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
- 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

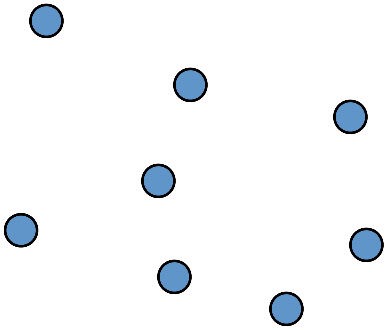
General aim of the study

- ***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

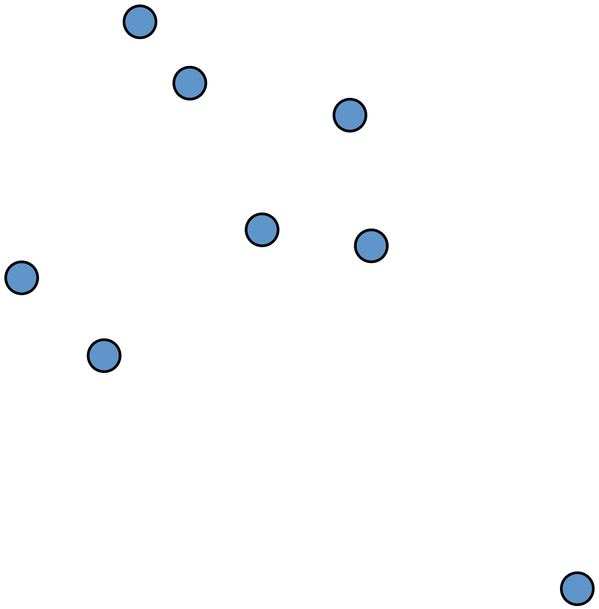
Environment

- ? Plane



Environment

- Plane



Environment

⍰ Plane

⍰ Discrete

Environment

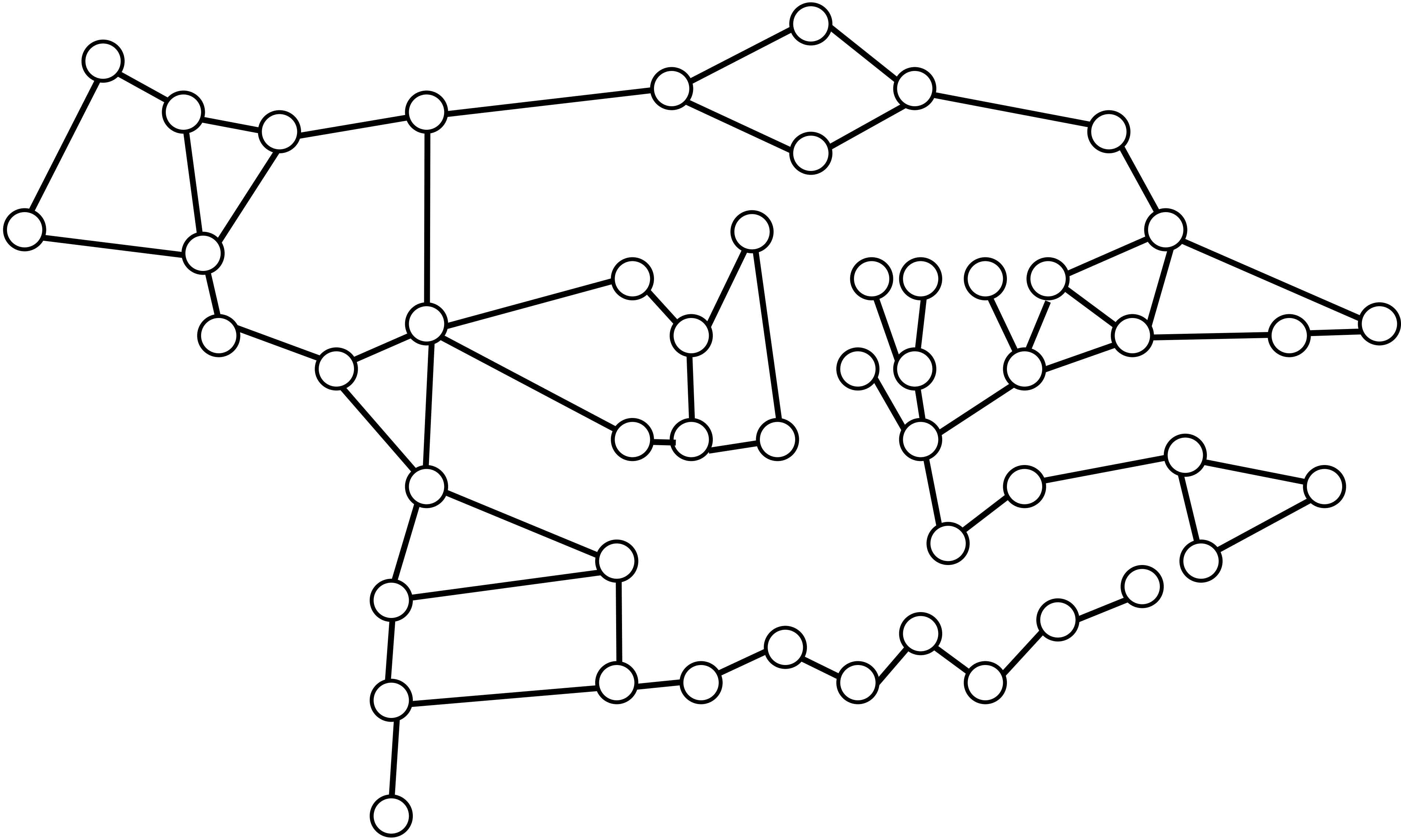
? Plane

? Discrete

Environment

? Plane

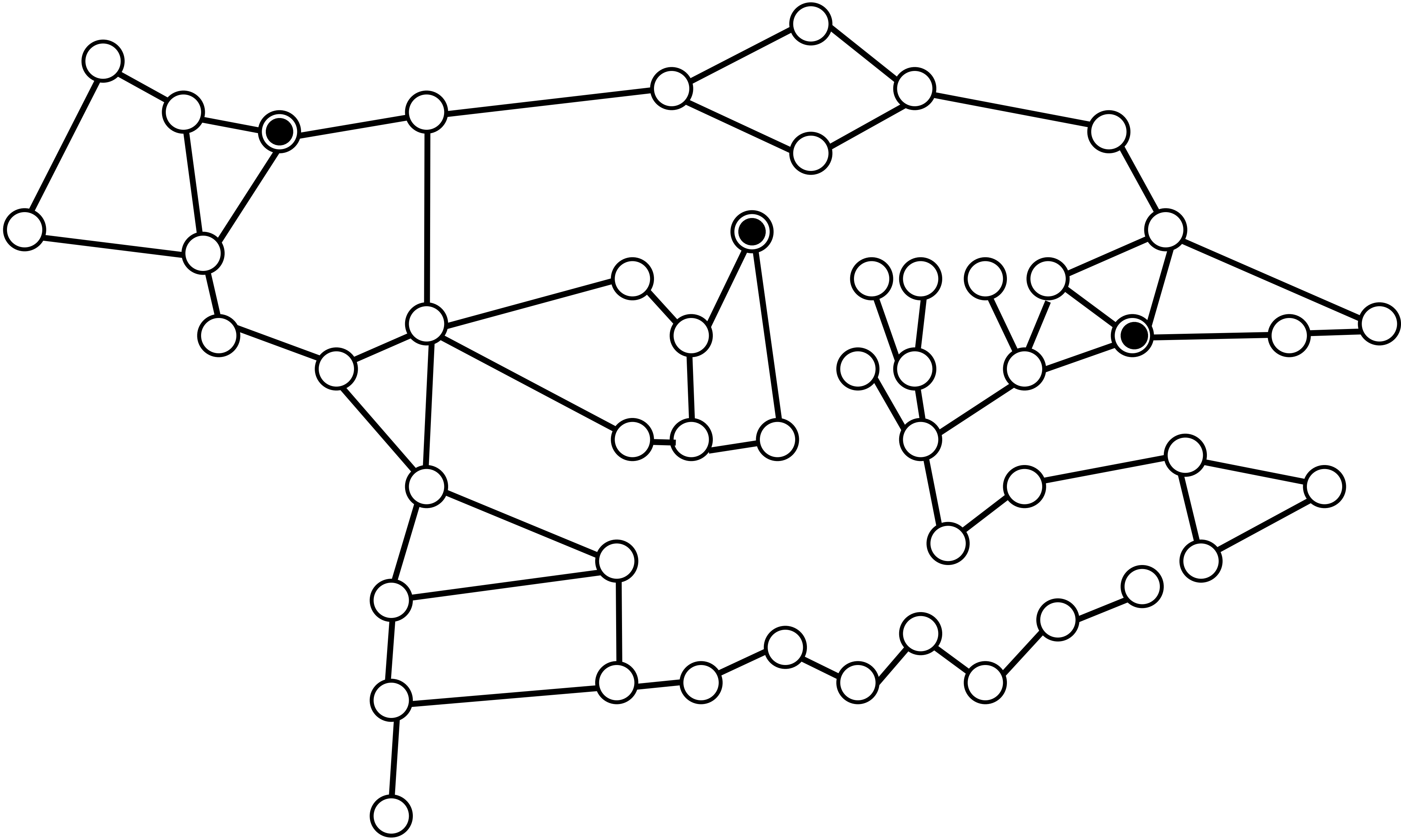
? Discrete



Environment

? Plane

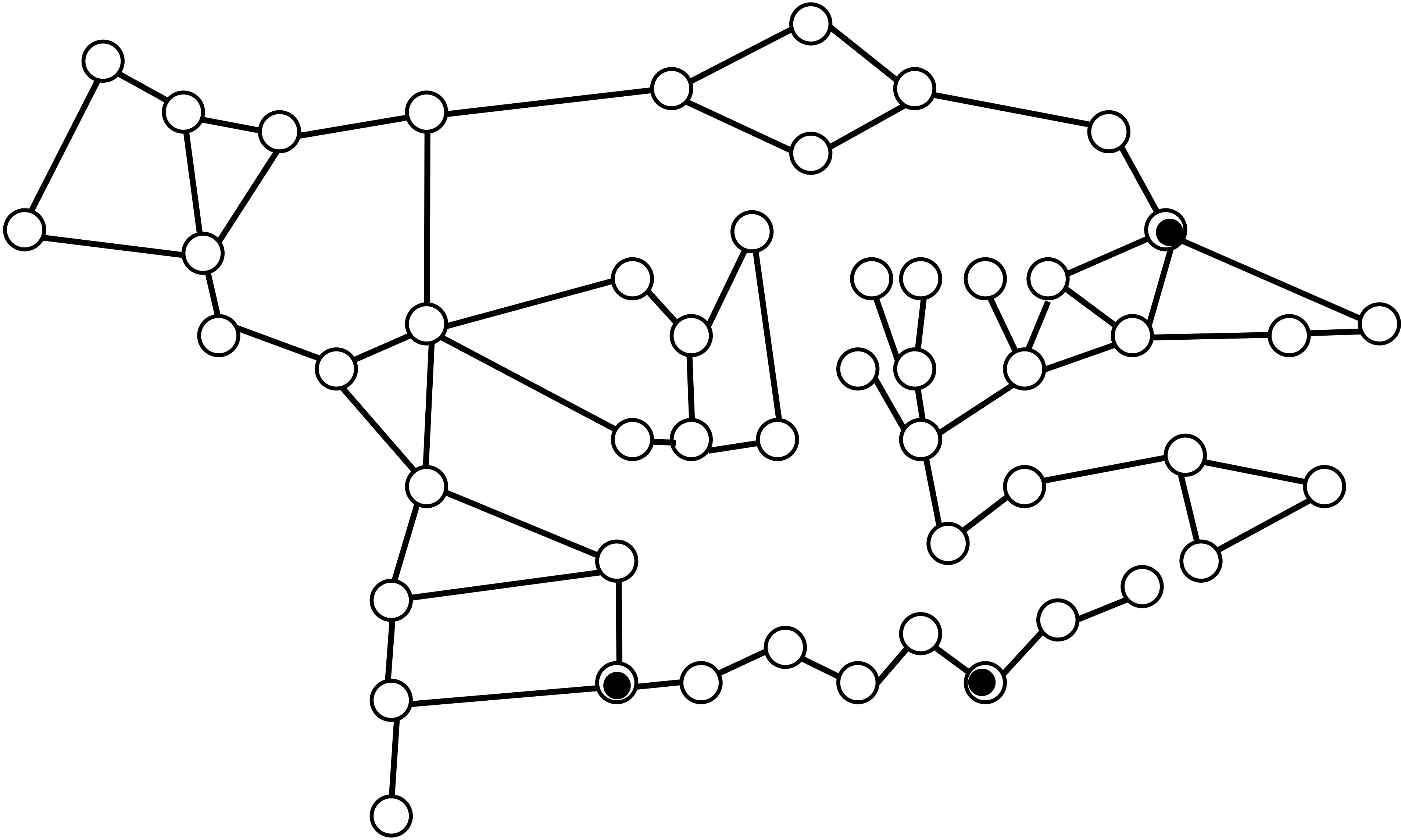
? Discrete



Environment

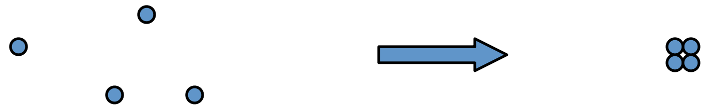
? Plane

? Discrete

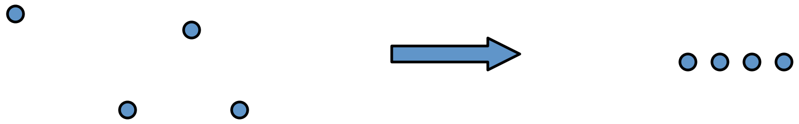


Cooperative Primitives over the Plane

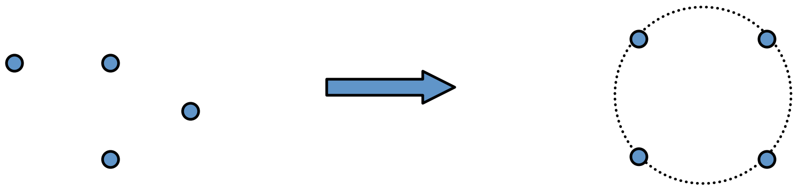
- Gathering



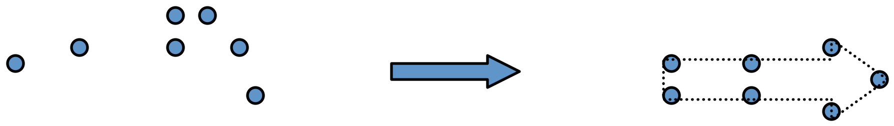
- Alignment



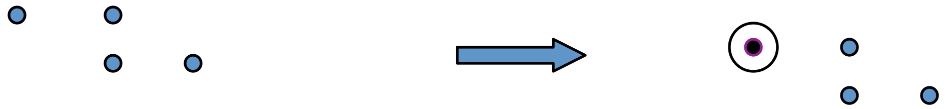
- Circle Formation (n-gon)



- Other Patterns

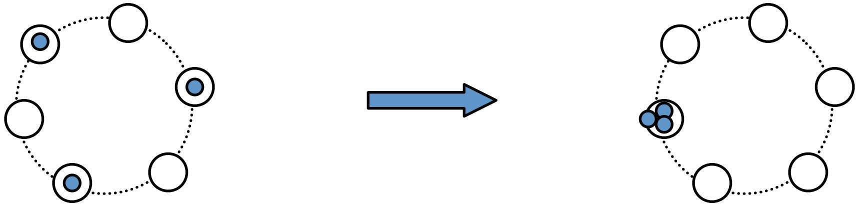


- Election

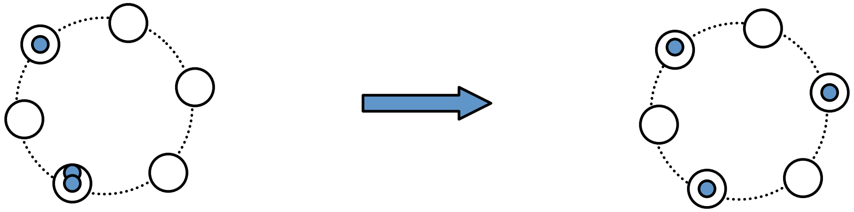


Cooperative Primitive Tasks in discrete environment

- 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)

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

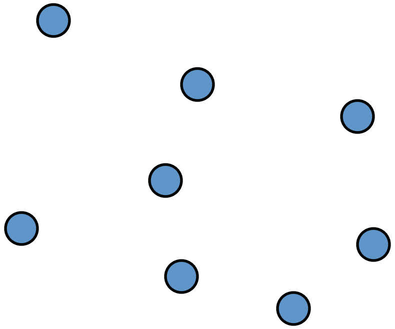
The Model

(Yamashita et al., SIROCCO 1996)

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

? Sensors

? No explicit communication

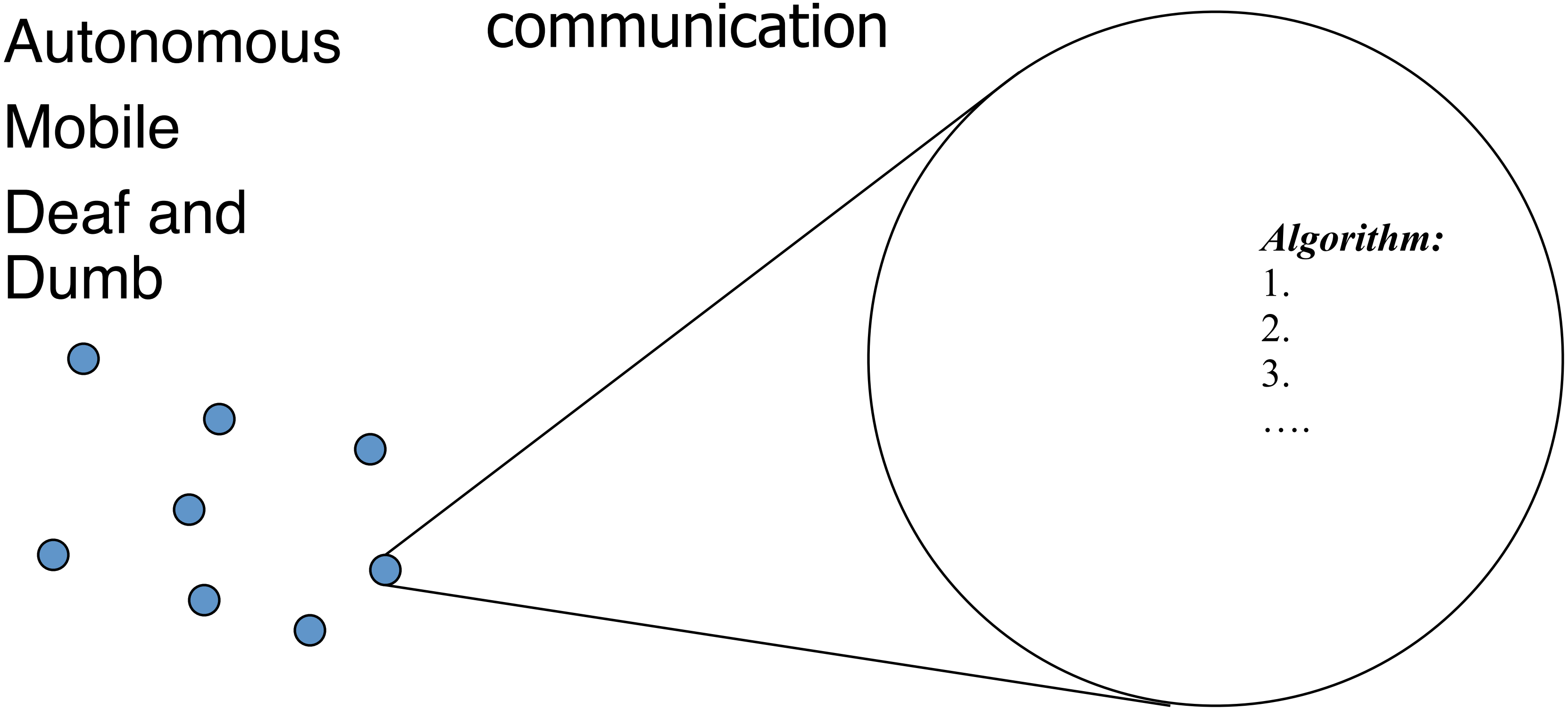


The Model
(Yamashita et al., SIROCCO 1996)

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

? Sensors

? No explicit communication

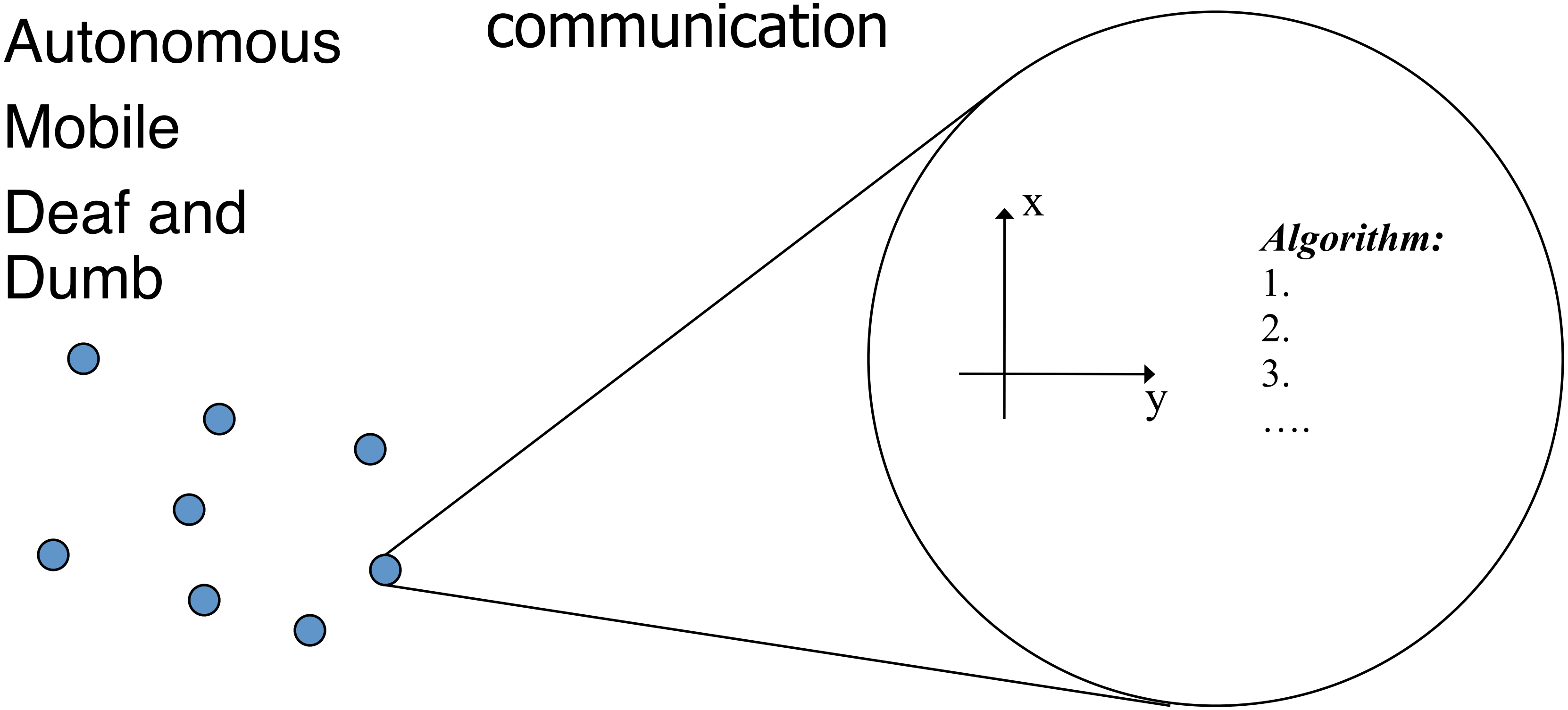


The Model

(Yamashita et al., SIROCCO 1996)

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

- ? Sensors
- ? No explicit communication

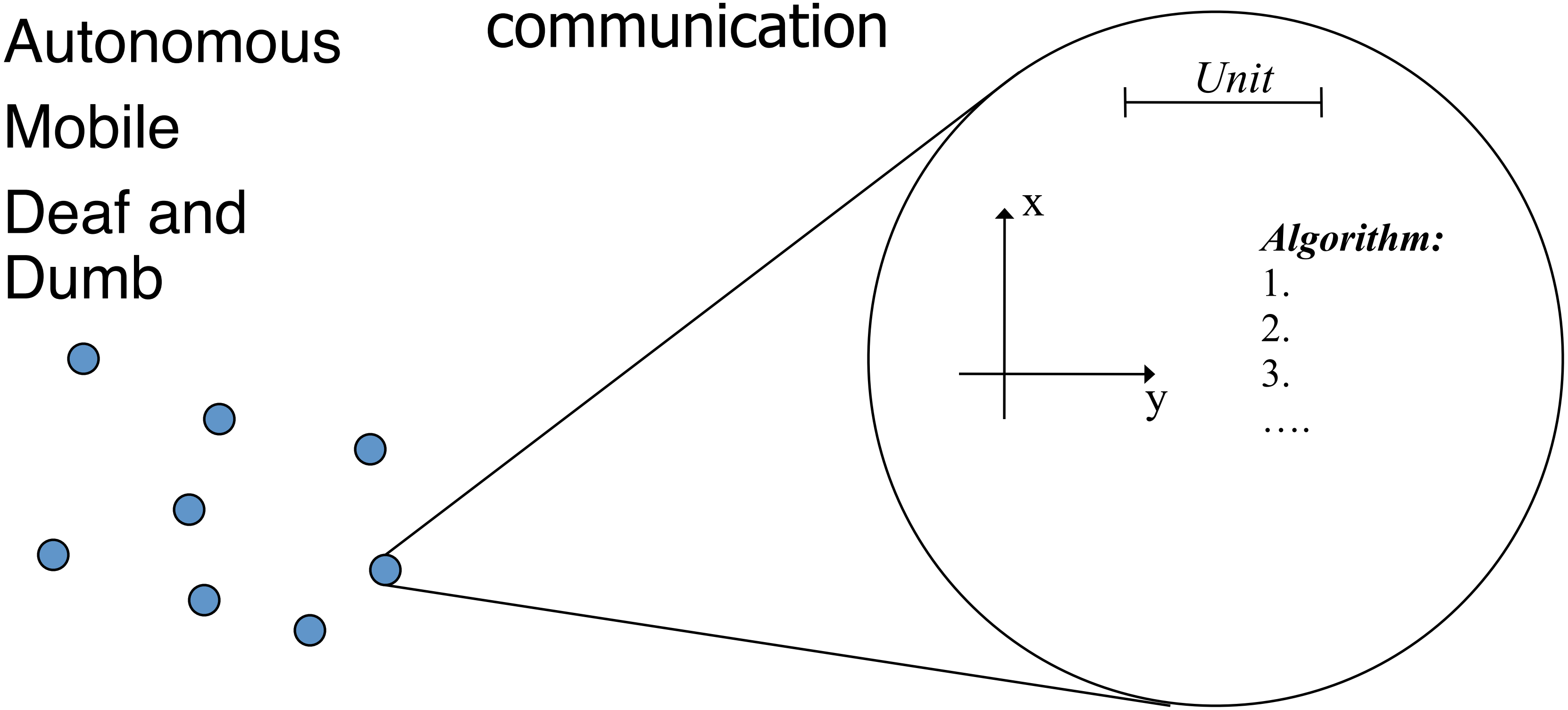


The Model
(Yamashita et al., SIROCCO 1996)

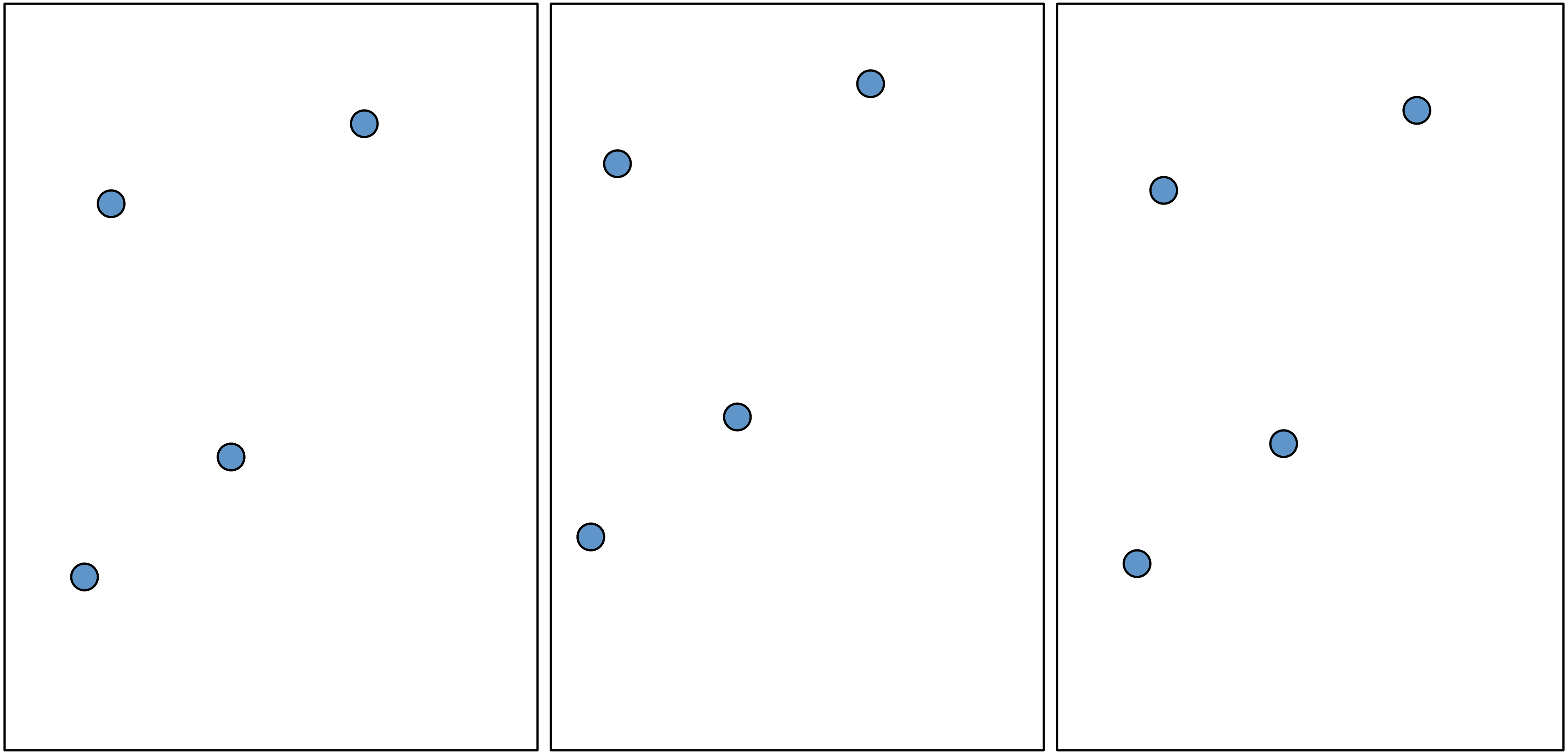
- Homogeneous
- Anonymous
- Autonomous
- Mobile
- Deaf and Dumb

? Sensors

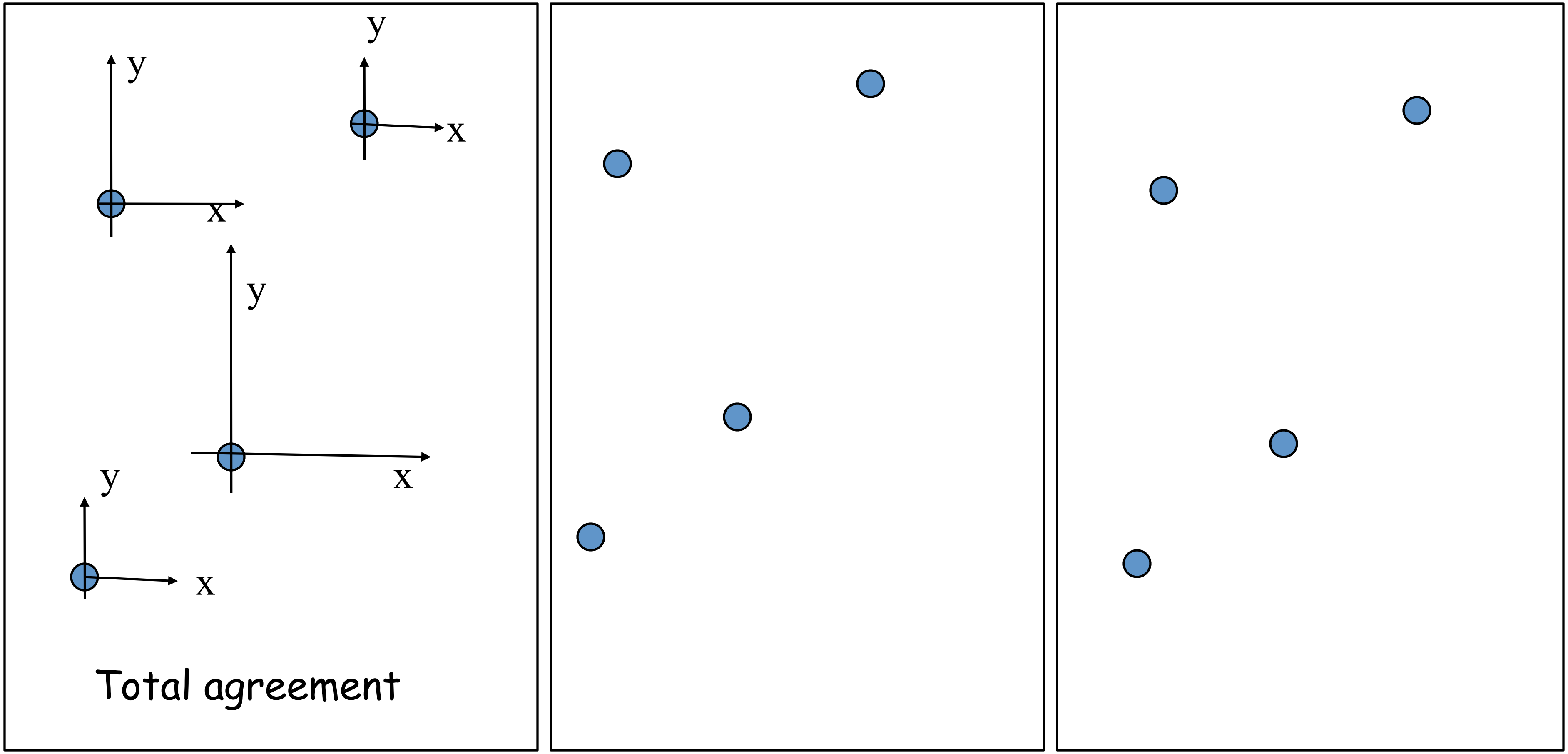
? No explicit communication



-Assumptions on Robots' power-
Agreement on Coordinate System

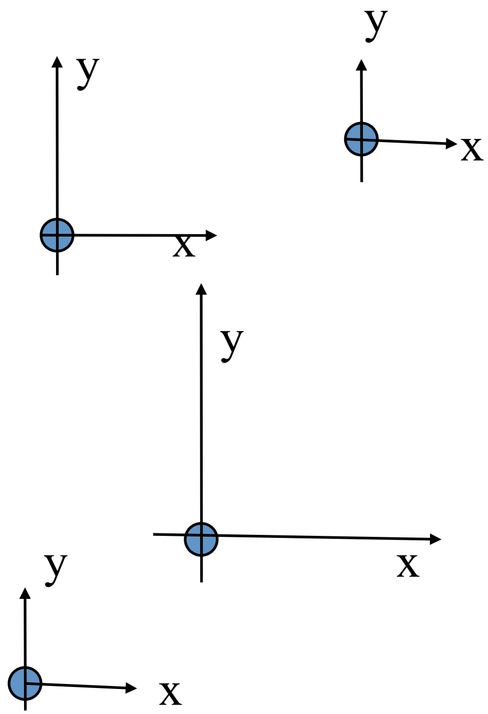


-Assumptions on Robots' power- Agreement on Coordinate System

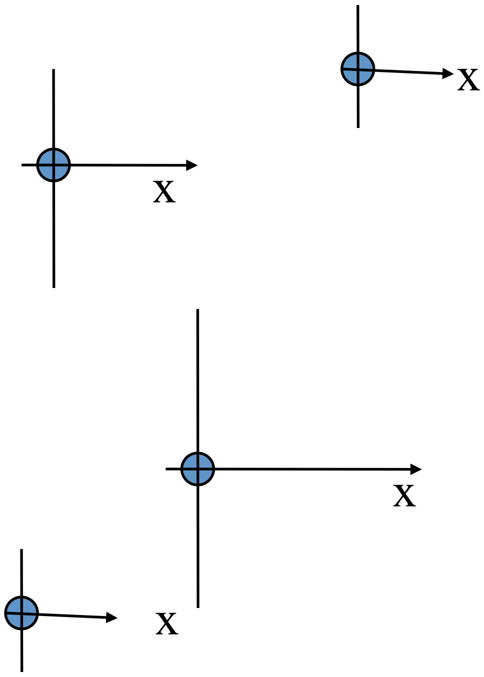


Total agreement

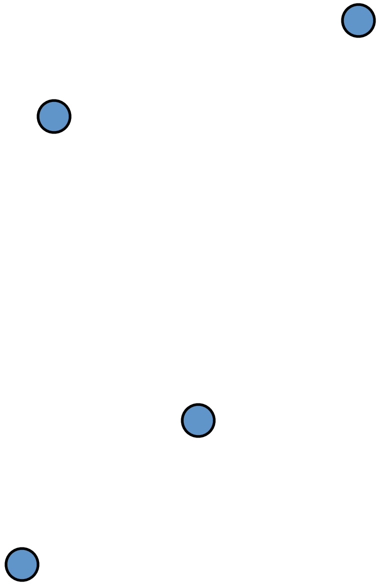
-Assumptions on Robots' power- Agreement on Coordinate System



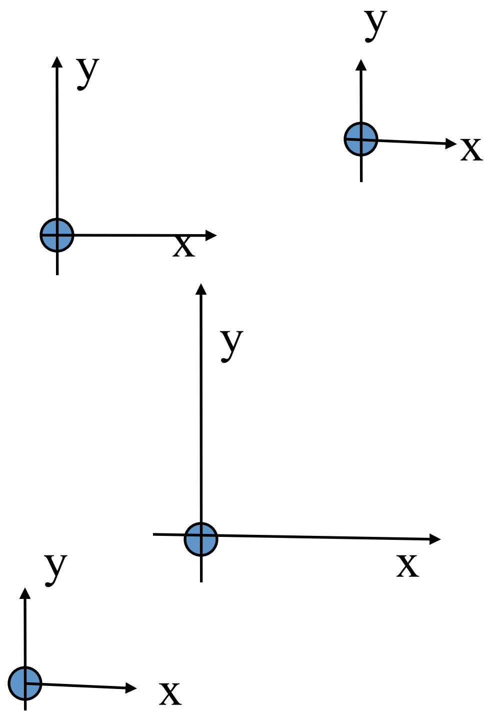
Total agreement



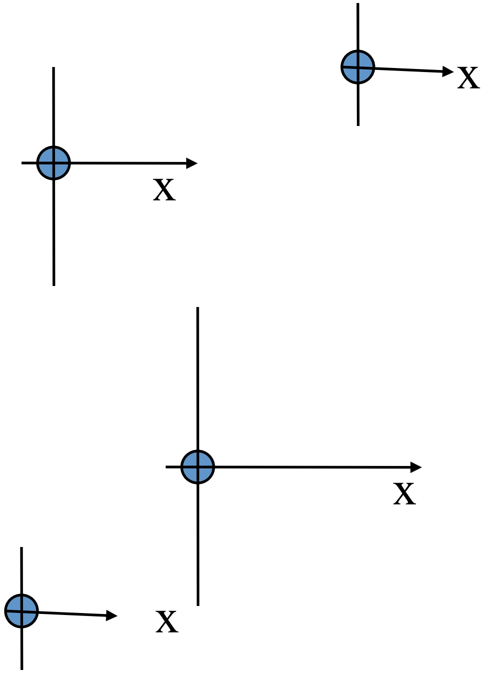
Agreement on one axis direction and orientation



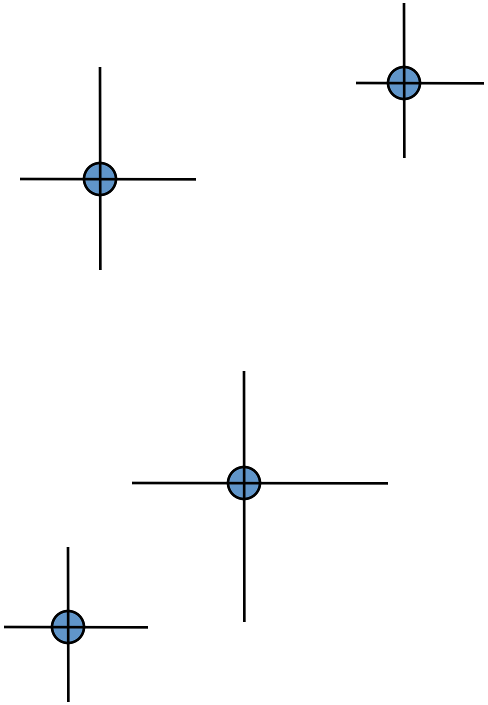
-Assumptions on Robots' power-
Agreement on Coordinate System



Total agreement

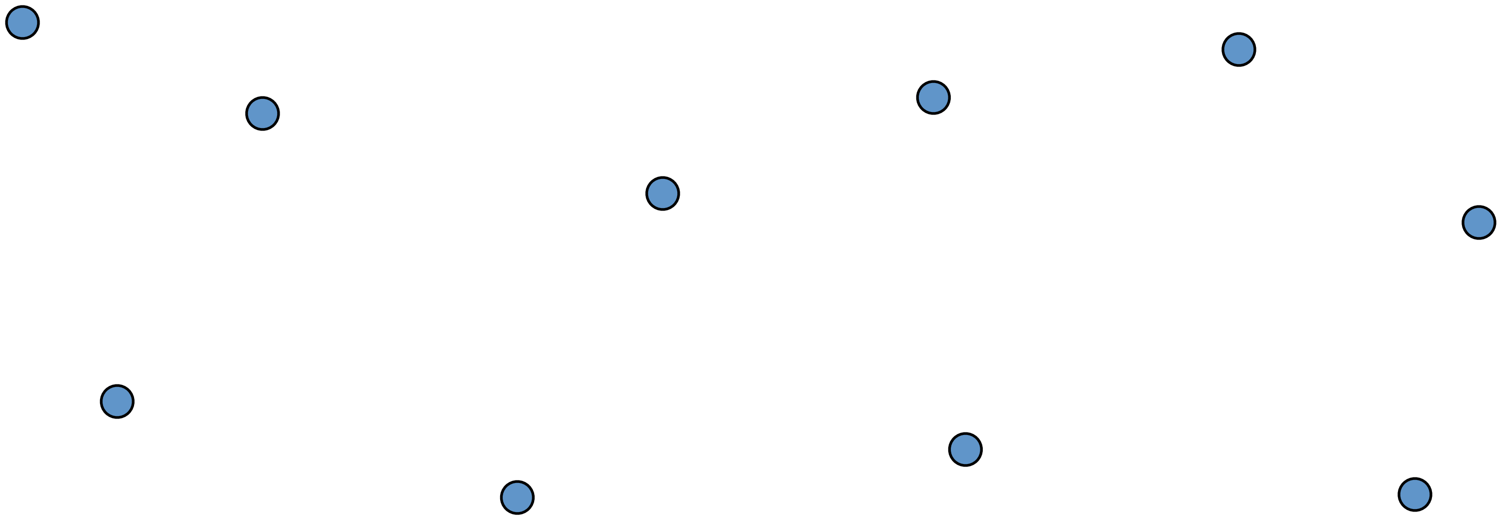


Agreement on one axis direction and orientation

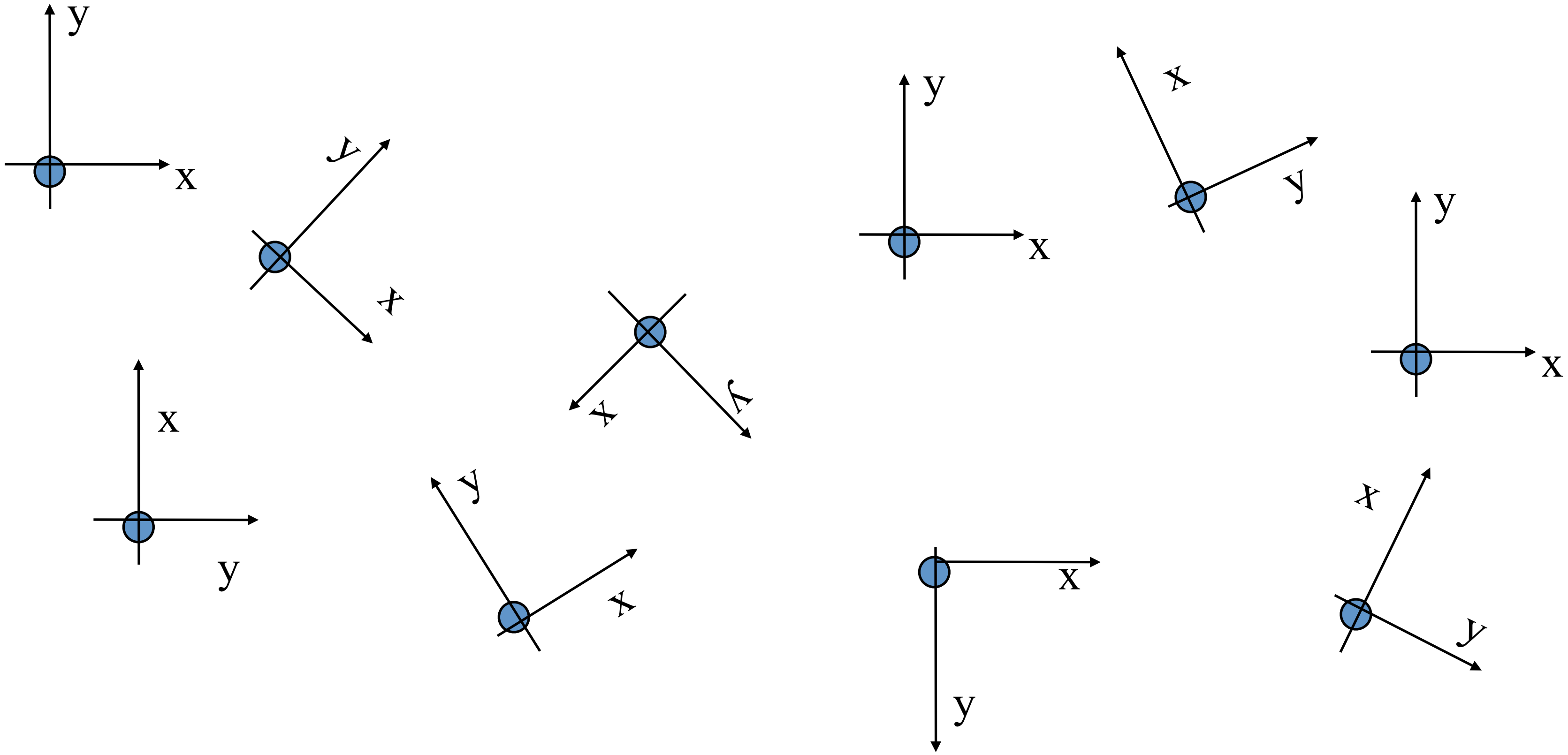


Agreement on directions

- Assumptions on Robots' power - No Agreement



- Assumptions on Robots' power - No Agreement

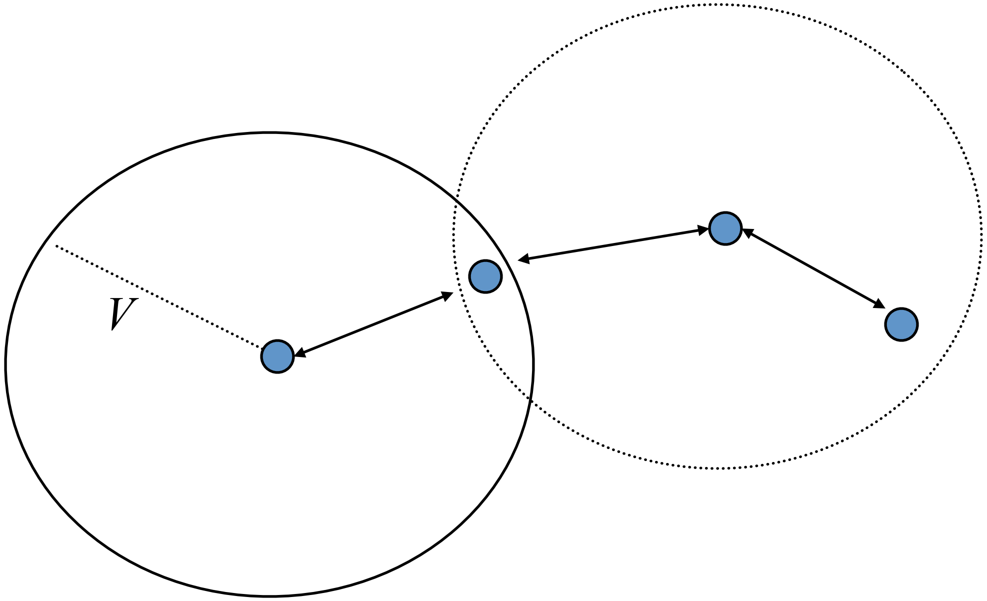


Models of Orientation

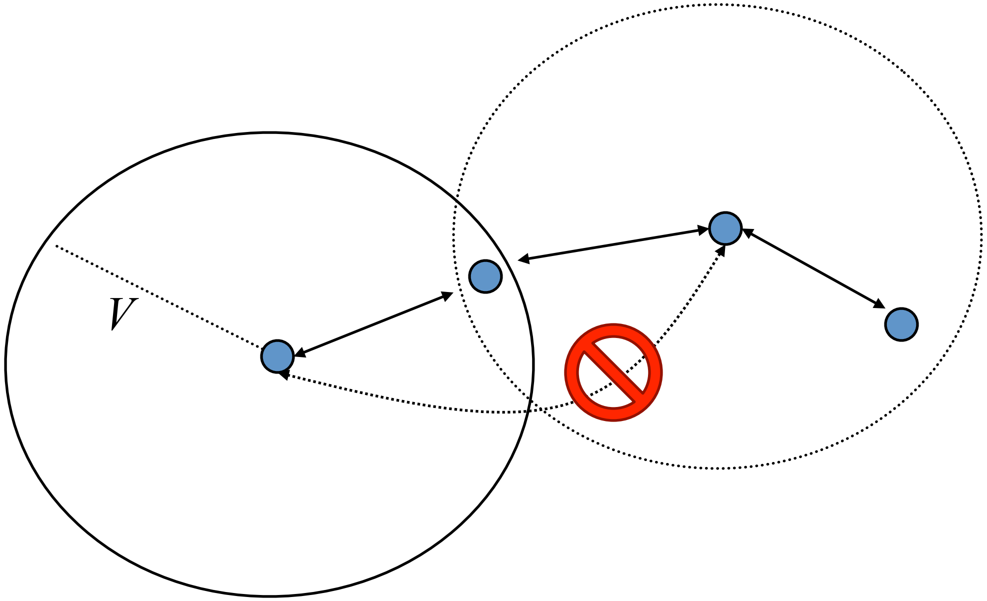
- Full-compass: Axes and polarities of both axes.
- Half-compass: Both axes known, but positive polarity of only one axis (in other axis, robots may have different views of positive direction).
- Direction-only: Both axes, but not polarities.
- Axes-only: Both axes, but not polarities. In addition, robots 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 -
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)
  - Semi-synchronous
  - Fully-synchronous

Instantaneous Actions of **SSync**

Suzuki *et al.*, 1996

Instantaneous Actions of **SSync**

Suzuki *et al.*, 1996

- Discrete Time 0,1,...

Instantaneous Actions of **SSync**

Suzuki *et al.*, 1996

- Discrete Time 0,1,...

- At each time instant t, every robot r_i is either Active or Inactive

Instantaneous Actions of **SSync**

Suzuki *et al.*, 1996

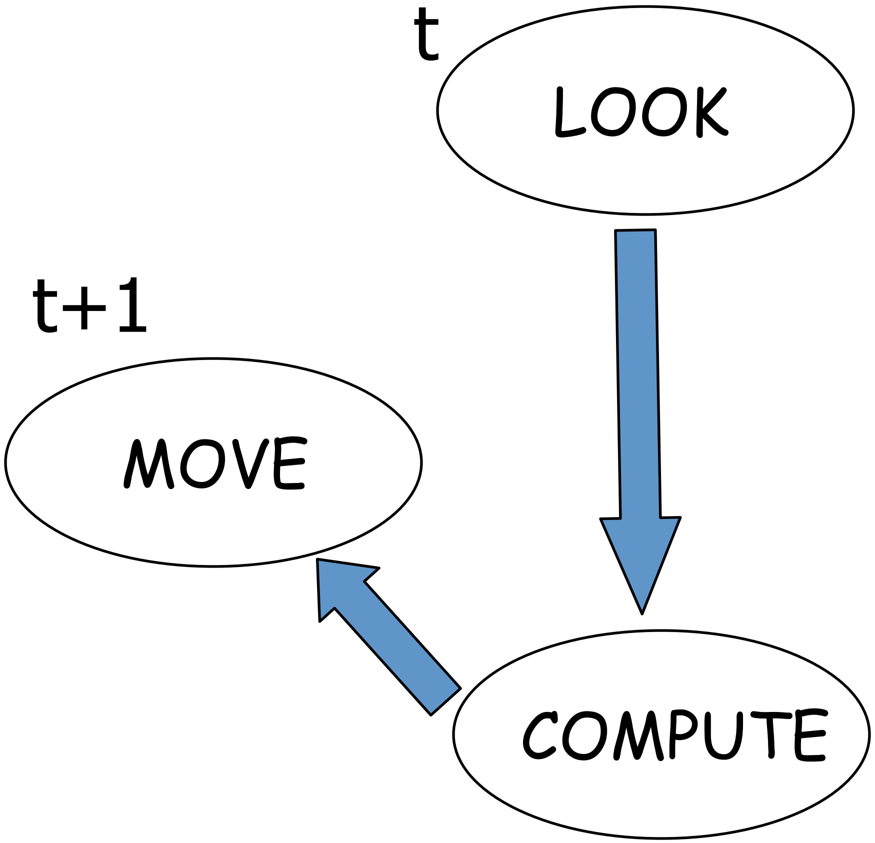
- Discrete Time 0,1,...

- 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 **SSync**

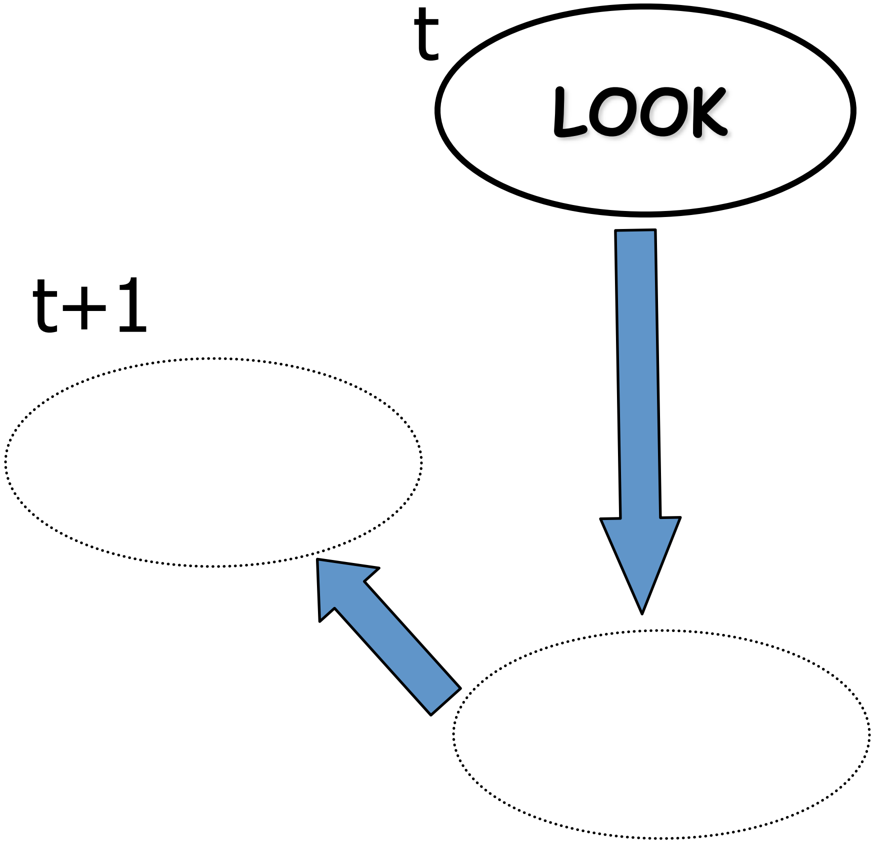
Suzuki *et al.*, 1996

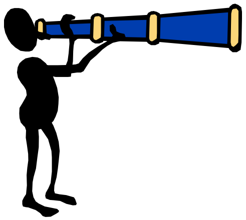


Phases of an Active robot

Instantaneous Actions of **SSync**

Suzuki *et al.*, 1996





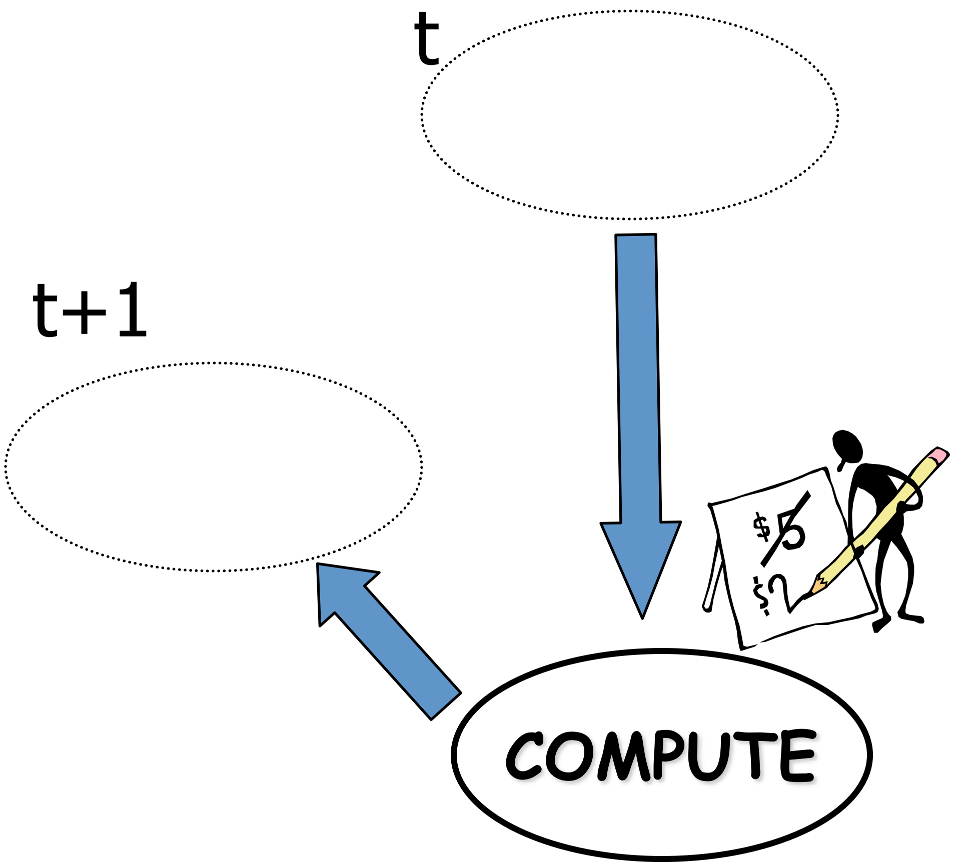
Uses its sensors
to observe the world.

result = **SNAPSHOT**
of the world

Visibility:
Unlimited
Limited

Instantaneous Actions of **SSync**

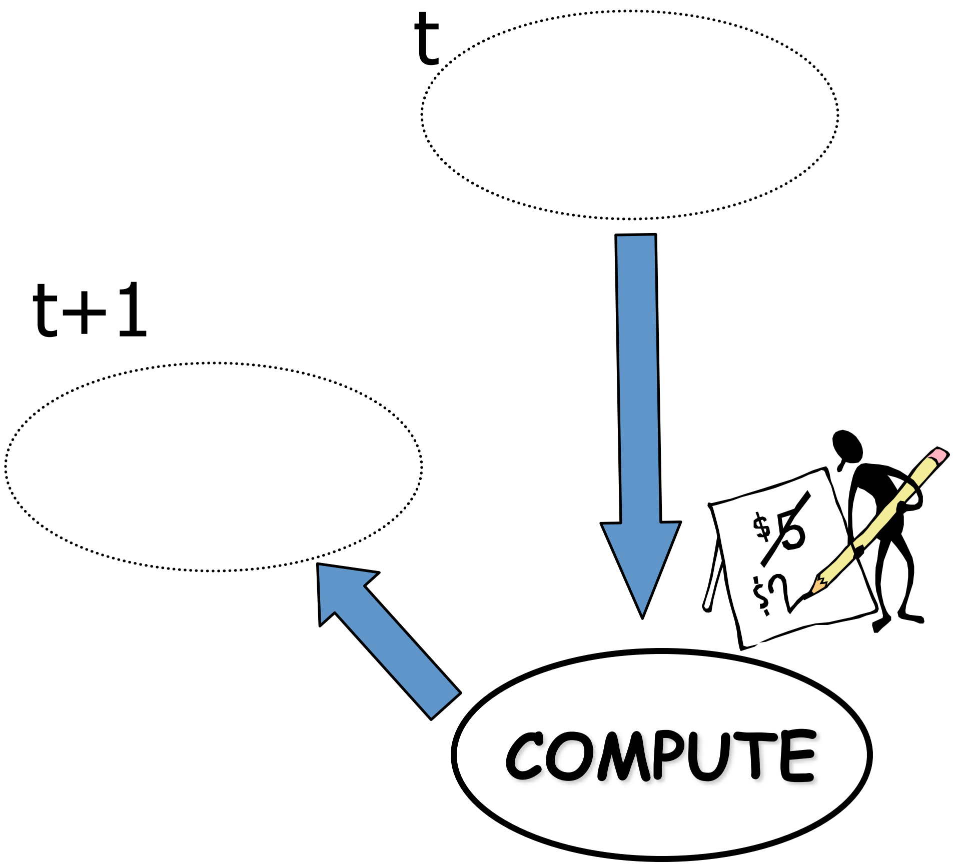
Suzuki *et al.*, 1996



Execute algorithm, ψ

Instantaneous Actions of **SSync**

Suzuki *et al.*, 1996



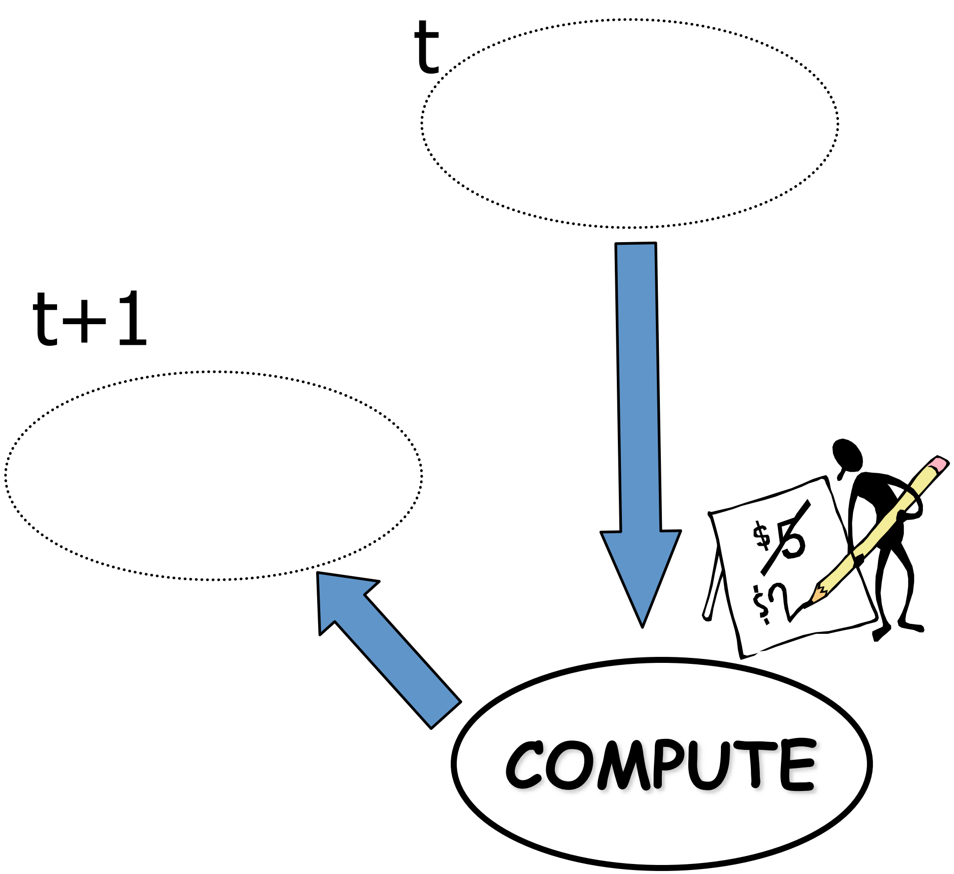
Execute algorithm, ψ

input = positions of the robots

result = destination point p

Instantaneous Actions of **SSync**

Suzuki *et al.*, 1996



t

t+1

COMPUTE

Execute algorithm, ψ

input = positions of the robots

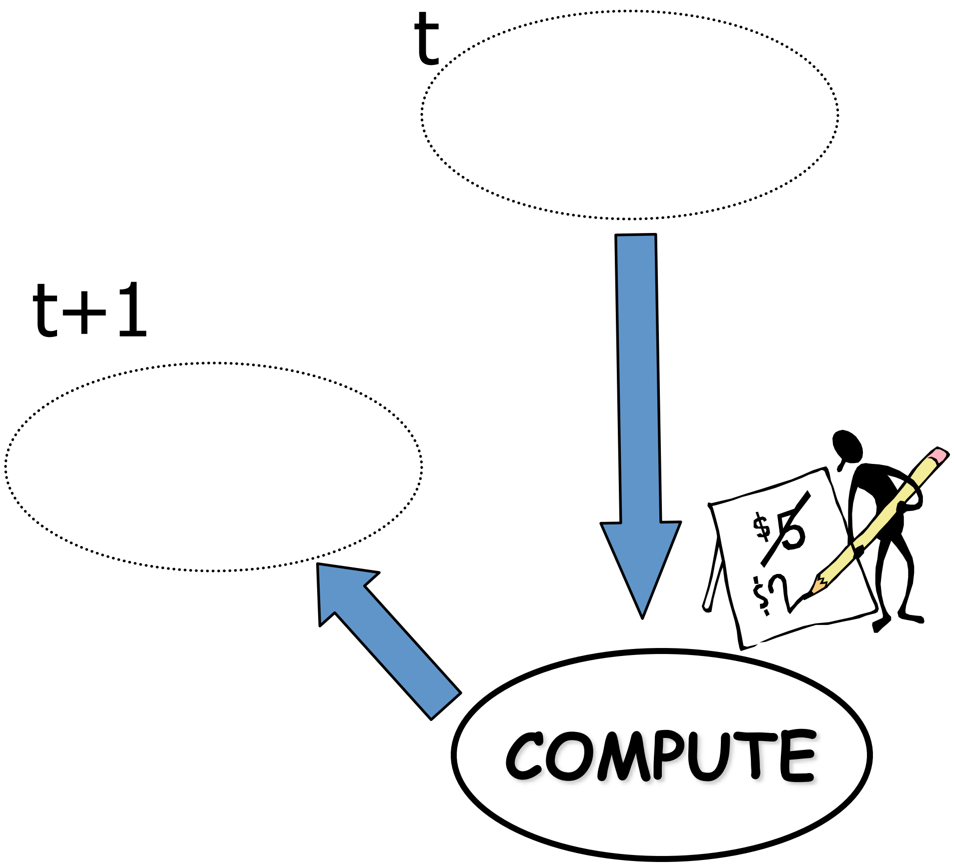
result = destination point p

Oblivious:

positions of the robots retrieved in the last Look

Instantaneous Actions of **SSync**

Suzuki *et al.*, 1996



Execute algorithm, ψ

input = positions of the robots

result = destination point p

Oblivious:

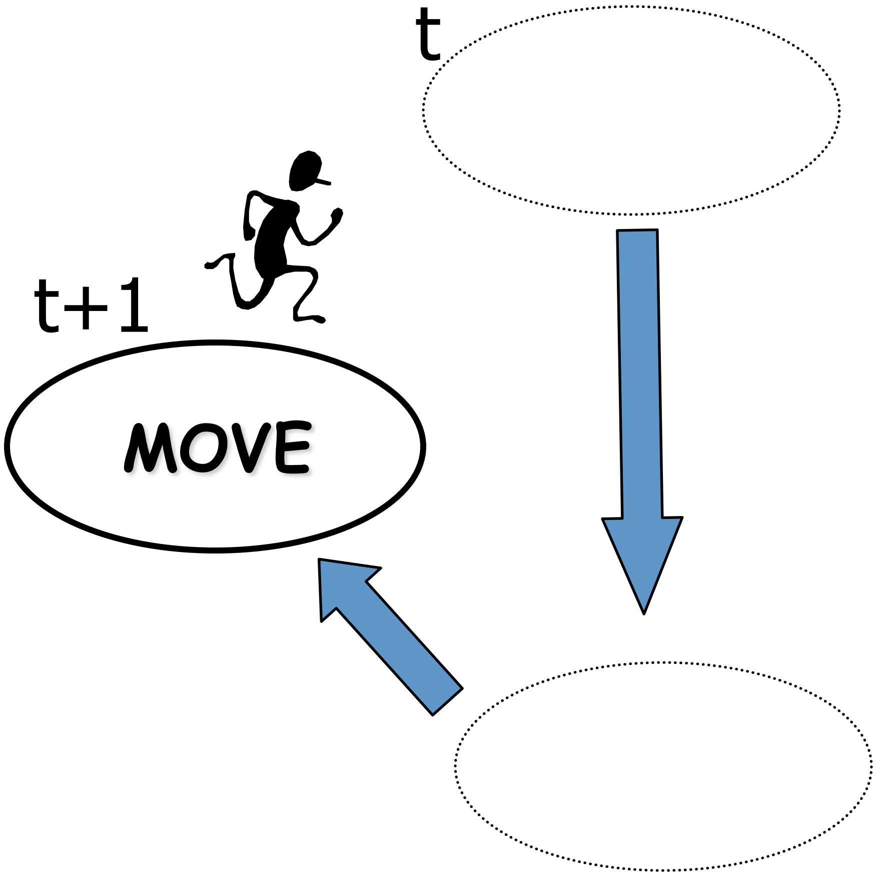
positions of the robots retrieved in the last Look

Non Oblivious:

positions of the robots since the beginning

Instantaneous Actions of **SSync**

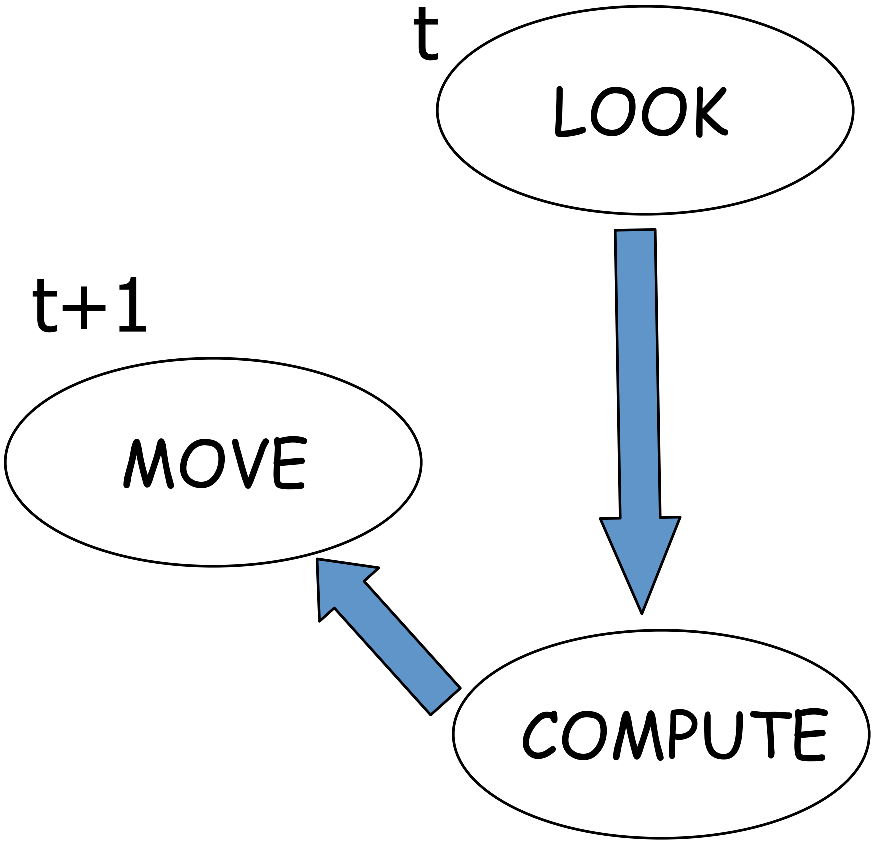
Suzuki *et al.*, 1996



The robot moves towards the computed destination

Instantaneous Actions of **SSync**

Suzuki *et al.*, 1996

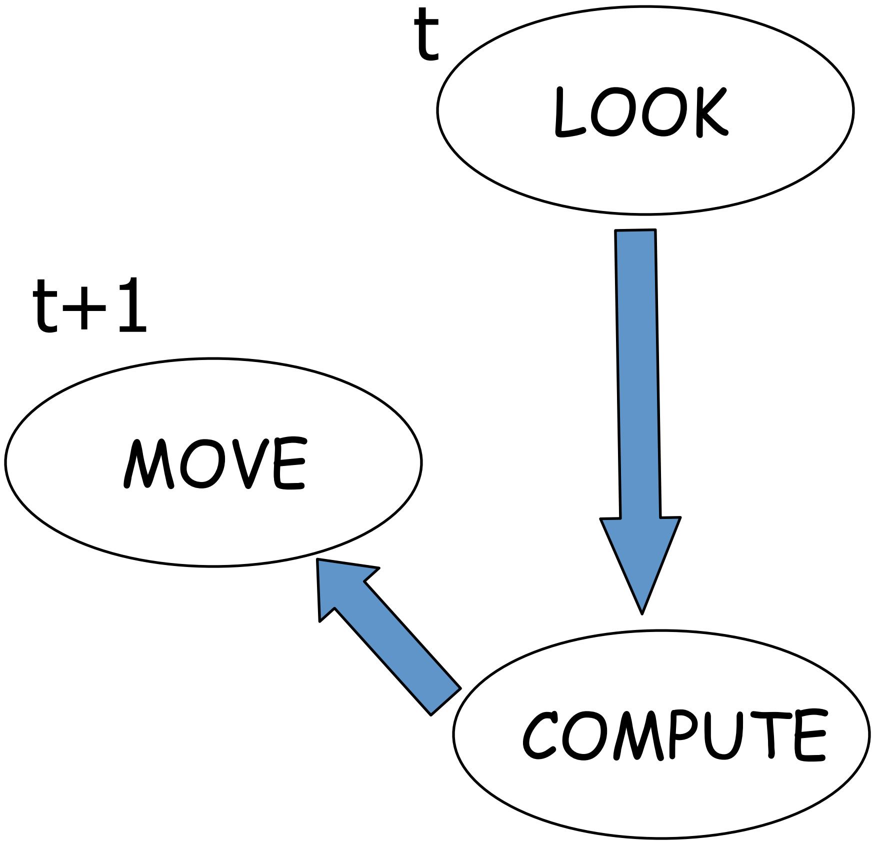


$p_i(t)$: Position of r_i at t

p: point returned by ψ

Instantaneous Actions of **SSync**

Suzuki *et al.*, 1996



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

p: point returned by ψ

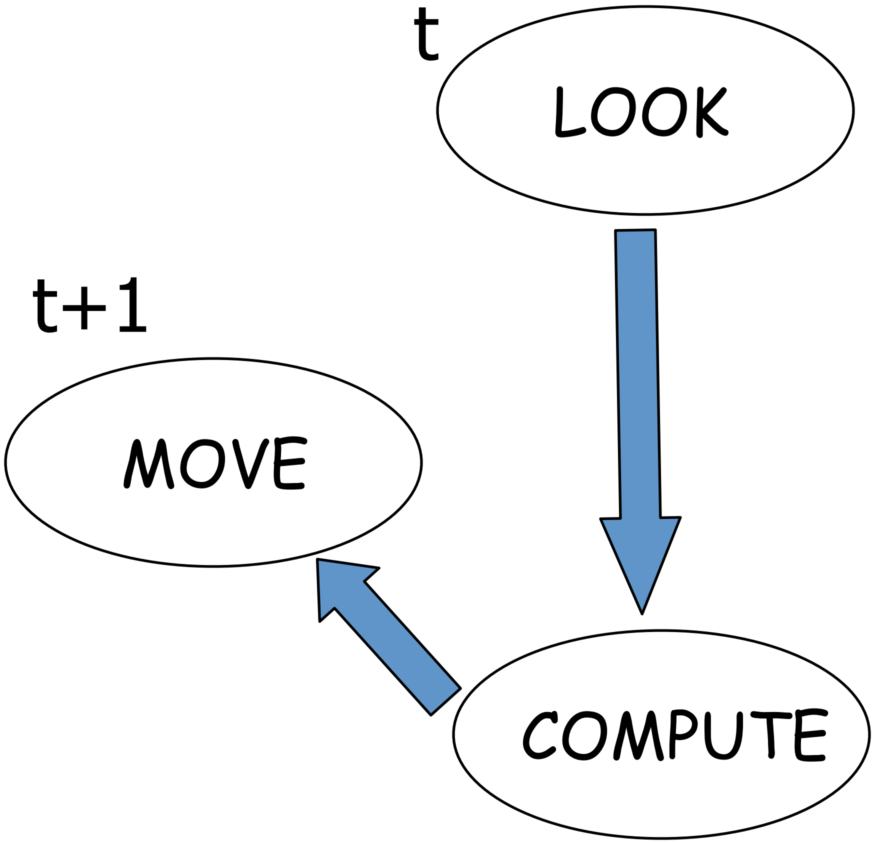
For all $t \geq 0$,

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

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

Instantaneous Actions of **SSync**

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

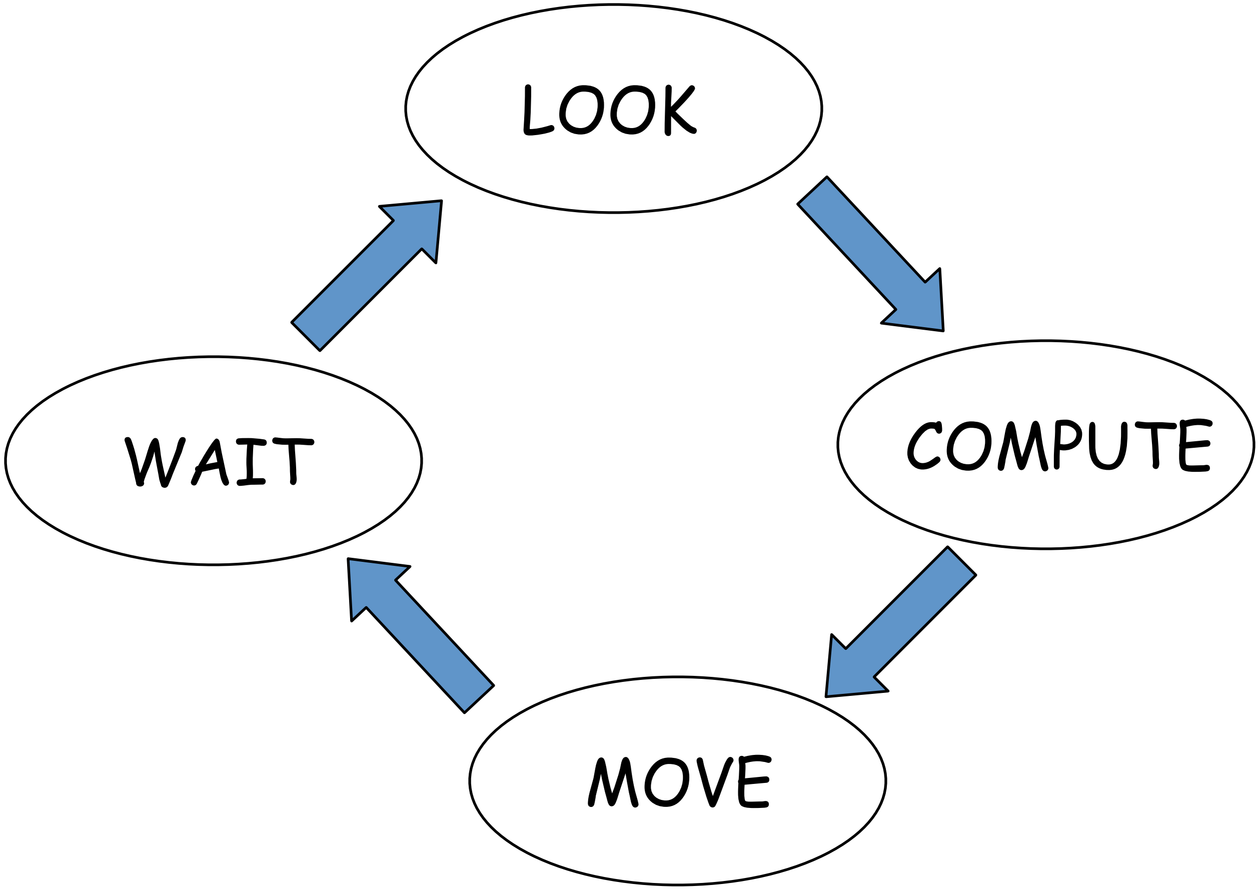
r_i executes L-C-M

atomically!

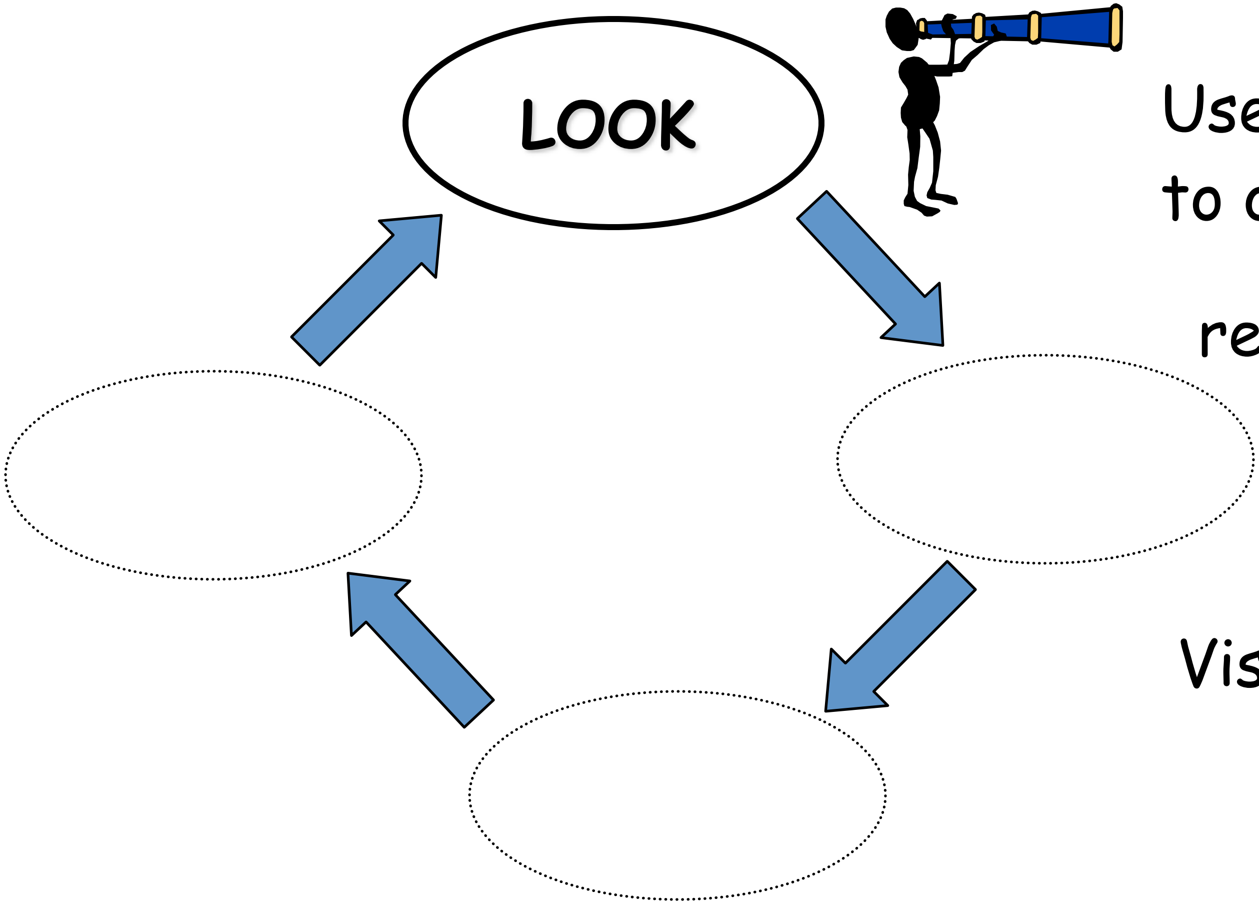
Asynchronicity

- In 1999 asynchronicity was introduced in the model
  - ASync (a.k.a. CORDA)

The Life Cycle (ASync)



The Life Cycle (ASync)

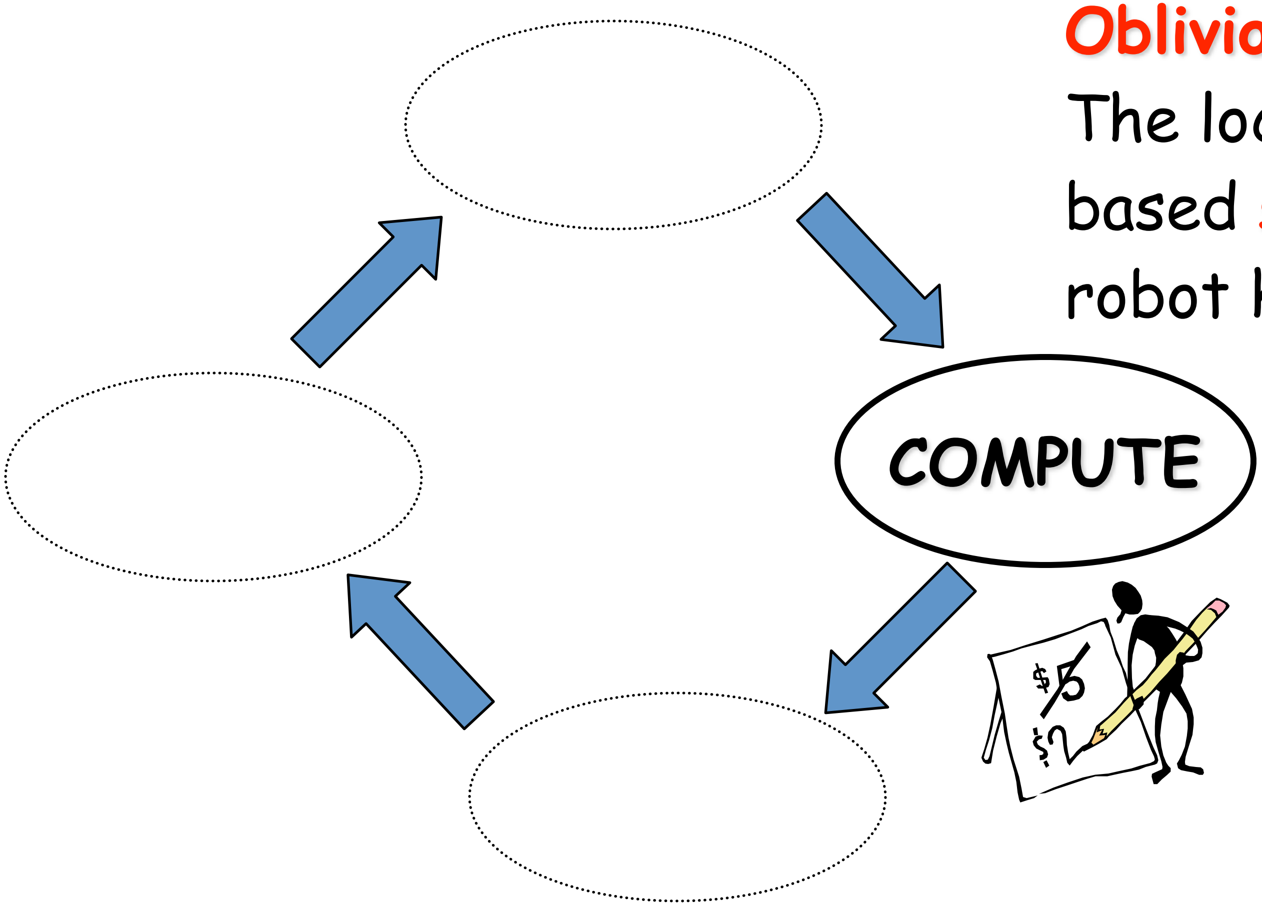


Uses its sensors
to observe the world.

result = **SNAPSHOT**
of the world

Visibility:
Unlimited
Limited

The Life Cycle (ASync)

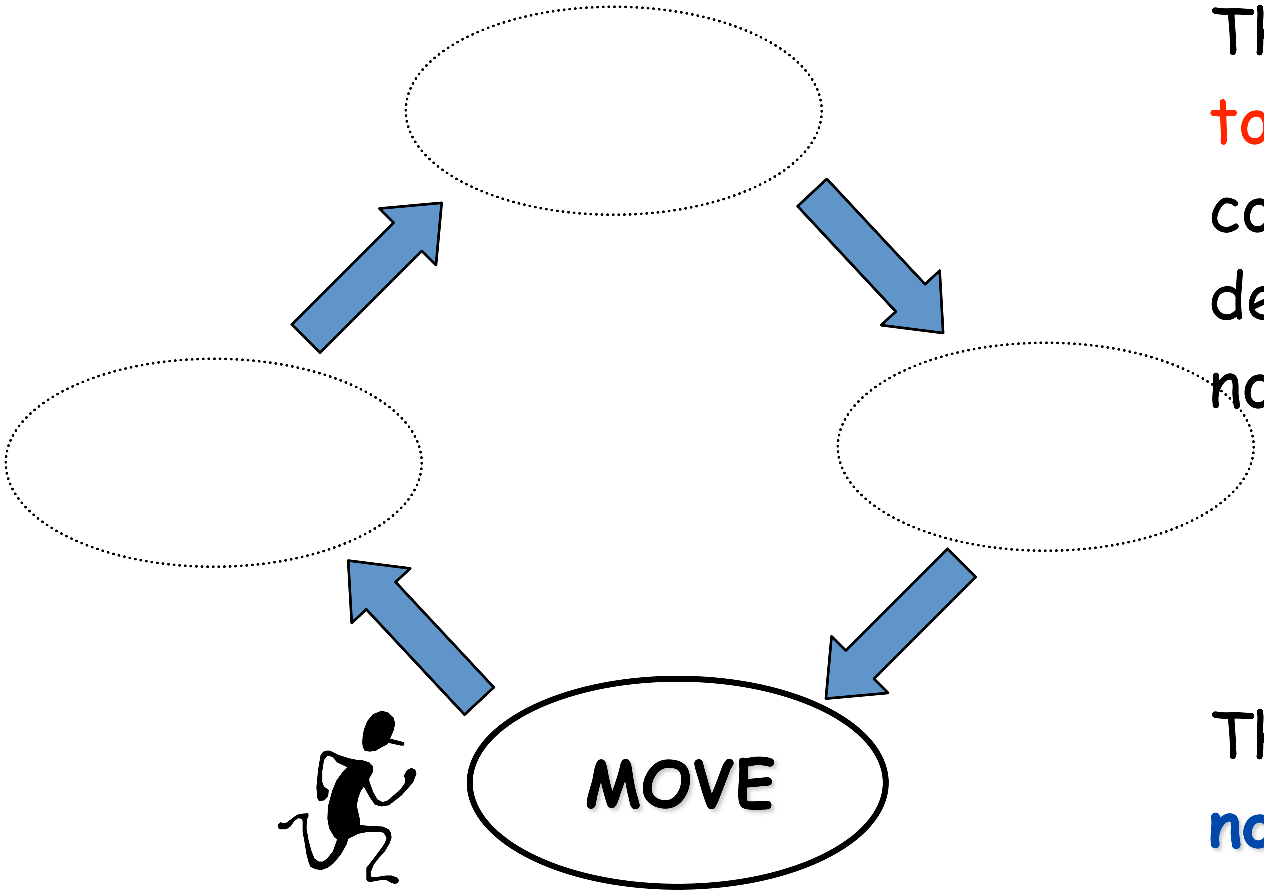


Obliviousness:
The local computation is based **solely** on what the robot has just observed

input = position of the other robots

result = destination point

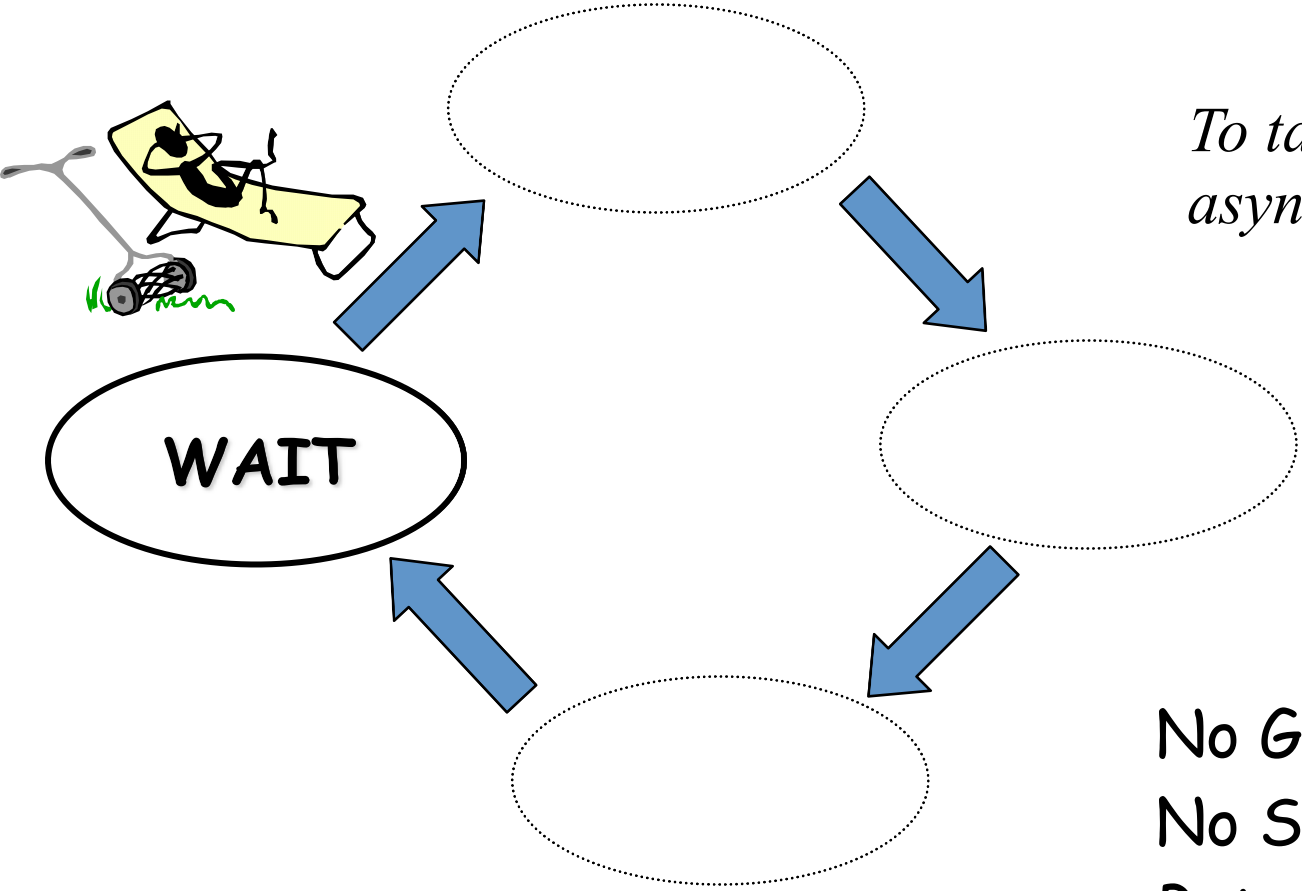
The Life Cycle (ASync)



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

The movement is **non-instantaneous**.

The Life Cycle (ASync)



To take into account asynchronous behavior.

No Global Clock
No Synchronization…
But cannot wait forever

SSync vs ASync

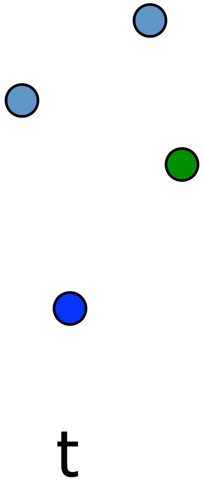
SSync

- **Instantaneous actions.**

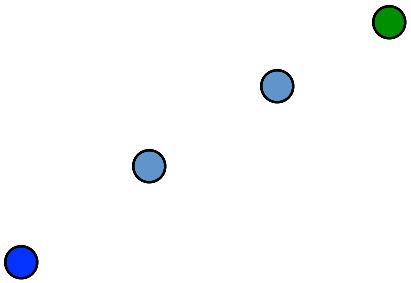
ASync

- **Full asynchronicity.**

Instantaneous Actions in Sync



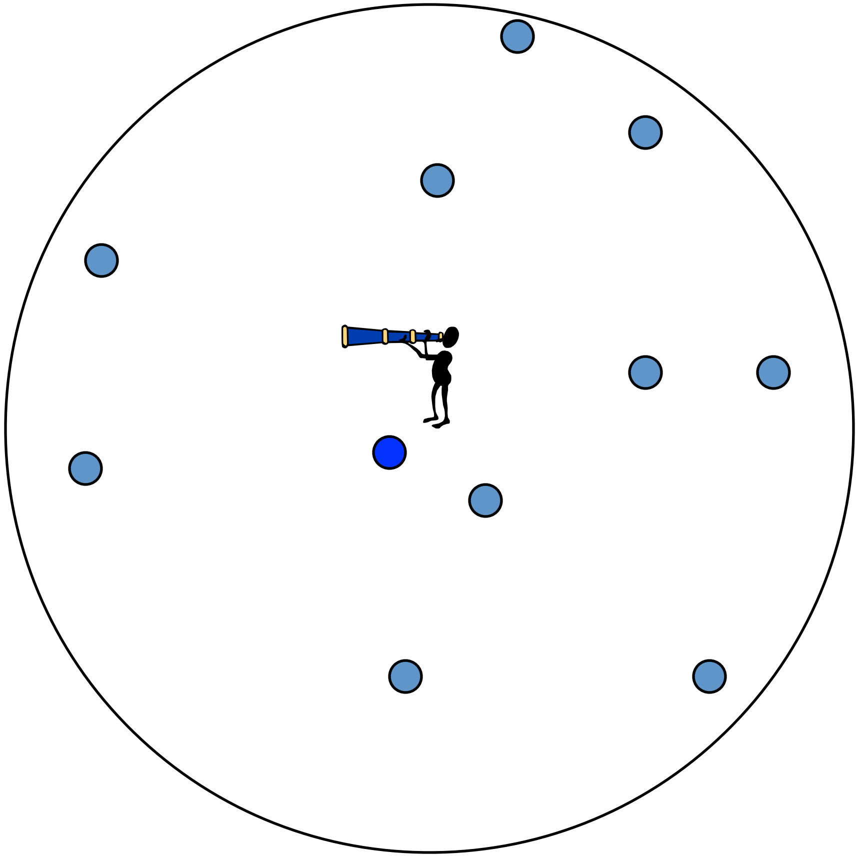
Instantaneous Actions in Sync



t+1

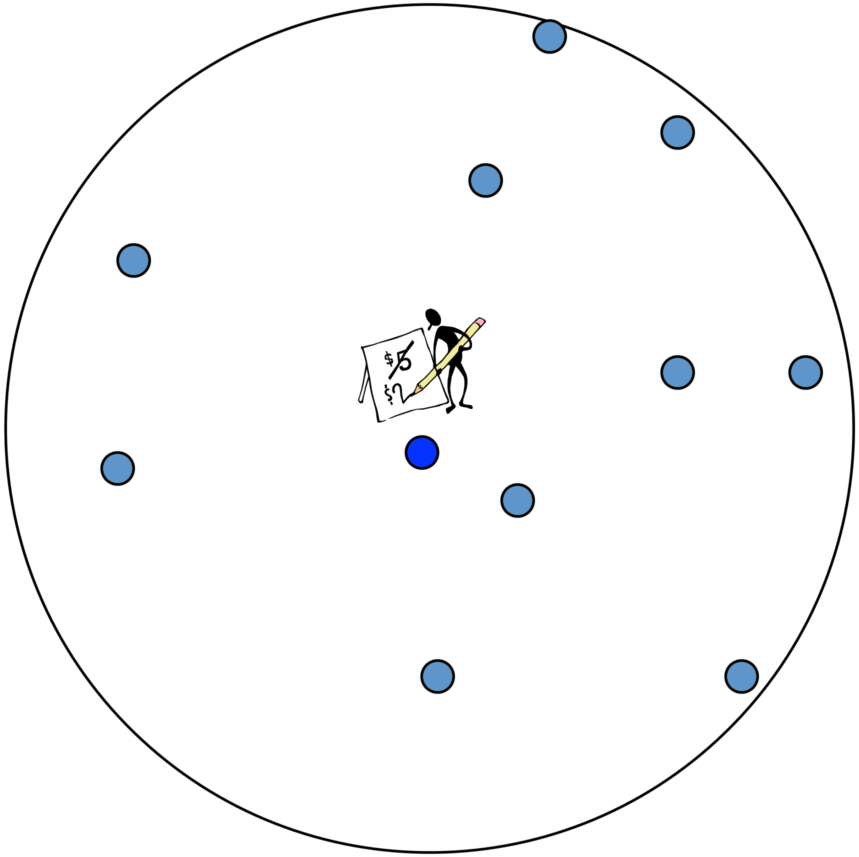
Asynchronicity in ASync

LOOK



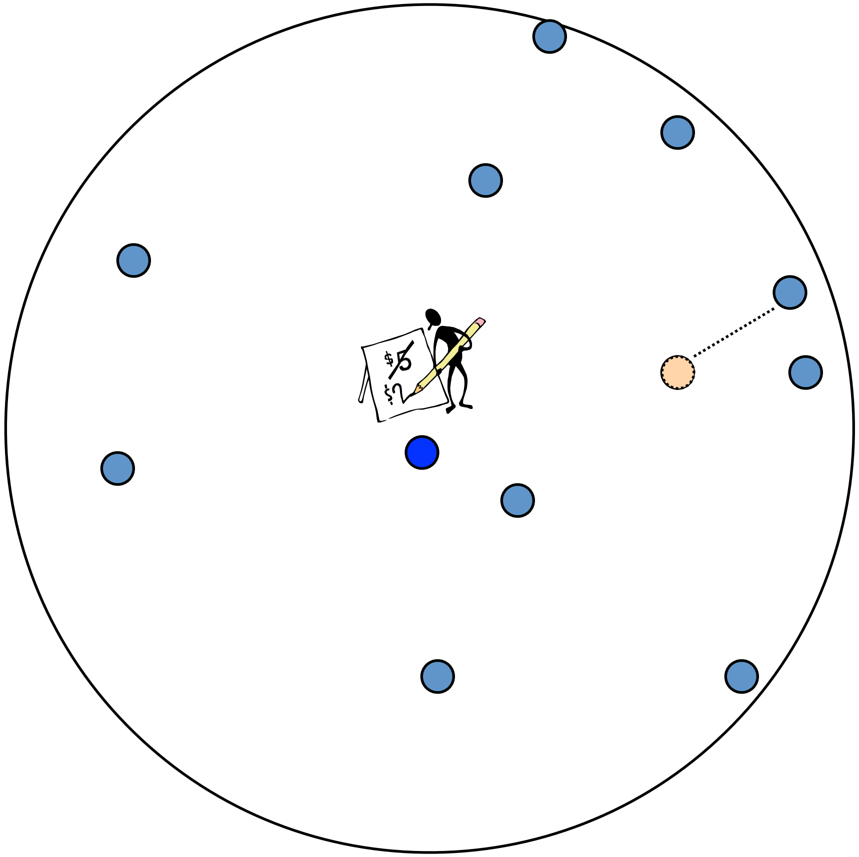
Asynchronicity in ASync

COMPUTE



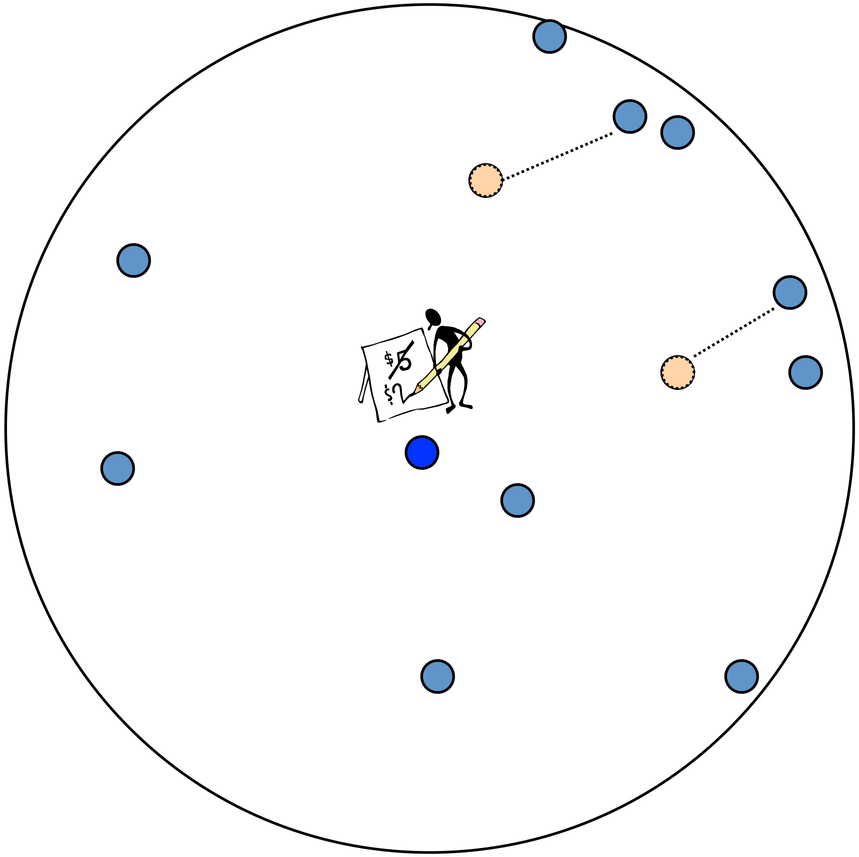
Asynchronicity in ASync

COMPUTE



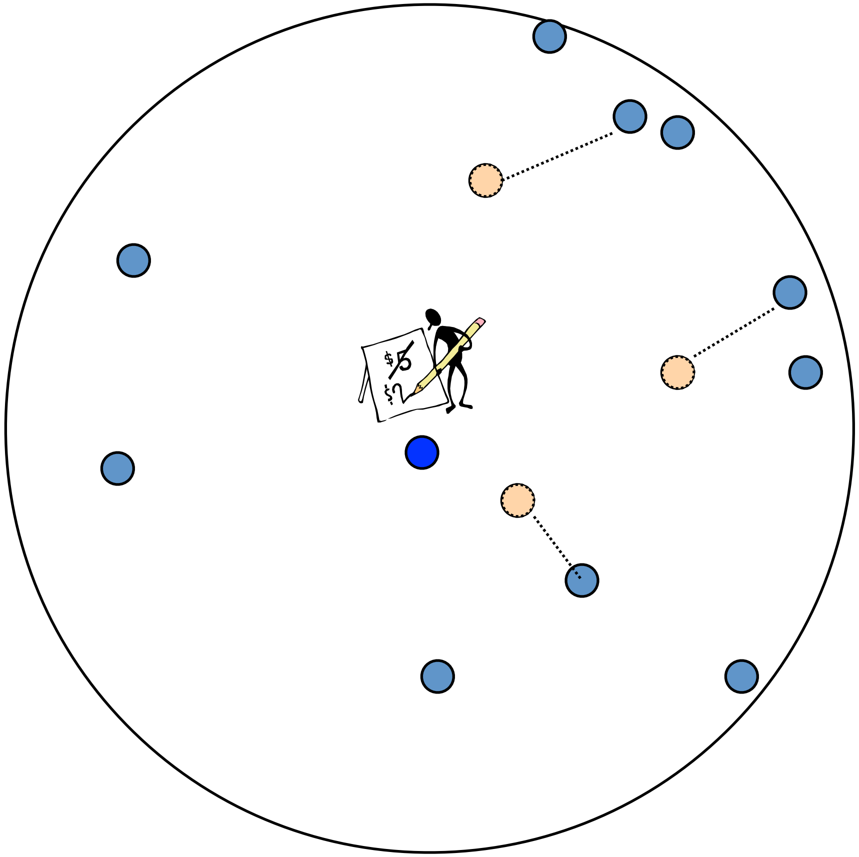
Asynchronicity in ASync

COMPUTE

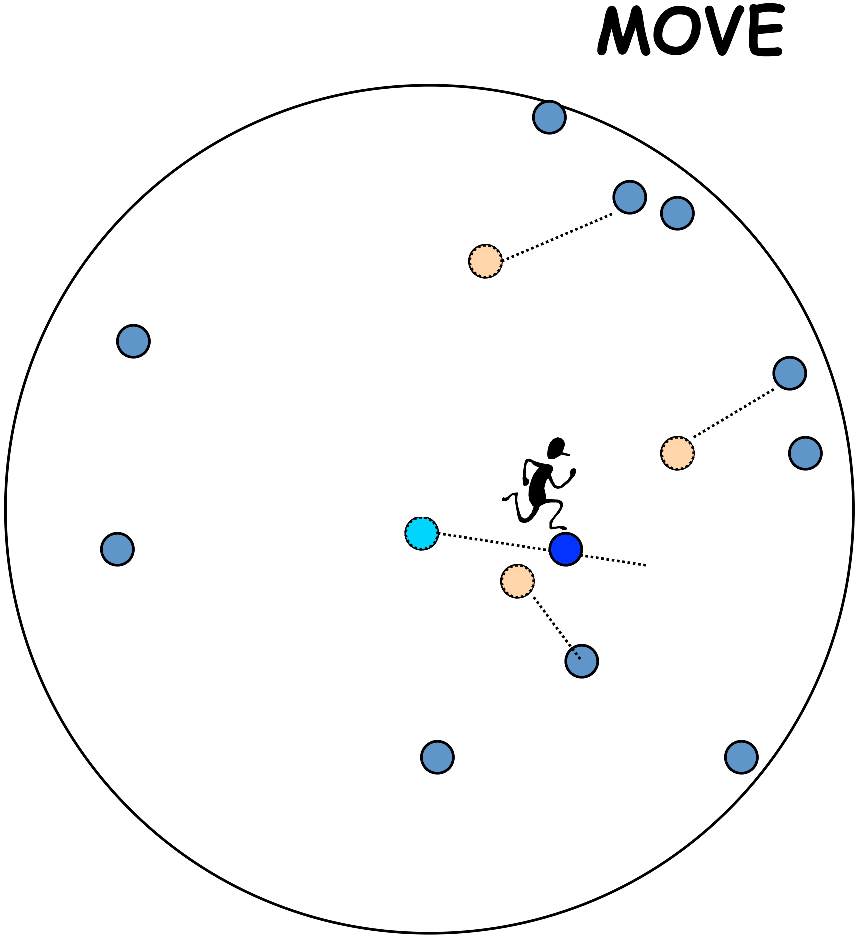


Asynchronicity in ASync

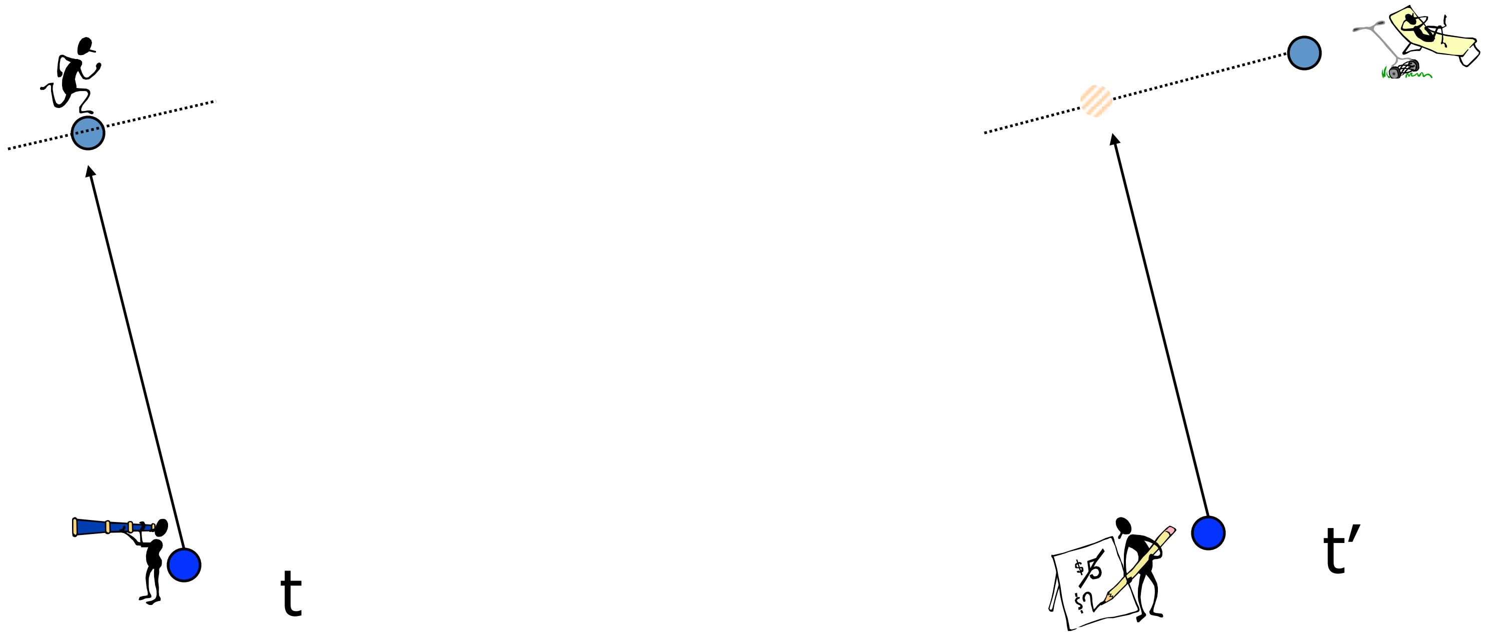
COMPUTE



Asynchronicity in ASync



Asynchronicity in ASync



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

Timing Models

ASYNC (CORDA) -*Fully asynchronous*

[Flocchini et. Al, 1999]

Arbitrary & varying operation rates and delays

SSYNC (*SYm*) - *Semi-synchronous*

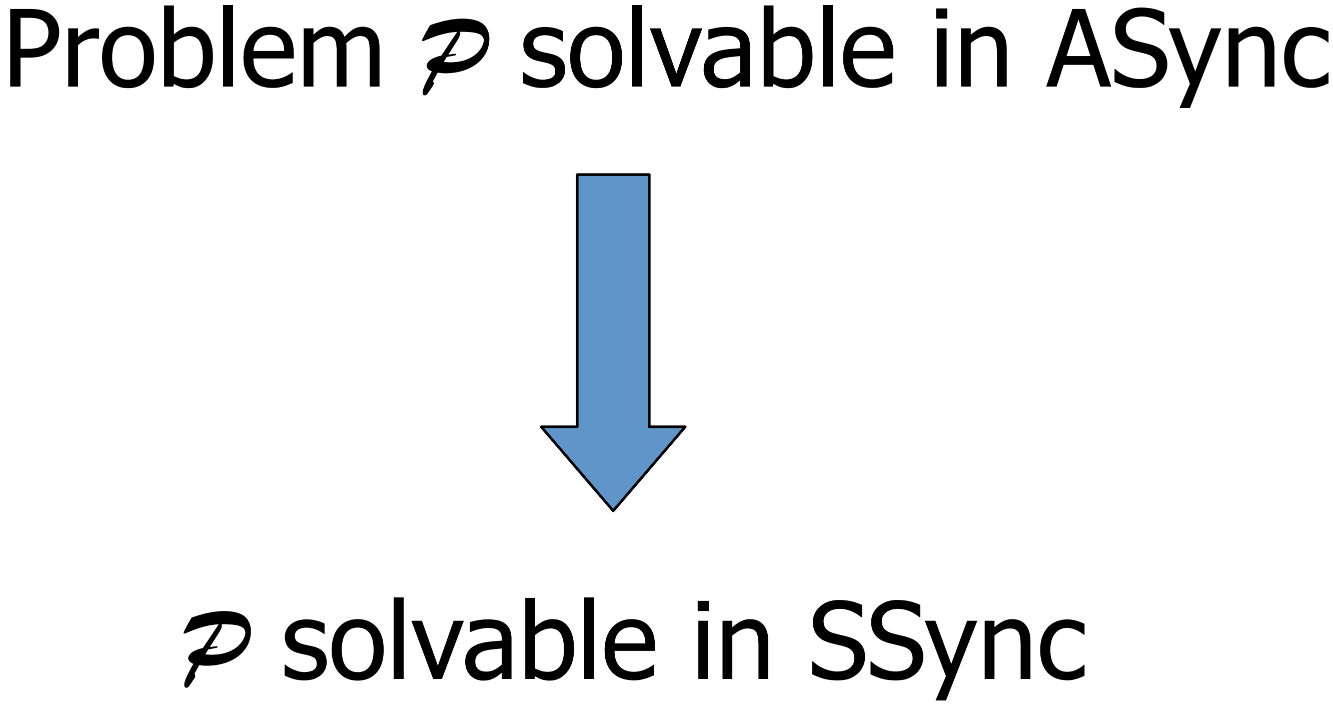
[Suzuki+Yamashita, 1996]

Fixed time cycles, but robots may be active / inactive

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

Fixed time cycles, all robots active in every cycle

ASync *vs*. SSync

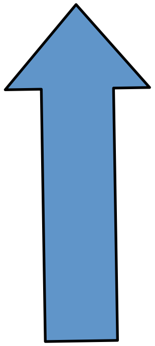


Problem $\mathcal{P}$ solvable in ASync

$\mathcal{P}$ solvable in SSync

ASync *vs.* SSync

Problem $\mathcal{P}$ unsolvable in ASync



$\mathcal{P}$ unsolvable in SSync