Graph Exploration by Automata

David Ilcinkas

LRI, Université Paris-Sud, France

Oct. 4, 2004
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 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
Outline

1. Introduction
2. Feasibility
   - Labyrinths
   - Graphs
   - Hydra
3. Minimizing memory
4. Minimizing time
5. Bonus
6. Conclusion
Particular case: Labyrinths

Definition

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

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
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

Hoffmann, 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

Hoffmann, 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
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

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.
Arbitrary graphs

Automata and Labyrinths

No finite automaton can explore all labyrinths $\implies$ graphs.

Automaten in planaren Graphen

No finite team of finite cooperative automata can explore all graphs.
How to trap an automaton

**Trap for a specific automaton**

- finite automaton $\implies$ 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

Trap for a specific automaton
- finite automaton $\implies$ 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

<table>
<thead>
<tr>
<th>Trap for a specific automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td>finite automaton $\implies$ behaviour becomes periodic</td>
</tr>
<tr>
<td>trap the automaton in a cycle</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trap for $k$ non-cooperative automata</th>
</tr>
</thead>
<tbody>
<tr>
<td>defined recursively</td>
</tr>
<tr>
<td>place a trap for $k - 1$ in every edge</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trap for $k$ cooperative automata</th>
</tr>
</thead>
<tbody>
<tr>
<td>again defined recursively</td>
</tr>
<tr>
<td>place a trap for $k - 1$ in every edge</td>
</tr>
<tr>
<td>much more complicated meta-structure</td>
</tr>
</tbody>
</table>
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

Robot 1  Robot 3
2  3
4  1

Robot 2
5
4
2
1

David Ilcinkas
Graph Exploration by Automata
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
Full knowledge $\iff$ Single state

**Definition: Hydra**

Multi-headed automaton

$k$ globally-cooperative automata $\iff$ $k$-head hydra
Fraigniaud, Ilcinkas, Markou, Pelc
Power of communication in cooperative exploration of graphs

- a 2-head hydra is not universal
- a 3-head hydra is not universal

Conjecture: No hydra is universal
Fraigniaud, Ilcinkas, Markou, Pelc
Power of communication in cooperative exploration of graphs

- a 2-head hydra $\not\geq$ 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
Fraigniaud, Ilcinkas, Markou, Pelc

Power of communication in cooperative exploration of graphs

- a 2-head hydra $\not\succeq$ 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 $\not\supset$ 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

Conjecture: No hydra is universal
Fraigniaud, Ilcinkas, Markou, Pelc
Power of communication in cooperative exploration of graphs

- a 2-head hydra $\not\succ$ 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
Fraigniaud, Ilcinkas, Markou, Pelc

Power of communication in cooperative exploration of graphs

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

Conjecture: No hydra is universal
Conclusion

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

Corollary

a $k$-head hydra $\geq$ a team of $k$ locally-cooperative automata
Hydra vs team

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

Corollary

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

a 2-head hydra \( \not\subseteq \) a finite team of automata

Corollary

a \( k \)-head hydra \( \not\supseteq \) 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**
  - 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
Traps 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. Minimizing time
5. Bonus
6. Conclusion
Different tasks

- **Perpetual exploration**
  the robot is required to stop after completing exploration

- **Exploration with stop**
  the robot has 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
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

Fraigniaud, 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.
- DFS is space optimal: $\Theta(D \log \Delta)$ bits
DFS is optimal

**Algorithm**

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

Lower bound
Attach a tree to the trap in order to reduce the diameter.
Outline

1. Introduction
2. Feasibility
3. Minimizing memory
4. Minimizing time
   - Some references
   - Other frameworks
5. Bonus
6. Conclusion
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)
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

- Piecemeal exploration
  - Digraphs
  - Geometric exploration
  - Rooms with obstacles
  - UTS, UXS
A lot of frameworks

- Piecemeal exploration
- Digraphs
  - Geometric exploration
  - Rooms with obstacles
  - UTS, UXS
A lot of frameworks

- Piecemeal exploration
- Digraphs
- Geometric exploration
- Rooms with obstacles
- UTS, UXS
A lot of frameworks

- Piecemeal exploration
- Digraphs
- Geometric exploration
- Rooms with obstacles

UTS, UXS
A lot of frameworks

- Piecemeal exploration
- Digraphs
- Geometric exploration
- Rooms with obstacles
- UTS, UXS
Outline

1. Introduction
2. Feasibility
3. Minimizing memory
4. Minimizing time
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
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
Three colors are enough

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

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

- choose arbitrarily a node as the root
- colors all nodes according to their distance $d$ to the root
  - distance $d \approx 0[n]$ red
  - distance $d \approx 1[n]$ blue
  - distance $d \approx 2[n]$ 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
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 ?
- Better understanding of UTS, UXS, automata.
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 ?
- Better understanding of UTS, UXS, automata.