

Satisfaire un internaute impatient est difficile¹

Fedor V. Fomin¹ Frédéric Giroire² Alain Jean-Marie³
Dorian Mazauric^{2,4} **Nicolas Nisse²**

¹ University of Bergen, Norway

² MASCOTTE, INRIA, I3S (CNRS, UNS) Sophia Antipolis, France

³ LIRMM, MAESTRO, INRIA, Montpellier, France

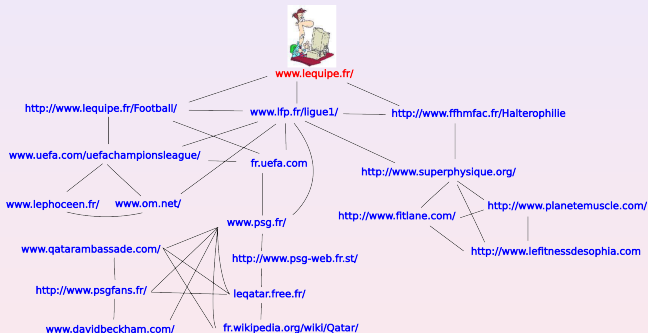
⁴ Columbia University, New York, USA

AlgoTel, la Grande Motte, 31st May, 2012

¹ to be presented next week at FUN'12 by Fred

Prefetching for faster data access

Web: pre-loading web pages before the web Surfer accesses it
Goal: Avoid the web Surfer to wait

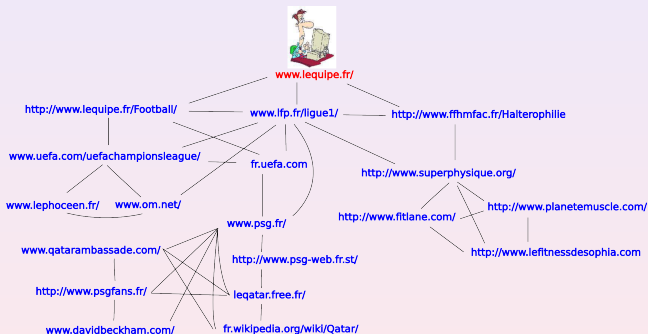


CACHE

www.lequipe.fr/

Prefetching for faster data access

the web Surfer starts from a given web page in cache
try to load web pages in cache → TAKES TIME



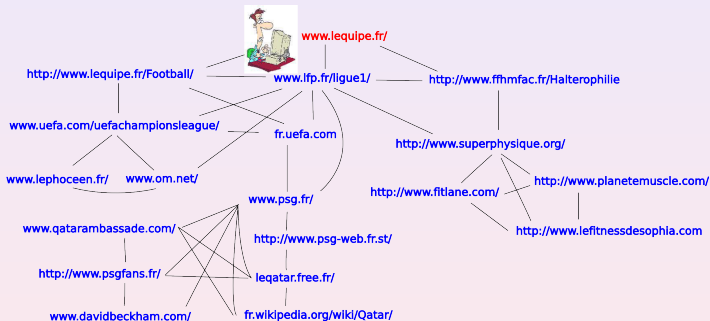
CACHE

www.lequipe.fr/ www.lfp.fr/ligue1/ <http://www.lequ...>

Prefetching for faster data access

at some point, web Surfer moves
if web page reached already in the cache

OK

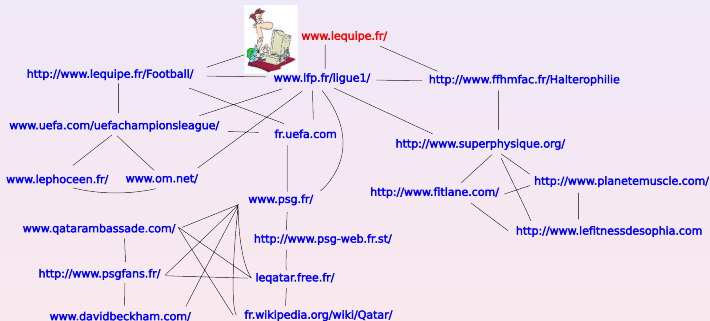


CACHE

www.lequipe.fr/ www.lfp.fr/ligue1/ <http://www.lequipe.fr/Football/> <http://www.ffhmfac.fr/Halterophilie>
www.uefa.com/uefacham...

Prefetching for faster data access

web Surfer follows the hyperlinks in an unpredictable way
web pages to be loaded may be “guessed”

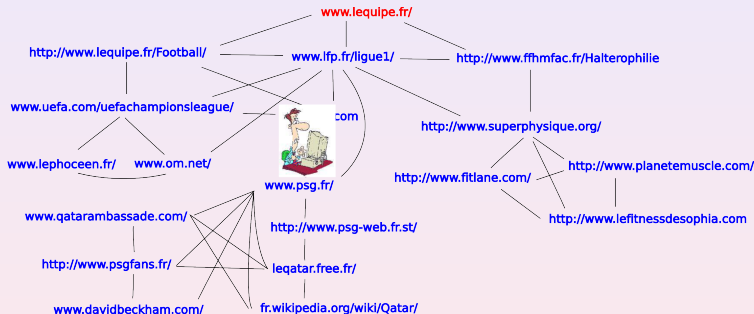


CACHE

www.lequipe.fr/ www.lfp.fr/ligue1/ <http://www.lequipe.fr/Football/> <http://www.ffhmfac.fr/Halterophilie>
www.uefa.com/uefachampionsleague/ fr.uefa.com www.om.net/ www.psg.fr/
<http://www.superphysique.org/>

Prefetching for faster data access

even if we guessed well
choices might be too numerous

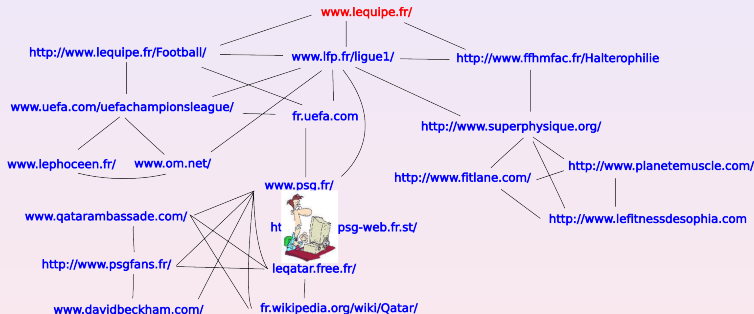


CACHE

www.lequipe.fr/ www.lfp.fr/ligue1/ <http://www.lequipe.fr/Football/> <http://www.ffhmfac.fr/Halterophilie>
www.uefa.com/uefachampionsleague/ fr.uefa.com www.om.net/ www.psg.fr/
<http://www.superp...>

Prefetching for faster data access

web Surfer may access a page not in cache
and has to wait !!



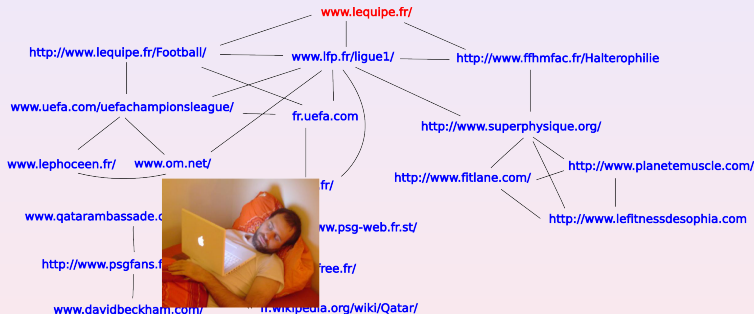
CACHE

www.lequipe.fr/ www.lfp.fr/ligue1/ <http://www.lequipe.fr/Football/> <http://www.ffhmfac.fr/Halterophilie>
www.uefa.com/uefachampionsleague/ fr.uefa.com www.om.net/ www.psq.fr/
<http://www.superphysique.org/>
www.qatarambassade.com/ <http://www.psgfans.fr/> <http://www.psg-web.fr.st/> [www.davidbec...](http://www.davidbeckham.com/)

Prefetching for faster data access

web Surfer may access a page not in cache

not good for research...



CACHE

www.lequipe.fr/ www.lfp.fr/ligue1/ <http://www.lequipe.fr/Football/> <http://www.ffhmfac.fr/Halterophilie>
www.uefa.com/uefachampionsleague/ fr.uefa.com www.om.net/ www.psg.fr/
<http://www.superp...>
www.qatarambassade.com/ <http://www.psgfans.fr/> <http://www.psg-web.fr/st/> www.davidbec...

Prefetching for faster data access

Issue: download speed, NOT size of cache

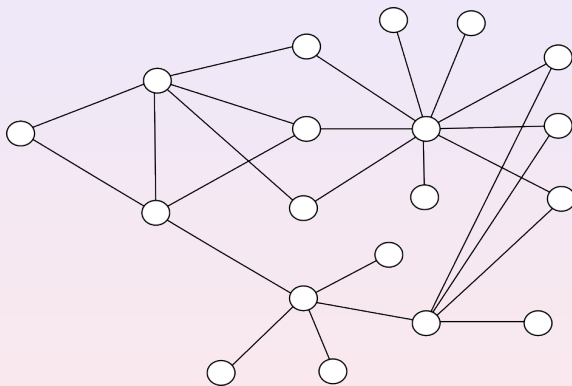
Instance: network KNOWN ((di)graph)

Related work: Probabilistic algorithms
(arcs + transition probabilities)

- Markovian model [Vitter,Krishnan. JACM'96]
[Morad,Jean-Marie. ROADEF'10]
- Stochastic Dynamic Programming framework
[Joseph,Grunwald. ISCA'97]
[Grigoras,Charvillat,Douze. ACM Multimedia'02]

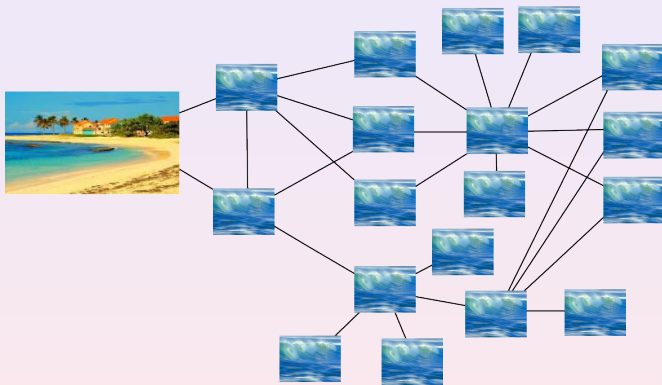
Our work: minimize download speed
to insure web Surfer never waits
(worst case, deterministic)

Model



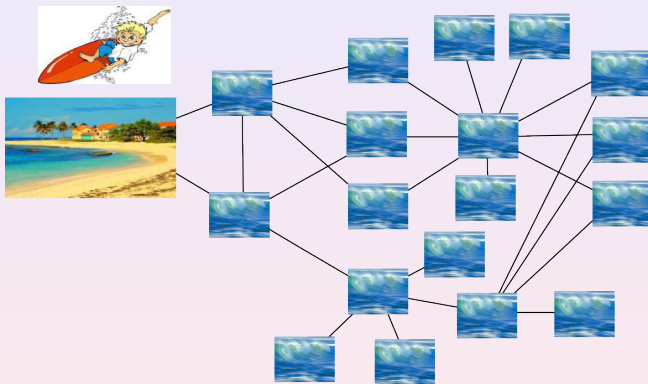
Web = connected (di)graph

Model



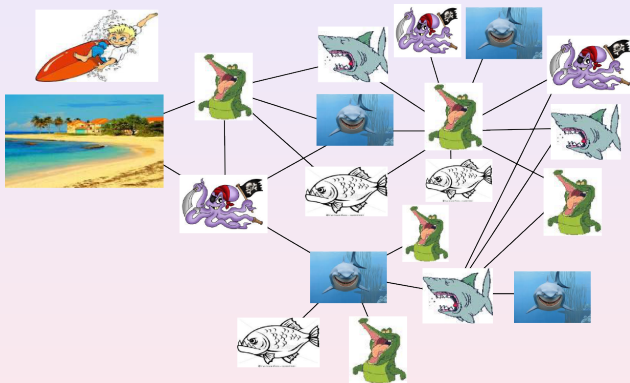
Web = (di)graph with a specified starting point (beach)

Model



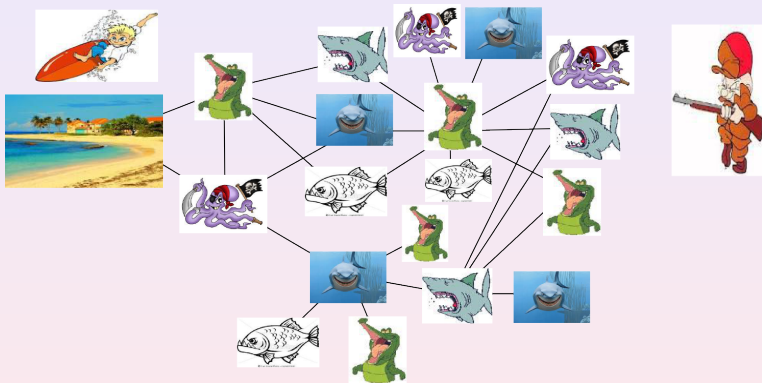
then, we need a (web) Surfer

Model



unloaded page = dangerous node

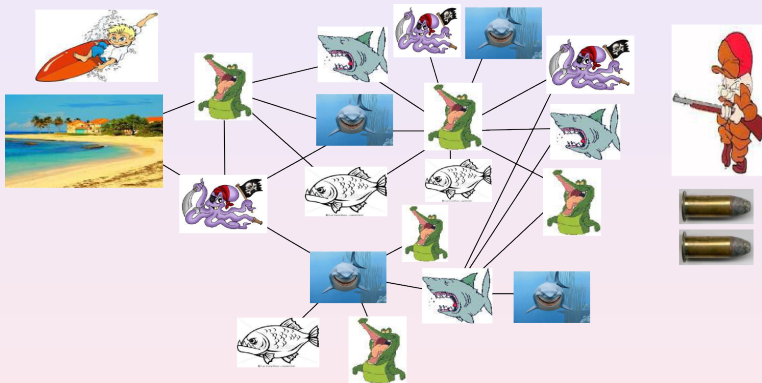
Model



we need someone to help the Surfer

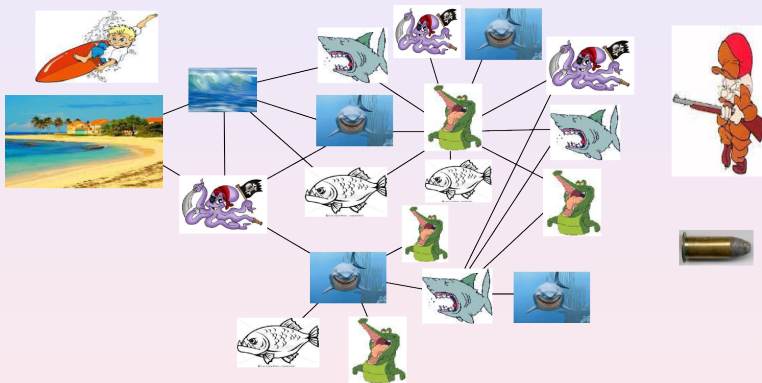
let's call it *the Guard*

Model



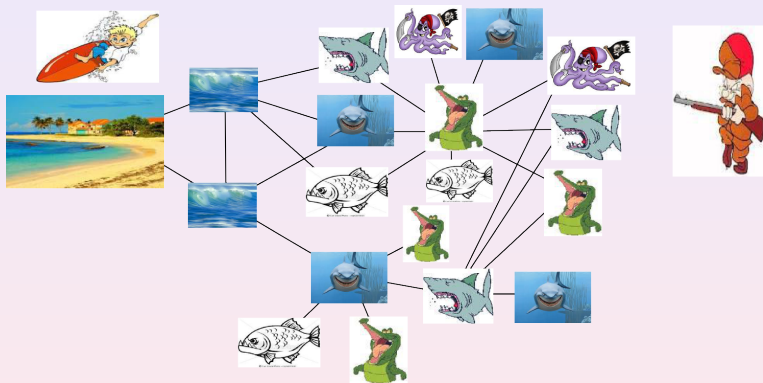
download speed = amount of bullets

Model



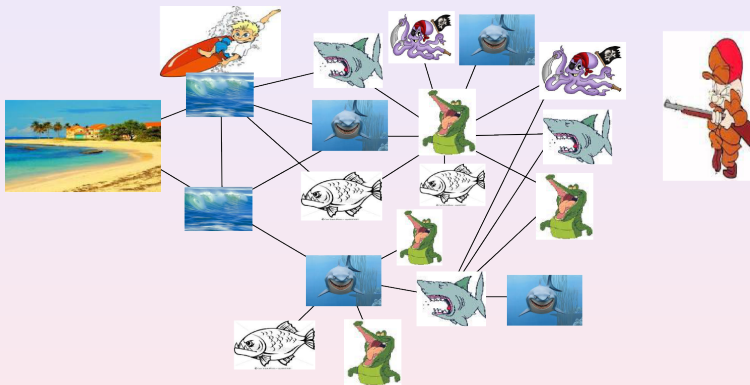
Guard uses one bullet to secure one node

Model



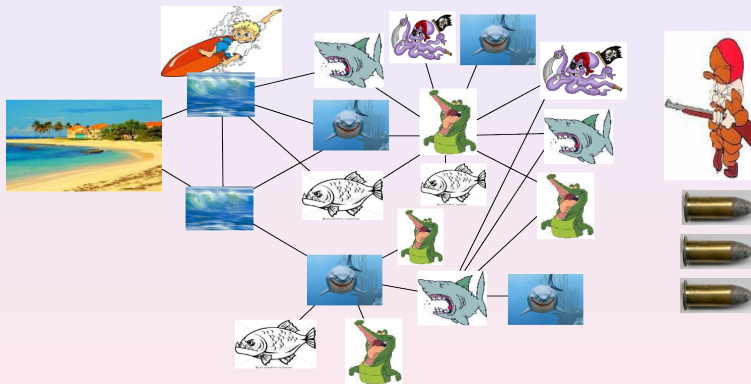
clearly, $\text{degree}(\text{beach}) \leq \#$ of bullets required to save Surfer

Model



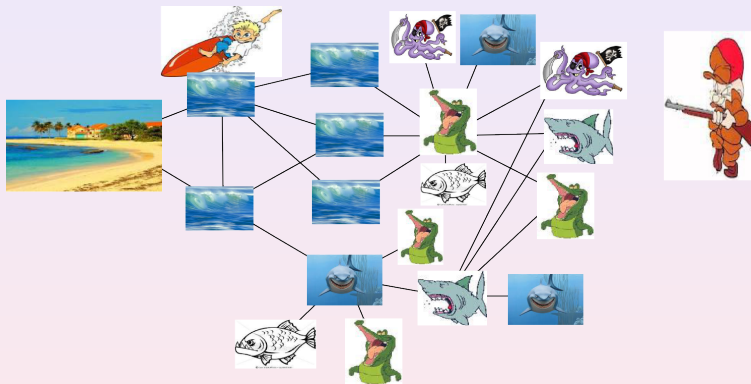
then, Surfer may move

Model



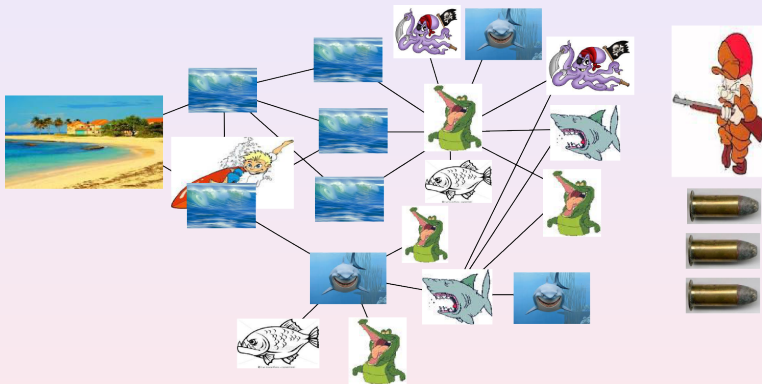
here one more bullet is needed

Model



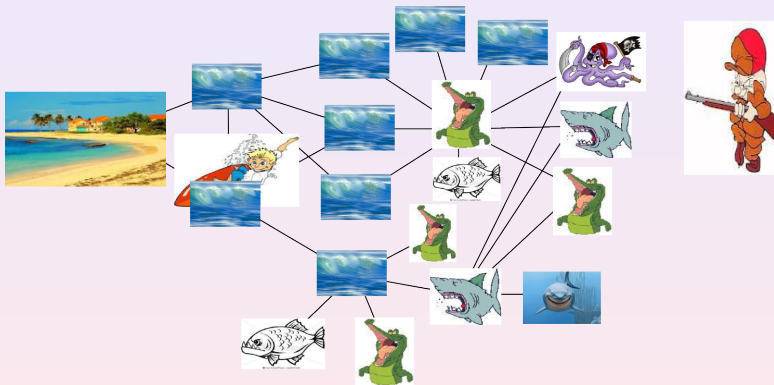
$$\text{degree}(\text{beach}) \leq \text{amount of bullets} \leq \text{max degree}$$

Model



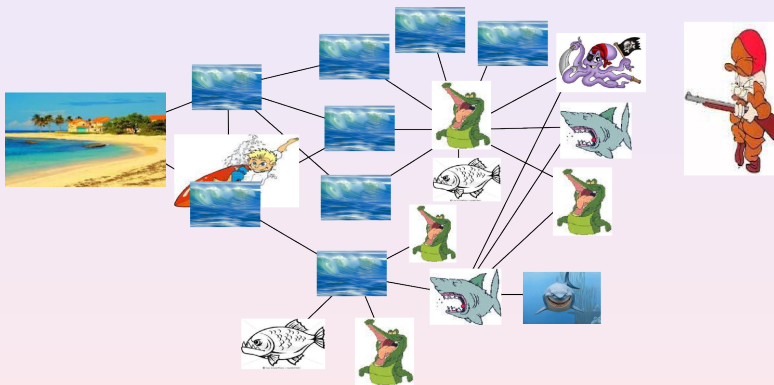
Surfer may move anywhere in its neighborhood

Model



bullets may be used to prevent future moves

Model



are 3 bullets sufficient to make sure Surfer never eaten?

Model: a Two players game

- a *Surfer*  starts from safe homebase v_0 
in G , a dangerous graph     
- a *Guard*  with some amount k of bullets 

Turn by turn:

- 1 the guard secures $\leq k$ nodes;
- 2 then, the Surfer may move to an adjacent node.

Defeat: Surfer in unsafe node

Victory: G safe

Minimize amount of bullets to win for any Surfer's trajectory

Surveillance number of G (connected) from v_0 : $sn(G, v_0)$

Model: a Two players game

- a *Surfer*  starts from safe homebase v_0 
in G , a dangerous graph     
- a *Guard*  with some amount k of bullets 

Turn by turn:

- 1 the guard *secures* $\leq k$ nodes;
- 2 then, the Surfer may *move to an adjacent node*.

Defeat: Surfer in unsafe node   **Victory:** G safe 

Minimize amount of bullets to win for any Surfer's trajectory
Surveillance number of G (connected) from v_0 : $sn(G, v_0)$

Model: a Two players game

- a *Surfer*  starts from safe homebase v_0 
in G , a dangerous graph     
- a *Guard*  with some amount k of bullets 

Turn by turn:

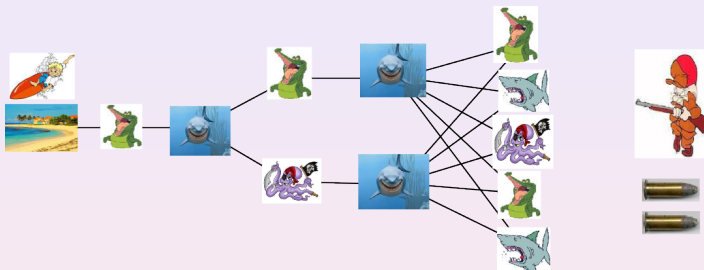
- 1 the guard *secures* $\leq k$ nodes;
- 2 then, the Surfer may *move to an adjacent node*.

Defeat: Surfer in unsafe node   **Victory:** G safe 

Minimize amount of bullets to win for any Surfer's trajectory

Surveillance number of G (**connected**) from v_0 : $sn(G, v_0)$

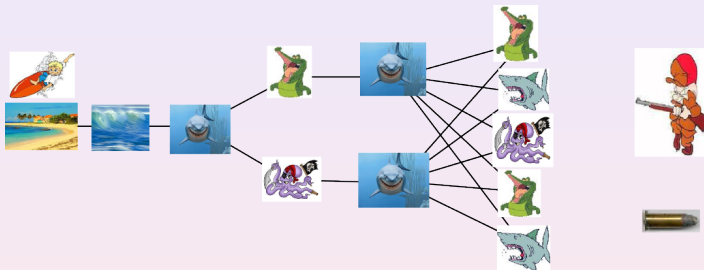
First Example



with 1 bullet: after 2 steps, Surfer faces 2 dangerous nodes!!

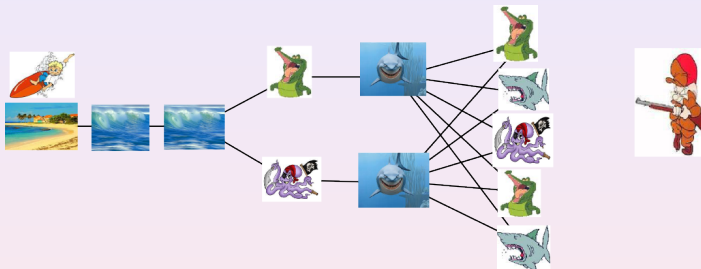
$$sn(G, v_0) > 1$$

First Example



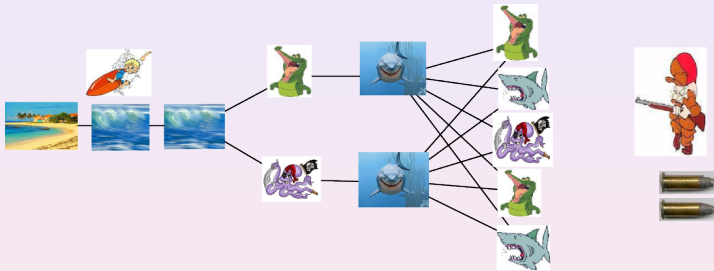
Guard uses his bullets

First Example



Guard uses (all) his bullets

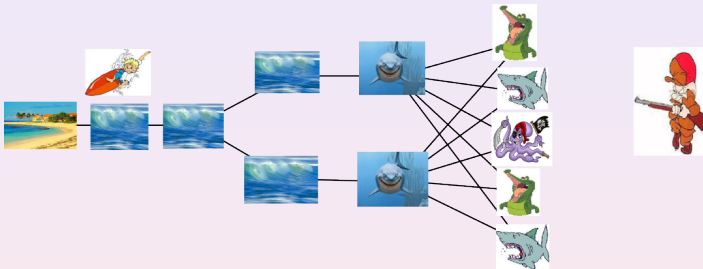
First Example



Guard uses (all) his bullets, **then** Surfer may move

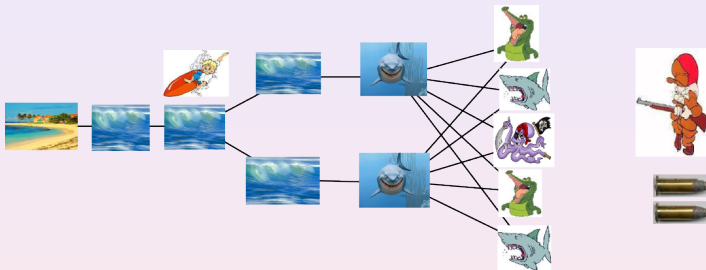
Clearly: worst case if Surfer always move

First Example



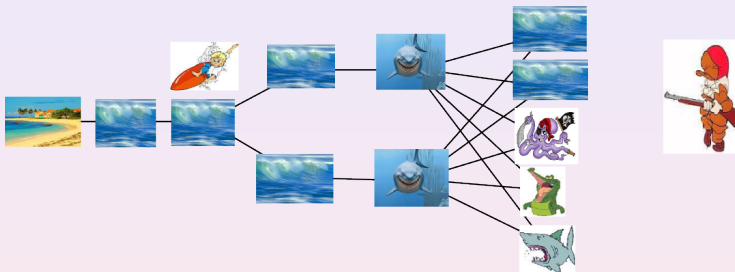
Guard uses (all) his bullets, **then** Surfer may move

First Example



Guard uses (all) his bullets, **then** Surfer moves

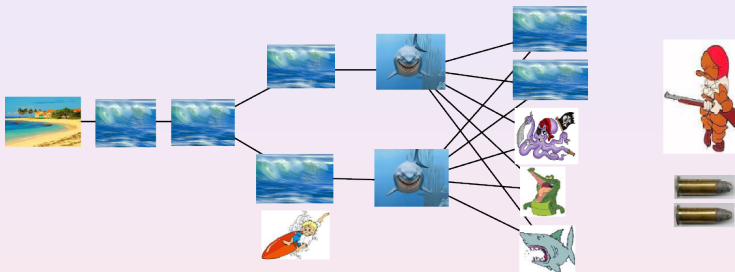
First Example



Guard uses (all) his bullets, **then** Surfer moves

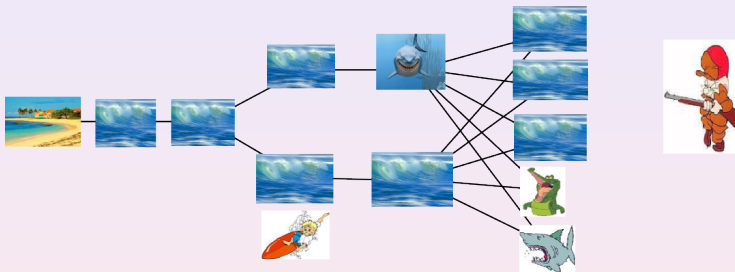
Guard may secure **any** node in the graph

First Example



Guard uses (all) his bullets, **then** Surfer moves

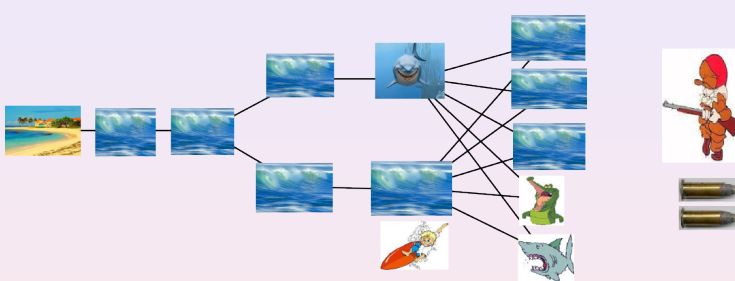
First Example



Guard uses (all) his bullets, **then** Surfer moves

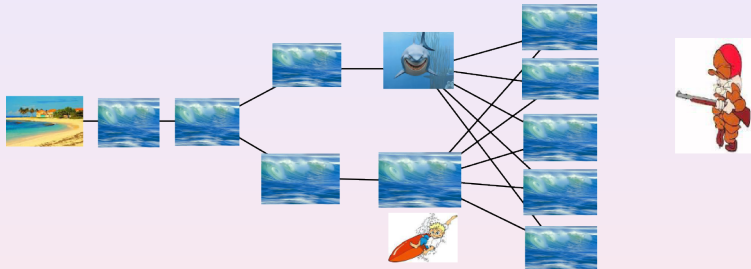
strategy: safe nodes + Surfer's node $\Rightarrow \leq k$ nodes to secure

First Example



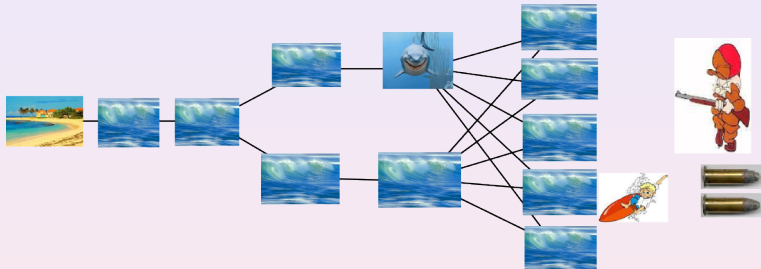
Guard uses (all) his bullets, **then** Surfer moves

First Example



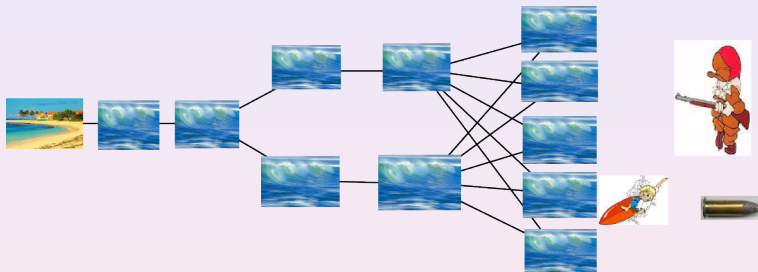
Guard uses (all) his bullets, **then** Surfer moves

First Example



Guard uses (all) his bullets, **then** Surfer moves

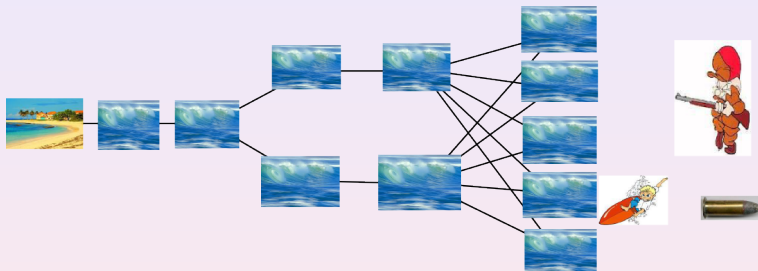
First Example



Guard uses (all) his bullets, **then** Surfer moves

All nodes safe: Victory **against this trajectory of the Surfer**

First Example

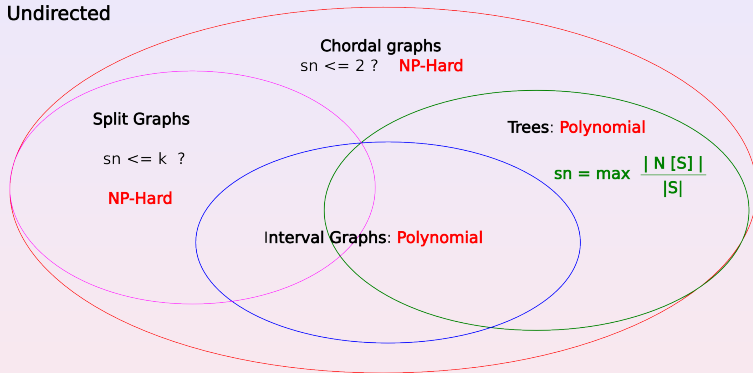


In this example, all Surfer's trajectory similar (by symmetry)

Victory whatever Surfer's trajectory $\Rightarrow sn(G, v_0) = 2$

Results: Complexity, Algorithms and Combinatoric

Undirected



Directed

DAGs: $sn \leq 4$? **P-SPACE-Complete**

DAGs: $sn \leq 2$? **NP-Hard**

General

$$(out)\text{-degree}(v_0) \leq sn \leq \max \begin{pmatrix} \text{degree}(v_0) \\ \text{max degree} - 1 \\ \text{max out-degree} \end{pmatrix}$$

$O(2^n)$ exact algorithm

Positive Results

Combinatorial characterization in Trees

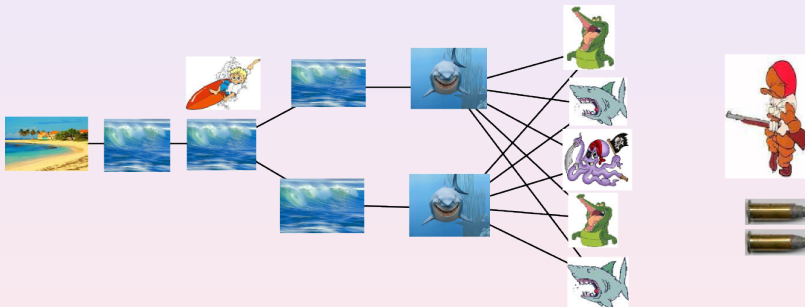
For any **tree** T , $v_0 \in V(T)$, $sn(T, v_0) = \max \lceil \frac{|N[S]|-1}{|S|} \rceil$, taken for any subtree S containing v_0 .

Exact Algorithms

- $O(2^n)$ algorithm in n -node graphs;
- $sn(T, v_0)$ can be computed in time $O(n \log n)$ in any n -node **tree** T and for any $v_0 \in V(T)$;
- $sn(G, v_0)$ can be computed in time $O(n \cdot \Delta^3)$ in any n -node **interval graph** G with maximum degree Δ and for any $v_0 \in V(T)$.

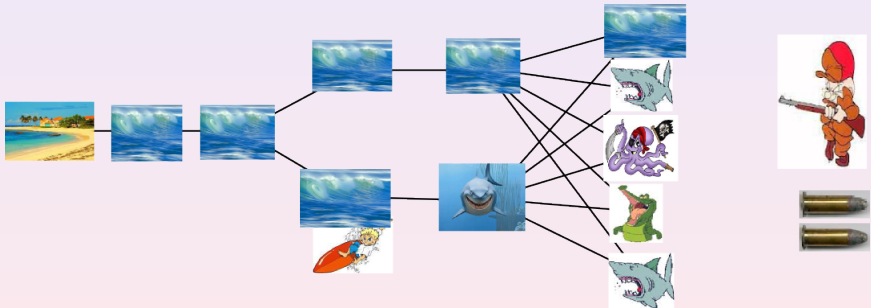
Further Work: Connected Variant

Constraint: safe vertices must induce a connected subgraph



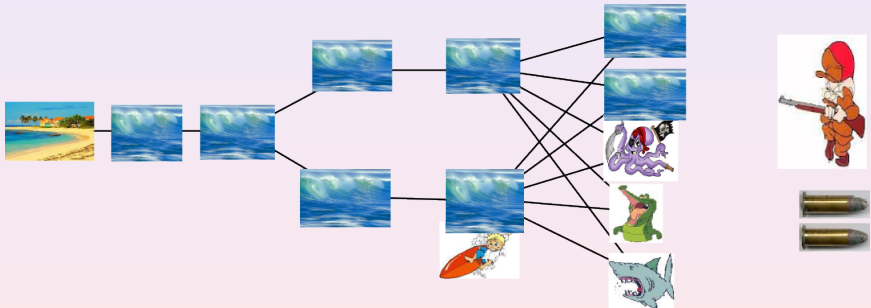
Further Work: Connected Variant

Constraint: safe vertices must induce a connected subgraph



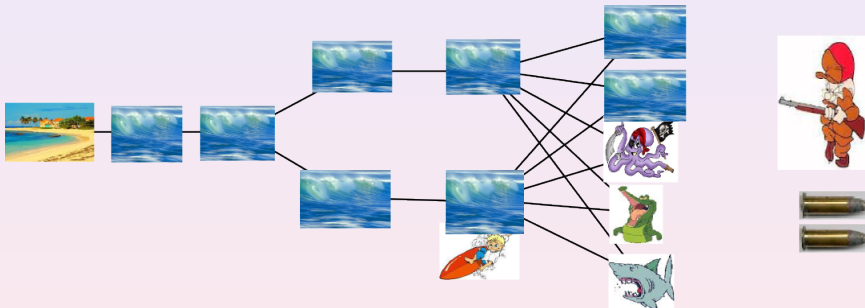
Further Work: Connected Variant

Constraint: safe vertices must induce a connected subgraph



Further Work: Connected Variant

Constraint: safe vertices must induce a connected subgraph



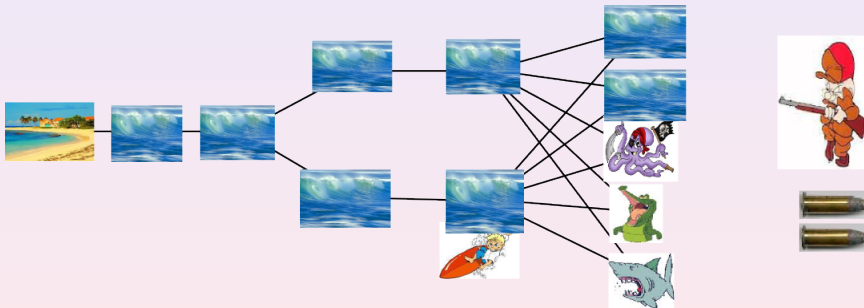
Connectivity costs:

$$\text{connected-}sn(G, v_0) = 3 > sn(G, v_0) = 2$$

All previous results hold for the connected variant

Further Work: Connected Variant

Constraint: safe vertices must induce a connected subgraph



Connectivity costs: $\text{connected-}sn(G, v_0) = 3 = sn(G, v_0) + 1$
 $\exists? G$ and v_0 such that $c\text{-}sn(G, v_0) \geq sn(G, v_0) + 2$????

Open Questions

- complexity in bounded degree graphs?
(polynomial if $\Delta \leq 3$)
- complexity in bounded treewidth graphs?
- $\exists? c < 2$ and $O(c^n)$ algorithm in n -node graphs?
- cost of connectivity? $\frac{\text{connected } sn}{sn} \leq cte?$
 $\exists? G$ and v_0 such that $c \cdot sn(G, v_0) \geq sn(G, v_0) + 2$????
- online variant?
- ...

Thank you for your attention²

²No seafood... no animal has been hurt when preparing this talk.