Remembering Without Memory: Tree Exploration by Asynchronous Oblivious Robots

Paola Flocchini[1] David Ilcinkas[2]
Andrzej Pelc[3] Nicola Santoro[4]

[1]University of Ottawa, Canada
[2]CNRS, Université Bordeaux I, France
[3]Université du Québec en Outaouais, Canada
[4]Carleton University, Canada

SIROCCO '08
June 18, 2008

Problem

Model/context

- Anonymous graphs
- Team of robots
  - sensing the environment by taking a snapshot of it
  - that do not communicate
  - that are anonymous and oblivious

Goal: exploration with stop

- Each node must be visited by at least one robot.
- All robots must stop after finite time.

The Look-Compute-Move cycle

Look

The robot takes a rooted instantaneous snapshot of the network and its robots, with multiplicity detection ("zero", "one", or "more than one" robots).

Compute

Based on this observation, it decides to stay idle or to move to some neighbouring node.

Move

In the latter case it instantaneously moves towards its destination.

Identical oblivious asynchronous robots

Identical

Robots have no IDs. They execute the same program.

Oblivious

The robots have no memory of observations, computations and moves made in previous cycles.

Asynchronous

The time between Look, Compute, and Move operations is finite but unbounded.

Reminder:

Non-communicating

No communication mechanisms between robots, even locally.

Precisions concerning the model

Initial configurations

Arbitrary but without multiplicity (at most 1 robot / node)
(necessary for termination)

In case of symmetry

- Compute : choice of an equivalence class of neighbors
- Actual choice: made by the adversary (i.e. worst case)

Multiplicity detection

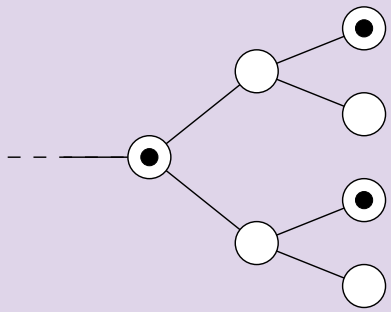
"zero", "one", or "more than one" robots

Precisions concerning the model

Initial configurations

Arbitrary but without multiplicity (at most 1 robot / node)
(necessary for termination)

In case of symmetry



- Compute : choice of an equivalence class of neighbors
- Actual choice: made by the adversary (i.e. worst case)

Multiplicity detection

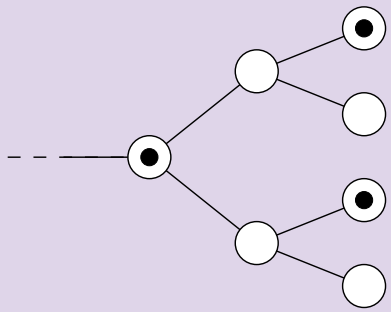
"zero", "one", or "more than one" robots

Precisions concerning the model

Initial configurations

Arbitrary but without multiplicity (at most 1 robot / node)
(necessary for termination)

In case of symmetry



- Compute : choice of an equivalence class of neighbors
- Actual choice: made by the adversary (i.e. worst case)

Multiplicity detection

"zero", "one", or "more than one" robots

Related work (1)

In the plane

Related work (1)

In the plane

- [Suzuki, Yamashita, SIAMJC'99] semi-synchronous model
- [Flocchini et al., ISAAC'99] asynchronous model
- [Cieliebak et al., ICALP'03] gathering assuming multiplicity detection
- [Prencipe, SIROCCO'05] gathering unsolvable without multiplicity detection
- [Flocchini et al., TCS'05] limited visibility
- [Agmon, Peleg, SODA'04] fault tolerant gathering
- [Czyzowicz et al., OPODIS'06] gathering of few fat robots

In graphs

- [Klasing et al., ISAAC'06] gathering in rings

Related work (1)

In the plane

- [Suzuki, Yamashita, SIAMJC'99] semi-synchronous model
- [Flocchini et al., ISAAC'99] asynchronous model
- [Cieliebak et al., ICALP'03] gathering assuming multiplicity detection
- [Prencipe, SIROCCO'05] gathering unsolvable without multiplicity detection
- [Flocchini et al., TCS'05] limited visibility
- [Agmon, Peleg, SODA'04] fault tolerant gathering
- [Czyzowicz et al., OPODIS'06] gathering of few fat robots

In graphs

- [Klasing et al., ISAAC'06] gathering in rings

Related work (1)

In the plane

- [Suzuki, Yamashita, SIAMJC'99] semi-synchronous model
- [Flocchini et al., ISAAC'99] asynchronous model
- [Cieliebak et al., ICALP'03] gathering assuming multiplicity detection
- [Prencipe, SIROCCO'05] gathering unsolvable without multiplicity detection
- [Flocchini et al., TCS'05] limited visibility
- [Agmon, Peleg, SODA'04] fault tolerant gathering
- [Czyzowicz et al., OPODIS'06] gathering of few fat robots

In graphs

- [Klasing et al., ISAAC'06] gathering in rings

Related work (1)

In the plane

- [Suzuki, Yamashita, SIAMJC'99] semi-synchronous model
- [Flocchini et al., ISAAC'99] asynchronous model
- [Cieliebak et al., ICALP'03] gathering assuming multiplicity detection
- [Prencipe, SIROCCO'05] gathering unsolvable without multiplicity detection
- [Flocchini et al., TCS'05] limited visibility
- [Agmon, Peleg, SODA'04] fault tolerant gathering
- [Czyzowicz et al., OPODIS'06] gathering of few fat robots

In graphs

- [Klasing et al., ISAAC'06] gathering in rings

Related work (1)

In the plane

- [Suzuki, Yamashita, SIAMJC'99] semi-synchronous model
- [Flocchini et al., ISAAC'99] asynchronous model
- [Cieliebak et al., ICALP'03] gathering assuming multiplicity detection
- [Prencipe, SIROCCO'05] gathering unsolvable without multiplicity detection
- [Flocchini et al., TCS'05] limited visibility
- [Agmon, Peleg, SODA'04] fault tolerant gathering
- [Czyzowicz et al., OPODIS'06] gathering of few fat robots

In graphs

- [Klasing et al., ISAAC'06] gathering in rings

Related work (1)

In the plane

- [Suzuki, Yamashita, SIAMJC'99] semi-synchronous model
- [Flocchini et al., ISAAC'99] asynchronous model
- [Cieliebak et al., ICALP'03] gathering assuming multiplicity detection
- [Prencipe, SIROCCO'05] gathering unsolvable without multiplicity detection
- [Flocchini et al., TCS'05] limited visibility
- [Agmon, Peleg, SODA'04] fault tolerant gathering
- [Czyzowicz et al., OPODIS'06] gathering of few fat robots

In graphs

- [Klasing et al., ISAAC'06] gathering in rings

Related work (1)

In the plane

- [Suzuki, Yamashita, SIAMJC'99] semi-synchronous model
- [Flocchini et al., ISAAC'99] asynchronous model
- [Cieliebak et al., ICALP'03] gathering assuming multiplicity detection
- [Prencipe, SIROCCO'05] gathering unsolvable without multiplicity detection
- [Flocchini et al., TCS'05] limited visibility
- [Agmon, Peleg, SODA'04] fault tolerant gathering
- [Czyzowicz et al., OPODIS'06] gathering of few fat robots

In graphs

- [Klasing et al., ISAAC'06] gathering in rings

Related work (1)

In the plane

- [Suzuki, Yamashita, SIAMJC'99] semi-synchronous model
- [Flocchini et al., ISAAC'99] asynchronous model
- [Cieliebak et al., ICALP'03] gathering assuming multiplicity detection
- [Prencipe, SIROCCO'05] gathering unsolvable without multiplicity detection
- [Flocchini et al., TCS'05] limited visibility
- [Agmon, Peleg, SODA'04] fault tolerant gathering
- [Czyzowicz et al., OPODIS'06] gathering of few fat robots

In graphs

- [Klasing et al., ISAAC'06] gathering in rings

Related work (2)

[Flocchini, Ilcinkas, Pelc, and Santoro. OPODIS 2007]

Lemma

Exploration of a n-node ring by k robots is

- impossible if $k|n$ but $k \neq n$;
- possible if $gcd(n, k) = 1$, for $k \geq 17$.

Corollary

$\Theta(\log n)$ robots are necessary and sufficient in the n-node ring

Related work (2)

[Flocchini, Ilcinkas, Pelc, and Santoro. OPODIS 2007]

Lemma

Exploration of a n-node ring by k robots is

- impossible if $k|n$ but $k \neq n$;
- possible if $gcd(n, k) = 1$, for $k \geq 17$.

Corollary

$\Theta(\log n)$ robots are necessary and sufficient in the n-node ring

Smallest exploring team

Exploration

We say that exploration of a graph is possible with k robots, if there exists an algorithm enabling the robots to perform exploration with stop of this graph starting from any initial configuration of the k robots (thus, without multiplicity).

Smallest exploring team

Minimum number of robots that can explore any graph of a given family.

Smallest exploring team

Exploration

We say that exploration of a graph is possible with k robots, if there exists an algorithm enabling the robots to perform exploration with stop of this graph starting from any initial configuration of the k robots (thus, without multiplicity).

Smallest exploring team

Minimum number of robots that can explore any graph of a given family.

Our results

Main result

Trees of maximum degree 3:

- $\Theta(\log n / \log \log n)$ robots

Justification of the restrictions

- $\Theta(\log n)$ robots in cycles [OPODIS 2007]
- $\Theta(n)$ robots in some trees of maximum degree 4 (complete ternary trees)

Our results

Main result

Trees of maximum degree 3:

- $\Theta(\log n / \log \log n)$ robots

Justification of the restrictions

- $\Theta(\log n)$ robots in cycles [OPODIS 2007]
- $\Theta(n)$ robots in some trees of maximum degree 4 (complete ternary trees)

Our results

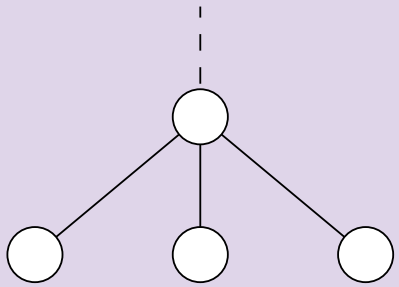
Main result

Trees of maximum degree 3:

- $\Theta(\log n / \log \log n)$ robots

Justification of the restrictions

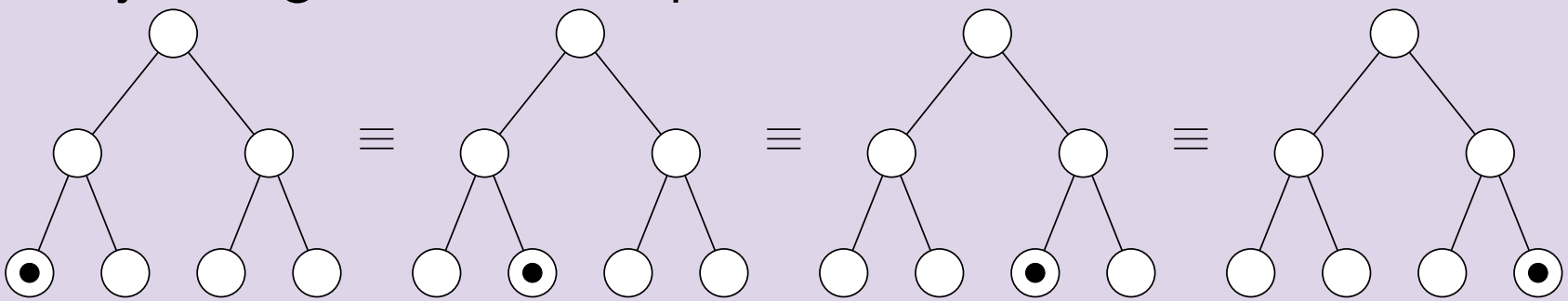
- $\Theta(\log n)$ robots in cycles [OPODIS 2007]
- $\Theta(n)$ robots in some trees of maximum degree 4 (complete ternary trees)



Lower bound: $\Omega(\log n / \log \log n)$ robots

Observation

Many configurations are equivalent for the robots



Sketch of the proof

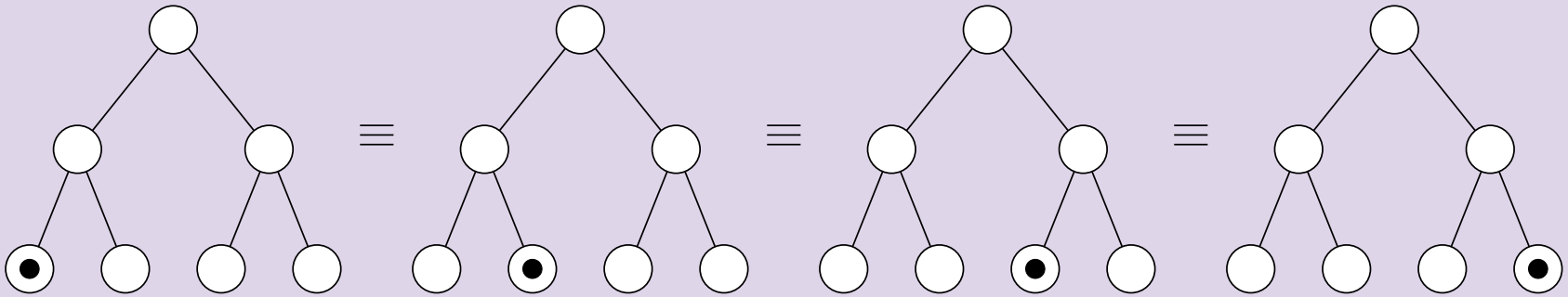
Complete binary tree, synchronous case

- few robots $\Rightarrow$ few different snapshots, say x
- at most x different snapshots $\Rightarrow$ at most $x \cdot k$ explored nodes before stopping

Lower bound: $\Omega(\log n / \log \log n)$ robots

Observation

Many configurations are equivalent for the robots



Sketch of the proof

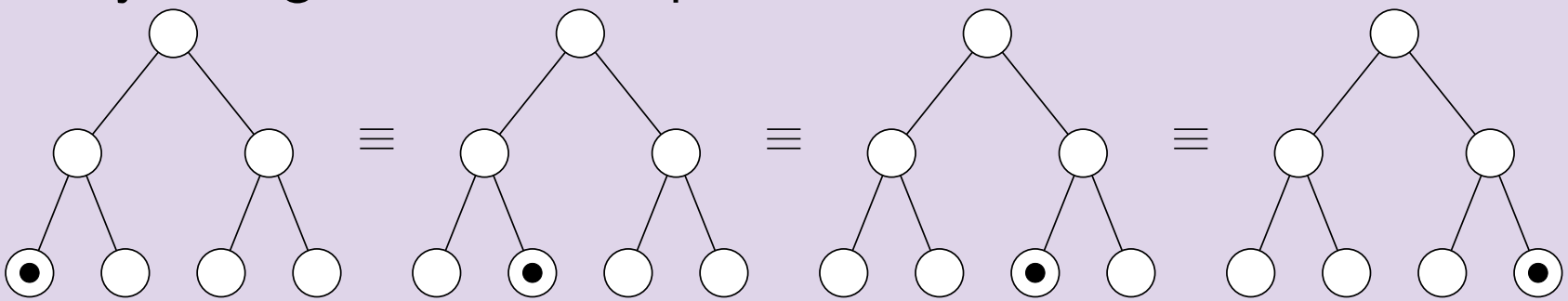
Complete binary tree, synchronous case

- few robots $\Rightarrow$ few different snapshots, say x
- at most x different snapshots $\Rightarrow$ at most $x \cdot k$ explored nodes before stopping

Lower bound: $\Omega(\log n / \log \log n)$ robots

Observation

Many configurations are equivalent for the robots



Sketch of the proof

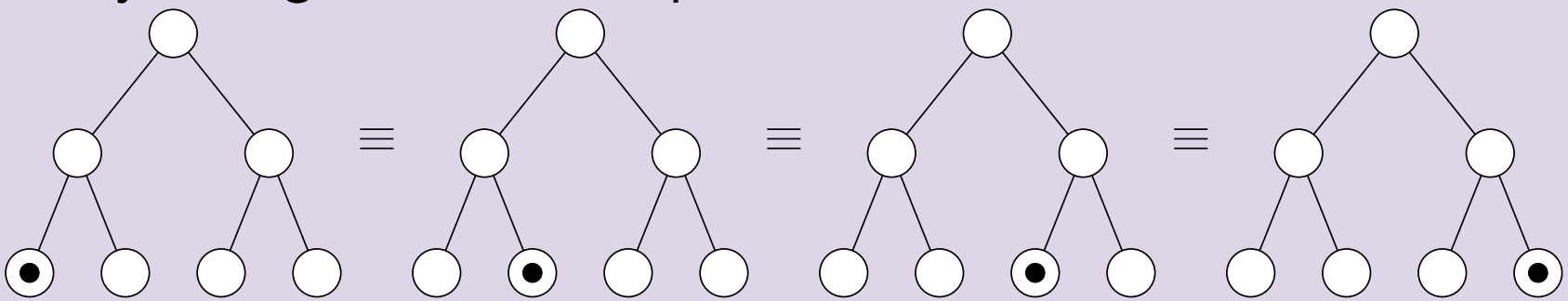
Complete binary tree, synchronous case

- few robots $\Rightarrow$ few different snapshots, say x
- at most x different snapshots $\Rightarrow$ at most $x \cdot k$ explored nodes before stopping

Lower bound: $\Omega(\log n / \log \log n)$ robots

Observation

Many configurations are equivalent for the robots



Sketch of the proof

Complete binary tree, synchronous case

- few robots $\Rightarrow$ few different snapshots, say x
- at most x different snapshots $\Rightarrow$ at most $x \cdot k$ explored nodes before stopping

Upper bound: $O(\log n / \log \log n)$ robots

Theorem

For any n, there exists a team of $k \in \Theta(\log n / \log \log n)$ robots, with $k \equiv 5 \pmod 6$ that can explore all n-node trees of maximum degree 3, starting from any initial configuration.

Main ideas

- A team of three robots aims at exploring the tree
- All other robots are used to keep track of progress
- A visual pattern, called the "brain", formed by the robots counts the number of explored leaves
- The tree is divided into few pieces and is explored piece by piece.

Upper bound: $O(\log n / \log \log n)$ robots

Theorem

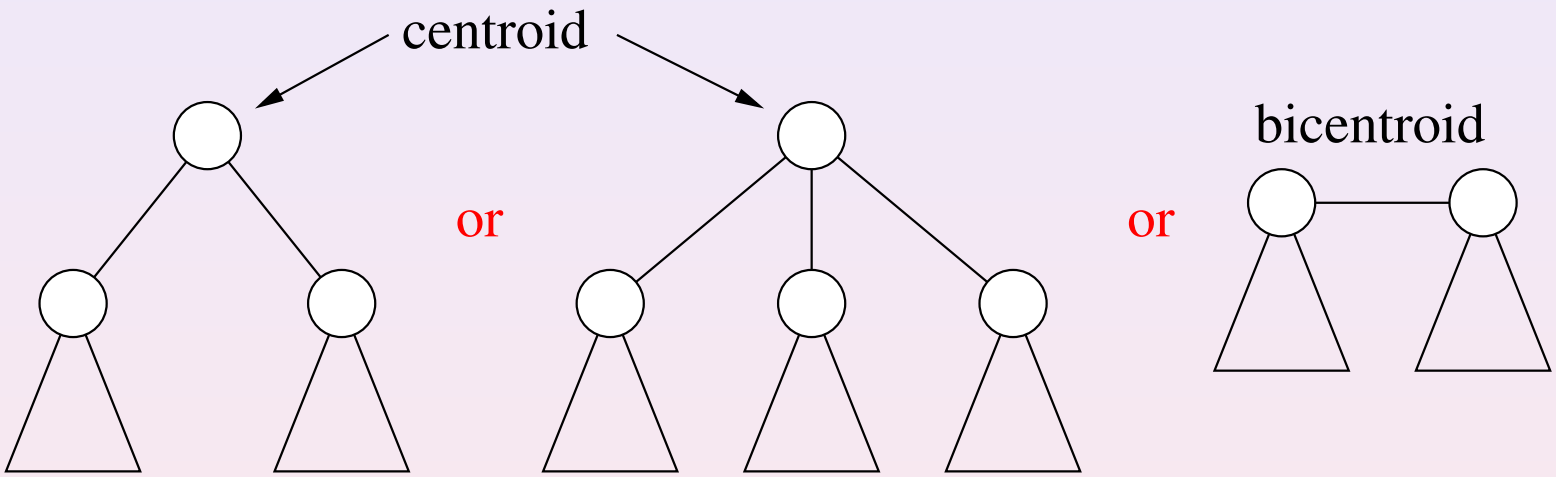
For any n, there exists a team of $k \in \Theta(\log n / \log \log n)$ robots, with $k \equiv 5 \pmod 6$ that can explore all n-node trees of maximum degree 3, starting from any initial configuration.

Main ideas

- A team of three robots aims at exploring the tree
- All other robots are used to keep track of progress
- A visual pattern, called the "brain", formed by the robots counts the number of explored leaves
- The tree is divided into few pieces and is explored piece by piece.

Pieces

The centroid defines pieces in the tree.

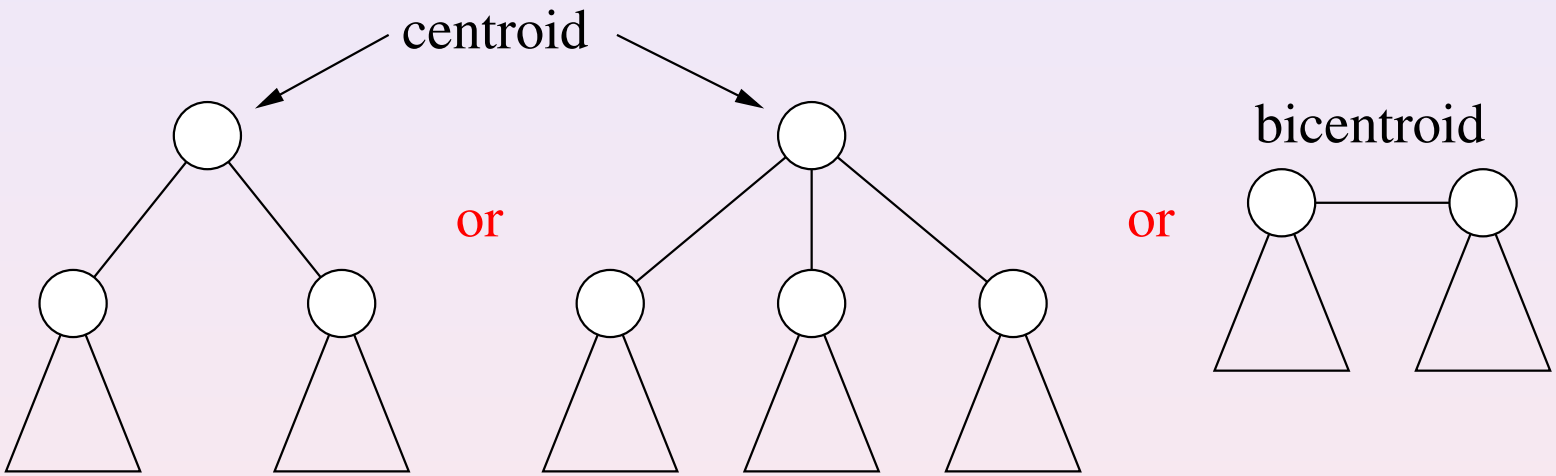


Property

The two largest pieces have size at least $n/4$.

Pieces

The centroid defines pieces in the tree.



Property

The two largest pieces have size at least $n/4$.

Phase 1 (1)

Goal: Make room in the pieces and create one multiplicity

Steps

- Any robot goes down if it does not create a multiplicity
- A leader is elected in the heaviest piece P (i.e. the one with the largest number of robots)
- The leader helps in creating a single multiplicity in P

Property

The core zone is connected and is formed by at least $\frac{n}{\log n}$ nodes.

Phase 1 (1)

Goal: Make room in the pieces and create one multiplicity

Steps

- Any robot goes down if it does not create a multiplicity
- A leader is elected in the heaviest piece P (i.e. the one with the largest number of robots)
- The leader helps in creating a single multiplicity in P

Property

The core zone is connected and is formed by at least $\frac{n}{\log n}$ nodes.

Phase 1 (1)

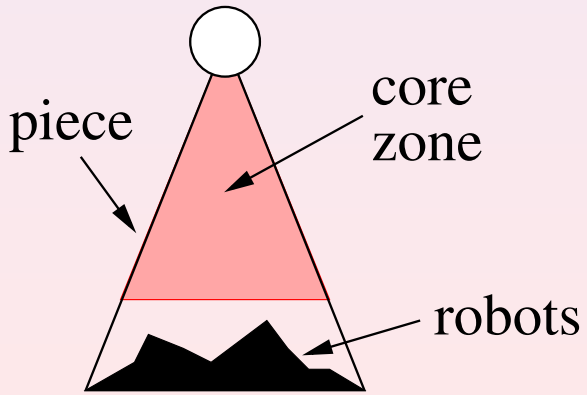
Goal: Make room in the pieces and create one multiplicity

Steps

- Any robot goes down if it does not create a multiplicity
- A leader is elected in the heaviest piece P (i.e. the one with the largest number of robots)
- The leader helps in creating a single multiplicity in P

Property

The core zone is connected and is formed by at least $\frac{n}{\log n}$ nodes.



Phase 1 (2)

Observation

In a piece, the number of robots having the same view is always a power of two and thus either even or one (solitaire).

Corollary

- A piece of odd weight has a (local) leader
- Since $k \equiv 5 \pmod 6$, there always exists a global leader
- It is possible to have a single heaviest piece P, having a leader

Phase 1 (2)

Observation

In a piece, the number of robots having the same view is always a power of two and thus either even or one (solitaire).

Corollary

- A piece of odd weight has a (local) leader
- Since $k \equiv 5 \pmod 6$, there always exists a global leader
- It is possible to have a single heaviest piece P, having a leader

Phase 2

The brain

It synchronizes the actions of the robots and counts the number of explored leaves.

Goal of Phase 2

- Construct and initialize the brain in the core zone of the largest piece Q (different from P) by moving robots from the heavy piece P, using the leader to break symmetries.
- Form the exploring team of three robots in P.
- Remove (move in Q) all other robots in Q.

Phase 2

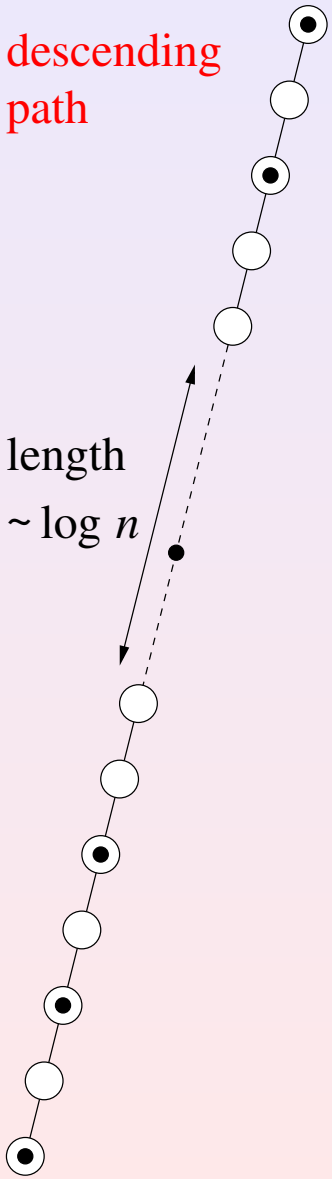
The brain

It synchronizes the actions of the robots and counts the number of explored leaves.

Goal of Phase 2

- Construct and initialize the brain in the core zone of the largest piece Q (different from P) by moving robots from the heavy piece P, using the leader to break symmetries.
- Form the exploring team of three robots in P.
- Remove (move in Q) all other robots in Q.

A counter



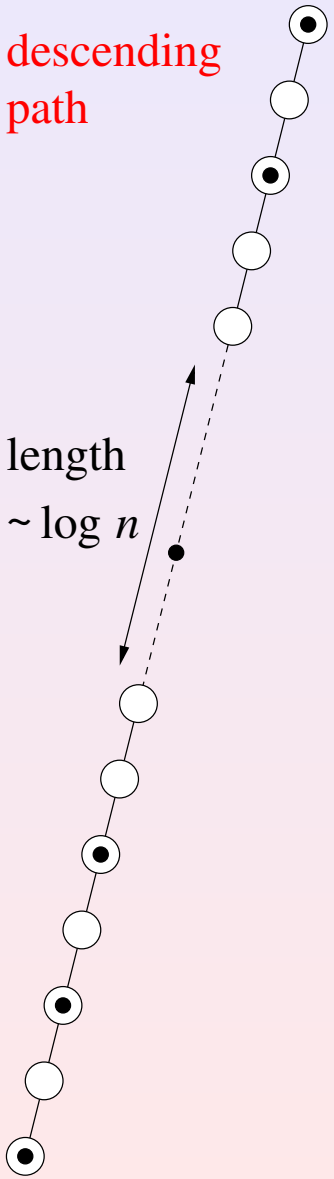
Lemma

In a core zone of size m, one can construct $\log^2 m$ disjoint descending paths of length $\frac{1}{4}\log m$.

Counter

One can construct a counter with range n by using $\Theta(\log n / \log\log n)$ descending paths and thus $\Theta(\log n / \log\log n)$ robots.

A counter



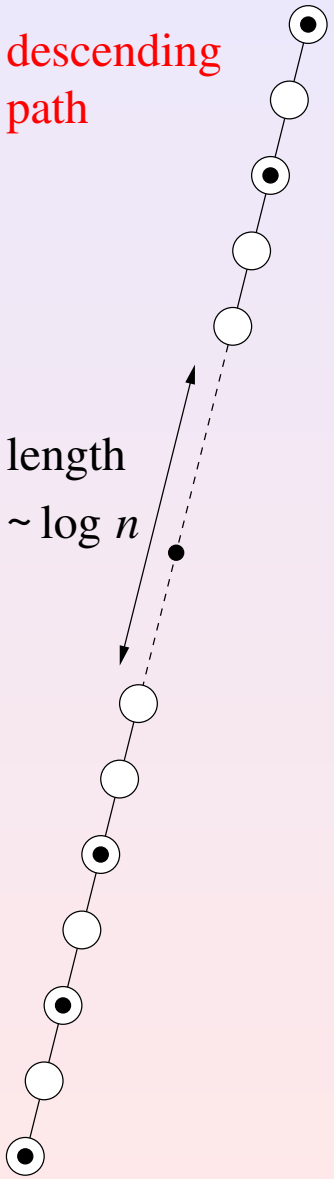
Lemma

In a core zone of size m, one can construct $\log^2 m$ disjoint descending paths of length $\frac{1}{4}\log m$.

Counter

One can construct a counter with range n by using $\Theta(\log n / \log\log n)$ descending paths and thus $\Theta(\log n / \log\log n)$ robots.

A counter



Lemma

In a core zone of size m, one can construct $\log^2 m$ disjoint descending paths of length $\frac{1}{4}\log m$.

Counter

One can construct a counter with range n by using $\Theta(\log n / \log\log n)$ descending paths and thus $\Theta(\log n / \log\log n)$ robots.

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

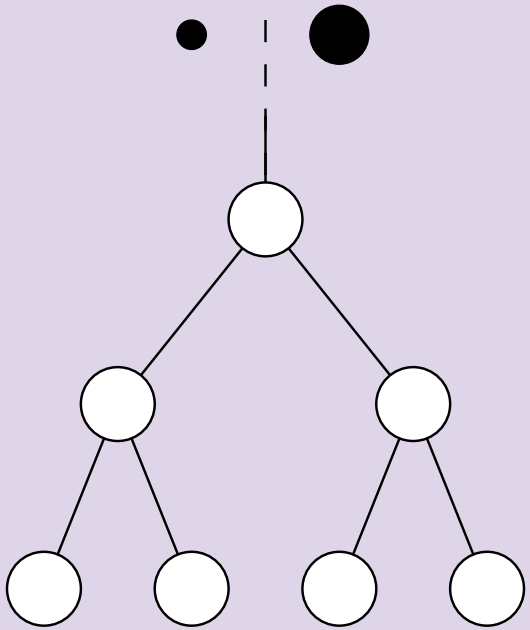
Exploration of a pair of leaves

Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves

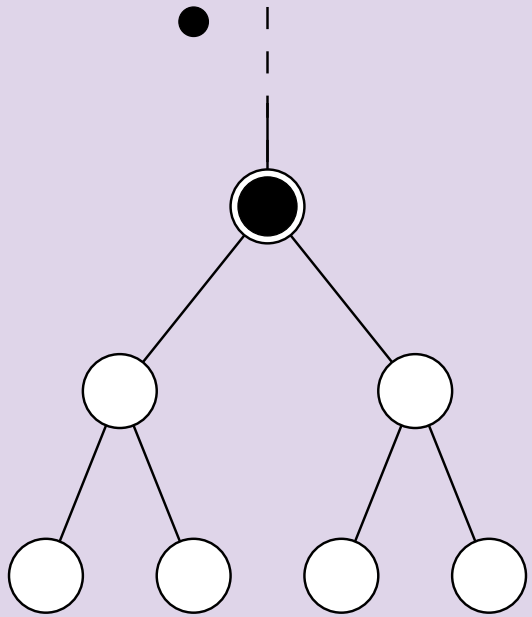


Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves

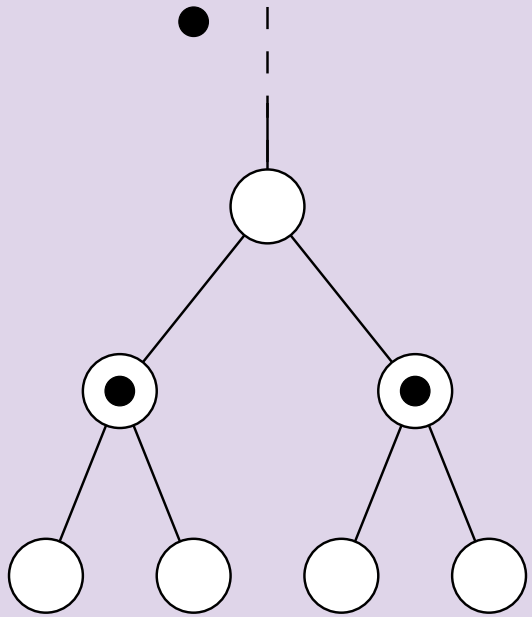


Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves

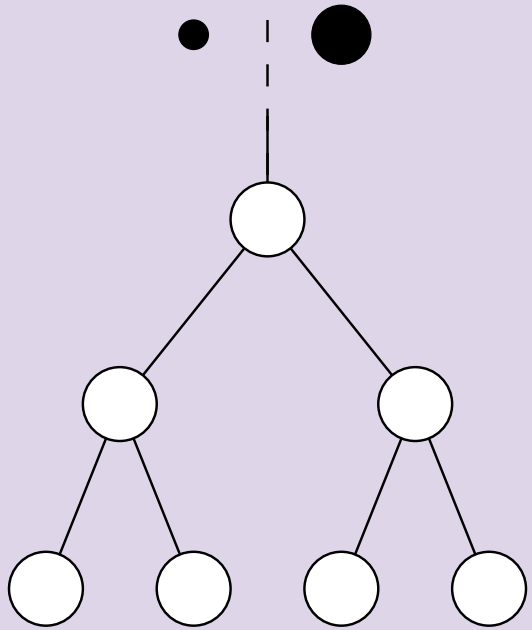


Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves

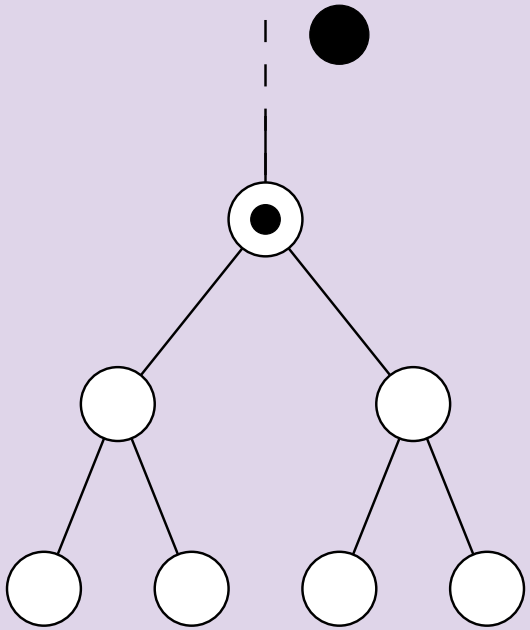


Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves

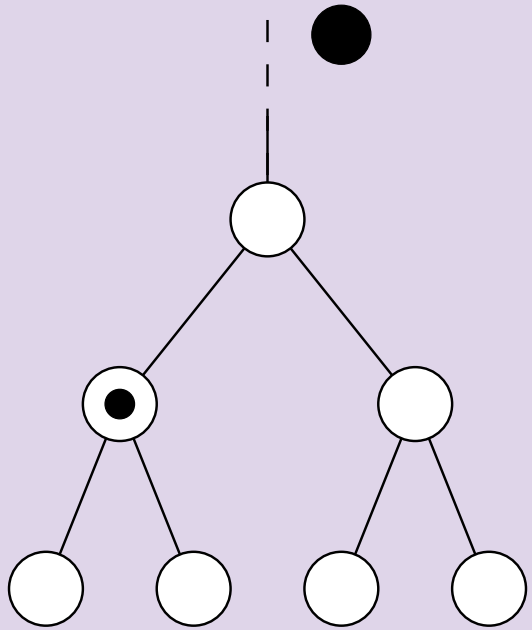


Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves

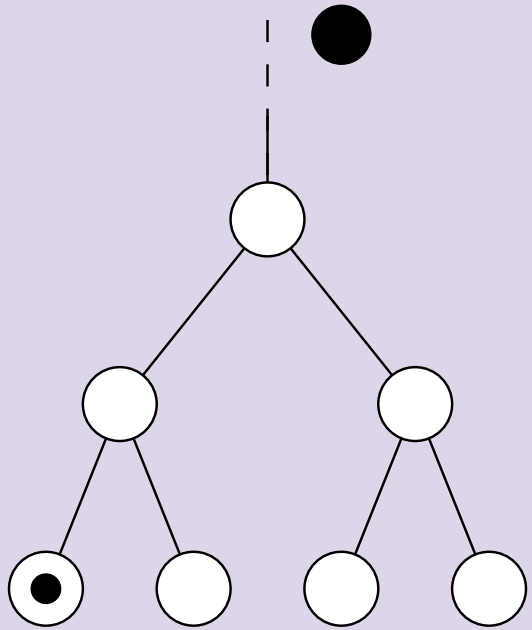


Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves

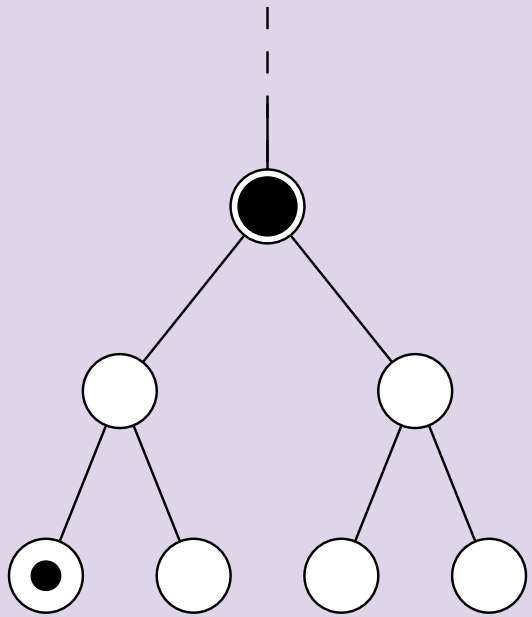


Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves

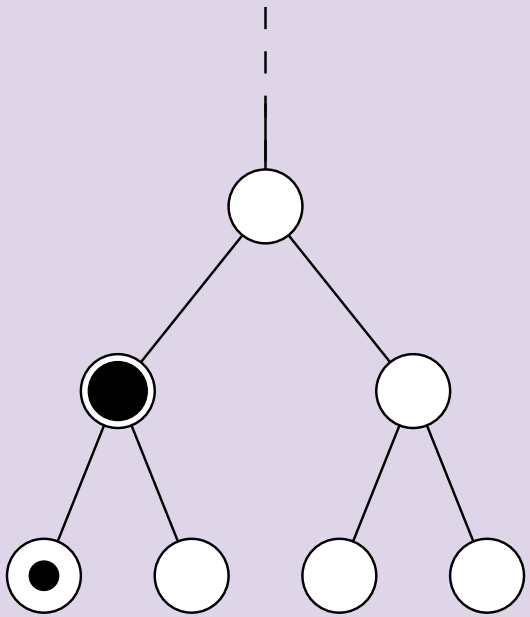


Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves

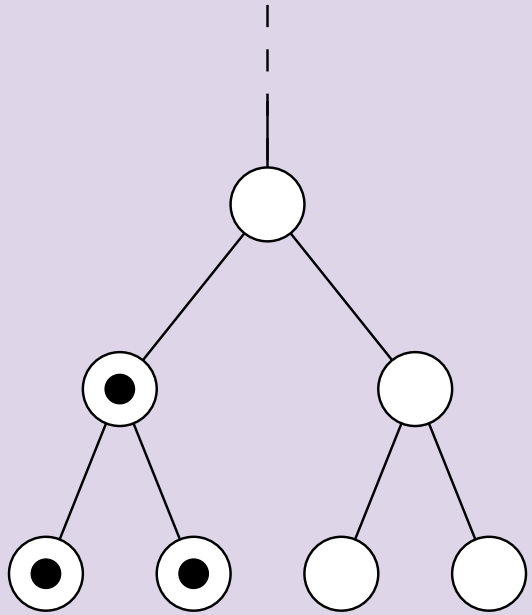


Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves

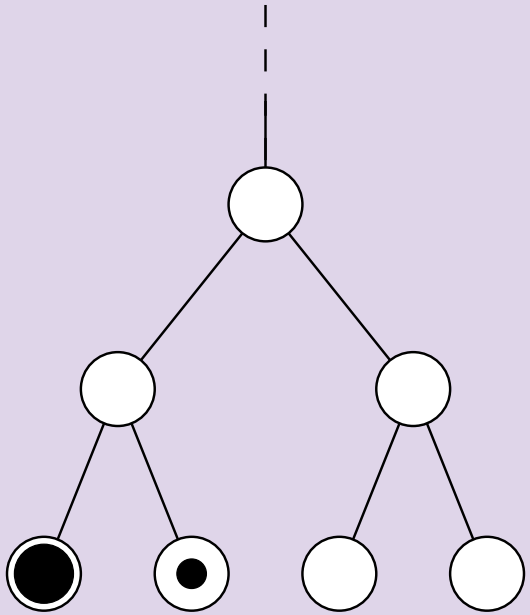


Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves

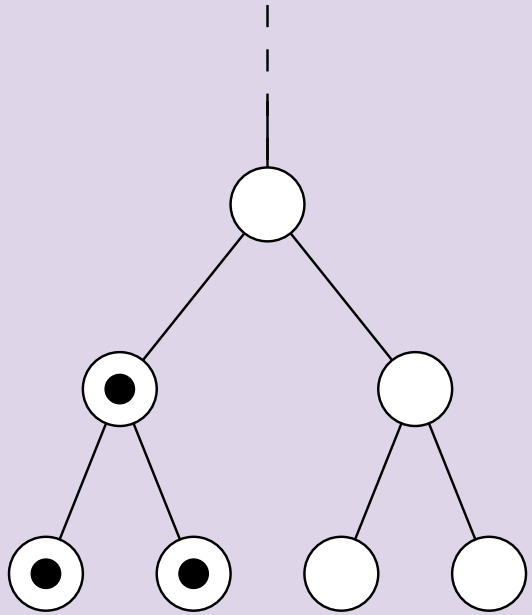


Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves

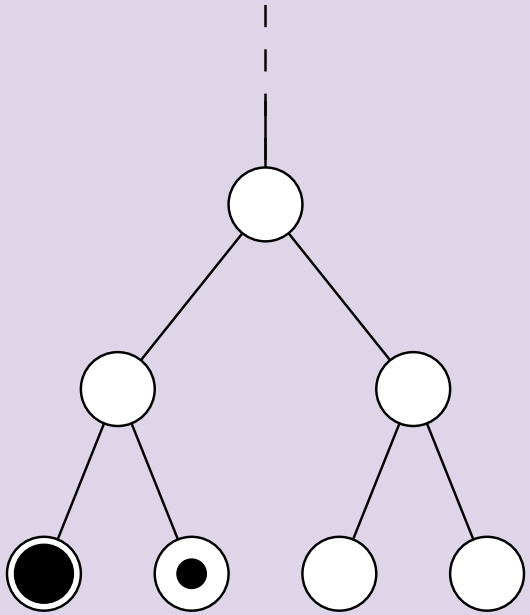


Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves

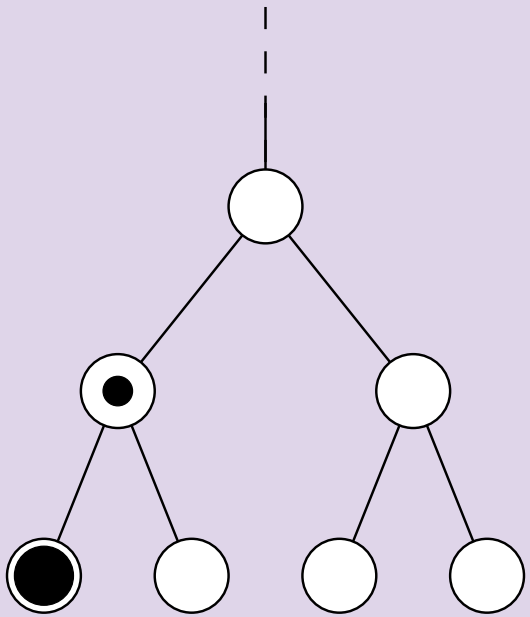


Use the counter value to determine the next leaf/pair of leaves to be explored

Phase 3

Goal: Explore P'', the largest of the pieces other than Q.

Exploration of a pair of leaves



Use the counter value to determine the next leaf/pair of leaves to be explored

Remaining phases

Phase 4

Relocate the brain from Q to P''

Phase 5

Explore piece Q and stop if there are only two pieces

Phase 6

Reinitialize the brain and relocate the exploring team in the unexplored piece

Phase 7

Explore the last piece and stop

Remaining phases

Phase 4

Relocate the brain from Q to P''

Phase 5

Explore piece Q and stop if there are only two pieces

Phase 6

Reinitialize the brain and relocate the exploring team in the unexplored piece

Phase 7

Explore the last piece and stop

Remaining phases

Phase 4

Relocate the brain from Q to P''

Phase 5

Explore piece Q and stop if there are only two pieces

Phase 6

Reinitialize the brain and relocate the exploring team in the unexplored piece

Phase 7

Explore the last piece and stop

Remaining phases

Phase 4

Relocate the brain from Q to P''

Phase 5

Explore piece Q and stop if there are only two pieces

Phase 6

Reinitialize the brain and relocate the exploring team in the unexplored piece

Phase 7

Explore the last piece and stop

A small sample of the problems to solve

- How to create a single multiplicity in Phase 1 without blocking the other robots?
- How to break symmetries using the leader? (problem of trapped solitaires)
- How to move multiplicities? How to move robots to their precise targets?
- How do the leader and the other robots cross each other in path-like trees?
- Is the counter up-to-date or currently updating?
- How to remember the phase number?
- ...

A small sample of the problems to solve

- How to create a single multiplicity in Phase 1 without blocking the other robots?
- How to break symmetries using the leader? (problem of trapped solitaires)
- How to move multiplicities? How to move robots to their precise targets?
- How do the leader and the other robots cross each other in path-like trees?
- Is the counter up-to-date or currently updating?
- How to remember the phase number?
- ...

A small sample of the problems to solve

- How to create a single multiplicity in Phase 1 without blocking the other robots?
- How to break symmetries using the leader? (problem of trapped solitaires)
- How to move multiplicities? How to move robots to their precise targets?
- How do the leader and the other robots cross each other in path-like trees?
- Is the counter up-to-date or currently updating?
- How to remember the phase number?
- ...

A small sample of the problems to solve

- How to create a single multiplicity in Phase 1 without blocking the other robots?
- How to break symmetries using the leader? (problem of trapped solitaires)
- How to move multiplicities? How to move robots to their precise targets?
- How do the leader and the other robots cross each other in path-like trees?
- Is the counter up-to-date or currently updating?
- How to remember the phase number?
- ...

A small sample of the problems to solve

- How to create a single multiplicity in Phase 1 without blocking the other robots?
- How to break symmetries using the leader? (problem of trapped solitaires)
- How to move multiplicities? How to move robots to their precise targets?
- How do the leader and the other robots cross each other in path-like trees?
- Is the counter up-to-date or currently updating?
- How to remember the phase number?
- ...

A small sample of the problems to solve

- How to create a single multiplicity in Phase 1 without blocking the other robots?
- How to break symmetries using the leader? (problem of trapped solitaires)
- How to move multiplicities? How to move robots to their precise targets?
- How do the leader and the other robots cross each other in path-like trees?
- Is the counter up-to-date or currently updating?
- How to remember the phase number?
- ...

A small sample of the problems to solve

- How to create a single multiplicity in Phase 1 without blocking the other robots?
- How to break symmetries using the leader? (problem of trapped solitaires)
- How to move multiplicities? How to move robots to their precise targets?
- How do the leader and the other robots cross each other in path-like trees?
- Is the counter up-to-date or currently updating?
- How to remember the phase number?
- ...

Conclusion and perspectives

Open problem

- Cycles: $\Theta(\log n)$
- Trees of maximum degree 3: $\Theta(\log n / \log \log n)$
- Arbitrary trees: $\Theta(n)$

What about arbitrary graphs (of maximum degree 3)?

Perspectives

- Limited visibility
- Fault tolerant protocols

Conclusion and perspectives

Open problem

- Cycles: $\Theta(\log n)$
- Trees of maximum degree 3: $\Theta(\log n / \log \log n)$
- Arbitrary trees: $\Theta(n)$

What about arbitrary graphs (of maximum degree 3)?

Perspectives

- Limited visibility
- Fault tolerant protocols