

Oracle size: a new measure of difficulty

Pierre Fraigniaud¹ David Ilcinkas² Andrzej Pelc³

¹CNRS, LRI, Université Paris-Sud, France

²LRI, Université Paris-Sud, France

³Département d'informatique, Université du Québec en Outaouais, Canada

FRAGILE meeting
May 23, 2006

Oracle

Framework

- Distributed computing
- Mobile computing

Observation

The quality of the algorithmic solutions often depends on the given amount of knowledge about the topology.

Oracle

Model the amount of **knowledge** about the topology that is given to the nodes and/or the mobile agents.

Exploration of anonymous digraphs

Exploration by a robot using one pebble
(pebble = node-marker that the robot can drop at and remove from nodes during its traversal)

[Bender, Fernández, Ron, Sahai, Vadhan, Inf. & Comp. 2002]

No information

Impossible in polynomial time
(possible with $\Theta(\log \log n)$ pebbles)

Knowledge of an upper bound \hat{n} on the number of nodes
Possible in time polynomial in \hat{n}

Exploration of anonymous digraphs

Exploration by a robot using one pebble
(pebble = node-marker that the robot can drop at and remove from nodes during its traversal)

[Bender, Fernández, Ron, Sahai, Vadhan, Inf. & Comp. 2002]

No information

Impossible in polynomial time
(possible with $\Theta(\log \log n)$ pebbles)

Knowledge of an upper bound \hat{n} on the number of nodes
Possible in time polynomial in \hat{n}

Exploration of anonymous digraphs

Exploration by a robot using one pebble
(pebble = node-marker that the robot can drop at and remove from nodes during its traversal)

[Bender, Fernández, Ron, Sahai, Vadhan, Inf. & Comp. 2002]

No information

Impossible in polynomial time
(possible with $\Theta(\log \log n)$ pebbles)

Knowledge of an upper bound n on the number of nodes

Possible in time polynomial in n

Exploration of anonymous digraphs

Exploration by a robot using one pebble
(pebble = node-marker that the robot can drop at and remove from nodes during its traversal)

[Bender, Fernández, Ron, Sahai, Vadhan, Inf. & Comp. 2002]

No information

Impossible in polynomial time
(possible with $\Theta(\log \log n)$ pebbles)

Knowledge of an upper bound \hat{n} on the number of nodes

Possible in time polynomial in \hat{n}

Broadcast in radio networks

Synchronous deterministic broadcast in n -node networks of diameter D

(only its own identity)

[Clementi, Monti, Silvestri, SODA 2001] : time $\Omega(n \log D)$

of the network

[Kowalski, Pelc, to appear] : time $O(D + \log^2 n)$

Broadcast in radio networks

Synchronous deterministic broadcast in n -node networks of diameter D

No information (only its own identity)

[Clementi, Monti, Silvestri, SODA 2001] : time $\Omega(n \log D)$

of the network

[Kowalski, Pelc, to appear] : time $O(D + \log^2 n)$

Broadcast in radio networks

Synchronous deterministic broadcast in n -node networks of diameter D

No information (only its own identity)

[Clementi, Monti, Silvestri, SODA 2001] : time $\Omega(n \log D)$

Complete knowledge of the network

[Kowalski, Pelc, to appear] : time $O(D + \log^2 n)$

Knowledge of the topology within radius ρ

Wakeup in arbitrary networks

[Awerbuch, Goldreich, Peleg, Vainish, J. of ACM, 1990]:
 $\Theta(\min\{m, n^{1+\Theta(1)/\rho}\})$ messages of bounded length

Routing in Kleinberg's model

[Fraigniaud, Gavoille, Paul, PODC 2004]:
Expected number of steps of greedy routing = $f(\rho)$

Knowledge of the topology within radius ρ

Wakeup in arbitrary networks

[Awerbuch, Goldreich, Peleg, Vainish, J. of ACM, 1990]:
 $\Theta(\min\{m, n^{1+\Theta(1)/\rho}\})$ messages of bounded length

Routing in Kleinberg's model

[Fraigniaud, Gavoille, Paul, PODC 2004]:
Expected number of steps of greedy routing = $f(\rho)$

Knowledge of the topology within radius ρ

Wakeup in arbitrary networks

[Awerbuch, Goldreich, Peleg, Vainish, J. of ACM, 1990]:
 $\Theta(\min\{m, n^{1+\Theta(1)/\rho}\})$ messages of bounded length

Routing in Kleinberg's model

[Fraigniaud, Gavoille, Paul, PODC 2004]:
Expected number of steps of greedy routing = $f(\rho)$

Definition of the oracle

Definition

- The oracle provides a binary string $\mathcal{O}(G)$ to the nodes and/or to the mobile agents
- Size of an oracle: $|\mathcal{O}(G)|$

Interesting questions

What is the minimum size of an oracle permitting to solve problem \mathcal{P} (in a given amount of time) ?

Quantitative questions about the required knowledge, regardless of what kind of knowledge is supplied.

Definition of the oracle

Definition

- The oracle provides a **binary string** $\mathcal{O}(G)$ to the nodes and/or to the mobile agents
- **Size** of an oracle: $|\mathcal{O}(G)|$

Interesting questions

What is the **minimum size** of an oracle permitting to solve problem \mathcal{P} (in a given amount of time) ?

Quantitative questions about the required knowledge, regardless of what kind of knowledge is supplied.

Definition of the oracle

Definition

- The oracle provides a **binary string** $\mathcal{O}(G)$ to the nodes and/or to the mobile agents
- **Size** of an oracle: $|\mathcal{O}(G)|$

Interesting questions

What is the **minimum size of an oracle** permitting to solve problem \mathcal{P} (in a given amount of time) ?

Quantitative questions about the required knowledge, regardless of what **kind** of knowledge is supplied.

Examples

Mobile computing

Tree exploration with small competitive ratio

Distributed computing

Wakeup and broadcast in a linear number of messages

Examples

Mobile computing

Tree exploration with small competitive ratio

Distributed computing

Wakeup and broadcast in a linear number of messages

Examples

Mobile computing

Tree exploration with small competitive ratio

Distributed computing

Wakeup and broadcast in a linear number of messages

Outline

- 1 Introduction
- 2 Mobile computing: tree exploration**
- 3 Distributed computing: wakeup and broadcast
- 4 Conclusion and perspectives

Application to mobile computing

Tree exploration by a robot

Goal

To visit all nodes and traverse all edges as fast as possible

Measure

Competitive ratio of an algorithm \mathcal{A} (cf. online algorithms):

$$\frac{\text{length of the path followed by } \mathcal{A}}{\text{length of the shortest path covering the tree}}$$

(maximized over all graphs and all starting nodes)

Application to mobile computing

Tree exploration by a robot

Goal

To visit all nodes and traverse all edges as fast as possible

Measure

Competitive ratio of an algorithm \mathcal{A} (cf. online algorithms):

$$\frac{\text{length of the path followed by } \mathcal{A}}{\text{length of the shortest path covering the tree}}$$

(maximized over all graphs and all starting nodes)

Application to mobile computing

Tree exploration by a robot

Goal

To visit all nodes and traverse all edges as fast as possible

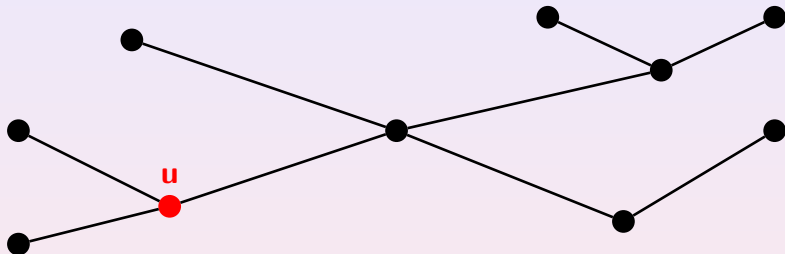
Measure

Competitive ratio of an algorithm \mathcal{A} (cf. online algorithms):

$$\frac{\text{length of the path followed by } \mathcal{A}}{\text{length of the shortest path covering the tree}}$$

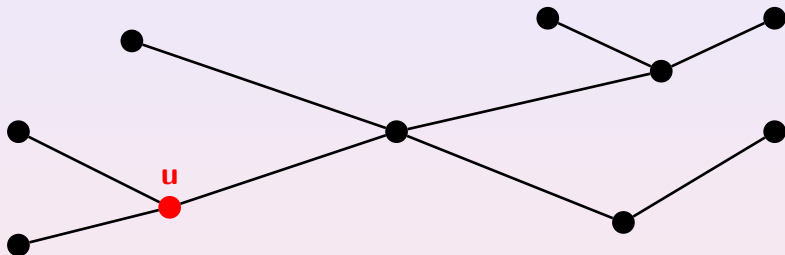
(maximized over all graphs and all starting nodes)

Example



Remarks

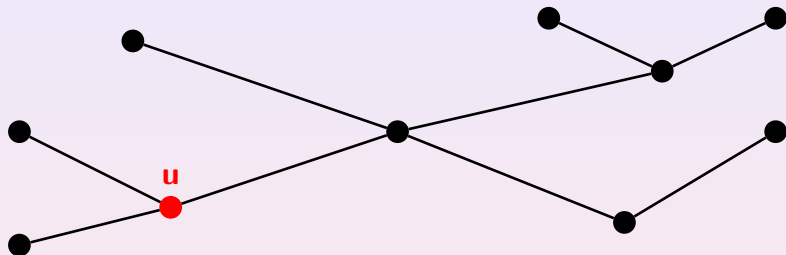
Example



Remarks

- Shortest covering walk has length $2(n - 1) - ecc(u)$
- Worst case length of a DFS traversal is $2(n - 1) - 1$

Example



Remarks

- Shortest covering walk has length $2(n - 1) - ecc(u)$
- Worst case length of a DFS traversal is $2(n - 1) - 1$

Known results

[Dessmark, Pelc, TCS, 2004]

DFS has competitive ratio **2** (holds for arbitrary graphs)

No information

No algorithm can have competitive ratio smaller than 2

Knowledge of the graph (without any label)

One can achieve competitive ratio smaller than 2 (this is not the case for arbitrary graphs)

Statement of our problem

What is the minimum size of an oracle permitting to obtain a competitive ratio smaller than 2?

Known results

[Dessmark, Pelc, TCS, 2004]

DFS has competitive ratio **2** (holds for arbitrary graphs)

No information

No algorithm can have competitive ratio smaller than 2

Knowledge of the graph (without any label)

One can achieve competitive ratio smaller than 2 (this is not the case for arbitrary graphs)

Statement of our problem

What is the minimum size of an oracle permitting to obtain a competitive ratio smaller than 2?

Known results

[Dessmark, Pelc, TCS, 2004]

DFS has competitive ratio **2** (holds for arbitrary graphs)

No information

No algorithm can have competitive ratio smaller than 2

Knowledge of the graph (without any label)

One can achieve competitive ratio smaller than 2 (this is not the case for arbitrary graphs)

Statement of our problem

What is the minimum size of an oracle permitting to obtain a competitive ratio smaller than 2?

Known results

[Dessmark, Pelc, TCS, 2004]

DFS has competitive ratio **2** (holds for arbitrary graphs)

No information

No algorithm can have competitive ratio smaller than 2

Knowledge of the graph (without any label)

One can achieve competitive ratio smaller than 2 (this is not the case for arbitrary graphs)

Statement of our problem

What is the **minimum size of an oracle** permitting to obtain a **competitive ratio smaller than 2?**

Our results (1)

Main result

Threshold $\approx \log \log D$ bits ($D = \text{diameter}$)

Notations

- $f(D)$ = maximum size of the oracle for trees of diameter at most D
- $f(D) = \log \log D - g(D)$

Precise statement

competitive ratio < 2 possible $\Leftrightarrow g$ bounded from above

Our results (1)

Main result

Threshold $\approx \log \log D$ bits ($D = \text{diameter}$)

Notations

- $f(D)$ = maximum size of the oracle for trees of diameter at most D
- $f(D) = \log \log D - g(D)$

Precise statement

competitive ratio < 2 possible $\Leftrightarrow g$ bounded from above

Our results (1)

Main result

Threshold $\approx \log \log D$ bits ($D = \text{diameter}$)

Notations

- $f(D)$ = maximum size of the oracle for trees of diameter at most D
- $f(D) = \log \log D - g(D)$

Precise statement

competitive ratio < 2 possible $\Leftrightarrow g$ bounded from above

Our results (2)

Interpretation

$$\log \log D - c \text{ bits} \longleftrightarrow D_0 \leq D \leq c' D_0$$

Complementary result

Exact value of D , but without any other information
 \Rightarrow competitive ratio at least 2

Our results (2)

Interpretation

$$\log \log D - c \text{ bits} \longleftrightarrow D_0 \leq D \leq c' D_0$$

Complementary result

Exact value of D , but without any other information
 \Rightarrow competitive ratio at least 2

Upper bound

Remark

DFS is efficient when

- diameter D is small
- number n of nodes is small
- D significantly smaller than n

Oracle

- bit b :
 - 0 \rightarrow DFS (if D significantly smaller than n)
 - 1 \rightarrow much more subtle algorithm (otherwise)
- integer k : approximation of D using $\log \log D - c$ bits

Upper bound

Remark

DFS is efficient when

- diameter D is small
- number n of nodes is small
- D significantly smaller than n

Oracle

- bit b :
 - 0 \rightarrow DFS (if D significantly smaller than n)
 - 1 \rightarrow much more subtle algorithm (otherwise)
- integer k : approximation of D using $\log \log D - c$ bits

Upper bound

Remark

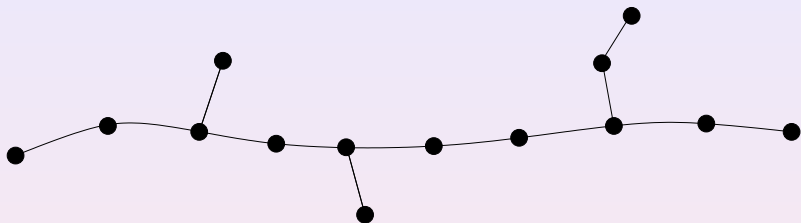
DFS is efficient when

- diameter D is small
- number n of nodes is small
- D significantly smaller than n

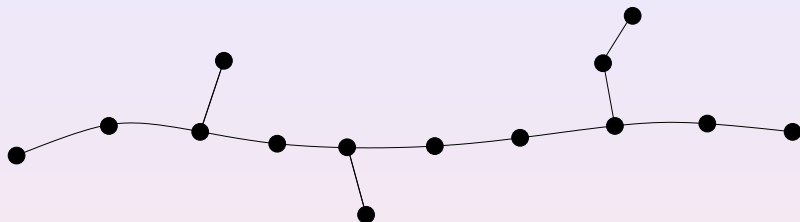
Oracle

- bit b :
 - 0 \rightarrow DFS (if D significantly smaller than n)
 - 1 \rightarrow much more subtle algorithm (otherwise)
- integer k : approximation of D using $\log \log D - c$ bits

Main ideas of the algorithm

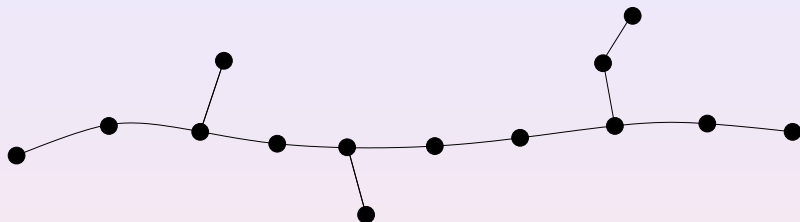


Main ideas of the algorithm



- **Local exploration** to handle the small subtrees pending from the main path
- Exploration at probing distance $0.3D_0$ to succeed in the line, where D_0 is an approximation of D

Main ideas of the algorithm



- **Local exploration** to handle the small subtrees pending from the main path
- Exploration at **probing distance $0.3D_0$** to succeed in the line, where D_0 is an approximation of D

Lower bound

Sketch of the proof

Proved for the lines

Lower bound

Sketch of the proof

Proved for the lines

- $\log \log D - g(D)$ bits, where g is unbounded from above
- \Rightarrow no good approximation of D
- \Rightarrow competitive ratio at least 2 in the lines

Lower bound

Sketch of the proof

Proved for the lines

- $\log \log D - g(D)$ bits, where g is unbounded from above
- \Rightarrow no good approximation of D
- \Rightarrow competitive ratio at least 2 in the lines

Lower bound

Sketch of the proof

Proved for the lines

- $\log \log D - g(D)$ bits, where g is unbounded from above
- \Rightarrow no good approximation of D
- \Rightarrow competitive ratio at least 2 in the lines

Outline

- 1 Introduction
- 2 Mobile computing: tree exploration
- 3 Distributed computing: wakeup and broadcast**
- 4 Conclusion and perspectives

Application to distributed computing

Goal

To disseminate a message M from a source to all the nodes of the network

- **Wakeup:** A node cannot transmit before receiving the message M
- **Broadcast:** A node can transmit without restriction

Efficiency constraint

Number of messages must be linear in n

Application to distributed computing

Goal

To disseminate a message M from a source to all the nodes of the network

- **Wakeup:** A node cannot transmit before receiving the message M
- **Broadcast:** A node can transmit without restriction

Efficiency constraint

Number of messages must be linear in n

Our results

Wakeup

Minimum oracle size is $\Theta(n \log n)$ bits.

Broadcast

- Efficient broadcast with an oracle of size $O(n)$

Our results

Wakeup

Minimum oracle size is $\Theta(n \log n)$ bits.

Broadcast

- Efficient broadcast with an oracle of size $O(n)$
- No oracle of size $o(n)$ can permit to broadcast efficiently

Our results

Wakeup

Minimum oracle size is $\Theta(n \log n)$ bits.

Broadcast

- Efficient broadcast with an oracle of size $O(n)$
- No oracle of size $o(n)$ can permit to broadcast efficiently

Strength of our results

Lower bounds

- Synchronous environment
- Node identifiers between 1 and n
- Arbitrary long messages

Upper bounds

- Asynchronous environment
- No identifiers
- Bounded-length messages

Moreover, our upper bounds are constructive

Strength of our results

Lower bounds

- Synchronous environment
- Node identifiers between 1 and n
- Arbitrary long messages

Upper bounds

- Asynchronous environment
- No identifiers
- Bounded-length messages

Moreover, our upper bounds are constructive

Strength of our results

Lower bounds

- Synchronous environment
- Node identifiers between 1 and n
- Arbitrary long messages

Upper bounds

- Asynchronous environment
- No identifiers
- Bounded-length messages

Moreover, our upper bounds are constructive

Wakeup: upper bound

Theorem

There exists an **oracle of size $O(n \log n)$** permitting the **wakeup** with a linear number of messages.

Proof

- Spanning tree rooted at the source
- Oracle: list of ports leading to the children

Wakeup: upper bound

Theorem

There exists an **oracle of size $O(n \log n)$** permitting the **wakeup** with a linear number of messages.

Proof

- Spanning tree rooted at the source
- Oracle: list of ports leading to the children

Wakeup: lower bound

Theorem

The minimum oracle size permitting the **wakeup** with a linear number of messages is $\Omega(n \log n)$.

Sketch of the proof

- Complete graph K_n with n subdivided edges

Wakeup: lower bound

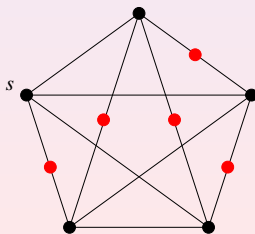
Theorem

The minimum oracle size permitting the **wakeup** with a linear number of messages is $\Omega(n \log n)$.

Sketch of the proof

- Complete graph K_n with n subdivided edges

• $\Omega(n \log n)$ bits are necessary to reduce the $\binom{n}{n}$ choices



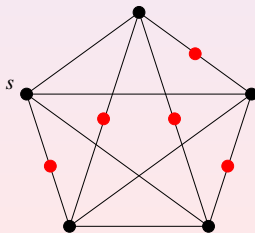
Wakeup: lower bound

Theorem

The minimum oracle size permitting the **wakeup** with a linear number of messages is $\Omega(n \log n)$.

Sketch of the proof

- Complete graph K_n with n subdivided edges
- $\Omega(n \log n)$ bits are necessary to reduce the $\binom{n^2}{n}$ choices



Broadcast: upper bound

Theorem

There exists an **oracle of size $O(n)$** permitting the **broadcast** with a linear number of messages.

Sketch of the proof

- Spanning tree of the network
- Oracle: informs one of the two extremities of a tree edge

Technical arguments $\Rightarrow O(n)$ bits are sufficient

Broadcast: upper bound

Theorem

There exists an **oracle of size $O(n)$** permitting the **broadcast** with a linear number of messages.

Sketch of the proof

- Spanning tree of the network
- Oracle: informs one of the two extremities of a tree edge

Technical arguments $\Rightarrow O(n)$ bits are sufficient

Broadcast: lower bound

Theorem

No oracle of size $o(n)$ can permit to broadcast efficiently.

Sketch of the proof

- Complete graph K_n with n/k special edges
- A k -node complete graph in each special edge

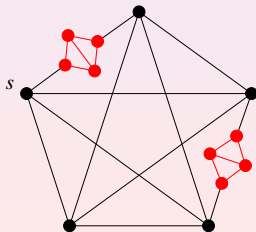
Broadcast: lower bound

Theorem

No oracle of size $o(n)$ can permit to broadcast efficiently.

Sketch of the proof

- Complete graph K_n with n/k special edges
- A k -node complete graph in each special edge
- Size $\Omega(n/k)$ necessary to have less than nk messages



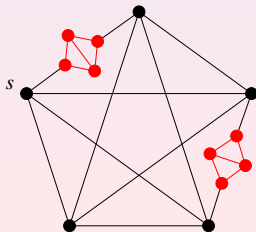
Broadcast: lower bound

Theorem

No oracle of size $o(n)$ can permit to broadcast efficiently.

Sketch of the proof

- Complete graph K_n with n/k special edges
- A k -node complete graph in each special edge
- Size $\Omega(n/k)$ necessary to have less than nk messages



Outline

- 1 Introduction
- 2 Mobile computing: tree exploration
- 3 Distributed computing: wakeup and broadcast
- 4 Conclusion and perspectives**

Conclusion and perspectives

Mobile computing

Tree exploration

- Minimum oracle size for efficient exploration: $\approx \log \log D$

Distributed computing

Wakeup and broadcast

- Constraint: linear number of messages
 - Wakeup: $\Theta(n \log n)$ bits
 - Broadcast: $O(n)$ bits
- Quantitative comparison between similar tasks

Perspectives

Oracle: new concept applicable to a broad range of problems

Conclusion and perspectives

Mobile computing

Tree exploration

- Minimum oracle size for efficient exploration: $\approx \log \log D$

Distributed computing

Wakeup and broadcast

- Constraint: linear number of messages
 - Wakeup: $\Theta(n \log n)$ bits
 - Broadcast: $O(n)$ bits
- **Quantitative comparison** between similar tasks

Perspectives

Oracle: new concept applicable to a broad range of problems

Conclusion and perspectives

Mobile computing

Tree exploration

- Minimum oracle size for efficient exploration: $\approx \log \log D$

Distributed computing

Wakeup and broadcast

- Constraint: linear number of messages
 - Wakeup: $\Theta(n \log n)$ bits
 - Broadcast: $O(n)$ bits
- **Quantitative comparison** between similar tasks

Perspectives

Oracle: **new concept applicable to a broad range of problems**