

Graph Exploration by Automata

David Ilcinkas

LRI, Université Paris-Sud, France

FRAGILE meeting
Jan. 26, 2005

Graph exploration

Goal

A mobile entity has to **traverse** every edge of an **unknown anonymous** graph.

Motivation

- exploration of environments unreachable by humans
- network maintenance
- map drawing

Graph exploration

Goal

A mobile entity has to **traverse** every edge of an **unknown anonymous** graph.

Motivation

- exploration of environments unreachable by humans
- network maintenance
- map drawing

Unknown, anonymous

Unknown

- Unknown topology
- Unknown size (no upper bound)

Anonymous

- No node labeling
- Local edge labeling

Unknown, anonymous

Unknown

- Unknown topology
- Unknown size (no upper bound)

Anonymous

- No node labeling
- Local edge labeling

Unknown, anonymous

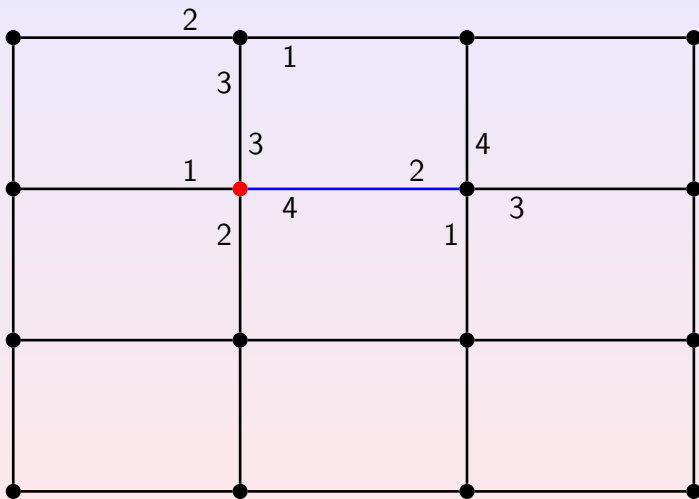
Unknown

- Unknown topology
- Unknown size (no upper bound)

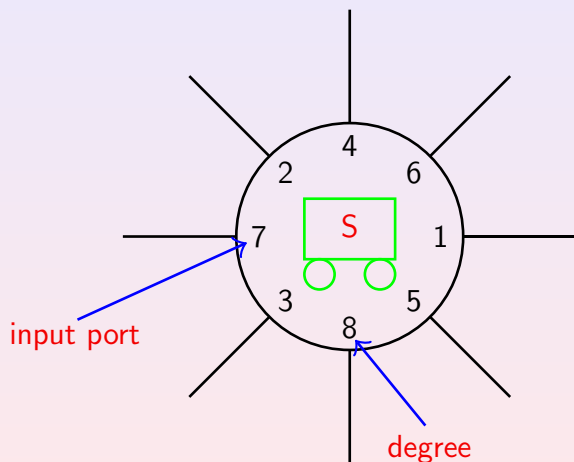
Anonymous

- No node labeling
- Local edge labeling

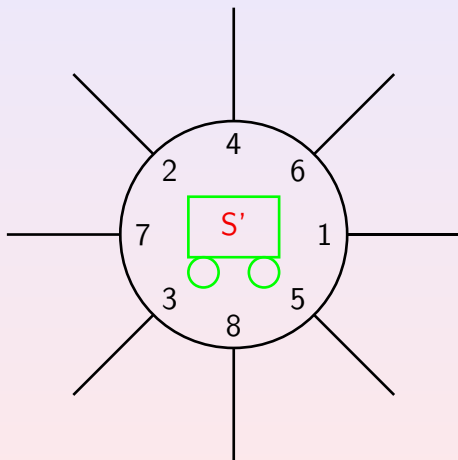
Example of an anonymous graph



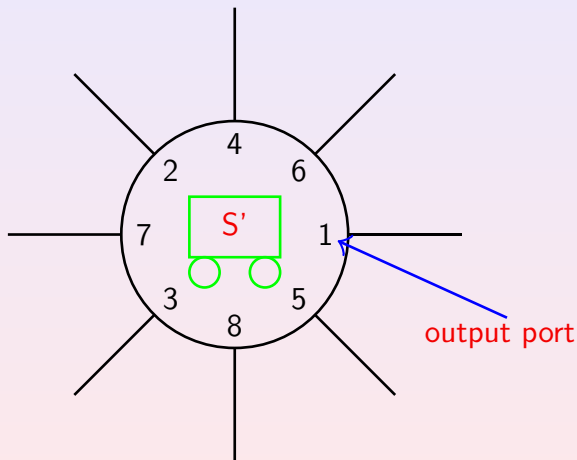
Input of the automaton



Input of the automaton



Input of the automaton



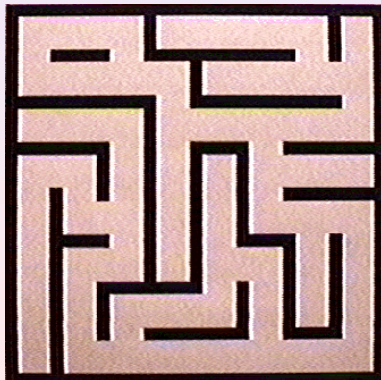
Outline

- 1 Introduction
- 2 Feasibility
 - Labyrinths
 - Graphs
 - Hydra
- 3 Minimizing memory
- 4 Other models
- 5 Bonus

Particular case: Labyrinths

Definition

- two-dimensional, obstructed chess-board
- directions are known: North, South, East, West



Results on labyrinths

Budach, Math. Nachrichten 1978

Automata and Labyrinths

\nexists universal finite automaton

Blum, Kozen, FOCS 1978

On the power of the compass

\exists a universal two-pebble automaton

\exists two cooperative automata that can explore all labyrinths

\nexists two finite automata that can explore all graphs

Hoffmann, FCT 1981

One pebble does not suffice to search plane labyrinths

\nexists universal one-pebble automaton for labyrinths

Results on labyrinths

Budach, Math. Nachrichten 1978

Automata and Labyrinths

\nexists universal finite automaton

Blum, Kozen, FOCS 1978

On the power of the compass

\exists a universal two-pebble automaton

\exists two cooperative automata that can explore all labyrinths

\nexists two finite automata that can explore all graphs

Hofmann, FCT 1981

One pebble does not suffice to search plane labyrinths

\nexists universal one-pebble automaton for labyrinths

Results on labyrinths

Budach, Math. Nachrichten 1978

Automata and Labyrinths

∄ universal finite automaton

Blum, Kozen, FOCS 1978

On the power of the compass

∃ a universal two-pebble automaton

∃ two cooperative automata that can explore all labyrinths

∄ two finite automata that can explore all graphs

Hofmann, FCT 1981

One pebble does not suffice to search plane labyrinths

∄ universal one-pebble automaton for labyrinths

Results on labyrinths

Budach, Math. Nachrichten 1978

Automata and Labyrinths

∄ universal finite automaton

Blum, Kozen, FOCS 1978

On the power of the compass

∃ a universal two-pebble automaton

∃ two cooperative automata that can explore all labyrinths

∄ two finite automata that can explore all graphs

Hofmann, FCT 1981

One pebble does not suffice to search plane labyrinths

∄ universal one-pebble automaton for labyrinths

Results on labyrinths

Budach, Math. Nachrichten 1978

Automata and Labyrinths

⊘ universal finite automaton

Blum, Kozen, FOCS 1978

On the power of the compass

∃ a universal two-pebble automaton

∃ two cooperative automata that can explore all labyrinths

⊘ two finite automata that can explore all graphs

Hoffmann, FCT 1981

One pebble does not suffice to search plane labyrinths

⊘ universal one-pebble automaton for labyrinths

Arbitrary graphs

Budach, Math. Nachrichten, 1978

Automata and Labyrinths

No finite automaton can explore all labyrinths \Rightarrow graphs.

Roth, Acta Informatica, 1980

Automaten in planaren Graphen

No finite team of finite cooperative automata can explore all graphs.

Arbitrary graphs

Budach, Math. Nachrichten, 1978

Automata and Labyrinths

No finite automaton can explore all labyrinths \Rightarrow graphs.

Rollik, Acta Informatica, 1980

Automaten in planaren Graphen

No finite team of finite cooperative automata can explore all graphs.

How to trap an automaton (Rollik)

Trap for a specific automaton

- finite automaton \iff behaviour becomes periodic
- trap the automaton in a cycle

Trap for k non-cooperative automata

- defined recursively
- place a trap for $k - 1$ in every edge

Trap for k cooperative automata

- again defined recursively
- place a trap for $k - 1$ in every edge
- much more complicated meta-structure

How to trap an automaton (Rollik)

Trap for a specific automaton

- finite automaton \iff behaviour becomes periodic
- trap the automaton in a cycle

Trap for k non-cooperative automata

- defined recursively
- place a trap for $k - 1$ in every edge

Trap for k cooperative automata

- again defined recursively
- place a trap for $k - 1$ in every edge
- much more complicated meta-structure

How to trap an automaton (Rollik)

Trap for a specific automaton

- finite automaton \iff behaviour becomes periodic
- trap the automaton in a cycle

Trap for k non-cooperative automata

- defined recursively
- place a trap for $k - 1$ in every edge

Trap for k cooperative automata

- again defined recursively
- place a trap for $k - 1$ in every edge
- much more complicated meta-structure

Cooperation models

Local cooperation (Rollik)

Robots only communicate if they are at the **same node** at the **same time**.

Global cooperation

Robots **always communicate**.

Especially, robots know when other robots meet.

Cooperation models

Local cooperation (Rollik)

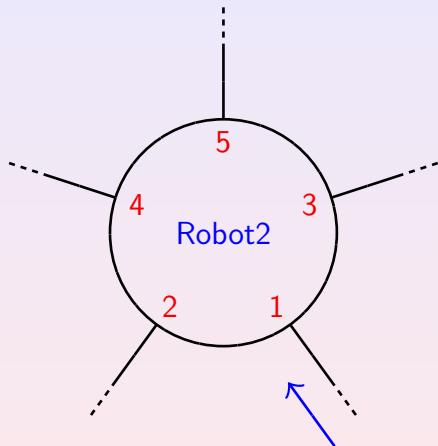
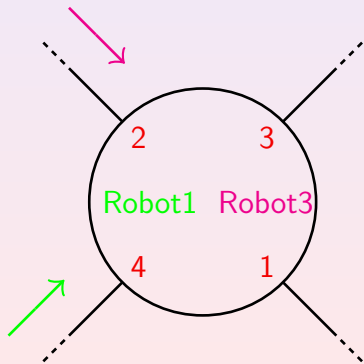
Robots only communicate if they are at the **same node** at the **same time**.

Global cooperation

Robots **always communicate**.

Especially, robots know when other robots meet.

Example



Hydra

Full knowledge \iff Single state

Definition: Hydra

Multi-headed automaton

k globally-cooperative automata \iff k -head hydra

Hydra

Full knowledge \iff Single state

Definition: Hydra

Multi-headed automaton

k globally-cooperative automata \iff *k*-head hydra

Hydra

Full knowledge \iff Single state

Definition: Hydra

Multi-headed automaton

k globally-cooperative automata \iff k -head hydra

Power of the hydra

Fraignaud, Ilcinkas, Markou, Pelc

Power of communication in cooperative exploration of graphs

-
-
-
-
-

Conjecture : No hydra is universal

Power of the hydra

Fraigniaud, Ilcinkas, Markou, Pelc

Power of communication in cooperative exploration of graphs

- a 2-head hydra $\not\equiv$ a finite team of automata
- a k -head hydra \succ a team of k locally-cooperative automata
- a 2-head hydra is not universal
- a 3-head hydra \succ a 2-head hydra
- a 3-head hydra is not universal

Conjecture : No hydra is universal

Power of the hydra

Fraignaud, Ilcinkas, Markou, Pelc

Power of communication in cooperative exploration of graphs

- a 2-head hydra $\not\asymp$ a finite team of automata
- a k -head hydra \succ a team of k locally-cooperative automata
- a 2-head hydra is not universal
- a 3-head hydra \succ a 2-head hydra
- a 3-head hydra is not universal

Conjecture : No hydra is universal

Power of the hydra

Fraigniaud, Ilcinkas, Markou, Pelc

Power of communication in cooperative exploration of graphs

- a 2-head hydra $\not\asymp$ a finite team of automata
- a k -head hydra \succ a team of k locally-cooperative automata
- a 2-head hydra is not universal
- a 3-head hydra \succ a 2-head hydra
- a 3-head hydra is not universal

Conjecture : No hydra is universal

Power of the hydra

Fraigniaud, Ilcinkas, Markou, Pelc

Power of communication in cooperative exploration of graphs

- a 2-head hydra $\not\preceq$ a finite team of automata
- a k -head hydra \succ a team of k locally-cooperative automata
- a 2-head hydra is not universal
- a 3-head hydra \succ a 2-head hydra
- a 3-head hydra is not universal

Conjecture : No hydra is universal

Power of the hydra

Fraignaud, Ilcinkas, Markou, Pelc

Power of communication in cooperative exploration of graphs

- a 2-head hydra $\not\preceq$ a finite team of automata
- a k -head hydra \succ a team of k locally-cooperative automata
- a 2-head hydra is not universal
- a 3-head hydra \succ a 2-head hydra
- a 3-head hydra is not universal

Conjecture : No hydra is universal

Power of the hydra

Fraignaud, Ilcinkas, Markou, Pelc

Power of communication in cooperative exploration of graphs

- a 2-head hydra $\not\asymp$ a finite team of automata
- a k -head hydra \succ a team of k locally-cooperative automata
- a 2-head hydra is not universal
- a 3-head hydra \succ a 2-head hydra
- a 3-head hydra is not universal

Conjecture : No hydra is universal

Power of the hydra

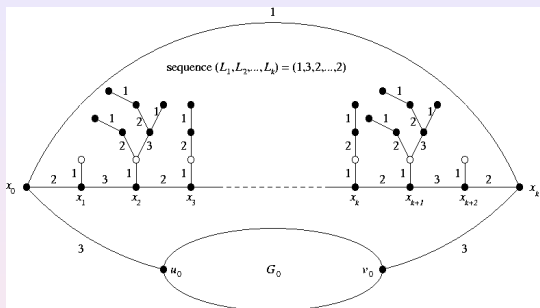
Fraignaud, Ilcinkas, Markou, Pelc

Power of communication in cooperative exploration of graphs

- a 2-head hydra $\not\preceq$ a finite team of automata
- a k -head hydra \succ a team of k locally-cooperative automata
- a 2-head hydra is not universal
- a 3-head hydra \succ a 2-head hydra
- a 3-head hydra is not universal

Conjecture : No hydra is universal

Hydra vs team



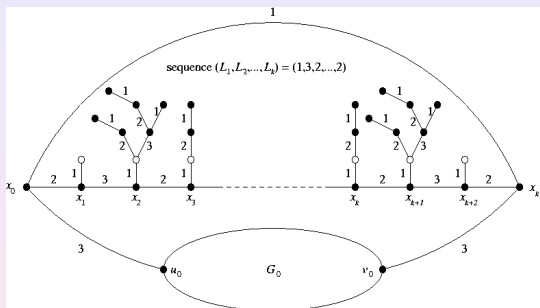
Conclusion

a 2-head hydra $\not\leq$ a finite team of automata

Corollary

a k -head hydra \succ a team of k locally-cooperative automata

Hydra vs team



Conclusion

a 2-head hydra $\not\leq$ a finite team of automata

Corollary

a k -head hydra \succ a team of k locally-cooperative automata

Trap for a 2-head hydra

d -homogeneous graph

- d -regular
- edge-colored (same label at both extremities)

In d -homogeneous graphs:
2-head hydra \equiv team of two automata

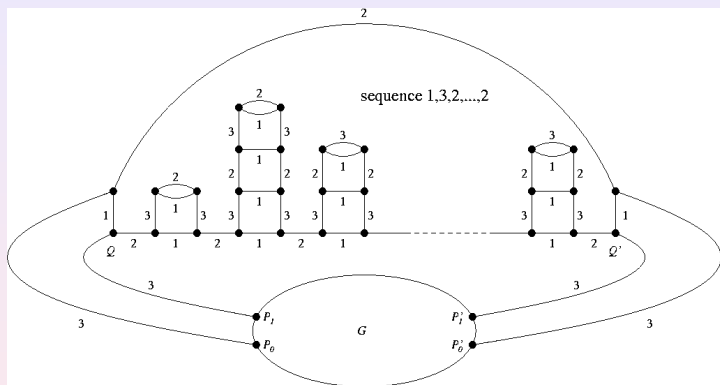
Trap for a 2-head hydra

d -homogeneous graph

- d -regular
- edge-colored (same label at both extremities)

In d -homogeneous graphs:
2-head hydra \equiv team of two automata

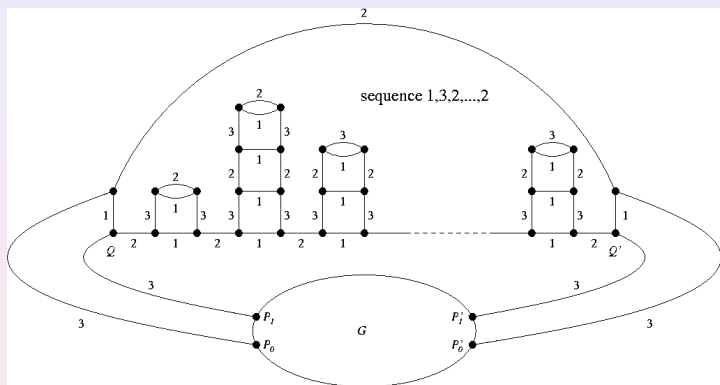
2-head hydra vs 3-head hydra



Conclusion

3-head hydra \succ 2-head hydra

2-head hydra vs 3-head hydra



Conclusion

3-head hydra \succ 2-head hydra

Trap for a 3-head hydra

- If heads are separated
 - Simulated by a bigger automaton
 - The head headless meta-structure
 - The head headless meta-structure
 - The head headless meta-structure
 - The head headless meta-structure
 - The head headless meta-structure
 - The head headless meta-structure
 - The head headless meta-structure
 - The head headless meta-structure
 - The head headless meta-structure
 - The head headless meta-structure
- If heads are not separated (bounded distance between each other)
 - Simulated by a bigger automaton
 - Heads trapped in a meta-meta-structure

Trap for a 3-head hydra

- If heads are separated
 - Two heads (together or not)
 - The third head does not help
 - Trap for a 2-head hydra
 - One head alone
 - Receive periodic information from the other heads
 - Simulated by a bigger automaton
 - Head trapped in a meta-structure
- If heads are not separated (bounded distance between each other)
 - Simulated by a bigger automaton
 - Heads trapped in a meta-meta-structure

Trap for a 3-head hydra

- If heads are separated
 - Two heads (together or not)
 - The third head does not help
 - Trap for a 2-head hydra
 - One head alone
 - Receive periodic information from the other heads
 - Simulated by a bigger automaton
 - Head trapped in a meta-structure
- If heads are not separated (bounded distance between each other)
 - Simulated by a bigger automaton
 - Heads trapped in a meta-meta-structure

Trap for a 3-head hydra

- If heads are separated
 - Two heads (together or not)
 - The third head does not help
 - Trap for a 2-head hydra
 - One head alone
 - Receive periodic information from the other heads
 - Simulated by a bigger automaton
 - Head trapped in a meta-structure
- If heads are not separated (bounded distance between each other)
 - Simulated by a bigger automaton
 - Heads trapped in a meta-meta-structure

Trap for a 3-head hydra

- If heads are separated
 - Two heads (together or not)
 - The third head does not help
 - Trap for a 2-head hydra
 - One head alone
 - Receive periodic information from the other heads
 - Simulated by a bigger automaton
 - Head trapped in a meta-structure
- If heads are not separated (bounded distance between each other)
 - Simulated by a bigger automaton
 - Heads trapped in a meta-meta-structure

Outline

- 1 Introduction
- 2 Feasibility
- 3 Minimizing memory**
 - Different tasks
 - Trees
 - General graphs
- 4 Other models
- 5 Bonus

Different tasks

- **Perpetual exploration**
- **Exploration with stop**
the robot is required to stop after completing exploration
- **Exploration with return**
the robot has to stop at its starting node
- **Mapping**
the robot has to output an edge-labeled isomorphic copy of the graph

Trees

Diks, Fraigniaud, Kranakis, Pelc, SODA 2002

Tree exploration with little memory

- Perpetual: $\Theta(\log \Delta)$ bits
- With stop: $\Omega(\log \log \log n)$ bits
- With return: $\Omega(\log n)$, $O(\log^2 n)$ bits

Arbitrary graphs

Fraignaud, Ilcinkas, Peer, Pelc, Peleg, MFCS 2004

Graph exploration by a finite automaton

Perpetual exploration:

- For any K -state automaton, there exists a trap of at most $K + 1$ nodes.
- Thus an automaton which can explore all graphs of at most K nodes has at least K states.
- DFS is space optimal: $\Theta(D \log \Delta)$ bits

Arbitrary graphs

Fraignaud, Ilcinkas, Peer, Pelc, Peleg, MFCS 2004

Graph exploration by a finite automaton

Perpetual exploration:

- For any K -state automaton, there exists a trap of at most $K + 1$ nodes.
- Thus an automaton which can explore all graphs of at most K nodes has at least K states.
- DFS is space optimal: $\Theta(D \log \Delta)$ bits

Arbitrary graphs

Fraignaud, Ilcinkas, Peer, Pelc, Peleg, MFCS 2004

Graph exploration by a finite automaton

Perpetual exploration:

- For any K -state automaton, there exists a trap of at most $K + 1$ nodes.
- Thus an automaton which can explore all graphs of at most K nodes has at least K states.
- **DFS is space optimal**: $\Theta(D \log \Delta)$ bits

DFS is optimal

Algorithm (upper bound)

- Depth-first search (**DFS**) of increasing depth
- Memory: a stack of port numbers leading to the root
→ $O(D \log \Delta)$ bits

Lower bound

Attach a tree to the trap in order to reduce the diameter.

DFS is optimal

Algorithm (upper bound)

- Depth-first search (**DFS**) of increasing depth
- Memory: a stack of port numbers leading to the root
→ $O(D \log \Delta)$ bits

Lower bound

Attach a tree to the trap in order to reduce the diameter.

Outline

- 1 Introduction
- 2 Feasibility
- 3 Minimizing memory
- 4 Other models**
 - Minimizing time
 - Other frameworks
 - Problme continu
 - UTS, UXS
- 5 Bonus

Minimizing time

Time = number of edge traversals

Dudek, Jenkin, Milos, Wilkes, IEEE TRA 1991

Robotic exploration as graph construction

- Mapping with pebbles in $O(mn)$

Deng, Mirzalan, IEEE 1996

Competitive robot mapping with homogeneous markers

- Competitive ratio: mapping / map verification
- With a single pebble, DJMW's algorithm is optimal (relaxed depth-one strategies)

Minimizing time

Time = number of edge traversals

Dudek, Jenkin, Milios, Wilkes, IEEE TRA 1991

Robotic exploration as graph construction

- Mapping with pebbles in $O(mn)$

Deng, Mirzalan, IEEE 1996

Competitive robot mapping with homogeneous markers

- Competitive ratio: mapping / map verification
- With a single pebble, DJMW's algorithm is optimal (relaxed depth-one strategies)

Minimizing time

Time = number of edge traversals

Dudek, Jenkin, Milios, Wilkes, IEEE TRA 1991

Robotic exploration as graph construction

- Mapping with pebbles in $O(mn)$

Deng, Mirzaian, IEEE 1996

Competitive robot mapping with homogeneous markers

- Competitive ratio: mapping / map verification
- With a single pebble, DJMW's algorithm is optimal (relaxed depth-one strategies)

A lot of frameworks

- Labeled graphs
- Geometric exploration
- Digraphs
- Piecemeal exploration

A lot of frameworks

- Labeled graphs
- Geometric exploration
- Digraphs
- Piecemeal exploration

A lot of frameworks

- Labeled graphs
- Geometric exploration
- Digraphs
- Piecemeal exploration

A lot of frameworks

- Labeled graphs
- Geometric exploration
- Digraphs
- Piecemeal exploration

Rooms with obstacles

Deng, Kameda, Papadimitriou, JACM 1998

How to learn an unknown environment I: the rectilinear case

-
-

Rooms with obstacles

Deng, Kameda, Papadimitriou, JACM 1998

How to learn an unknown environment I: the rectilinear case

-
-

Rooms with obstacles

Deng, Kameda, Papadimitriou, JACM 1998

How to learn an unknown environment I: the rectilinear case

- GT : **Gallery Tour**
Visit the whole museum, from entrance to exit
- WR : **Watchman's Route**
Visit the whole museum and go back to the starting point

Rooms with obstacles

Deng, Kameda, Papadimitriou, JACM 1998

How to learn an unknown environment I: the rectilinear case

- GT : **Gallery Tour**
Visit the whole museum, from entrance to exit
- WR : **Watchman's Route**
Visit the whole museum and go back to the starting point

UTS, UXS

UTS = Universal Traversal Sequence

Sequence of port numbers, i.e., of edges to traverse

UXS = Universal Exploration Sequence

Sequence of increments of port numbers

"0" means "backtrack"

"1" means "take the first to the right"

Goal

- to minimize the length of the sequence
- to minimize the amount of memory used to construct it

UTS, UXS

UTS = Universal Traversal Sequence

Sequence of port numbers, i.e., of edges to traverse

UXS = Universal Exploration Sequence

Sequence of increments of port numbers

“0” means “backtrack”

“1” means “take the first to the right”

Goal

- to minimize the length of the sequence
- to minimize the amount of memory used to construct it

UTS, UXS

UTS = Universal Traversal Sequence

Sequence of port numbers, i.e., of edges to traverse

UXS = Universal Exploration Sequence

Sequence of increments of port numbers

“0” means “backtrack”

“1” means “take the first to the right”

Goal

- to minimize the length of the sequence
- to minimize the amount of memory used to construct it

Outline

- 1 Introduction
- 2 Feasibility
- 3 Minimizing memory
- 4 Other models
- 5 Bonus**
 - Base
 - Three colors
 - Only two colors ?

Coloring nodes

An oracle colors (labels) the nodes to help the automaton.

Basic idea

spanning tree: the label tells which edges are in the tree

Enhanced labeling

only edge leading to the parent $\rightarrow \Delta$ colors

Coloring nodes

An oracle colors (labels) the nodes to help the automaton.

Basic idea

spanning tree: the label tells which edges are in the tree

Enhanced labeling

only edge leading to the parent $\rightarrow \Delta$ colors

Coloring nodes

An oracle colors (labels) the nodes to help the automaton.

Basic idea

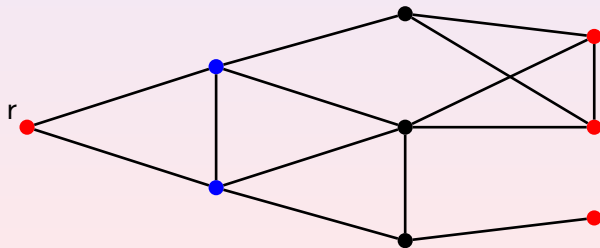
spanning tree: the label tells which edges are in the tree

Enhanced labeling

only edge leading to the parent \rightarrow Δ colors

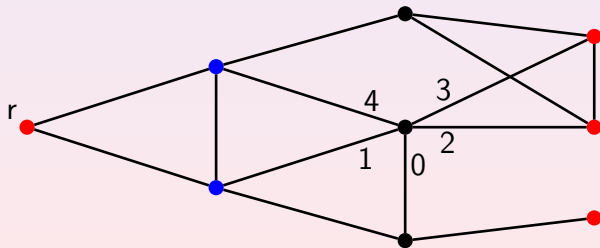
Three colors are enough

- choose arbitrarily a node as the root
- colors all nodes according to their distance d to the root
 - distance $d \cong 0[n]$ red
 - distance $d \cong 1[n]$ blue
 - distance $d \cong 2[n]$ black



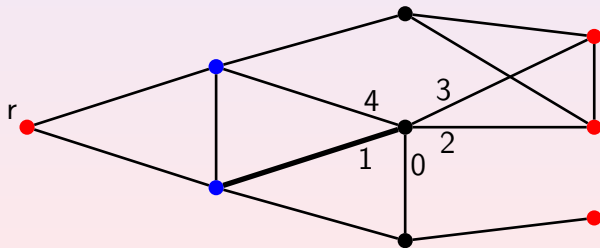
Three colors are enough

- choose arbitrarily a node as the root
- colors all nodes according to their distance d to the root
 - distance $d \cong 0[n]$ red
 - distance $d \cong 1[n]$ blue
 - distance $d \cong 2[n]$ black



Three colors are enough

- choose arbitrarily a node as the root
- colors all nodes according to their distance d to the root
 - distance $d \cong 0[n]$ red
 - distance $d \cong 1[n]$ blue
 - distance $d \cong 2[n]$ black



Only two colors ?

- Layer 1: red
- Layer 2: blue
- Layer 3: red
- Layer 4: red
- Layer 5: red
- Layer 6: blue
- Layer 7: blue
- Layer 8: blue

Outline

- 1 Introduction
- 2 Feasibility
- 3 Minimizing memory
- 4 Other models
- 5 Bonus
- 6 Conclusion**

Open problems

Hydra

- $\forall k$ $(k + 1)$ -head hydra \succ k -head hydra ?
- Is there a k such that the k -head hydra is universal ?

USTCON

- Is USTCON in log-space ($SL = L$) ? **YES**
- Better understanding of UTS, UXS, automata, hydra.

Open problems

Hydra

- $\forall k$ $(k + 1)$ -head hydra \succ k -head hydra ?
- Is there a k such that the k -head hydra is universal ?

USTCON

- Is USTCON in log-space ($SL = L$) ? YES
- Better understanding of UTS, UXS, automata, hydra.

Open problems

Hydra

- $\forall k$ $(k + 1)$ -head hydra \succ k -head hydra ?
- Is there a k such that the k -head hydra is universal ?

USTCON

- Is USTCON in log-space ($SL = L$) ? **YES**
- Better understanding of UTS, UXS, automata, hydra.