Capture Time in Variants of Cops & Robbers Games

Natasha Komarov

Dartmouth College

Thesis Defense
July 30, 2013

Table of Contents

Table of Contents

Set-up

Set-up

- Games are played on G: a connected, undirected, simple graph on n vertices.

Set-up

- Games are played on G: a connected, undirected, simple graph on n vertices.
- Two players:

Set-up

- Games are played on G: a connected, undirected, simple graph on n vertices.
- Two players:
  - pursuer (a.k.a. cop/hunter)

Set-up

- Games are played on G: a connected, undirected, simple graph on n vertices.
- Two players:
  - pursuer (a.k.a. cop/hunter)
  - evader (a.k.a. robber/drunk/mole/sitter/gambler)

Set-up

- Games are played on G: a connected, undirected, simple graph on n vertices.
- Two players:
  - pursuer (a.k.a. cop/hunter)
  - evader (a.k.a. robber/drunk/mole/sitter/gambler)

  move vertex to vertex on G.

Set-up

- Games are played on G: a connected, undirected, simple graph on n vertices.
- Two players:
  - pursuer (a.k.a. cop/hunter)
  - evader (a.k.a. robber/drunk/mole/sitter/gambler)

  move vertex to vertex on G.
- **Capture** occurs when cop and robber occupy same vertex at same time.

Set-up

- Games are played on G: a connected, undirected, simple graph on n vertices.
- Two players:
  - pursuer (a.k.a. cop/hunter)
  - evader (a.k.a. robber/drunk/mole/sitter/gambler)

  move vertex to vertex on G.
- **Capture** occurs when cop and robber occupy same vertex at same time.
- The cop's goal is to capture the robber in the minimal possible number of steps.

Set-up

- Games are played on G: a connected, undirected, simple graph on n vertices.
- Two players:
  - pursuer (a.k.a. cop/hunter)
  - evader (a.k.a. robber/drunk/mole/sitter/gambler)

  move vertex to vertex on G.
- **Capture** occurs when cop and robber occupy same vertex at same time.
- The cop's goal is to capture the robber in the minimal possible number of steps.
- The robber's goal is to evade capture as long as possible.

Set-up

- Games are played on G: a connected, undirected, simple graph on n vertices.
- Two players:
  - pursuer (a.k.a. cop/hunter)
  - evader (a.k.a. robber/drunk/mole/sitter/gambler)

  move vertex to vertex on G.
- **Capture** occurs when cop and robber occupy same vertex at same time.
- The cop's goal is to capture the robber in the minimal possible number of steps.
- The robber's goal is to evade capture as long as possible.
- A **move** consists of a step by the cop followed by a step by the robber

Set-up

- Games are played on G: a connected, undirected, simple graph on n vertices.
- Two players:
  - pursuer (a.k.a. cop/hunter)
  - evader (a.k.a. robber/drunk/mole/sitter/gambler)

  move vertex to vertex on G.
- **Capture** occurs when cop and robber occupy same vertex at same time.
- The cop's goal is to capture the robber in the minimal possible number of steps.
- The robber's goal is to evade capture as long as possible.
- A **move** consists of a step by the cop followed by a step by the robber (like chess).

Original game

Original game

- Introduced by Nowakowski & Winkler [5] and (independently) Quilliot [7].

Original game

- Introduced by Nowakowski & Winkler [5] and (independently) Quilliot [7].
- Cop and robber move alternately from vertex to adjacent vertex, with full information about each other's positions.

Original game

- Introduced by Nowakowski & Winkler [5] and (independently) Quilliot [7].
- Cop and robber move alternately from vertex to adjacent vertex, with full information about each other's positions.
- Graphs on which a cop can win (i.e. capture) in finite time are called **cop-win**.

Original game

- Introduced by Nowakowski & Winkler [5] and (independently) Quilliot [7].
- Cop and robber move alternately from vertex to adjacent vertex, with full information about each other's positions.
- Graphs on which a cop can win (i.e. capture) in finite time are called **cop-win**.
- Game takes no more than $n-4$ moves on cop-win graphs with $n \geq 7$ [1, 3]. (Note: original game can't take more than n^2 on any graph, including directed graphs.)

Table of Contents

Game set-up

Game set-up

- Suggested by Ross Churchley [4].

Game set-up

- Suggested by Ross Churchley [4].
- Evader is now a drunk: random walker.

Game set-up

- Suggested by Ross Churchley [4].
- Evader is now a drunk: random walker.
- Cop and drunk still move alternately with full information.

Game set-up

- Suggested by Ross Churchley [4].
- Evader is now a drunk: random walker.
- Cop and drunk still move alternately with full information.
- Cop always wins (probability 1)

Game set-up

- Suggested by Ross Churchley [4].
- Evader is now a drunk: random walker.
- Cop and drunk still move alternately with full information.
- Cop always wins (probability 1); how long does it take (on average)?

Game set-up

- Suggested by Ross Churchley [4].
- Evader is now a drunk: random walker.
- Cop and drunk still move alternately with full information.
- Cop always wins (probability 1); how long does it take (on average)?

Theorem

On a connected, undirected, simple graph on n vertices, a cop can capture a drunk in expected time $n+o(n)$.

Is that obvious?

Is that obvious?

[Spoiler: it isn't.]

Is that obvious?

[Spoiler: it isn't.]

Perhaps most obvious cop strategy: greedy algorithm (i.e. minimize distance at each step)

Is that obvious?

[Spoiler: it isn't.]

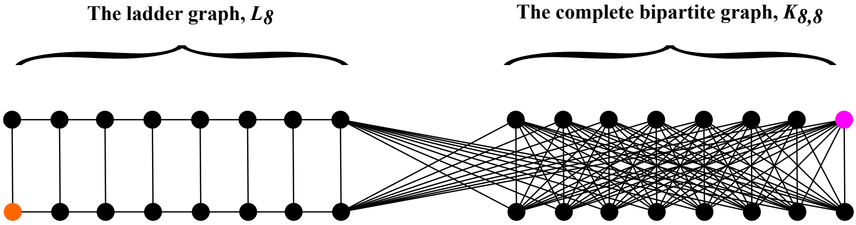
Perhaps most obvious cop strategy: greedy algorithm (i.e. minimize distance at each step)... fails!

Is that obvious?

[Spoiler: it isn't.]

Perhaps most obvious cop strategy: greedy algorithm (i.e. minimize distance at each step)... fails!

Example. "Ladder to the Basement"



Retargeting

Retargeting

- Cop's problem was **retargeting** too often: cop is made indecisive by an indecisive drunk.

Retargeting

- Cop's problem was **retargeting** too often: cop is made indecisive by an indecisive drunk.
- How about retargeting less often?

Retargeting

- Cop's problem was **retargeting** too often: cop is made indecisive by an indecisive drunk.
- How about retargeting less often?

Lemma

The probability that a random walk will "mess up" during 4 consecutive steps is at least $\frac{n^{-2/3}}{4}$.

Retargeting

Lemma

The probability that a random walk will "mess up" during 4 consecutive steps is at least $\frac{n^{-2/3}}{4}$.

- So if cop and drunk start d apart, takes $4(4n^{2/3})(d-3)$ moves on average to get down to distance 3.

Retargeting

Lemma

The probability that a random walk will "mess up" during 4 consecutive steps is at least $\frac{n^{-2/3}}{4}$.

- So if cop and drunk start d apart, takes $4(4n^{2/3})(d-3)$ moves on average to get down to distance 3.
- After that, greedy algorithm only takes Δ more steps on average ($\Delta =$ highest degree).

Four-stage Strategy

- d can be as big as $n-1$, and $4(4n^{2/3})((n-1)-3)$ is too big to get the $n+o(n)$ bound.

Four-stage Strategy

- d can be as big as $n-1$, and $4(4n^{2/3})((n-1)-3)$ is too big to get the $n+o(n)$ bound.
- Need to do something else to get closer quicker!

Four-stage Strategy

- d can be as big as $n-1$, and $4(4n^{2/3})((n-1)-3)$ is too big to get the $n+o(n)$ bound.
- Need to do something else to get closer quicker!
- Stage 1: retarget upon arrival.

Four-stage Strategy

- d can be as big as $n-1$, and $4(4n^{2/3})((n-1)-3)$ is too big to get the $n+o(n)$ bound.
- Need to do something else to get closer quicker!
- Stage 1: retarget upon arrival. What's the distance when that's done?

Four-stage Strategy

- d can be as big as $n-1$, and $4(4n^{2/3})((n-1)-3)$ is too big to get the $n+o(n)$ bound.
- Need to do something else to get closer quicker!
- Stage 1: retarget upon arrival. What's the distance when that's done?

Lemma

Expected distance after Stage 1 is $O^*(\sqrt{n})$.

Four-stage Strategy

Lemma

Expected distance after Stage 1 is $O^*(\sqrt{n})$.

- This is a corollary of the Varopoulos-Carne bound [6, 8]:

Theorem

Let $P = (p(x,y))_{x,y \in V(G)}$ *be the transition matrix associated with a simple random walk* $\{x_0, x_1, \ldots\}$ *on* G. *Then*

$$p^t(x,y) \le \sqrt{e}\sqrt{\frac{\deg(y)}{\deg(x)}} \exp\left(-\frac{(d(x,y)-1)^2}{2(t-1)}\right)$$

where $p^t(x,y) = \mathbb{P}(x_t = y | x_0 = x)$.

Four-stage Strategy

- Stage 2: repeat.

Four-stage Strategy

- Stage 2: repeat. Similar argument yields:

Lemma

Expected distance after Stage 2 is $O^*(\sqrt[4]{n})$.

Four-stage Strategy

- Stage 2: repeat. Similar argument yields:

Lemma

Expected distance after Stage 2 is $O^*(\sqrt[4]{n})$.

- Stage 3: retarget every 4 steps until distance is at most 4, then

Four-stage Strategy

- Stage 2: repeat. Similar argument yields:

Lemma

Expected distance after Stage 2 is $O^*(\sqrt[4]{n})$.

- Stage 3: retarget every 4 steps until distance is at most 4, then
- Stage 4: greedy algorithm until caught.

Four-stage Strategy

- Stage 2: repeat. Similar argument yields:

Lemma

Expected distance after Stage 2 is $O^*(\sqrt[4]{n})$.

- Stage 3: retarget every 4 steps until distance is at most 4, then
- Stage 4: greedy algorithm until caught.
- How long do the four stages take?

Four-stage Strategy

- Stage 2: repeat. Similar argument yields:

Lemma

Expected distance after Stage 2 is $O^*(\sqrt[4]{n})$.

- Stage 3: retarget every 4 steps until distance is at most 4, then
- Stage 4: greedy algorithm until caught.
- How long do the four stages take?
  - Stage 1: $\leq \text{diam}(G)$

Four-stage Strategy

- Stage 2: repeat. Similar argument yields:

Lemma

Expected distance after Stage 2 is $O^*(\sqrt[4]{n})$.

- Stage 3: retarget every 4 steps until distance is at most 4, then
- Stage 4: greedy algorithm until caught.
- How long do the four stages take?
  - Stage 1: $\leq \mathrm{diam}(G)$
  - Stage 2: $O^*(\sqrt{n})$

Four-stage Strategy

- Stage 2: repeat. Similar argument yields:

Lemma

Expected distance after Stage 2 is $O^*(\sqrt[4]{n})$.

- Stage 3: retarget every 4 steps until distance is at most 4, then
- Stage 4: greedy algorithm until caught.
- How long do the four stages take?
  - Stage 1: $\leq \mathrm{diam}(G)$
  - Stage 2: $O^*(\sqrt{n})$
  - Stage 3: $4(4n^{2/3})(O^*(\sqrt[4]{n}) - 3) = O^*(n^{11/12})$.

Four-stage Strategy

- Stage 2: repeat. Similar argument yields:

Lemma

Expected distance after Stage 2 is $O^*(\sqrt[4]{n})$.

- Stage 3: retarget every 4 steps until distance is at most 4, then
- Stage 4: greedy algorithm until caught.
- How long do the four stages take?
  - Stage 1: $\leq \mathrm{diam}(G)$
  - Stage 2: $O^*(\sqrt{n})$
  - Stage 3: $4(4n^{2/3})(O^*(\sqrt[4]{n}) - 3) = O^*(n^{11/12})$.
  - Stage 4: $\leq \Delta$

Four-stage Strategy

- Stage 2: repeat. Similar argument yields:

Lemma

Expected distance after Stage 2 is $O^*(\sqrt[4]{n})$.

- Stage 3: retarget every 4 steps until distance is at most 4, then
- Stage 4: greedy algorithm until caught.
- How long do the four stages take?
  - Stage 1: $\leq \mathrm{diam}(G)$
  - Stage 2: $O^*(\sqrt{n})$
  - Stage 3: $4(4n^{2/3})(O^*(\sqrt[4]{n}) - 3) = O^*(n^{11/12})$.
  - Stage 4: $\leq \Delta$
- Graph theory fun fact: $\mathrm{diam}(G) + \Delta \leq n + 1$.

Four-stage Strategy

- Stage 2: repeat. Similar argument yields:

Lemma

Expected distance after Stage 2 is $O^*(\sqrt[4]{n})$.

- Stage 3: retarget every 4 steps until distance is at most 4, then
- Stage 4: greedy algorithm until caught.
- How long do the four stages take?
  - Stage 1: $\leq \operatorname{diam}(G)$
  - Stage 2: $O^*(\sqrt{n})$
  - Stage 3: $4(4n^{2/3})(O^*(\sqrt[4]{n}) - 3) = O^*(n^{11/12})$.
  - Stage 4: $\leq \Delta$
- Graph theory fun fact: $\operatorname{diam}(G) + \Delta \leq n + 1$.
- So total time is at most $n + o(n)$.

Table of Contents

Background

- Sitter is immobile: picks a vertex and remains there until caught.

Background

- Sitter is immobile: picks a vertex and remains there until caught.
- Cop can't see the sitter.

Background

- Sitter is immobile: picks a vertex and remains there until caught.
- Cop can't see the sitter.
- The expected capture time is known to be $n-1$ on a tree, and is between $|E(G)|/2$ and $|E(G)|$ in general. [2]

Background

- Sitter is immobile: picks a vertex and remains there until caught.
- Cop can't see the sitter.
- The expected capture time is known to be $n-1$ on a tree, and is between $|E(G)|/2$ and $|E(G)|$ in general. [2]
- We look at a cop using depth first search (DFS) and find

Background

- Sitter is immobile: picks a vertex and remains there until caught.
- Cop can't see the sitter.
- The expected capture time is known to be $n-1$ on a tree, and is between $|E(G)|/2$ and $|E(G)|$ in general. [2]
- We look at a cop using depth first search (DFS) and find:
  - DFS is an optimal strategy for the cop on a tree.

Background

- Sitter is immobile: picks a vertex and remains there until caught.
- Cop can't see the sitter.
- The expected capture time is known to be $n-1$ on a tree, and is between $|E(G)|/2$ and $|E(G)|$ in general. [2]
- We look at a cop using depth first search (DFS) and find:
  - DFS is an optimal strategy for the cop on a tree.
  - Expected capture time is strictly less than $n-1$ on any non-tree using DFS.

Background

- Sitter is immobile: picks a vertex and remains there until caught.
- Cop can't see the sitter.
- The expected capture time is known to be $n-1$ on a tree, and is between $|E(G)|/2$ and $|E(G)|$ in general. [2]
- We look at a cop using depth first search (DFS) and find:
  - DFS is an optimal strategy for the cop on a tree.
  - Expected capture time is strictly less than $n-1$ on any non-tree using DFS.
  - Expected capture time is at least $\frac{n+1}{2}$ on any graph using DFS.

Table of Contents

Cop vs. Gambler

- Gambler: not constrained by edges, but must pick a **gamble** (probability distribution on the vertices of G) and stick to it.

Cop vs. Gambler

- Gambler: not constrained by edges, but must pick a **gamble** (probability distribution on the vertices of G) and stick to it.
- Cop: constrained by edges but knows gambler's gamble.

Cop vs. Gambler

- Gambler: not constrained by edges, but must pick a **gamble** (probability distribution on the vertices of G) and stick to it.
- Cop: constrained by edges but knows gambler's gamble.
- How long does it take cop to capture gambler (on average)?

Cop vs. Gambler

- Gambler: not constrained by edges, but must pick a **gamble** (probability distribution on the vertices of G) and stick to it.
- Cop: constrained by edges but knows gambler's gamble.
- How long does it take cop to capture gambler (on average)?

Theorem

The cop vs. gambler game takes expected time exactly n on any (connected, undirected, simple) graph.

Cop vs. Gambler

- Gambler: not constrained by edges, but must pick a **gamble** (probability distribution on the vertices of G) and stick to it.
- Cop: constrained by edges but knows gambler's gamble.
- How long does it take cop to capture gambler (on average)?

Theorem

The cop vs. gambler game takes expected time exactly n on any (connected, undirected, simple) graph.

Bonus: this remains true whether the cop gets to choose her initial position or not.

Table of Contents

Game set-up

- Hunter: not constrained by edges but plays in the dark.

Game set-up

- Hunter: not constrained by edges but plays in the dark.
- Mole: constrained by edges and must move, but can see hunter.

Game set-up

- Hunter: not constrained by edges but plays in the dark.
- Mole: constrained by edges and must move, but can see hunter.
- Players move simultaneously.

Game set-up

- Hunter: not constrained by edges but plays in the dark.
- Mole: constrained by edges and must move, but can see hunter.
- Players move simultaneously.
- On what graphs can hunter guarantee capture of mole in bounded time (call these **hunter-win**)?

Game set-up

- Hunter: not constrained by edges but plays in the dark.
- Mole: constrained by edges and must move, but can see hunter.
- Players move simultaneously.
- On what graphs can hunter guarantee capture of mole in bounded time (call these **hunter-win**)? (Equivalently: hunter plays against genius, prescient mole who always makes the moves guaranteed to maximize capture time.)

Game set-up

- Hunter: not constrained by edges but plays in the dark.
- Mole: constrained by edges and must move, but can see hunter.
- Players move simultaneously.
- On what graphs can hunter guarantee capture of mole in bounded time (call these **hunter-win**)? (Equivalently: hunter plays against genius, prescient mole who always makes the moves guaranteed to maximize capture time.)

Theorem

A graph is hunter-win if and only if it is a lobster.

Characterization

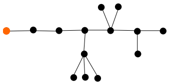
Definition

A **lobster** is a tree containing a path P such that all vertices are within distance 2 of P.

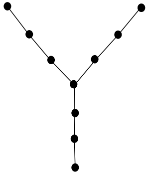
Characterization

Definition

A **lobster** is a tree containing a path P such that all vertices are within distance 2 of P.



A lobster.



Not a lobster.

Characterization

Lemma

Lobsters are hunter-win.

Characterization

Lemma

Lobsters are hunter-win.

Proof

Characterization

Lemma

Lobsters are hunter-win.

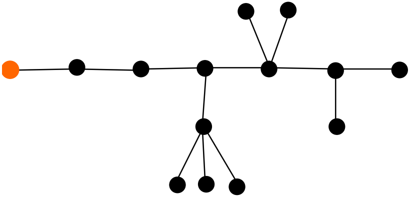
Proof by picture.

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture.

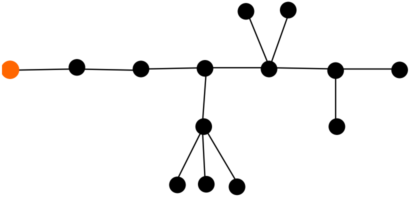


Characterization

Lemma

Lobsters are hunter-win.

Proof by picture.



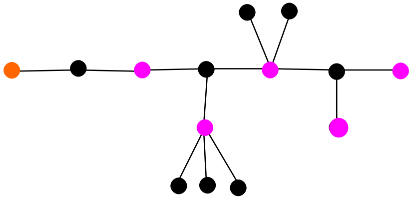
Define an **odd** (resp. **even**) **mole** to be a mole who starts at an odd (resp. even) distance from the marked vertex.

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture.



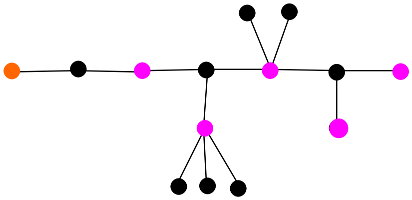
Define an **odd** (resp. **even**) **mole** to be a mole who starts at an odd (resp. even) distance from the marked vertex.

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 1^{st} step:



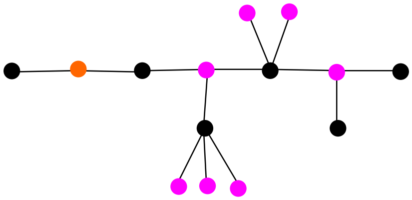
Orange vertex = hunter's position
Purple vertex = even mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 2^{nd} step:



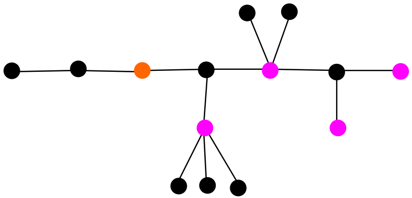
Orange vertex = hunter's position
Purple vertex = even mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 3^{rd} step:



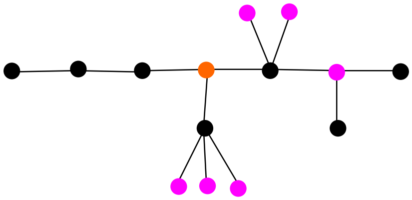
Orange vertex = hunter's position
Purple vertex = even mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 4^{th} step:



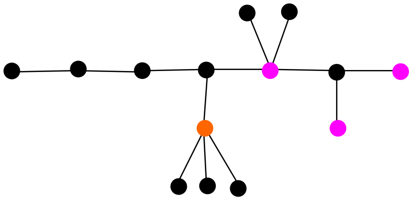
Orange vertex = hunter's position
Purple vertex = even mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 5^{th} step:



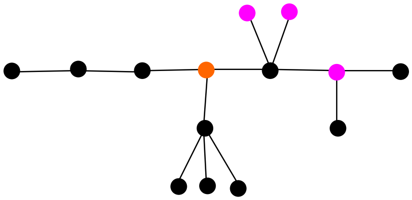
Orange vertex = hunter's position
Purple vertex = even mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 6^{th} step:



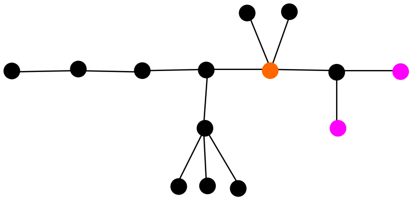
Orange vertex = hunter's position
Purple vertex = even mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 7^{th} step:



Orange vertex = hunter's position

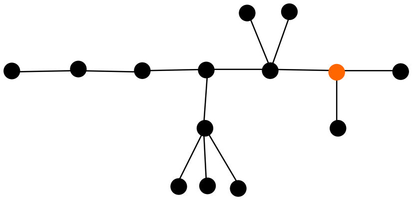
Purple vertex = even mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 8^{th} step:



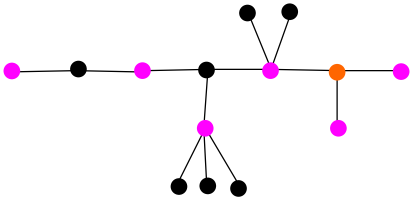
Orange vertex = hunter's position
Purple vertex = even mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 8^{th} step:



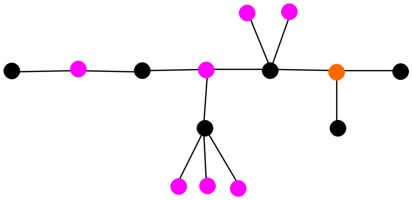
Orange vertex = hunter's position
Purple vertex = odd mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 9^{th} step:



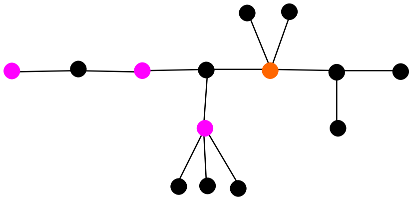
Orange vertex = hunter's position
Purple vertex = odd mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 10^{th} step:



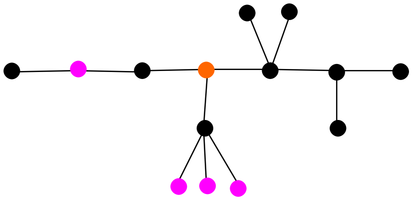
Orange vertex = hunter's position
Purple vertex = odd mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 11^{th} step:



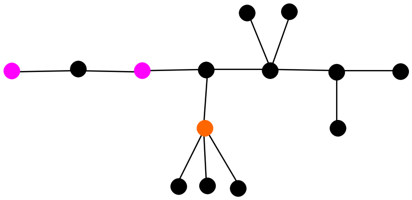
Orange vertex = hunter's position
Purple vertex = odd mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 12^{th} step:



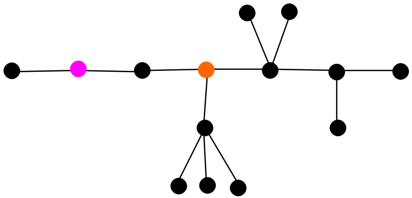
Orange vertex = hunter's position
Purple vertex = odd mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 13^{th} step:



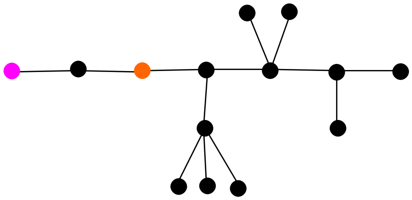
Orange vertex = hunter's position
Purple vertex = odd mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 14^{th} step:



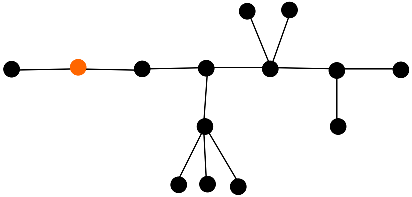
Orange vertex = hunter's position
Purple vertex = odd mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture. After hunter's 15^{th} step:



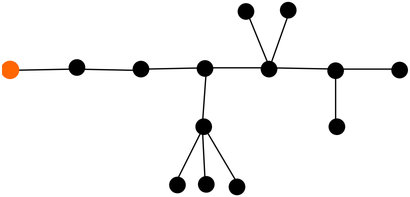
Orange vertex = hunter's position
Purple vertex = odd mole's possible positions

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture.



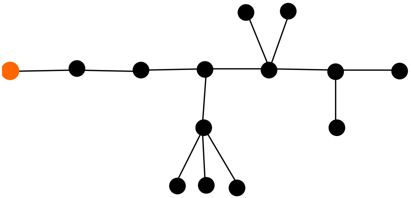
This is an optimal strategy for the hunter, by the way.

Characterization

Lemma

Lobsters are hunter-win.

Proof by picture.

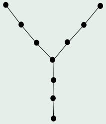


Q.E.D.!

Characterization

Lemma

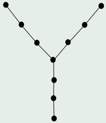
A graph G is a lobster if and only if it is a tree that doesn't contain the three-legged spider:



Characterization

Lemma

A graph G is a lobster if and only if it is a tree that doesn't contain the three-legged spider:



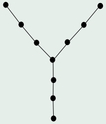
Lemma

A graph G is mole-win if:

Characterization

Lemma

A graph G is a lobster if and only if it is a tree that doesn't contain the three-legged spider:



Lemma

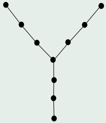
A graph G is mole-win if:

- *G is the three-legged spider.*

Characterization

Lemma

A graph G is a lobster if and only if it is a tree that doesn't contain the three-legged spider:



Lemma

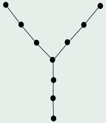
A graph G is mole-win if:

- *G is the three-legged spider.*
- *G is the cycle C_n.*

Characterization

Lemma

A graph G is a lobster if and only if it is a tree that doesn't contain the three-legged spider:



Lemma

A graph G is mole-win if:

- *G is the three-legged spider.*
- *G is the cycle C_n.*
- *G contains a mole-win subgraph.*

References

A. Bonato, P. Golovach, G. Hahn, and J. Kratochvil, The capture time of a graph, *Discrete Math.* **309** (2009) 5588–5595.

S. Gal, Search games with mobile and immobile hider, *SIAM J. Control & Opt.* **17** (1979) 99–122.

T. Gavenčiak, Games on graphs, *Charles University*, Prague (2007).

G. MacGillivray, private communication, ca. May 2011.

R. J. Nowakowski and P. Winkler, Vertex to vertex pursuit in a graph, *Discrete Math.* **43** (1983), 235–239.

R. Peyre, A probabilistic approach to Carne's bound, *Potential Anal.* **29** (2008) #1, 17–36.

A. Quilliot, Thesis, Homomorphismes, points fixes, rétractations et jeux de poursuite dans les graphes, les ensembles ordonnés et les espaces métriques, *Université de Paris VI* (1983).

N. Th. Varopoulos, Long range estimations for Markov chains, *Bull. Sci. Math.* **109** (1985) 225–252.

Thank you!

Table of Contents

Lemma

Let G be any graph and let $x_0 \in V(G)$ be any vertex in G. Let $\{x_0, x_1, x_2, \ldots\}$ be any random walk on G beginning at x_0. Then $\mathbb{P}(d(x_0, x_4) < 4) \geq 1/s$, where $s = 4n^{2/3}$.

Lemma

Expected distance after Stage 1 is less than $1+\sqrt{n(1+5\log n)}$.

Lemma

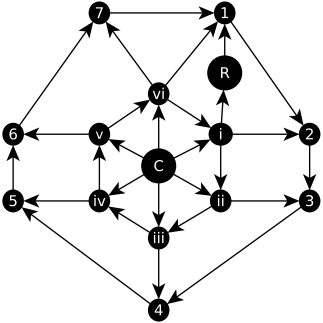
Expected distance after Stage 2 is less than $(5\log n)^{3/4}n^{1/4}$.

A quadratic digraph

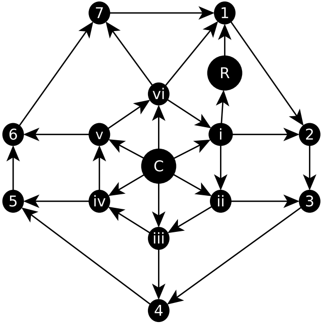
Define a $R(k)$ to be a reflexive directed graph on $n = 2k+1$ vertices consisting of:

- an "outer ring" comprised of a (counterclockwise)-directed k-cycle
- an "inner ring" comprised of a (counterclockwise)-directed $(k-1)$-cycle
- arcs from a vertex in the inner ring to a vertex in the outer ring configured such that $k-2$ vertices in the inner ring are incident with one such arc, and 1 vertex in the inner ring is incident with two such arcs
- an "internal vertex" (C) that is out-directed to every vertex in the inner ring
- an "external vertex" (R) incident with two arcs

The graph $R(7)$



The graph $R(7)$



Lemma

$R(k)$ is cop-win for all k and the capture time is $\Theta(n^2)$.