Deterministic Recurrent Communication and Synchronization in Restricted Sensor Networks

A. Fernández Anta M. A. Mosteiro Christopher Thraves

ASAP Research team
IRISA/INRIA Rennes

ALGOSENSORS 2010

Introduction

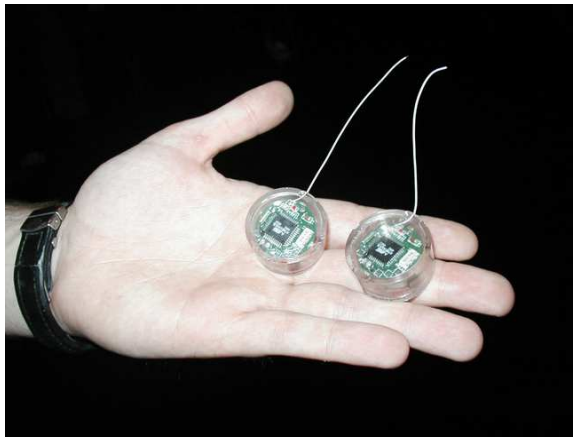
A sensor node

Capabilities

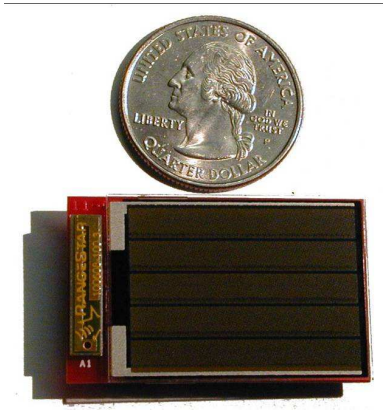
- processing
- sensing
- communication

Limitations

- range
- memory
- life cycle



University of California, Berkeley and Intel Berkeley Research Lab.



PicoBeacon

Berkeley Wireless Research Center

A sensor node

Capabilities

- processing
- sensing
- communication

Limitations

- range
- memory
- life cycle

A sensor node

Capabilities

- processing
- sensing
- communication

Limitations

- range
- memory
- life cycle

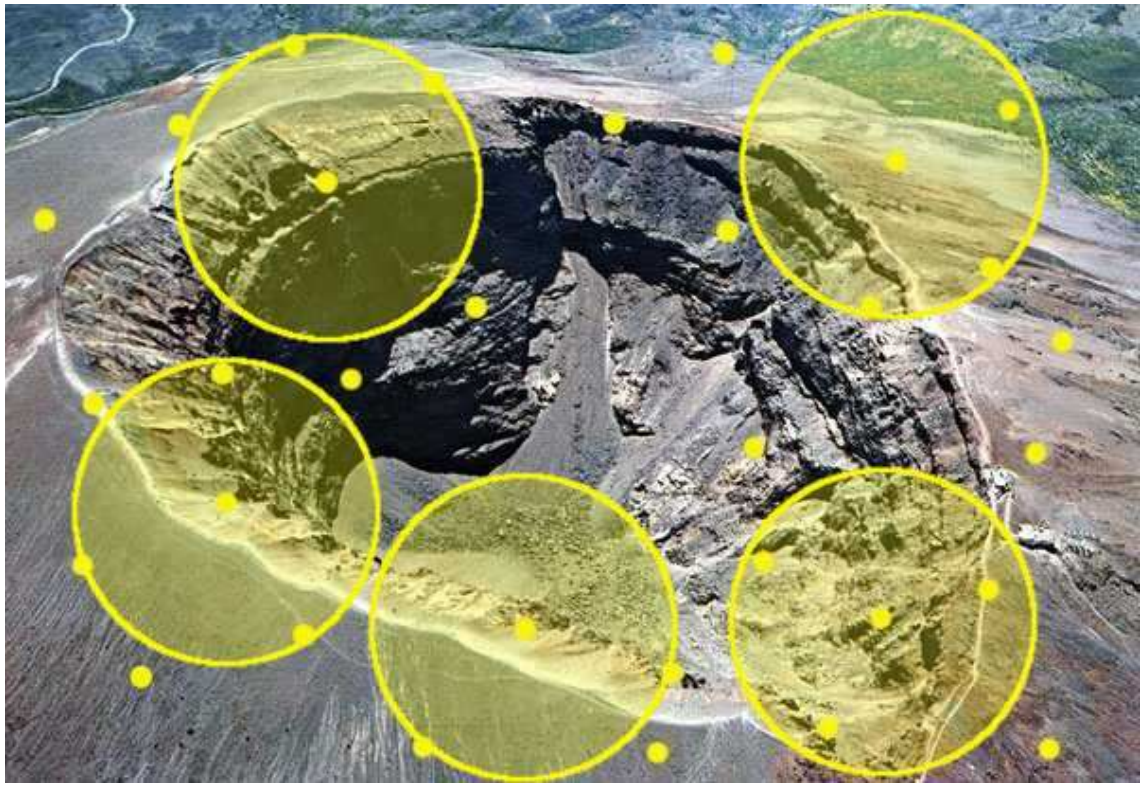
A sensor node

Capabilities

- processing
- sensing
- communication

Limitations

- range
- memory
- life cycle



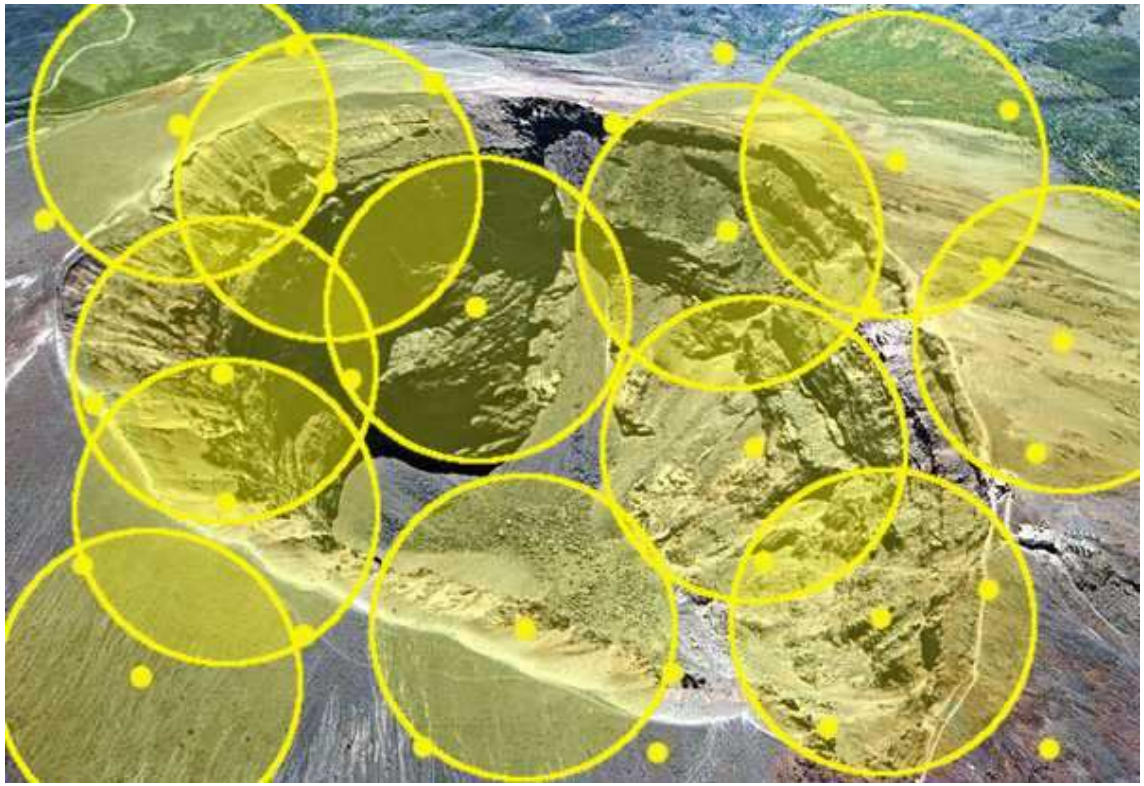
A sensor node

Capabilities

- processing
- sensing
- communication

Limitations

- range
- memory
- life cycle



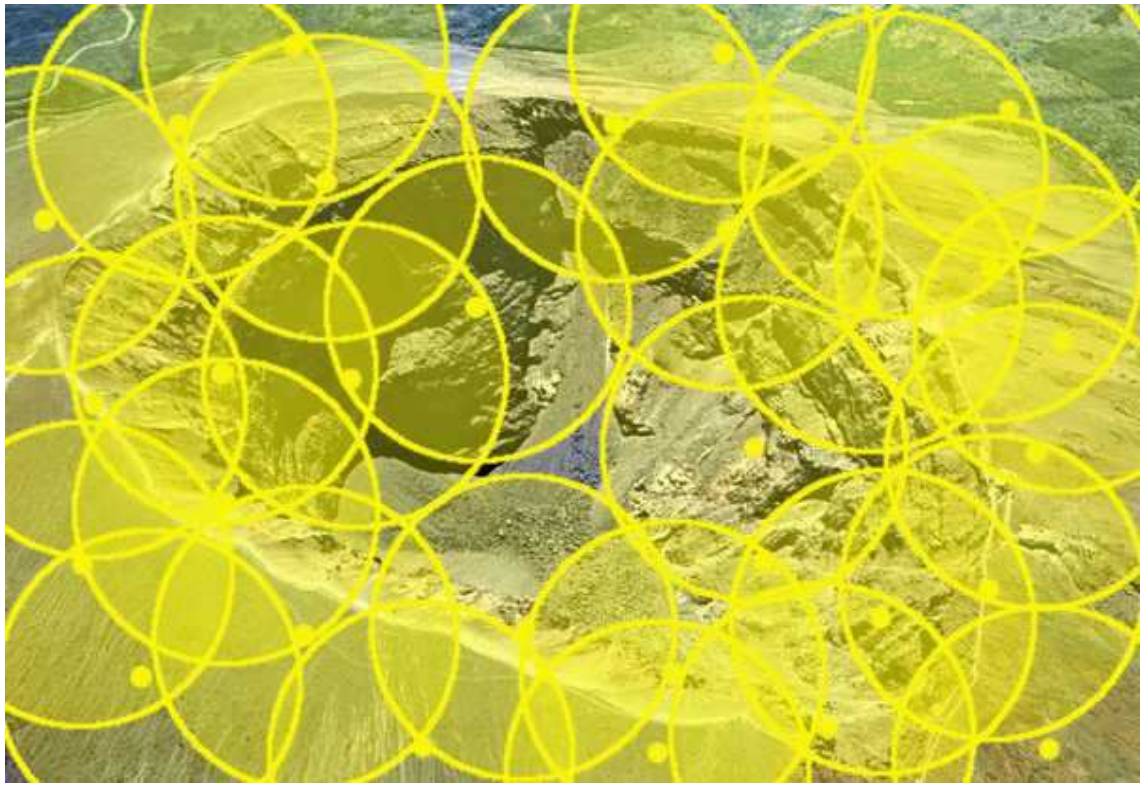
A sensor node

Capabilities

- processing
- sensing
- communication

Limitations

- range
- memory
- life cycle



Model and Problem Definition

Network and Nodes

- Deployed at random in the area of interest.
- Unique identification number (ID) in $\{0, 1, 2, \ldots, n-1\}$.
- Limited communication range (transmission = reception)

  $\Rightarrow$ nodes can duplicate their communication range.
- n, k and D are known by all the nodes in the system.

Network and Nodes

- Deployed at random in the area of interest.
- Unique identification number (ID) in $\{0, 1, 2, \ldots, n-1\}$.
- Limited communication range (transmission = reception)

  $\Rightarrow$ nodes can duplicate their communication range.
- n, k and D are known by all the nodes in the system.

Network and Nodes

- Deployed at random in the area of interest.
- Unique identification number (ID) in $\{0, 1, 2, \ldots, n-1\}$.
- Limited communication range (transmission = reception)

  $\Rightarrow$ nodes can duplicate their communication range.

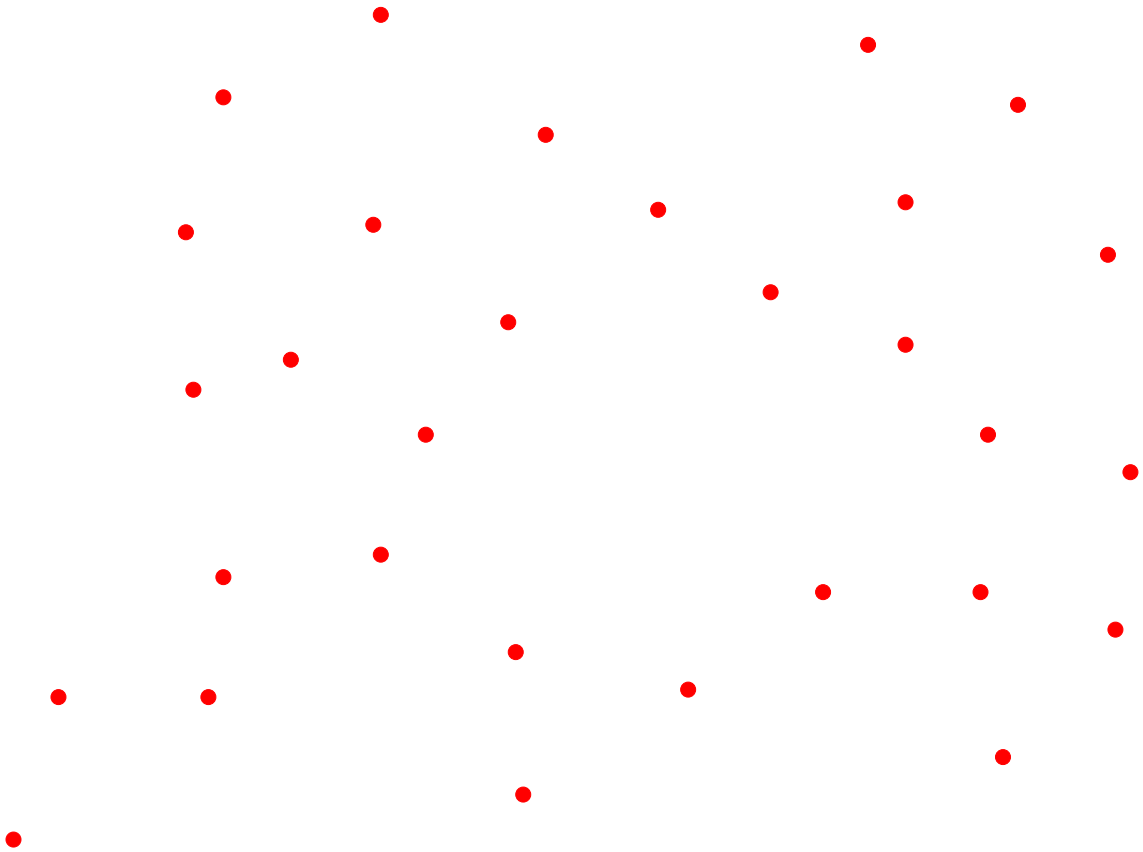
- n, k and D are known by all the nodes in the system.

Network and Nodes

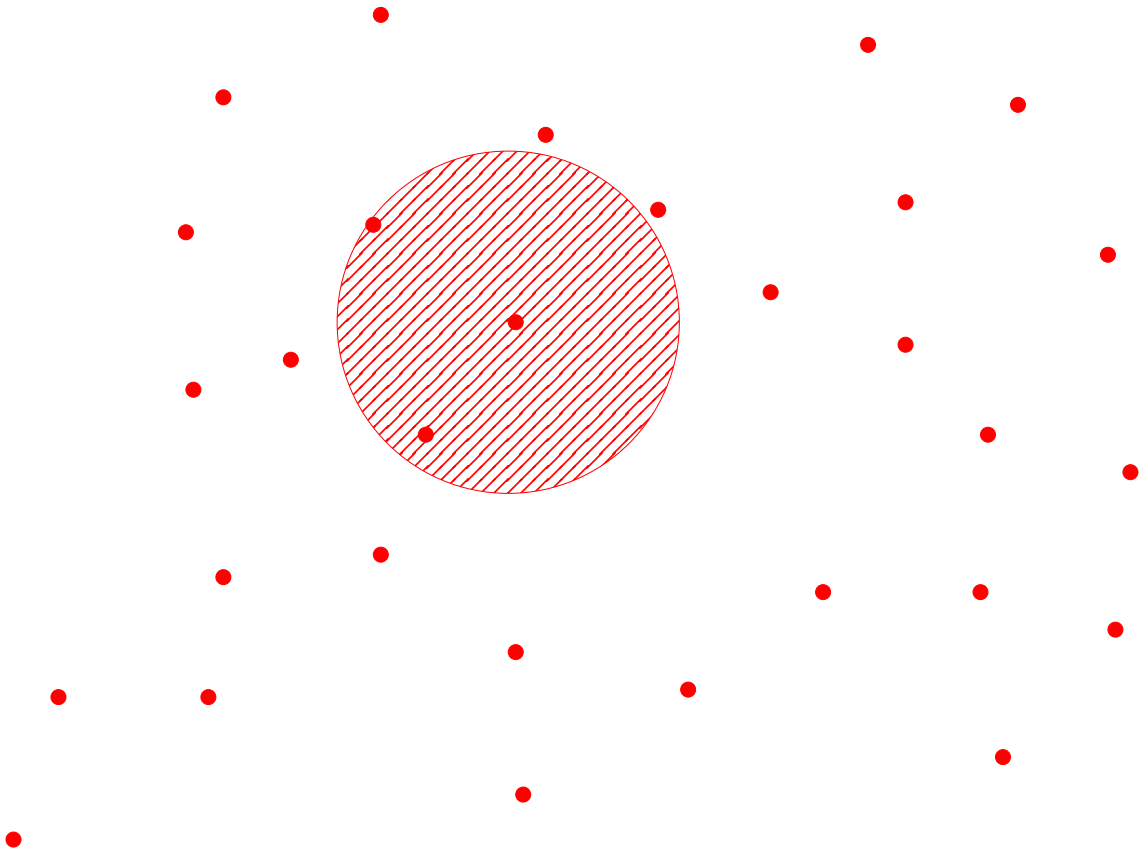
- Deployed at random in the area of interest.
- Unique identification number (ID) in $\{0, 1, 2, \ldots, n-1\}$.
- Limited communication range (transmission = reception)

  $\Rightarrow$ nodes can duplicate their communication range.
- n, k and D are known by all the nodes in the system.

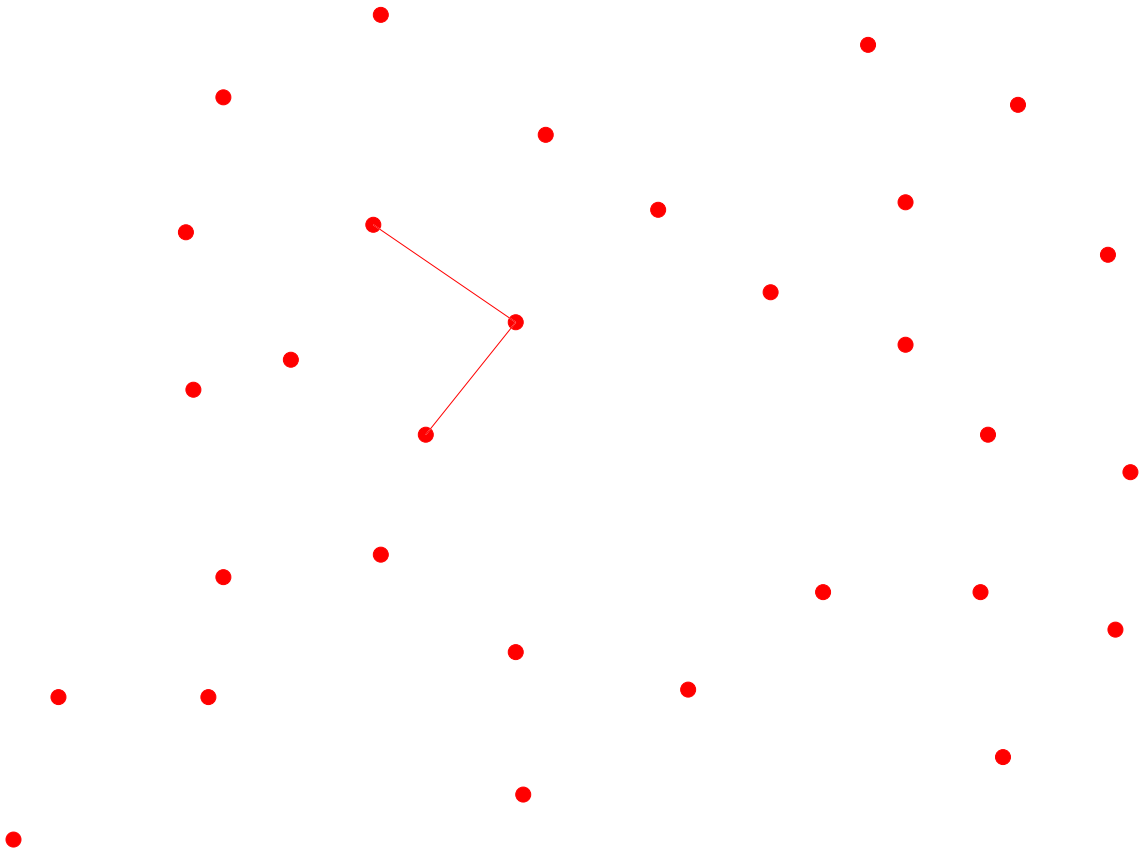
Network model: *Geometric Graph*



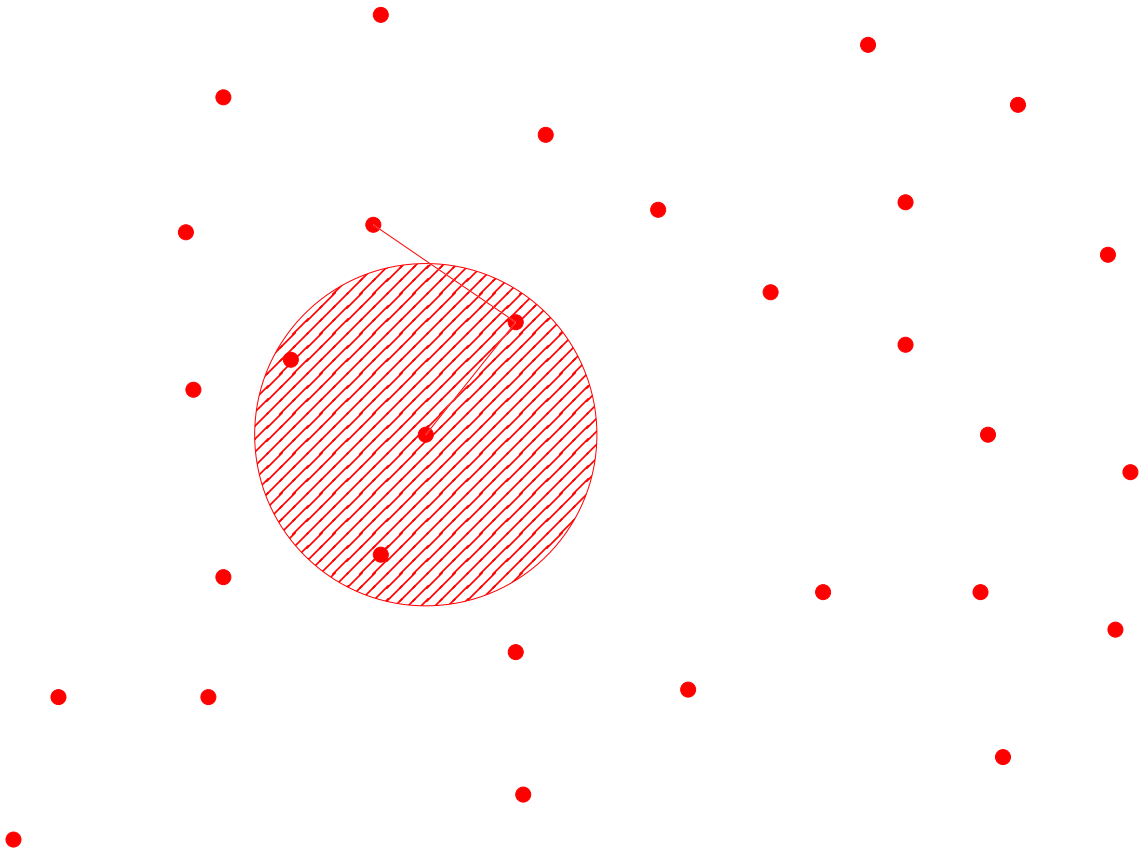
Network model: *Geometric Graph*



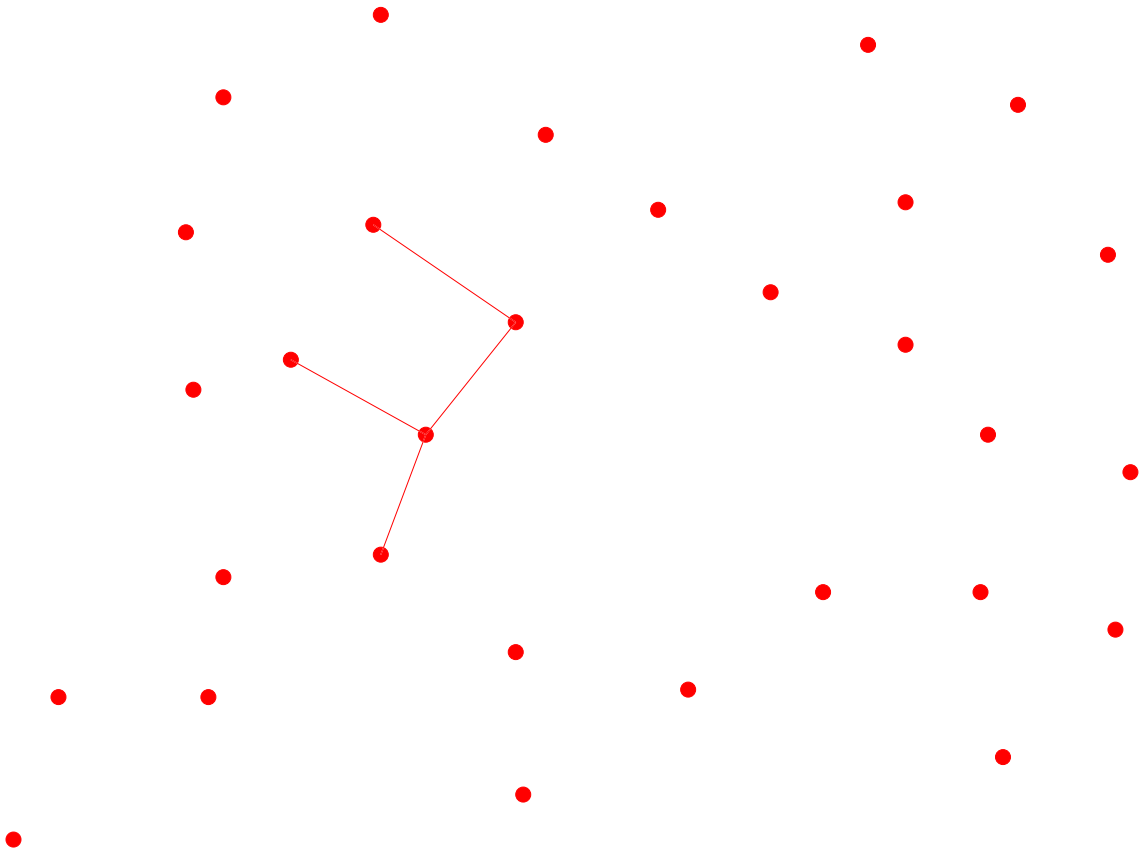
Network model: *Geometric Graph*



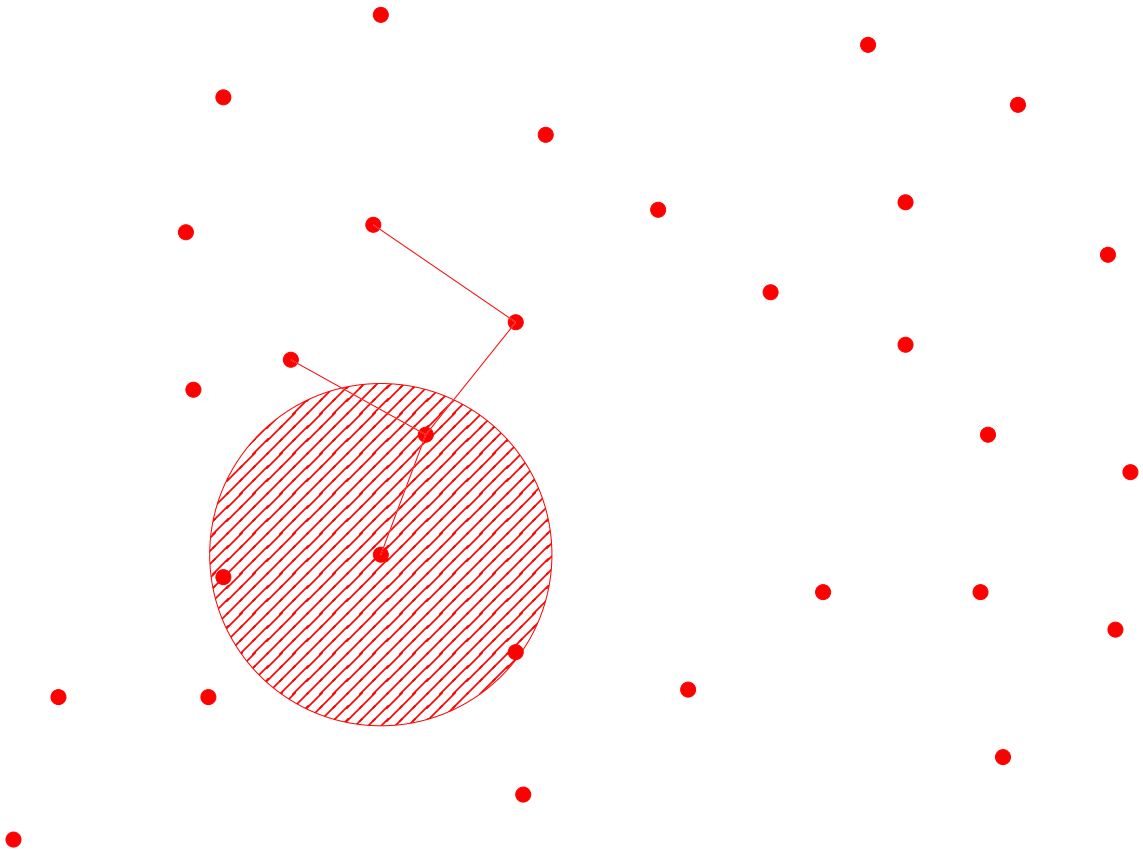
Network model: *Geometric Graph*



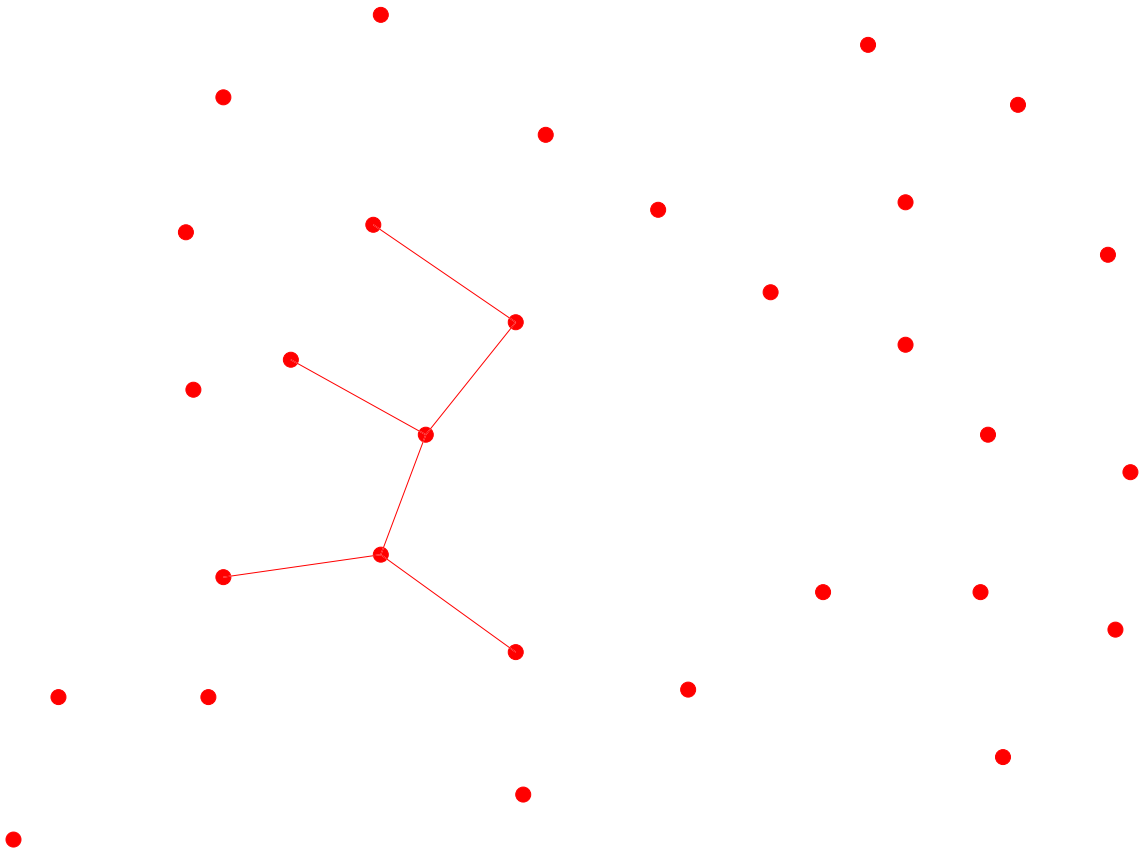
Network model: *Geometric Graph*



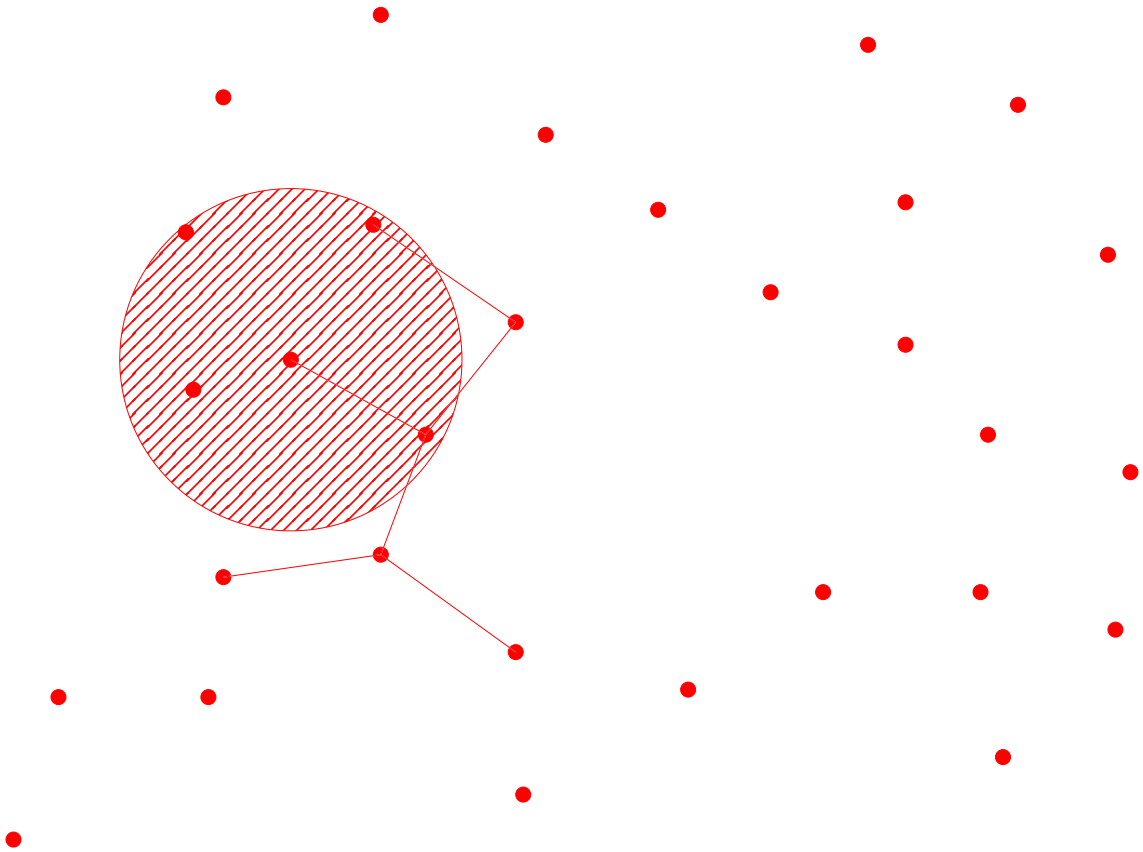
Network model: *Geometric Graph*



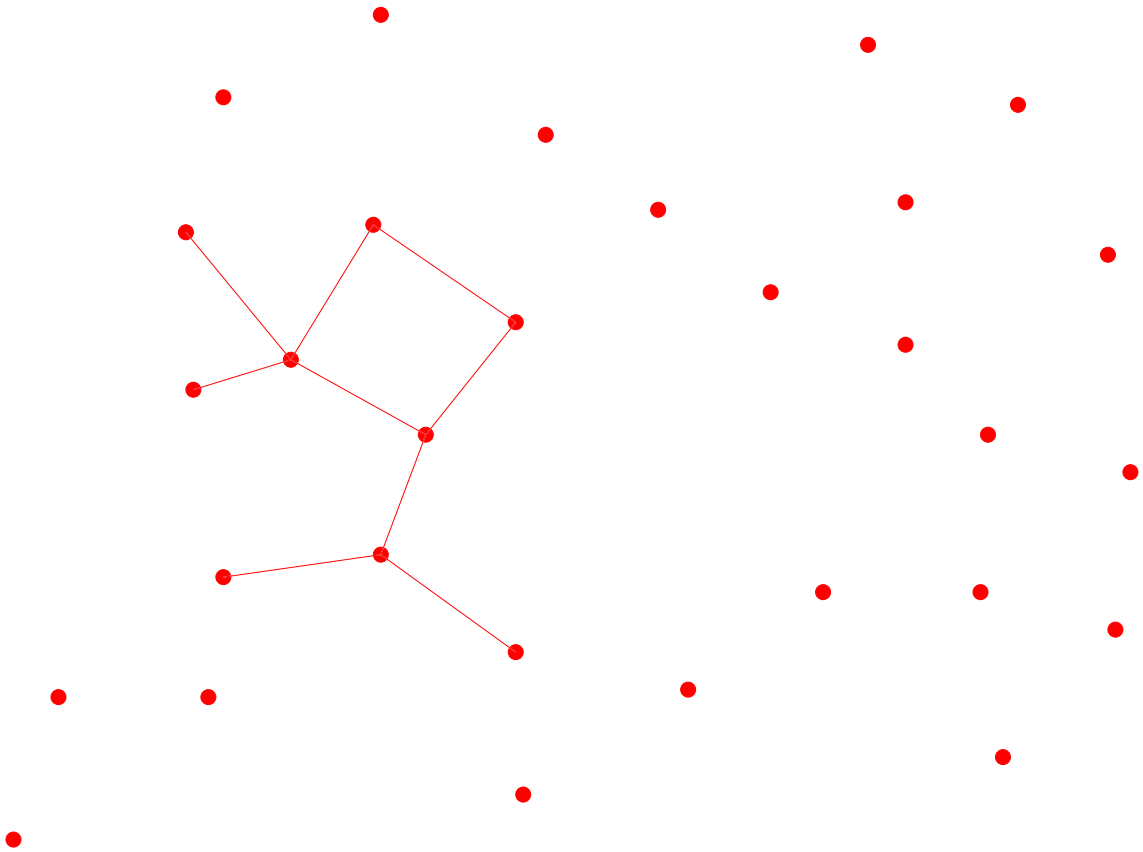
Network model: *Geometric Graph*



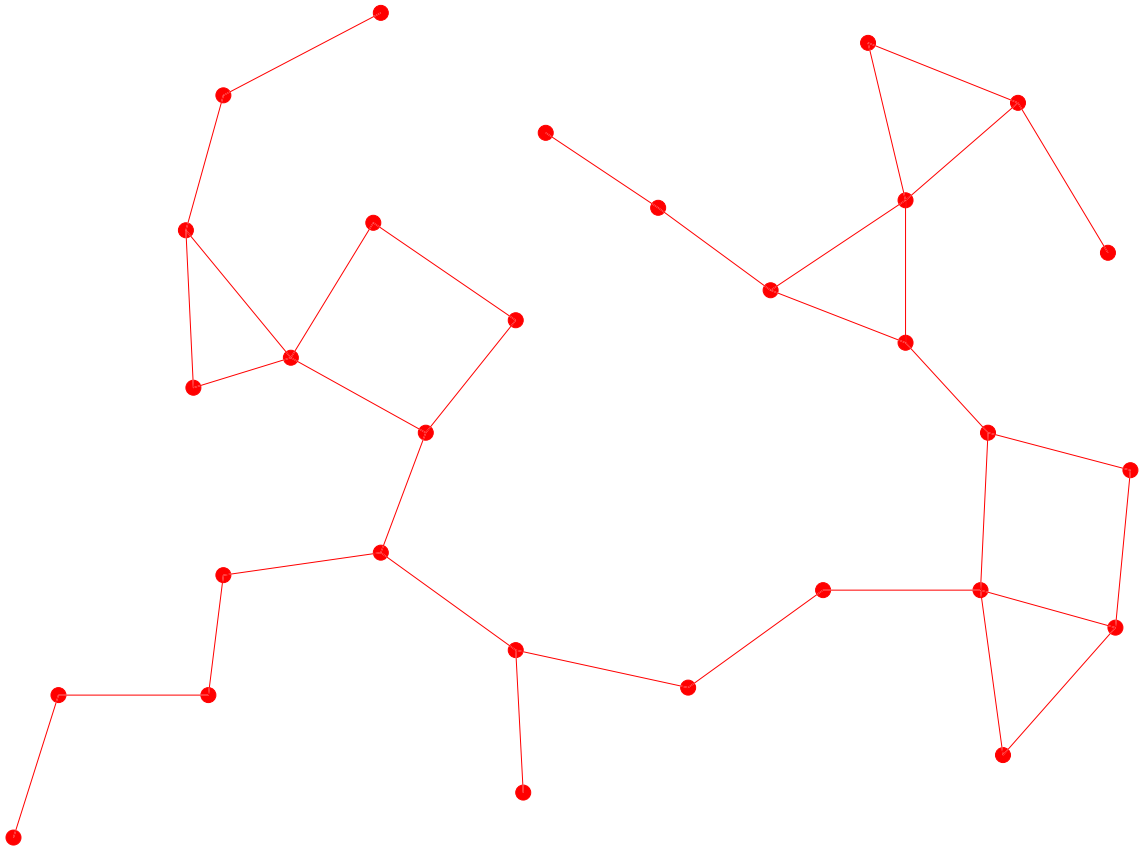
Network model: *Geometric Graph*



Network model: *Geometric Graph*



Network model: *Geometric Graph*



Local Synchrony

- Time is assumed to be slotted (steps).
- Each transmission occurs in a given slot.
- The slots of all nodes are in phase.
- Availability of a hardware clock mechanism: LOCAL-CLOCK.

Local Synchrony

- Time is assumed to be slotted (steps).
- Each transmission occurs in a given slot.
- The slots of all nodes are in phase.
- Availability of a hardware clock mechanism: LOCAL-CLOCK.

Local Synchrony

- Time is assumed to be slotted (steps).
- Each transmission occurs in a given slot.
- The slots of all nodes are in phase.
- Availability of a hardware clock mechanism: LOCAL-CLOCK.

Local Synchrony

- Time is assumed to be slotted (steps).
- Each transmission occurs in a given slot.
- The slots of all nodes are in phase.
- Availability of a hardware clock mechanism: LOCAL-CLOCK.

Node Reliability

Nodes may fail, BUT:

$\Rightarrow$ the network stays connected (one connected component) at all times

$\Rightarrow$ the first node awakened is always awake

$\Rightarrow$ each period when a node runs without failures lasts at least the length of the stabilization time.

Node Reliability

Nodes may fail, BUT:

$\Rightarrow$ the network stays connected (one connected component) at all times

$\Rightarrow$ the first node awakened is always awake

$\Rightarrow$ each period when a node runs without failures lasts at least the length of the stabilization time.

Node Reliability

Nodes may fail, BUT:

$\Rightarrow$ the network stays connected (one connected component) at all times

$\Rightarrow$ the first node awakened is always awake

$\Rightarrow$ each period when a node runs without failures lasts at least the length of the stabilization time.

Node Awakening

Definition

A τ-*adversary* is an adversary that awakens all the nodes of the network within a window time of size τ, i.e., no node is awakened at a time $t \geq \tau$. Additionally, a τ-adversary does not recover crashed nodes. The parameter τ is assumed known by the nodes.

Definition

An ∞-*adversary* is an adversary that has no restriction on when nodes are awakened.

DRC Problem

Definition

A distributed protocol solves the *deterministic recurrent communication* (DRC) problem if it guarantees that, for every step t and every pair $(u, v) \in E$, there is some step $t' \geq t$ such that, in step t', v receives an application message from u.

Why Deterministic Communication?

- Only one channel of communication

  $\Rightarrow$ must deal with collision of transmissions!

  Popular solution $\rightarrow$ random protocols.

- BUT scarcest resource is energy and

  random protocols $\Rightarrow$ redundant transmissions!.

  $\Rightarrow$ deterministic protocols may help.

Why Deterministic Communication?

- Only one channel of communication

  $\Rightarrow$ must deal with collision of transmissions!

  Popular solution $\rightarrow$ random protocols.

- BUT scarcest resource is energy and

  random protocols $\Rightarrow$ redundant transmissions!.

  $\Rightarrow$ deterministic protocols may help.

Why Deterministic Communication?

- Only one channel of communication

  $\Rightarrow$ must deal with collision of transmissions!

  Popular solution $\rightarrow$ random protocols.

- BUT scarcest resource is energy and

  random protocols $\Rightarrow$ redundant transmissions!.

  $\Rightarrow$ deterministic protocols may help.

Why Deterministic Communication?

- Only one channel of communication

  $\Rightarrow$ must deal with collision of transmissions!

  Popular solution $\rightarrow$ random protocols.

- BUT scarcest resource is energy and

  random protocols $\Rightarrow$ redundant transmissions!.

  $\Rightarrow$ deterministic protocols may help.

Motivation and Evaluation

- Sensor Networks application: monitor physical phenomena.

  $\Rightarrow$ protocols must guarantee communication infinitely many times.

- Optimization criteria:

  1) low energy cost.

  2) short delay between transmissions.

Motivation and Evaluation

- Sensor Networks application: monitor physical phenomena.

  $\Rightarrow$ protocols must guarantee communication infinitely many times.

- Optimization criteria:

  1) low energy cost.

  2) short delay between transmissions.

Motivation and Evaluation

- Sensor Networks application: monitor physical phenomena.

  $\Rightarrow$ protocols must guarantee communication infinitely many times.

- Optimization criteria:

  1) low energy cost.

  2) short delay between transmissions.

Motivation and Evaluation

- Sensor Networks application: monitor physical phenomena.

  $\Rightarrow$ protocols must guarantee communication infinitely many times.

- Optimization criteria:

  1) low energy cost.

  2) short delay between transmissions.

Related Work

- *Message passing*:

  [ABLP'92] Each node receives from all neighbors in $O(k^2 \log^2 n / \log(k \log n))$. $\rightarrow$ synchronous start. $\omega(1)$-degree bipartite-graphs requiring $\Omega(k \log k)$. $\rightarrow$ not embeddable in GG.

- *Broadcast & gossiping*:

  [CGR'00, CGOR'00, CR'03, CGGPR'02] $\rightarrow$ synchronous start, global clock, etc.

- *Selection*

  [Kowalski'05] Static, $\exists O(k \log(n/k))$, +[I'02]: $O(k \text{ polylog } n)$. $\rightarrow$ synchronous start. Dynamic $O(k^2 \log n)$. $\rightarrow$ nodes turn off upon succ. transmission.

- *Selective families*:

  [I'02] $\exists(k, n)$-selective families of size $O(k \text{ polylog } n)$.
  [DR'83] (m, k, n)-selectors must be $\Omega(\min\{n, k^2 \log_k n\})$ when $m = k$.
  [DBGV'03] (k, k, n)-selectors must be $\geq (k-1)^2 \log n / (4 \log(k-1) + O(1))$ and $\exists(k, k, n)$-selectors of size $O(k^2 \ln(n/k))$.
  All $\rightarrow$ synchronous start.

Our Solution

The Algorithm

We design our protocols assuming the existence of an *oblivious deterministic recurrent communication* (ORC) protocol that solves DRC with bounded delay and no start-up phase.

- Synchronization phase.
- Coloring phase.
- Aplication phase.

The Algorithm

We design our protocols assuming the existence of an *oblivious deterministic recurrent communication* (ORC) protocol that solves DRC with bounded delay and no start-up phase.

- Synchronization phase.
- Coloring phase.
- Aplication phase.

The Algorithm

We design our protocols assuming the existence of an *oblivious deterministic recurrent communication* (ORC) protocol that solves DRC with bounded delay and no start-up phase.

- Synchronization phase.
- Coloring phase.
- Aplication phase.

The Algorithm

We design our protocols assuming the existence of an *oblivious deterministic recurrent communication* (ORC) protocol that solves DRC with bounded delay and no start-up phase.

- Synchronization phase.
- Coloring phase.
- Aplication phase.

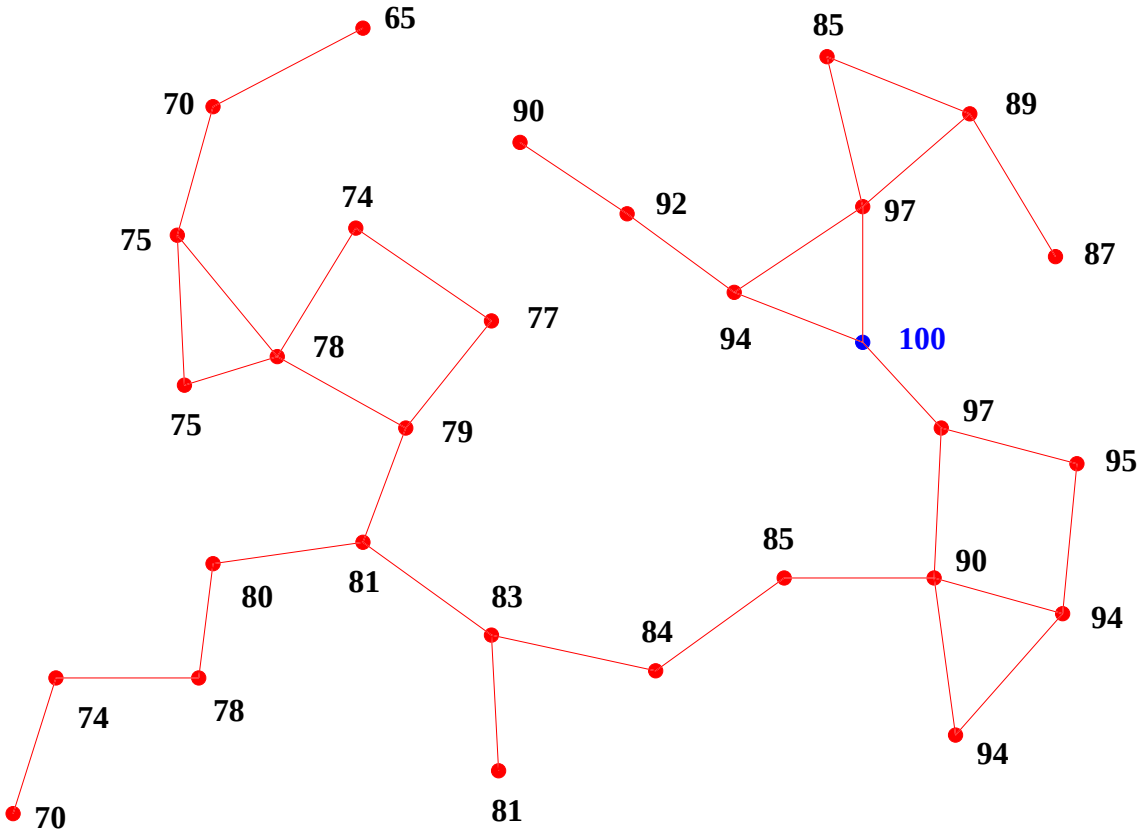
The Synchronization Problem

Definition

We say that a protocol solves the *synchronization problem* if there exists a time t from which the protocol guarantees that the network is synchronized at all times after t, and every node that awakes eventually gets synchronized. The maximum time between a node awaking and getting synchronized is the *synchronization time* of the protocol.

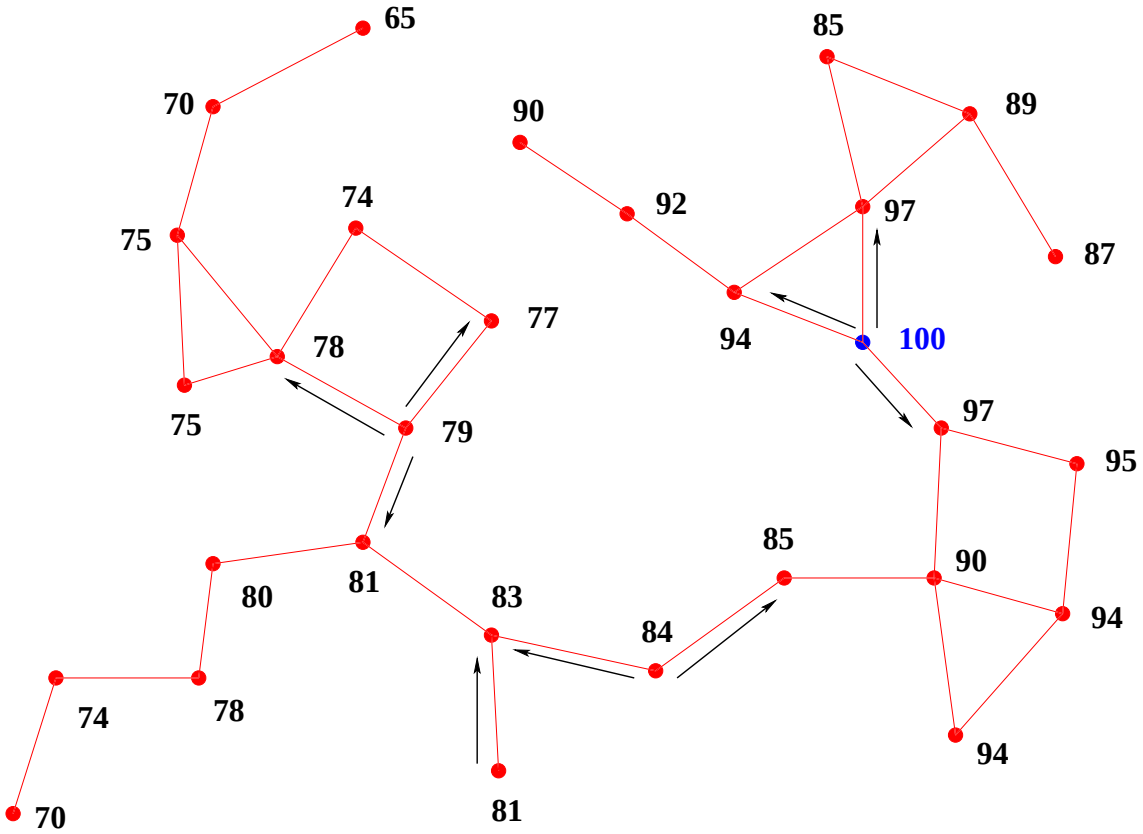
Synchronization Phase (r Network)

M. G. Gouda and T. Herman, *Stabilizing Unison.* IPL, 1990.



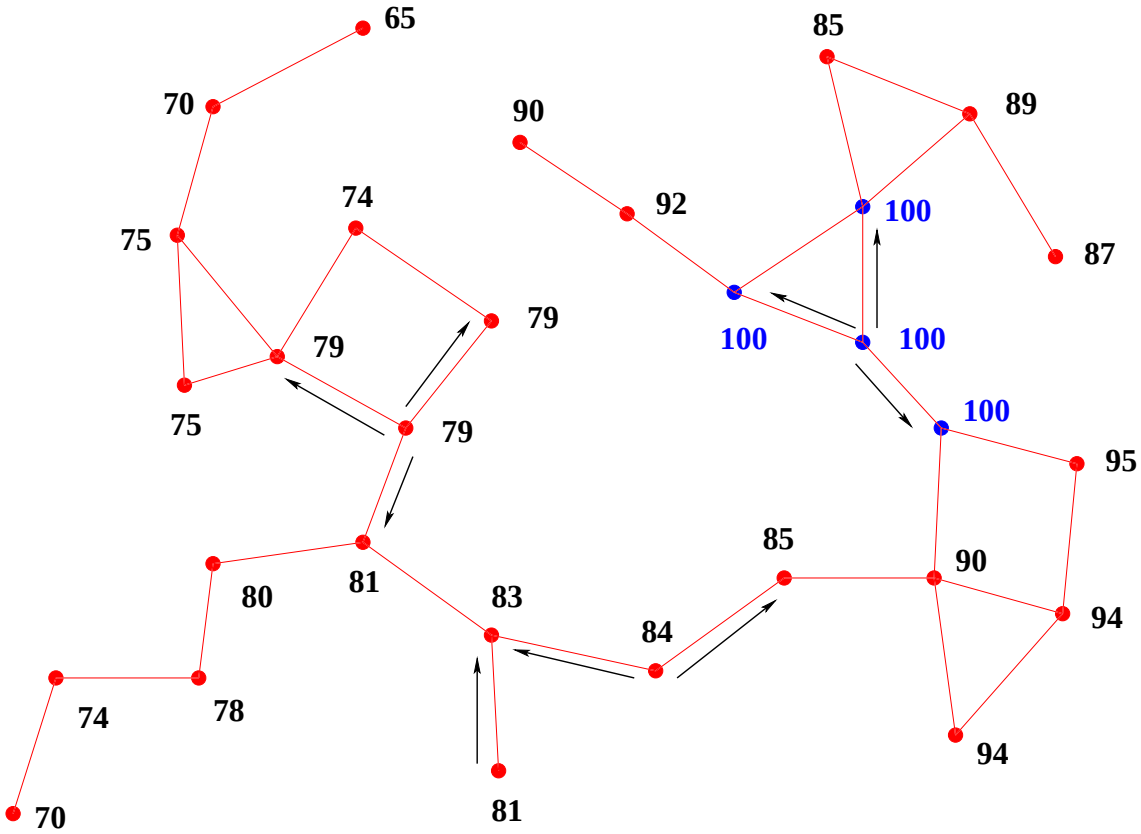
Synchronization Phase (r Network)

M. G. Gouda and T. Herman, *Stabilizing Unison.* IPL, 1990.



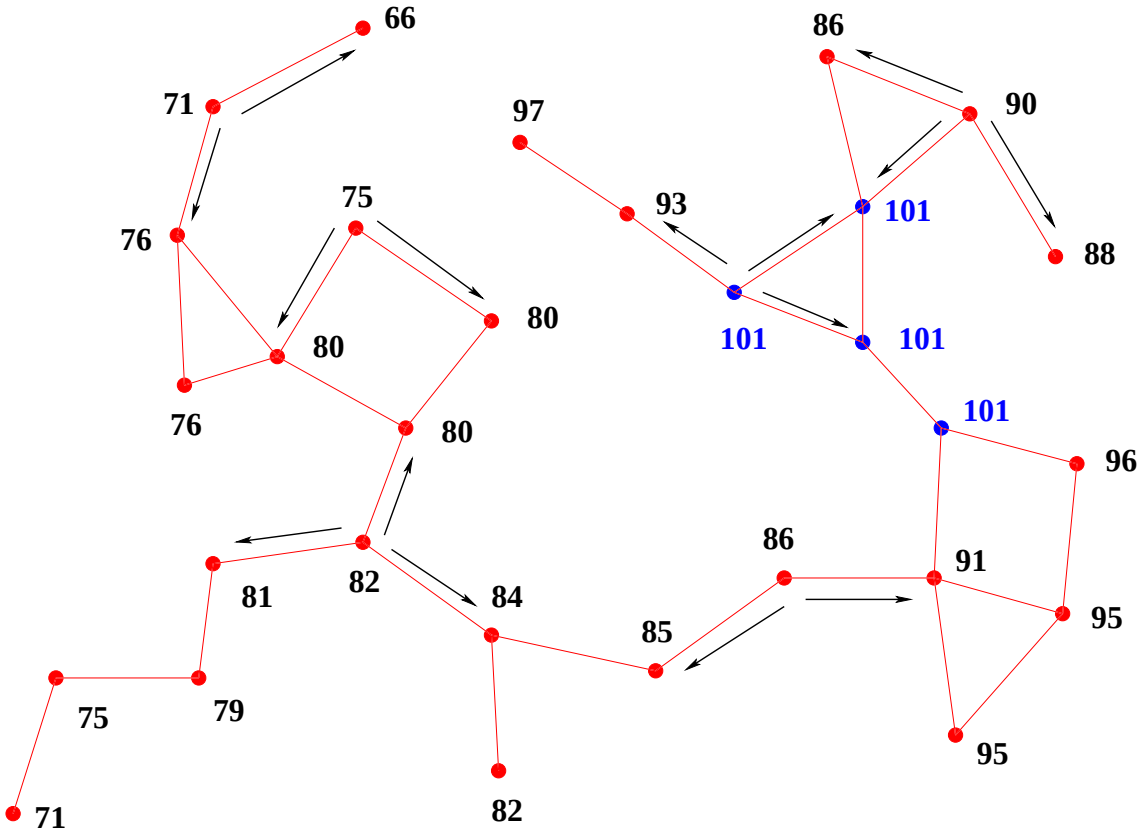
Synchronization Phase (r Network)

M. G. Gouda and T. Herman, *Stabilizing Unison.* IPL, 1990.



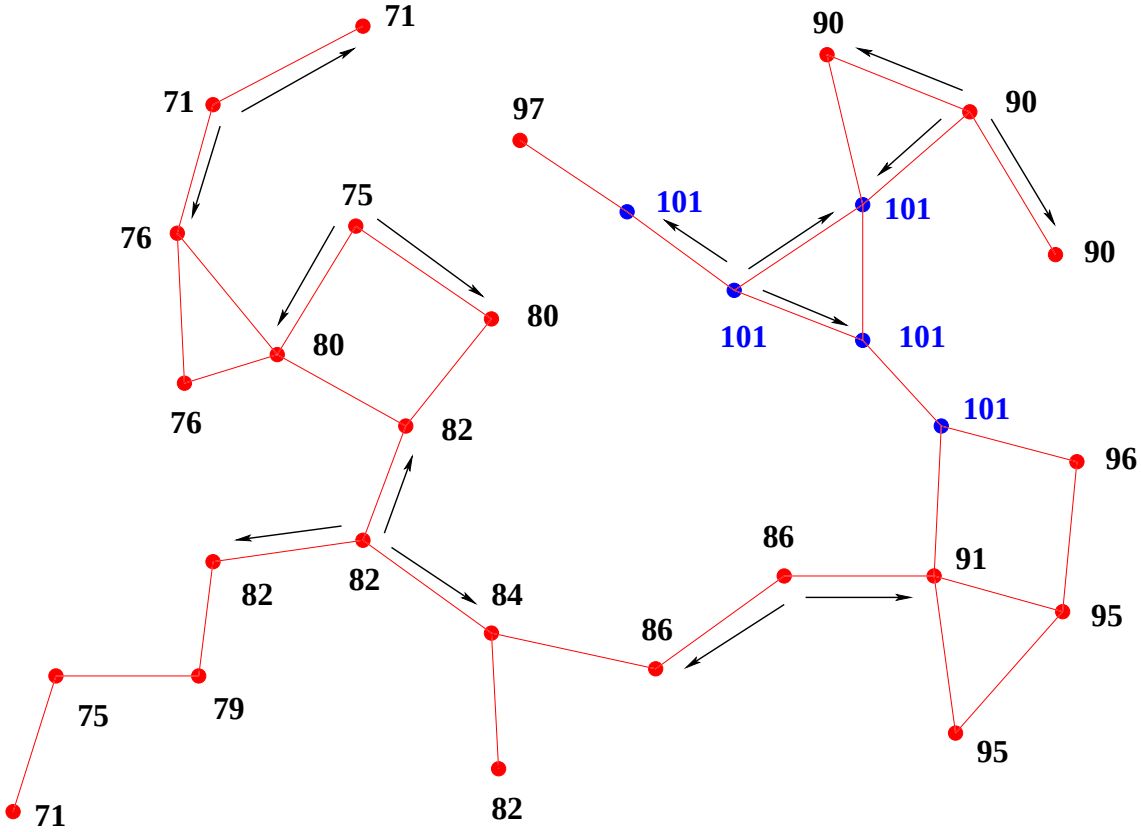
Synchronization Phase (r Network)

M. G. Gouda and T. Herman, *Stabilizing Unison.* IPL, 1990.



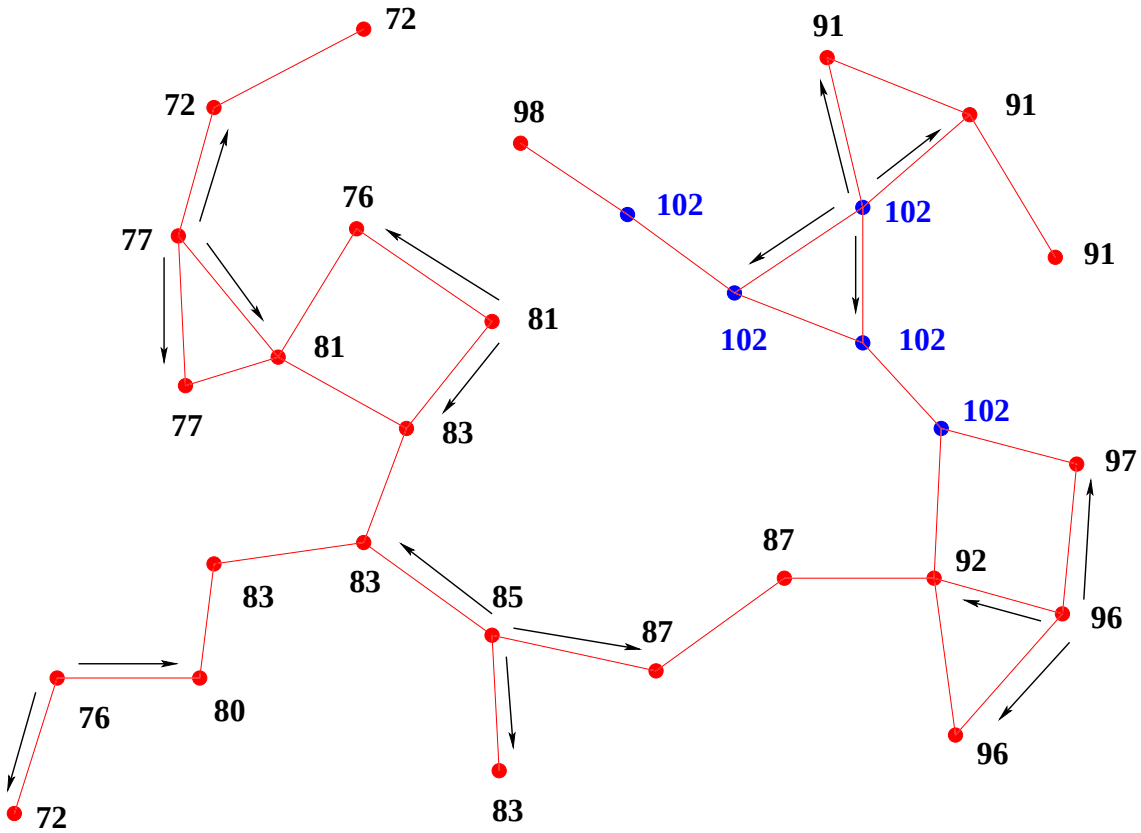
Synchronization Phase (r Network)

M. G. Gouda and T. Herman, *Stabilizing Unison.* IPL, 1990.



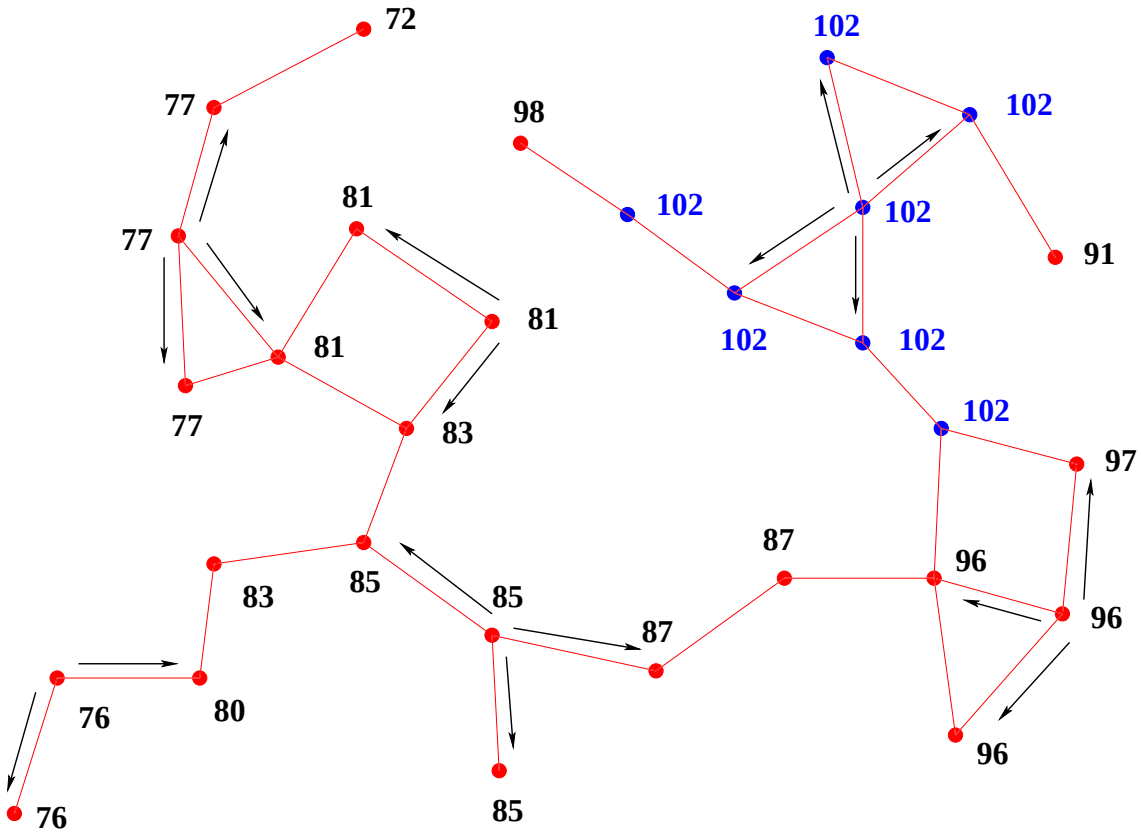
Synchronization Phase (r Network)

M. G. Gouda and T. Herman, *Stabilizing Unison.* IPL, 1990.



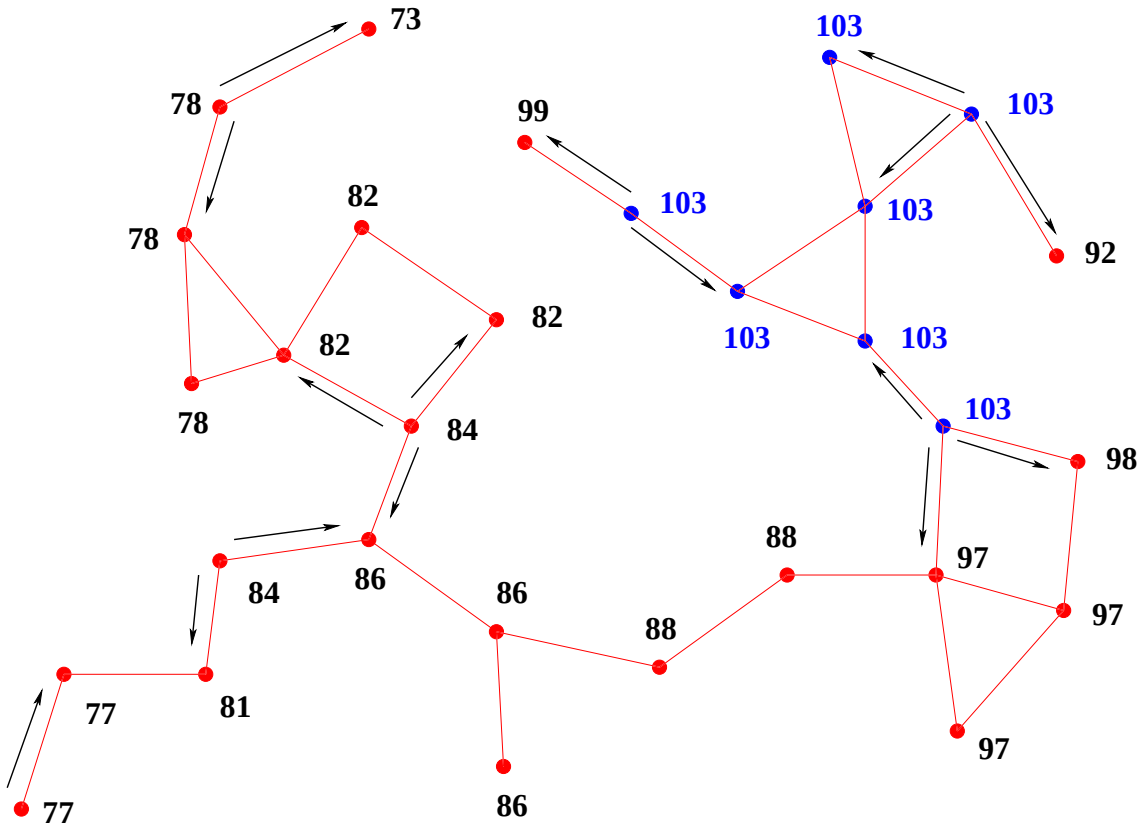
Synchronization Phase (r Network)

M. G. Gouda and T. Herman, *Stabilizing Unison.* IPL, 1990.



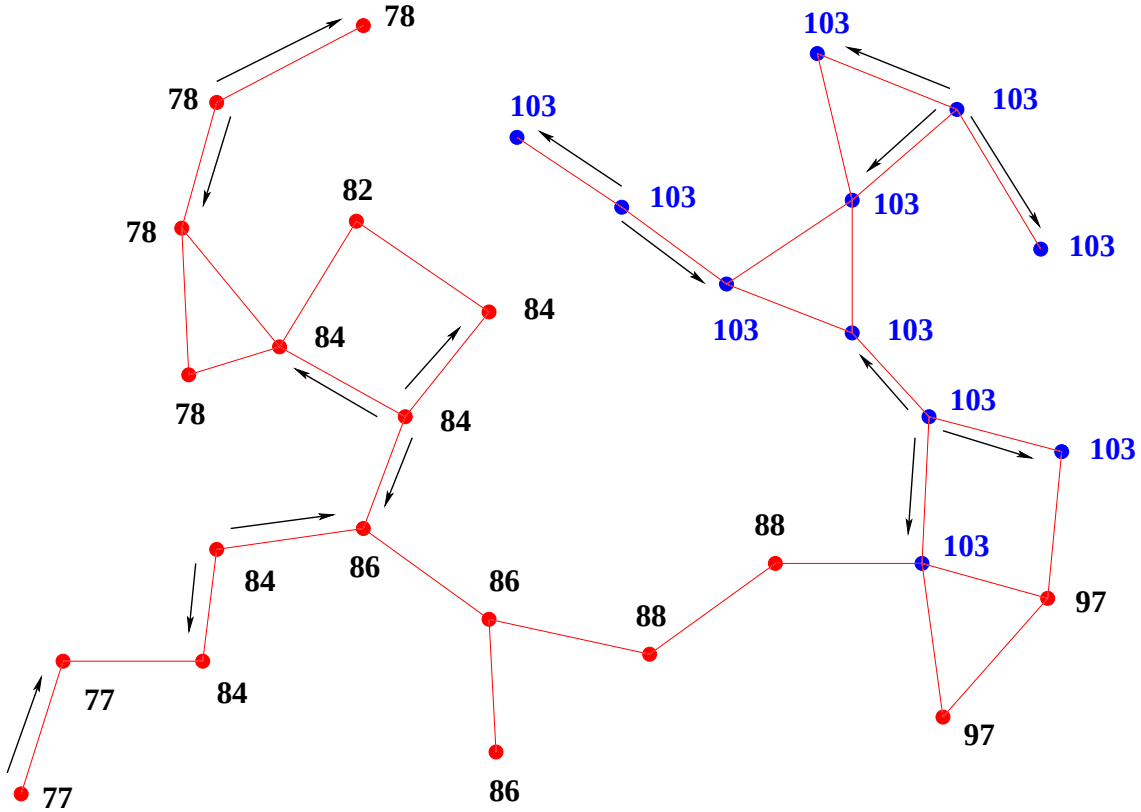
Synchronization Phase (r Network)

M. G. Gouda and T. Herman, *Stabilizing Unison.* IPL, 1990.



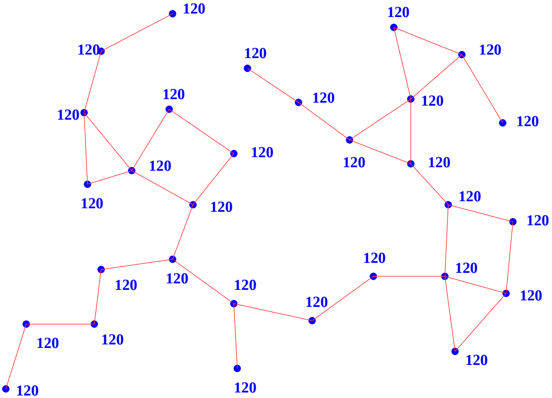
Synchronization Phase (r Network)

M. G. Gouda and T. Herman, *Stabilizing Unison.* IPL, 1990.



Synchronization Phase (r Network)

M. G. Gouda and T. Herman, *Stabilizing Unison.* IPL, 1990.



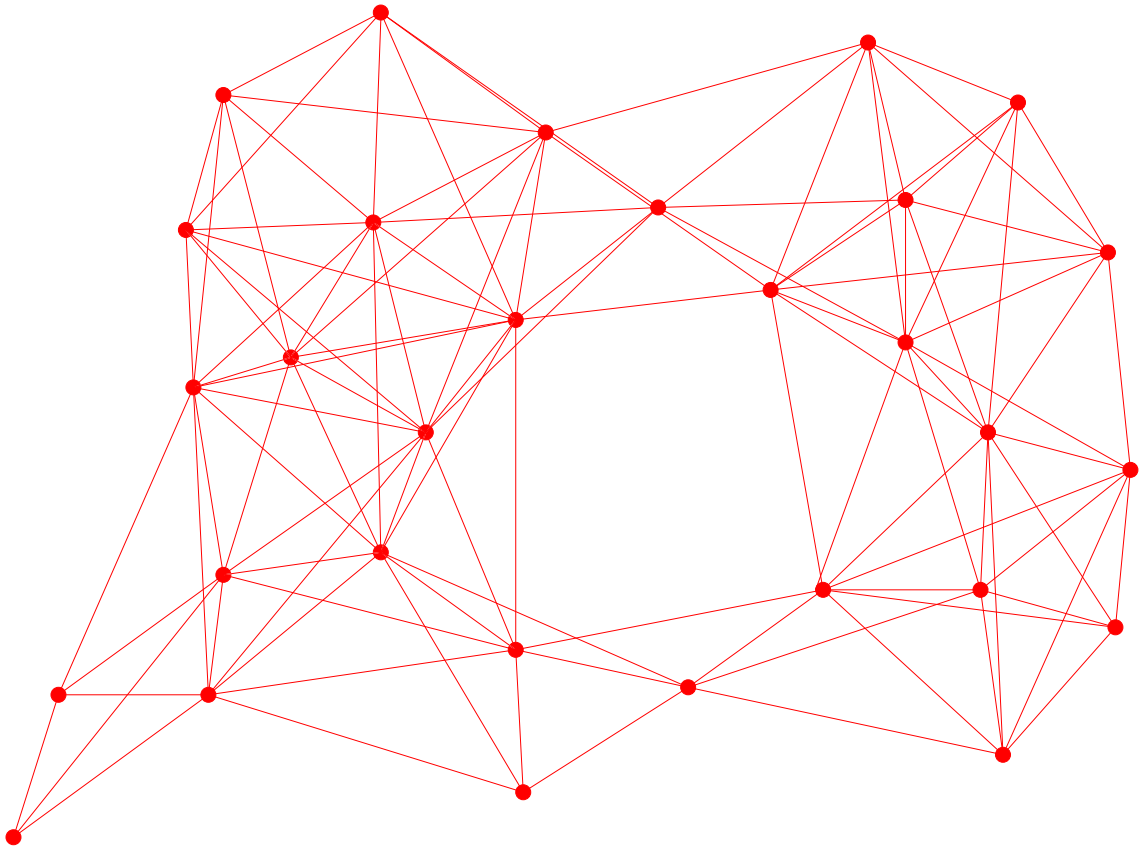
Synchronization Result

Theorem

The synchronization phase solves the synchronization problem under any ∞-adversary with synchronization time $T_1 + T_2$, where $T_1 = 3n^2 + 2nT$ and $T_2 = 2nT$.

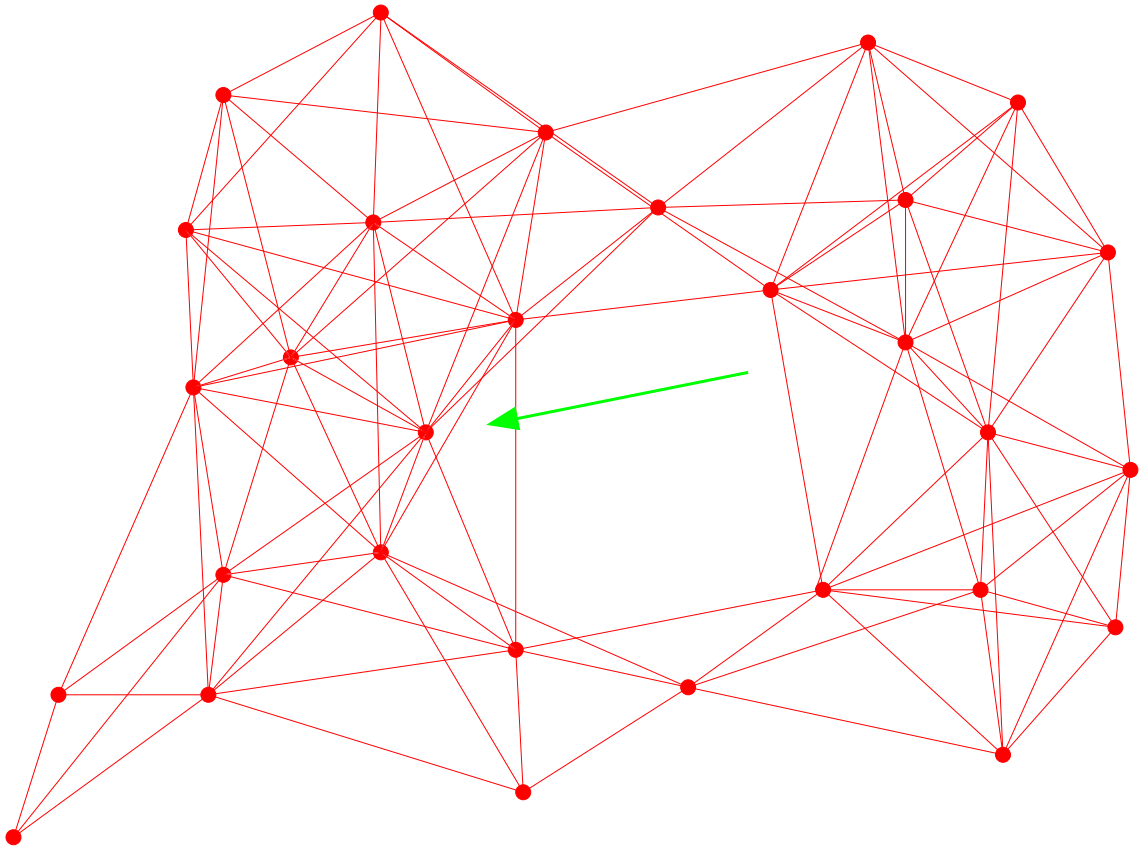
Coloring Phase

$2r$ Network



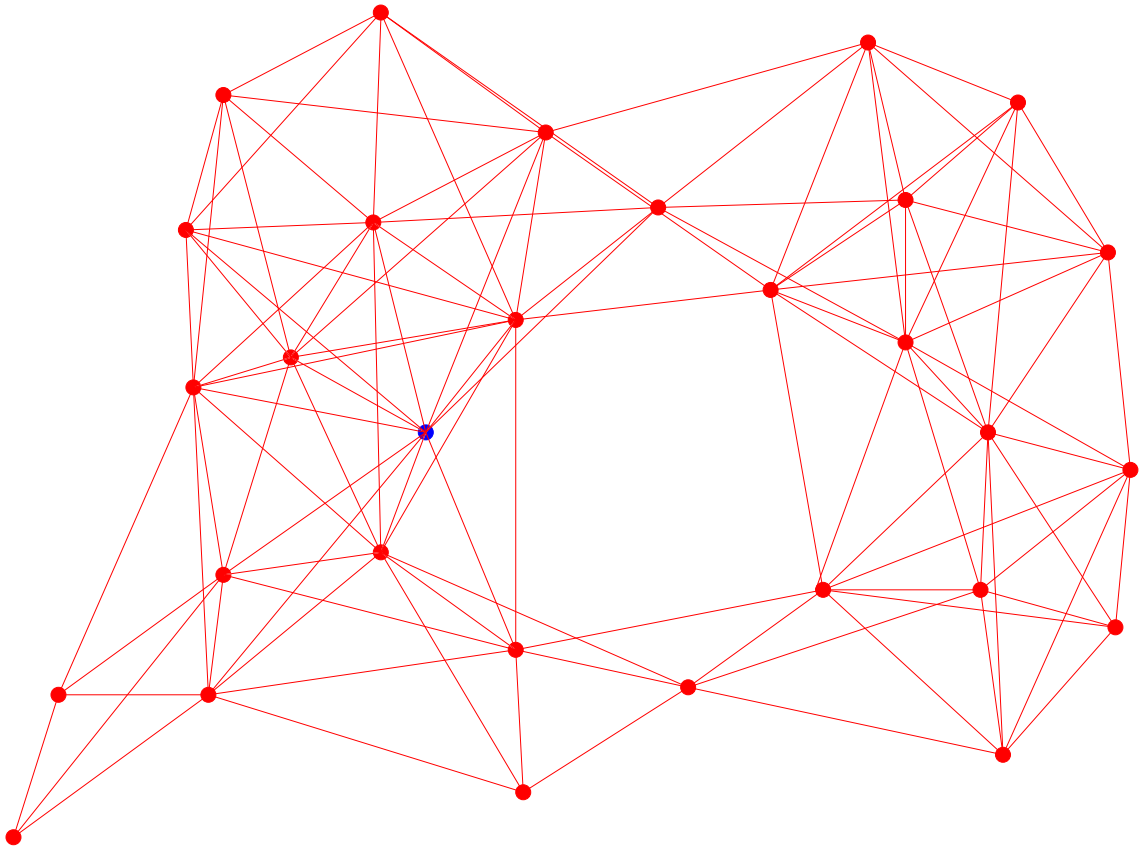
Coloring Phase

$2r$ Network



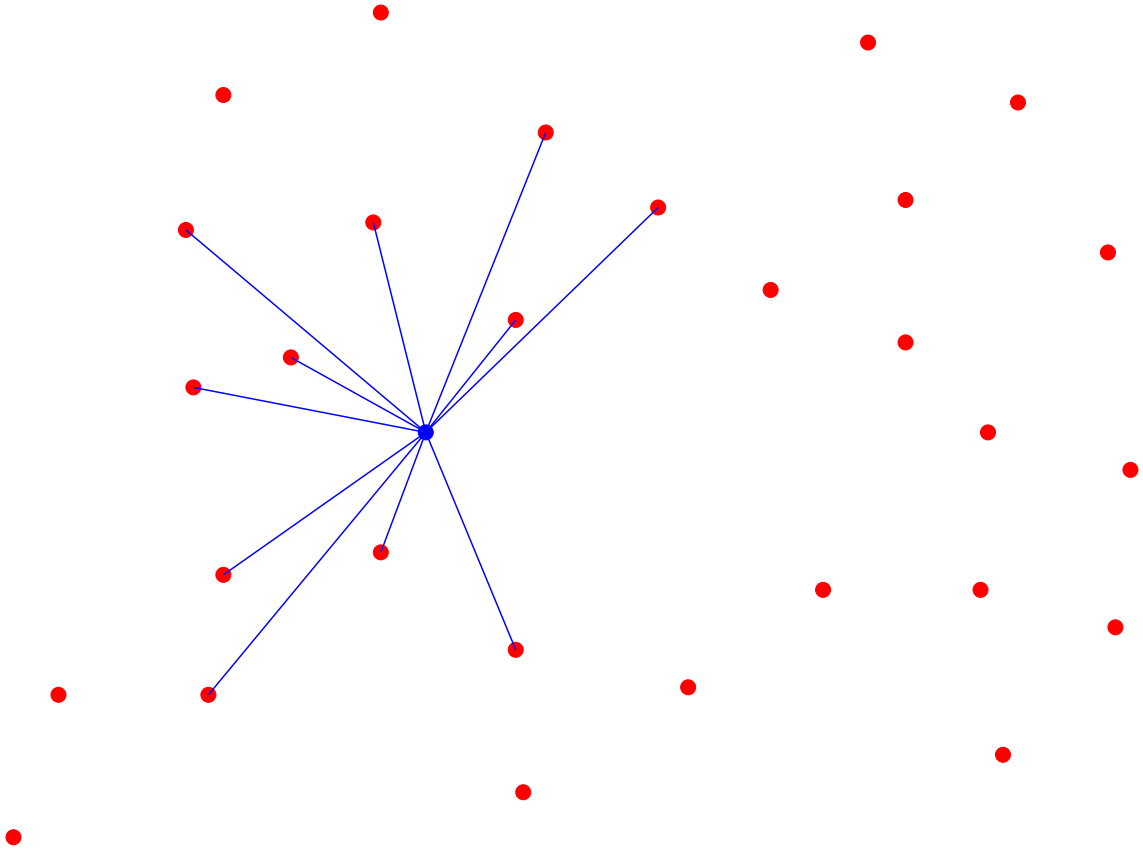
Coloring Phase

$2r$ Network



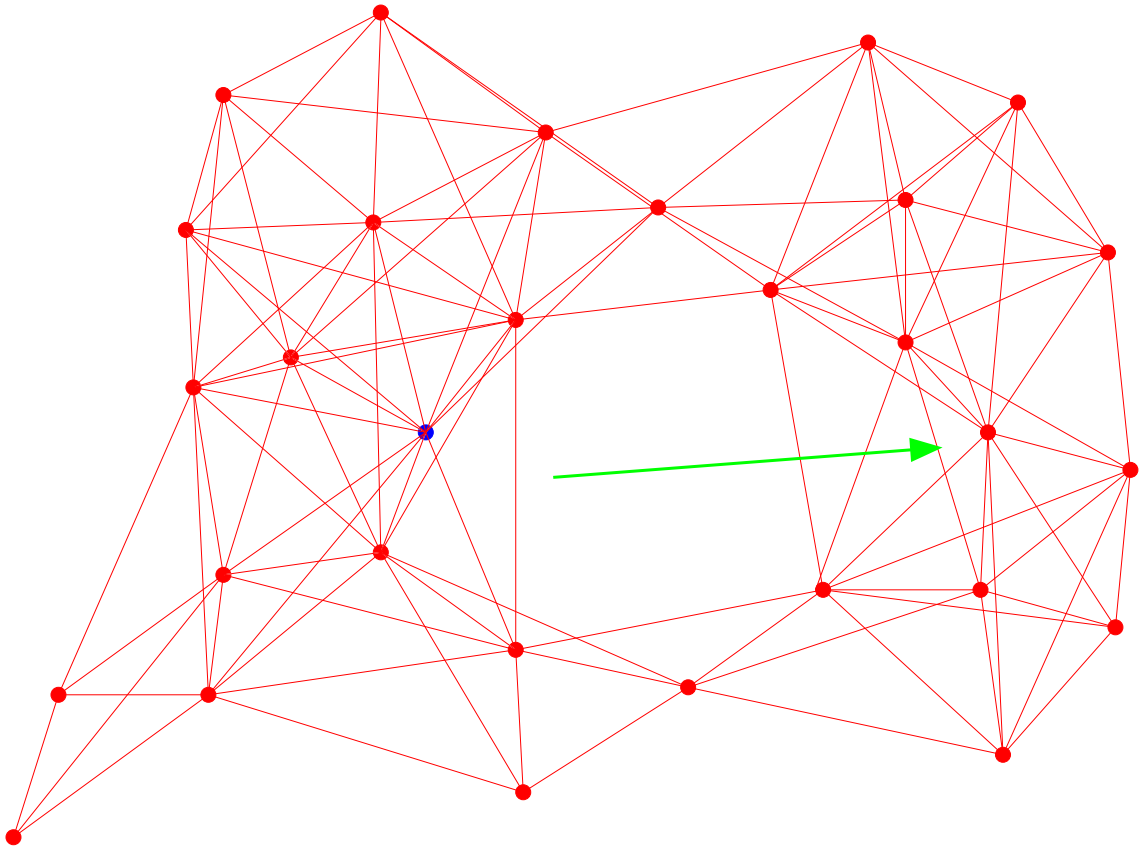
Coloring Phase

$2r$ Network



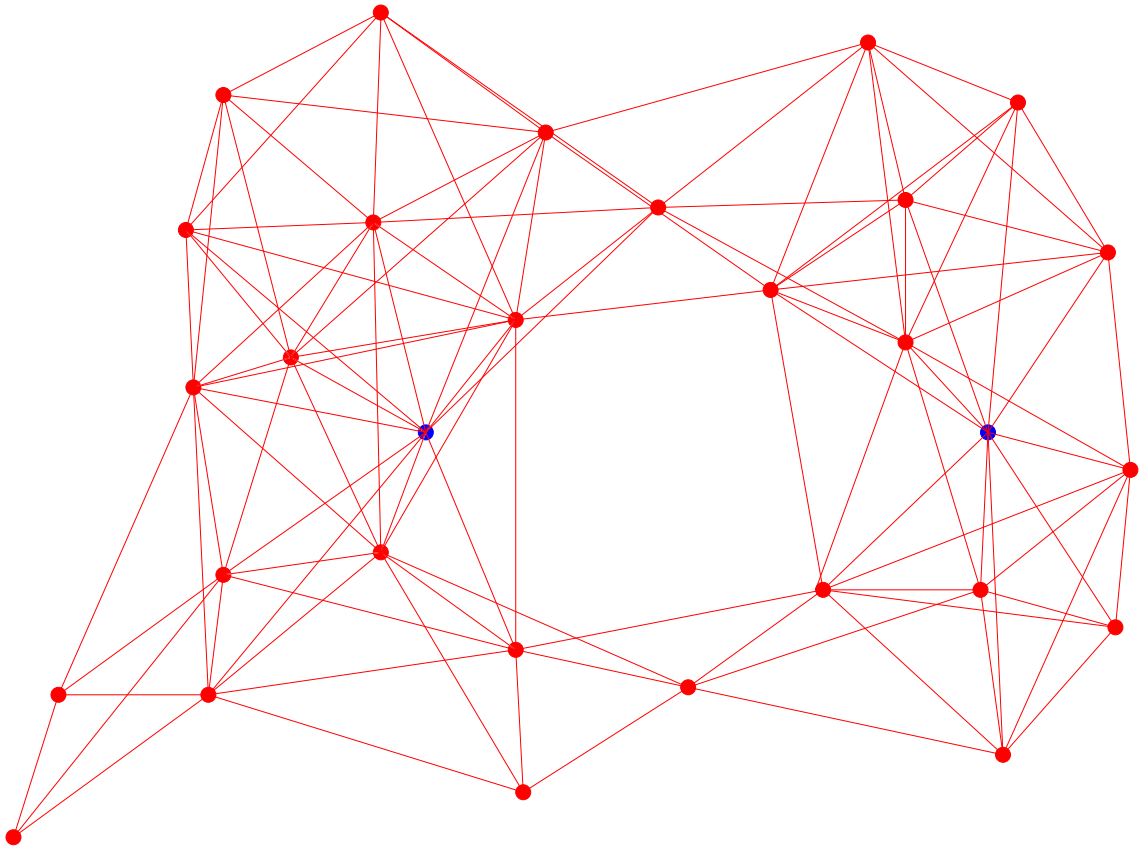
Coloring Phase

$2r$ Network



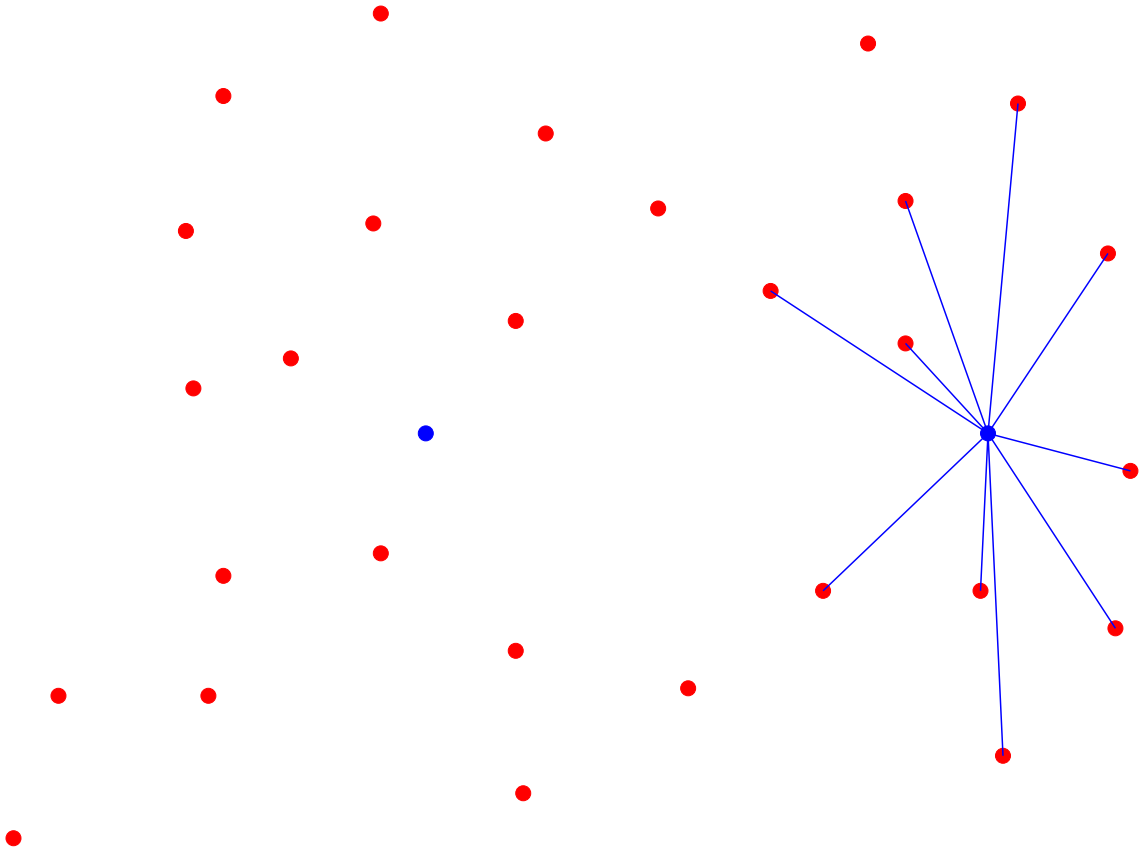
Coloring Phase

$2r$ Network



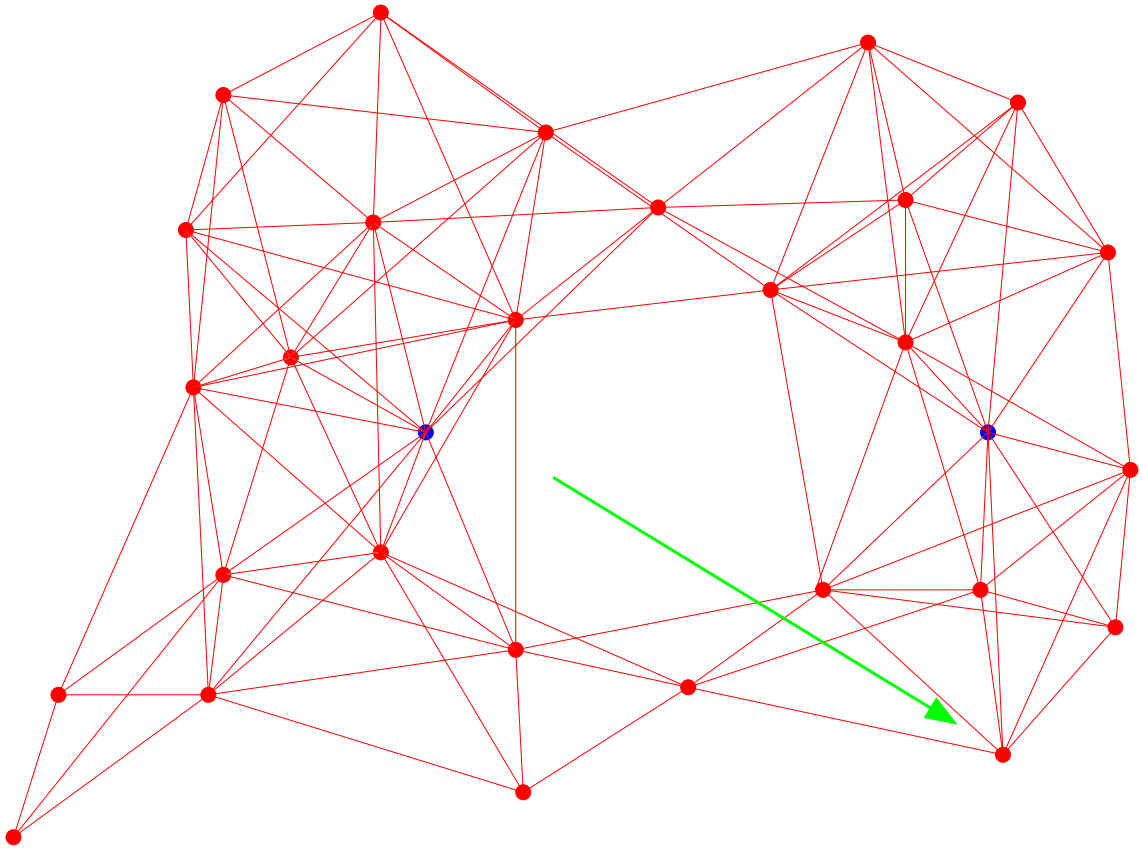
Coloring Phase

$2r$ Network



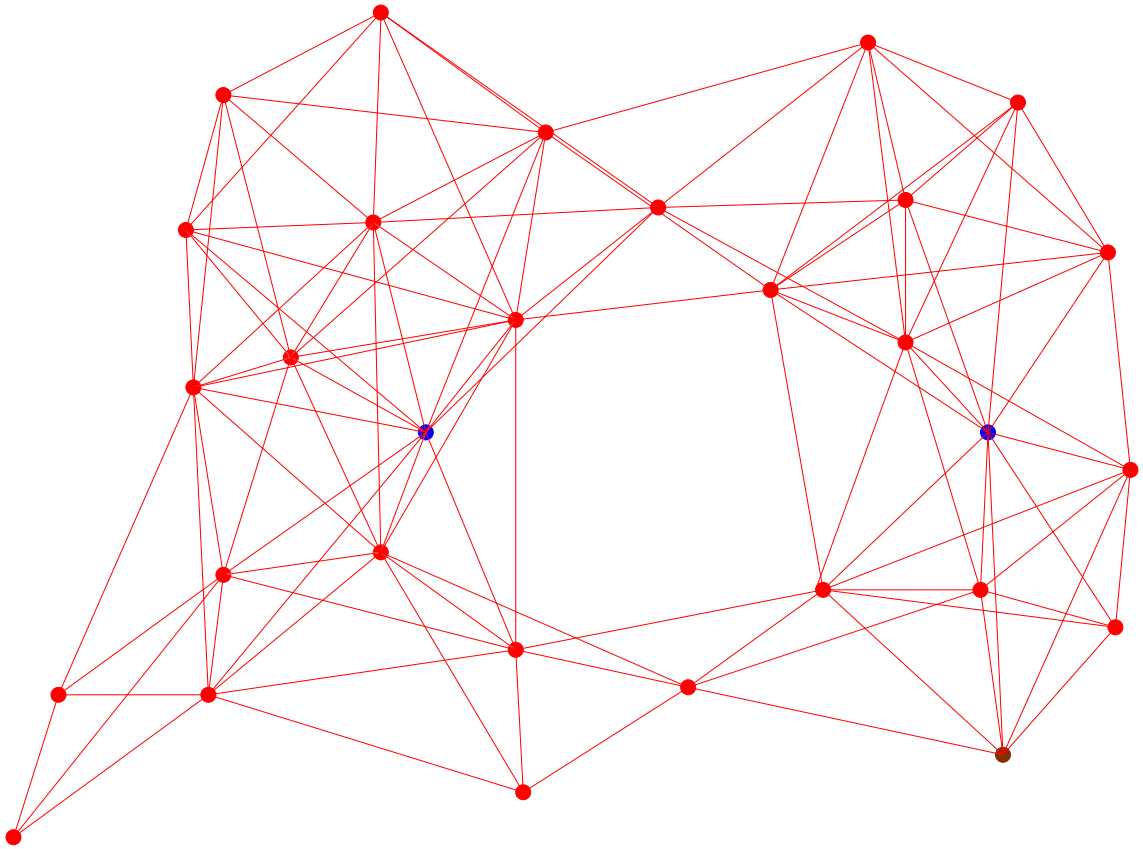
Coloring Phase

$2r$ Network



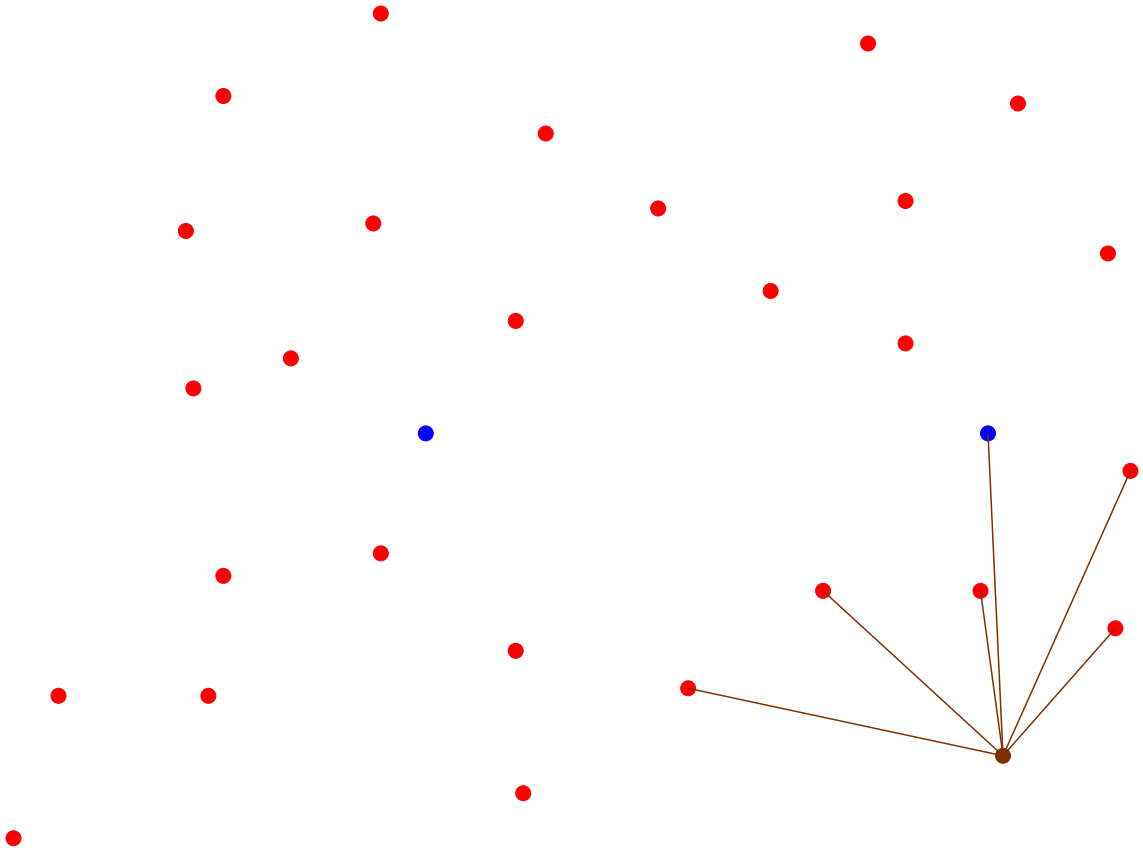
Coloring Phase

$2r$ Network



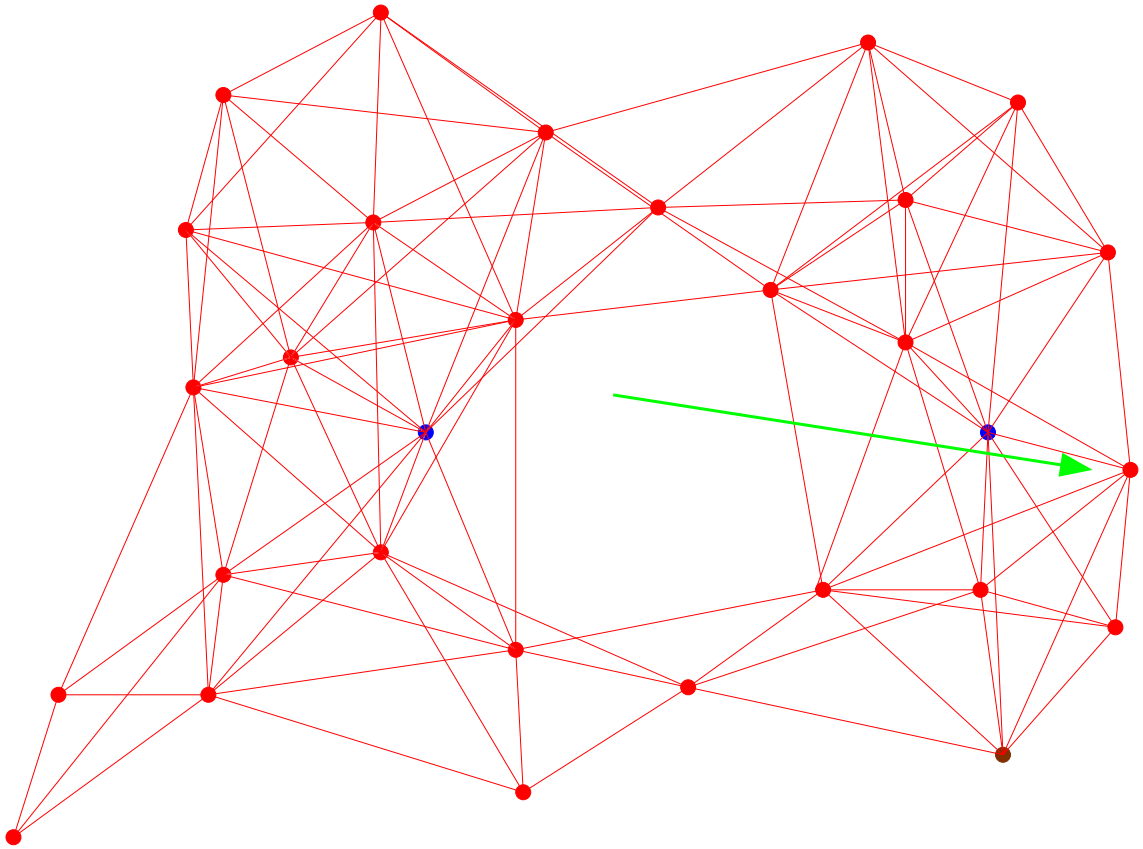
Coloring Phase

$2r$ Network



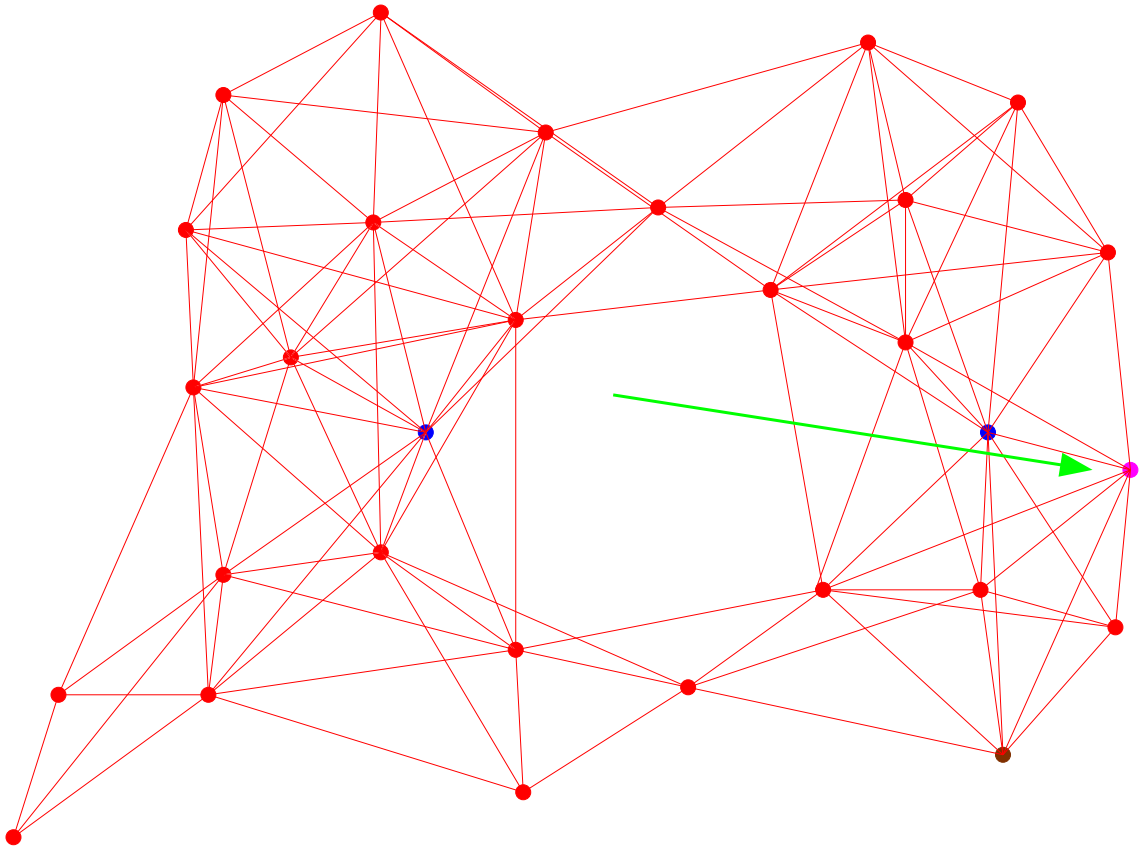
Coloring Phase

$2r$ Network



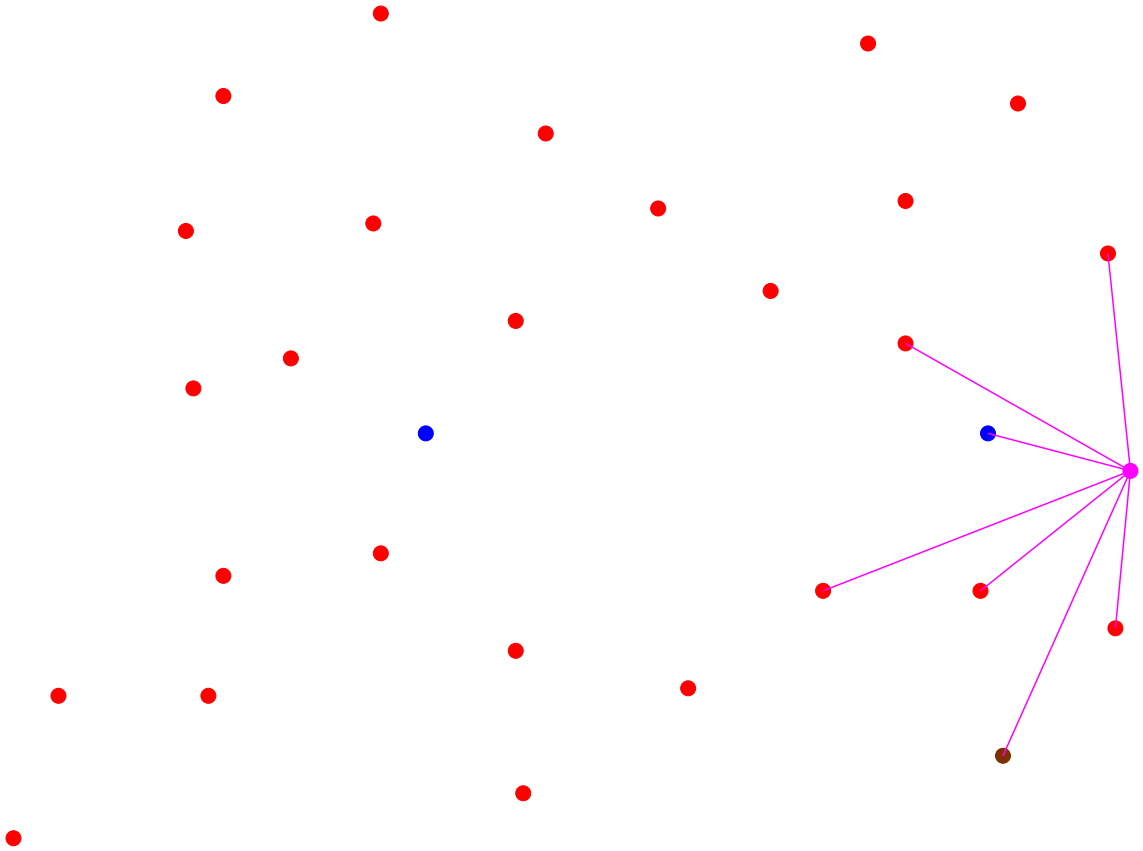
Coloring Phase

$2r$ Network



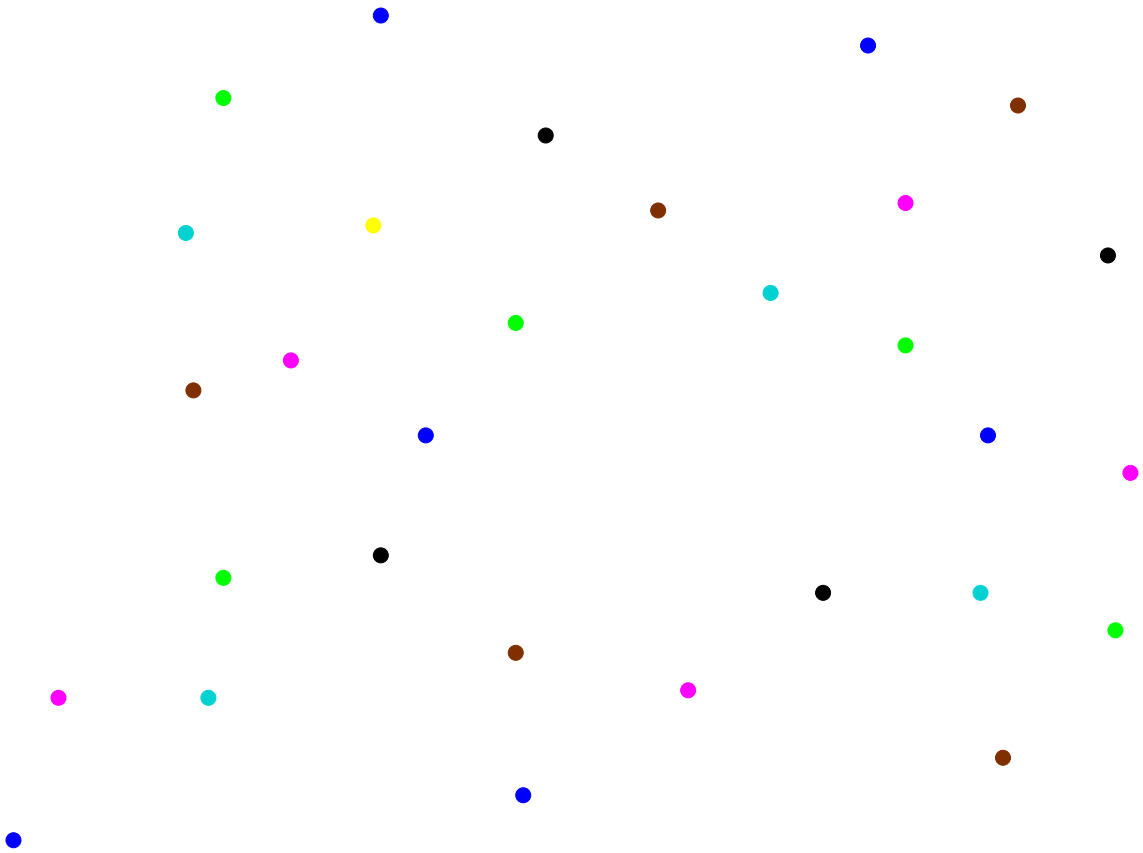
Coloring Phase

$2r$ Network



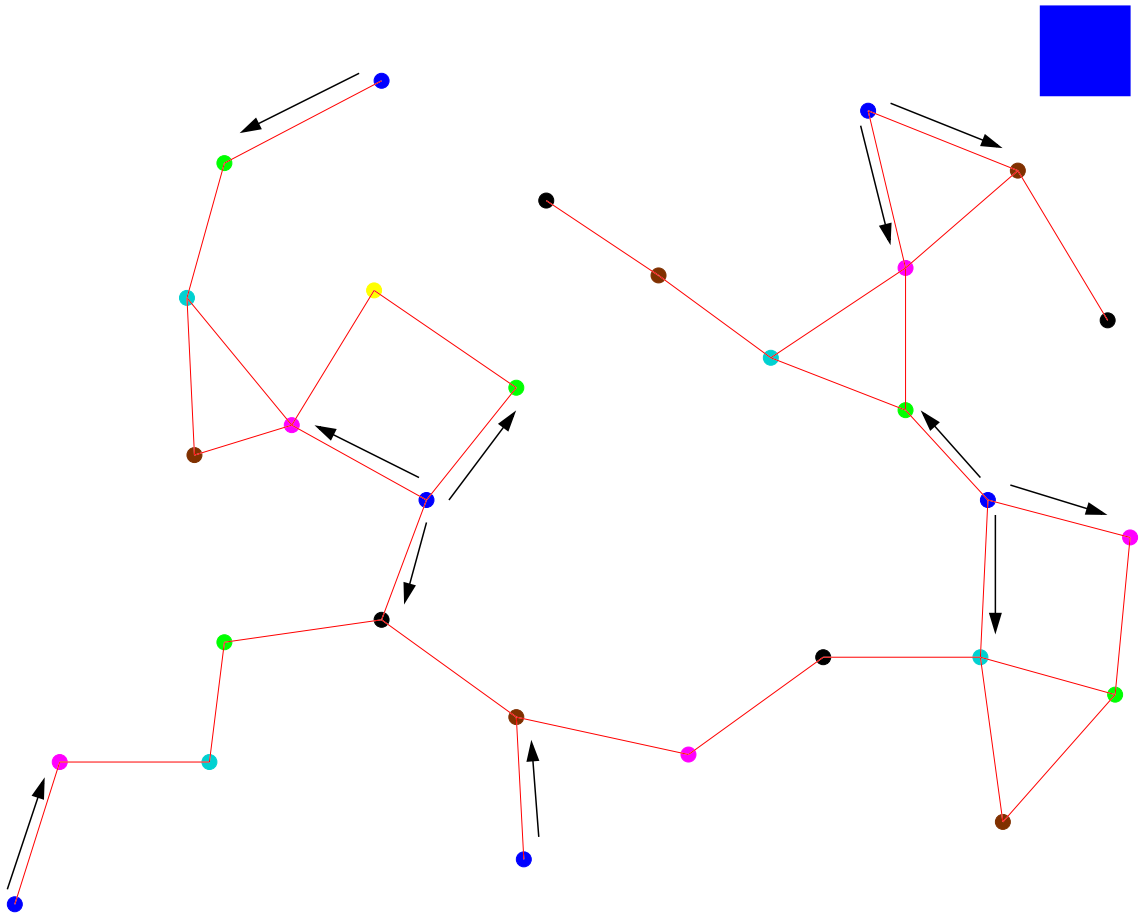
Coloring Phase

$2r$ Network



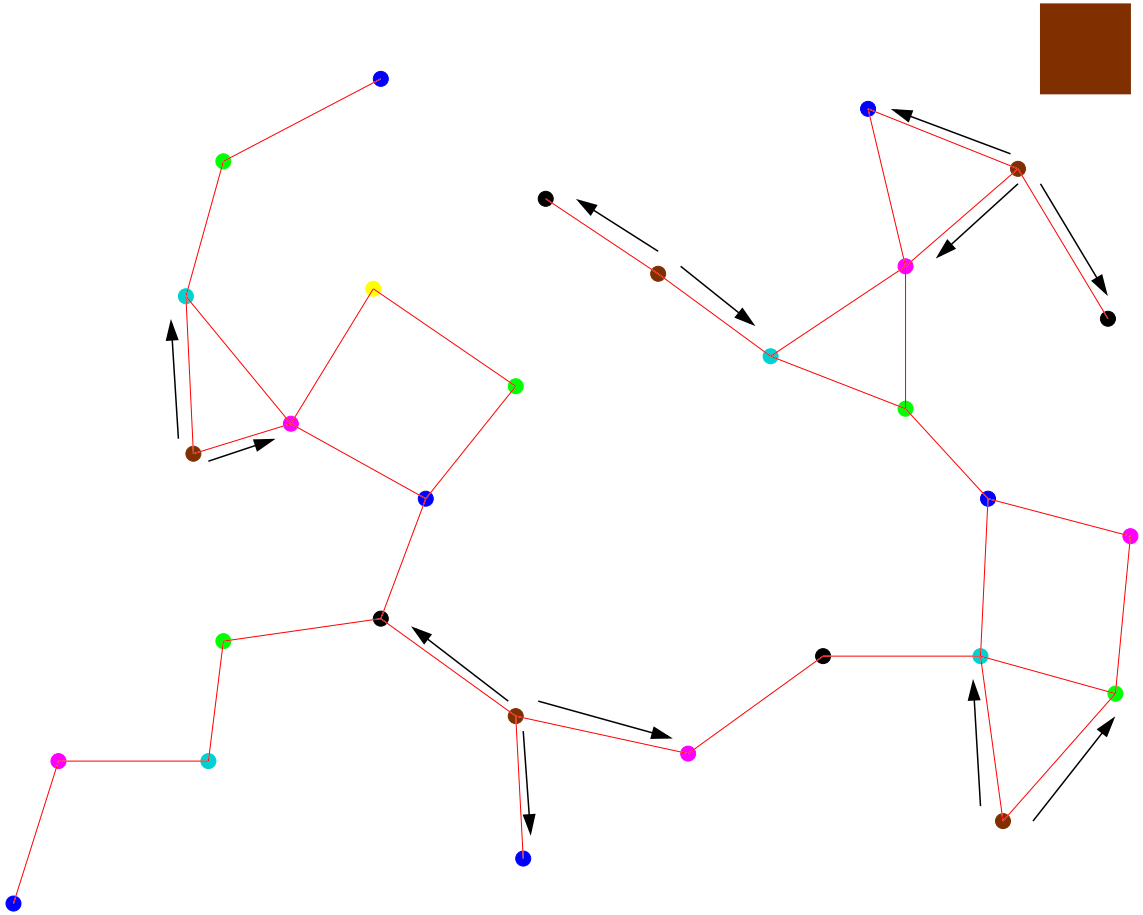
Aplication Phase

r Network



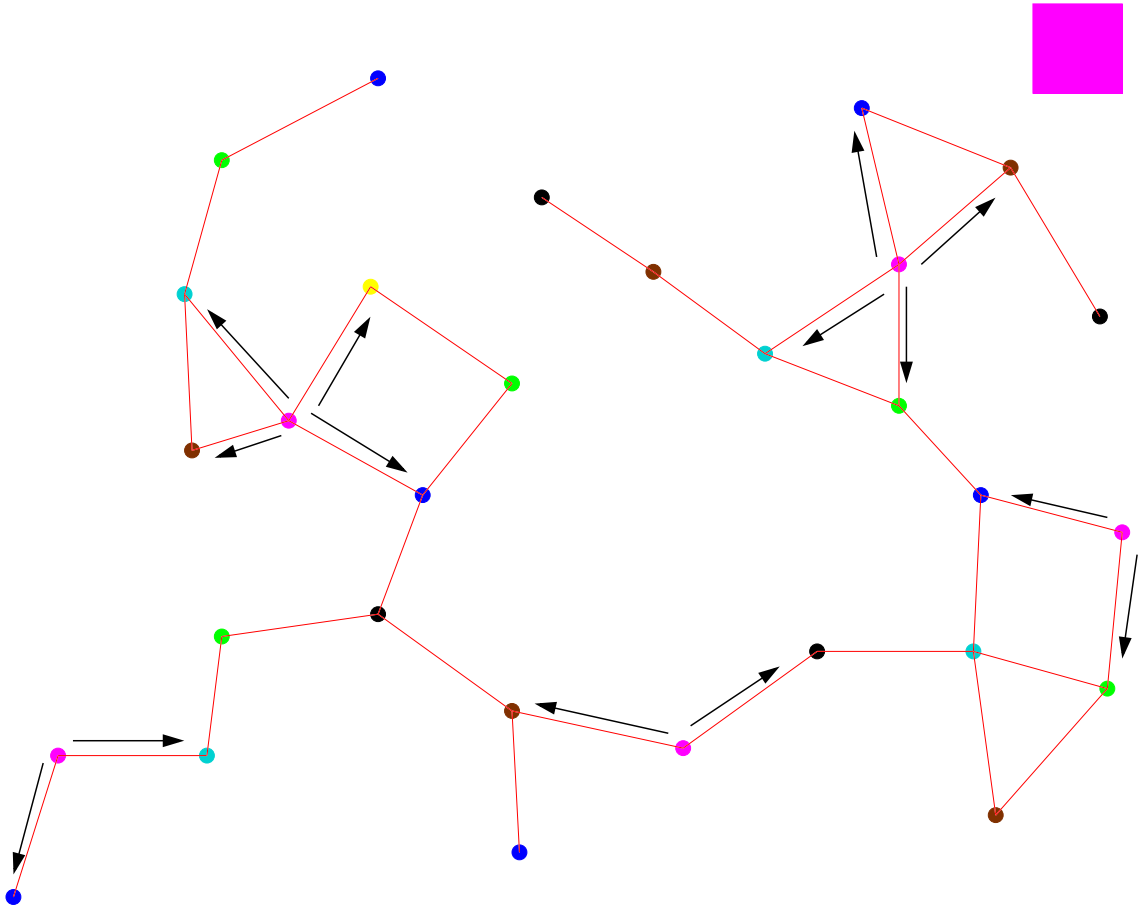
Aplication Phase

r Network



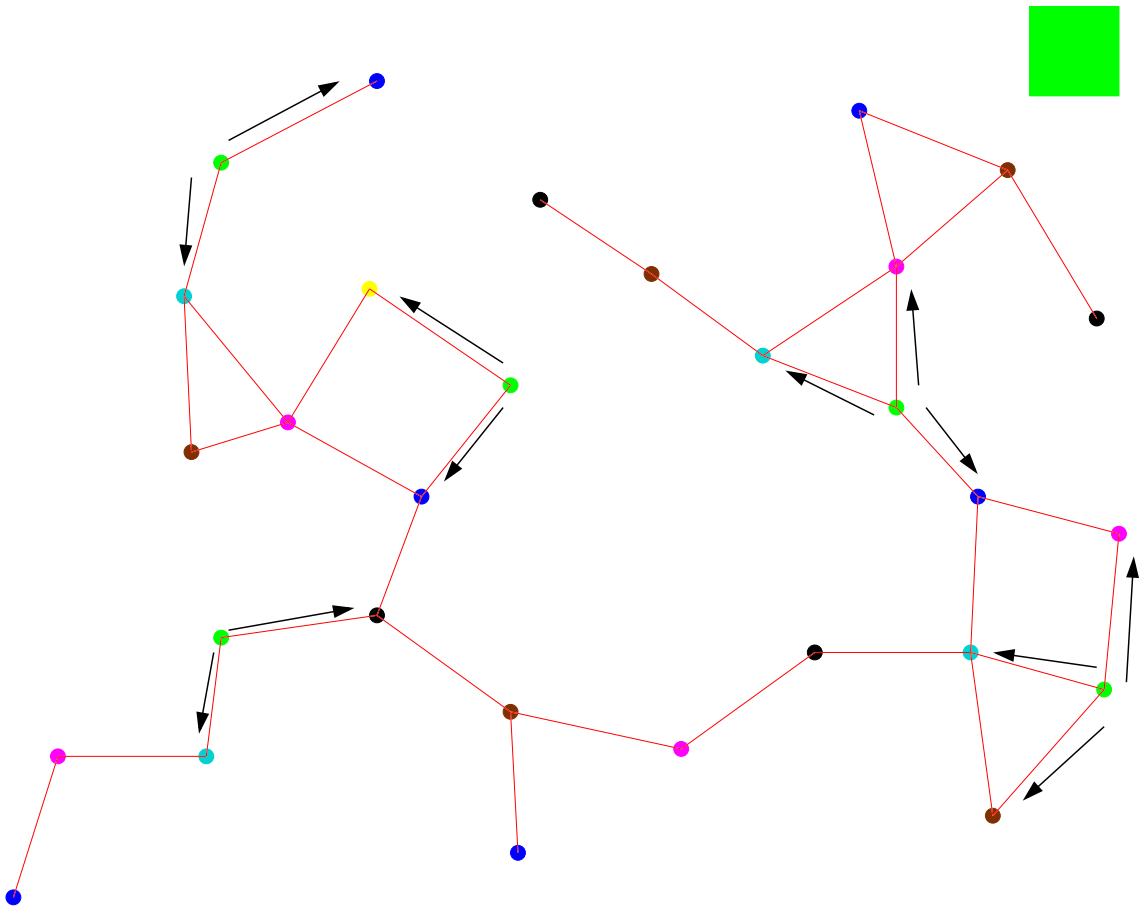
Aplication Phase

r Network



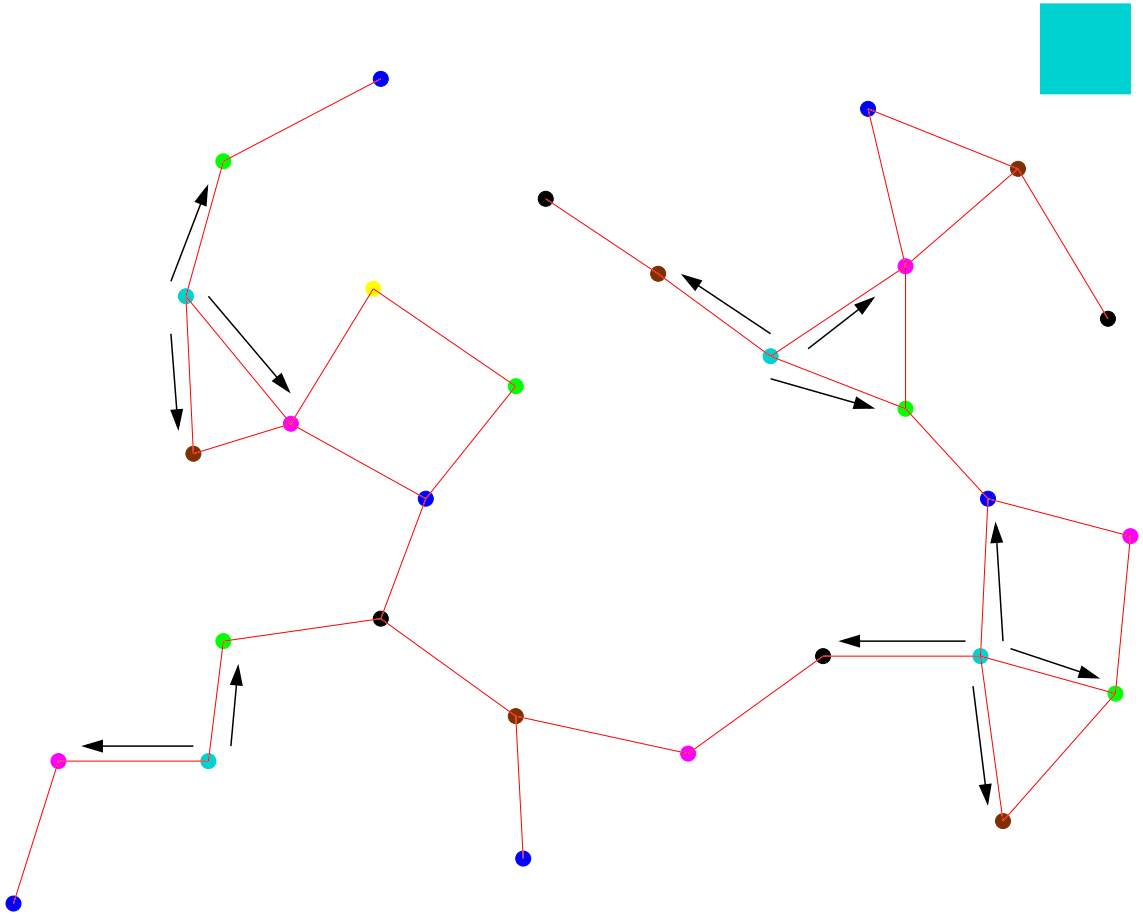
Aplication Phase

r Network



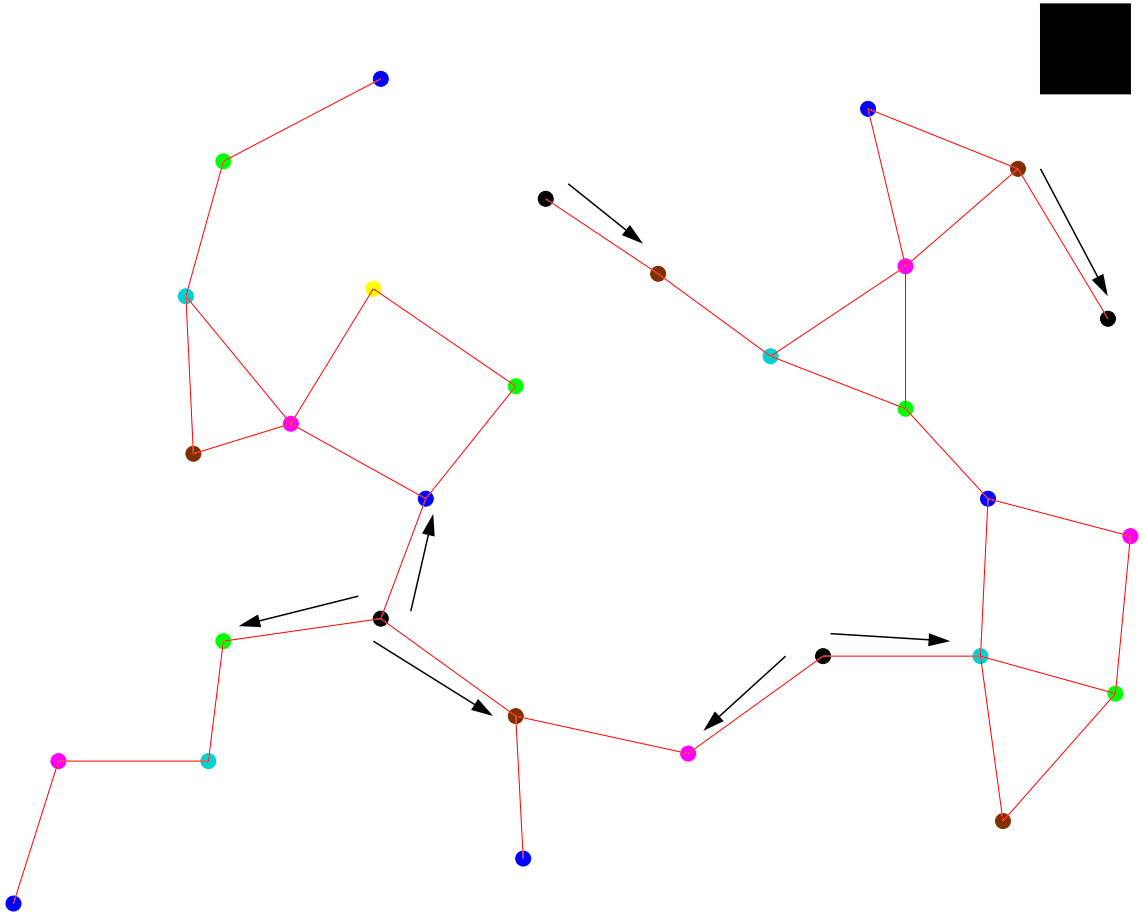
Aplication Phase

r Network



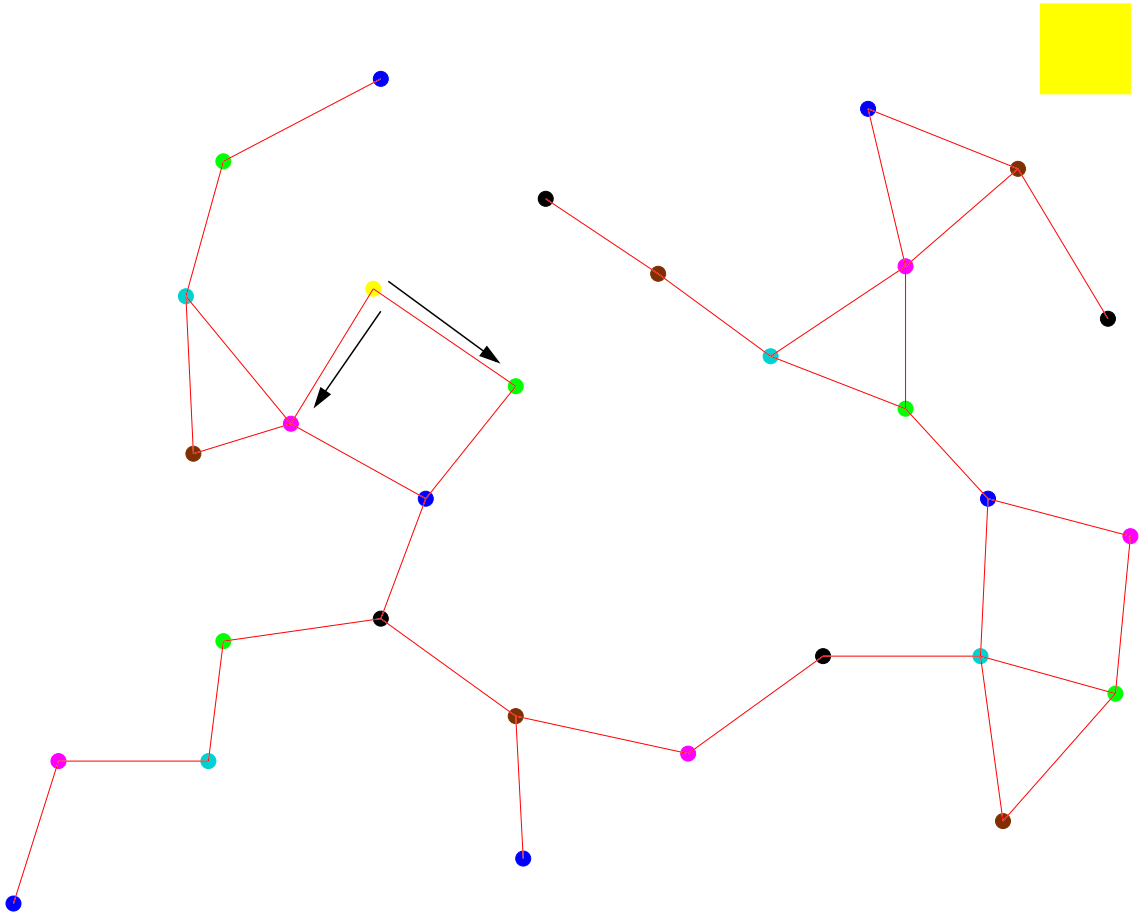
Aplication Phase

r Network



Aplication Phase

r Network



Our Results

Theorem

Given a Sensor Network of n nodes, the protocol presented solves the DRC problem under a τ-adversary with stabilization time at most $D \cdot T + \tau + n$, where T is the delay of the `ORC` protocol. The delay of this DRC protocol is $19(k+1)$ which is asymptotically optimal, and the message complexity is 0 which is optimal.

Theorem

Given a Sensor Network of n nodes, upon being woken up by a ∞-adversary, the protocol presented solves the DRC problem under an ∞-adversary with stabilization time at most $6n^2 + 4nT + 4n$, where T is the delay of the `ORC` protocol. The delay of this DRC protocol is $38(k+1)$ and the message complexity is $19(k+1)/n$, which are both asymptotically optimal.

Open Problems

Open Problems

- Reduce the stabilization time.
- How to merge disconnected components

  $\Rightarrow$ extend the failure model.

Open Problems

- Reduce the stabilization time.
- How to merge disconnected components

  $\Rightarrow$ extend the failure model.

Thank you