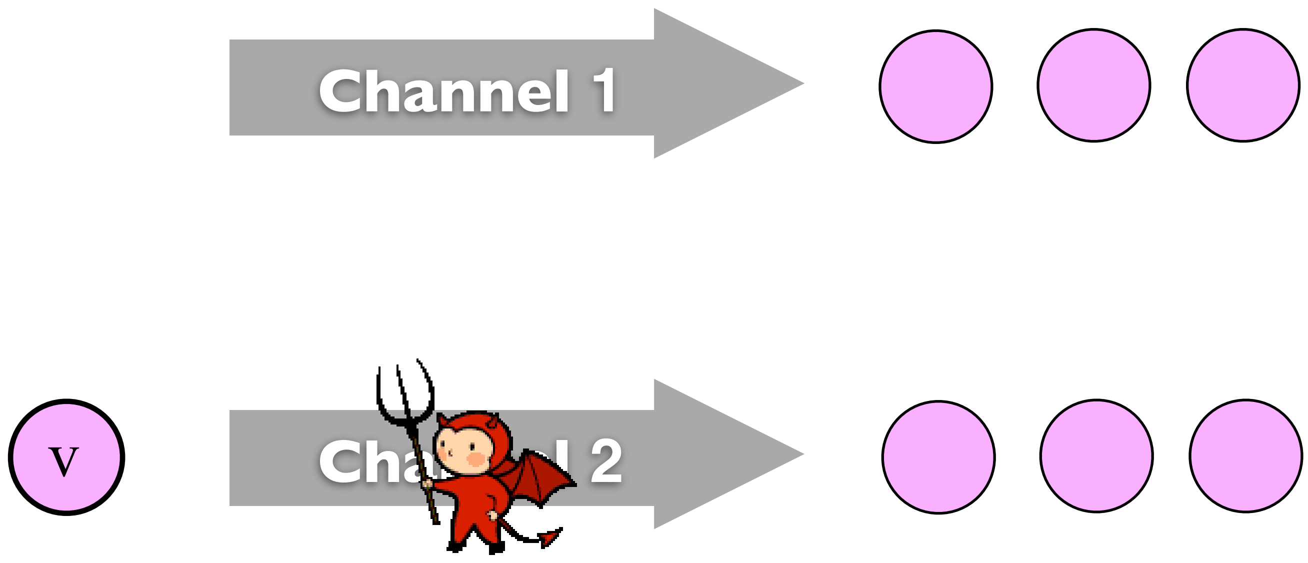
Multi-Channel Radio Network

Example: $c = 2,\ n = 7$



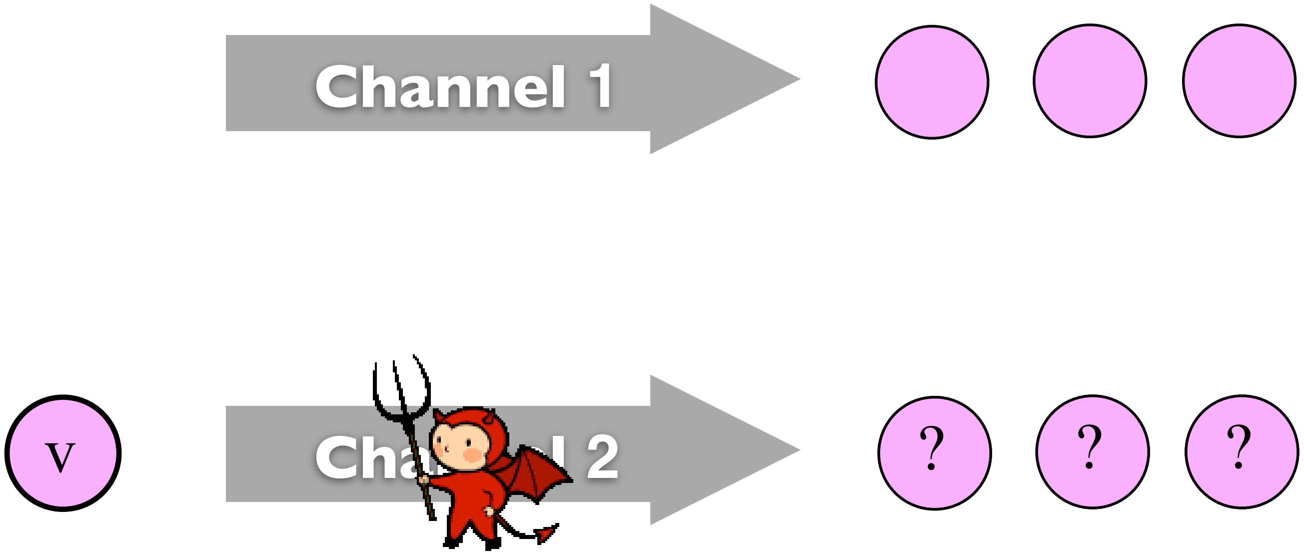
Multi-Channel Radio Network

Example: $c = 2, \ n = 7$



Multi-Channel Radio Network

Example: $c = 2, \ n = 7$



Gossip and ϵ–Gossip

Basic Problem:

- Each node begins round with value v_i.
- Eventually, every node learns all n values.

Gossip with ϵ error:

- All but ϵn of the values are successfully transmitted to $n-1$ nodes.

Deterministic, Oblivious Algorithms

Deterministic, Oblivious Algorithms

Observation: 0-Gossip is *impossible*.

Deterministic, Oblivious Algorithms

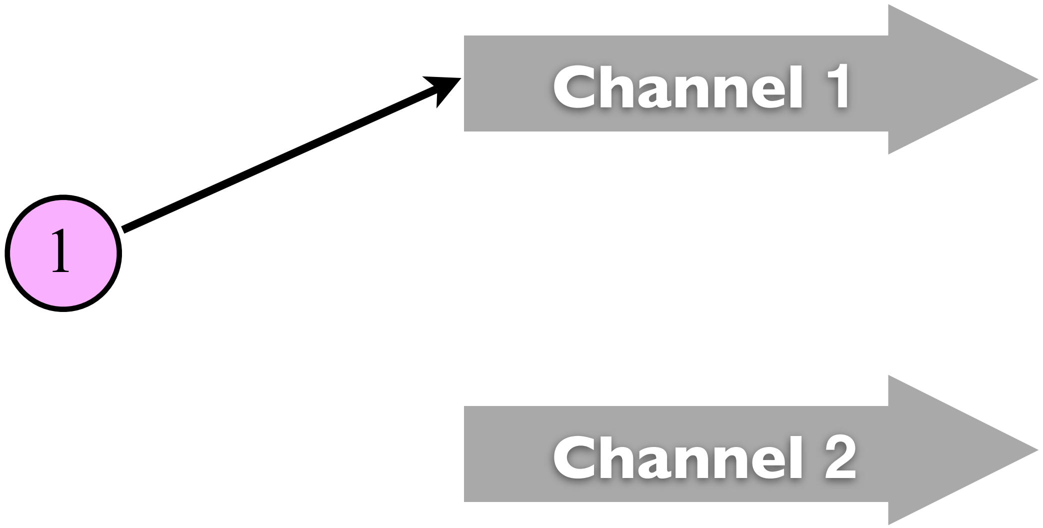
Observation: 0-Gossip is *impossible*.

- Adversary blocks node 1 every time it broadcasts:

Deterministic, Oblivious Algorithms

Observation: 0-Gossip is *impossible*.

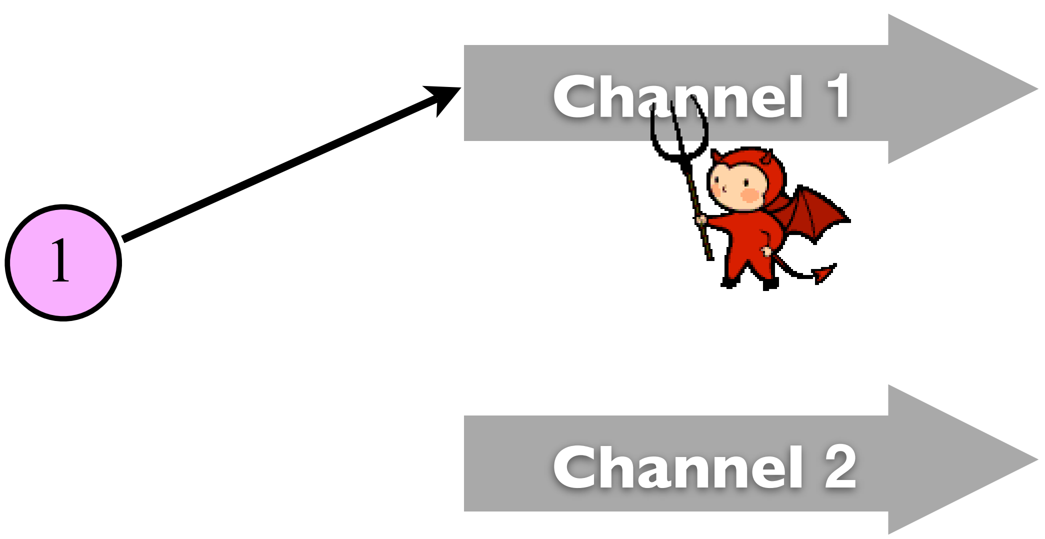
- Adversary blocks node 1 every time it broadcasts:



Deterministic, Oblivious Algorithms

Observation: 0-Gossip is *impossible*.

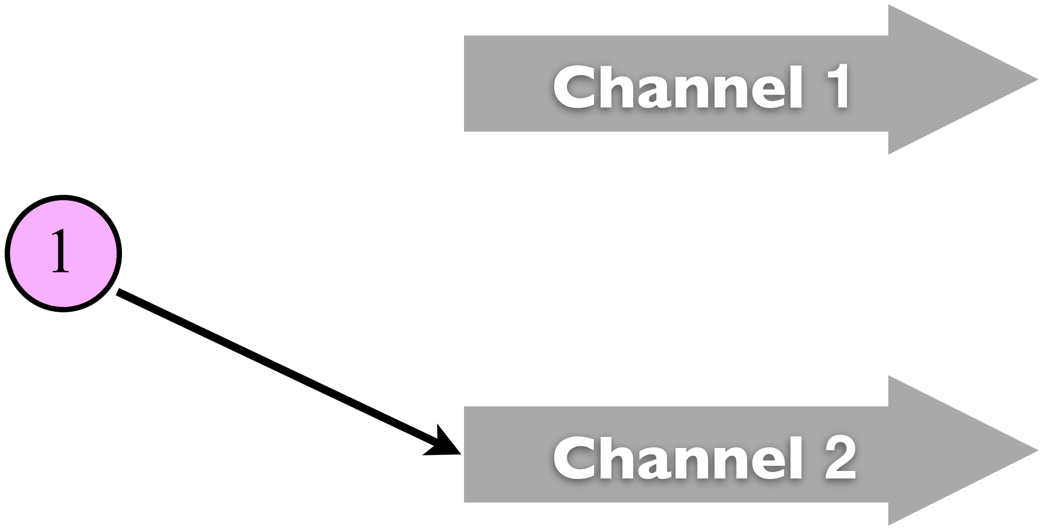
- Adversary blocks node 1 every time it broadcasts:



Deterministic, Oblivious Algorithms

Observation: 0-Gossip is *impossible*.

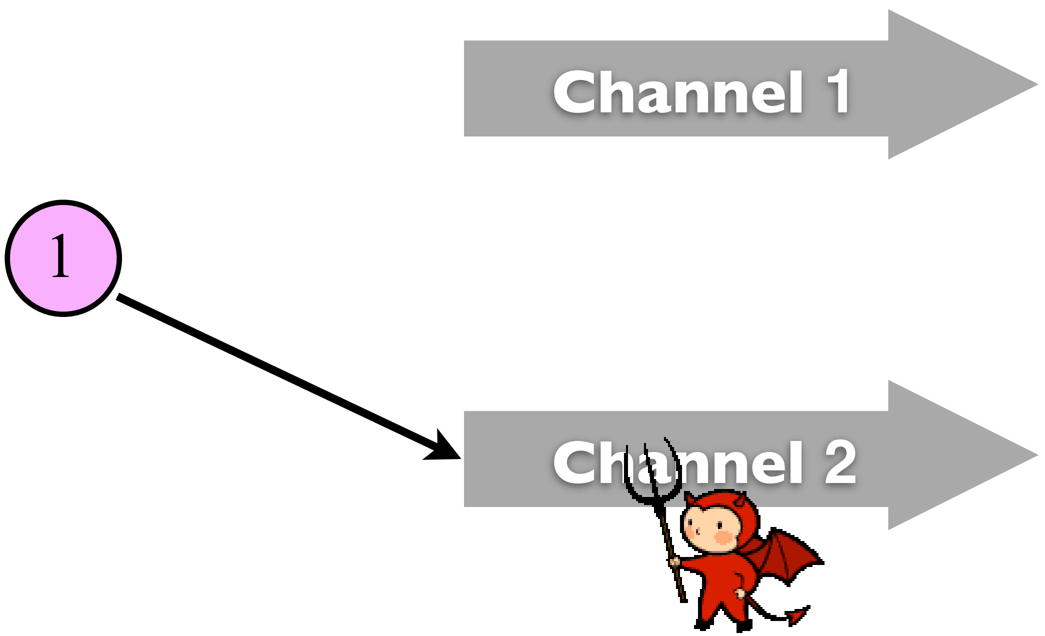
- Adversary blocks node 1 every time it broadcasts:



Deterministic, Oblivious Algorithms

Observation: 0-Gossip is *impossible*.

- Adversary blocks node 1 every time it broadcasts:

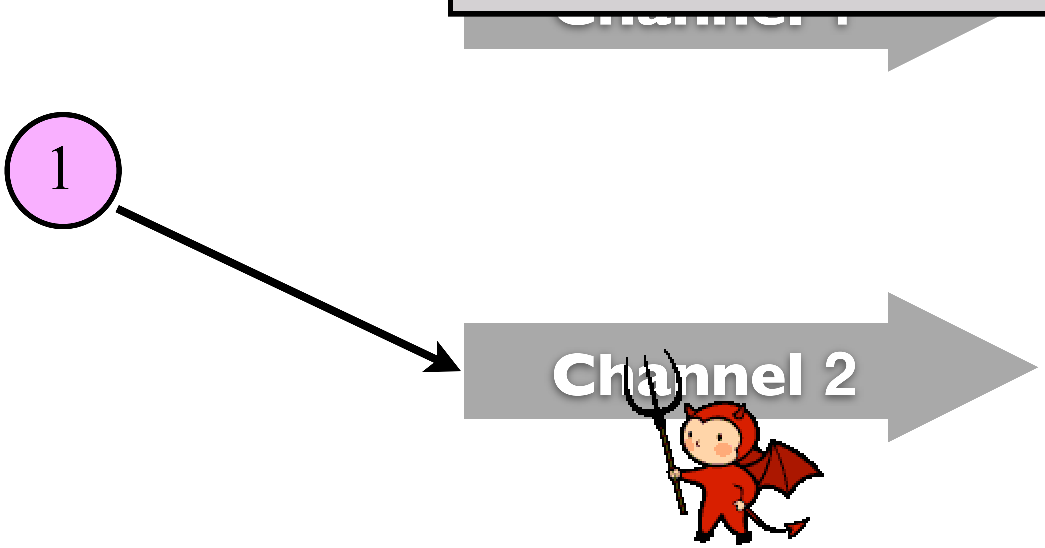


Deterministic, Oblivious Algorithms

Observation: 0-Gossip is *impossible*.

- Adversary blocks node 1 every time it broadcasts:

Node 1 never succeeds in broadcasting!



Gossip and ϵ–Gossip

Basic Problem:

- Each node begins round with value v_i.
- Eventually, every node learns all n values.

Gossip with ϵ error:

- All but ϵn of the values are successfully transmitted to $n - 1$ nodes.

Lower Bound

Theorem 1:

Every *deterministic*, *oblivious* algorithm for ϵ–Gossip requires at least $\Theta\left(\frac{n}{\epsilon c^2}\right)$ rounds (for small ϵ).

n	:	number of nodes
c	:	number of channels
ϵ	:	all but ϵn values are successfully transmitted

Lower Bound

Theorem 1:

Every *deterministic*, *oblivious* algorithm for ϵ–Gossip requires at least $\Theta\left(\frac{n}{\epsilon c^2}\right)$ rounds.

Example:

- Gossiping all but one value requires $\Theta\left(\frac{n^2}{c^2}\right)$ rounds.

Lower Bound

Key Idea: (example: $c = 2$)

- If two nodes *a* and *b* never broadcast in the same round, the adversary can always block both of them.
  - Adversary disrupts node *a* when *a* broadcasts.
  - Adversary disrupts node *b* when *b* broadcasts.
- Thus disseminating all but one rumor requires at least $\binom{n}{2}$ rounds.
- Generalize for other values of $c, \epsilon \ldots$

Lower Bound

Consider the graph G:

- There is an edge between *i* and *j* if (and only if) nodes *i* and *j* never broadcast in the same round.

- If there exists a clique of size *k* in G, then the adversary can prevent k nodes from broadcasting.

Lower Bound

Consider the graph G:

- There is an edge between *i* and *j* if (and only if) nodes *i* and *j* never broadcast in the same round.

Claim: If algorithm *A* solves ϵ–Gossip in *r* rounds, then:
(1) graph G has no clique larger than ϵn.
(2) graph G is missing at most $c^2 r$ edges.

Lower Bound

Turan's Theorem:

If graph G has no clique of size k+1:

Then graph G has at most:

$$\left(1 - \frac{1}{k}\right)\frac{n^2}{2}$$

edges.

Lower Bound

Corollary:

If graph G has no clique of size ϵn :

Then graph G is missing at least:

$$\binom{n}{2} - \left(1 - \frac{1}{\epsilon n - 1}\right)\frac{n^2}{2} \quad = \quad \frac{n^2}{2\epsilon n - 2} - \frac{n}{2}$$

$$= \boxed{\Theta\left(\frac{n}{\epsilon}\right) \text{ edges.}}$$

Lower Bound

Theorem 1:

Every *deterministic, oblivious* ϵ–Gossip algorithm requires at least $\Theta\left(\frac{n}{\epsilon c^2}\right)$ rounds.

Proof:

- Graph **G** has no clique of size ϵn and is missing at least $c^2 r \geq \Theta\left(\frac{n}{\epsilon}\right)$ edges, r = # of rounds.

Turan's Theorem

Upper Bound

Theorem 2:

There exists a *deterministic*, *oblivious* algorithm for ϵ–Gossip that terminates in $\Theta\left(\frac{n}{\epsilon c^2}\right)$ rounds (for small ϵ).

n	:	number of nodes
c	:	number of channels
ϵ	:	all but ϵn values are successfully transmitted

Upper Bound

Construct the algorithm in two steps:

1. Data Collection
   - aggregate data at a small number of nodes.

2. Data Dissemination
   - broadcast data from a small number to the rest.

Upper Bound

Data Collection

Partition the nodes into two sets:

1. Listeners:
   - $2c$ nodes, two per channel.
2. Broadcasters:
   - Divide into ϵn sets of size $1/\epsilon$.
   - For each, all but 1 node succeeds in broadcasting.
   - End result: all but ϵn values are known to a listener.

Upper Bound

Data Collection

- For each set of size $1/\epsilon$:
  - Divide into $2/\epsilon c$ subsets of size $c/2$.
  - For every pair of subsets, assign 1 round.

- Result:
  - Only one node in the set can be blocked.
  - Running time: $\binom{2/\epsilon c}{2} = \Theta\left(\frac{1}{\epsilon^2 c^2}\right)$

Upper Bound

Data Collection

- Overall running time (so far):

$$\epsilon n \cdot \Theta\left(\frac{1}{\epsilon^2 c^2}\right) = \Theta\left(\frac{n}{\epsilon c^2}\right)$$

number of subsets

rounds per subset

Upper Bound

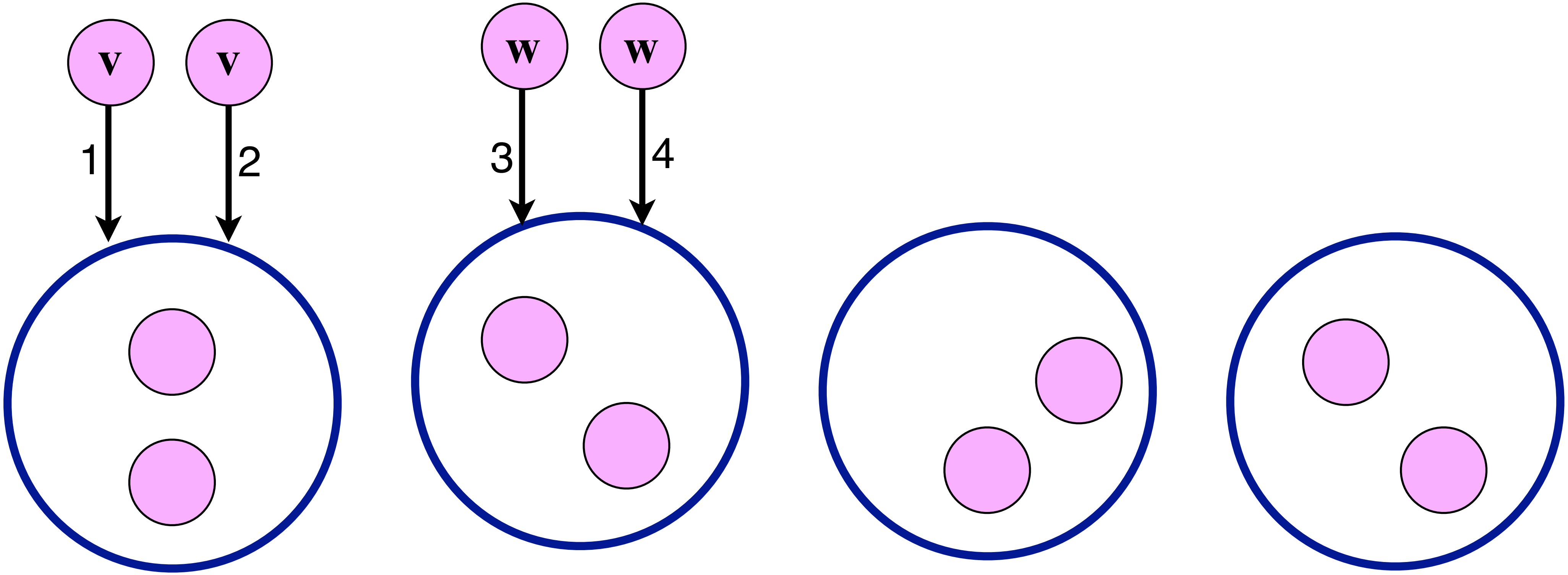
Data Dissemination

- Step 1: Divide the nodes into c sets, 1 per channel.
- Step 2: For each channel, disseminate data from 2 listeners to all the nodes in that channel's set.
- Step 3: Merge channel sets pairwise.

Upper Bound

Data Dissemination

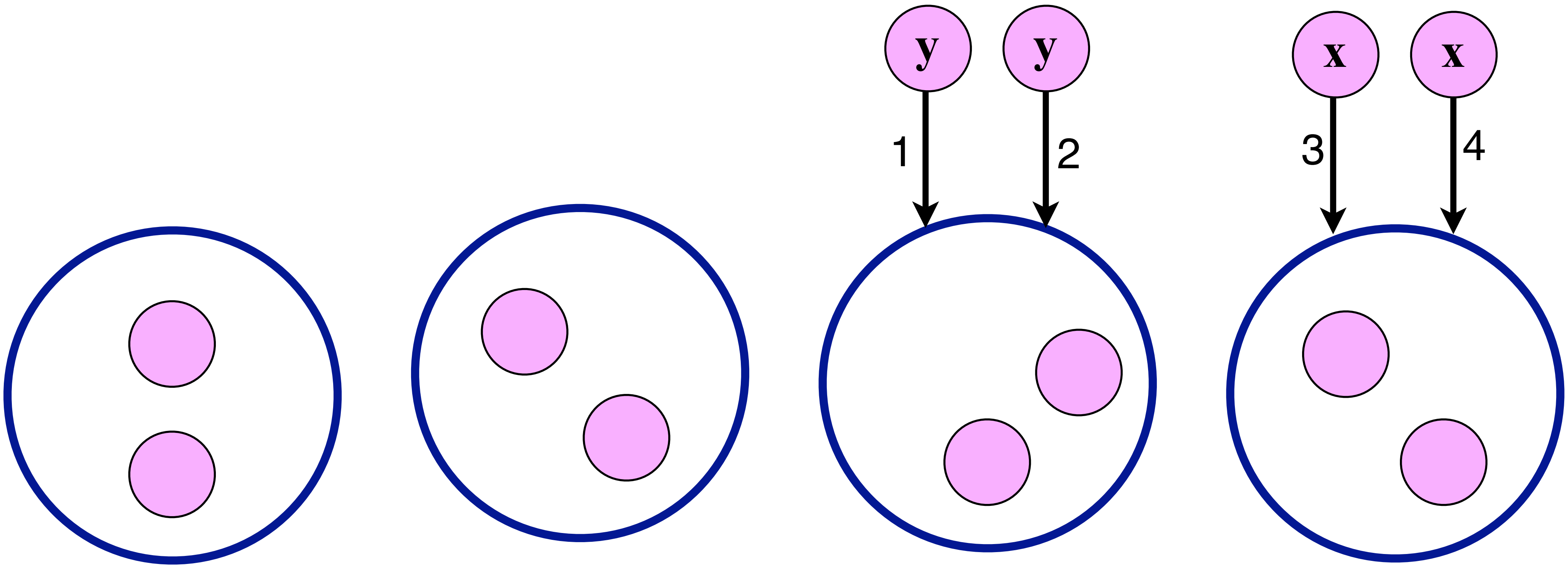
- Step 2: For each channel, disseminate data from 2 listeners to the nodes in that channel's set.



Upper Bound

Data Dissemination

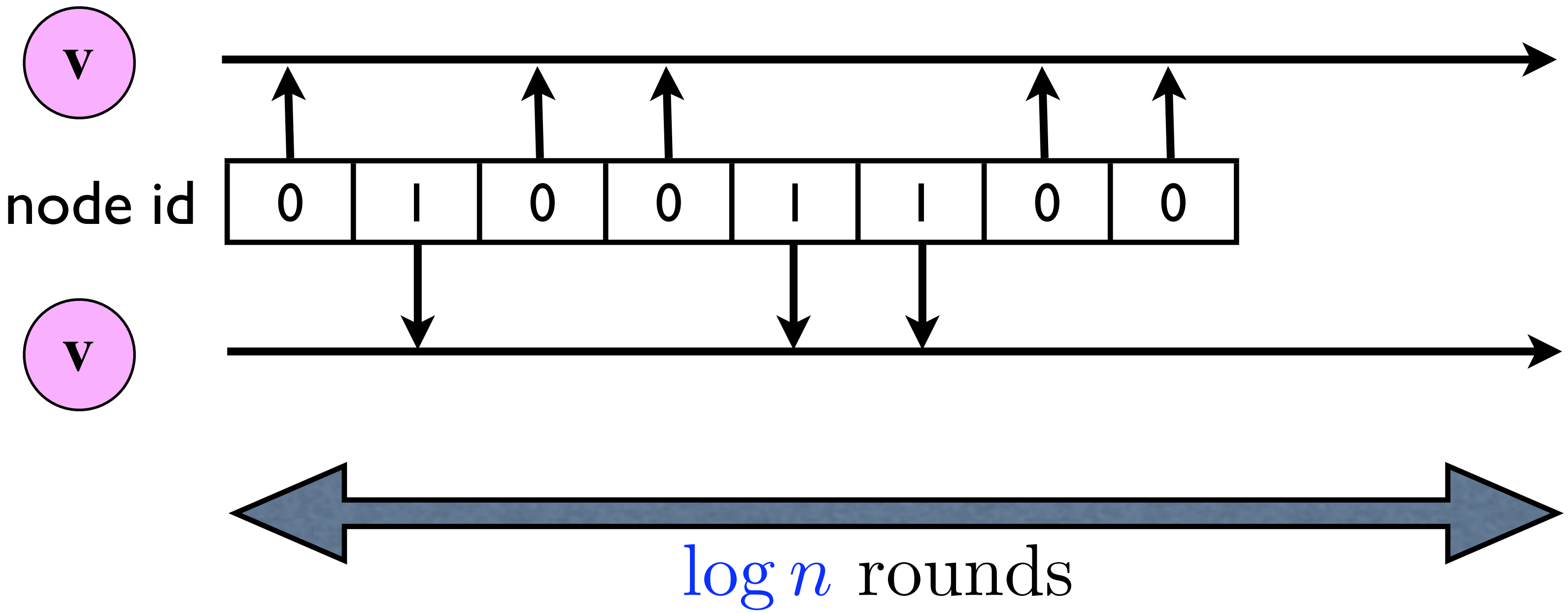
- Step 2: For each channel, disseminate data from 2 listeners to the nodes in that channel's set.



Upper Bound

Data Dissemination

- Step 2: For each channel, disseminate data from 2 listeners to the nodes in that channel's set.



Upper Bound

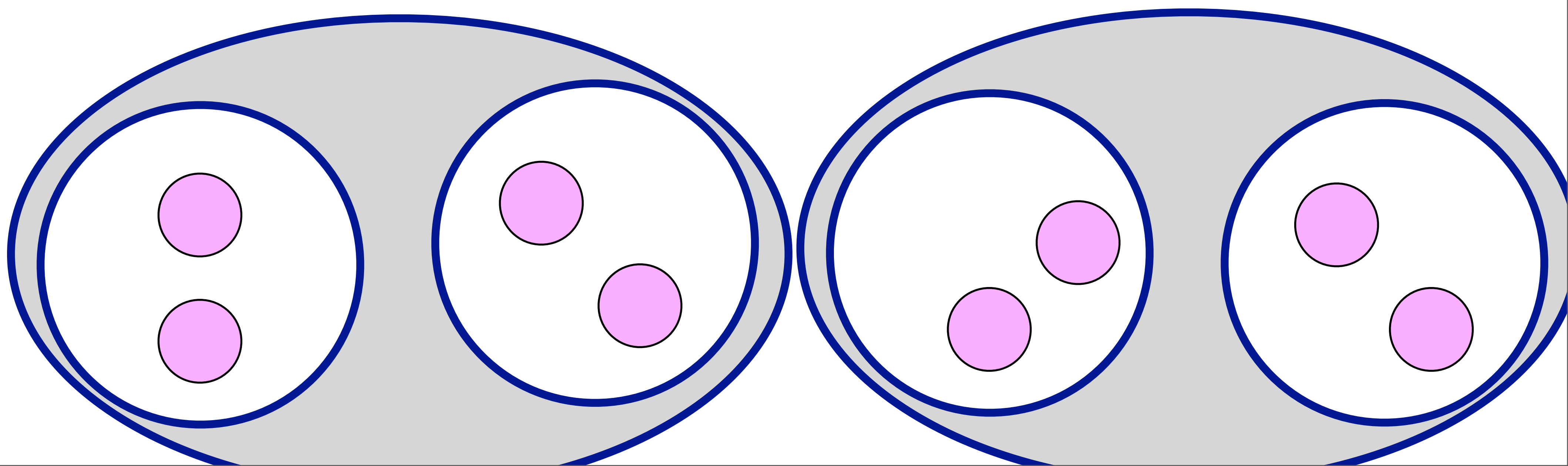
Data Dissemination

- Step 1: Divide the nodes into c sets, 1 per channel.
- Step 2: For each channel, disseminate data from 2 listeners to the nodes in that channel's set.
- Step 3: Merge channel sets together.

Upper Bound

Data Dissemination

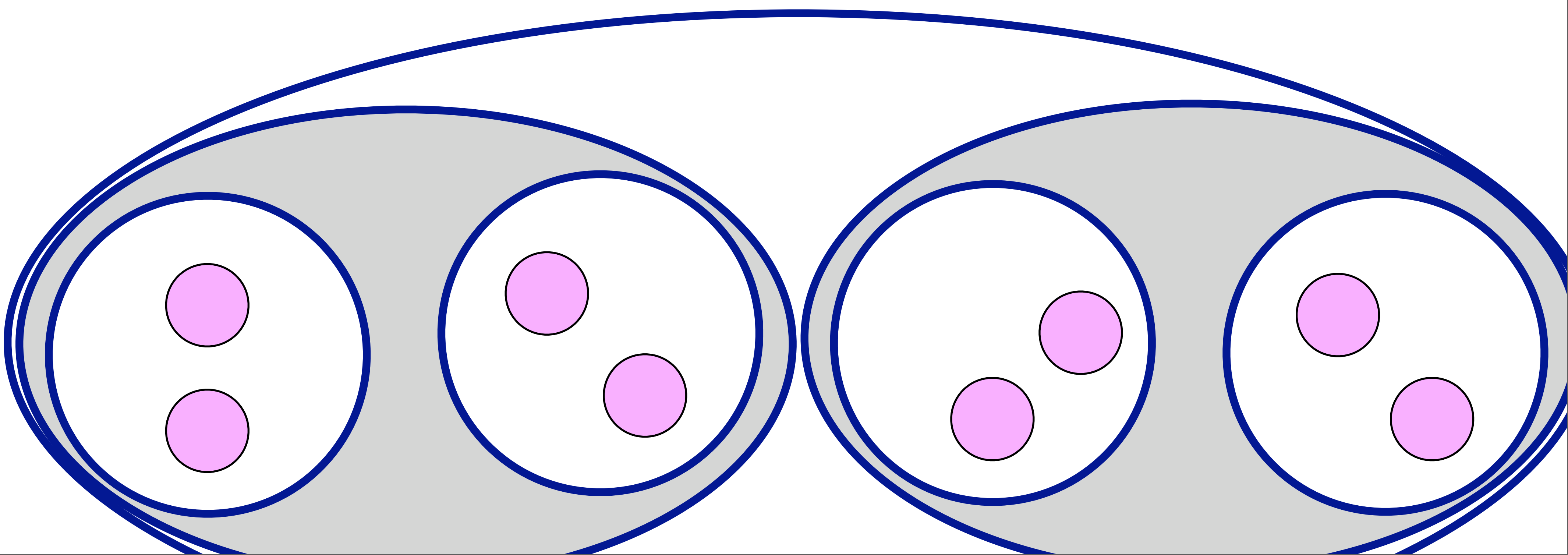
- Step 3: Merge channel sets together.



Upper Bound

Data Dissemination

- Step 3: Merge channel sets together.



Upper Bound

Data Dissemination

- Step 1: Divide the nodes into *c* sets, 1 per channel.
- Step 2: For each channel, disseminate data from 2 listeners to the nodes in that channel's set.
  - Number of rounds: $\log n$
- Step 3: Merge channel sets together.
  - Number of rounds: $\log^2 n$

Upper Bound

Data Collection + Data Dissemination

- Running time (for small ϵ):

$$\Theta\left(\frac{n}{\epsilon c^2} + \log^2 n\right) = \Theta\left(\frac{n}{\epsilon c^2}\right)$$

More Results

1. Actual bound 1-channel adversary:

$$\max\left(\Theta\left(\frac{(1-\epsilon)n}{c-1}+\log_c n\right),\Theta\left(\frac{(1-\epsilon)n}{\epsilon c^2}\right)\right)$$

2. Multi-channel adversary:

$$O\left(\frac{ne^{t+1}}{c\epsilon^t}+c(t+1)^t\log^{t+1}n\right)$$

- (Bound is tight when $\epsilon = t/n$.)

3. Adversary that corrupts nodes.
   - More listeners...
   - Increase running time by factor of t.

Open Questions and Ongoing Research

- *Adaptive* algorithms.
- *Secure* algorithms.
- *Multi-hop* networks.
- *Energy* efficiency and *energy* limited adversary.