Graph searching with advice

Nicolas Nisse David Soguet

LRI, Université Paris-Sud, France.

SIROCCO, June 2007

Graph searching problem

Goal

In an undirected simple graph,

- in which edges are contaminated ;
- a team of searchers is aiming at clearing the graph.

We want to find a strategy that clears the graph
using the minimum number of searchers.

Applications

- network security,
- decontaminating a set of polluted pipes,
- ...

Graph searching problem

Goal

In an undirected simple graph,

- in which edges are contaminated ;
- a team of searchers is aiming at clearing the graph.

We want to find a strategy that clears the graph
using the minimum number of searchers.

Applications

- network security,
- decontaminating a set of polluted pipes,
- ...

Graph searching in distributed settings

Distributed graph searching

- the searchers compute themselves a strategy ;
- the strategy must be computed and performed in polynomial time.

Distributed search problem

To design a *distributed protocol* that enables the *minimum number* of searchers to clear the network in *polynomial time*.

Search strategy

The searchers move along the edges.

An edge is cleared when it is traversed by a searcher.

A clear edge e is recontaminated if a path exists between e and a contaminated edge, and no searchers stand on this path.

A strategy consists of :

- Initially, all searchers are placed at the homebase v_0 ;
- sequence of moves of searcher ; a searcher can move if it **does not imply recontamination** ;
- until the graph is clear.

$\mathbf{s}(G, v_0)$: minimum number of searchers required to clear the graph G in this way, starting from v_0.

Search strategy

The searchers move along the edges.

An edge is cleared when it is traversed by a searcher.

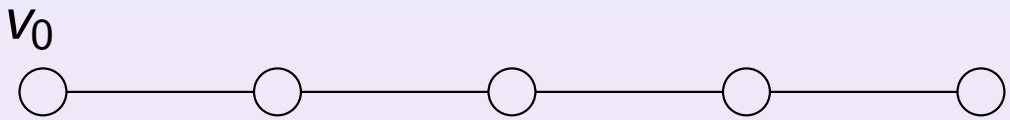
A clear edge e is recontaminated if a path exists between e and a contaminated edge, and no searchers stand on this path.

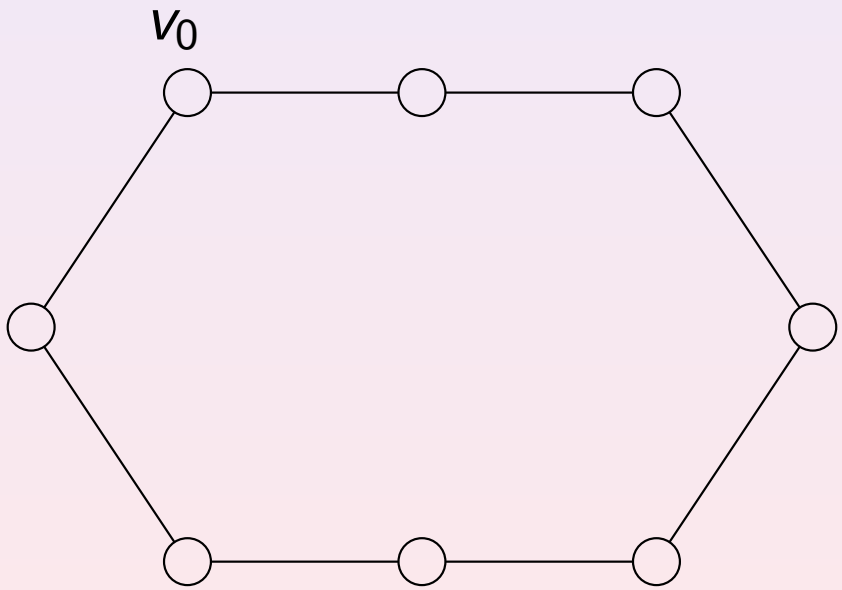
A strategy consists of :

- Initially, all searchers are placed at the homebase v_0 ;
- sequence of moves of searcher ; a searcher can move if it **does not imply recontamination** ;
- until the graph is clear.

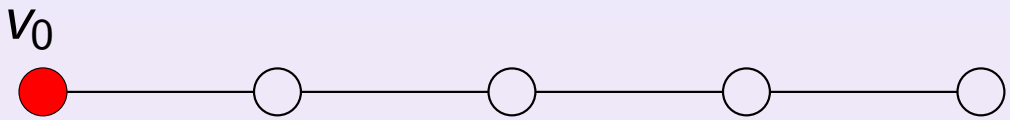
$\mathbf{s}(G, v_0)$: minimum number of searchers required to clear the graph G in this way, starting from v_0.

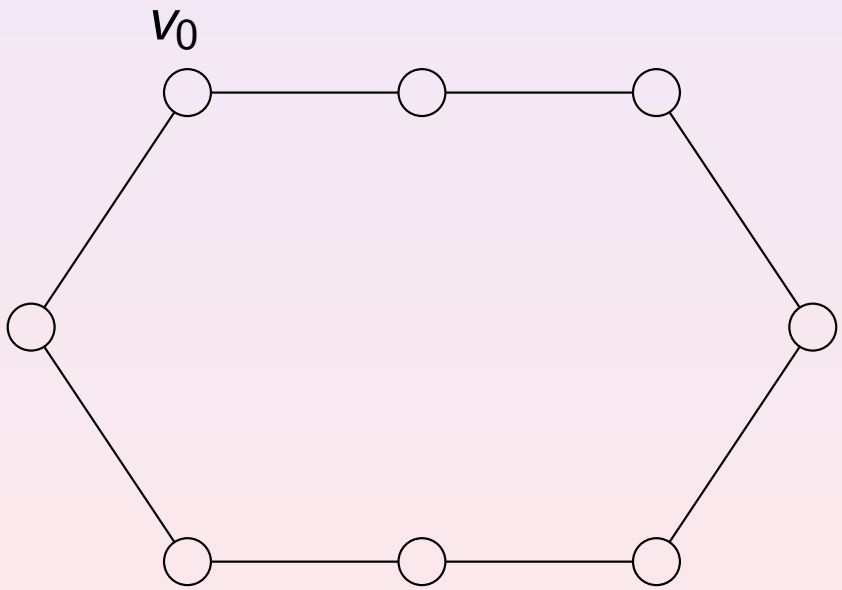
Two simple examples : the path and the ring



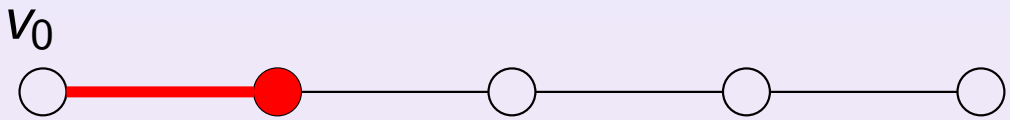


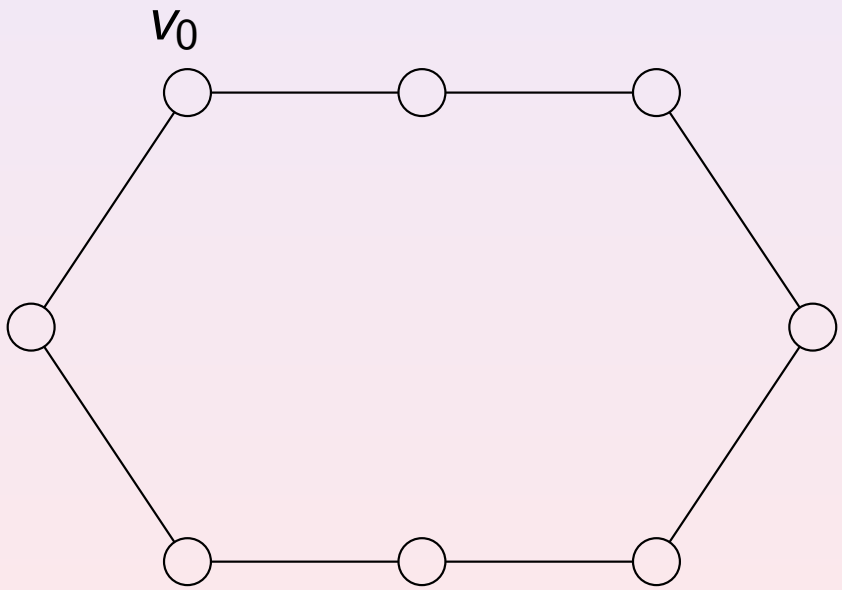
Two simple examples : the path and the ring



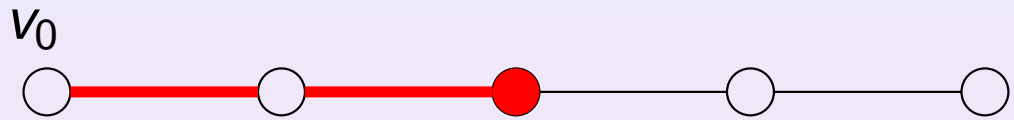


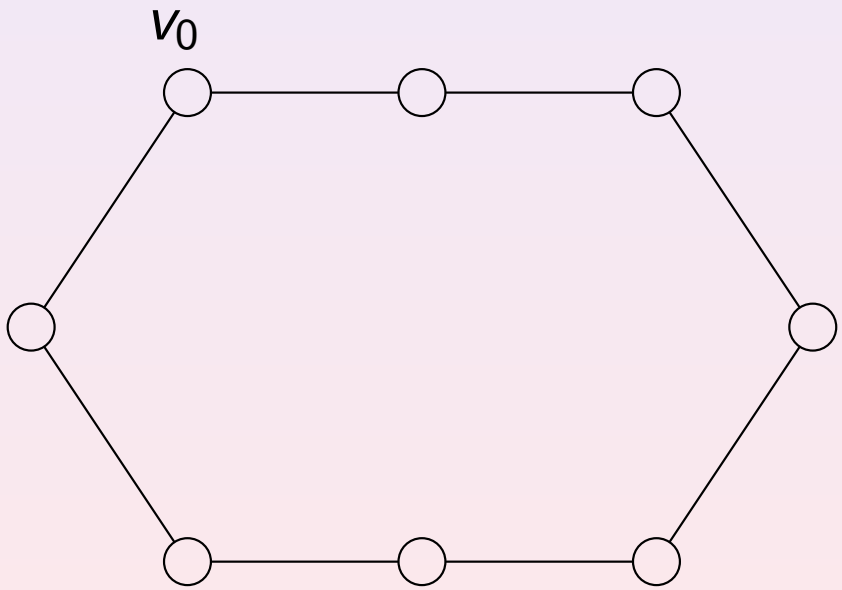
Two simple examples : the path and the ring



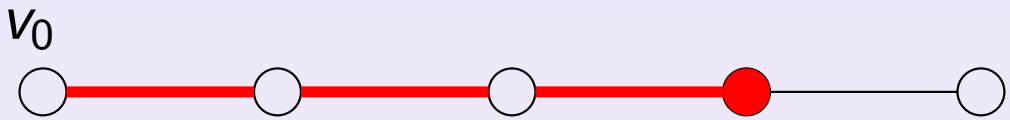


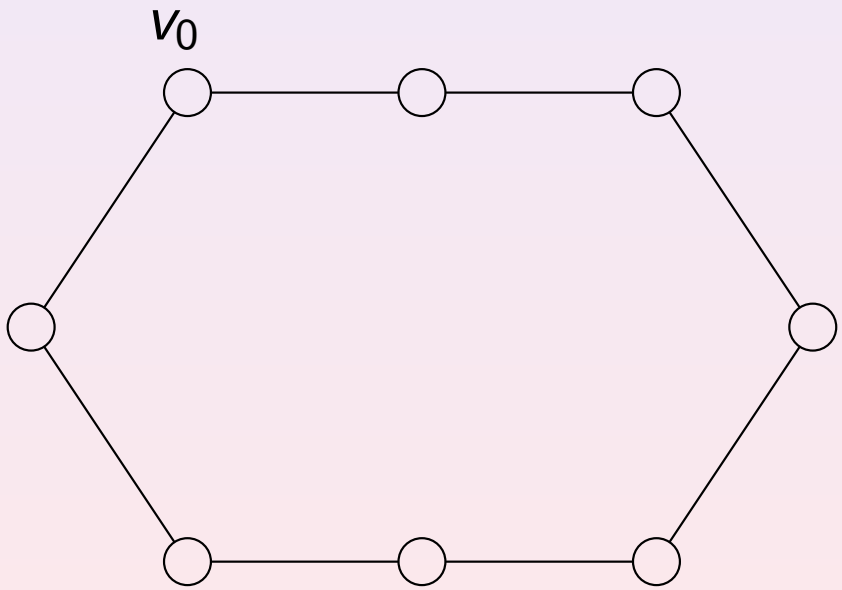
Two simple examples : the path and the ring



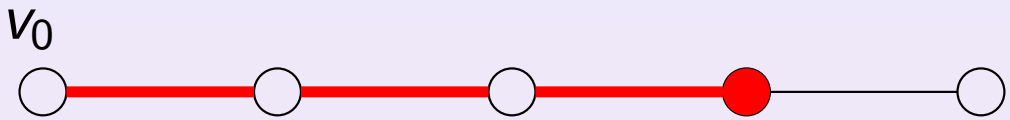


Two simple examples : the path and the ring

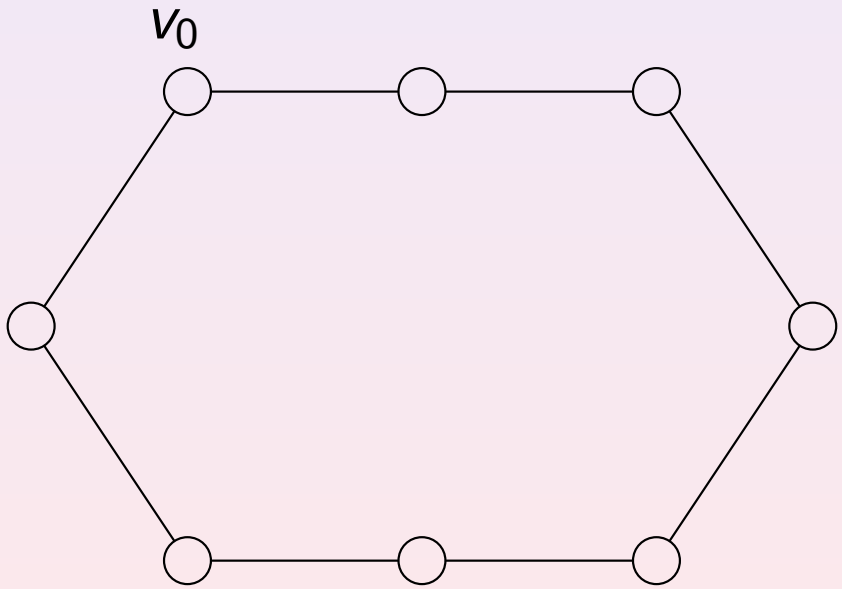




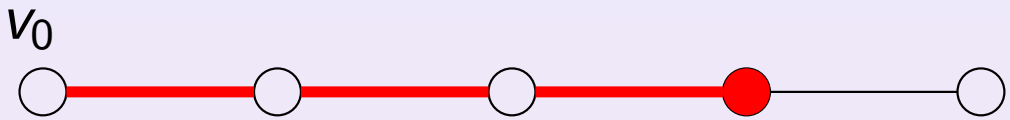
Two simple examples : the path and the ring



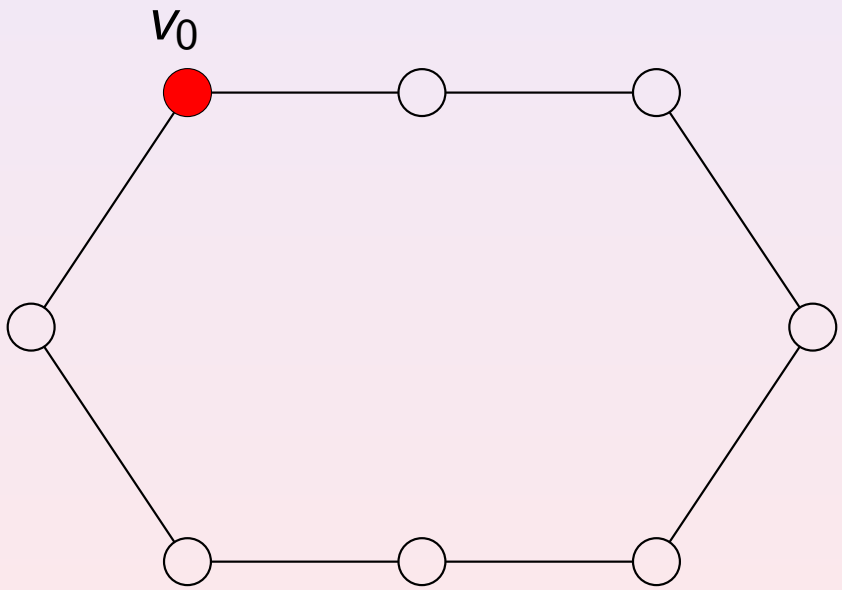
$\mathbf{s(path, v_0)=1}$



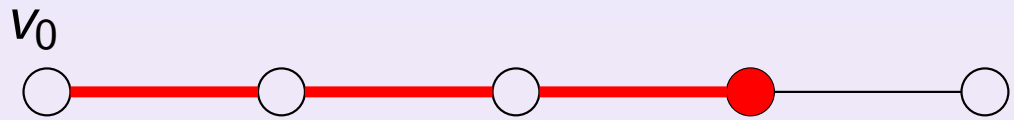
Two simple examples : the path and the ring



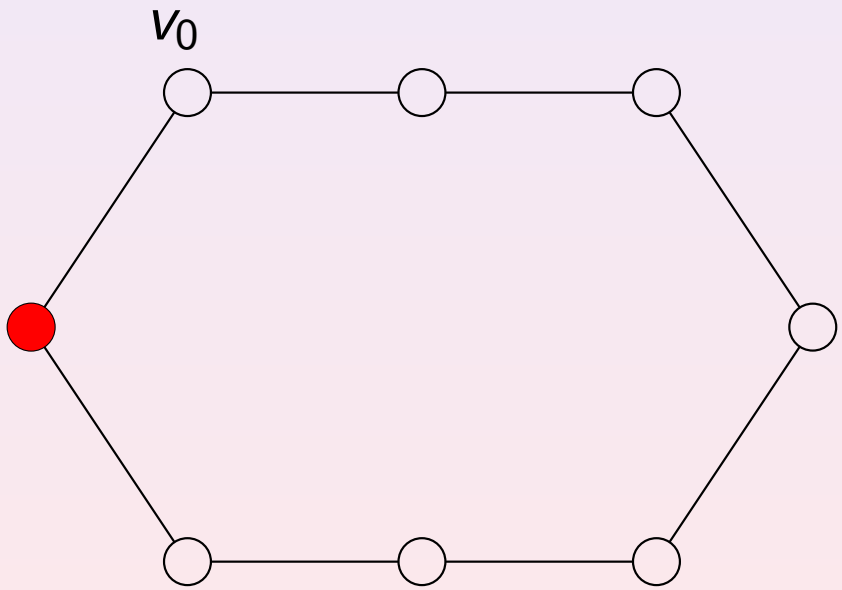
$\mathbf{s(path, v_0)=1}$



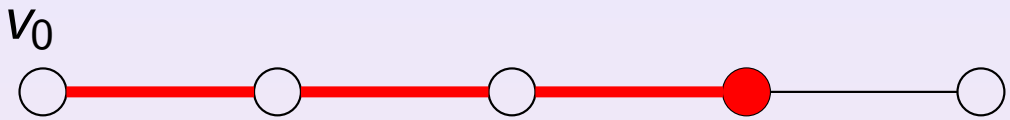
Two simple examples : the path and the ring



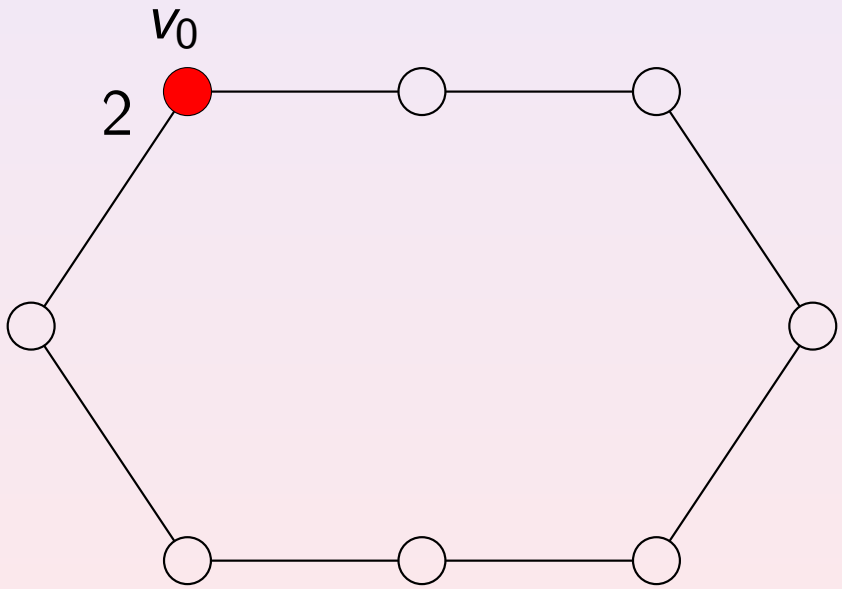
$\mathbf{s(path, v_0)=1}$



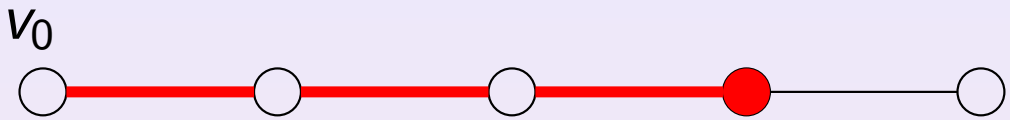
Two simple examples : the path and the ring



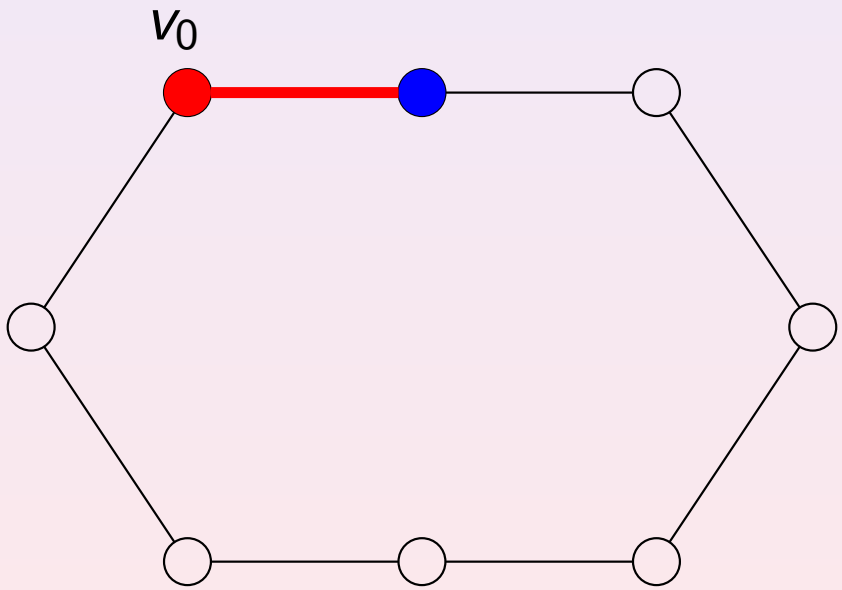
s(path,v_0)=1



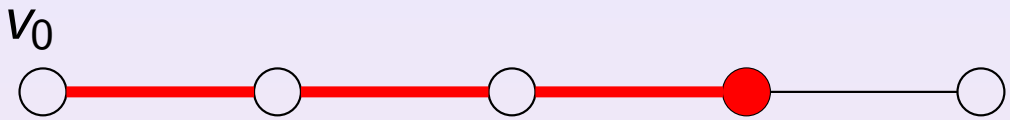
Two simple examples : the path and the ring



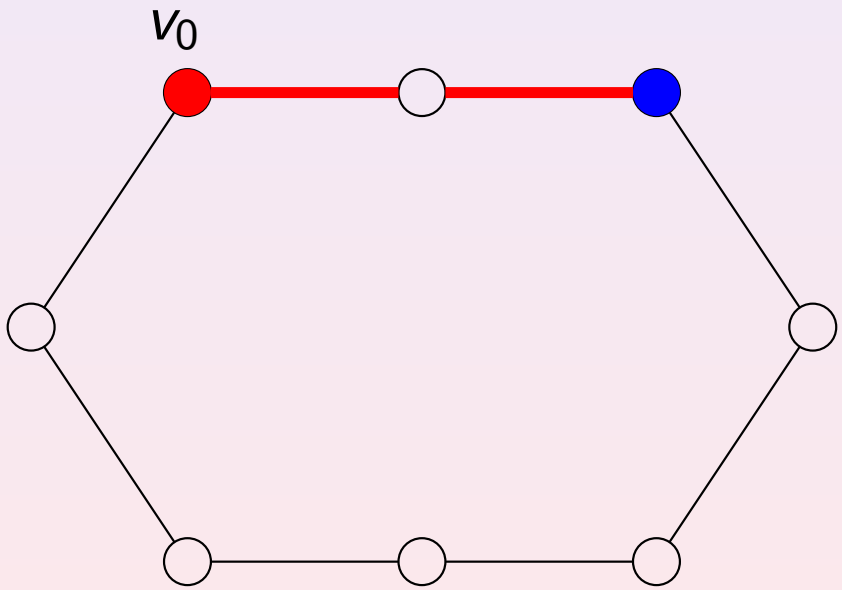
$\mathbf{s(path, v_0)=1}$



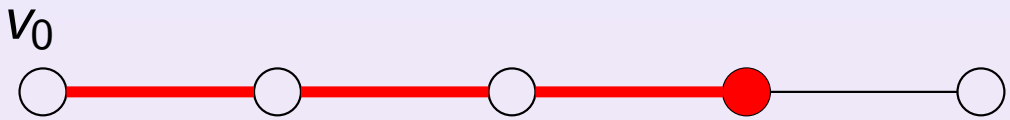
Two simple examples : the path and the ring



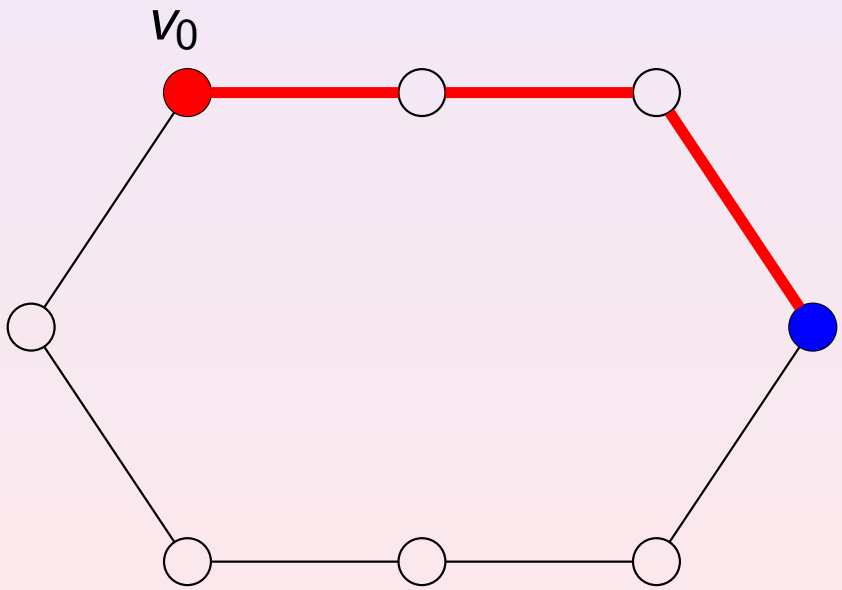
$\mathbf{s(path, v_0)=1}$



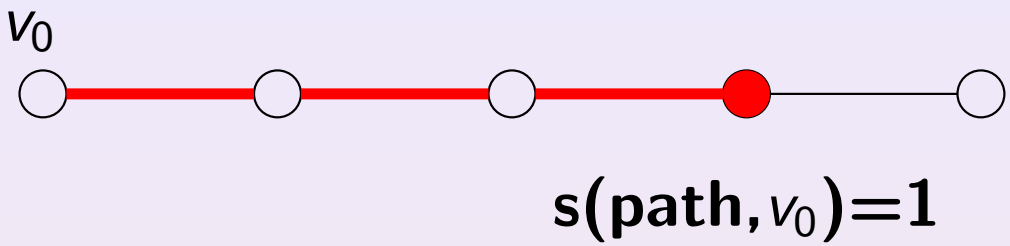
Two simple examples : the path and the ring



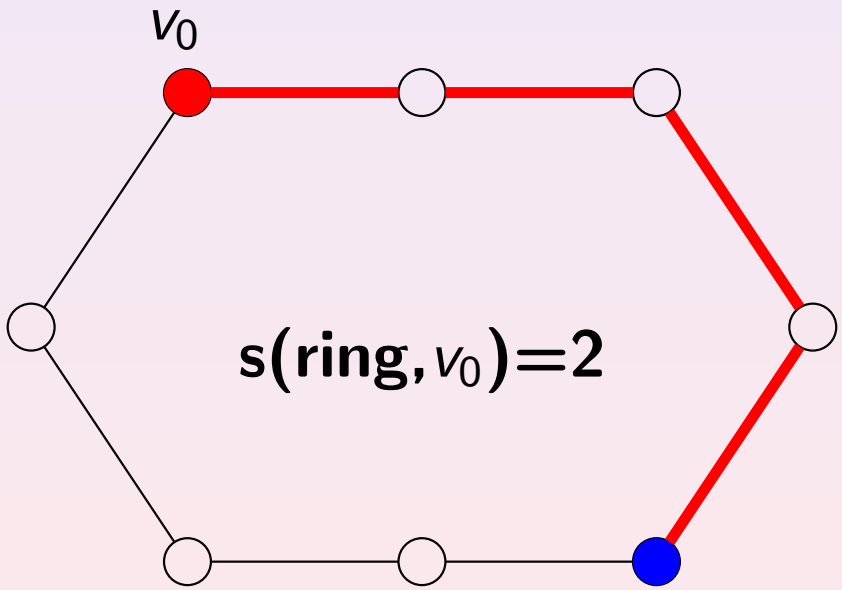
$\mathbf{s(path, v_0)=1}$



Two simple examples : the path and the ring



s(path,v_0)=1



s(ring,v_0)=2

Monotone connected search strategy

Monotone connected strategy

- Monotonicity : the contaminated part of the graph never grows (i.e., no recontamination can occur)
  $\Rightarrow$ **polynomial time**
- Connectivity : the cleared part is connected
  $\Rightarrow$ **safe communications**.

Remark : The problem of computing $\mathbf{s}(G, v_0)$ and the corresponding monotone connected strategy is NP-complete in a centralized setting [Megiddo *at al.* 88]

Monotone connected search strategy

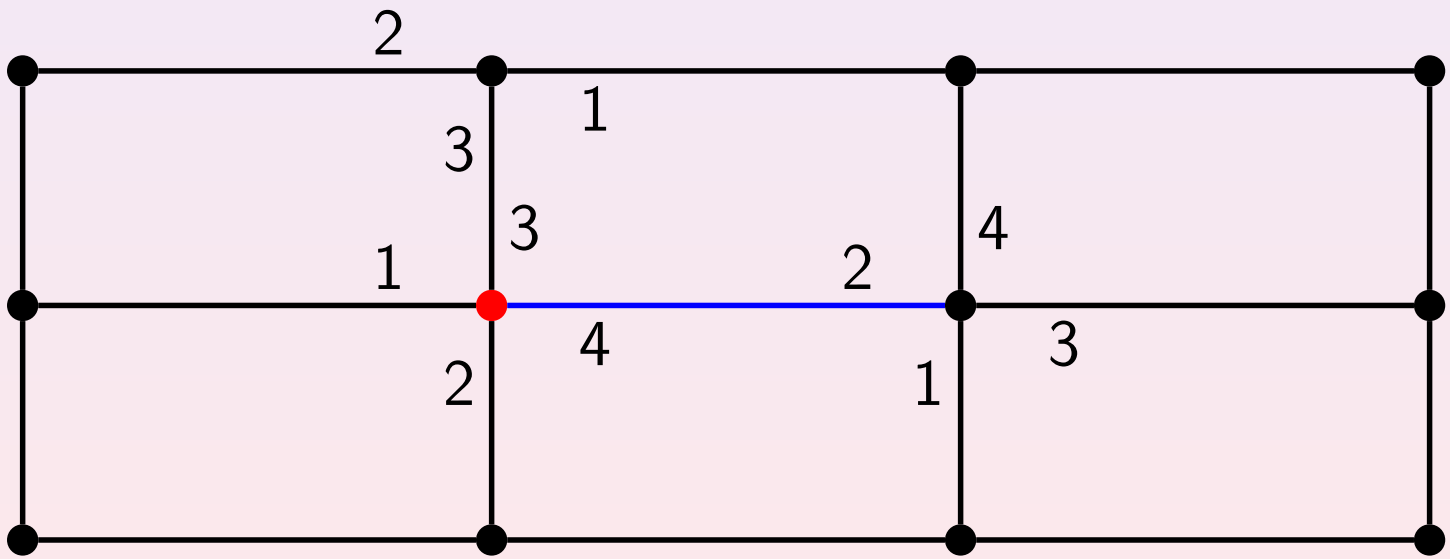
Monotone connected strategy

- Monotonicity : the contaminated part of the graph never grows (i.e., no recontamination can occur)
  $\Rightarrow$ **polynomial time**
- Connectivity : the cleared part is connected
  $\Rightarrow$ **safe communications**.

Remark : The problem of computing $\mathbf{s}(G, v_0)$ and the corresponding monotone connected strategy is NP-complete in a centralized setting [Megiddo *at al.* 88]

Model : Environment

- undirected connected graph ;
- local orientation of the edges ;
- synchronous/asynchronous environment.



Model : the searchers

- autonomous mobile computing entities with distinct IDs ;
- automata with $O(\log n)$ bits of memory.

Decision is computed locally and depends on :

- its current state ;
- the states of the other searchers present at the vertex ;
- if appropriate the incoming port number.

A searcher can decide to :

- leave a vertex via a specific port number ;
- switch its state.

Related work 1/2

The searchers have a prior knowledge of the topology.

Protocols to clear specific topologies

- **Tree** [Barrière, Flocchini, Fraigniaud and Santoro. 2002]
- **Mesh** [Flocchini, Luccio and Song. 2005]
- **Hypercube** [Flocchini, Huang and Luccio. 2005]
- **Tori** [Flocchini, Luccio and Song. 2006]
- **Siperski's graph** [Luccio. 2007]

A monotone connected and optimal strategy is performed.

Remark : Compared with the synchronous case, an additional searcher may be necessary and is sufficient in an asynchronous network to clear a graph in a monotone connected way [FLS05].

Related work 1/2

The searchers have a prior knowledge of the topology.

Protocols to clear specific topologies

- **Tree** [Barrière, Flocchini, Fraigniaud and Santoro. 2002]
- **Mesh** [Flocchini, Luccio and Song. 2005]
- **Hypercube** [Flocchini, Huang and Luccio. 2005]
- **Tori** [Flocchini, Luccio and Song. 2006]
- **Siperski's graph** [Luccio. 2007]

A monotone connected and optimal strategy is performed.

Remark : Compared with the synchronous case, an additional searcher may be necessary and is sufficient in an asynchronous network to clear a graph in a monotone connected way [FLS05].

Related work 1/2

The searchers have a prior knowledge of the topology.

Protocols to clear specific topologies

- **Tree** [Barrière, Flocchini, Fraigniaud and Santoro. 2002]
- **Mesh** [Flocchini, Luccio and Song. 2005]
- **Hypercube** [Flocchini, Huang and Luccio. 2005]
- **Tori** [Flocchini, Luccio and Song. 2006]
- **Siperski's graph** [Luccio. 2007]

A monotone connected and optimal strategy is performed.
Remark : Compared with the synchronous case, an additional searcher may be necessary and is sufficient in an asynchronous network to clear a graph in a monotone connected way [FLS05].

Related work 2/2

The searchers have no prior information about the graph.

Protocol to clear an unknown graph

Distributed chasing of network intruders
[Blin, Fraigniaud, Nisse and Vial. SIROCCO 2006]

A connected and optimal strategy is performed.

Problem : the strategy is not monotone and may be performed in expentional time.

Related work 2/2

The searchers have no prior information about the graph.

Protocol to clear an unknown graph

Distributed chasing of network intruders
[Blin, Fraigniaud, Nisse and Vial. SIROCCO 2006]

A connected and optimal strategy is performed.

Problem : the strategy is not monotone and may be performed in expentional time.

Problem

A natural question is :

What is the information that must be given to the searchers such that it exists a distributed protocol that enables them to clear all graphs in a monotone connected and optimal way ?

What kind of knowledge ?

Qualitative information

- Topology, size, diameter of the network ...

Quantitative information : advice [Fraigniaud *et al.* PODC06]

- Measure the **minimum number of bits of information** to **efficiently** perform a distributed task.

Problem

A natural question is :

What is the information that must be given to the searchers such that it exists a distributed protocol that enables them to clear all graphs in a monotone connected and optimal way ?

What kind of knowledge ?

Qualitative information

- Topology, size, diameter of the network ...

Quantitative information : advice [Fraigniaud *et al.* PODC06]

- Measure the **minimum number of bits of information** to **efficiently** perform a distributed task.

Problem

A natural question is :

What is the information that must be given to the searchers such that it exists a distributed protocol that enables them to clear all graphs in a monotone connected and optimal way ?

What kind of knowledge ?

Qualitative information

- Topology, size, diameter of the network ...

Quantitative information : advice [Fraigniaud *et al.* PODC06]

- Measure the **minimum number of bits of information** to **efficiently** perform a distributed task.

Advice, size of advice [Fraigniaud *et al.* 06]

A distributed problem $\mathcal{P}$
Instance of $\mathcal{P}$ (for example a graph G)
Advice : information that can be used to solve $\mathcal{P}$ *efficiently*

Information is modelized by

- An oracle $\mathcal{O}$ that assigns at any instance G a string of bits $\mathcal{O}(G)$ that is distributed on the vertices of G.
- size of advice $|\mathcal{O}(G)|$

Examples

- wake-up (linear number of messages) : $\Theta(n \log n)$ bits ;
- broadcast (linear number of messages) : $O(n)$ bits ;
- tree exploration, MST, graph coloring ...

Advice, size of advice [Fraigniaud *et al.* 06]

A distributed problem $\mathcal{P}$
Instance of $\mathcal{P}$ (for example a graph G)
Advice : information that can be used to solve $\mathcal{P}$ *efficiently*

Information is modelized by

- An oracle $\mathcal{O}$ that assigns at any instance G a string of bits $\mathcal{O}(G)$ that is distributed on the vertices of G.
- size of advice $|\mathcal{O}(G)|$

Examples

- wake-up (linear number of messages) : $\Theta(n \log n)$ bits ;
- broadcast (linear number of messages) : $O(n)$ bits ;
- tree exploration, MST, graph coloring ...

Advice, size of advice [Fraigniaud *et al.* 06]

Problem : distributed search problem
Instance : a graph G and a homebase $v_0 \in V(G)$.
Advice : to solve the problem in polynomial time.

Information is modelized by

- An oracle $\mathcal{O}$ that distributes a string of bits $\mathcal{O}(G, v_0)$ on the vertices of G.
- size of advice $|\mathcal{O}(G, v_0)|$

Question :
What is the minimum size of advice such that it exists a distributed protocol that efficiently solves the distributed search problem ?

Our results

Upper bound

$O(n \log n)$ bits of advice are sufficient to solve the distributed search problem.

We design an oracle $\mathcal{O}$ of size $O(n \log n)$ bits and a distributed protocol $\mathcal{P}$ using $\mathcal{O}$ that clears all graphs in a monotone connected and optimal way.

Lower bound

Any protocol using $o(n \log n)$ bits of advice cannot solve the distributed search problem.

We provide a class $\mathcal{C}$ of graphs such that any distributed protocol $\mathcal{P}$ using an advice of size less than $\Omega(n \log n)$ bits cannot clear all the graphs of $\mathcal{C}$.

Our results

Upper bound

$O(n \log n)$ bits of advice are sufficient to solve the distributed search problem.

We design an oracle $\mathcal{O}$ of size $O(n \log n)$ bits and a distributed protocol $\mathcal{P}$ using $\mathcal{O}$ that clears all graphs in a monotone connected and optimal way.

Lower bound

Any protocol using $o(n \log n)$ bits of advice cannot solve the distributed search problem.

We provide a class $\mathcal{C}$ of graphs such that any distributed protocol $\mathcal{P}$ using an advice of size less than $\Omega(n \log n)$ bits cannot clear all the graphs of $\mathcal{C}$.

Our results

Upper bound

$O(n \log n)$ bits of advice are sufficient to solve the distributed search problem.

We design an oracle $\mathcal{O}$ of size $O(n \log n)$ bits and a distributed protocol $\mathcal{P}$ using $\mathcal{O}$ that clears all graphs in a monotone connected and optimal way.

Lower bound

Any protocol using $o(n \log n)$ bits of advice cannot solve the distributed search problem.

We provide a class $\mathcal{C}$ of graphs such that any distributed protocol $\mathcal{P}$ using an advice of size less than $\Omega(n \log n)$ bits cannot clear all the graphs of $\mathcal{C}$.

Our results

Upper bound

$O(n \log n)$ bits of advice are sufficient to solve the distributed search problem.

We design an oracle $\mathcal{O}$ of size $O(n \log n)$ bits and a distributed protocol $\mathcal{P}$ using $\mathcal{O}$ that clears all graphs in a monotone connected and optimal way.

Lower bound

Any protocol using $o(n \log n)$ bits of advice cannot solve the distributed search problem.

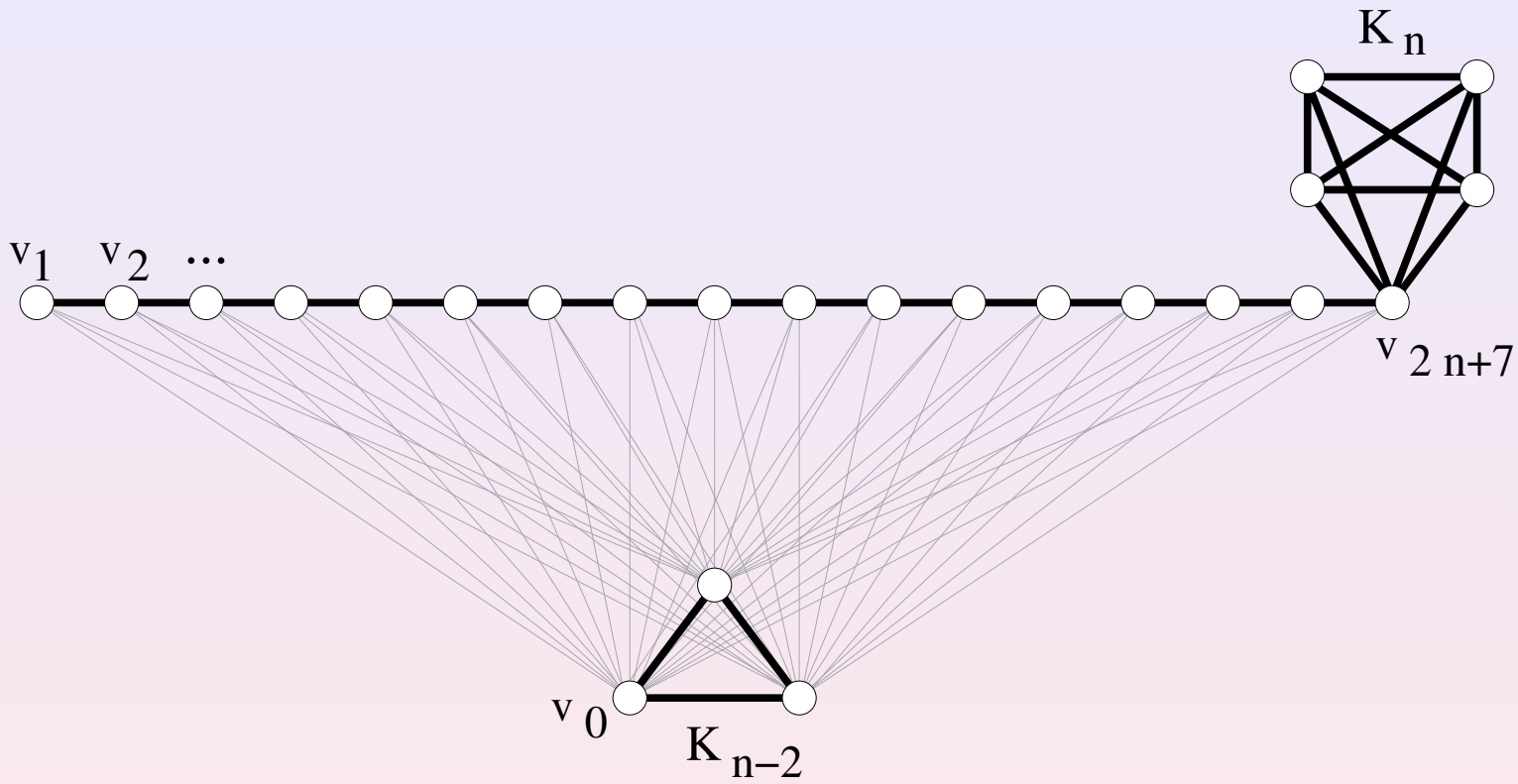
We provide a class $\mathcal{C}$ of graphs such that any distributed protocol $\mathcal{P}$ using an advice of size less than $\Omega(n \log n)$ bits cannot clear all the graphs of $\mathcal{C}$.

Idea of the upper bound : $O(n \log n)$

Let G be a graph, and $v_0 \in V(G)$

Let S be a monotone connected and optimal strategy for G.
Our oracle "encodes" S on the vertices of G.

The lower bound : $\Omega(n \log n)$



Class of graphs $(G_n)_{n \in N}$ (The figure corresponds to G_5).
All the monotone connected and optimal strategies in this class are **strongly constrained**.

Perspectives

What is the minimum quantity of advice that must be distributed **on each vertex** to solve the distributed search problem ?

How can we **relax the contraints** on the strategy to decrease the quantity of advice ?
Namely, what is the minimum quantity of advice to clear a graph **in polynomial time** (without imposing monotonicity) ?