Distributed Coordination in Swarms of Autonomous Mobile Robots

Franck Petit

Based on materials by:
- Giuseppe Prencipe, University of Pisa
- Nicola Santoro. Carlton University
- Paola Focchini,
- P. Widmayer,
- D. Peleg,
- X. Défago, JAIST
- M. Gradinariu, UPMC
- Y. Katayama
- V. Gervasi

Setting and Motivations

Robot Swarms

Swarm: Collection of independent, autonomously operating mobile robots

Robot Swarms

Swarm: Collection of independent, autonomously operating mobile robots

Typically, the robots in a swarm are

- Very small
- Very simple
- Very limited in capabilities:
  - Weak energy resources
  - Limited means of communication
  - Limited processing power

Why Multiple Robot Systems?

- Low cost: use several cheap & simple robots rather than single expensive one
- Can solve tasks impossible for a single robot (*e.g.*, sweep large regions)
- Can perform risky / hard tasks in hazardous / harsh environments
- Can tolerate destruction of some robots
- Applications:
  - Military operations
  - Space explorations
  - Search&rescue missions
  - Toxic spill cleanups
  - Fire fighting
  - Risky area surrounding or surveillance
  - Exploration of awkward environments
  - Large-scale construction
  - Environmental monitoring

Specific Tasks for Swarms

- Movement management
  - Movement limitation
  - Collision avoidance
  - 2D/3D settings
- Complex coordination operations
- Global control

  Most previous work: Centralized control, suitable for small robot team, inadequate for large swarms

  → Distributed control:
    - No central coordination
    - Scalability
    - Dynamicity

Setting

Distributed System whose Entities are simple units (robots) equipped with:

- Motorial Capabilities
  - Freely move on a 2 (or 3) dimensional environment
- Sensorial Capabilities
  - Sense the positions of the others in the environment

Why study oblivious (& relatively dumb) robots?

- Algorithms will work in a dynamic environment (where robots join/ leave the system)
- The system can start in (almost) any configuration
- Algorithms that work correctly for weaker robots will work for stronger robots

However….

- Few complex specialized units
  - Expensive
  - Not fault tolerant

- Many simple units
  - Not expensive
  - Modular
  - Fault tolerant

Problem:
Coordinate them

General Problem

General aim of the study

- ***Which are the elementary tasks that can be achieved deterministically?***
- ***What are the minimal conditions for this?***
- ***Given a task, what kind of local coordination is necessary so that the robots can accomplish it (deterministically)?***

Analyze from an algorithmic point of view the **distributed control** of a set of autonomous mobile robots

Problem

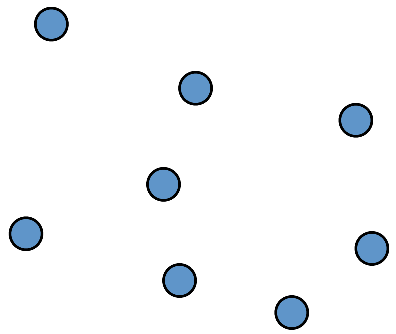
Analysis of minimum level of capabilities (sensorial, computational, and motorial) robots must possess to **COLLECTIVELY** solve a given task.

For globally accomplishing a given task: how "weak" can each single robot be?

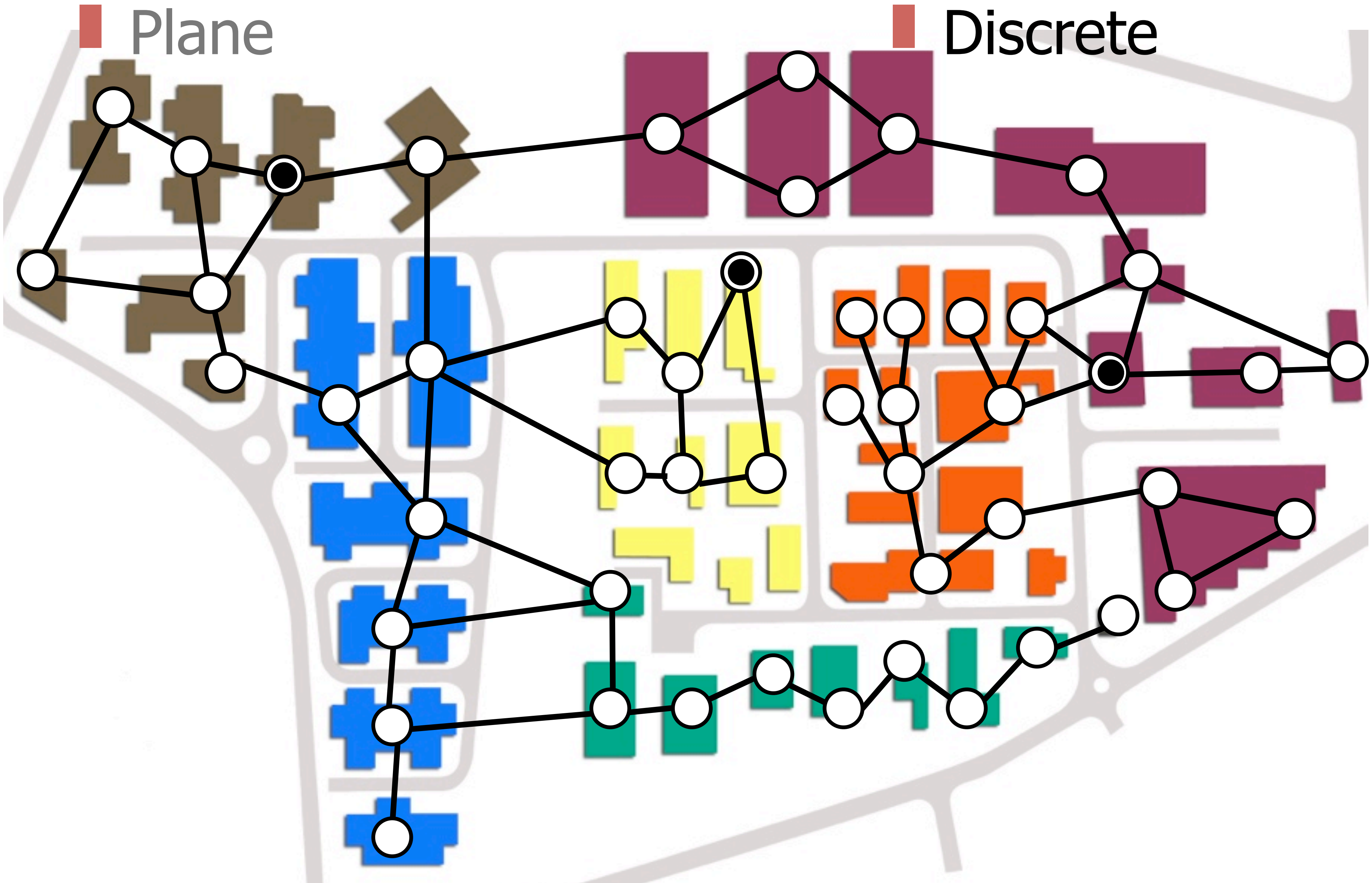
How much **local power** is necessary to perform **global computations**?

Environment

- Plane

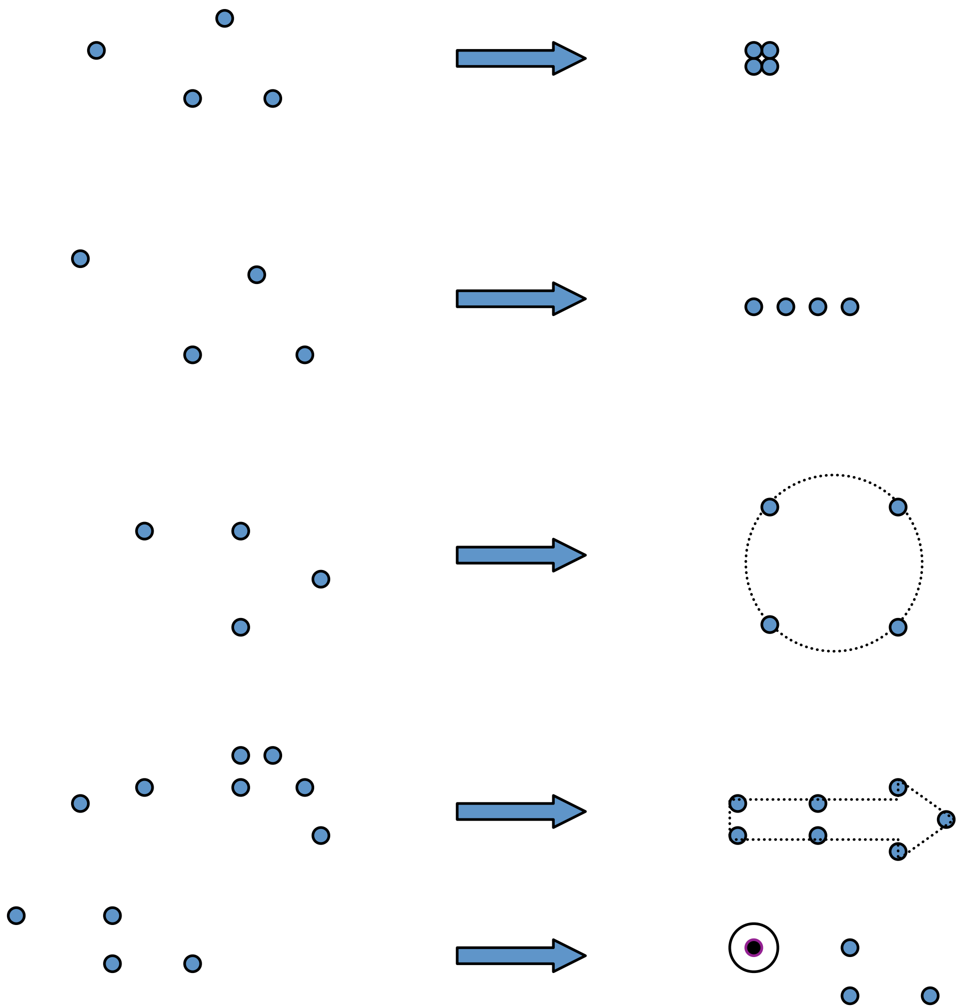


Environment



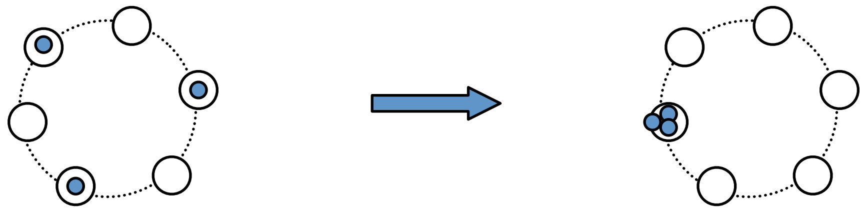
Cooperative Primitives over the Plane

- Gathering
- Alignment
- Circle Formation (n-gon)
- Other Patterns
- Election

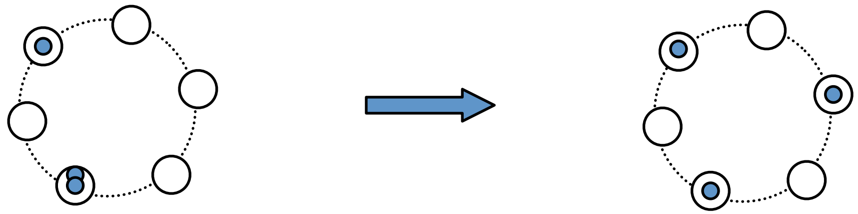


Cooperative Primitive Tasks in discrete environment

- Rendezvous



- Covering



- Exploration(s)

Approaches

Previous and Related Work

- Fukuda et al, 1989 (CEBOT)
- Brooks, 1985
- Mataric, 1994
- Cao at al, 1995 (survey)
- Durfee, 1995
- Balch and Arkin, 1998

Previous Work

Tipically...

- Heuristic solutions
- Convergence to solution

(Robotics, AI)

Very few works ...

- Provably correct solutions
- Termination in finite time

(Algorithmic approaches)

The Algorithmic Approach

- Study under what conditions on the robots' capabilities a given global task is solvable in finite time.

- Find **Algorithmic** solutions

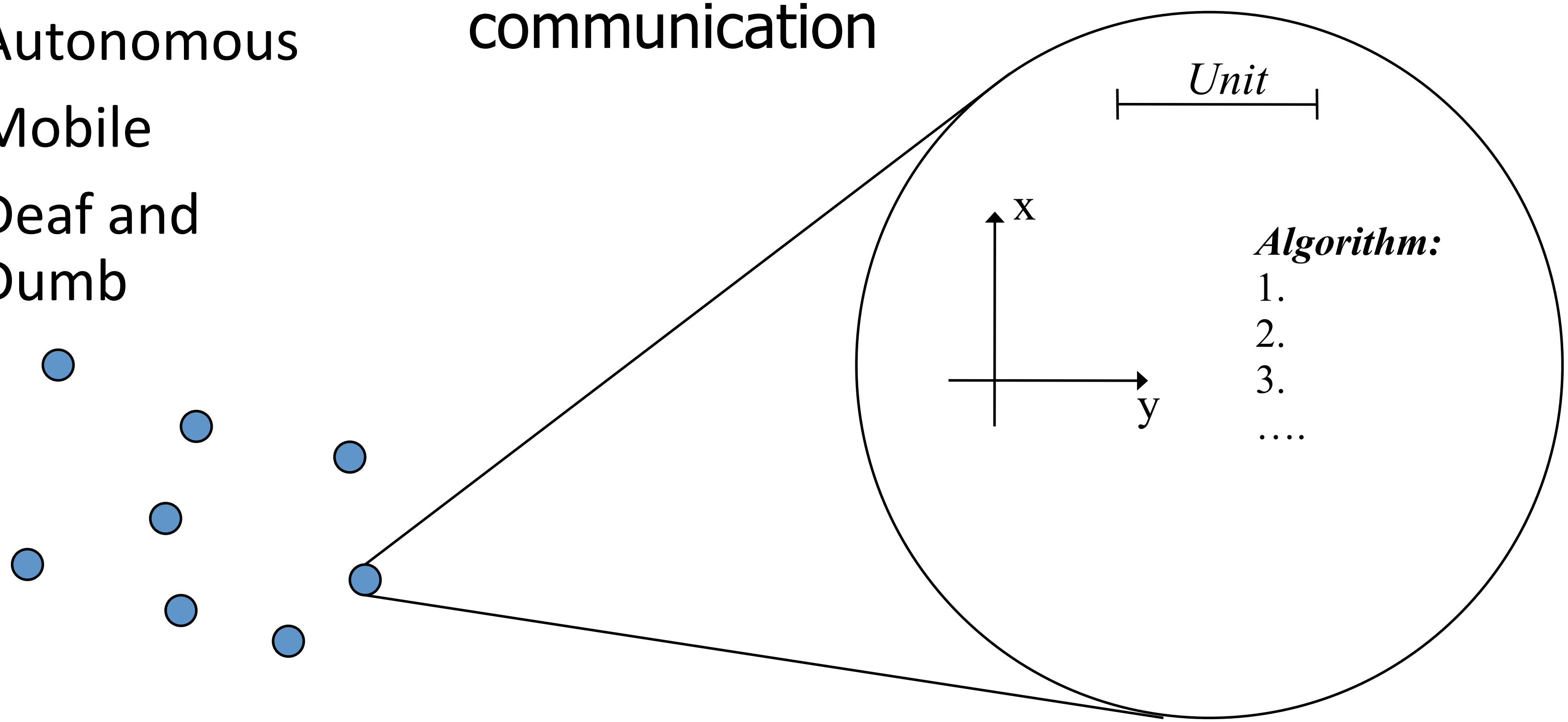
The Model

The Model

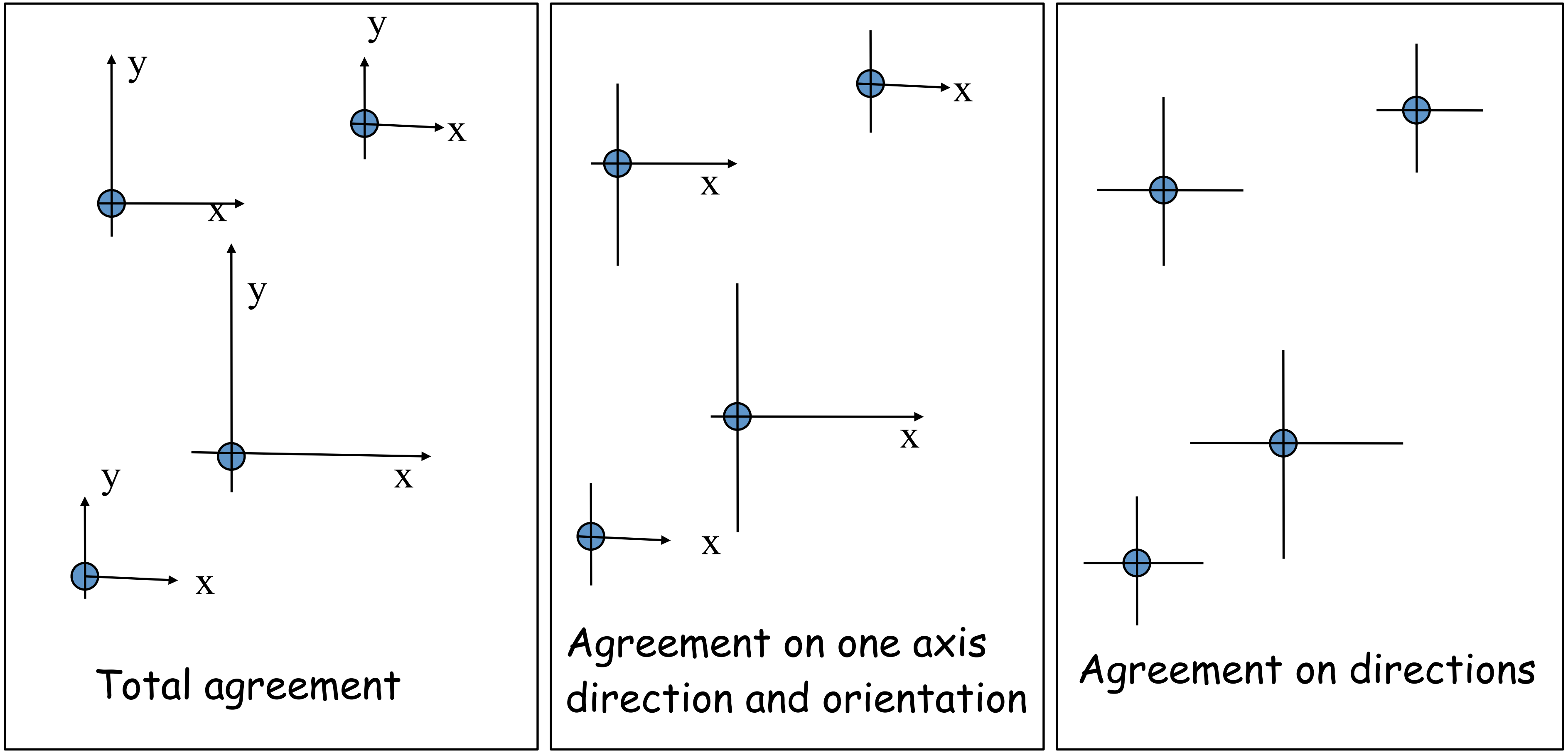
(Yamashita et al., SIROCCO 1996)

- Homogeneous
- Anonymous
- Autonomous
- Mobile
- Deaf and Dumb

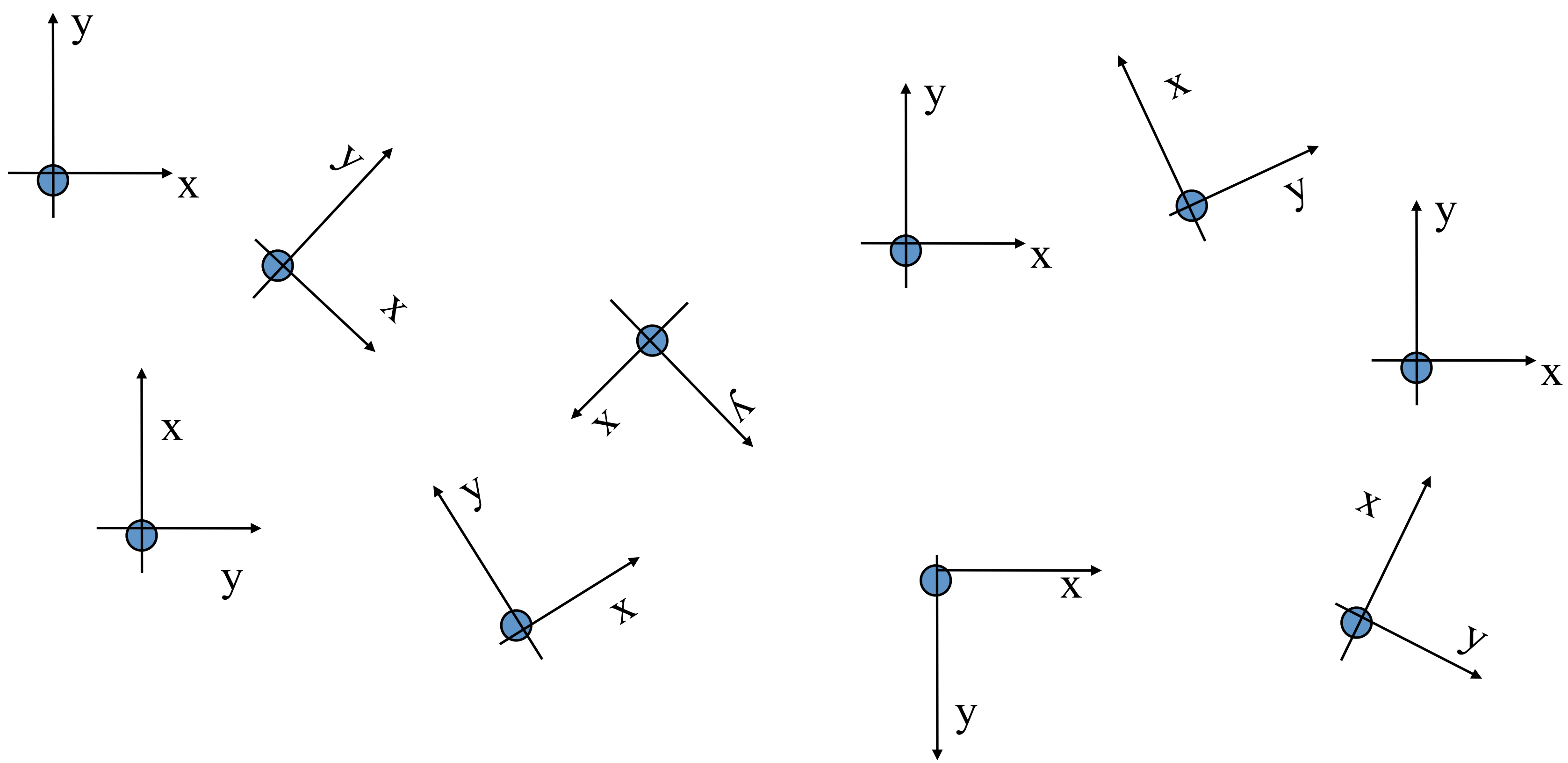
- Sensors
- No explicit communication



-Assumptions on Robots' power-
Agreement on Coordinate System



- Assumptions on Robots' power -
No Agreement

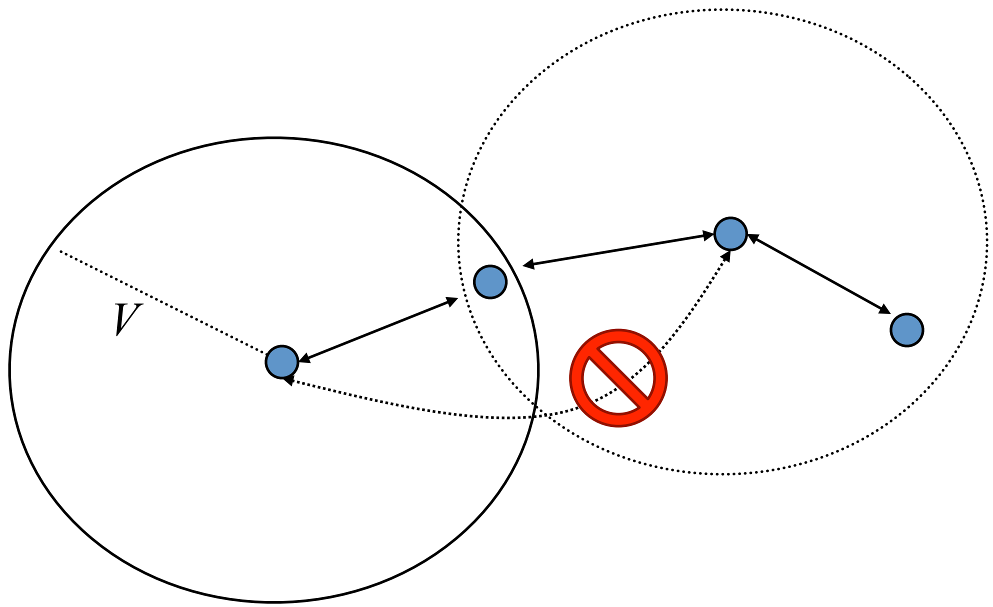


Models of Orientation

- Full-compass: Axes and polarities of both axes.
- Half-compass: Both axes known, but positive polarity of only one axis (in other axis, robots may have different views of positive direction).
- Direction-only: Both axes, but not polarities.
- Axes-only: Both axes, but not polarities. In addition, robots disagree on which axis is x and which is y.
- No-compass: No common orientation information.

Note: In general, robots do not share common unit distance or common origin point even in full-compass model

- Assumptions on Robots' Power -
Radius of Visibility: Limited /Unlimited



- Assumptions on Robots' Power -
Oblivious/Non Oblivious

Non-Oblivious: remember the positions of all the robots since the beginning of the computation

Oblivious: otherwise

Modeling Movements

Assumption 1

The maximum distance r_i can move in one step is bounded by $\varepsilon i>0$

Assumption 2

There is a lower bound $\delta_r>0$ on the distance a robot r can travel, unless its destination is closer than $\delta_r>0$.

Modeling The Time

- A critical aspect in every distributed system is the time
  - Synchronous?
  - Asynchronous?
- At the beginning the proposed model for robots was basically synchronous (SYNC, SSYNC, SYm)
  - Semi-synchronous
  - Fully-synchronous

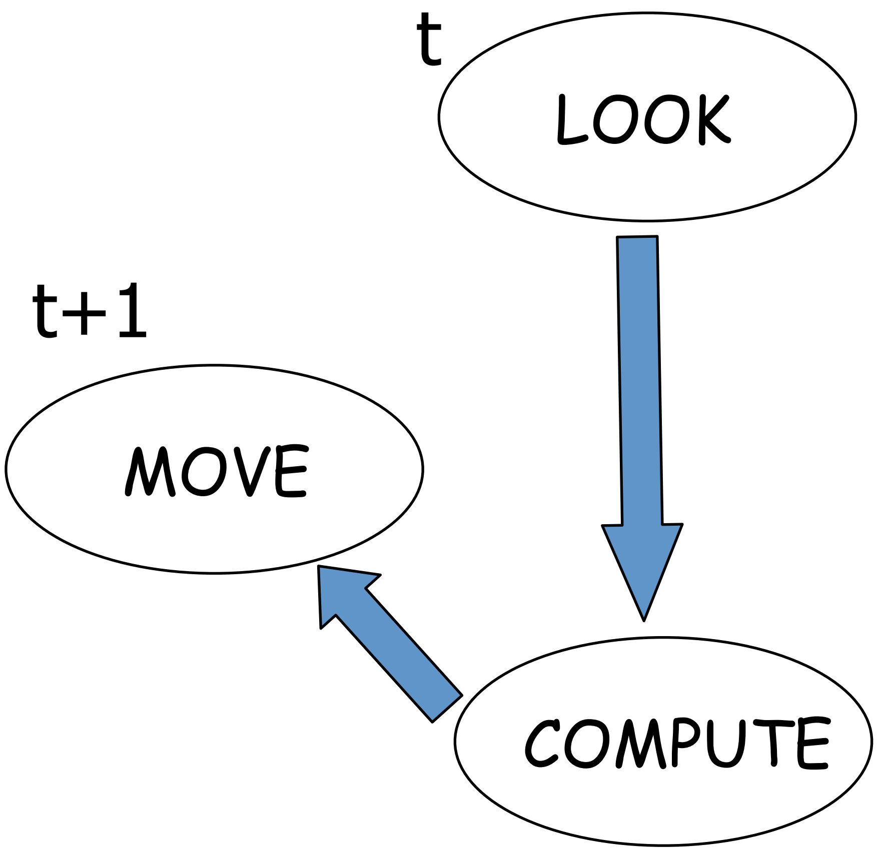
Instantaneous Actions of **SYm**

Suzuki *et al.*, 1996

- Discrete Time 0,1,...
- At each time instant t, every robot r_i is either Active or Inactive
- At least one Active robot at each time instant, and every robot is Active infinitely often

Instantaneous Actions of **SYm**

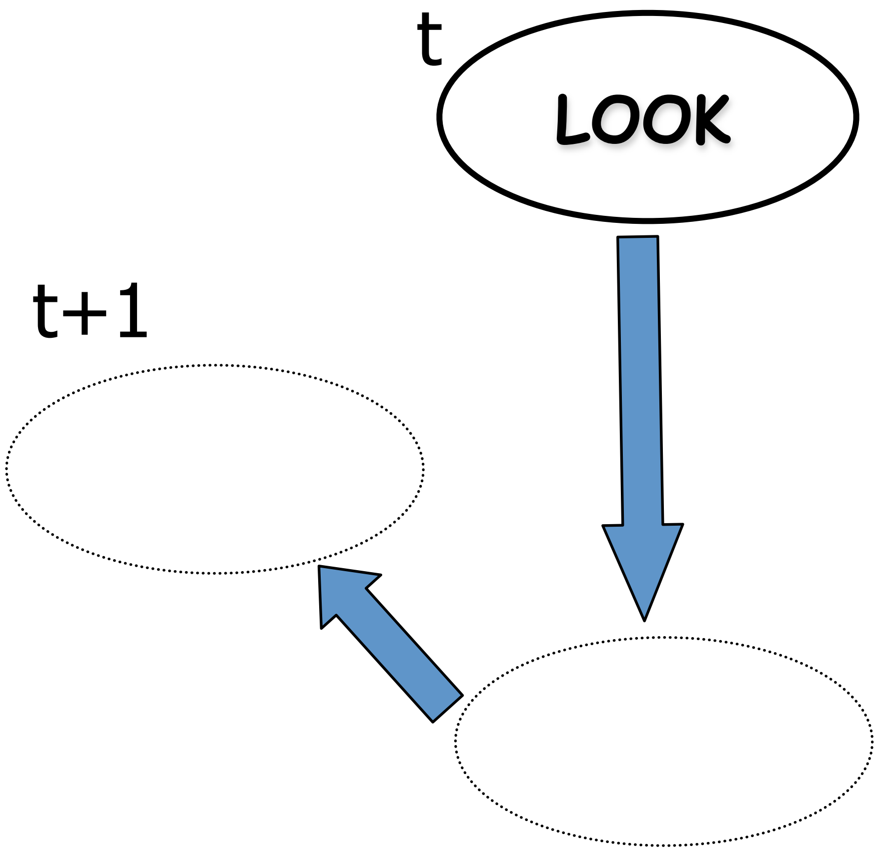
Suzuki *et al.*, 1996

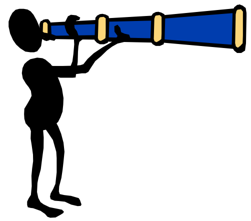


Phases of an Active robot

Instantaneous Actions of **SYm**

Suzuki *et al.*, 1996





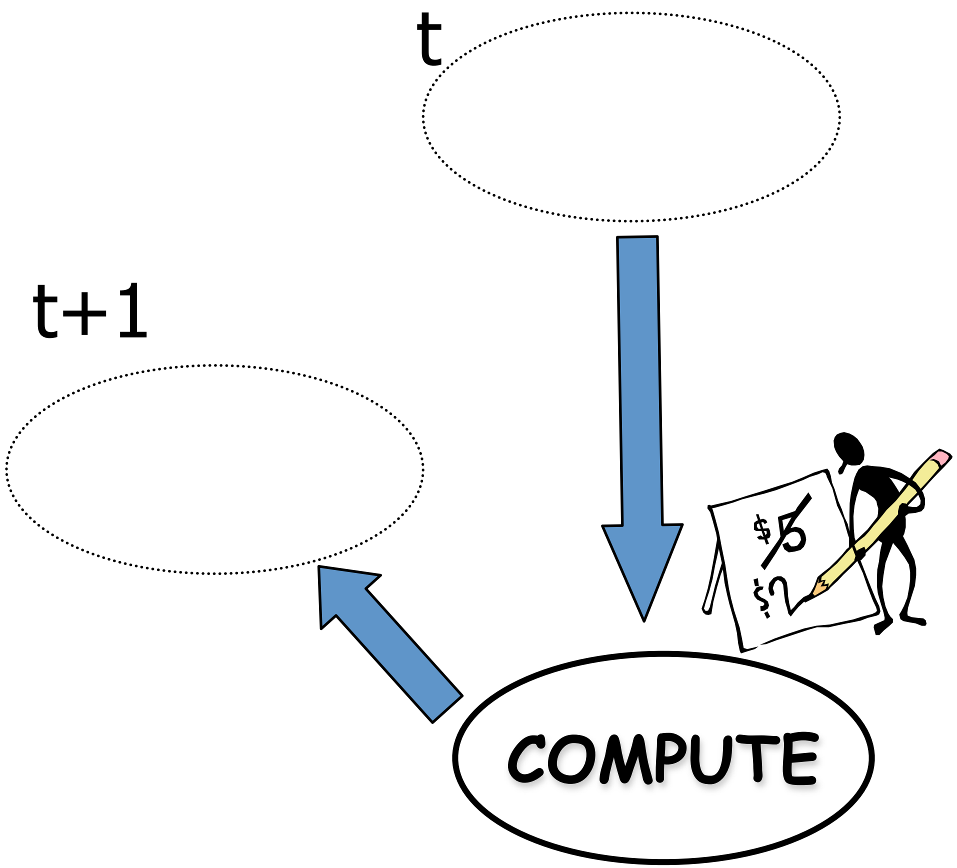
Uses its sensors
to observe the world.

result = **SNAPSHOT**
of the world

Visibility:
Unlimited
Limited

Instantaneous Actions of **SYm**

Suzuki *et al.*, 1996



Execute algorithm, ψ

input = positions of the robots

result = destination point p

Oblivious:

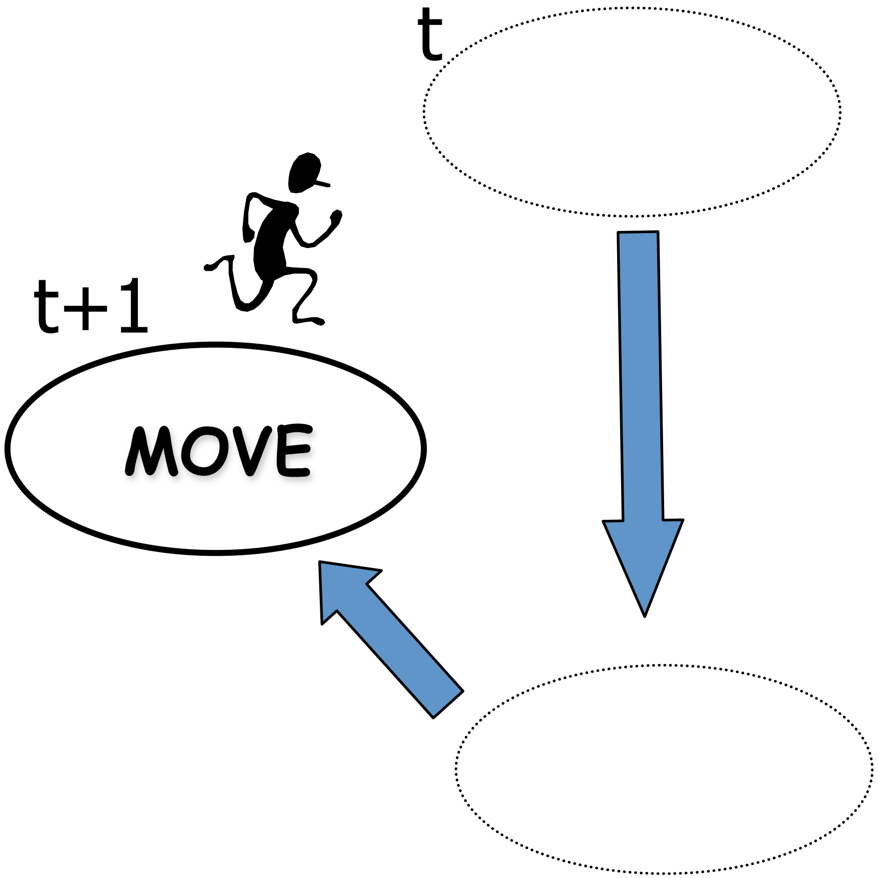
positions of the robots retrieved in the last Look

Non Oblivious:

positions of the robots since the beginning

Instantaneous Actions of **SYm**

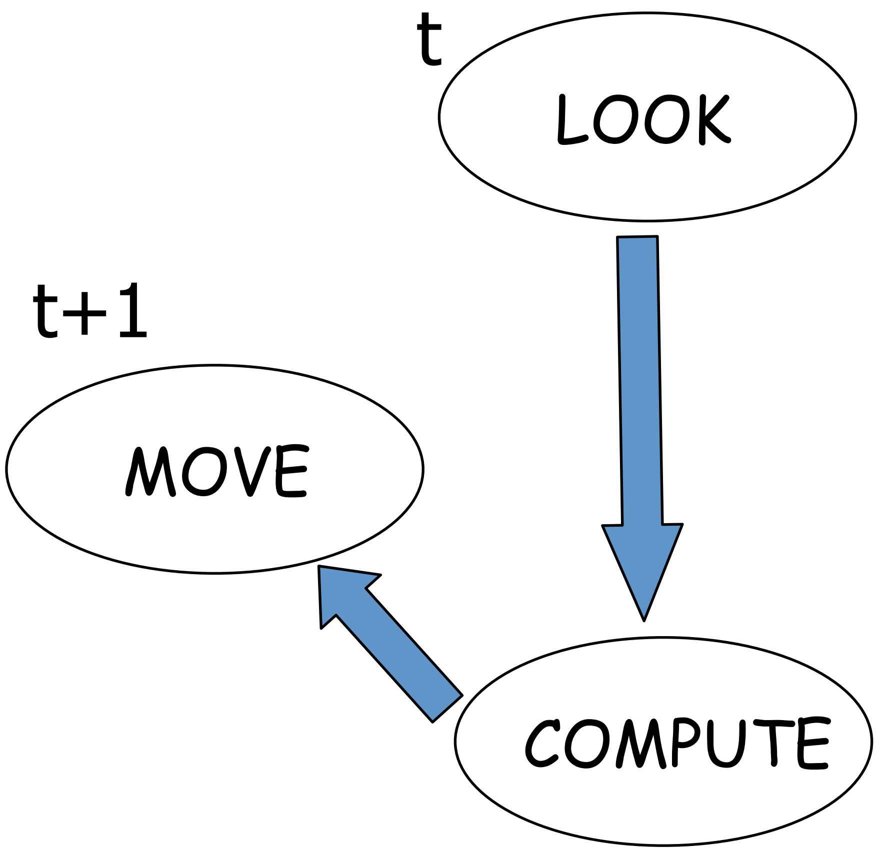
Suzuki *et al.*, 1996



The robot moves towards the computed destination

Instantaneous Actions of **SYm**

Suzuki *et al.*, 1996



$p_i(t)$: Position of r_i at t

p: point returned by ψ

For all $t \geq 0$,

r_i Inactive $\Rightarrow p_i(t+1)=p_i(t)$

r_i Active $\Rightarrow p_i(t+1)=p$

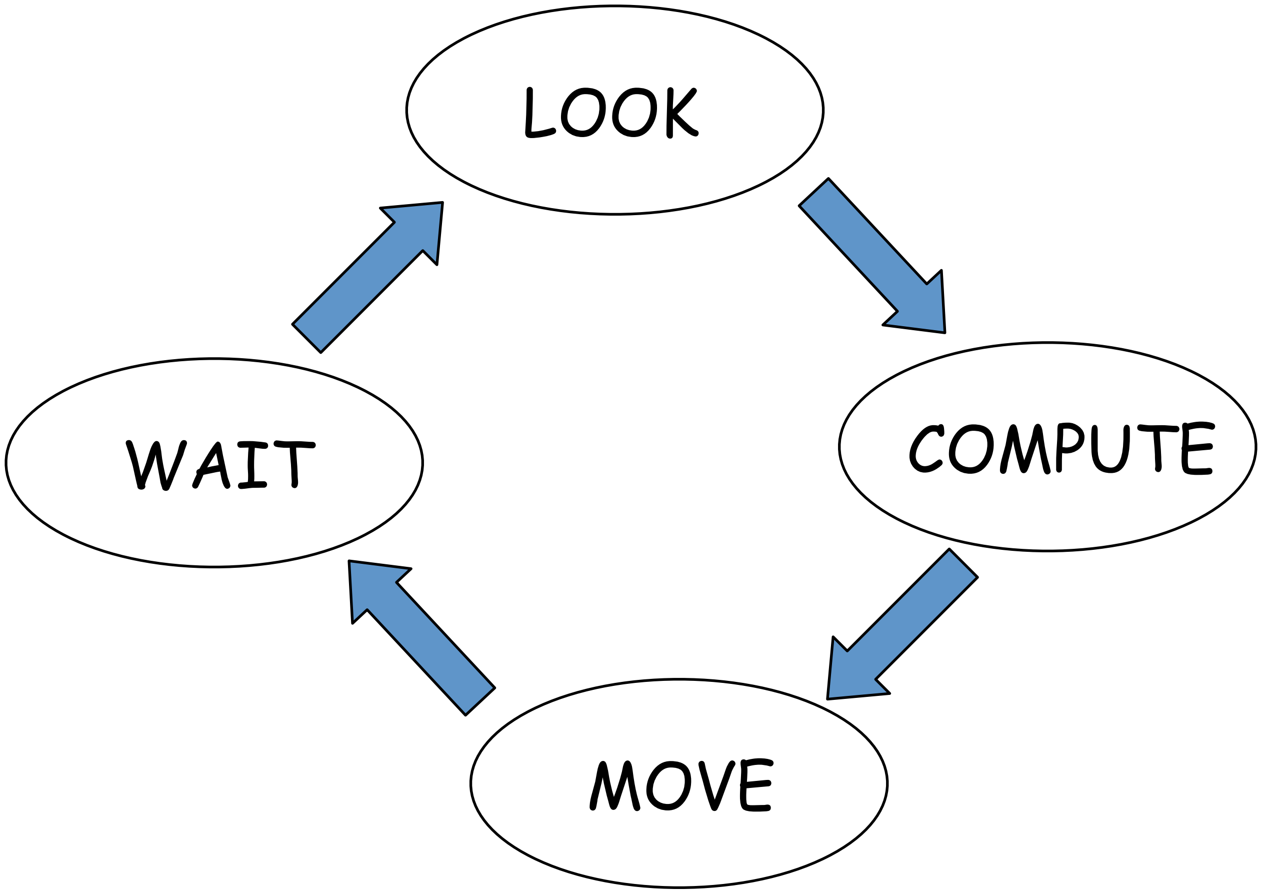
r_i executes L-C-M

atomically!

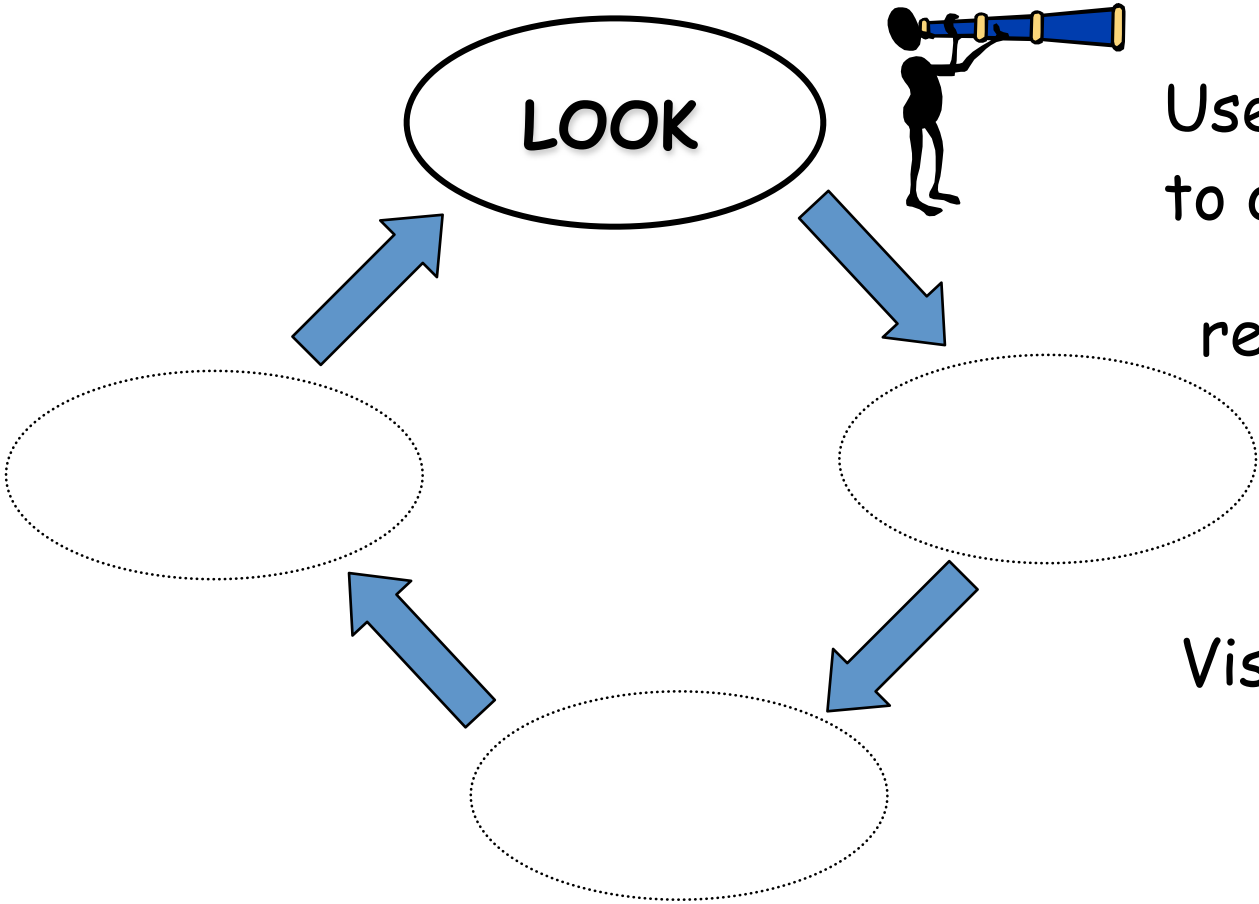
Asynchronicity

- In 1999 asynchronicity was introduced in the model
  - CORDA or ASYNC

The Life Cycle (Corda)



The Life Cycle (Corda)

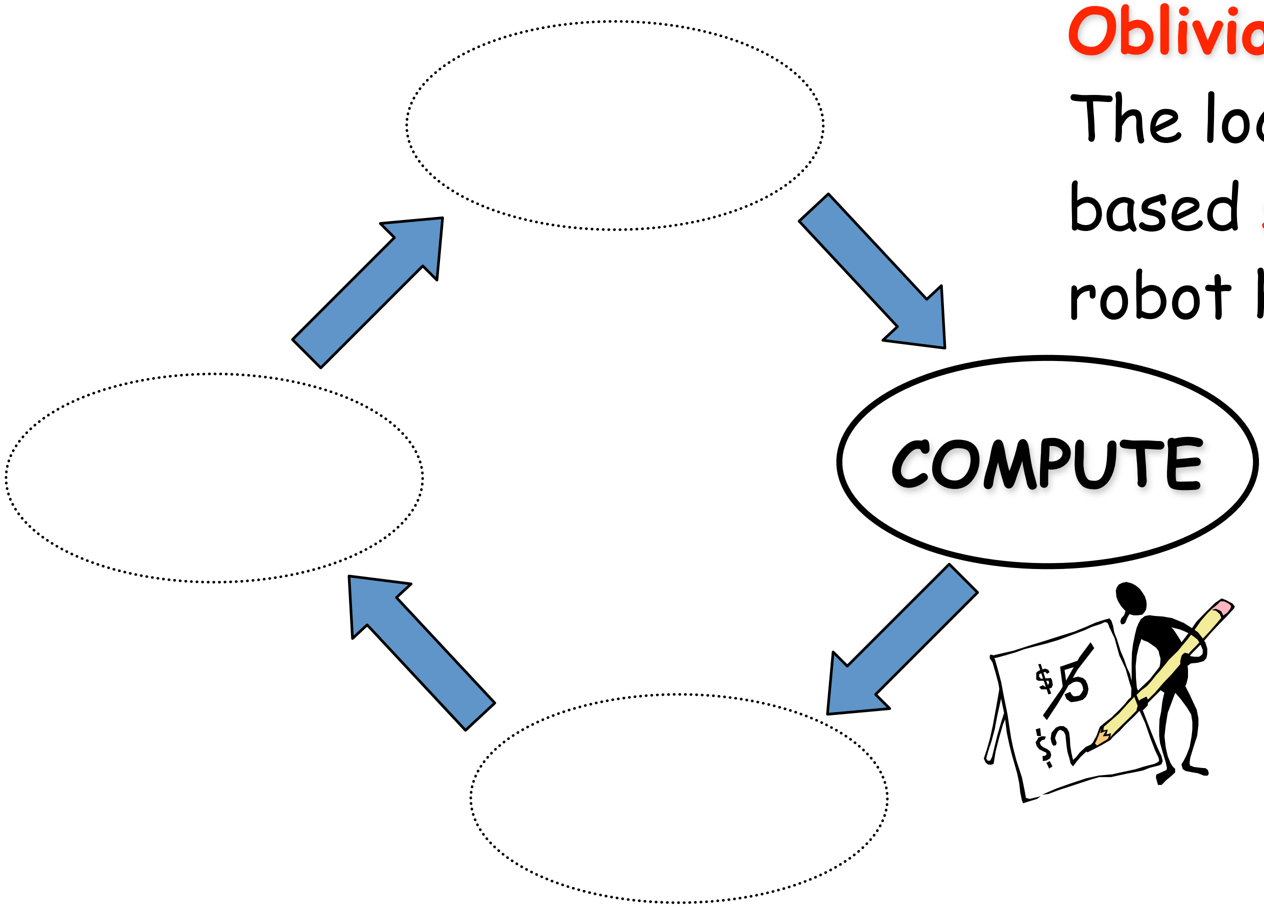


Uses its sensors
to observe the world.

result = **SNAPSHOT**
of the world

Visibility:
Unlimited
Limited

The Life Cycle (Corda)

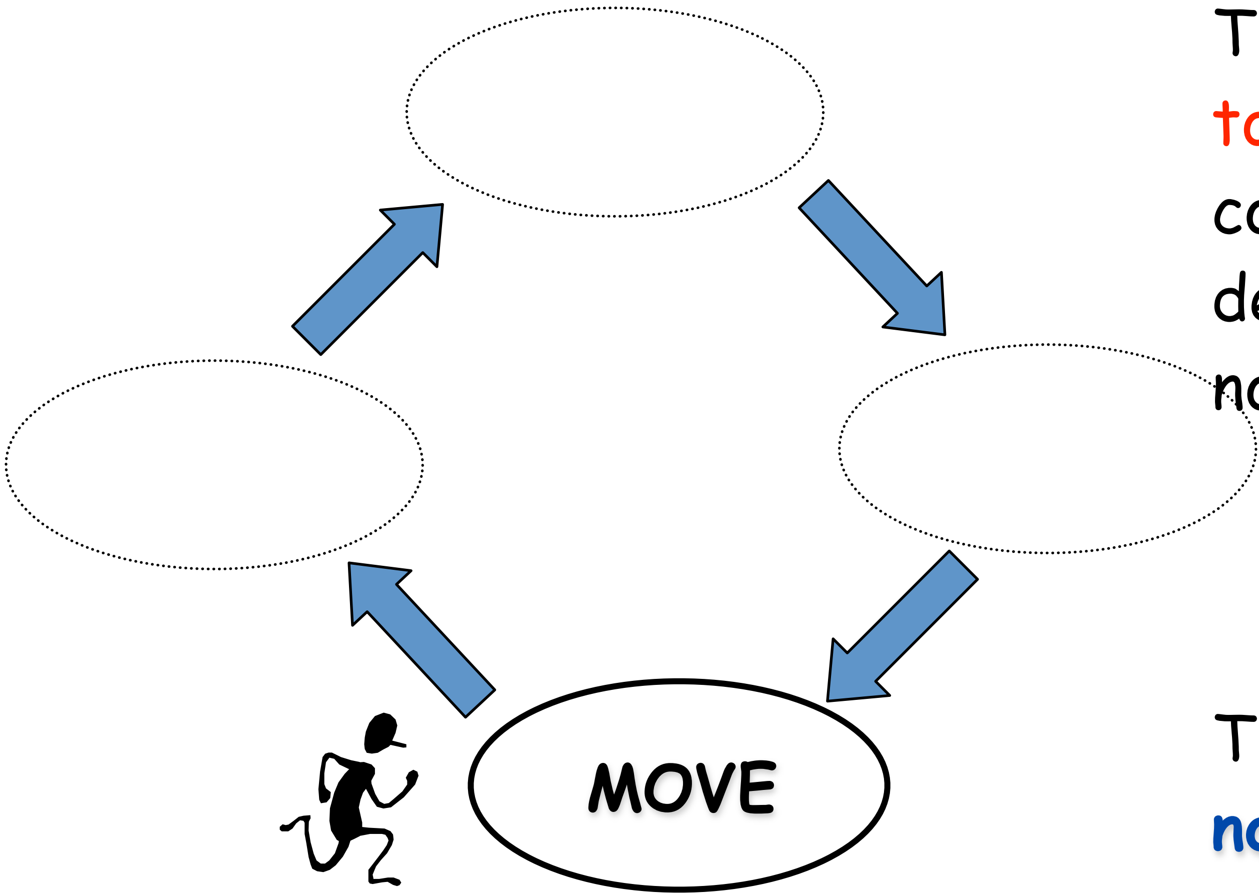


Obliviousness:
The local computation is based solely on what the robot has just observed

input = position of the other robots

result = destination point

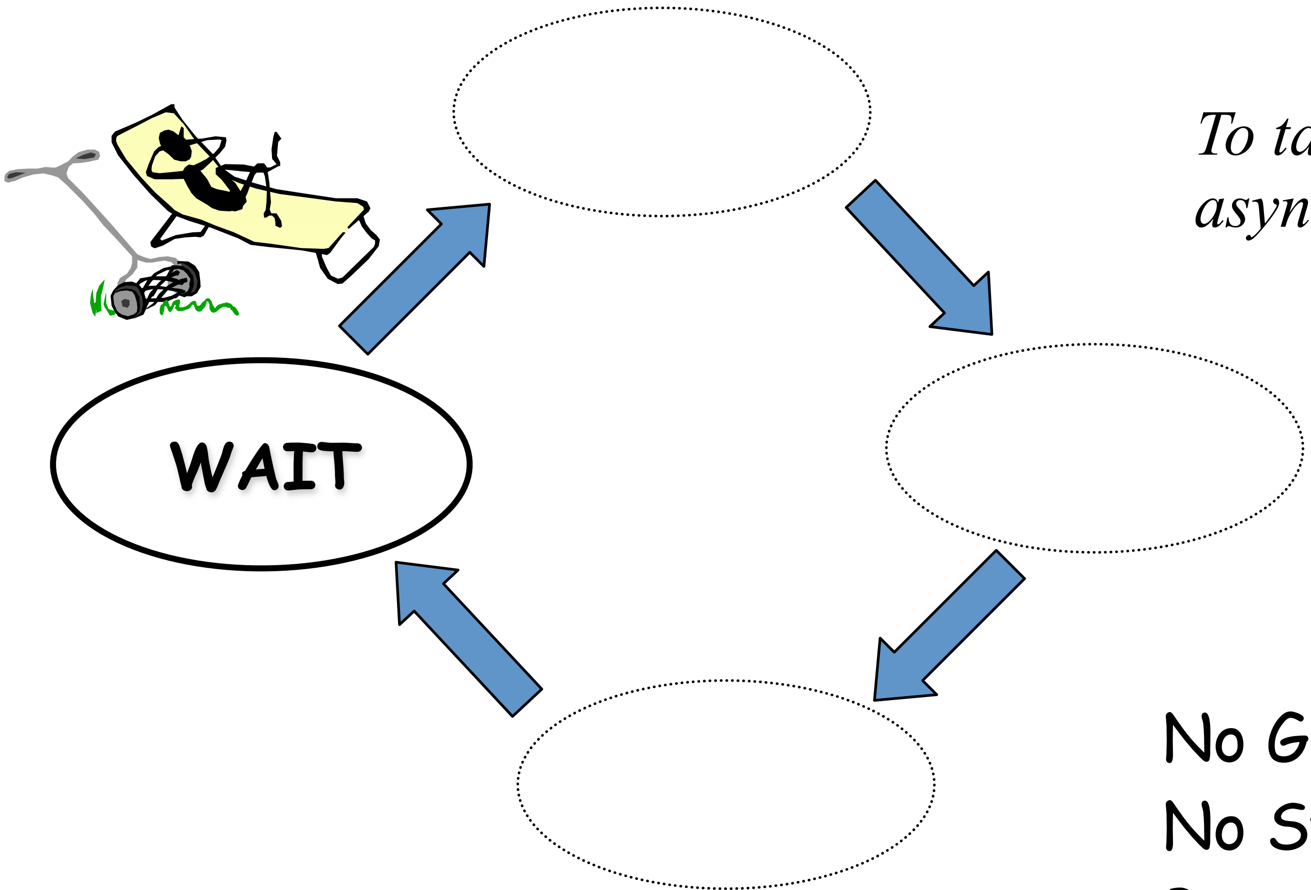
The Life Cycle (Corda)



The robot moves towards the computed destination (it might not reach it)

The movement is **non-instantaneous**.

The Life Cycle (Corda)



To take into account asynchronous behavior.

No Global Clock
No Synchronization…
But cannot wait forever

Sym vs CORDA

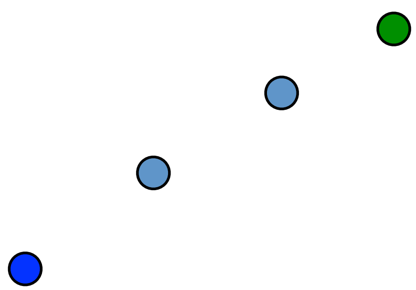
SYm

- **Instantaneous actions.**

CORDA

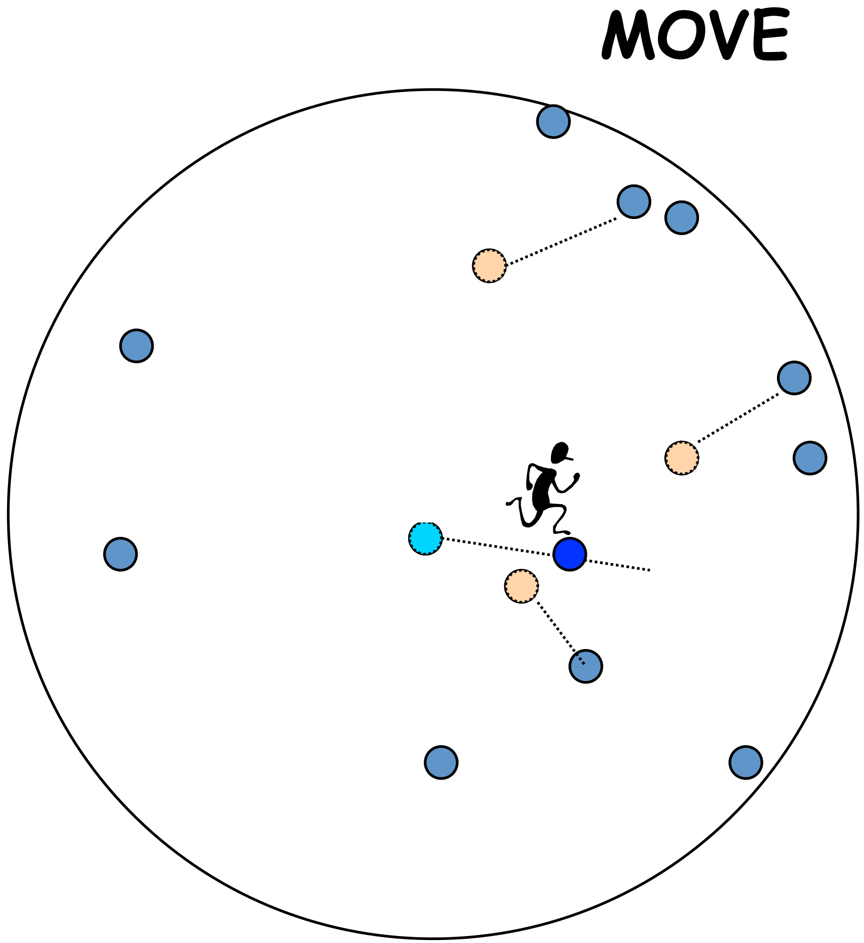
- **Full asynchronicity.**

Instantaneous Actions in SYm

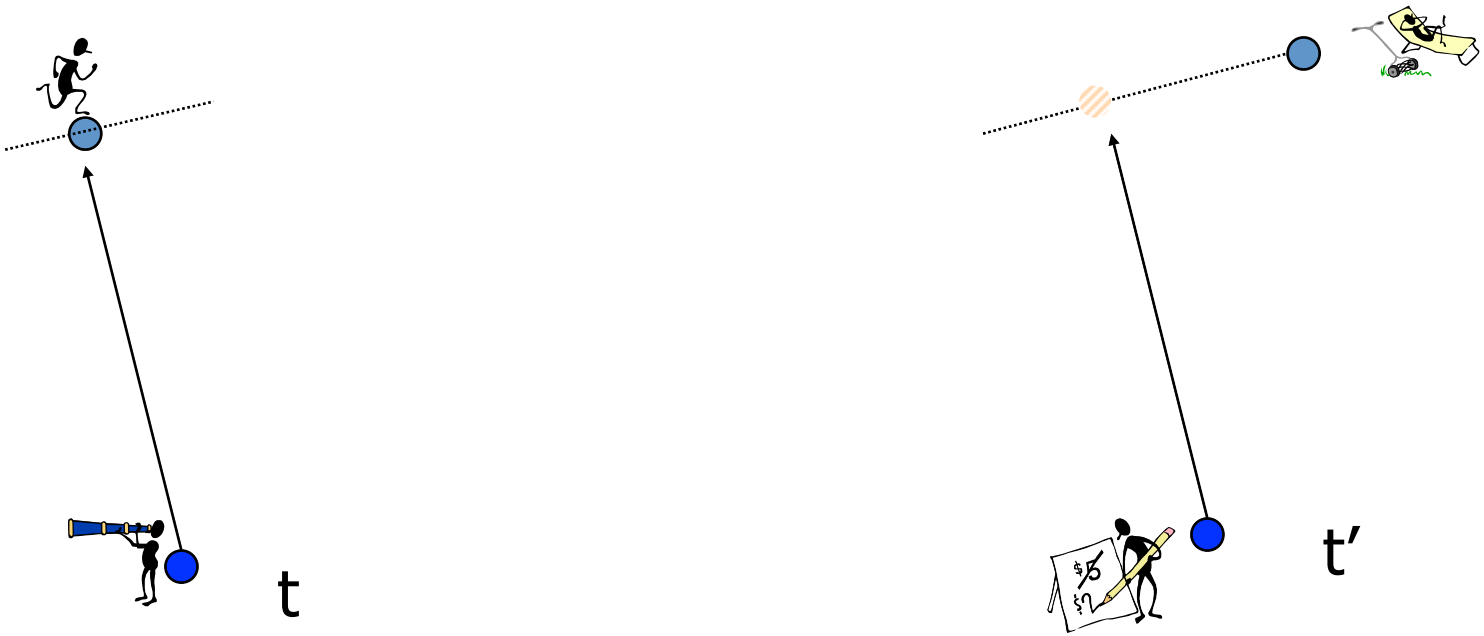


t+1

Asynchronicity in Corda



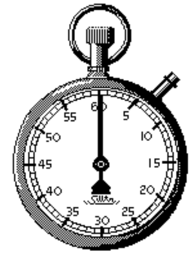
Asynchronicity in Corda



A robot could see other robots while they move !

A robot cannot distinguish between **moving** robots and **waiting** robots !

Timing Models



ASYNC (CORDA) *-Fully asynchronous*

[Flocchini et. Al, 1999]

Arbitrary & varying operation rates and delays

SSYNC *(Sym)* - *Semi-synchronous*

[Suzuki+Yamashita, 1996]

Fixed time cycles, but robots may be active / inactive

FSYNC - *Fully synchronous* [Suzuki+Yamashita, 1999]

Fixed time cycles, all robots active in every cycle

Corda vs. SYm

Problem $\mathcal{P}$ solvable in Corda

$\mathcal{P}$ solvable in SYm

Corda vs. SYm

Problem $\mathcal{P}$ unsolvable in Corda

$$\Uparrow$$

$\mathcal{P}$ unsolvable in SYm

Ring Exploration by Semi-Synchronous Oblivious Robots

Franck Petit

INRIA/LIP6-CNRS/UPMC

Context

- A team of *k* "*weak*" robots evolving into a ring of *n* nodes
  - Autonomous : No central authority
  - Anonymous : Undistinguishable
  - Oblivious : No mean to know the past
  - Disoriented : No mean to agree on a common direction or orientation

Context

- A team of k *"weak"* robots evolving into a ring of n nodes
  - Atomicity : In every configuration, each robot is located at exactly one node
  - Multiplicity : In every configuration, each node contains zero, one, or more than one robot

    (every robot is able to detect it)

Context

- A team of *k* "*weak*" robots evolving into a ring of *n* nodes
  - SSM : In every configuration, k' robots are activated ($0 < k' \leq k$)
  - The k' activated robots execute the cycle:
    1. Look : Instantaneous snapshot with Multiplicity detection
    2. Compute : Based on this observation, decides to either stay idle or move to one of the neighboring nodes
    3. Move : Move toward its destination

Problem

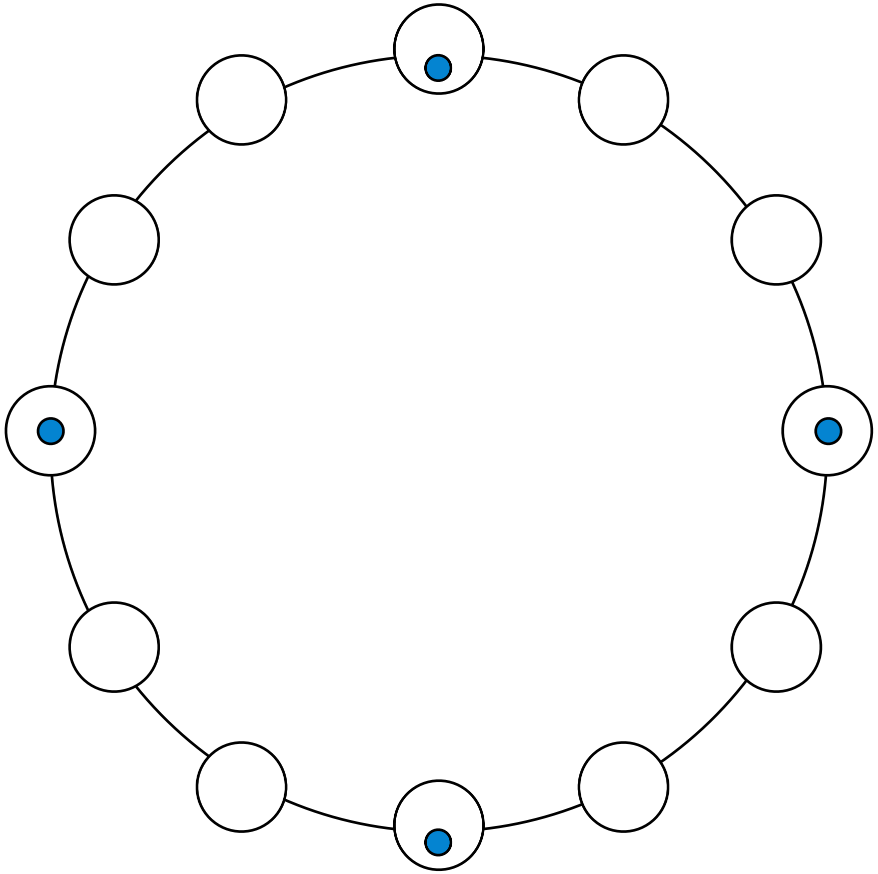
Starting from a configuration where no two robots are located at the same node:

- Exploration:
  Each node must be visited by at least one robot
- Termination:
  Eventually, every robot stays idle

- Performance: Number of robots
  ($k<n$)

Lower Bound (1/2)

Deterministic Exploration impossible if $k \mid n$



Lower Bound (2/2)

Theorem.

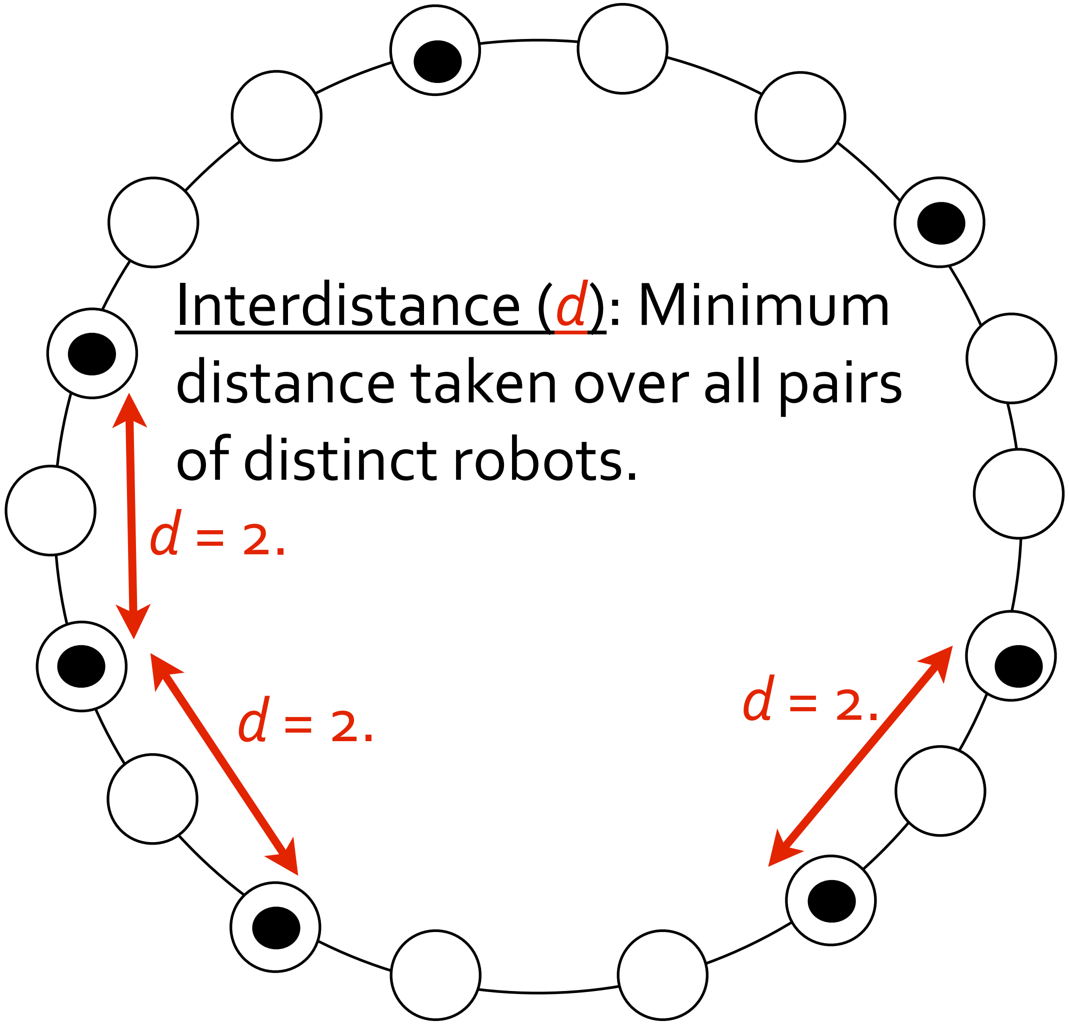
Let $m(n)$ be the minimal number of robots to deterministically explore a ring of size n.
There exists a constant c such that, for infinitely often many times, $m(n) \geq c \log n$.

- *Proof:*

1. Let n be the least common multiple of $1, 2, \ldots q$.
2. From the previous slide (1/2) : $m(n) \geq q + 1$.
3. From the Prime Number theorem: $\log n / q \rightarrow 1$.
4. This implies that there exists c s.t. $m(n) \geq c \log n$.

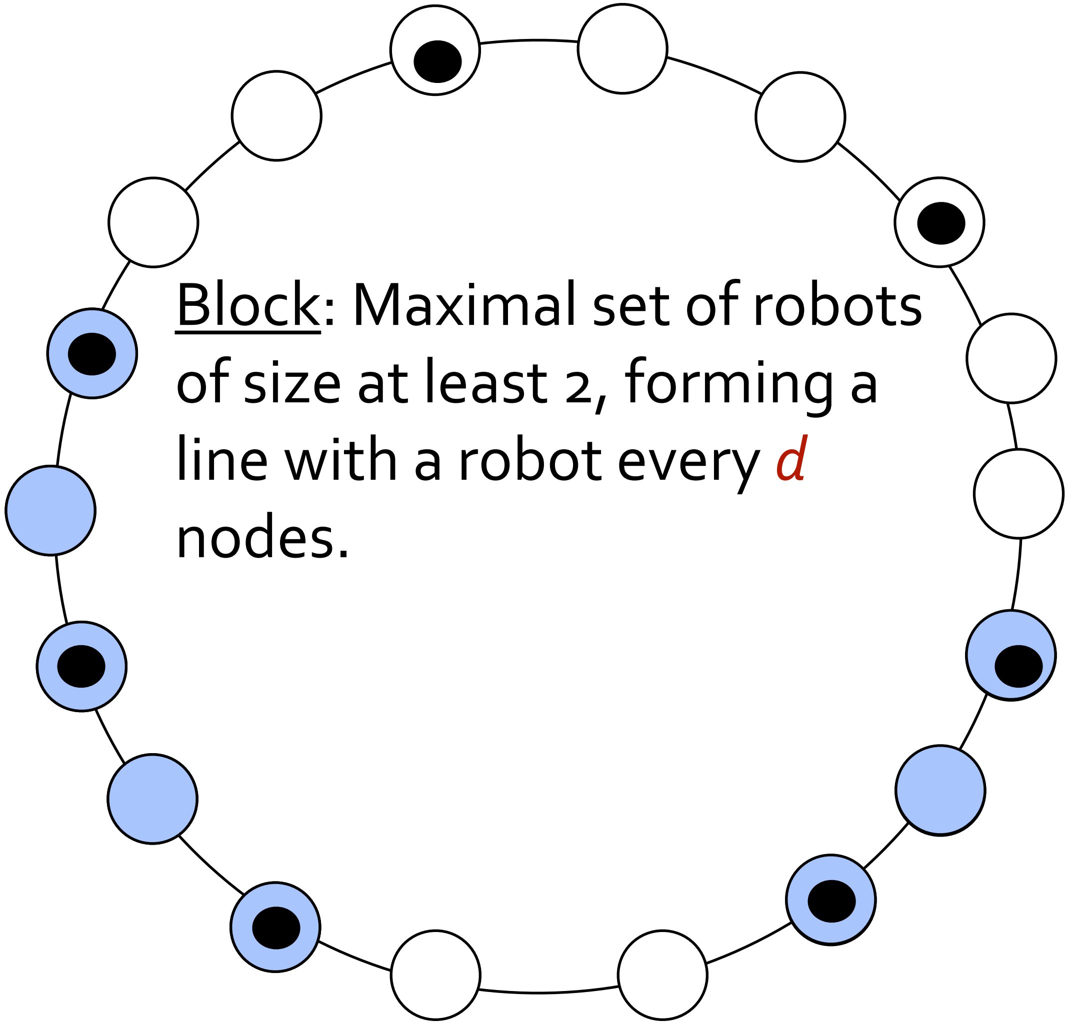
Deterministic Algorithm

- $k \mid n$, $k > \log n$, and $n > 16$



Deterministic Algorithm

- $k \mid n$, $k > \log n$, and $n > 16$

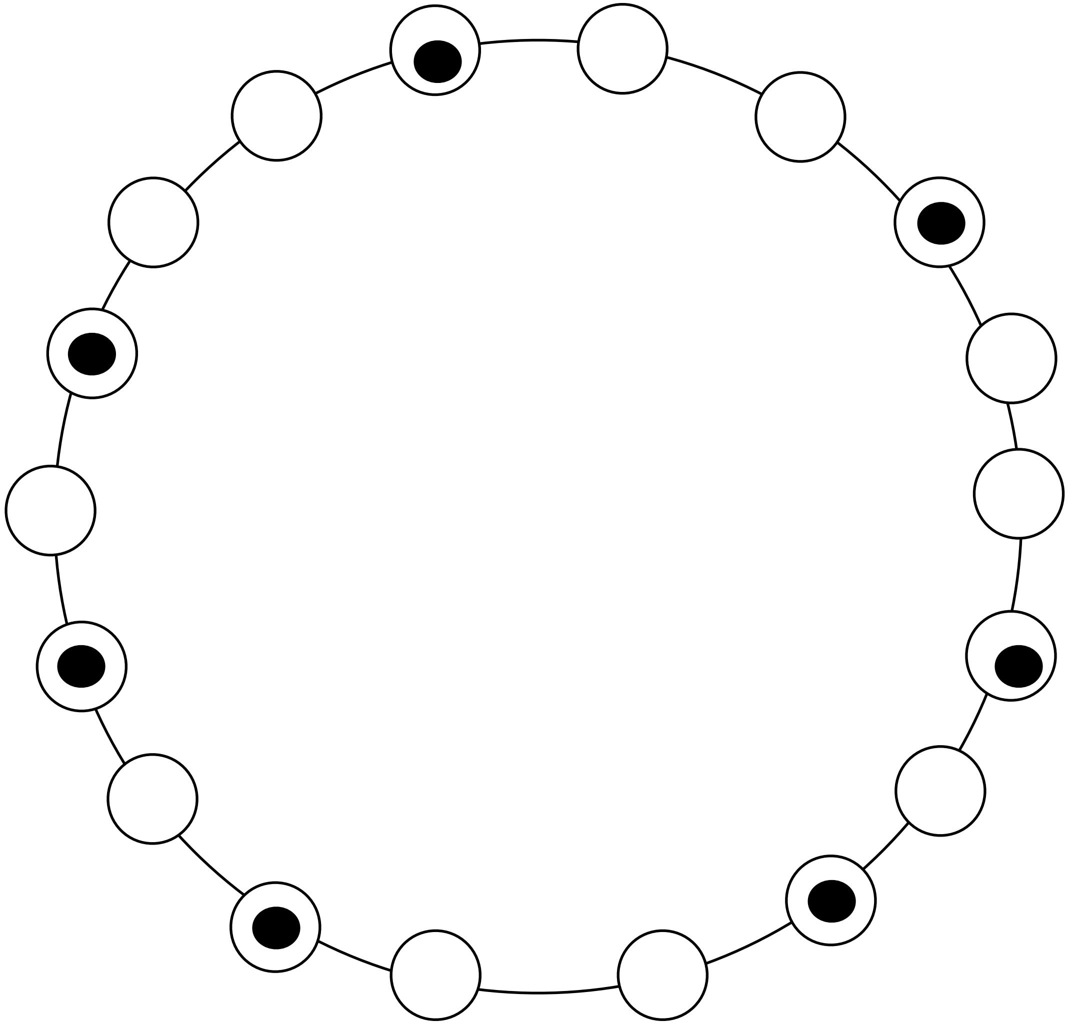


Deterministic Algorithm

- $k \mid n$, $k > \log n$, and $n > 16$
- Setup Phase:
  Goal: Transform the (arbitrary) initial configuration into a configuration of interdistance 1 where there is a single block or two blocks of the same size.
  Method: Decrease the number of blocks whenever possible. Otherwise, decrease the interdistance.
- Tower Phase:
  Goal: Create one or two multiplicities inside each block.
- Exploration Phase:
  Goal: Perform exploration until reaching an identified final configuration.

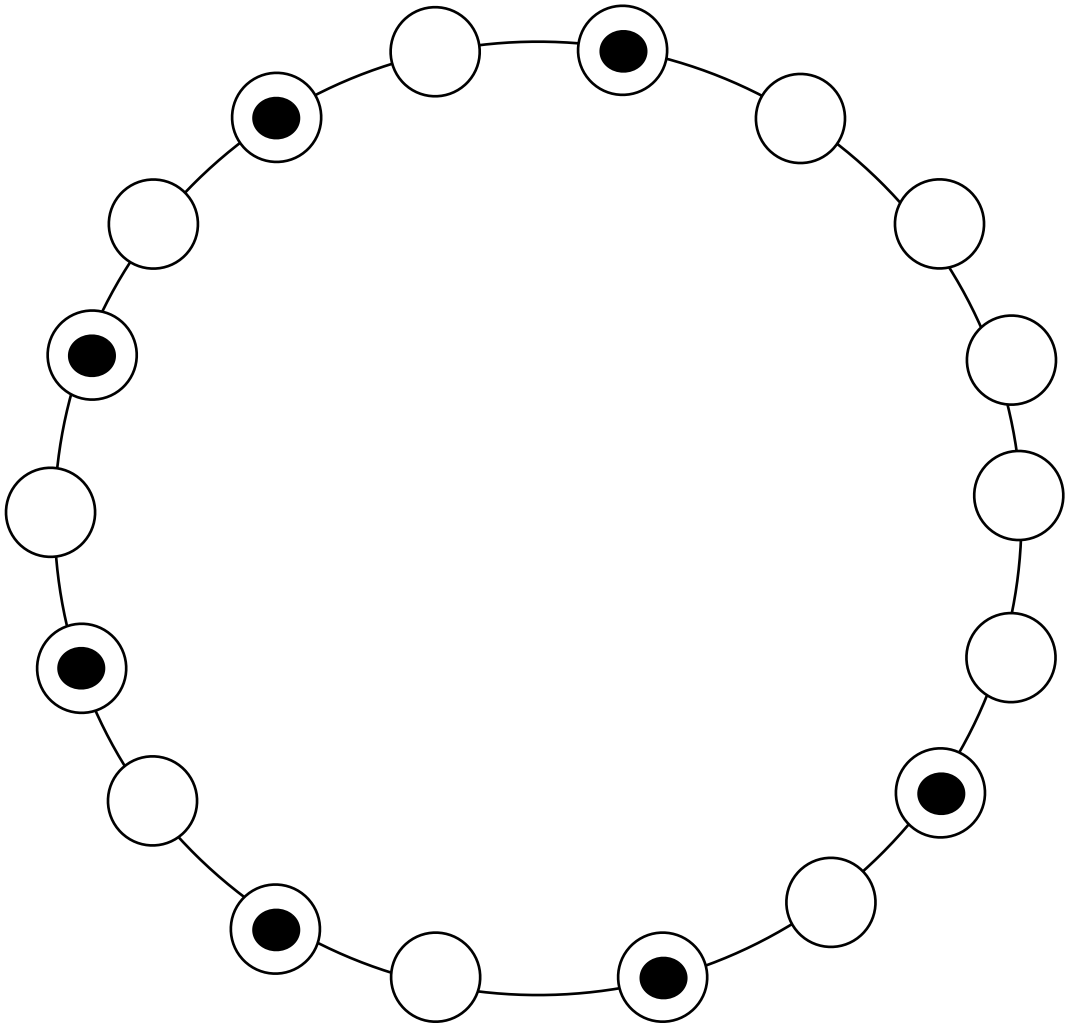
Deterministic Algorithm

- $k \mid n$, $k > \log n$, and $n > 16$



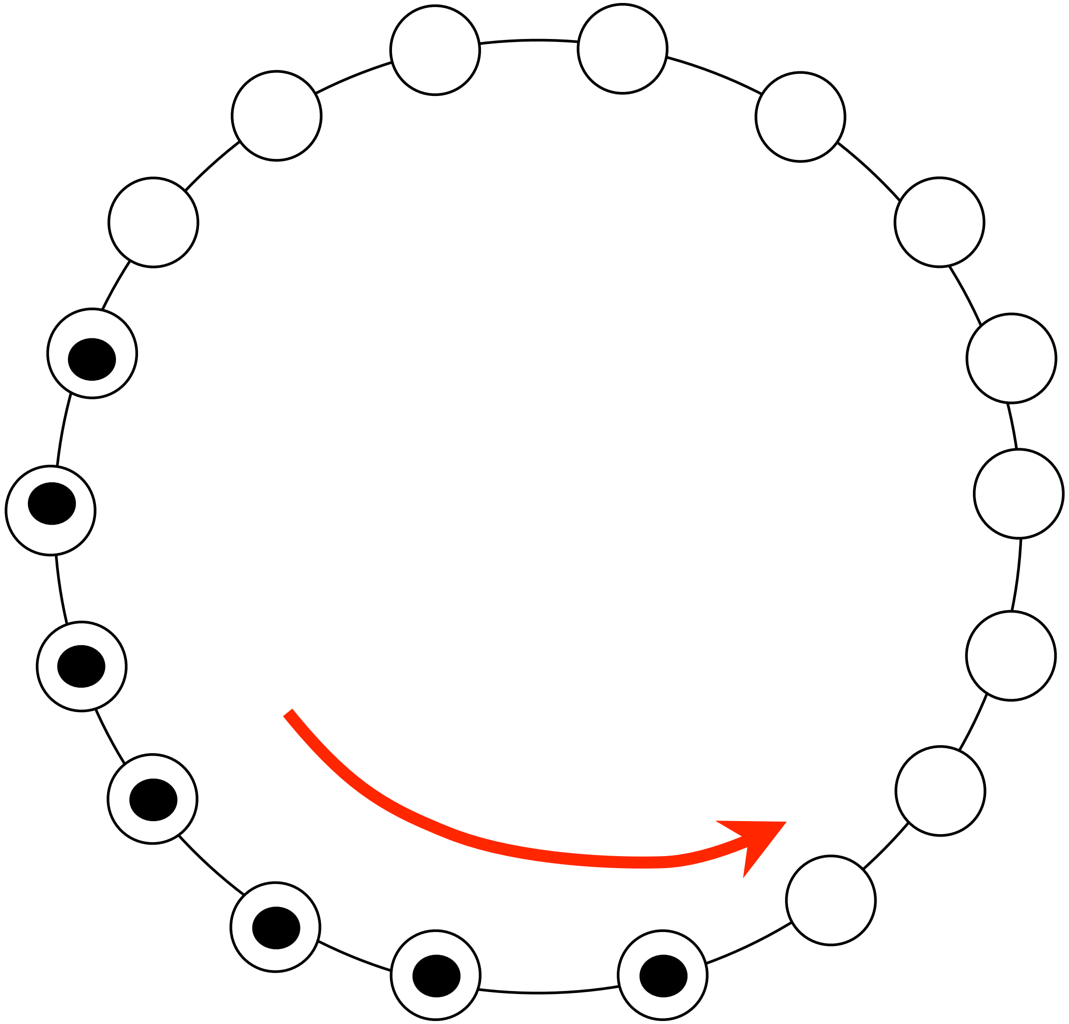
Deterministic Algorithm

- $k \mid n$, $k > \log n$, and $n > 16$



Deterministic Algorithm

- $k \mid n$, $k > \log n$, and $n > 16$



Optimality

- *$\Theta(\log n)$* robots are necessary and sufficient, provided that *k* does not divide *n*.
  A *deterministic* algorithm for $k \geq 17$

- Minimal Number of Robots?

- Probabilistic?

Optimality

Theorem.
4 probabilistic robots are necessary and sufficient, provided that $n > 8$

- n and k are not required to be coprime

1. Exploration impossible with less than *4* robots
2. An algorithm working with *4* probabilistic robots ($n > 8$)

Oblivious Robots

Termination → Implicit memory ← Exploration

Implicit memory

↓

At least one configuration that cannot be an initial configuration

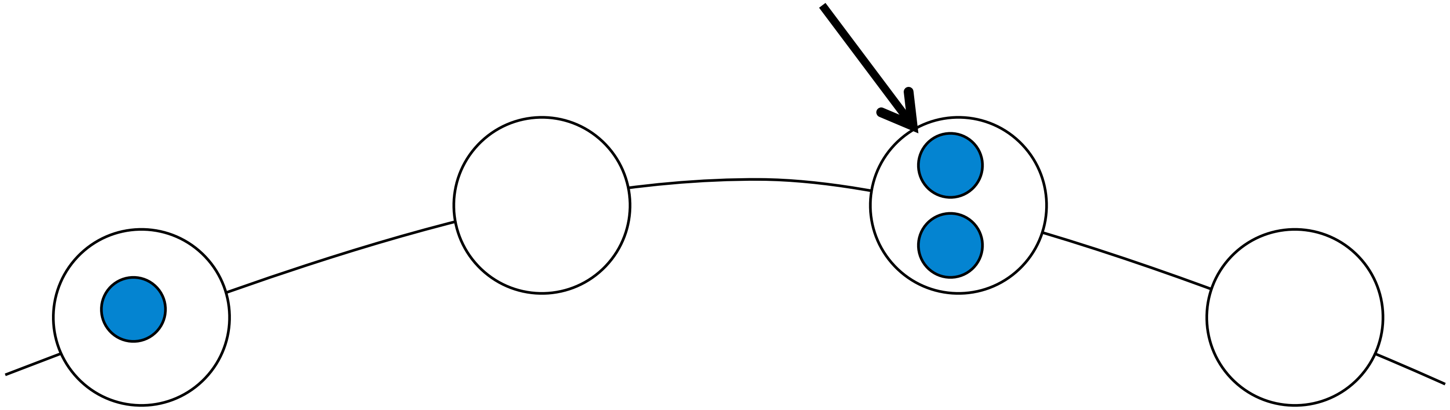
Remark.

If $n > k$, any terminal configuration of any protocol contains at least one *tower*.

Tower

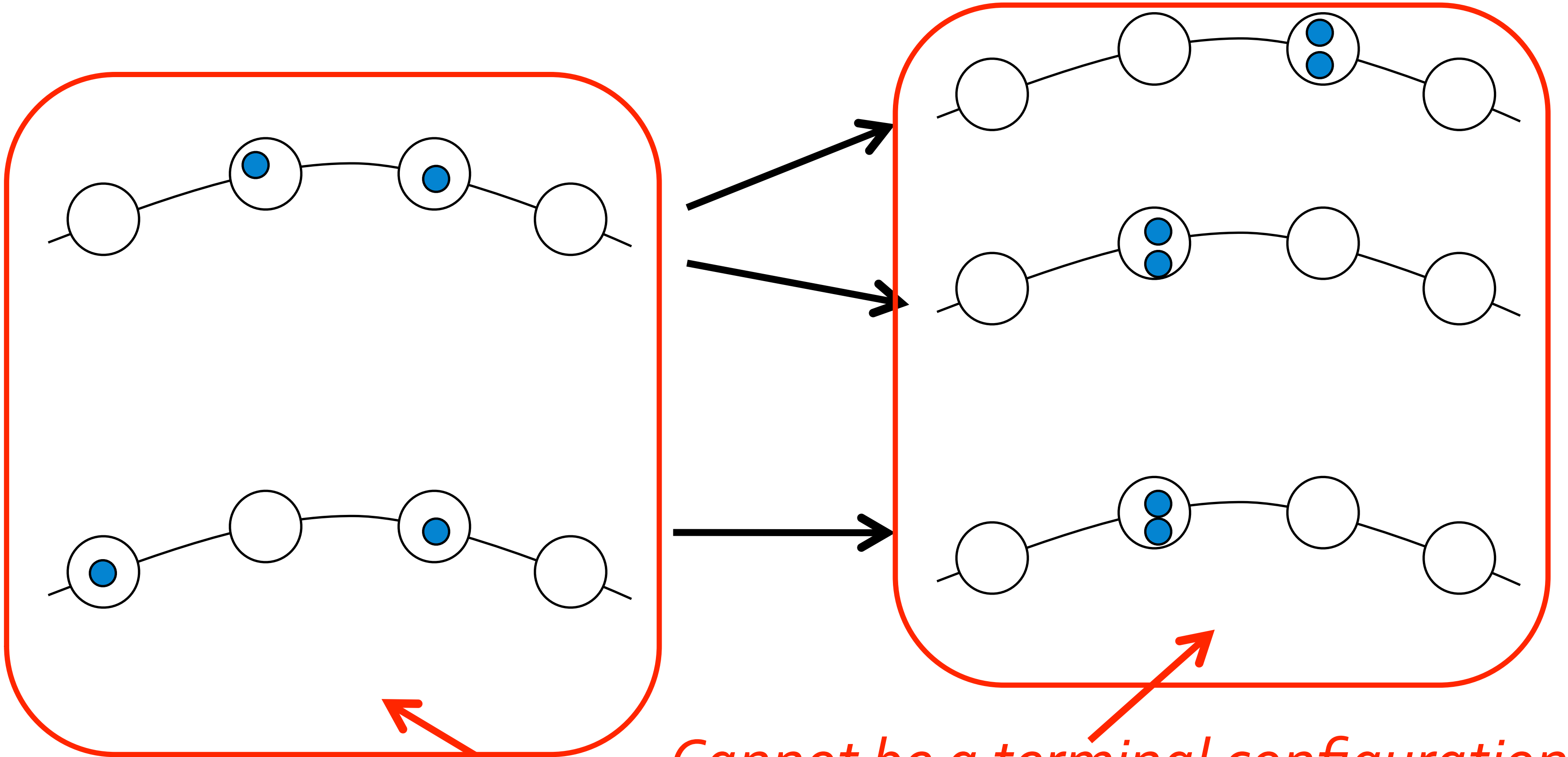
<u>*Definition*</u>.

A node with at least two robots.



$k \geq 2$

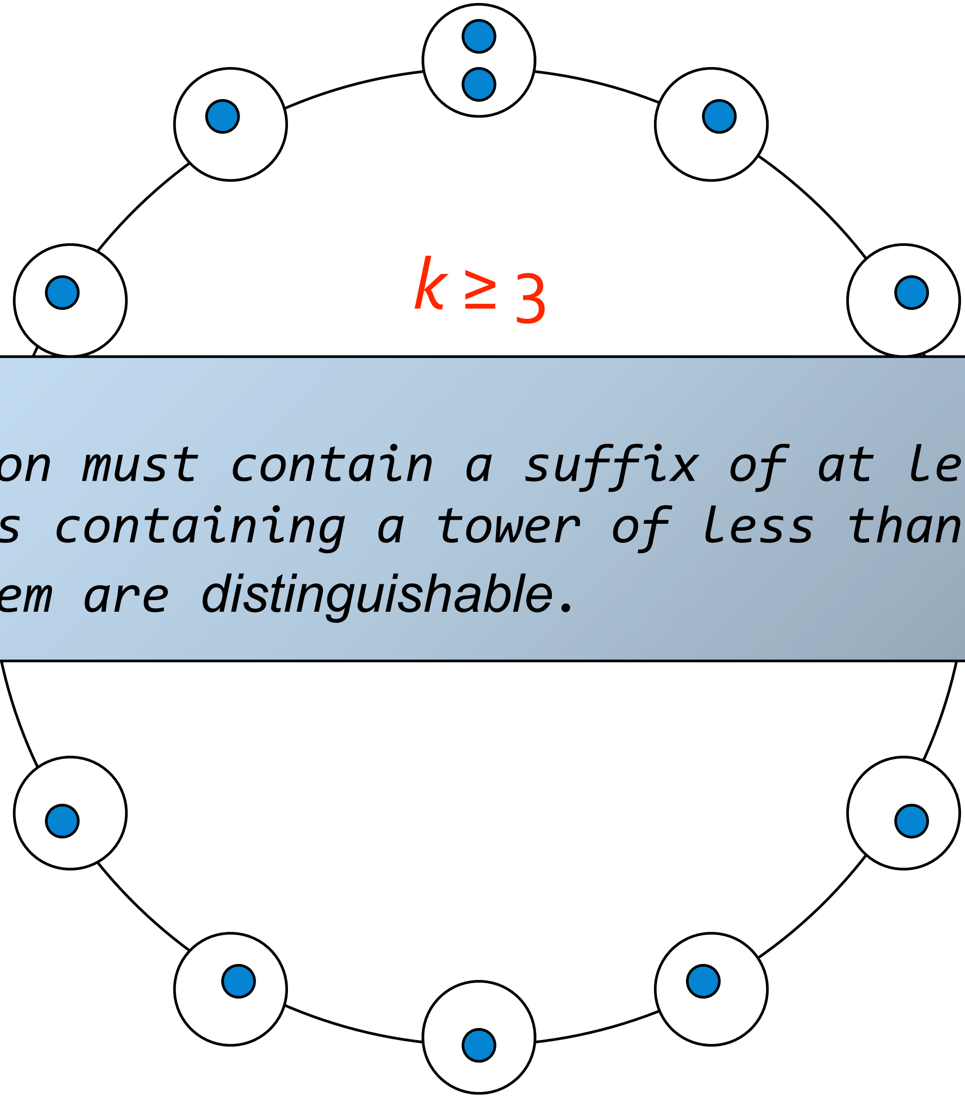
Tower Building



Cannot be a terminal configuration

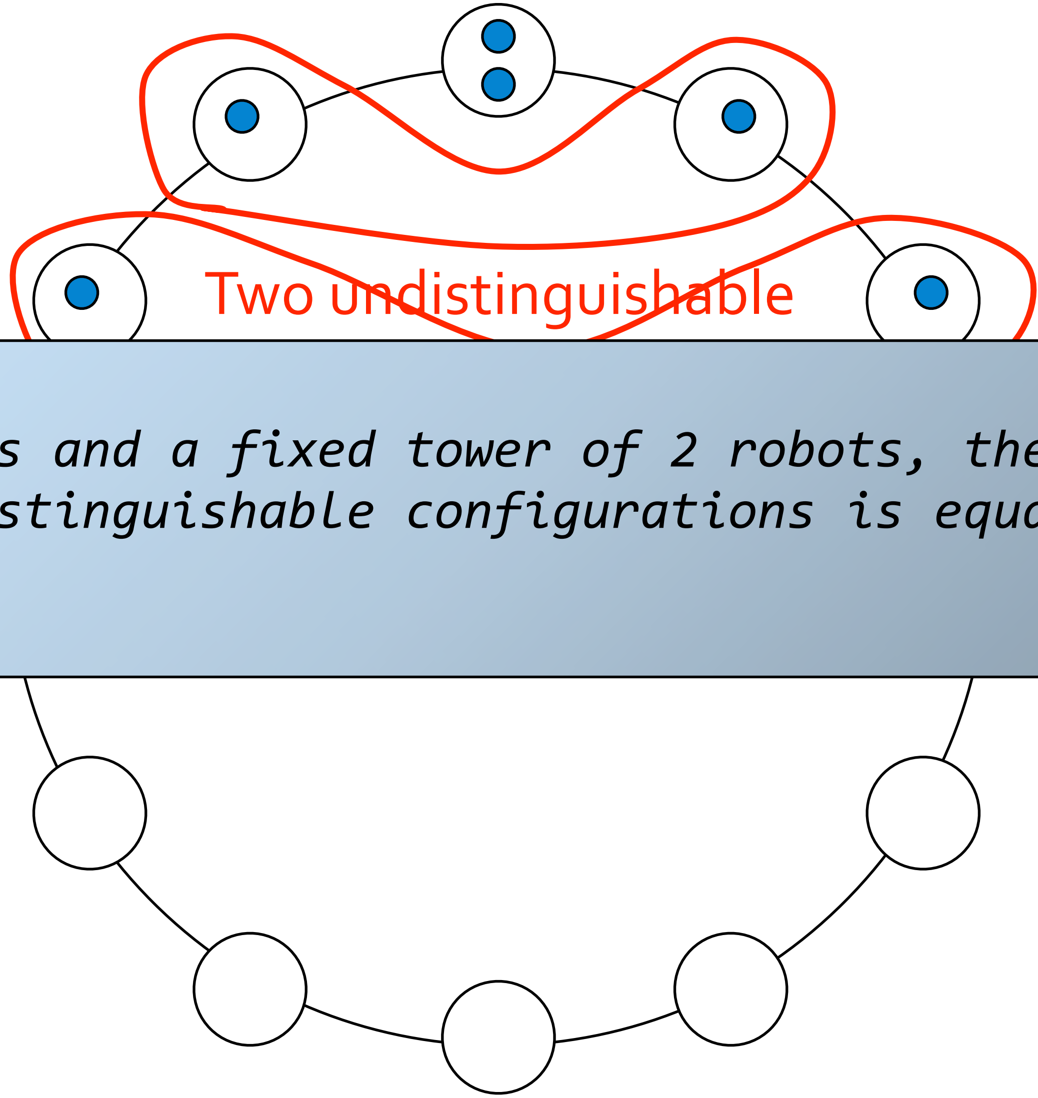
Can be an initial configuration

Enabling Exploration



<u>Lemma</u>.
Every execution must contain a suffix of at least $n-k+1$ configurations containing a tower of less than k robots and any two of them are distinguishable.

Enabling Exploration



***Lemma*.**
With 3 robots and a fixed tower of 2 robots, the maximum number of distinguishable configurations is equal to $\left\lfloor \frac{n}{2} \right\rfloor$.

Enabling Exploration

***Lemma*.**
For every $n > 4$, there exists no exploration protocol (even probabilistic) of a n-size ring with 3 robots.

Proof :

$$\left\lfloor \frac{n}{2} \right\rfloor \geq n - k + 1 \Rightarrow n \leq 4$$

Negative result

Theorem.
For every $n \geq 4$, there exists no exploration protocol (even probabilistic) of a n-size ring with three robots.

Proof :

There exists no protocol with 3 robots in a 4-size ring with a distributed scheduler.

Contribution

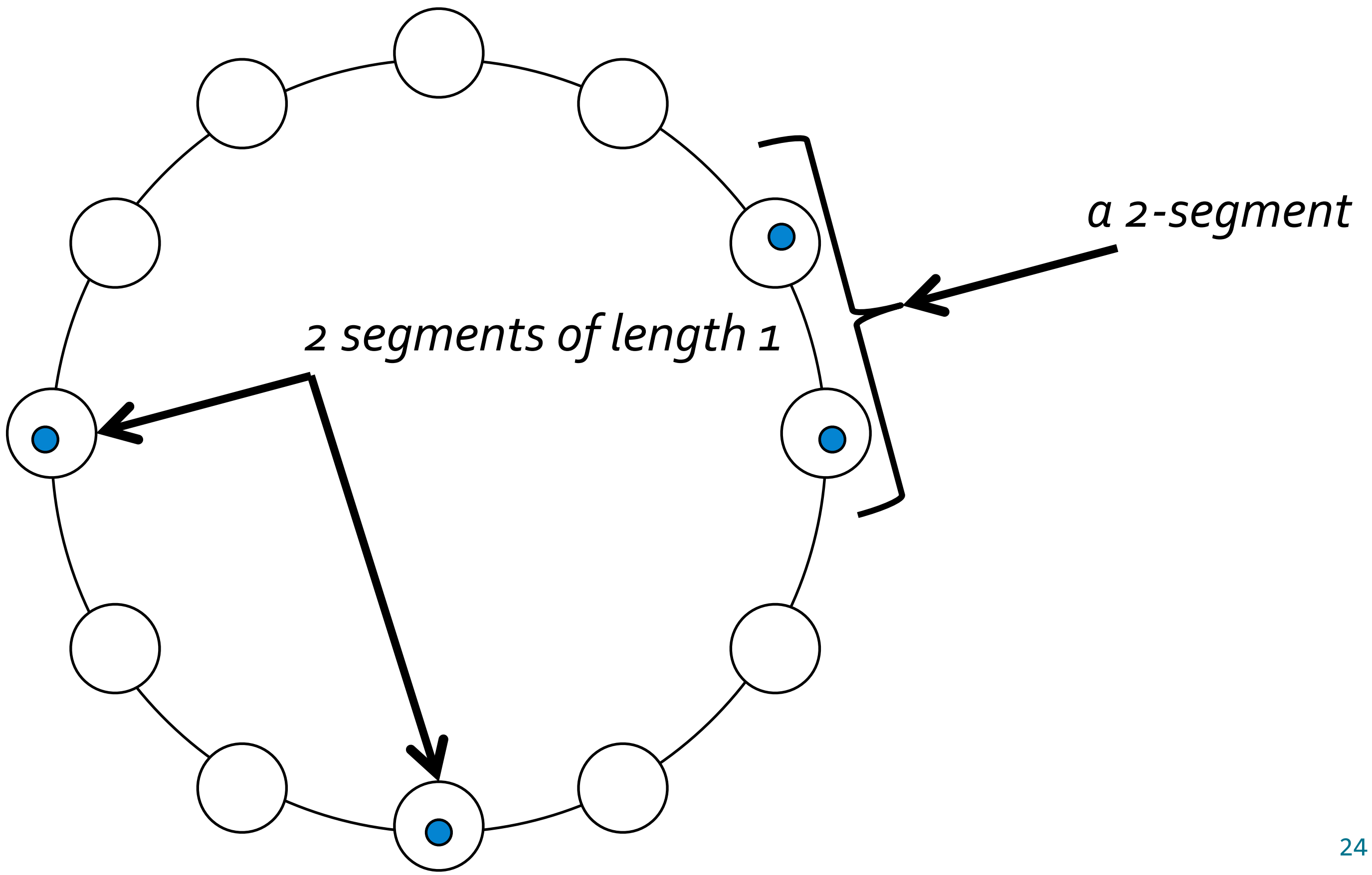
Theorem.
4 probabilistic robots are necessary and sufficient, provided that $n > 8$

- n and k are not required to be coprime

1. Exploration impossible with less than *4* robots
2. Give an algorithm working with *4* probabilistic robots ($n > 8$)

Definitions

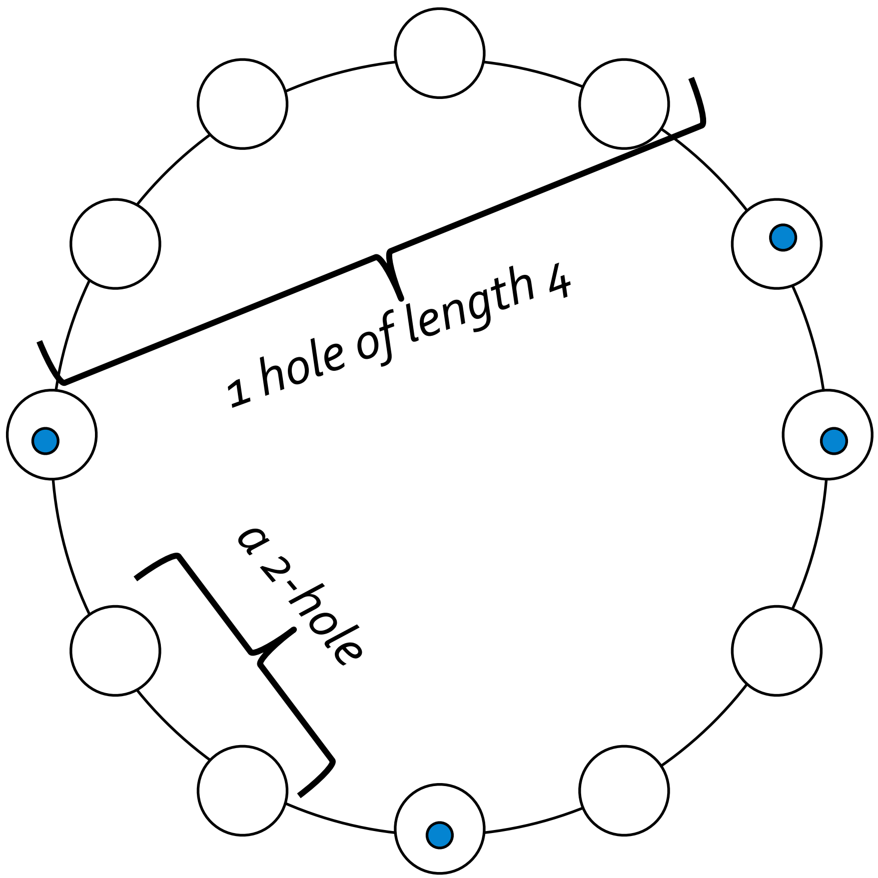
<u>*Segment*</u>.
A maximal non-empty elementary path of occupied nodes.



Definitions

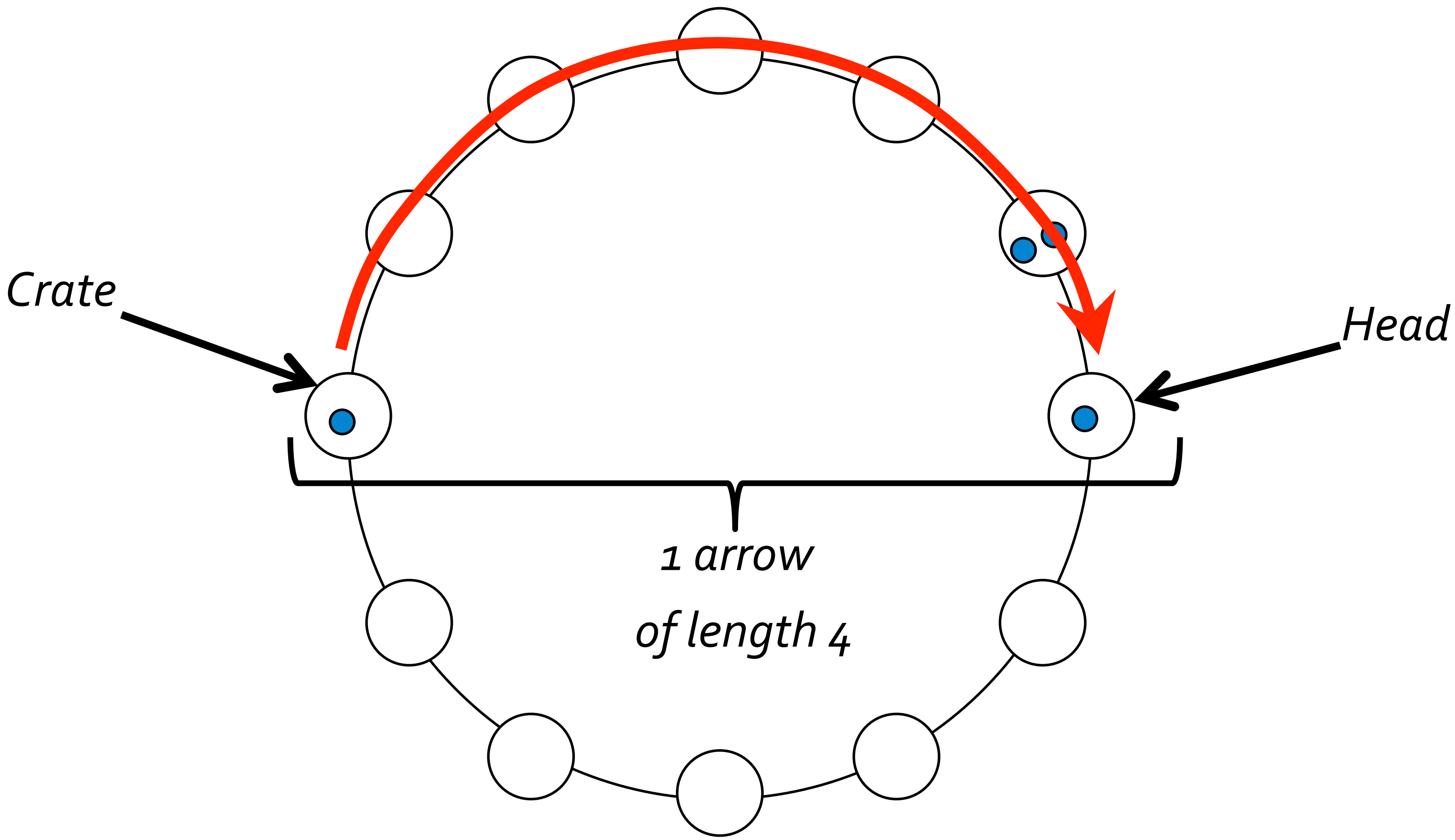
Hole.

A maximal non-empty elementary path of free nodes.



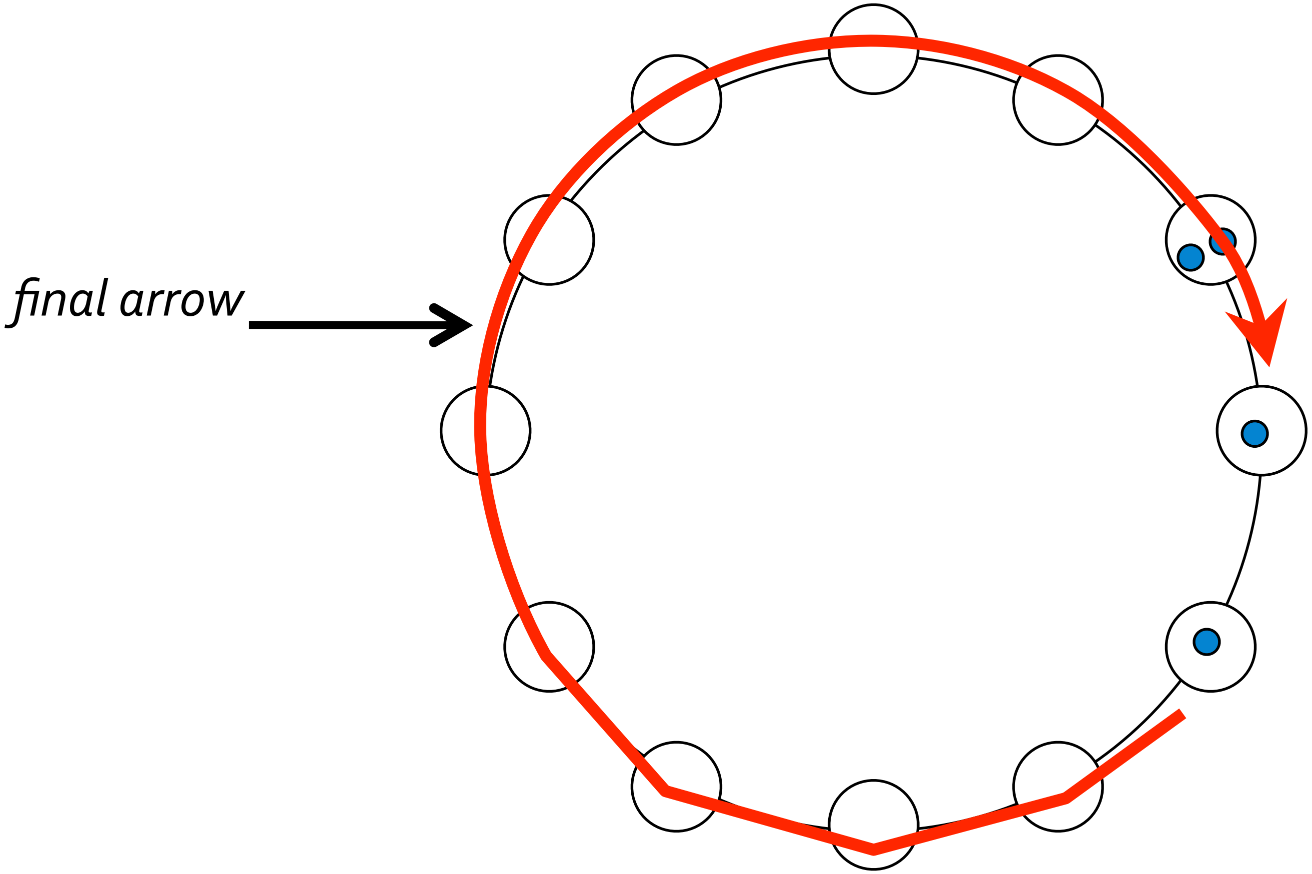
Definitions

Arrow. A 1-segment, followed by a non-empty elementary path of free nodes, a tower, and a 1-segment.



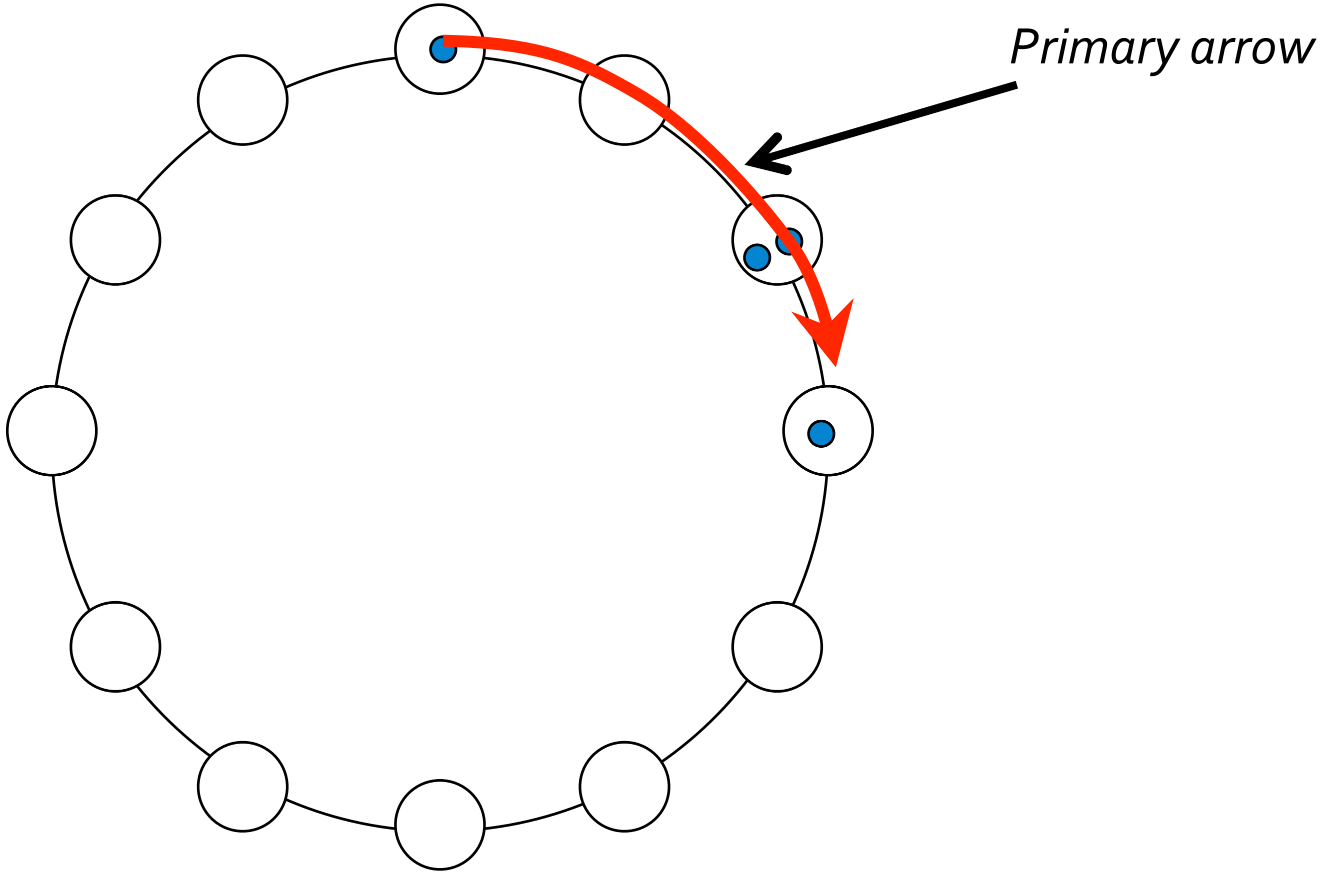
Definitions

<u>Arrow</u>. A 1-segment, followed by a non-empty elementary path of free nodes, a tower, and a 1-segment.



Definitions

<u>Arrow</u>. A 1-segment, followed by a non-empty elementary path of free nodes, a tower, and a 1-segment.

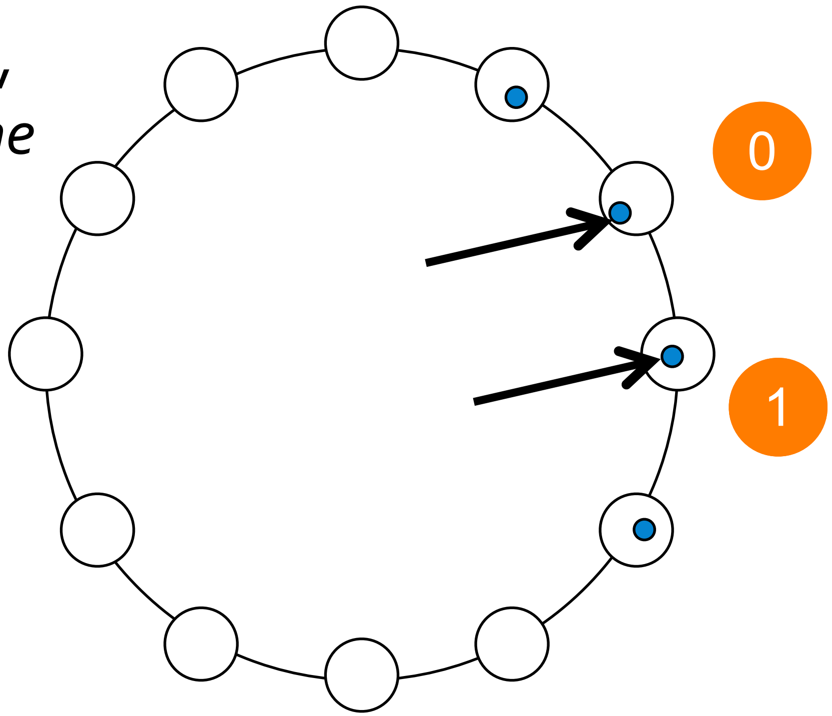


Algorithm

- *Initially, there is no tower*

1. Converge toward a 4-segment
2. Build a tower
3. Visit the ring and terminate

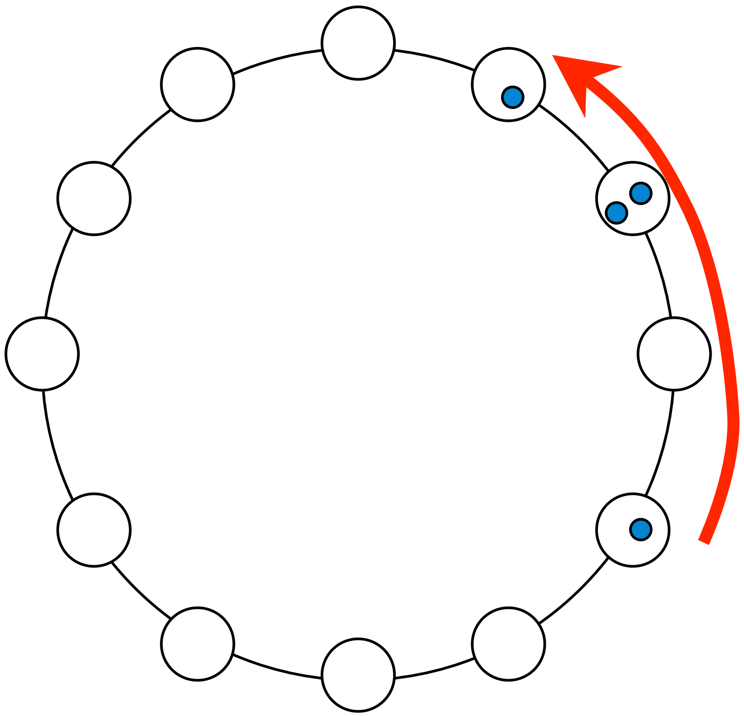
If I am an internal node, then I try to move on the other internal node.



Algorithm

- *Initially, there is no tower*

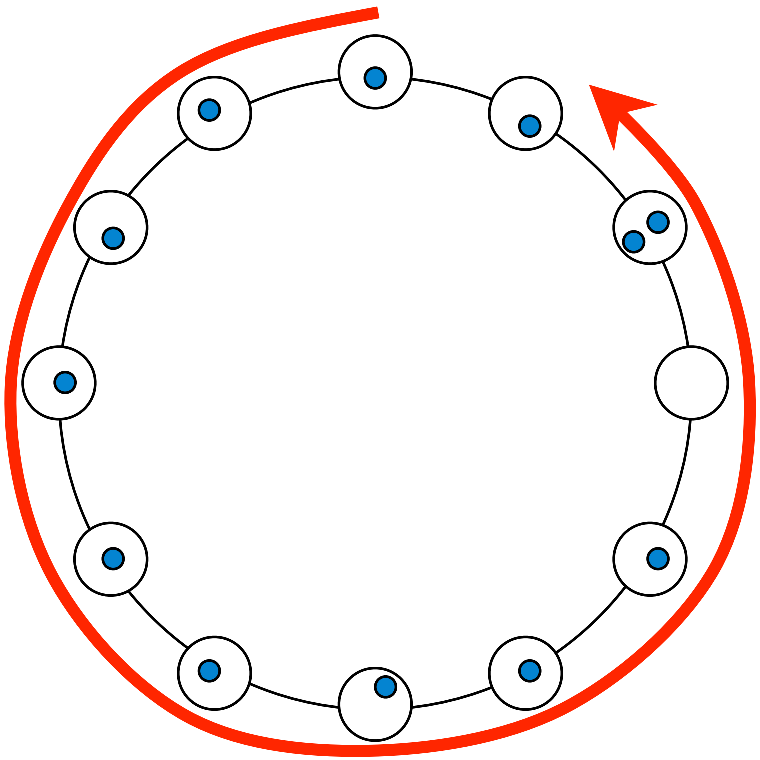
1. Converge toward a 4-segment
2. Build a tower ➔ Primary arrow
3. Visit the ring and terminate



Algorithm

- *Initially, there is no tower*

1. Converge toward a 4-segment
2. Build a tower ➔ Primary arrow
3. Visit the ring and terminate ➔ Final arrow

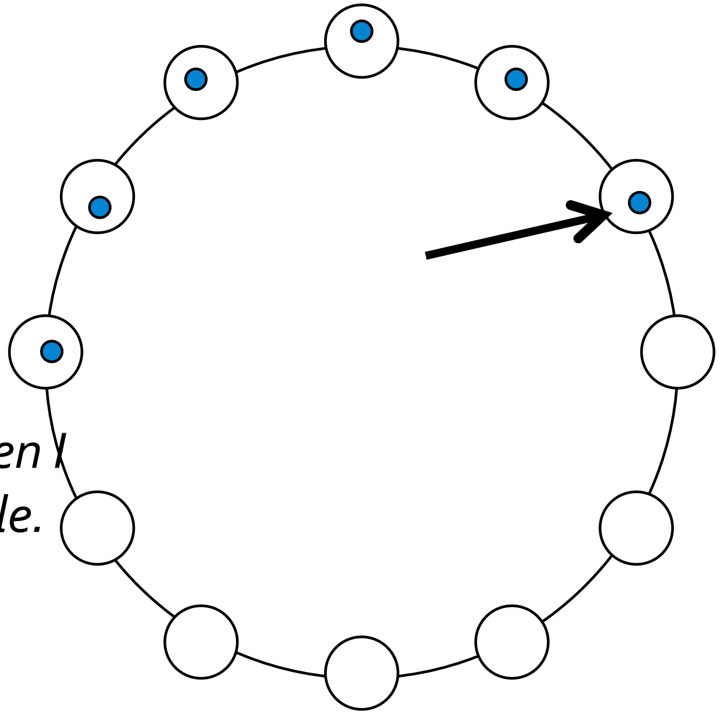


Algorithm

- *Initially, there is no tower*

1. Converge toward a 4-segment
2. Build a tower ➔ Primary arrow
3. Visit the ring and terminate ➔ Final arrow

a) 3-segment



If I am the isolated node, then I move through a shortest hole.

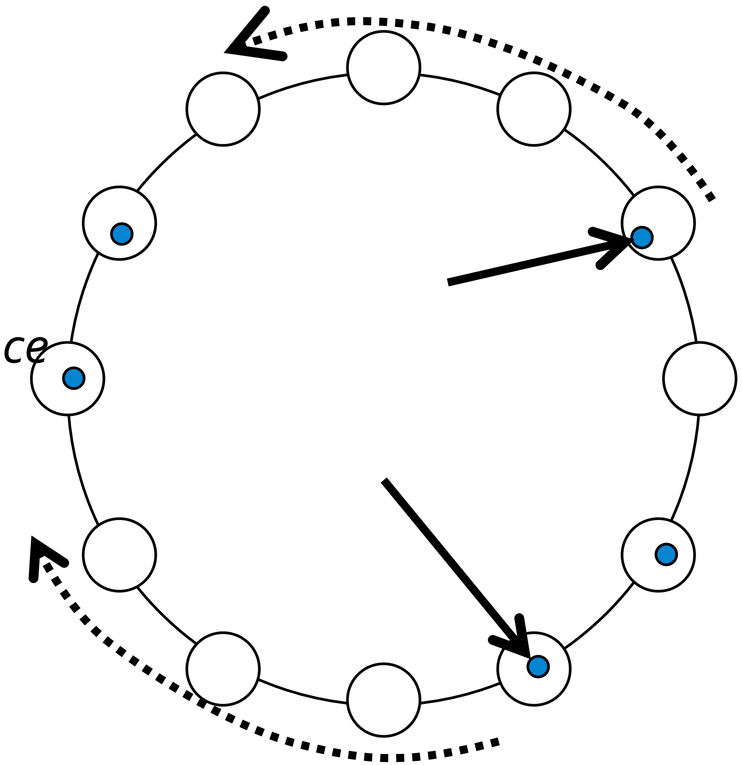
Algorithm

- *Initially, there is no tower*

1. Converge toward a 4-segment
2. Build a tower ➔ Primary arrow
3. Visit the ring and terminate ➔ Final arrow

a) 3-segment
b) a unique 2-segment

If I am at the closest distance from the 2-segment, then I move toward the closest extremity.



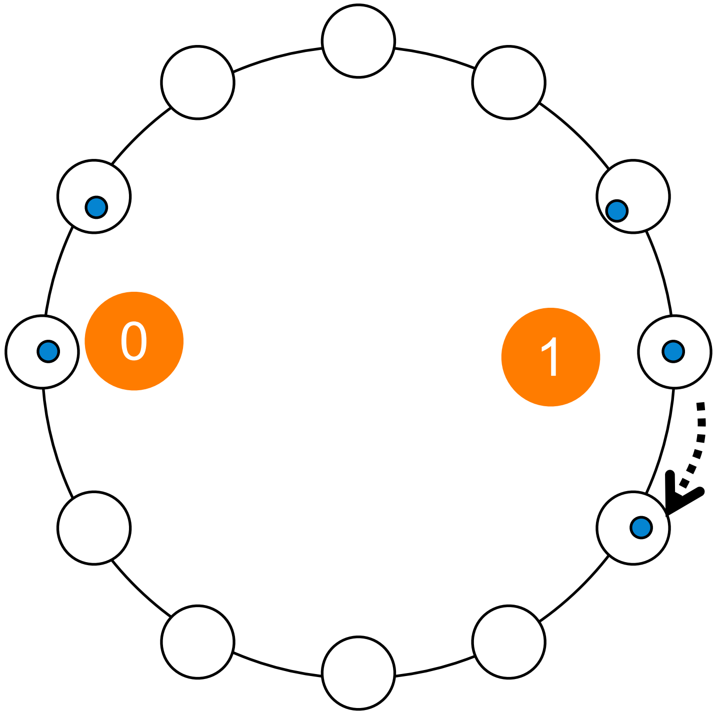
Algorithm

- *Initially, there is no tower*

1. Converge toward a 4-segment
2. Build a tower ➔ Primary arrow
3. Visit the ring and terminate ➔ Final arrow

a) 3-segment
b) a unique 2-segment
c) two 2-segments

If I am a neighbor of the longest hole, then I try to move toward the other 2-segment.



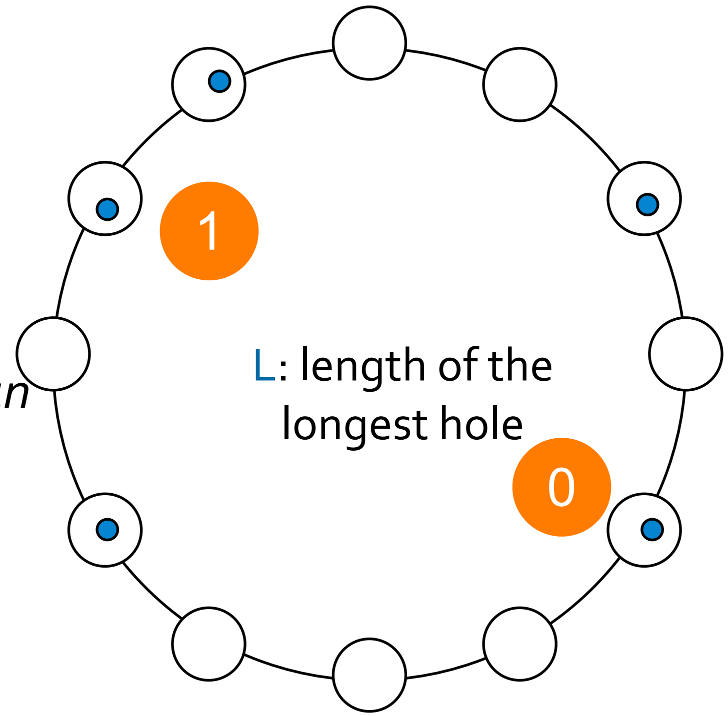
Algorithm

- *Initially, there is no tower*

1. Converge toward a 4-segment
2. Build a tower ➔ Primary arrow
3. Visit the ring and terminate ➔ Final arrow

a) 3-segment
b) a unique 2-segment
c) two 2-segments
d) four isolated nodes

If 4 robots are neighbors of an L-hole, then I try to move through my longest neighboring hole.



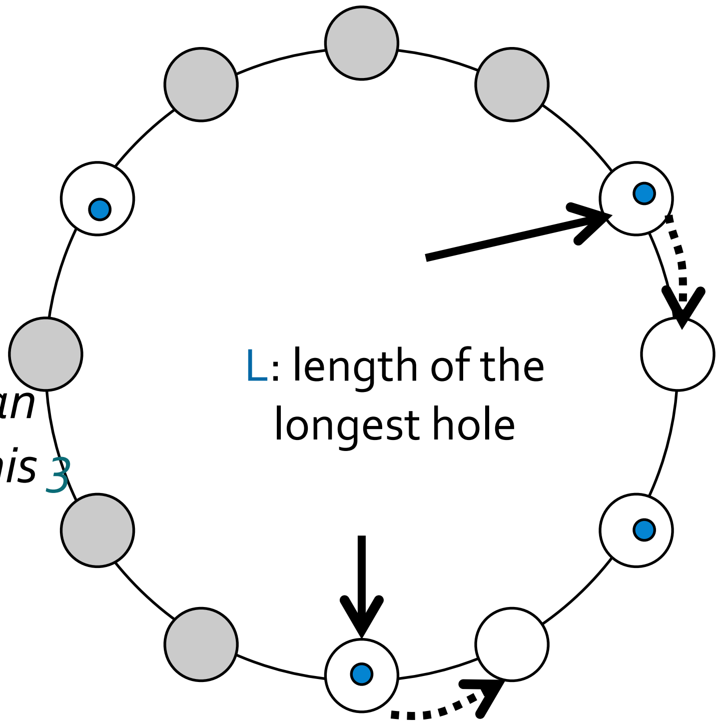
Algorithm

- *Initially, there is no tower*

1. Converge toward a 4-segment
2. Build a tower ➔ Primary arrow
3. Visit the ring and terminate ➔ Final arrow

a) 3-segment
b) a unique 2-segment
c) two 2-segments
d) four isolated nodes

If 3 robots are neighbors of an L-hole, then if I am one of this 3 robots and a neighbor of a smaller hole h, then I move through h.



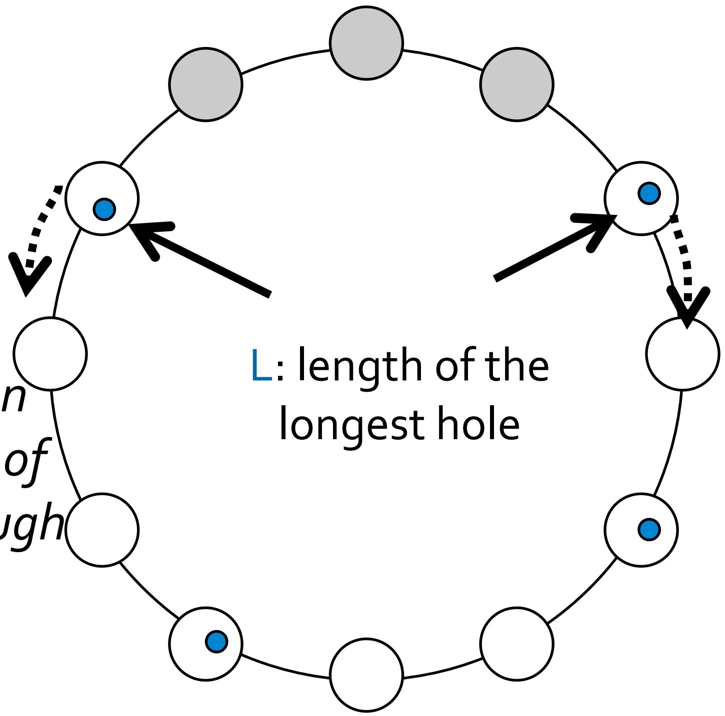
Algorithm

- *Initially, there is no tower*

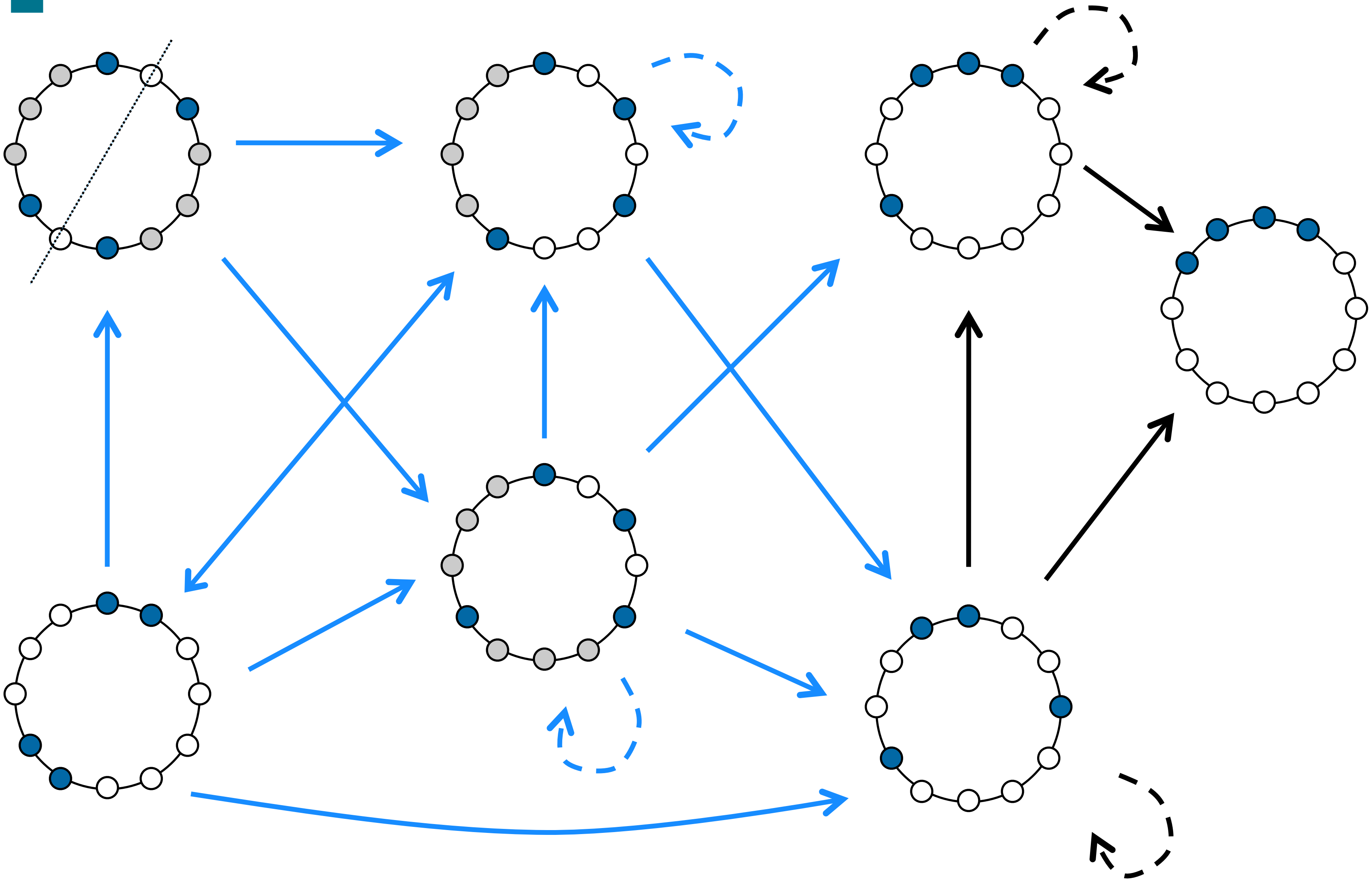
1. Converge toward a 4-segment
2. Build a tower ➔ Primary arrow
3. Visit the ring and terminate ➔ Final arrow

a) 3-segment
b) a unique 2-segment
c) two 2-segments
d) four isolated nodes

If 2 robots are neighbors of an L-hole, then if I am neighbor of the L-hole, then I move through the other neighboring hole.



Phase 1, Summary

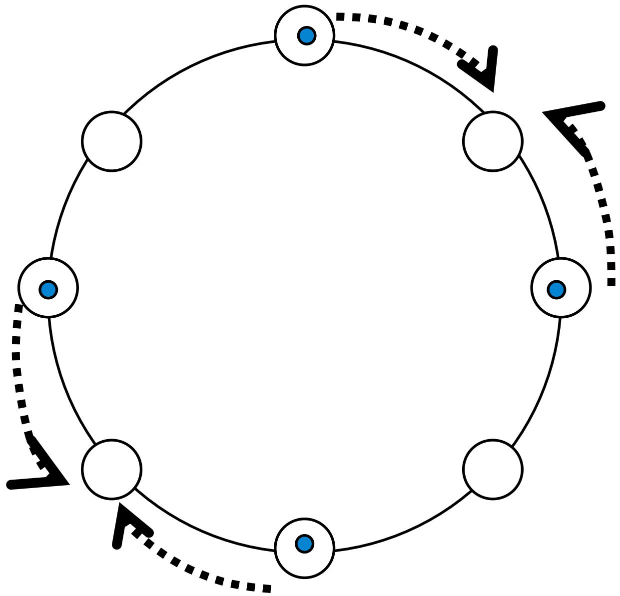


Proof

Lemma.
No tower is created during Phase 1 in a n-ring with n > 8.

Proof Bas:

With $n > 8$ and 4 robots, there always exists a hole of length greater than 1.



Proof

__Lemma__.
No tower is created during Phase 1 in a n-ring with $n > 8$.

__Lemma__.
Starting from any initial configuration, the system reaches in finite expected time a configuration containing a 4-segment.

__Theorem__.
The algorithm (Phases 1 to 3) is a probabilistic exploration protocol for 4 robots in a ring of $n > 8$ nodes.

Conclusion

n: Number of nodes
k: Number of agents

✓ *Ring* [*Flocchini et al.*, OPODIS 2007] [*Devismes et al.*, SIROCCO 2009] [*Lamani et al.*, SIROCCO 2010]

- Deterministic exploration impossible if k divides n (except if $k = n$)
- Asynchronous deterministic algorithm with $k > 16$

- Deterministic or probabilistic exploration impossible if $k < 4$
- Probabilistic algorithm impossible in asynchronous settings
- Optimal Semi-synchronous Probabilistic Algorithm

- Deterministic exploration impossible if $k < 5$ and n even
- Optimal asynchronous deterministic algorithm, k = 5 and n even
- Optimal semi-synchronous deterministic algorithm, k = 4 and n odd

Conclusion

n: Number of nodes
k: Number of agents

✓ *Ring* [*Flocchini et al.*, OPODIS 2007] [*Devismes et al.*, SIROCCO 2009] [*Lamani et al.*, SIROCCO 2010]

✓ *Tree* [*Flocchini et al.*, SIROCCO 2008]

- Asynchronous deterministic algorithm for trees with maximum degree equal to 3: $k \in \Theta(\log n / \log \log n)$
- Arbitrary tree: $k \in \Theta(\log n)$

✓ *Chain* [*Flocchini et al.*, IPL 2011]

- Characterization of k: $k = 3$, $k > 4$, or $k = 4$ and n odd

✓ *Grid* [*Devismes et al.*, SSS 2012]

- Deterministic or probabilistic exploration impossible if k < 2
- Optimal Semi-synchronous Deterministic Algorithm, $k = 3$

Optimal Grid Exploration by Asynchronous Oblivious Robots

Franck Petit

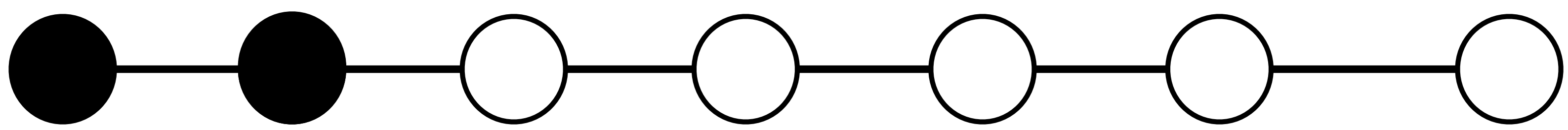
INRIA/LIP6-CNRS/UPMC

Impossibility Results (1)

✓ *Impossible if k<3*

Remark

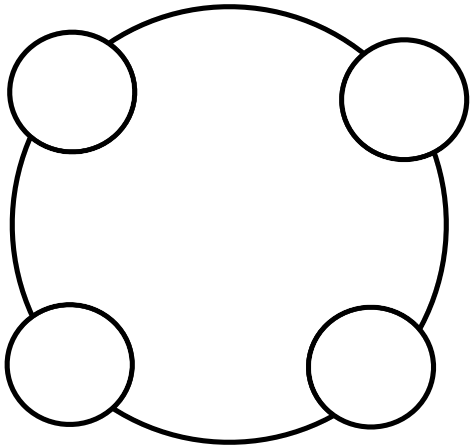
Any terminal configuration of any (probabilistic or deterministic) exploration protocol for a grid of n nodes using $k < n$ oblivious robots contains at least one tower.



Initial configuration

Impossibility Results (2)

✓ *Impossible for (2,2)-grid if k=3*



Impossible [Devismes 2009]

Impossibility Results (3)

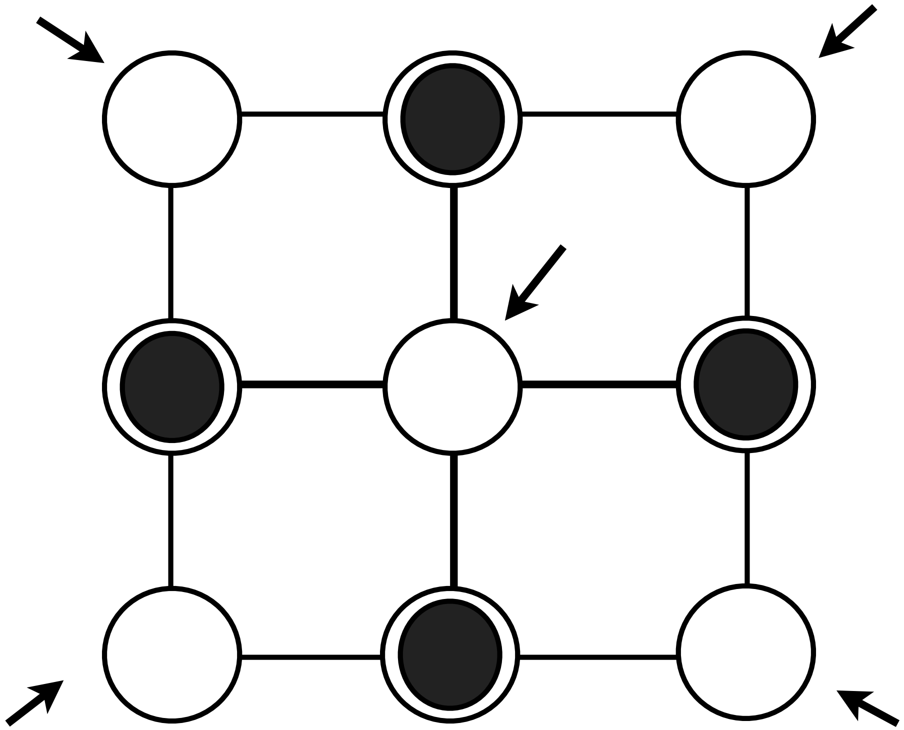
✓ *Impossible for (3,3)-grid if k=3*

- ❑ Tower of size 3 → At most one new node is visited
- ❑ Tower of size 2
  - The tower remains idle → 4 new nodes are visited (7 distinct nodes are visited in total)

    Impossible since n=6
  - The tower moves → Multiplicity of size 3

Impossibility Results (4)

- ✓ *Impossible for (3,3)-grid if k=4*



Algorithm

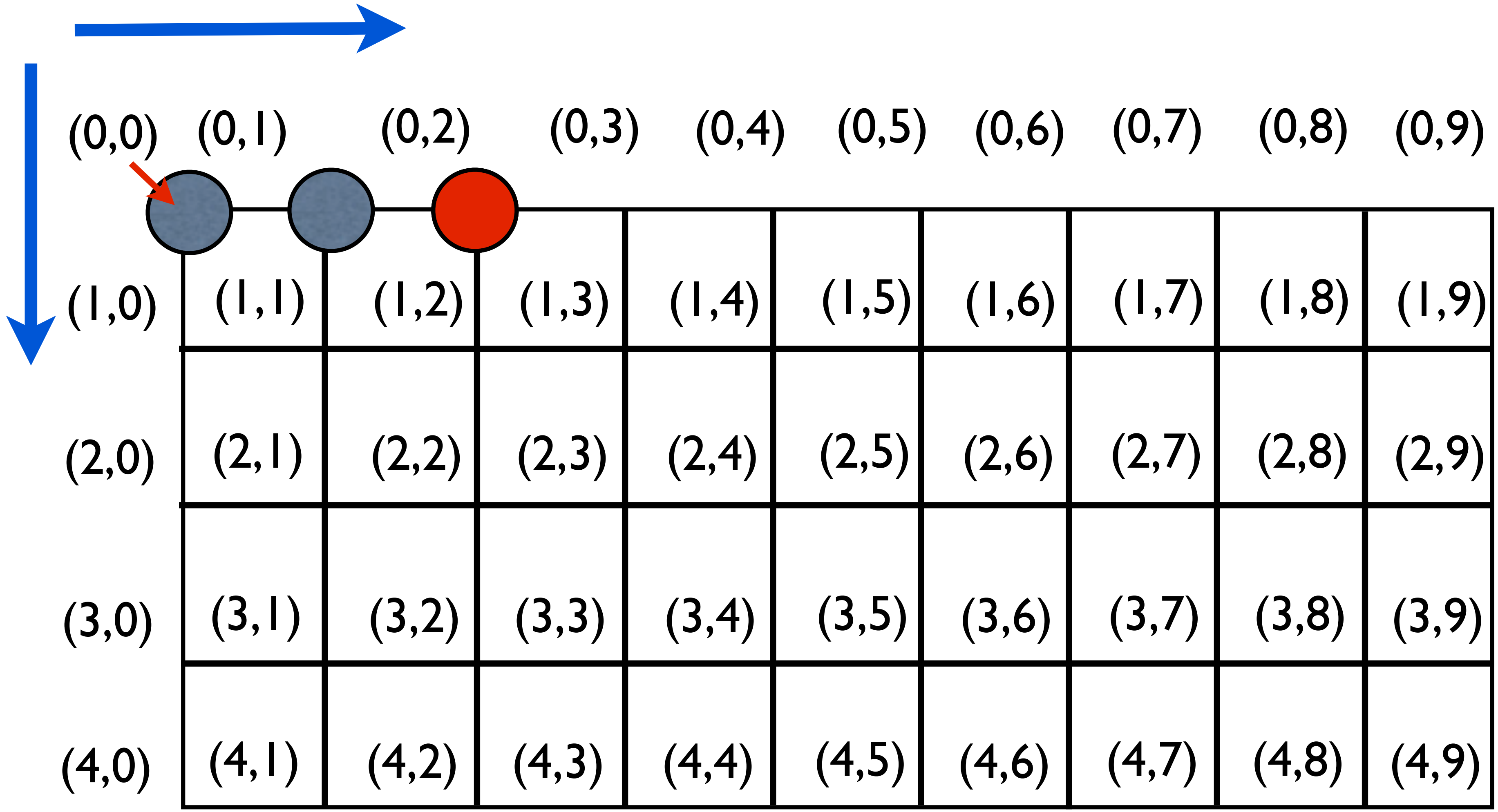
- ❑ Setting
  - ✓ (i,j)-grids such that j >3
  - ✓ k=3
  - ✓ Towerless initial configuration

Algorithm

- ❑Phase 1: Set-Up phase
- ❑Phase 2: Orientation phase
- ❑Phase 3: Exploration phase

Algorithm

❑ Phase 2: Exploration phase

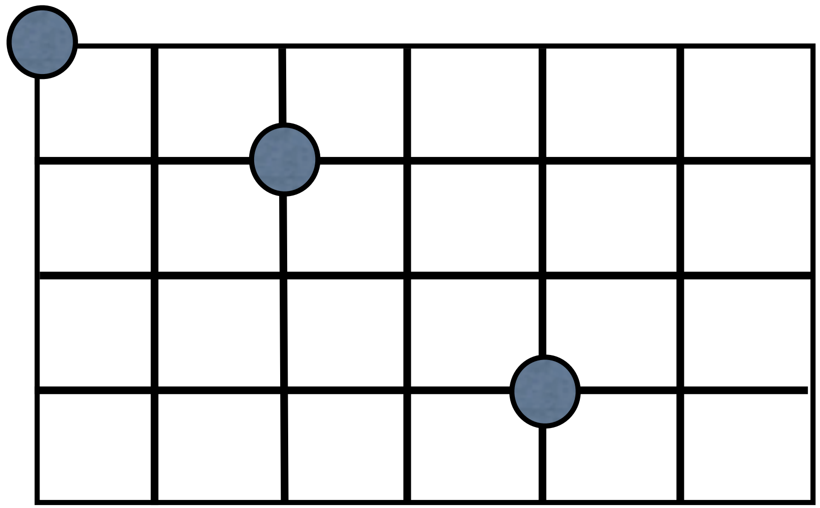


Algorithm

- Phase1: Set-Up

  ✓ Configuration of type Leader

  There is exactly one robot at a corner of the grid

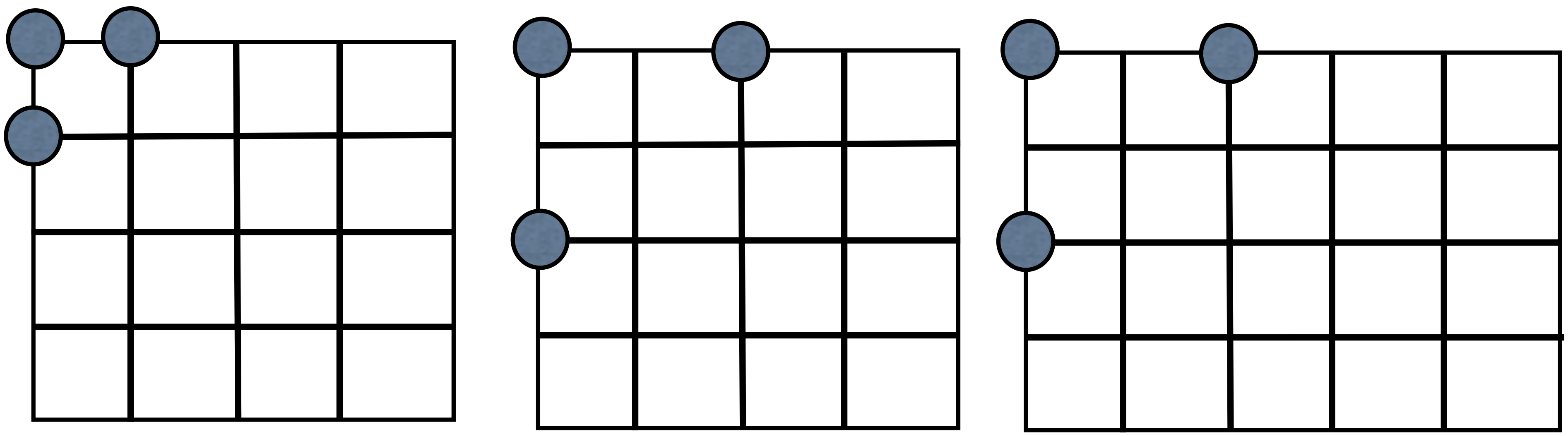


Algorithm

❑ Phase1: Set-Up

✓ Configuration of type Leader

There is exactly one robot at a corner of the grid

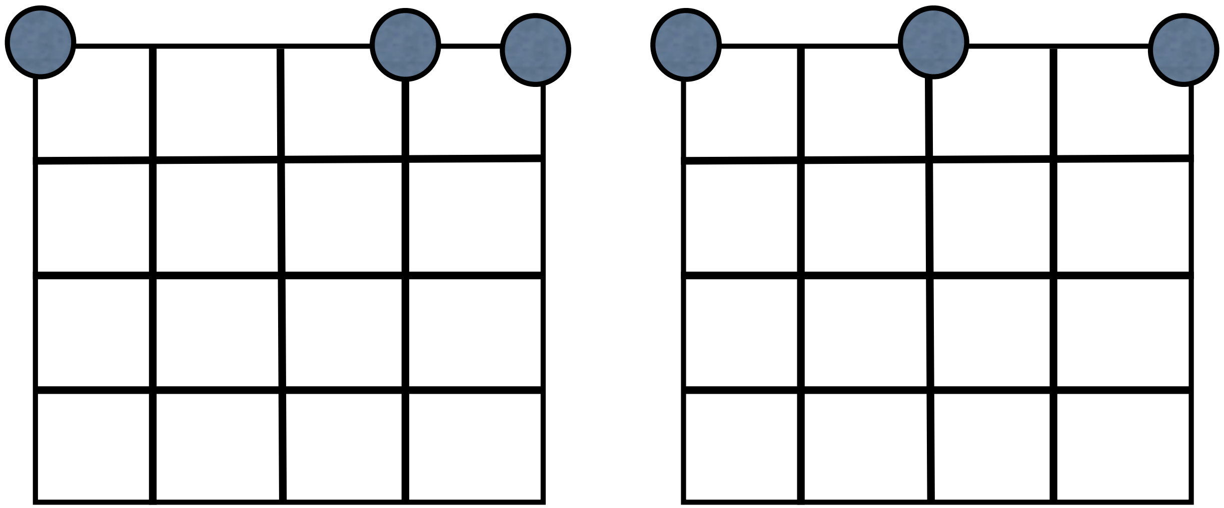


Algorithm

- Phase1: Set-Up

  ✓ Configuration of type Choice

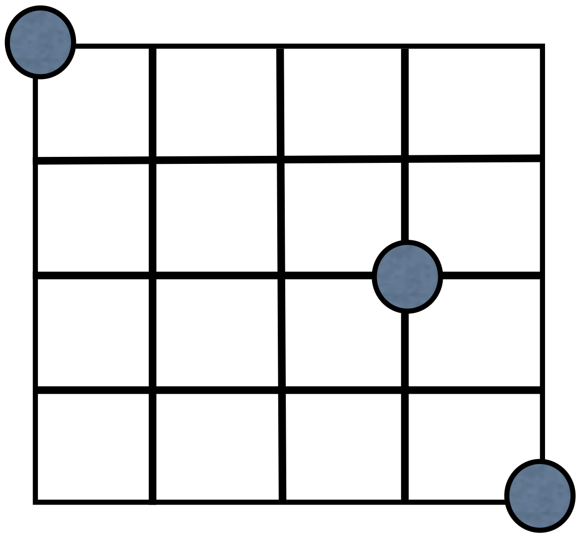
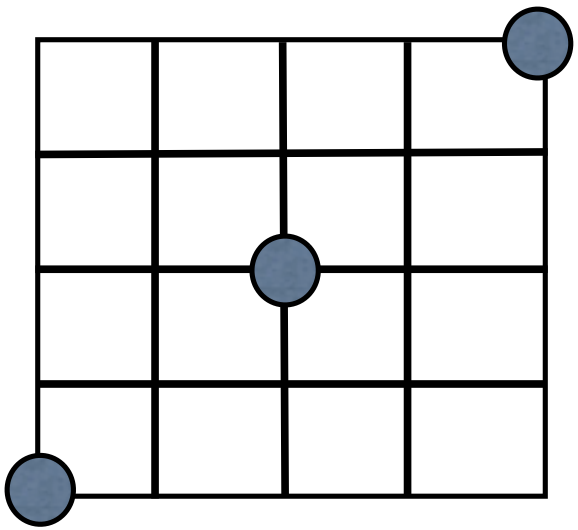
  There are at least two robots at a corner of the grid



Algorithm

- ❑Phase1: Set-Up
  - ✓ Configuration of type Choice

    There are at least two robots at a corner of the grid

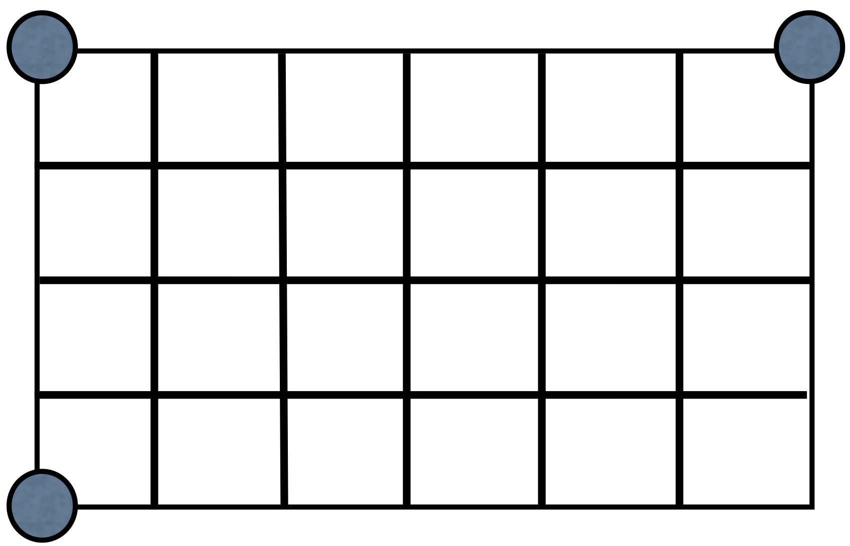



Algorithm

- Phase1: Set-Up

  ✓ Configuration of type Choice

  There are at least two robots at a corner of the grid

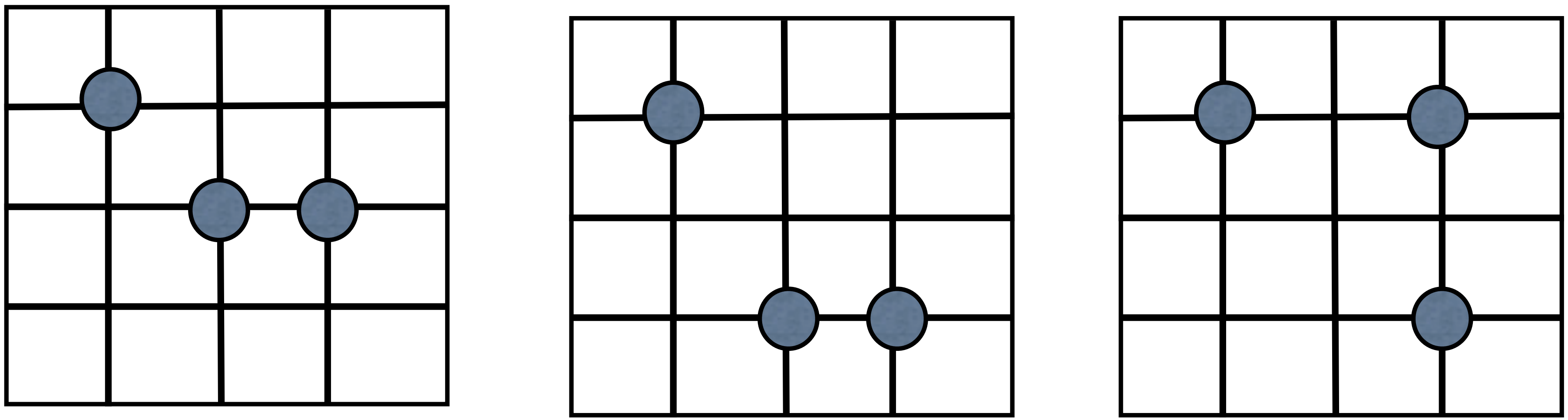


Algorithm

❑Phase1: Set-Up

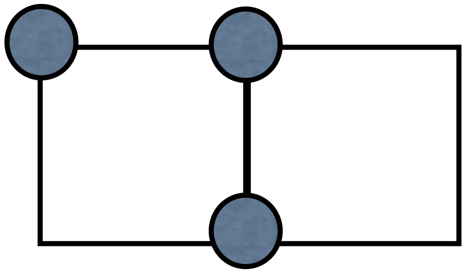
✓ Configuration of type Undefined

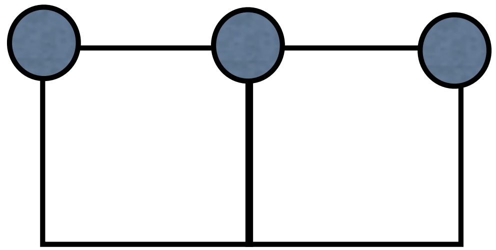
All the corners of the grid are free



Algorithm

- Special grid: (3,2) grids

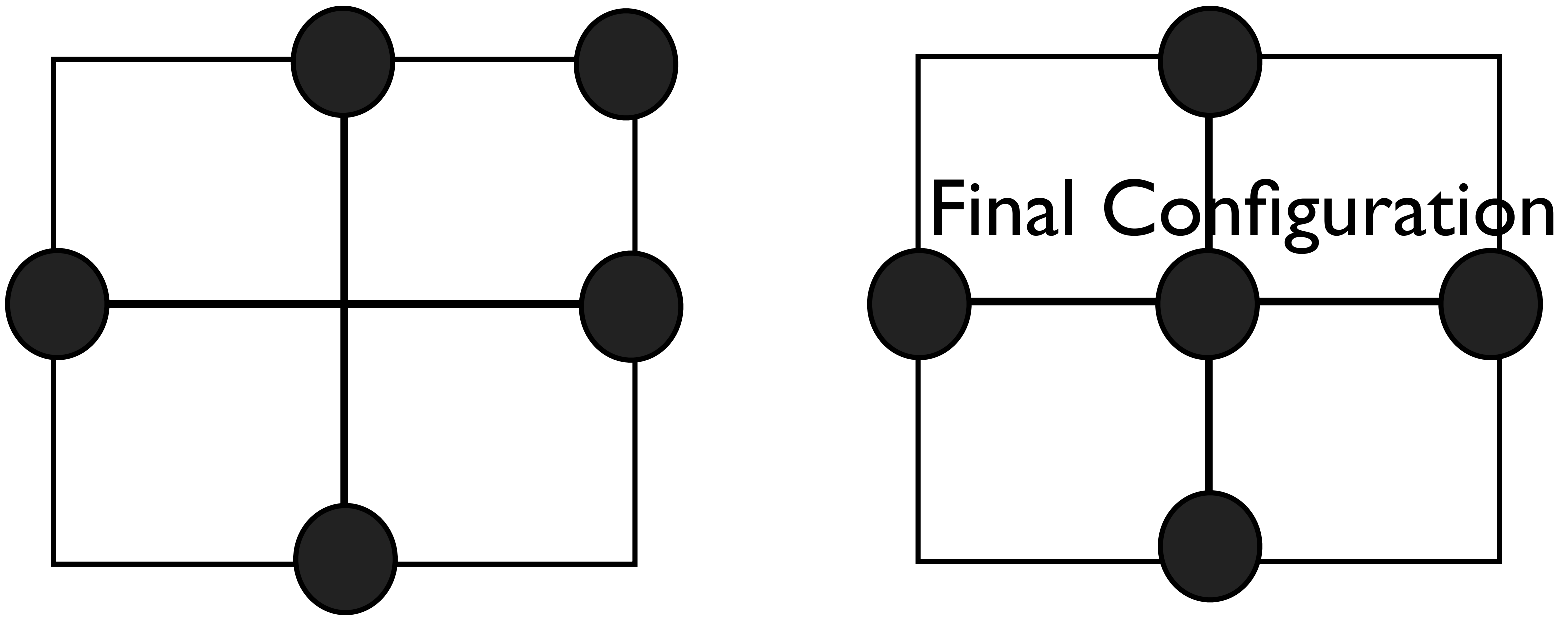




Final Configuration

Special case

- (3,3)-grids with 5 robots



Conclusion

- ✓ 5 robots are necessary and sufficient to explore (3,3)-grids
- ✓ 4 robots are necessary and sufficient to explore (2,2)-grids
- ✓ 3 robots are necessary and sufficient to explore (i,j)-grids such that j >3