Deterministic Algorithms for the Rendezvous problem in Networks

Euripides Markou

Department of Computer Science & Biomedical Informatics,
University of Central Greece, Lamia, Greece.

Moving and Computing, August, 2010,
Carleton University, Ottawa, Canada.

Outline

The Rendezvous Problem

Problem

How should two `mobile agents` move along the nodes of a network so as to ensure that they meet (`rendezvous`)?

Interesting questions

What is the 'weakest' possible condition which makes rendezvous possible?

- power (e.g. agents leaving messages at nodes, or having tokens)
- memory (e.g. for counting)
- knowledge (e.g. number of nodes in the network)

The Rendezvous Problem

Problem

How should two `mobile agents` move along the nodes of a network so as to ensure that they meet (`rendezvous`)?

Interesting questions

What is the 'weakest' possible condition which makes rendezvous possible?

- power (e.g. agents leaving messages at nodes, or having tokens)
- memory (e.g. for counting)
- knowledge (e.g. number of nodes in the network)

The Rendezvous Problem

Problem

How should two `mobile agents` move along the nodes of a network so as to ensure that they meet (`rendezvous`)?

Interesting questions

What is the 'weakest' possible condition which makes rendezvous possible?

- power (e.g. agents leaving messages at nodes, or having tokens)
- memory (e.g. for counting)
- knowledge (e.g. number of nodes in the network)

The Rendezvous Problem

Problem

How should two `mobile agents` move along the nodes of a network so as to ensure that they meet (`rendezvous`)?

Interesting questions

What is the 'weakest' possible condition which makes rendezvous possible?

- power (e.g. agents leaving messages at nodes, or having tokens)
- memory (e.g. for counting)
- knowledge (e.g. number of nodes in the network)

Outline

Strong Models...

[Yu, Yung, 1996]

Unsolvable in an arbitrary graph if the agents use the same deterministic algorithm.

[Baston, Gal, 2001]

Randomized algorithms or different deterministic algorithms.

[Kowalski, Pelc, 2004], [De Marco et al, 2005], [Dessmark et al, 2006]

Symmetry was broken by assuming that robots have distinct labels.

...and Weaker Models.

[Barriere et al, 2003], [Dobrev et al, 2004]

Anonymous agents leaving messages at nodes.

[Kranakis et al, 2003], [Sawchuk, 2004], [Gasieniec et al, 2006], [Czyzowicz et al, 2008

Anonymous agents using tokens in a ring.

[Flocchini et al, 2004]

Multiple agents in a ring.

Outline

Our model (I) [Kranakis, Krizanc, Markou, LATIN06]

Two identical mobile agents

1. placed in an $n \times m$ `anonymous, synchronous and oriented torus.`
2. running the same `deterministic` algorithm and carrying `indistinguishable tokens.`
3. have no knowledge about the size of the torus.
4. know the number of tokens they have.
5. start at the same time.

Our model (I) [Kranakis, Krizanc, Markou, LATIN06]

Two identical mobile agents

1. placed in an $n \times m$ `anonymous, synchronous and oriented torus`.
2. running the same `deterministic` algorithm and carrying `indistinguishable tokens`.
3. have no knowledge about the size of the torus.
4. know the number of tokens they have.
5. start at the same time.

Our model (I) [Kranakis, Krizanc, Markou, LATIN06]

Two identical mobile agents

1. placed in an $n \times m$ `anonymous, synchronous and oriented torus`.
2. running the same `deterministic` algorithm and carrying `indistinguishable tokens`.
3. have no knowledge about the size of the torus.
4. know the number of tokens they have.
5. start at the same time.

Our model (I) [Kranakis, Krizanc, Markou, LATIN06]

Two identical mobile agents

1. placed in an $n \times m$ `anonymous, synchronous and oriented torus`.
2. running the same `deterministic` algorithm and carrying `indistinguishable tokens`.
3. have no knowledge about the size of the torus.
4. know the number of tokens they have.
5. start at the same time.

Our model (I) [Kranakis, Krizanc, Markou, LATIN06]

Two identical mobile agents

1. placed in an $n \times m$ `anonymous, synchronous and oriented torus`.
2. running the same `deterministic` algorithm and carrying `indistinguishable tokens`.
3. have no knowledge about the size of the torus.
4. know the number of tokens they have.
5. start at the same time.

Our model (II)

At any single time unit a mobile agent occupies a node v of the torus and may

- stay there or move to an adjacent node,
- detect the presence of one or more tokens at v,
- release/take one or more tokens to/from v.

Agent's power

Memory permitting, an agent can count the number of nodes between tokens.

Our model (II)

At any single time unit a mobile agent occupies a node v of the torus and may

- stay there or move to an adjacent node,
- detect the presence of one or more tokens at v,
- release/take one or more tokens to/from v.

Agent's power

Memory permitting, an agent can count the number of nodes between tokens.

Our model (II)

At any single time unit a mobile agent occupies a node v of the torus and may

- stay there or move to an adjacent node,
- detect the presence of one or more tokens at v,
- release/take one or more tokens to/from v.

Agent's power

Memory permitting, an agent can count the number of nodes between tokens.

Our model (II)

At any single time unit a mobile agent occupies a node v of the torus and may

- stay there or move to an adjacent node,
- detect the presence of one or more tokens at v,
- release/take one or more tokens to/from v.

Agent's power

Memory permitting, an agent can count the number of nodes between tokens.

Our model (II)

At any single time unit a mobile agent occupies a node v of the torus and may

- stay there or move to an adjacent node,
- detect the presence of one or more tokens at v,
- release/take one or more tokens to/from v.

Agent's power

Memory permitting, an agent can count the number of nodes between tokens.

Our Questions

Is it possible to construct a deterministic algorithm

which, after a finite time, leads the agents to rendezvous at a node or while crossing an edge?
(no matter what are their starting positions or the size of the torus).

What is the minimum

- time (edge traversals) needed for rendezvous?
- memory needed?
- number of tokens?

Trade-offs

Between memory and movable or unmovable tokens

Our Questions

Is it possible to construct a deterministic algorithm

which, after a finite time, leads the agents to rendezvous at a node or while crossing an edge?
(no matter what are their starting positions or the size of the torus).

What is the minimum

- time (edge traversals) needed for rendezvous?
- memory needed?
- number of tokens?

Trade-offs

Between memory and movable or unmovable tokens

Our Questions

Is it possible to construct a deterministic algorithm

which, after a finite time, leads the agents to rendezvous at a node or while crossing an edge?
(no matter what are their starting positions or the size of the torus).

What is the minimum

- time (edge traversals) needed for rendezvous?
- memory needed?
- number of tokens?

Trade-offs

Between memory and movable or unmovable tokens

Our Questions

Is it possible to construct a deterministic algorithm

which, after a finite time, leads the agents to rendezvous at a node or while crossing an edge?
(no matter what are their starting positions or the size of the torus).

What is the minimum

- time (edge traversals) needed for rendezvous?
- memory needed?
- number of tokens?

Trade-offs

Between memory and movable or unmovable tokens

Two Identical Agents Using Tokens in an Anonymous Oriented Synchronous Ring.

[Kranakis et al, 2003]

- One `unmovable token`: $\Omega(\log \log n)$ memory for rendezvous *with detection*.
- One `movable token`: Constant memory for rendezvous *without detection*.

[Gasieniec et al, 2006]

- One `movable token`: $\Omega(\log \log n)$ memory for rendezvous *with detection*.
- Two `movable tokens`: Constant memory for rendezvous *with detection*.

Rendezvous With Detection vs Rendezvous Without Detection.

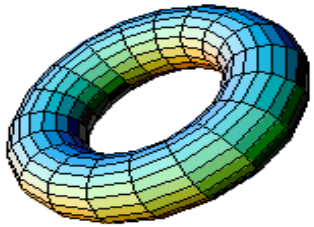
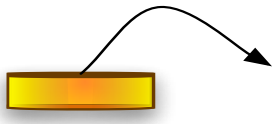
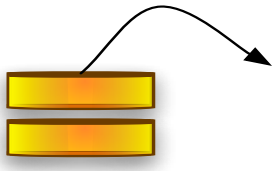
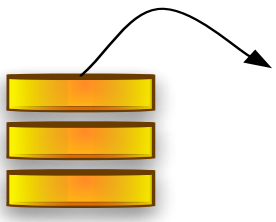
Lemma [KKM]

Let $(0, 0)$ be the initial position of agent A on the torus. If B's initial position is either $(n/2, 0)$ or $(0, m/2)$ or $(n/2, m/2)$ then the agents are incapable of meeting each other (no matter how many movable tokens, or how much memory they have).

Definitions

- Rendezvous Without Detection (RV): the agents meet each other if their initial distance is other than above,
- Rendezvous With Detection (RVD): otherwise they stop and declare that rendezvous is impossible.

Ring vs Torus

	 RVD requires Θ(log log n)	 RVD requires Θ(log n)
	RV in constant mem	RVD requires Θ(log n)
	RVD in constant mem	n x n: RVD in constant mem n x m: RV in constant mem
		RVD in constant mem

Memory Lower Bounds for Rendezvous. [KKM]

Two agents in a $n \times n$ torus:

- With a constant number of unmovable tokens need at least $\Omega(\log n)$ memory, each.
- With one movable token need at least $\Omega(\log n)$ memory, each.

Rendezvous Algorithms in a $n \times m$ Torus. [KKM]

One unmovable token and $O(\log n + \log m)$ memory

RVD in $O(n \cdot m)$ time units.

Two movable tokens and constant memory

- RVD in a $n \times n$ torus after $O(n^2)$ time units,
- RV in a $n \times m$ torus after $O(n^2 + m^2)$ time units; (can be adapted to RVD if n, m have a specific relation).

Three movable tokens and constant memory

RVD in $O(n^2 + m^2)$ time units.

Rendezvous Algorithms in a $n \times m$ Torus. [KKM]

One unmovable token and $O(\log n + \log m)$ memory

RVD in $O(n \cdot m)$ time units.

Two movable tokens and constant memory

- RVD in a $n \times n$ torus after $O(n^2)$ time units,
- RV in a $n \times m$ torus after $O(n^2 + m^2)$ time units; (can be adapted to RVD if n, m have a specific relation).

Three movable tokens and constant memory

RVD in $O(n^2 + m^2)$ time units.

Rendezvous Algorithms in a $n \times m$ Torus. [KKM]

One unmovable token and $O(\log n + \log m)$ memory

RVD in $O(n \cdot m)$ time units.

Two movable tokens and constant memory

- RVD in a $n \times n$ torus after $O(n^2)$ time units,
- RV in a $n \times m$ torus after $O(n^2 + m^2)$ time units; (can be adapted to RVD if n, m have a specific relation).

Three movable tokens and constant memory

RVD in $O(n^2 + m^2)$ time units.

Outline

One Agent in a $n \times n$ Torus.

Lemma

An agent without any tokens needs at least $\Omega(\log n)$ memory to visit all nodes of the torus.

Idea: After at most n repetitions of the first repeated state the agent does not visit new nodes.

Lemma

An agent with a constant number of unmovable tokens needs at least $\Omega(\log n)$ memory to visit all nodes of the torus.

One Agent in a $n \times n$ Torus.

Lemma

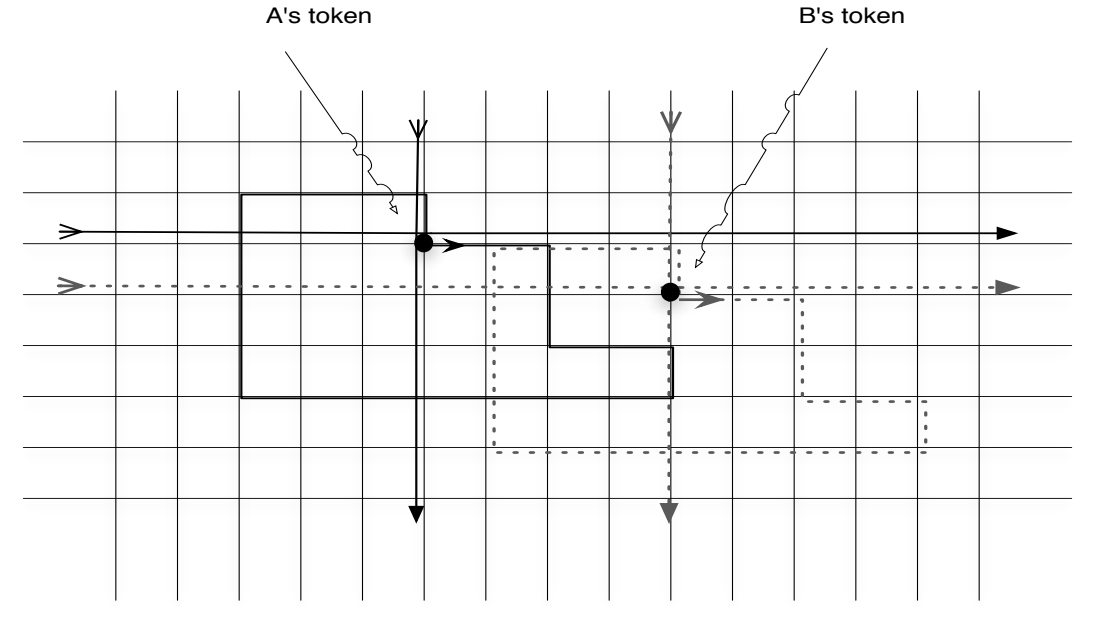
An agent without any tokens needs at least $\Omega(\log n)$ memory to visit all nodes of the torus.

Idea: After at most n repetitions of the first repeated state the agent does not visit new nodes.

Lemma

An agent with a constant number of unmovable tokens needs at least $\Omega(\log n)$ memory to visit all nodes of the torus.

Two Agents with one Unmovable Token need at least $\Omega(\log n)$ Memory to Rendezvous.



Idea: After placing agent A, place agent B so that:

- B releases its token at a node not visited by agent A and
- B avoids visiting the node where A's token lies.

Constant number of Unmovable Tokens vs one Movable Token.

Lemma

Two agents with a constant number of unmovable tokens need at least $\Omega(\log n)$ memory to rendezvous.

Lemma

Two agents with one movable token each need at least $\Omega(\log n)$ memory to rendezvous.

Idea: place the agents so that in any phase which starts when the agents move their tokens, up to the moment that they move their tokens again they do not meet each other's token.

Constant number of Unmovable Tokens vs one Movable Token.

Lemma

Two agents with a constant number of unmovable tokens need at least $\Omega(\log n)$ memory to rendezvous.

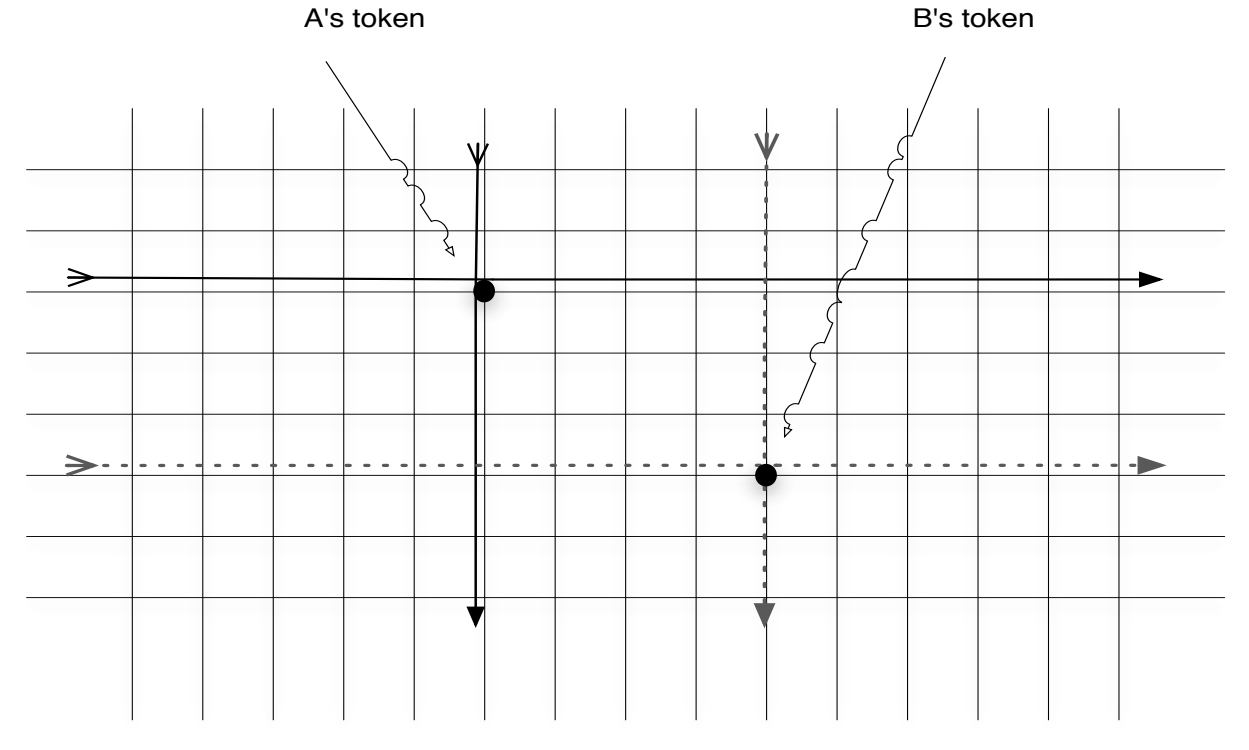
Lemma

Two agents with one movable token each need at least $\Omega(\log n)$ memory to rendezvous.

Idea: place the agents so that in any phase which starts when the agents move their tokens, up to the moment that they move their tokens again they do not meet each other's token.

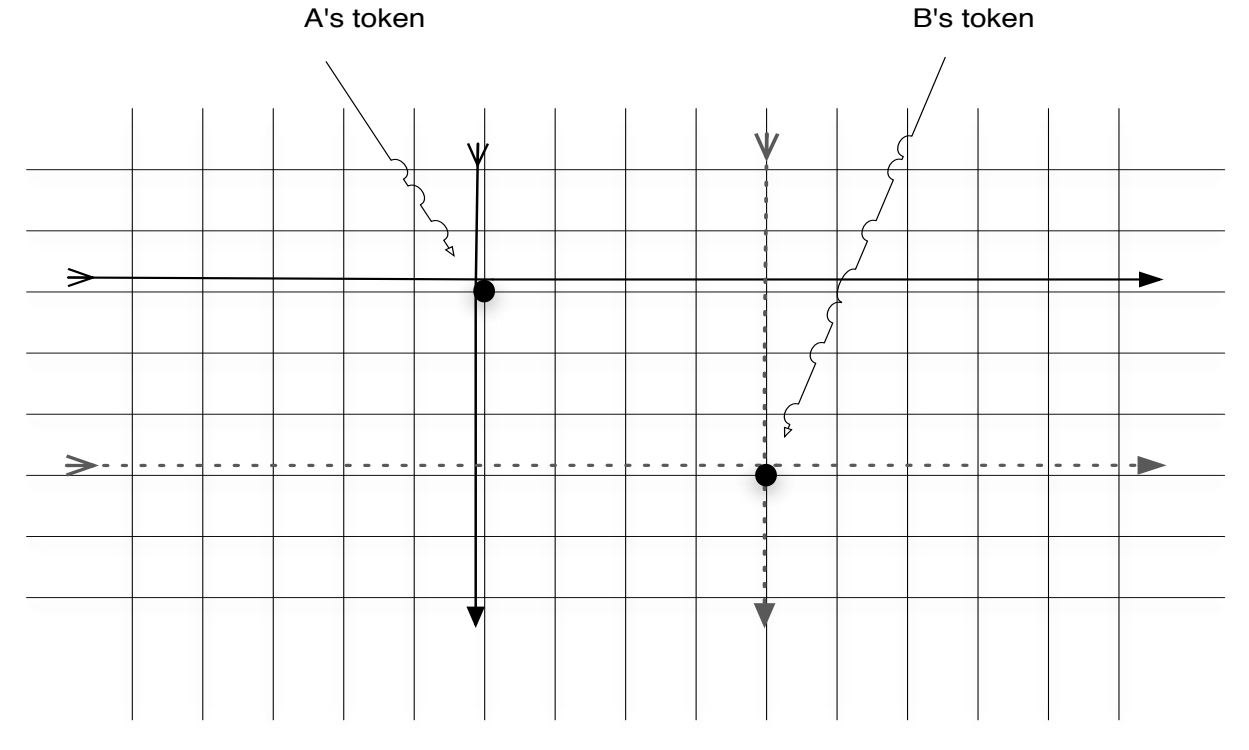
Outline

RVD in a $n \times m$ Torus with one Unmovable Token and $O(\log n + \log m)$ Memory in $O(n \cdot m)$ steps.



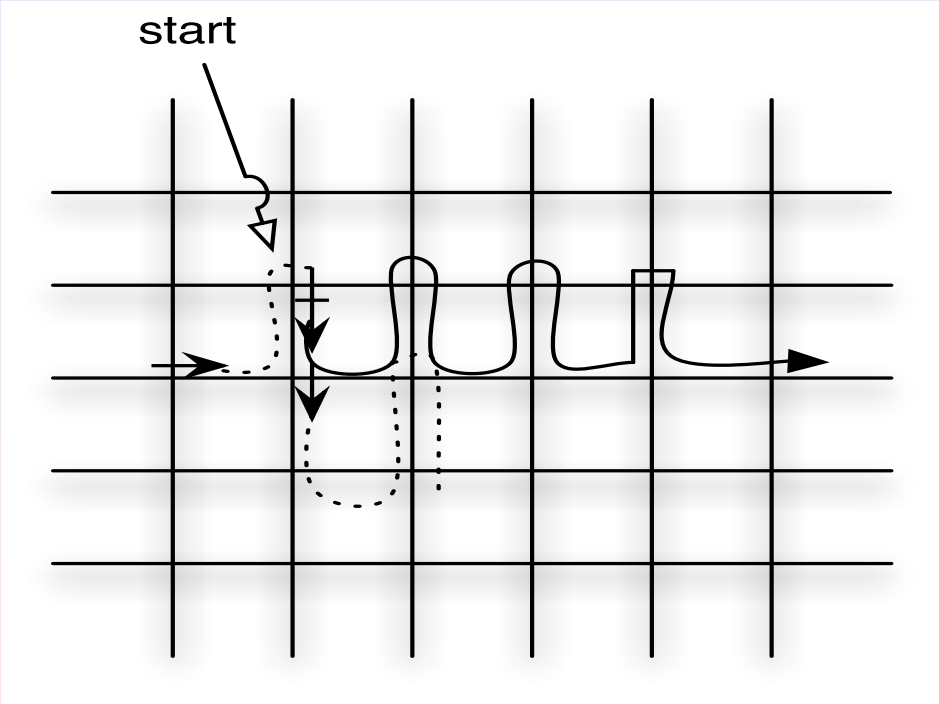
- Same-Ring: count horizontally; count vertically.
- Different-Ring: search one by one the horizontal rings.

RVD in a $n \times m$ Torus with one Unmovable Token and $O(\log n + \log m)$ Memory in $O(n \cdot m)$ steps.



- Same-Ring: count horizontally; count vertically.
- Different-Ring: search one by one the horizontal rings.

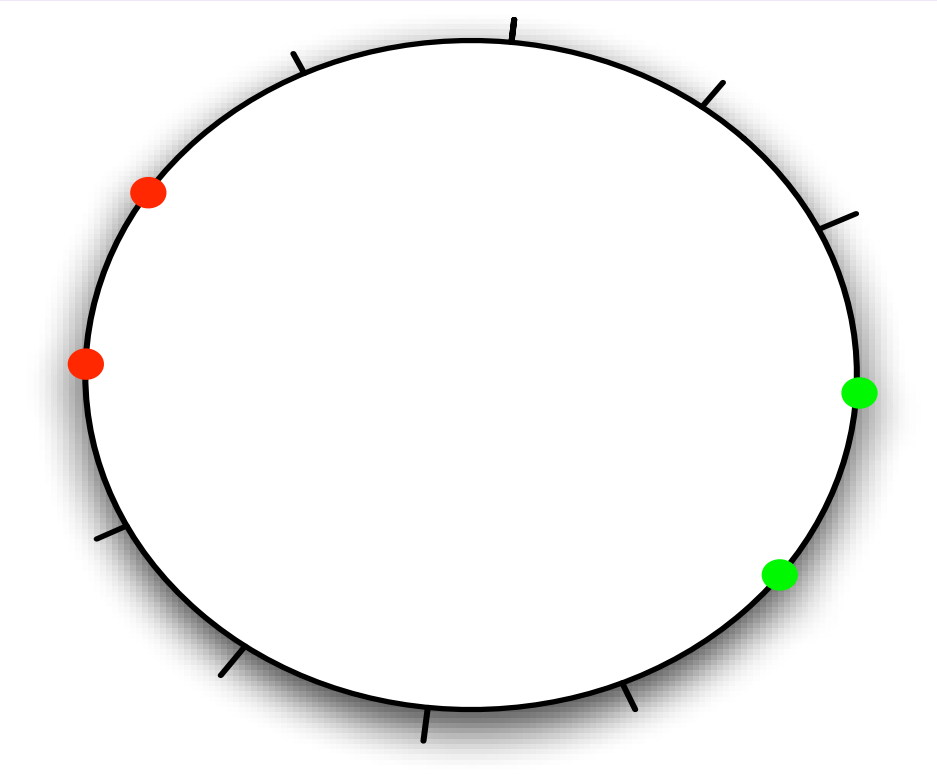
RVD in a $n \times n$ Torus with two Movable Tokens and Constant Memory in $O(n^2)$ steps (I).



1. Scan Horizontally

Decide whether you have started on the same ring or not.

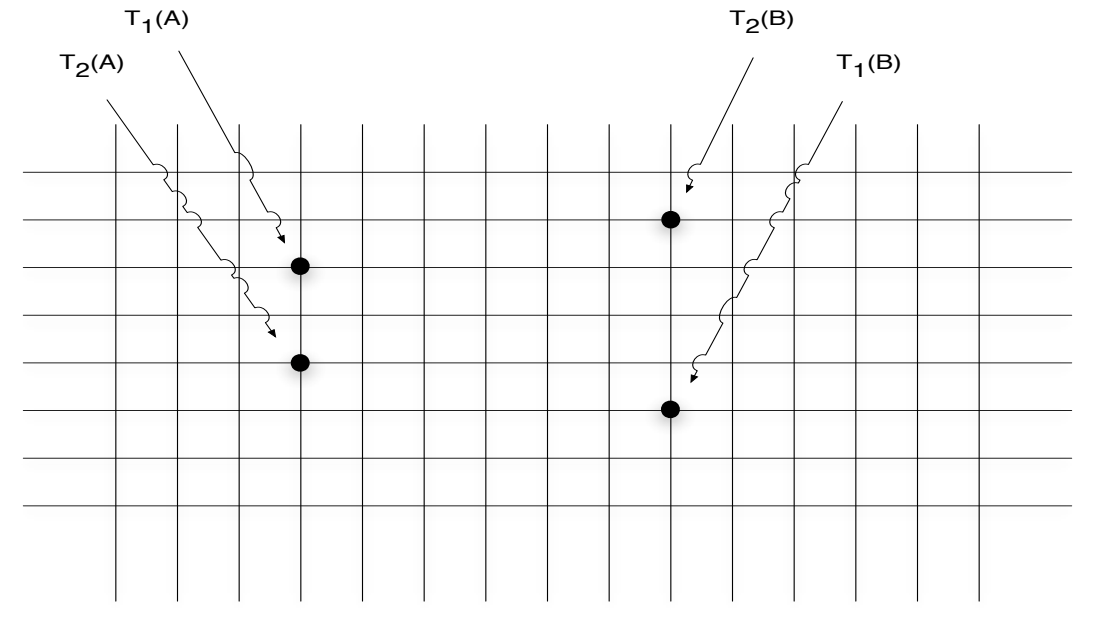
RVD in a $n \times n$ Torus with two Movable Tokens and Constant Memory $O(n^2)$ steps (II).



2. RVD in a Ring

If you have started in the same ring then RVD in the ring.

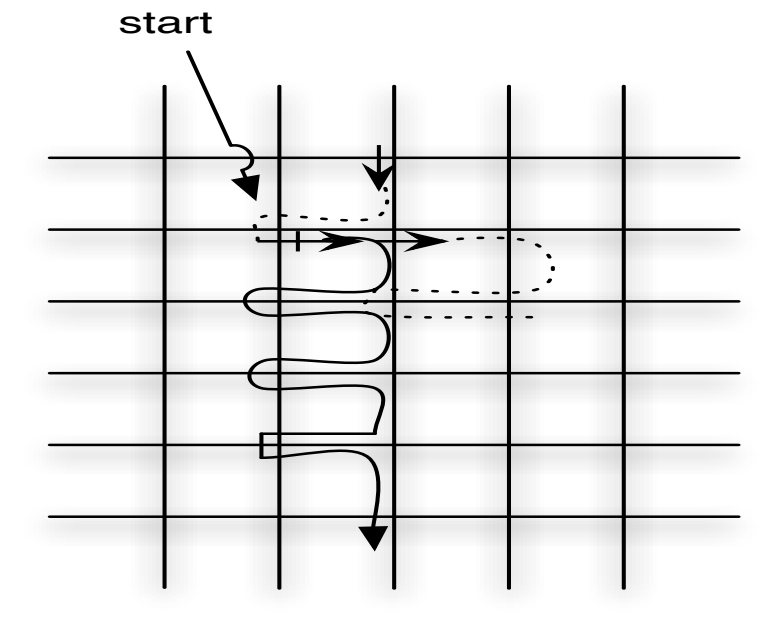
RVD in a $n \times n$ Torus with two Movable Tokens and Constant Memory $O(n^2)$ steps (III).



3. Mark a path

Mark a path with your tokens and try to 'catch' the other.
Not succeeded? then it should hold $d_y = n/2$.

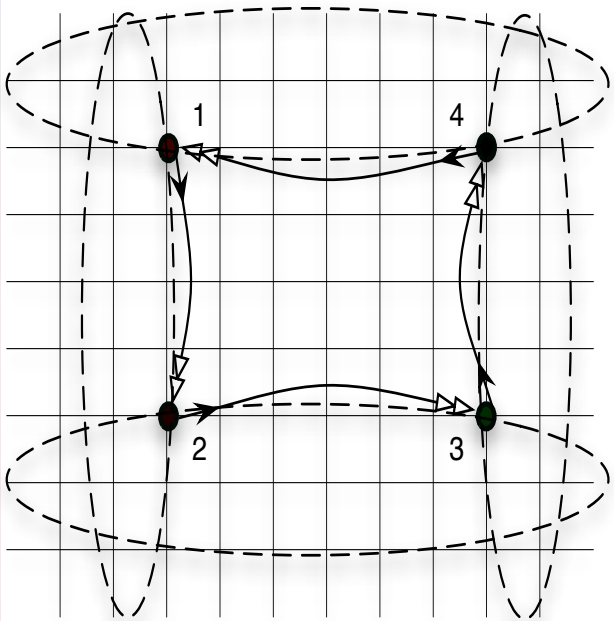
RVD in a $n \times n$ Torus with two Movable Tokens and Constant Memory $O(n^2)$ steps (IV).



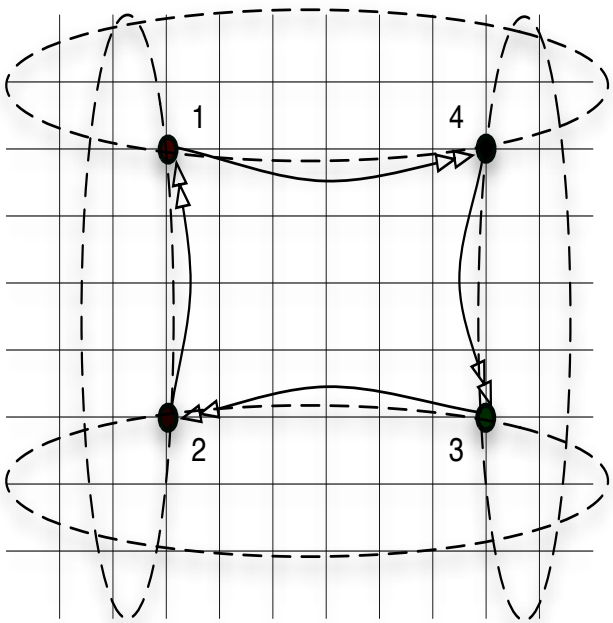
4. Scan Vertically

Scan vertically and try to 'catch' the other.
Not succeeded? then it should also hold $d_x = n/2$.

RV in a $n \times m$ Torus with two Movable Tokens and Constant Memory in $O(n^2 + m^2)$.



(a)



(b)

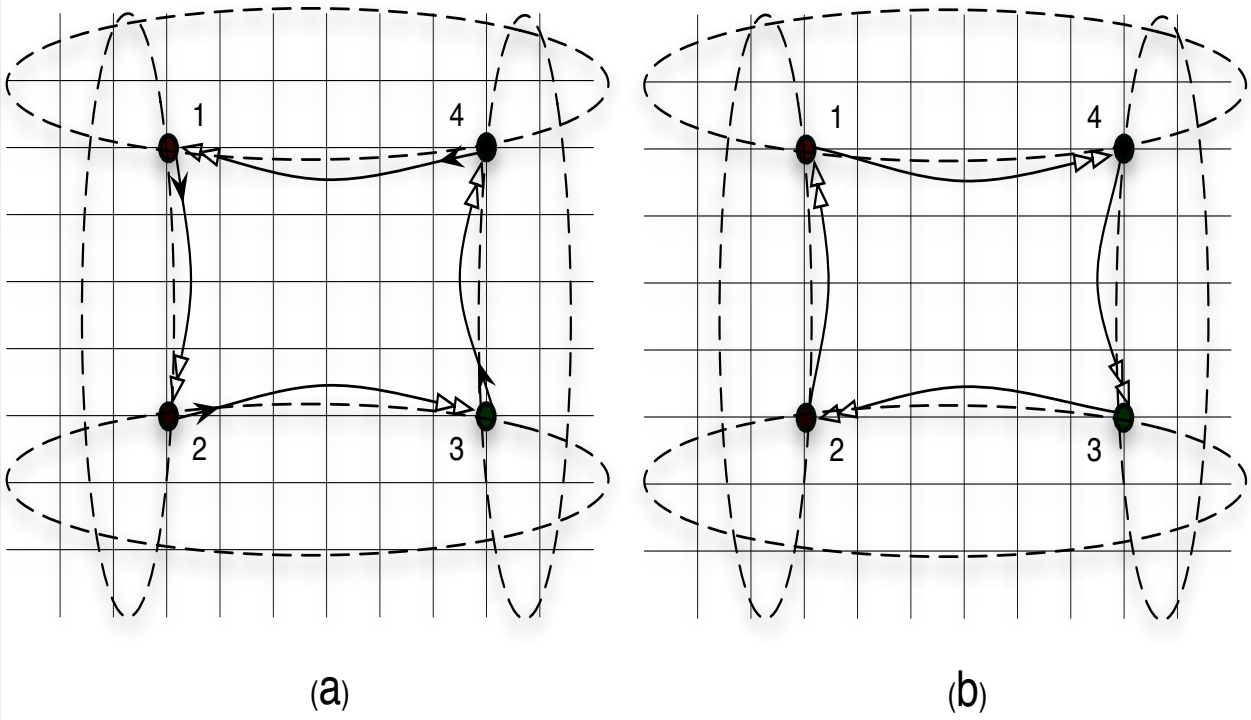
Different ring?

1. Build-Rectangle,
2. Chase the other

Can be adapted to an RVD algorithm when:

$$\frac{m-1}{10} \leq n \leq 10m + 1$$

RV in a $n \times m$ Torus with two Movable Tokens and Constant Memory in $O(n^2 + m^2)$.



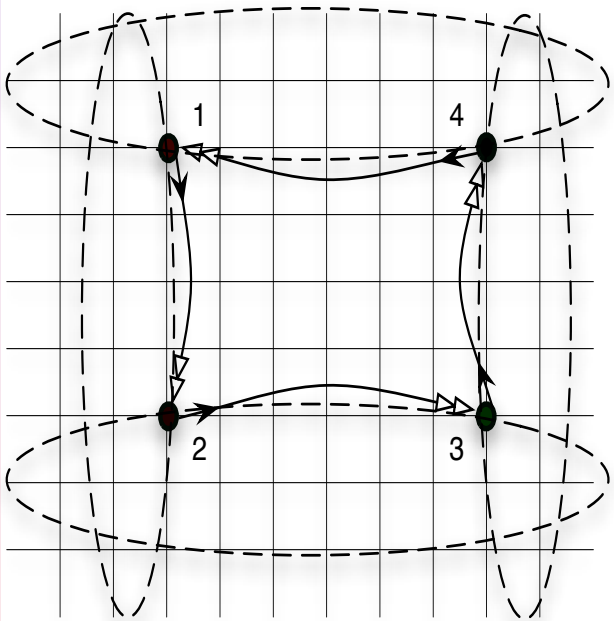
Different ring?

1. Build-Rectangle,
2. Chase the other

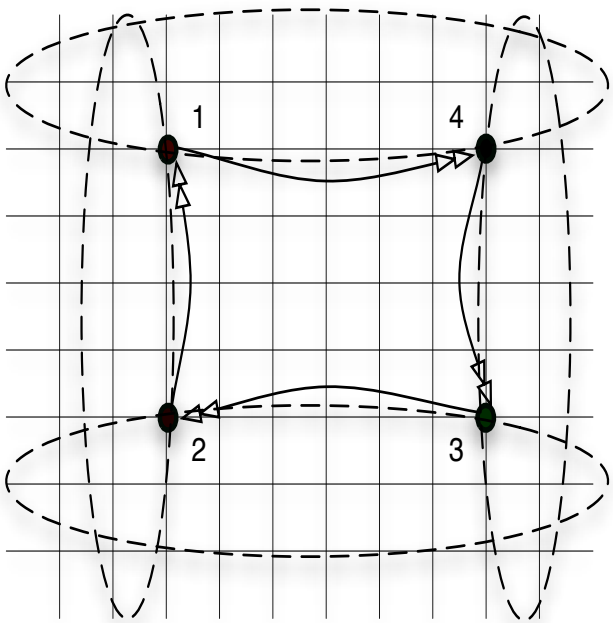
Can be adapted to an RVD algorithm when:

$$\frac{m-1}{10} \leq n \leq 10m + 1$$

RV in a $n \times m$ Torus with two Movable Tokens and Constant Memory in $O(n^2 + m^2)$.



(a)



(b)

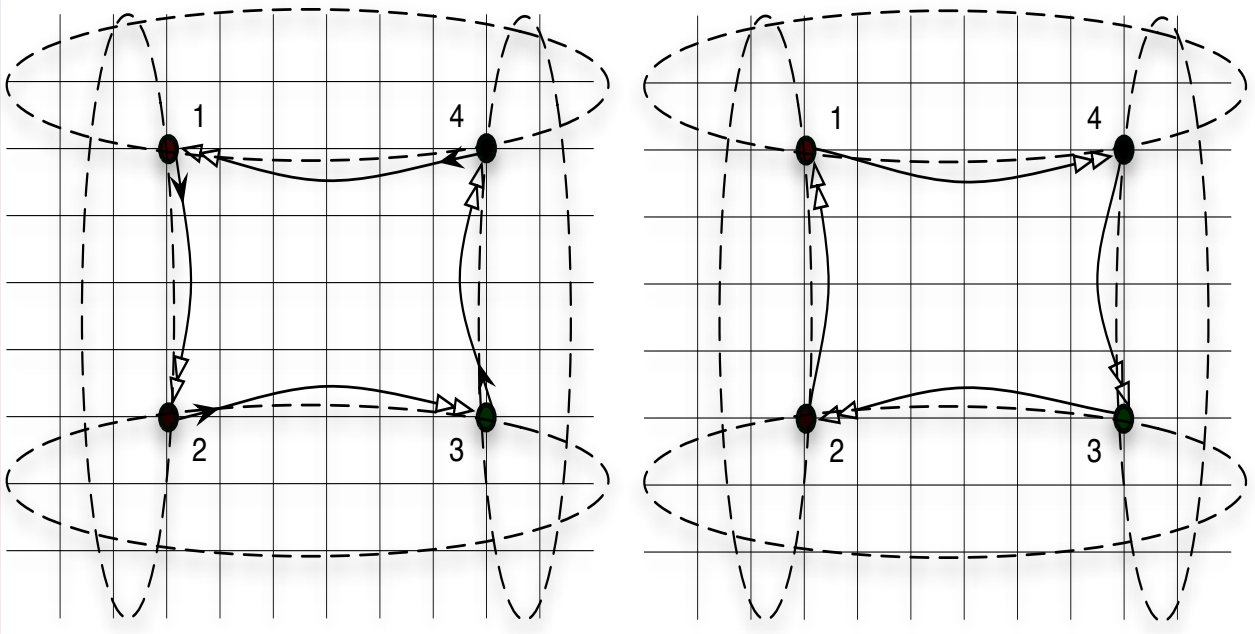
Different ring?

1. Build-Rectangle,
2. Chase the other

Can be adapted to an RVD algorithm when:

$$\frac{m-1}{10} \leq n \leq 10m + 1$$

RV in a $n \times m$ Torus with two Movable Tokens and Constant Memory in $O(n^2 + m^2)$.



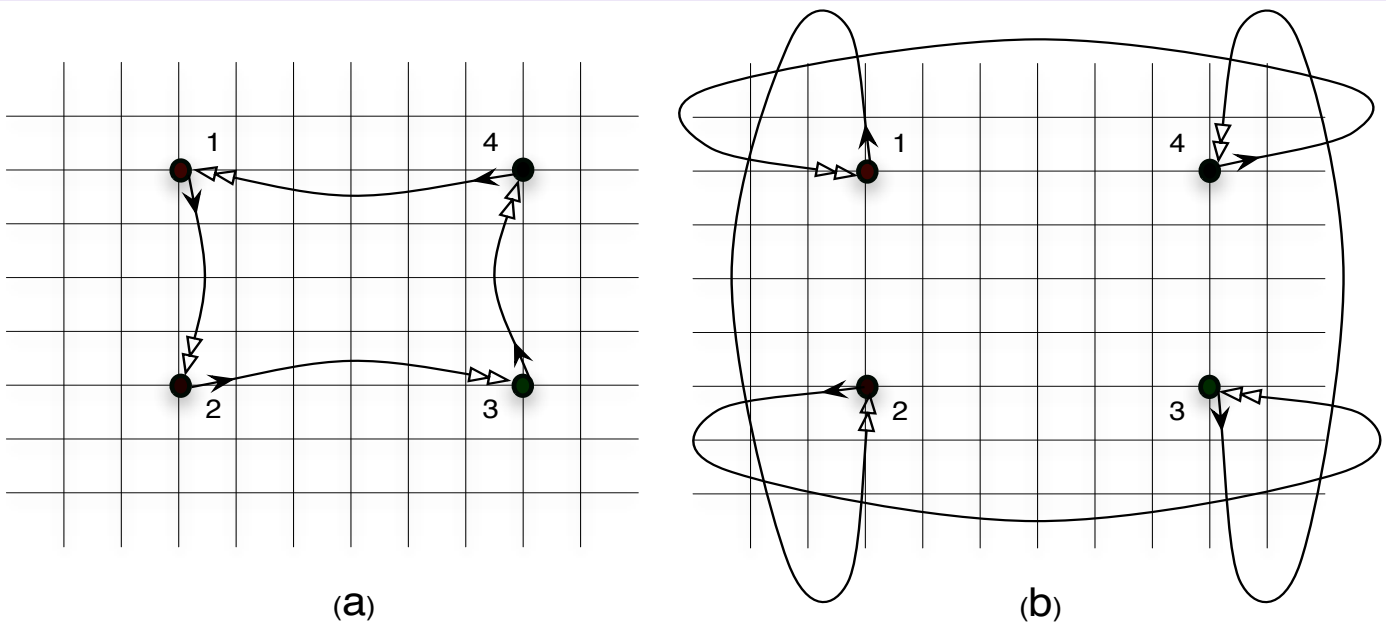
Different ring?

1. Build-Rectangle,
2. Chase the other

Can be adapted to an RVD algorithm when:

$$\frac{m-1}{10} \leq n \leq 10m + 1$$

RVD in a $n \times m$ Torus with three Movable Tokens and Constant Memory in $O(n^2 + m^2)$.



Different ring?

Build rectangle and move your third token along the rectangle (like the RVD algorithm in a ring with two tokens)

Conclusion - Open Questions

Conclusion on the torus

- A constant number of unmovable tokens are less powerful than two movable tokens.
- The hierarchy collapses on three tokens.

Open - Questions

- Are three movable tokens strictly more powerful than two with respect to RVD? (Is RVD possible with two movable tokens and constant memory in a $n \times m$ torus?)
- $d-$dimensional torus? (Is it true that with $d-1$ movable tokens the agents need $\Omega(\log n)$ memory? Is RVD possible with d movable tokens and constant memory?)
- not oriented, asynchronous torus? gathering?

Conclusion - Open Questions

Conclusion on the torus

- A constant number of unmovable tokens are less powerful than two movable tokens.
- The hierarchy collapses on three tokens.

Open - Questions

- Are three movable tokens strictly more powerful than two with respect to RVD? (Is RVD possible with two movable tokens and constant memory in a $n \times m$ torus?)
- $d-$dimensional torus? (Is it true that with $d-1$ movable tokens the agents need $\Omega(\log n)$ memory? Is RVD possible with d movable tokens and constant memory?)
- not oriented, asynchronous torus? gathering?

Other Results.

Multiple agents [Flocchini et al, 2004]

k robots equipped with one stationary token each, in a n-nodes synchronous unoriented ring.

- If both n, k are unknown to robots then rendezvous is unsolvable
- If k is known rendezvous can be solved with $O(\log n)$ memory.

Other Results.

Failing tokens [Flocchini et al, 2004]

k robots equipped with one stationary token each, in a n-nodes synchronous unoriented ring. At most $k-1$ tokens may fail (no longer visible to any robots).

- If all tokens fail or $gcd(k', n) \neq 1$ for some $k' \leq k$, then rendezvous is unsolvable
- If both n, k are unknown to robots then rendezvous is unsolvable
- If n or k is known, $gcd(k', n) = 1$ for all $k' \leq k$, rendezvous can be solved with $O(k \log n)$ memory.

Other Results.

Faulty tokens

- asynchronous rings [Das, 2008]
- arbitrary graphs [Das et al, 2008]

Other Results.

Rendezvous in spite of a black-hole [Dobrev et al, 2003]

k robots in a n-nodes asynchronous ring in spite of a black hole.

- It is impossible for all k agents to rendezvous
- If the ring is unoriented, then it is impossible for $k-1$ agents to rendezvous
- Either n or k must be known
- If k is known then rendezvous is solvable in an oriented ring with whiteboards

The Gathering Problem

The Gathering Problem

Mobile entities (robots), initially situated at different locations, have to gather at the same location (not determined in advance) and remain in it.

The Gathering Problem

A very weak scenario

- decentralized
- asynchronous
- no common knowledge
- no identities
- no central coordination
- no direct communication
- oblivious

The Gathering Problem

A very weak scenario

- decentralized
- asynchronous
- no common knowledge
- no identities
- no central coordination
- no direct communication
- oblivious

The Gathering Problem

A very weak scenario

- decentralized
- asynchronous
- no common knowledge
- no identities
- no central coordination
- no direct communication
- oblivious

The Gathering Problem

A very weak scenario

- decentralized
- asynchronous
- no common knowledge
- no identities
- no central coordination
- no direct communication
- oblivious

The Gathering Problem

A very weak scenario

- decentralized
- asynchronous
- no common knowledge
- no identities
- no central coordination
- no direct communication
- oblivious

The Gathering Problem

A very weak scenario

- decentralized
- asynchronous
- no common knowledge
- no identities
- no central coordination
- no direct communication
- oblivious

The Gathering Problem

A very weak scenario

- decentralized
- asynchronous
- no common knowledge
- no identities
- no central coordination
- no direct communication
- oblivious

The Gathering Problem

Difficulties

- The robots have to break symmetry by agreeing on a common meeting location,
- they cannot communicate directly but have to make decisions about their moves only by observing the environment,
- they have no memory and they operate asynchronously in an anonymous network.

The Gathering Problem

Difficulties

- The robots have to break symmetry by agreeing on a common meeting location,
- they cannot communicate directly but have to make decisions about their moves only by observing the environment,
- they have no memory and they operate asynchronously in an anonymous network.

The Gathering Problem

Difficulties

- The robots have to break symmetry by agreeing on a common meeting location,
- they cannot communicate directly but have to make decisions about their moves only by observing the environment,
- they have no memory and they operate asynchronously in an anonymous network.

Outline

Related Work.

Anonymous identical robots that cannot send any messages and communicate with the environment only by observing it. This model was initially used to study deterministic gathering in the case of robots moving freely in the plane.

[Cieliebak, 2004]

Robots have memory.

[Ando et al, 1999], [Flocchini, Peleg, Prencipe, ...]

Robots are oblivious, i.e., they do not have any memory of past observations.

[Ando et al, 1999], [Suzuki, Yamashita, 1999]

Cycles are executed synchronously by all active robots.

Related Work.

Anonymous identical robots that cannot send any messages and communicate with the environment only by observing it. This model was initially used to study deterministic gathering in the case of robots moving freely in the plane.

[Cieliebak, 2004]

Robots have memory.

[Ando et al, 1999], [Flocchini, Peleg, Prencipe, ...]

Robots are oblivious, i.e., they do not have any memory of past observations.

[Ando et al, 1999], [Suzuki, Yamashita, 1999]

Cycles are executed synchronously by all active robots.

Getting closer to our model.

[Cieliebak et al, 2004], [Cohen, Peleg, 2004]

Cycles are executed asynchronously.

[Flocchini, Prencipe, Santoro, Widmayer, 2005]

Gathering is possible in the asynchronous model if robots have the same orientation of the plane, even with limited visibility.

[Cieliebak, 2003], [Prencipe, 2005]

Without orientation:

- gathering is possible when the robots have the capability of multiplicity detection,
- gathering is impossible otherwise.

Outline

Our model (I) [Klasing, Markou, Pelc, TCS08]

A number of identical mobile agents (robots)

- placed in an anonymous and unoriented ring (at most one robot at a node),
- operate in Look-Compute-Move cycles.

In one cycle, a robot based on the perceived configuration,

- takes a snapshot of the current configuration (Look), then,
- makes a decision to stay idle or to move to one of its adjacent nodes (Compute),
- and in the latter case makes an instantaneous move to this neighbor (Move).

Our model (I) [Klasing, Markou, Pelc, TCS08]

A number of identical mobile agents (robots)

- placed in an anonymous and unoriented ring (at most one robot at a node),
- operate in Look-Compute-Move cycles.

In one cycle, a robot based on the perceived configuration,

- takes a snapshot of the current configuration (Look), then,
- makes a decision to stay idle or to move to one of its adjacent nodes (Compute),
- and in the latter case makes an instantaneous move to this neighbor (Move).

Our model (I) [Klasing, Markou, Pelc, TCS08]

A number of identical mobile agents (robots)

- placed in an anonymous and unoriented ring (at most one robot at a node),
- operate in Look-Compute-Move cycles.

In one cycle, a robot based on the perceived configuration,

- takes a snapshot of the current configuration (Look), then,
- makes a decision to stay idle or to move to one of its adjacent nodes (Compute),
- and in the latter case makes an instantaneous move to this neighbor (Move).

Our model (I) [Klasing, Markou, Pelc, TCS08]

A number of identical mobile agents (robots)

- placed in an anonymous and unoriented ring (at most one robot at a node),
- operate in Look-Compute-Move cycles.

In one cycle, a robot based on the perceived configuration,

- takes a snapshot of the current configuration (Look), then,
- makes a decision to stay idle or to move to one of its adjacent nodes (Compute),
- and in the latter case makes an instantaneous move to this neighbor (Move).

Our model (I) [Klasing, Markou, Pelc, TCS08]

A number of identical mobile agents (robots)

- placed in an anonymous and unoriented ring (at most one robot at a node),
- operate in Look-Compute-Move cycles.

In one cycle, a robot based on the perceived configuration,

- takes a snapshot of the current configuration (Look), then,
- makes a decision to stay idle or to move to one of its adjacent nodes (Compute),
- and in the latter case makes an instantaneous move to this neighbor (Move).

Our model (I) [Klasing, Markou, Pelc, TCS08]

A number of identical mobile agents (robots)

- placed in an anonymous and unoriented ring (at most one robot at a node),
- operate in Look-Compute-Move cycles.

In one cycle, a robot based on the perceived configuration,

- takes a snapshot of the current configuration (Look), then,
- makes a decision to stay idle or to move to one of its adjacent nodes (Compute),
- and in the latter case makes an instantaneous move to this neighbor (Move).

Our model (II) [KMP]

Rules

1. Cycles are performed asynchronously for each robot.
2. Moves are instantaneous.
3. The robots are memoryless (oblivious), i.e. they do not have any memory of past observations.
4. The robots are anonymous and execute the same deterministic algorithm.

Observations

- Robots move based on significantly outdated perceptions.
- The target node is decided by the robot during a Compute operation, solely on the basis of the location of other robots perceived in the previous Look operation.

Our model (II) [KMP]

Rules

1. Cycles are performed asynchronously for each robot.
2. Moves are instantaneous.
3. The robots are memoryless (oblivious), i.e. they do not have any memory of past observations.
4. The robots are anonymous and execute the same deterministic algorithm.

Observations

- Robots move based on significantly outdated perceptions.
- The target node is decided by the robot during a Compute operation, solely on the basis of the location of other robots perceived in the previous Look operation.

Our model (II) [KMP]

Rules

1. Cycles are performed asynchronously for each robot.
2. Moves are instantaneous.
3. The robots are memoryless (oblivious), i.e. they do not have any memory of past observations.
4. The robots are anonymous and execute the same deterministic algorithm.

Observations

- Robots move based on significantly outdated perceptions.
- The target node is decided by the robot during a Compute operation, solely on the basis of the location of other robots perceived in the previous Look operation.

Our model (II) [KMP]

Rules

1. Cycles are performed asynchronously for each robot.
2. Moves are instantaneous.
3. The robots are memoryless (oblivious), i.e. they do not have any memory of past observations.
4. The robots are anonymous and execute the same deterministic algorithm.

Observations

- Robots move based on significantly outdated perceptions.
- The target node is decided by the robot during a Compute operation, solely on the basis of the location of other robots perceived in the previous Look operation.

Our model (II) [KMP]

Rules

1. Cycles are performed asynchronously for each robot.
2. Moves are instantaneous.
3. The robots are memoryless (oblivious), i.e. they do not have any memory of past observations.
4. The robots are anonymous and execute the same deterministic algorithm.

Observations

- Robots move based on significantly outdated perceptions.
- The target node is decided by the robot during a Compute operation, solely on the basis of the location of other robots perceived in the previous Look operation.

Our model (II) [KMP]

Rules

1. Cycles are performed asynchronously for each robot.
2. Moves are instantaneous.
3. The robots are memoryless (oblivious), i.e. they do not have any memory of past observations.
4. The robots are anonymous and execute the same deterministic algorithm.

Observations

- Robots move based on significantly outdated perceptions.
- The target node is decided by the robot during a Compute operation, solely on the basis of the location of other robots perceived in the previous Look operation.

Our model (II) [KMP]

Rules

1. Cycles are performed asynchronously for each robot.
2. Moves are instantaneous.
3. The robots are memoryless (oblivious), i.e. they do not have any memory of past observations.
4. The robots are anonymous and execute the same deterministic algorithm.

Observations

- Robots move based on significantly outdated perceptions.
- The target node is decided by the robot during a Compute operation, solely on the basis of the location of other robots perceived in the previous Look operation.

Main Results [KMP]

Gathering is impossible for any number of robots under this weak scenario. We add multiplicity detection capability.

For an odd number of robots:

- Gathering is feasible if and only if the initial configuration is not periodic.
- We give a gathering algorithm for any such configuration.

For an even number of robots:

- We decide feasibility of gathering except for one type of symmetric configurations.
- We provide gathering algorithms for initial configurations proved to be gatherable.

Main Results [KMP]

Gathering is impossible for any number of robots under this weak scenario. We add multiplicity detection capability.

For an odd number of robots:

- Gathering is feasible if and only if the initial configuration is not periodic.
- We give a gathering algorithm for any such configuration.

For an even number of robots:

- We decide feasibility of gathering except for one type of symmetric configurations.
- We provide gathering algorithms for initial configurations proved to be gatherable.

Main Results [KMP]

Gathering is impossible for any number of robots under this weak scenario. We add multiplicity detection capability.

For an odd number of robots:

- Gathering is feasible if and only if the initial configuration is not periodic.
- We give a gathering algorithm for any such configuration.

For an even number of robots:

- We decide feasibility of gathering except for one type of symmetric configurations.
- We provide gathering algorithms for initial configurations proved to be gatherable.

Main Results [KMP]

Gathering is impossible for any number of robots under this weak scenario. We add multiplicity detection capability.

For an odd number of robots:

- Gathering is feasible if and only if the initial configuration is not periodic.
- We give a gathering algorithm for any such configuration.

For an even number of robots:

- We decide feasibility of gathering except for one type of symmetric configurations.
- We provide gathering algorithms for initial configurations proved to be gatherable.

Main Results [KMP]

Gathering is impossible for any number of robots under this weak scenario. We add multiplicity detection capability.

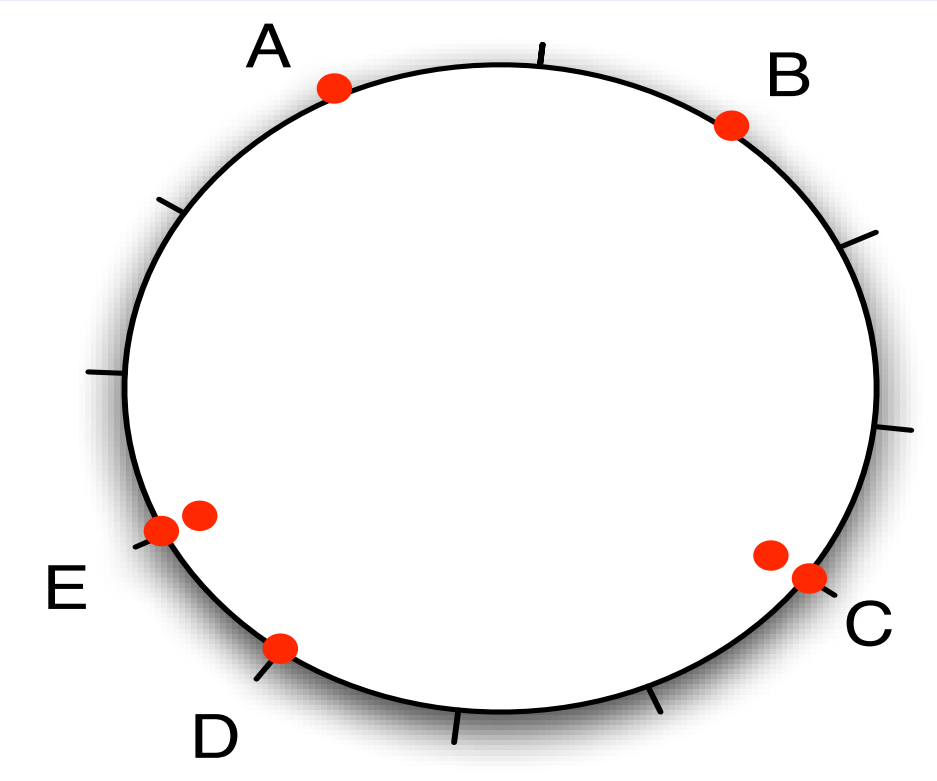
For an odd number of robots:

- Gathering is feasible if and only if the initial configuration is not periodic.
- We give a gathering algorithm for any such configuration.

For an even number of robots:

- We decide feasibility of gathering except for one type of symmetric configurations.
- We provide gathering algorithms for initial configurations proved to be gatherable.

A view of a configuration



Configuration by A

$((2, 3, 3, 1, 3), (5, 9))$

Configuration by B

$((3, 3, 1, 3, 2), (3, 7))$

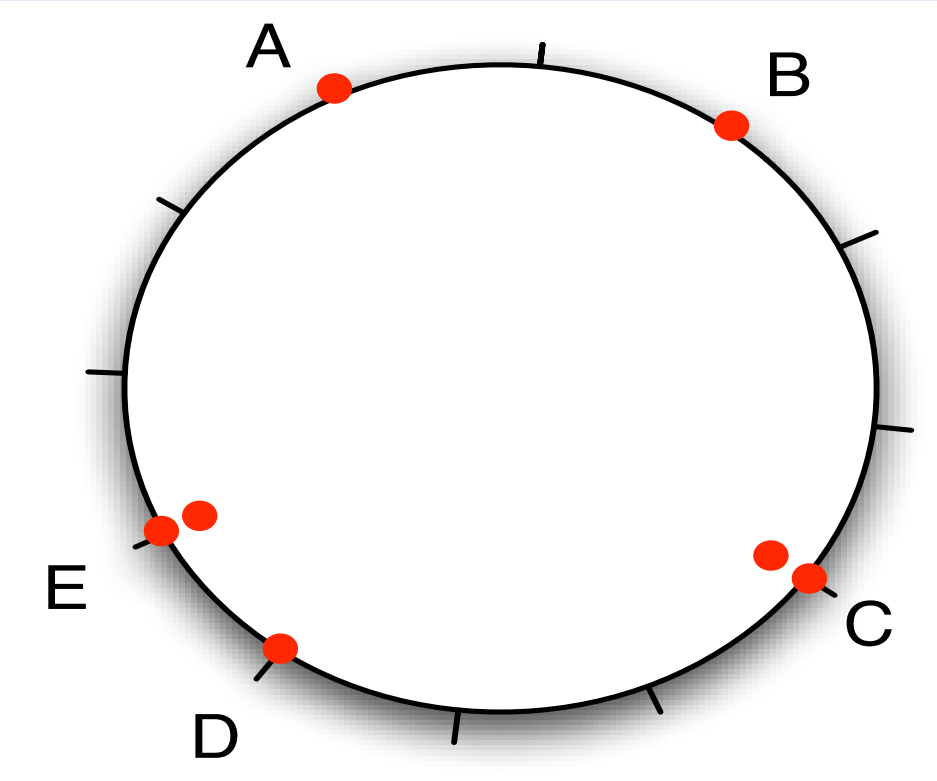
Configuration by C

$((3, 1, 3, 2, 3), (0, 4))$

View of robot A

$\{((2, 3, 3, 1, 3), (5, 9)), ((3, 1, 3, 3, 2), (3, 7))\}$

A view of a configuration



Configuration by A

$((2,3,3,1,3),(5,9))$

Configuration by B

$((3,3,1,3,2),(3,7))$

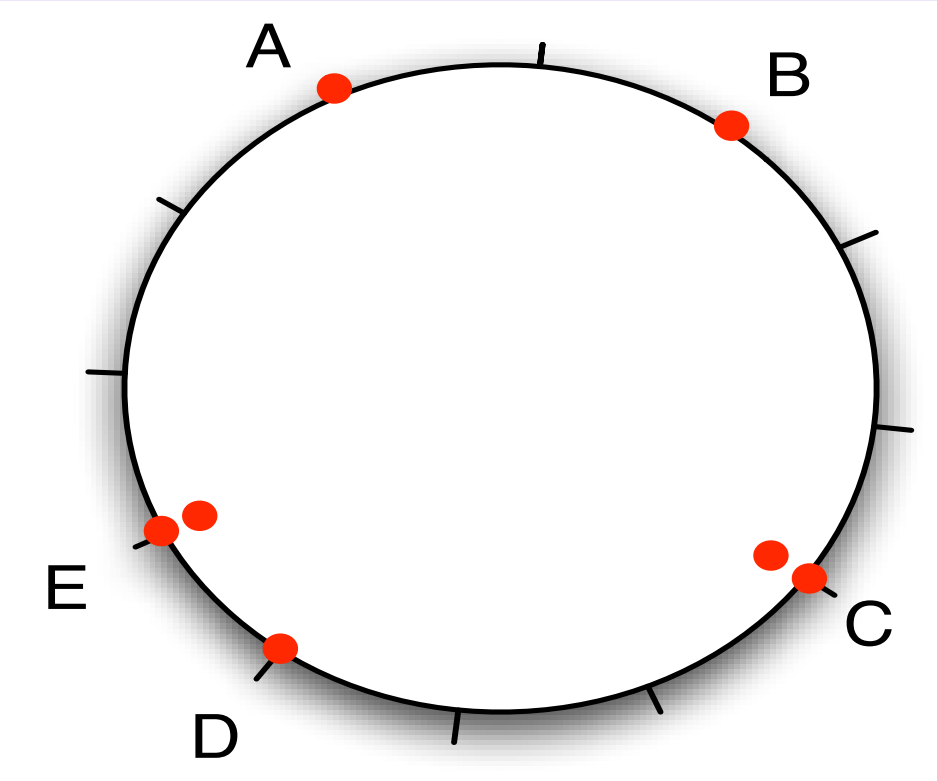
Configuration by C

$((3,1,3,2,3),(0,4))$

View of robot A

$\{((2,3,3,1,3),(5,9)),((3,1,3,3,2),(3,7))\}$

A view of a configuration



Configuration by A

$((2, 3, 3, 1, 3), (5, 9))$

Configuration by B

$((3, 3, 1, 3, 2), (3, 7))$

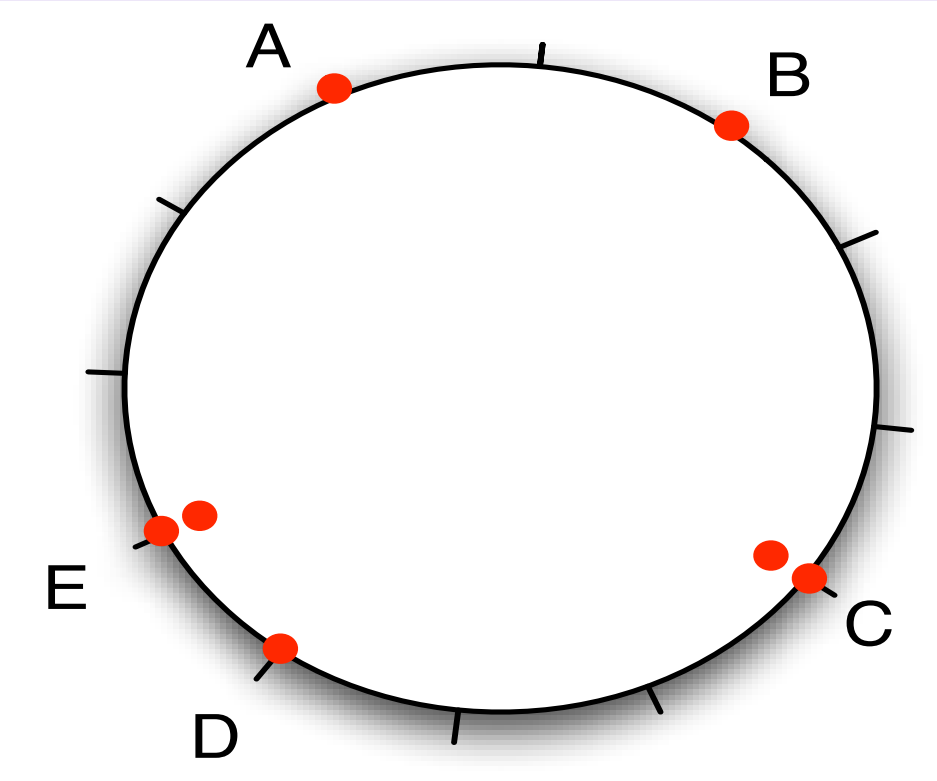
Configuration by C

$((3, 1, 3, 2, 3), (0, 4))$

View of robot A

$\{((2, 3, 3, 1, 3), (5, 9)), ((3, 1, 3, 3, 2), (3, 7))\}$

A view of a configuration



Configuration by A

$((2, 3, 3, 1, 3), (5, 9))$

Configuration by B

$((3, 3, 1, 3, 2), (3, 7))$

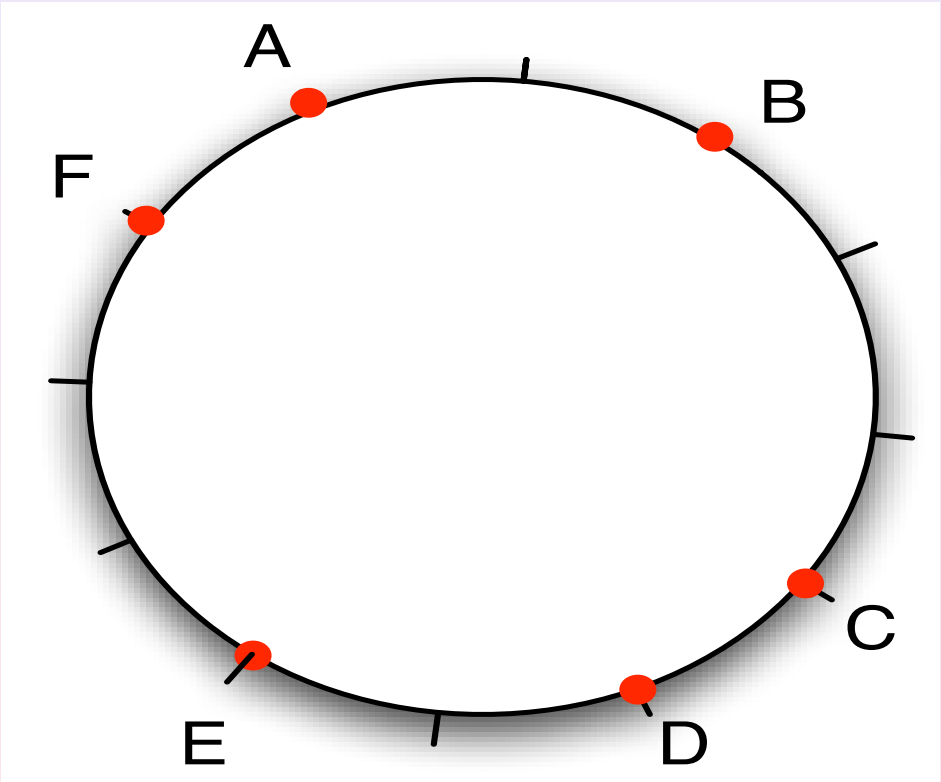
Configuration by C

$((3, 1, 3, 2, 3), (0, 4))$

View of robot A

$$\{((2, 3, 3, 1, 3), (5, 9)), ((3, 1, 3, 3, 2), (3, 7))\}$$

A periodic configuration



Periodic configuration

A configuration without multiplicities is called periodic if it is a concatenation of at least two copies of a subsequence p.

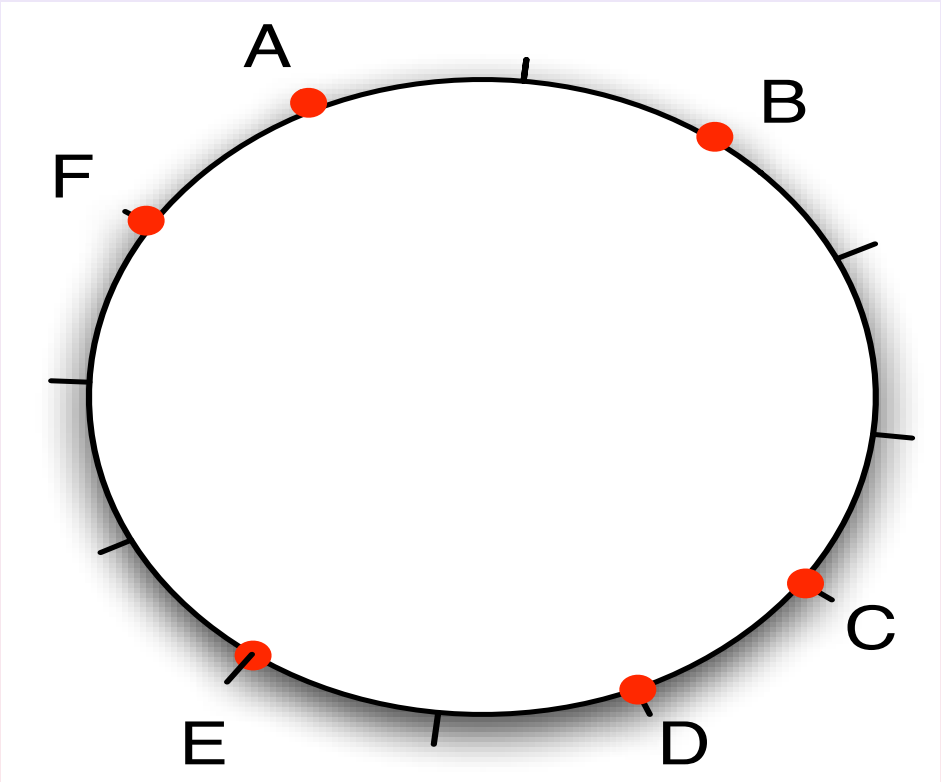
Configuration by A

$(2, 3, 1, 2, 3, 1)$

View of robot A

$\{(2, 3, 1, 2, 3, 1), (1, 3, 2, 1, 3, 2)\}$

A periodic configuration



Periodic configuration

A configuration without multiplicities is called periodic if it is a concatenation of at least two copies of a subsequence p.

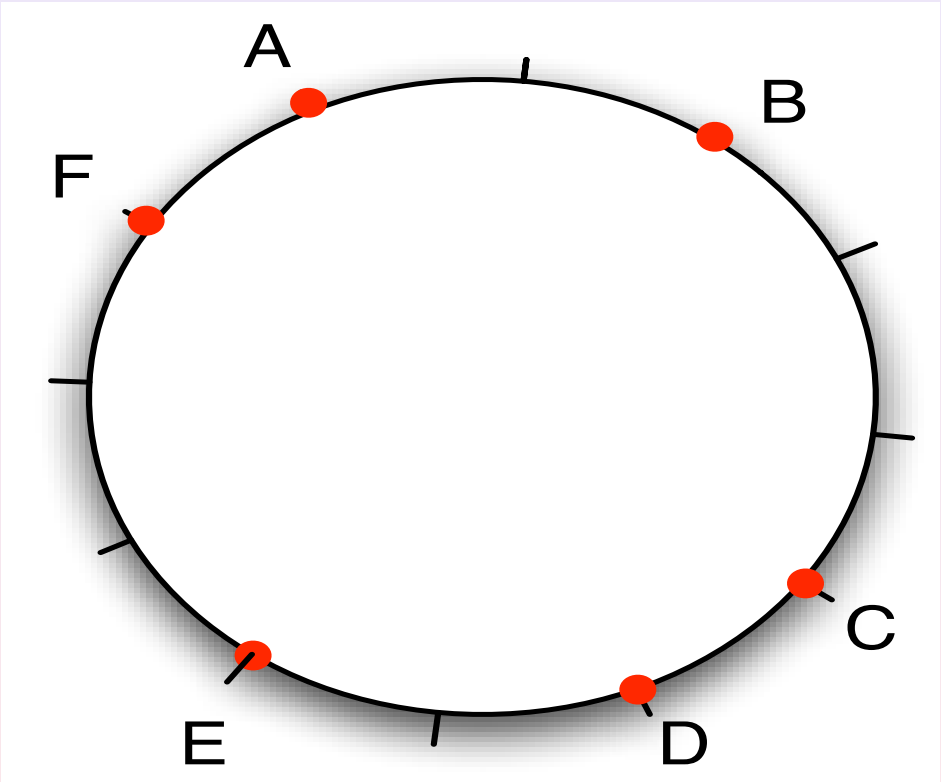
Configuration by A

$(2, 3, 1, 2, 3, 1)$

View of robot A

$\{(2, 3, 1, 2, 3, 1), (1, 3, 2, 1, 3, 2)\}$

A periodic configuration



Periodic configuration

A configuration without multiplicities is called periodic if it is a concatenation of at least two copies of a subsequence p.

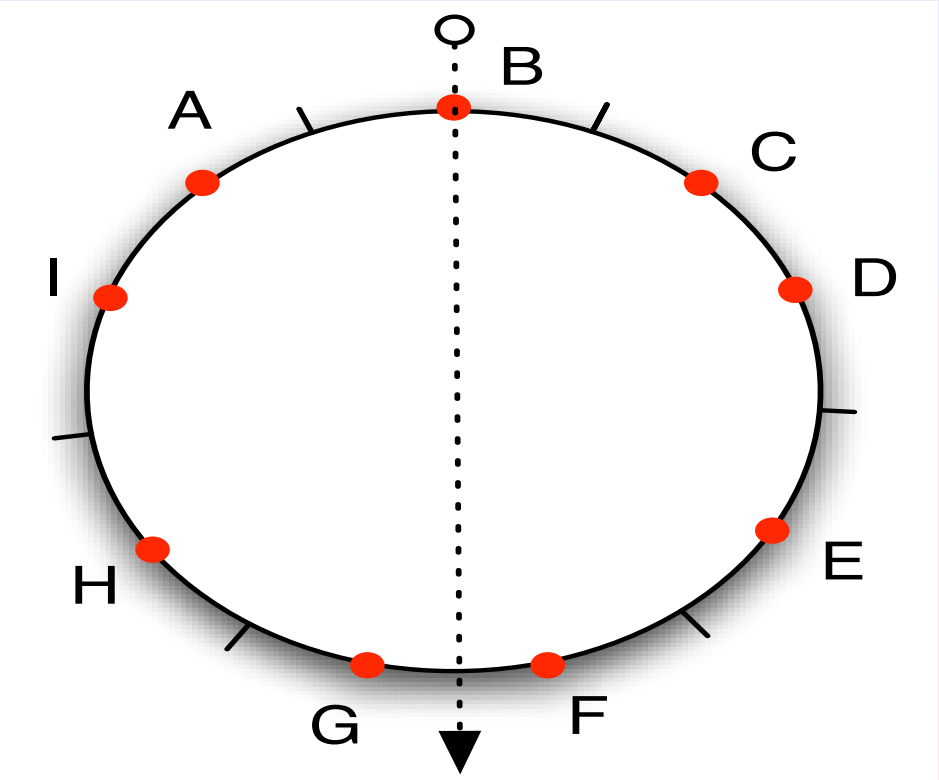
Configuration by A

$(2, 3, 1, 2, 3, 1)$

View of robot A

$$\{(2, 3, 1, 2, 3, 1), (1, 3, 2, 1, 3, 2)\}$$

A symmetric (and periodic configuration)



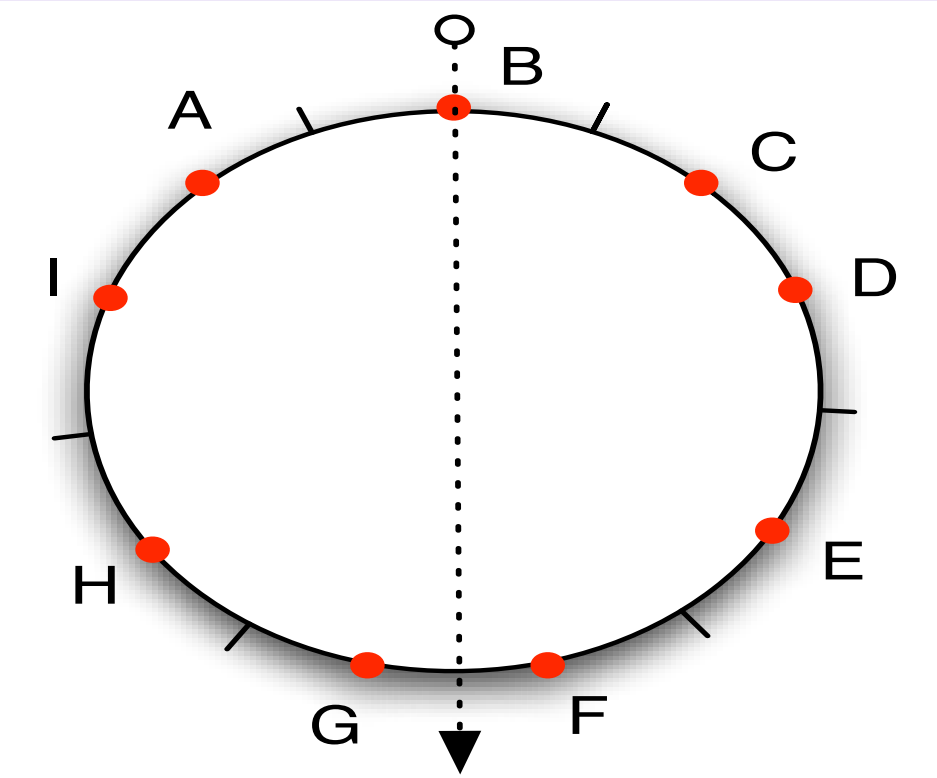
Symmetric configuration

A configuration without multiplicities is called symmetric if there exists an axis of symmetry of the ring, such that the set of the nodes occupied by robots is symmetric with respect to this axis.

View of robot A

$$\{(2,2,1,2,2,1,2,2,1),(1,2,2,1,2,2,1,2,2)\}$$

A symmetric (and periodic configuration)



Symmetric configuration

A configuration without multiplicities is called symmetric if there exists an axis of symmetry of the ring, such that the set of the nodes occupied by robots is symmetric with respect to this axis.

View of robot A

$$\{(2,2,1,2,2,1,2,2,1),(1,2,2,1,2,2,1,2,2)\}$$

Property lemmas

Definition

A configuration without multiplicities is called rigid if the views of all robots are distinct.

Lemma

A configuration without multiplicities is non-rigid, if and only if it is either periodic or symmetric.

Lemma

If a configuration without multiplicities is non-rigid and non-periodic then it has exactly one axis of symmetry.

Property lemmas

Definition

A configuration without multiplicities is called rigid if the views of all robots are distinct.

Lemma

A configuration without multiplicities is non-rigid, if and only if it is either periodic or symmetric.

Lemma

If a configuration without multiplicities is non-rigid and non-periodic then it has exactly one axis of symmetry.

Property lemmas

Definition

A configuration without multiplicities is called rigid if the views of all robots are distinct.

Lemma

A configuration without multiplicities is non-rigid, if and only if it is either periodic or symmetric.

Lemma

If a configuration without multiplicities is non-rigid and non-periodic then it has exactly one axis of symmetry.

Outline

Impossibility results

Lemmas

1. Gathering any 2 robots is impossible on any ring.
2. If multiplicity detection is not available then gathering any $k > 1$ robots is impossible on any ring.

Theorems

1. Gathering is impossible for any periodic configuration.
2. Gathering is impossible for any edge-edge symmetric configuration.

Impossibility results

Lemmas

1. Gathering any 2 robots is impossible on any ring.
2. If multiplicity detection is not available then gathering any $k > 1$ robots is impossible on any ring.

Theorems

1. Gathering is impossible for any periodic configuration.
2. Gathering is impossible for any edge-edge symmetric configuration.

Impossibility results

Lemmas

1. Gathering any 2 robots is impossible on any ring.
2. If multiplicity detection is not available then gathering any $k > 1$ robots is impossible on any ring.

Theorems

1. Gathering is impossible for any periodic configuration.
2. Gathering is impossible for any edge-edge symmetric configuration.

Impossibility results

Lemmas

1. Gathering any 2 robots is impossible on any ring.
2. If multiplicity detection is not available then gathering any $k > 1$ robots is impossible on any ring.

Theorems

1. Gathering is impossible for any periodic configuration.
2. Gathering is impossible for any edge-edge symmetric configuration.

Impossibility results

Lemmas

1. Gathering any 2 robots is impossible on any ring.
2. If multiplicity detection is not available then gathering any $k > 1$ robots is impossible on any ring.

Theorems

1. Gathering is impossible for any periodic configuration.
2. Gathering is impossible for any edge-edge symmetric configuration.

Impossibility results

Lemmas

1. Gathering any 2 robots is impossible on any ring.
2. If multiplicity detection is not available then gathering any $k > 1$ robots is impossible on any ring.

Theorems

1. Gathering is impossible for any periodic configuration.
2. Gathering is impossible for any edge-edge symmetric configuration.

Outline

Gathering configurations with a single multiplicity

Procedure Single-Multiplicity-Gathering

if R is at the multiplicity **then** do not move
else
  if none of the segments between R
    and the multiplicity is free
  then do not move
  else move towards the multiplicity along the shortest
    of the free segments or along any of them
    in the case of equality.

Lemma

Procedure Single-Multiplicity-Gathering performs gathering of robots for any configuration with a single multiplicity.

Gathering configurations with a single multiplicity

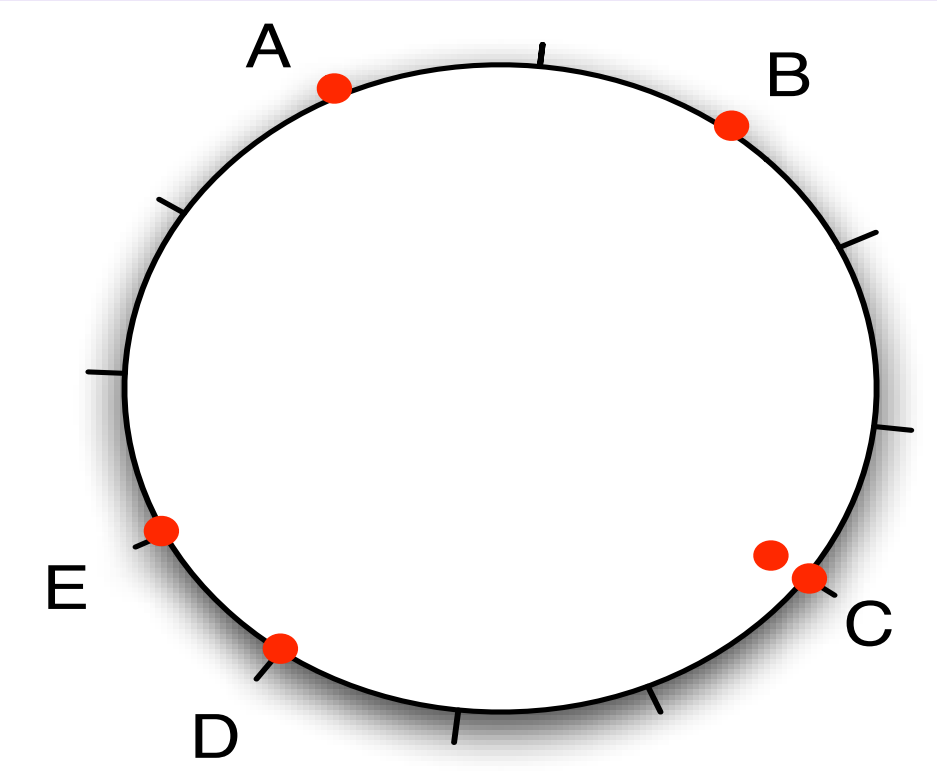
Procedure Single-Multiplicity-Gathering

if R is at the multiplicity **then** do not move
else
  if none of the segments between R
    and the multiplicity is free
  then do not move
  else move towards the multiplicity along the shortest
    of the free segments or along any of them
    in the case of equality.

Lemma

Procedure Single-Multiplicity-Gathering performs gathering of robots for any configuration with a single multiplicity.

Gathering configurations with a single multiplicity



Configuration by A

$((2, 3, 3, 1, 3), (5))$

Configuration by B

$((3, 3, 1, 3, 2), (3))$

Configuration by C

$((3, 1, 3, 2, 3), (0))$

View of robot A

$$\{((2, 3, 3, 1, 3), (5)), ((3, 1, 3, 3, 2), (7))\}$$

Gathering rigid configurations

Procedure Rigid-Gathering

- elect a pair of neighboring robots A, B at a maximum distance
- choose the robot which has the other neighboring robot closer ($*$ ties can be broken easily $*$)
- move the elected robot towards the direction which increases the maximum distance

Lemma

Procedure Rigid-Gathering performs gathering of robots for any rigid configuration without multiplicities.

Gathering rigid configurations

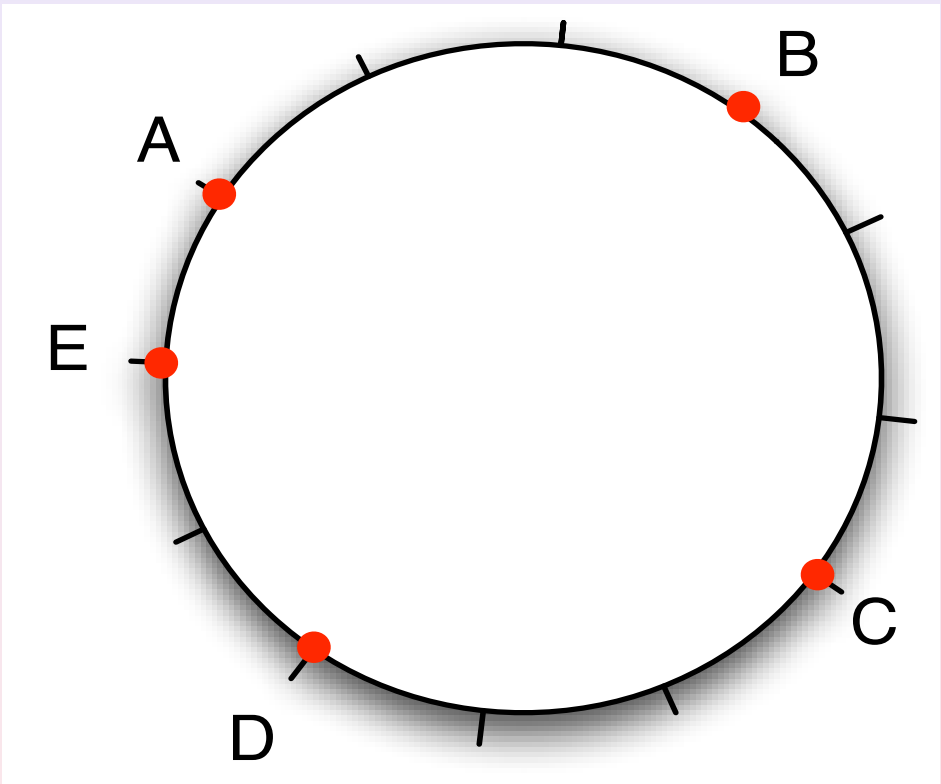
Procedure Rigid-Gathering

- elect a pair of neighboring robots A, B at a maximum distance
- choose the robot which has the other neighboring robot closer ($*$ ties can be broken easily $*$)
- move the elected robot towards the direction which increases the maximum distance

Lemma

Procedure Rigid-Gathering performs gathering of robots for any rigid configuration without multiplicities.

Gathering rigid configurations



Configuration by A

$(3, 3, 3, 2, 1)$

Configuration by B

$(3, 3, 2, 1, 3)$

Configuration by C

$(3, 2, 1, 3, 3)$

View of robot A

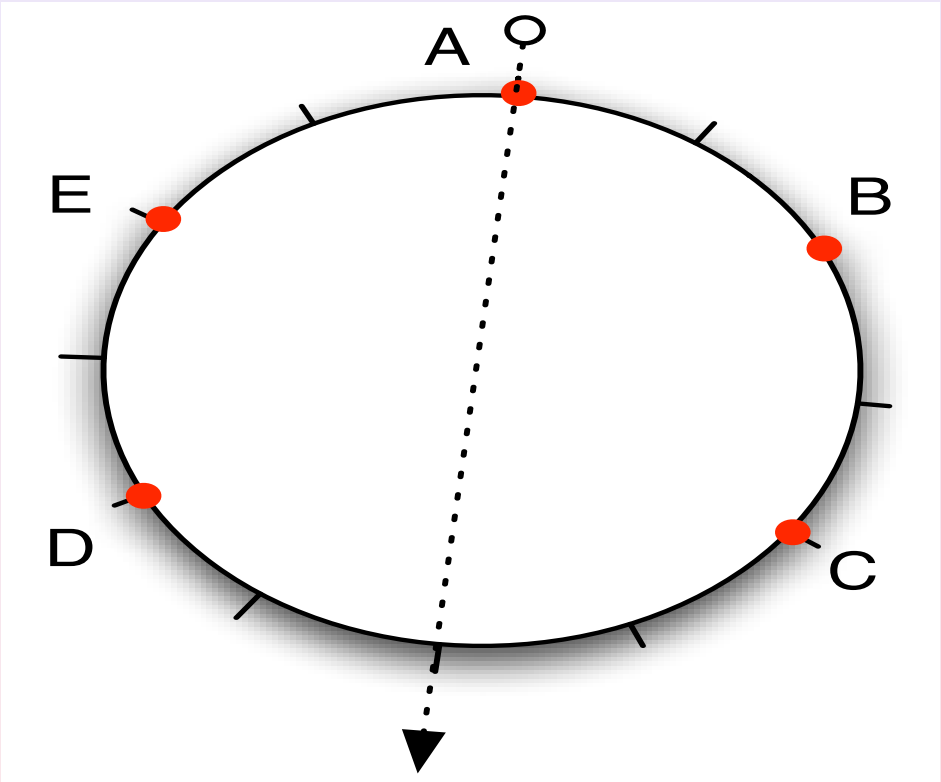
$$\{(3, 3, 3, 2, 1), (1, 2, 3, 3, 3)\}$$

Gathering an odd number of robots

Algorithm Odd-Gathering

if the configuration is periodic **then** gathering is impossible
else
    if the configuration has a single multiplicity
    then Single-Multiplicity-Gathering
    else
        if the configuration is rigid **then** Rigid-Gathering
        else
            if R is axial **then** move (to any of the adjacent nodes)

Gathering an odd number of robots



Configuration by A

$(2, 2, 4, 2, 2)$

Configuration by B

$(2, 4, 2, 2, 2)$

Configuration by C

$(4, 2, 2, 2, 2)$

View of robot A

$$\{(2, 2, 4, 2, 2), (2, 2, 4, 2, 2)\}$$

Gathering an odd number of robots

Let C be a symmetric configuration of an odd number of robots, without multiplicities. Let C' be the configuration resulting from C by moving the axial node to any of the adjacent nodes.

Lemma

If C' does not have multiplicities then it is not periodic.

Proof: Since C is symmetric, C' is of the form $(a+1, b_1, \ldots, b_{s-1}, b_s, b_{s-1}, \ldots, b_1, a-1)$. Suppose that C' has a period d of length p. Then $d_1 = a+1$, $d_p = a-1$. But because of C's symmetry, $d_p = d_1$. Contradiction.

Gathering an odd number of robots

Let C be a symmetric configuration of an odd number of robots, without multiplicities. Let C' be the configuration resulting from C by moving the axial node to any of the adjacent nodes.

Lemma

If C' does not have multiplicities then it is not periodic.
Proof: Since C is symmetric, C' is of the form $(a+1, b_1, \ldots, b_{s-1}, b_s, b_{s-1}, \ldots, b_1, a-1)$. Suppose that C' has a period d of length p. Then $d_1 = a+1$, $d_p = a-1$. But because of C's symmetry, $d_p = d_1$. Contradiction.

Gathering an odd number of robots

Lemma

After less than k moves, either the configuration C' has a single multiplicity or it is rigid.

Proof: Let x be the distance between the axial robot and anyone of its neighbors.

1. Exactly one value appears in C an odd number of times.
2. In C' this value is either $x-1$ or $x+1$.
3. The total number of occurrences in C' of integers of different parity than this value is strictly less than in C.

Theorem

For an odd number of robots, gathering is feasible if and only if the initial configuration is not periodic.

Gathering an odd number of robots

Lemma

After less than k moves, either the configuration C' has a single multiplicity or it is rigid.

Proof: Let x be the distance between the axial robot and anyone of its neighbors.

1. Exactly one value appears in C an odd number of times.
2. In C' this value is either $x-1$ or $x+1$.
3. The total number of occurrences in C' of integers of different parity than this value is strictly less than in C.

Theorem

For an odd number of robots, gathering is feasible if and only if the initial configuration is not periodic.

Gathering an odd number of robots

Lemma

After less than k moves, either the configuration C' has a single multiplicity or it is rigid.

Proof: Let x be the distance between the axial robot and anyone of its neighbors.

1. Exactly one value appears in C an odd number of times.
2. In C' this value is either $x-1$ or $x+1$.
3. The total number of occurrences in C' of integers of different parity than this value is strictly less than in C.

Theorem

For an odd number of robots, gathering is feasible if and only if the initial configuration is not periodic.

Gathering an odd number of robots

Lemma

After less than k moves, either the configuration C' has a single multiplicity or it is rigid.

Proof: Let x be the distance between the axial robot and anyone of its neighbors.

1. Exactly one value appears in C an odd number of times.
2. In C' this value is either $x-1$ or $x+1$.
3. The total number of occurrences in C' of integers of different parity than this value is strictly less than in C.

Theorem

For an odd number of robots, gathering is feasible if and only if the initial configuration is not periodic.

Gathering an odd number of robots

Lemma

After less than k moves, either the configuration C' has a single multiplicity or it is rigid.

Proof: Let x be the distance between the axial robot and anyone of its neighbors.

1. Exactly one value appears in C an odd number of times.
2. In C' this value is either $x-1$ or $x+1$.
3. The total number of occurrences in C' of integers of different parity than this value is strictly less than in C.

Theorem

For an odd number of robots, gathering is feasible if and only if the initial configuration is not periodic.

Gathering an odd number of robots

Lemma

After less than k moves, either the configuration C' has a single multiplicity or it is rigid.

Proof: Let x be the distance between the axial robot and anyone of its neighbors.

1. Exactly one value appears in C an odd number of times.
2. In C' this value is either $x-1$ or $x+1$.
3. The total number of occurrences in C' of integers of different parity than this value is strictly less than in C.

Theorem

For an odd number of robots, gathering is feasible if and only if the initial configuration is not periodic.

Gathering an even number of robots

Theorem [Klasing, Kosowski, Navarra, 2008]

Gathering is possible for any non-periodic configuration without edge-edge symmetry of more than 18 robots.

Conclusion

For an odd number of robots

- gathering is feasible if and only if in the initial configuration, robots can elect a node occupied by a robot.

For an even number of robots

- gathering is impossible when
  - either the number of robots is 2, or
  - the configuration is periodic or,
  - there is an edge-edge symmetry
- gathering is feasible for all non-periodic configurations without edge-edge symmetry of more than 18 robots

Other results.

- Fault tolerance for robots in the plane [Agmon, Peleg, 2006]
- Limited visibility in the plane [Flocchini, 2005]
- Gathering in edge-labeled graphs [Chalopin, Flocchini, Mans, Santoro, 2009]

Open Questions

Problems

- Identify impossibility of rendezvous
- Other communication models
- Trade-offs among time-memory-knowledge-equipment
- Fault tolerance

Thanks!