Sensor Network Gossiping or How to Break the Broadcast Lower Bound

Martín Farach-Colton[1] Miguel A. Mosteiro[1,2]

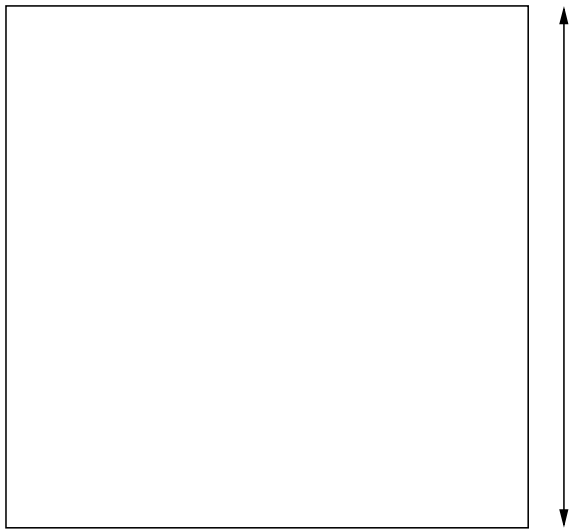
[1]Department of Computer Science
Rutgers University

[2]LADyR (Distributed Algorithms and Networks Lab)
Universidad Rey Juan Carlos

ISAAC 2007

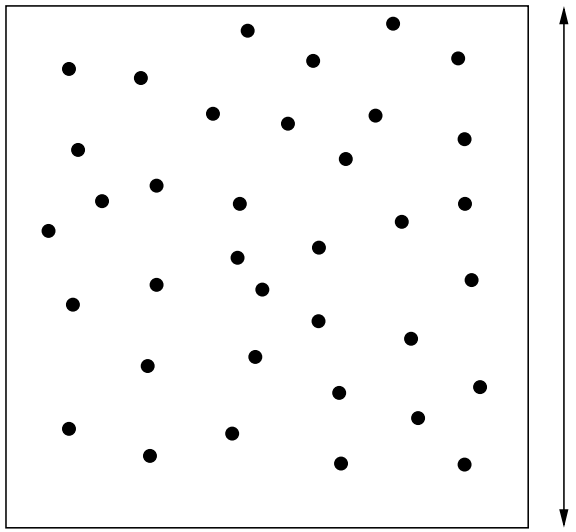
Information Dissemination in Radio Networks

Radio Network = abstraction of a radio communication network



Information Dissemination in Radio Networks

Radio Network = abstraction of a radio communication network



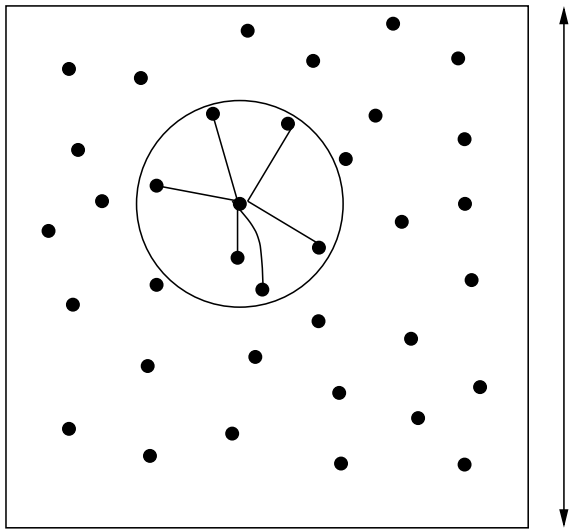
k nodes
hold a piece of information to disseminate.

- $k = 1 \rightarrow$ *Broadcast* [BGI'92,KM'98]
- $k = n: \rightarrow$ *Gossiping* [CGLP'01,LP'02]
- k arbitrary: $\rightarrow$ *k-selection* [K'05]

We study
Gossiping in Sensor Networks

Information Dissemination in Radio Networks

Radio Network = abstraction of a radio communication network



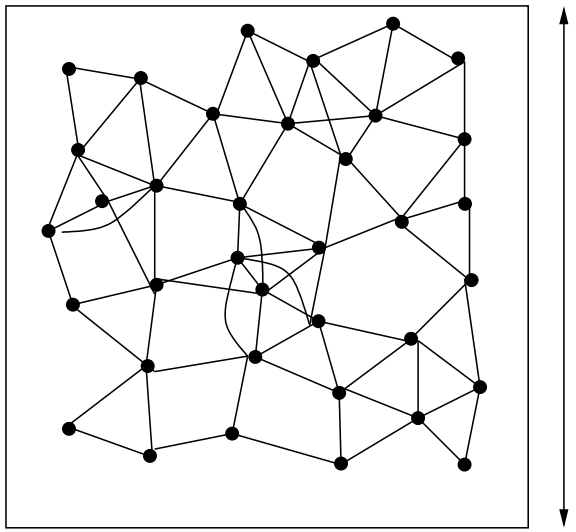
k nodes
hold a piece of information to disseminate.

- $k = 1 \rightarrow$ *Broadcast* [BGI'92,KM'98]
- $k = n: \rightarrow$ *Gossiping* [CGLP'01,LP'02]
- k arbitrary: $\rightarrow$ *k-selection* [K'05]

We study
Gossiping in Sensor Networks

Information Dissemination in Radio Networks

Radio Network = abstraction of a radio communication network



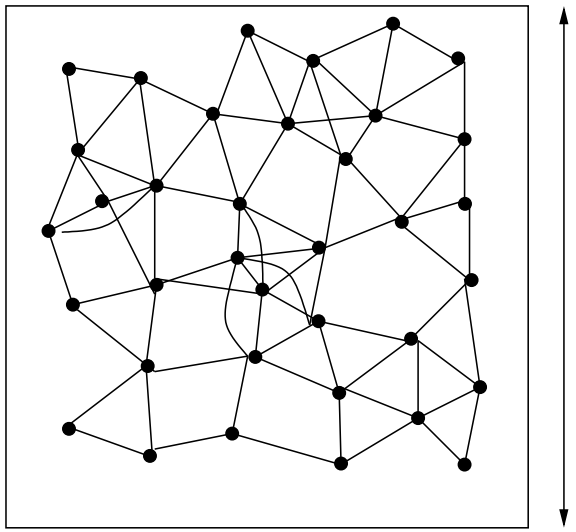
k nodes
hold a piece of information to disseminate.

- $k = 1 \rightarrow$ *Broadcast* [BGI'92,KM'98]
- $k = n: \rightarrow$ *Gossiping* [CGLP'01,LP'02]
- k arbitrary: $\rightarrow$ *k-selection* [K'05]

We study
Gossiping in Sensor Networks

Information Dissemination in Radio Networks

Radio Network = abstraction of a radio communication network



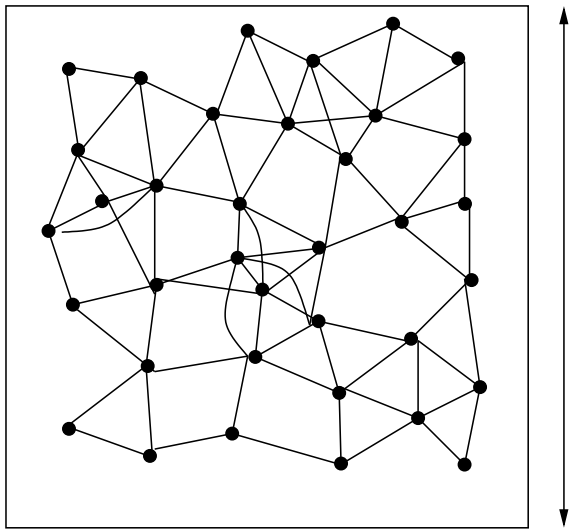
k nodes
hold a piece of information to diseminate.

- $k = 1 \rightarrow$ *Broadcast* [BGI'92,KM'98]
- $k = n$: $\rightarrow$ *Gossiping* [CGLP'01,LP'02]
- k arbitrary: $\rightarrow$ *k-selection* [K'05]

We study
Gossiping in Sensor Networks

Information Dissemination in Radio Networks

Radio Network = abstraction of a radio communication network



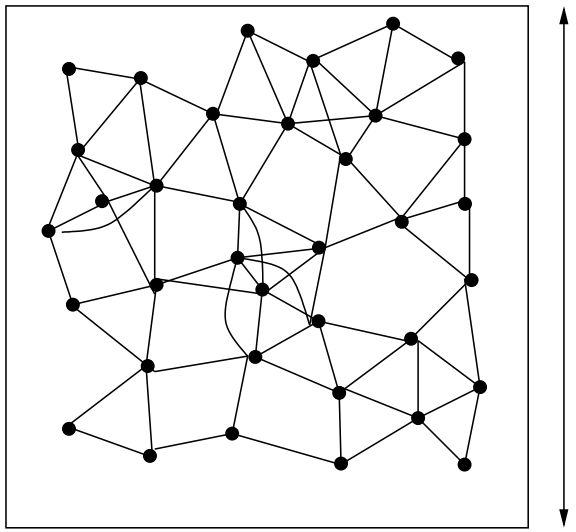
k nodes
hold a piece of information to diseminate.

- $k = 1 \rightarrow$ *Broadcast* [BGI'92,KM'98]
- $k = n$: $\rightarrow$ *Gossiping* [CGLP'01,LP'02]
- k arbitrary: $\rightarrow$ *k-selection* [K'05]

We study
Gossiping in Sensor Networks

Information Dissemination in Radio Networks

Radio Network = abstraction of a radio communication network



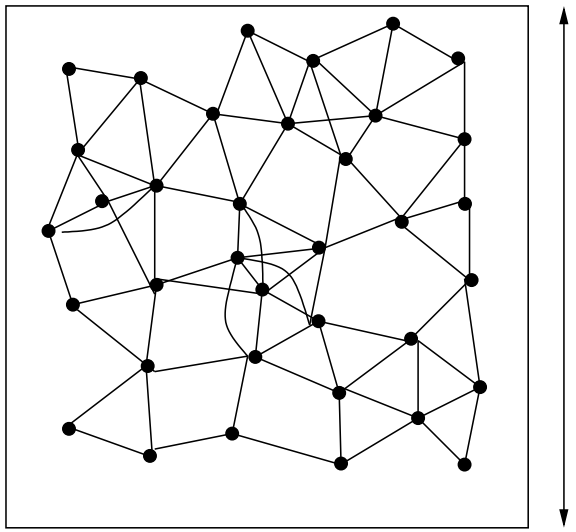
k nodes
hold a piece of information to diseminate.

- $k = 1 \rightarrow$ *Broadcast* [BGI'92,KM'98]
- $k = n$: $\rightarrow$ *Gossiping* [CGLP'01,LP'02]
- k arbitrary: $\rightarrow$ *k-selection* [K'05]

We study
Gossiping in Sensor Networks

Information Dissemination in Radio Networks

Radio Network = abstraction of a radio communication network



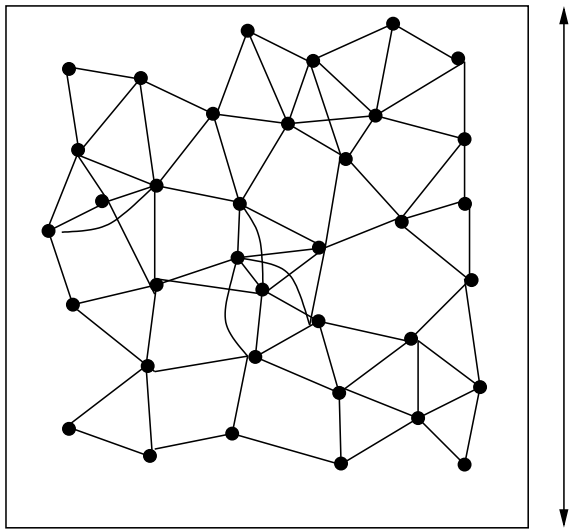
k nodes
hold a piece of information to diseminate.

- $k = 1 \rightarrow$ *Broadcast* [BGI'92,KM'98]
- $k = n$: $\rightarrow$ *Gossiping* [CGLP'01,LP'02]
- k arbitrary: $\rightarrow$ *k-selection* [K'05]

We study
Gossiping in Sensor Networks

Information Dissemination in Radio Networks

Radio Network = abstraction of a radio communication network



k nodes
hold a piece of information to diseminate.

- $k = 1 \rightarrow$ *Broadcast* [BGI'92,KM'98]
- $k = n$: $\rightarrow$ *Gossiping* [CGLP'01,LP'02]
- k arbitrary: $\rightarrow$ *k-selection* [K'05]

We study
Gossiping in Sensor Networks

A Sensor Network

Sensor Node Capabilities

- processing
- sensing
- communication

Sensor Node Limitations

- range
- memory
- life cycle

A Sensor Network

Sensor Node Capabilities

- processing
- sensing
- communication

Sensor Node Limitations

- range
- memory
- life cycle

A Sensor Network

Sensor Node Capabilities

- processing
- sensing
- communication

Sensor Node Limitations

- range
- memory
- life cycle

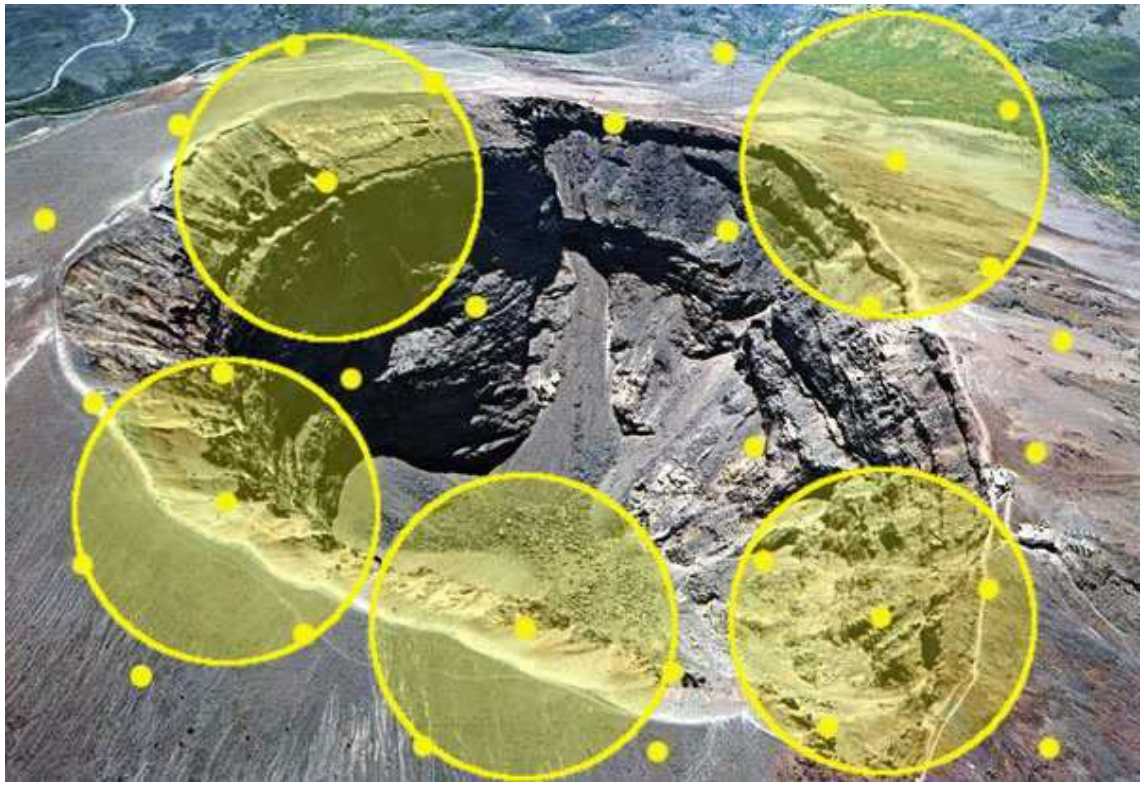
A Sensor Network

Sensor Node Capabilities

- processing
- sensing
- communication

Sensor Node Limitations

- range
- memory
- life cycle



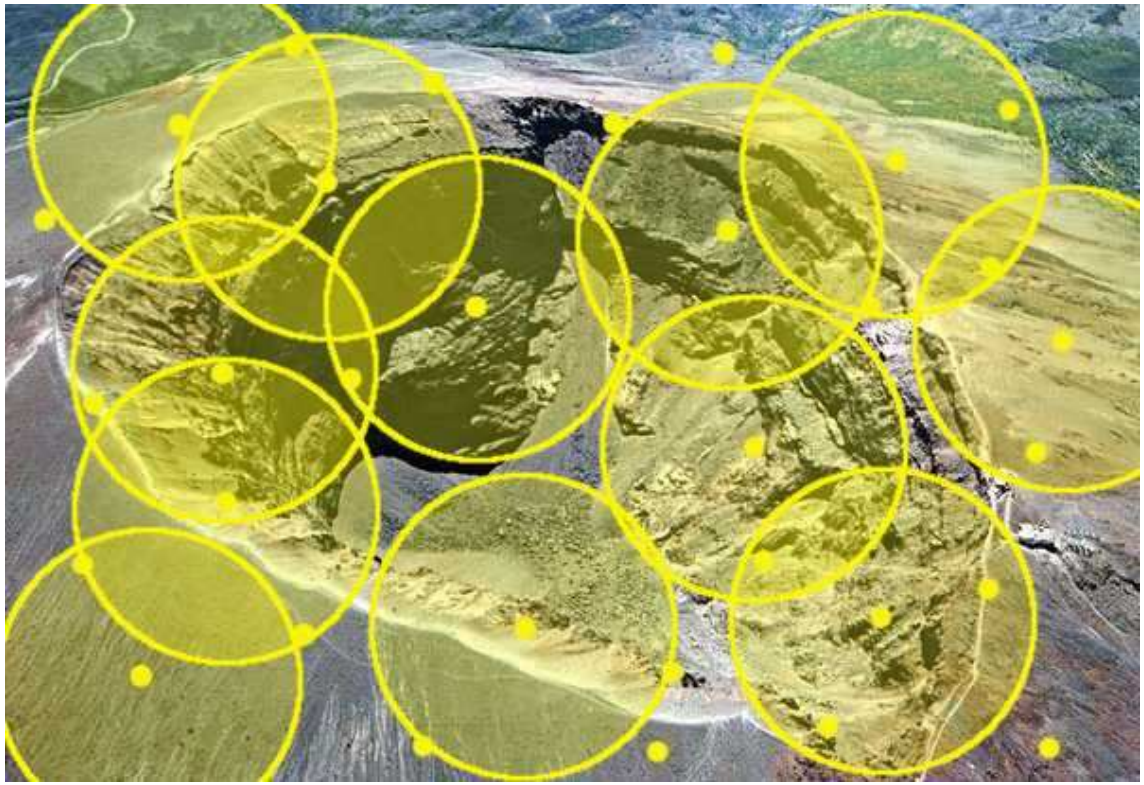
A Sensor Network

Sensor Node Capabilities

- processing
- sensing
- communication

Sensor Node Limitations

- range
- memory
- life cycle



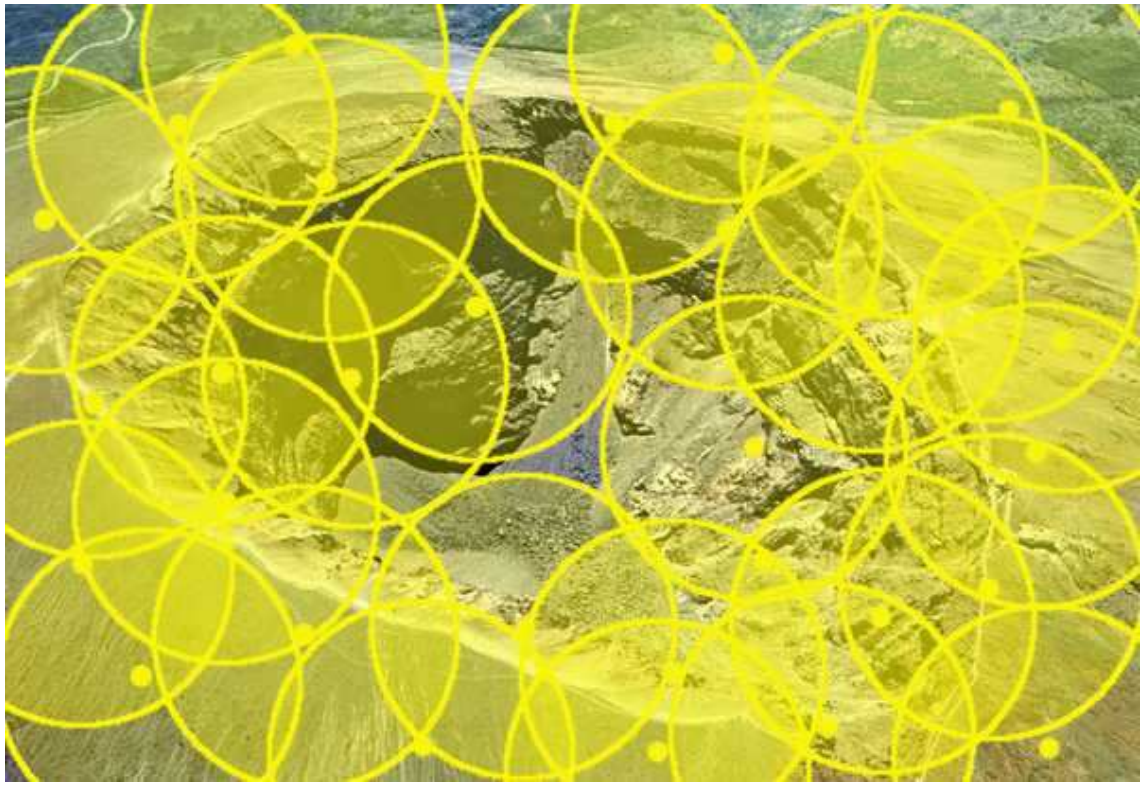
A Sensor Network

Sensor Node Capabilities

- processing
- sensing
- communication

Sensor Node Limitations

- range
- memory
- life cycle



Related Work
Upper Bounds

- Symmetric Radio Networks:

BII'93 $O(n \log^2 n)$ expected (BFS tree).
CGLP'01 same, w.h.p.

- Asymmetric connected Radio Networks:

CGR'01 $O(n \log^3 n \log(n/\epsilon))$ with prob $1 - \epsilon$ and $O(n \log^4 n)$ expected (limited broadcast doubles message copies per phase).
LP'02 same, reduced by a log factor (limited broadcast is randomized).
CR'03 $O(n \log^2 n)$ w.h.p. (linear randomized broadcast by special distribution).
CGR'00 $O(n^{3/2} \log^2 n)$ (deterministic, selecting sequences).

- ALL: globally synchronous, and $\Omega(nm)$ memory size, all but first: $\Omega(nm)$ message size.

- Sensor Networks

R'07 $O(\sqrt{n} \log n)$ w.h.p. in RGGs (claimed optimal using KM's lower bound, but includes pre-coloring).

Related Work
Lower Bounds

- *Gossiping*:

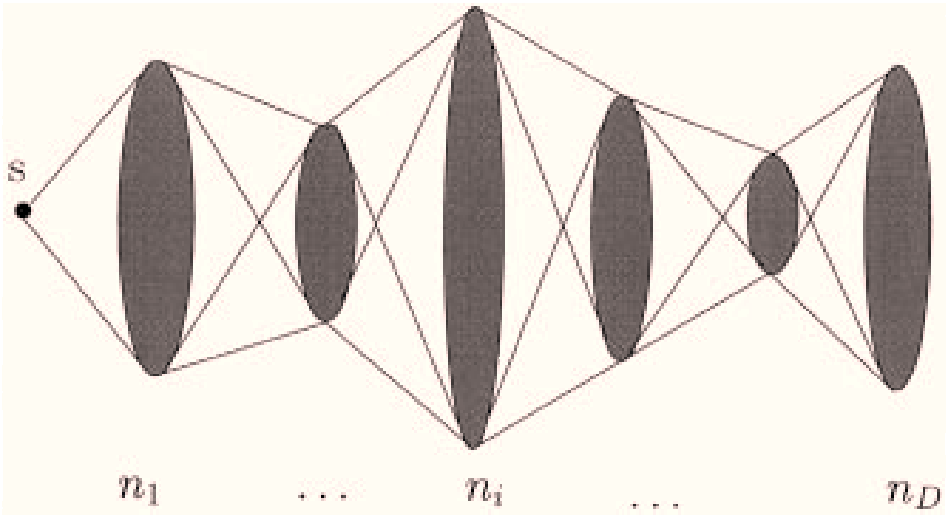
CGLP'01 deterministic oblivious (no history): $\geq n^2/2 - n/2 + 1$
fair (same p_{trans}) protocols: $\forall n \leq q \leq n^2/2$, $\exists$ asymmetric network s.t. $\Omega(q)$ expected.
GP'02 $\Omega(n^2)$ asymmetric networks
$\Omega(n \log n)$ symmetric networks not embeddable in GG.

- *Broadcast* (no preprocessing):

BDP'97 $\Omega(D \log n)$ globally synchronous, nodes know message history.
CMS'01 $\Omega(n \log D)$ symmetric networks, nodes are not synchronized.
KP'04 $\Omega(n^{1/4})$, diameter 4.
KM'98 $\Omega(D \log(n/D))$ expected (best, more on this...)

Related Work
Broadcast Lower Bound

[KM'98] proved $\Omega(D \log(n/D))$ expected, showing a layered structure



Crucial assumption:

"...any other processor is inactive"

"until receiving a message for the first time."

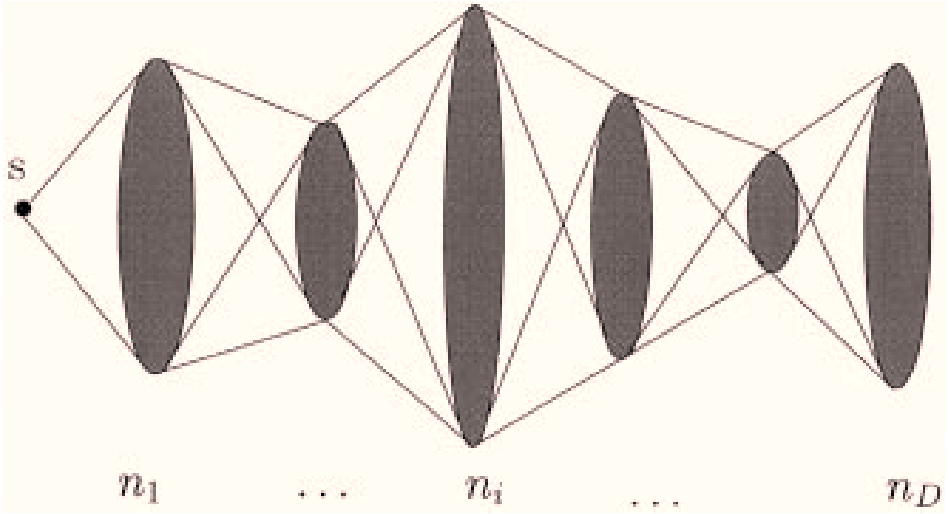
Crucial in proof:

all layer nodes run same uniform protocol,

upon receiving the broadcast message.

Related Work
Broadcast Lower Bound

[KM'98] proved $\Omega(D \log(n/D))$ expected, showing a layered structure



Crucial assumption:

"...any other processor is inactive"

"until receiving a message for the first time."

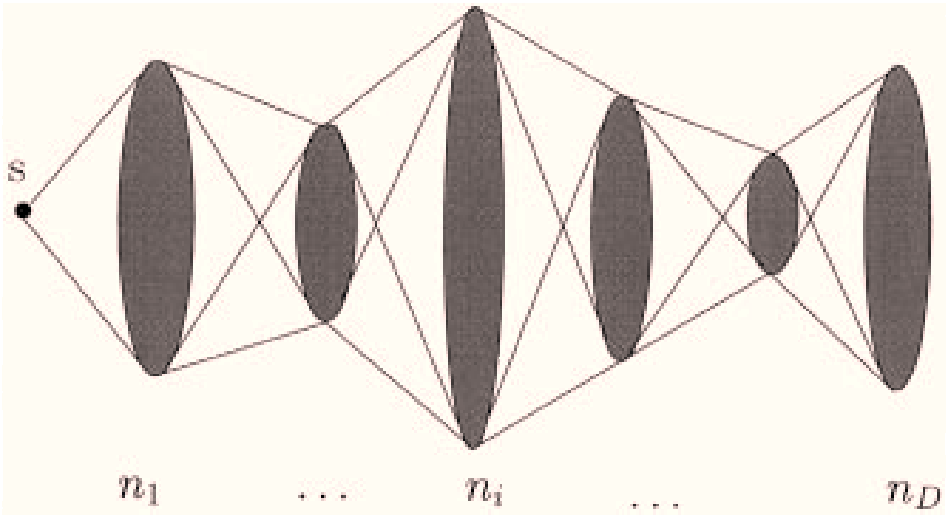
Crucial in proof:

all layer nodes run same uniform protocol,

upon receiving the broadcast message.

Related Work
Broadcast Lower Bound

[KM'98] proved $\Omega(D \log(n/D))$ expected, showing a layered structure



Crucial assumption:

"...any other processor is inactive"

"until receiving a message for the first time."

Crucial in proof:

all layer nodes run same uniform protocol,

upon receiving the broadcast message.

Node Constraints Model

Sensor Networks

The Weak Sensor Model
[BGI 92, FCFM 05]

- Local synchronism.
- Adversarial wake-up schedule.
- Low-info channel contention:
  - Radio tx on a shared channel.
  - No collision detection.
  - Non-simultaneous rx and tx.
- Constant memory size.
- Limited life cycle.
- Short transmission range.
- Discrete tx power range.
- One channel of communication.
- No position information.
- Unreliability.

$tx = transmission.$
$rx = reception.$

Node Constraints Model

Sensor Networks

The Weak Sensor Model
[BGI 92, FCFM 05]

- Local synchronism.
- Adversarial wake-up schedule.
- Low-info channel contention:
  - Radio tx on a shared channel.
  - No collision detection.
  - Non-simultaneous rx and tx.
- Constant memory size.
- Limited life cycle.
- Short transmission range.
- Discrete tx power range.
- One channel of communication.
- No position information.
- Unreliability.

$$tx = transmission.$$
$$rx = reception.$$

Our Results

Sensor Network:

- n nodes
- range of transmission r
- diameter D
- max degree Δ
- nodes only know n.
- all nodes hold message of size m to disseminate.
- $O(nm)$ message and memory size.

Gossiping algorithm:

- $O(\Delta + D)$ w.h.p. relaxed-WSM-compatible
- $\Omega(D)$ and $\Omega(\Delta)$ are lower bounds $\Rightarrow$ optimal.

Observations:

- time improvement with no global synchronism (exploits geometry)
- classical broadcast lower bound of KM can be broken (by pre-processing)

Our Results

Sensor Network:

- n nodes
- range of transmission r
- diameter D
- max degree Δ
- nodes only know n.
- all nodes hold message of size m to disseminate.
- $O(nm)$ message and memory size.

Gossiping algorithm:

- $O(\Delta + D)$ w.h.p. relaxed-WSM-compatible
- $\Omega(D)$ and $\Omega(\Delta)$ are lower bounds $\Rightarrow$ optimal.

Observations:

- time improvement with no global synchronism (exploits geometry)
- classical broadcast lower bound of KM can be broken (by pre-processing)

Our Results

Sensor Network:

- n nodes
- range of transmission r
- diameter D
- max degree Δ
- nodes only know n.
- all nodes hold message of size m to disseminate.
- $O(nm)$ message and memory size.

Gossiping algorithm:

- $O(\Delta + D)$ w.h.p. relaxed-WSM-compatible
- $\Omega(D)$ and $\Omega(\Delta)$ are lower bounds $\Rightarrow$ optimal.

Observations:

- time improvement with no global synchronism (exploits geometry)
- classical broadcast lower bound of KM can be broken (by pre-processing)

Gossiping Algorithm

- Partition nodes in *masters* and *slaves*
  - every slave is at $d \leq ar$ from some master $(0 < a < 1/3)$
  - every pair of masters are at $d > ar$
    $\rightarrow$ MIS(ar)
- Every master reserves blocks of time steps for local use
  - master and slaves communicate without collisions within r
    $\rightarrow$ Coloring(r), using counter (achives local synch. and coll. detection)

Gossiping Algorithm

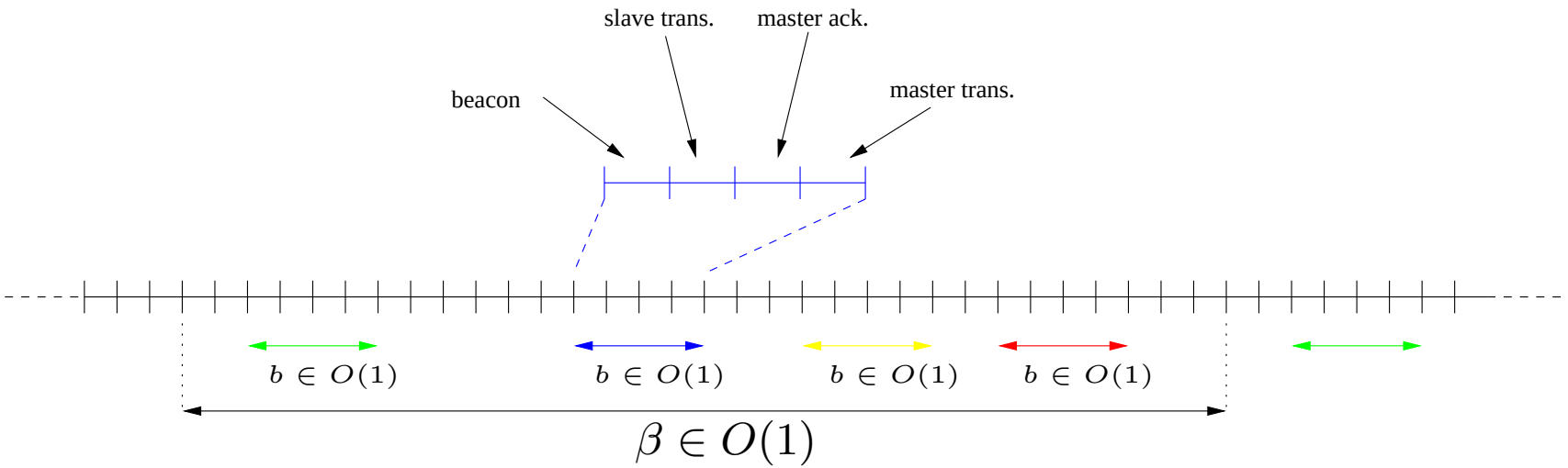
- Partition nodes in *masters* and *slaves*
  - every slave is at $d \leq ar$ from some master $(0 < a < 1/3)$
  - every pair of masters are at $d > ar$
    $\rightarrow$ MIS(ar)
- Every master reserves blocks of time steps for local use
  - master and slaves communicate without collisions within r
    $\rightarrow$ Coloring(r), using counter (achives local synch. and coll. detection)

Gossiping Algorithm

- Partition nodes in *masters* and *slaves*
  - every slave is at $d \leq ar$ from some master $(0 < a < 1/3)$
  - every pair of masters are at $d > ar$
    $\rightarrow$ MIS(ar)
- Every master reserves blocks of time steps for local use
  - master and slaves communicate without collisions within r
    $\rightarrow$ Coloring(r), using counter (achives local synch. and coll. detection)

Gossiping Algorithm

- Partition nodes in *masters* and *slaves*
  - every slave is at $d \leq ar$ from some master $(0 < a < 1/3)$
  - every pair of masters are at $d > ar$
    $\rightarrow$ MIS(ar)
- Every master reserves blocks of time steps for local use
  - master and slaves communicate without collisions within r
    $\rightarrow$ Coloring(r), using counter (achives local synch. and coll. detection)



Gossiping Algorithm

- Partition nodes in *masters* and *slaves*
  - every slave is at $d \leq ar$ from some master $(0 < a < 1/3)$
  - every pair of masters are at $d > ar$
    $\rightarrow$ MIS(ar)
- Every master reserves blocks of time steps for local use
  - master and slaves communicate without collisions within r
    $\rightarrow$ Coloring(r), using counter (achives local synch. and coll. detection)
- Every master maintains set of messages received
  - initially set contains own message only
  - slaves pass message to master (using reserved blocks and radius ar)
  - master adds messages to set
    $\rightarrow$ window back-on/back-off + $O(\log^2 n)$ times $p_{trans} = 1/\log n$
- Every master disseminates local set (using reserved blocks)
  - masters deterministically pass set to neighboring masters (radius r)
  - masters add messages received from other masters to local set
    $\rightarrow$ flooding among masters

Gossiping Algorithm

- Partition nodes in *masters* and *slaves*
  - every slave is at $d \leq ar$ from some master $(0 < a < 1/3)$
  - every pair of masters are at $d > ar$
    $\rightarrow$ MIS(ar)
- Every master reserves blocks of time steps for local use
  - master and slaves communicate without collisions within r
    $\rightarrow$ Coloring(r), using counter (achives local synch. and coll. detection)
- Every master maintains set of messages received
  - initially set contains own message only
  - slaves pass message to master (using reserved blocks and radius ar)
  - master adds messages to set
    $\rightarrow$ window back-on/back-off + $O(\log^2 n)$ times $p_{trans} = 1/\log n$
- Every master disseminates local set (using reserved blocks)
  - masters deterministically pass set to neighboring masters (radius r)
  - masters add messages received from other masters to local set
    $\rightarrow$ flooding among masters

Gossiping Algorithm

- Partition nodes in *masters* and *slaves*
  - every slave is at $d \leq ar$ from some master $(0 < a < 1/3)$
  - every pair of masters are at $d > ar$
    $\rightarrow$ MIS(ar)
- Every master reserves blocks of time steps for local use
  - master and slaves communicate without collisions within r
    $\rightarrow$ Coloring(r), using counter (achives local synch. and coll. detection)
- Every master maintains set of messages received
  - initially set contains own message only
  - slaves pass message to master (using reserved blocks and radius ar)
  - master adds messages to set
    $\rightarrow$ window back-on/back-off + $O(\log^2 n)$ times $p_{trans} = 1/\log n$
- Every master disseminates local set (using reserved blocks)
  - masters deterministically pass set to neighboring masters (radius r)
  - masters add messages received from other masters to local set
    $\rightarrow$ flooding among masters

Gossiping Algorithm

- Partition nodes in *masters* and *slaves*
  - every slave is at $d \leq ar$ from some master $(0 < a < 1/3)$
  - every pair of masters are at $d > ar$
    $\rightarrow$ MIS(ar)
- Every master reserves blocks of time steps for local use
  - master and slaves communicate without collisions within r
    $\rightarrow$ Coloring(r), using counter (achives local synch. and coll. detection)
- Every master maintains set of messages received
  - initially set contains own message only
  - slaves pass message to master (using reserved blocks and radius ar)
  - master adds messages to set
    $\rightarrow$ window back-on/back-off + $O(\log^2 n)$ times $p_{trans} = 1/\log n$
- Every master disseminates local set (using reserved blocks)
  - masters deterministically pass set to neighboring masters (radius r)
  - masters add messages received from other masters to local set
    $\rightarrow$ flooding among masters

Gossiping Algorithm
Time efficiency

Assume phase synchronism

1. Partition nodes in *masters* and *slaves*
   $\Rightarrow$ MIS $\rightarrow O(\log^2 n)$
2. Every master reserves blocks of time steps for local use
   $\Rightarrow$ Coloring $\rightarrow O(\log n)$
3. Every master maintains set of messages received
   $\Rightarrow$ window back-on/back-off $\rightarrow O(\Delta + \log^2 n \log \Delta)$
4. Every master disseminates local set
   $\Rightarrow$ flooding among masters $\rightarrow O(D)$

Overall:

$$O(\log^2 n + \log n + \Delta + \log^2 n \log \Delta + D) \in O(\Delta + D)$$

even without synch.

Gossiping Algorithm
Time efficiency

Assume phase synchronism

1. Partition nodes in *masters* and *slaves*
   $\Rightarrow$ MIS $\rightarrow O(\log^2 n)$
2. Every master reserves blocks of time steps for local use
   $\Rightarrow$ Coloring $\rightarrow O(\log n)$
3. Every master maintains set of messages received
   $\Rightarrow$ window back-on/back-off $\rightarrow O(\Delta + \log^2 n \log \Delta)$
4. Every master disseminates local set
   $\Rightarrow$ flooding among masters $\rightarrow O(D)$

Overall:

$$O(\log^2 n + \log n + \Delta + \log^2 n \log \Delta + D) \in O(\Delta + D)$$

even without synch.

Gossiping Algorithm
Time efficiency

Assume phase synchronism

1. Partition nodes in *masters* and *slaves*
   $\Rightarrow$ MIS $\rightarrow O(\log^2 n)$
2. Every master reserves blocks of time steps for local use
   $\Rightarrow$ Coloring $\rightarrow O(\log n)$
3. Every master maintains set of messages received
   $\Rightarrow$ window back-on/back-off $\rightarrow O(\Delta + \log^2 n \log \Delta)$
4. Every master disseminates local set
   $\Rightarrow$ flooding among masters $\rightarrow O(D)$

Overall:

$$O(\log^2 n + \log n + \Delta + \log^2 n \log \Delta + D) \in O(\Delta + D)$$

even without synch.

Gossiping Algorithm
Time efficiency

Assume phase synchronism

1. Partition nodes in *masters* and *slaves*
   $\Rightarrow$ MIS $\rightarrow O(\log^2 n)$
2. Every master reserves blocks of time steps for local use
   $\Rightarrow$ Coloring $\rightarrow O(\log n)$
3. Every master maintains set of messages received
   $\Rightarrow$ window back-on/back-off $\rightarrow O(\Delta + \log^2 n \log \Delta)$
4. Every master disseminates local set
   $\Rightarrow$ flooding among masters $\rightarrow O(D)$

Overall:

$$O(\log^2 n + \log n + \Delta + \log^2 n \log \Delta + D) \in O(\Delta + D)$$

even without synch.

Gossiping Algorithm
Time efficiency

Assume phase synchronism

1. Partition nodes in *masters* and *slaves*
   $\Rightarrow$ MIS $\rightarrow O(\log^2 n)$
2. Every master reserves blocks of time steps for local use
   $\Rightarrow$ Coloring $\rightarrow O(\log n)$
3. Every master maintains set of messages received
   $\Rightarrow$ window back-on/back-off $\rightarrow O(\Delta + \log^2 n \log \Delta)$
4. Every master disseminates local set
   $\Rightarrow$ flooding among masters $\rightarrow O(D)$

Overall:

$$O(\log^2 n + \log n + \Delta + \log^2 n \log \Delta + D) \in O(\Delta + D)$$

even without synch.

Gossiping Algorithm
Time efficiency

Assume phase synchronism

1. Partition nodes in *masters* and *slaves*
   $\Rightarrow$ MIS $\rightarrow O(\log^2 n)$
2. Every master reserves blocks of time steps for local use
   $\Rightarrow$ Coloring $\rightarrow O(\log n)$
3. Every master maintains set of messages received
   $\Rightarrow$ window back-on/back-off $\rightarrow O(\Delta + \log^2 n \log \Delta)$
4. Every master disseminates local set
   $\Rightarrow$ flooding among masters $\rightarrow O(D)$

Overall:

$$O(\log^2 n + \log n + \Delta + \log^2 n \log \Delta + D) \in O(\Delta + D)$$

even without synch.

Thank you