Computability Theory and the Game Cops and Robbers on Graphs

Alexa McLeod

Department of Mathematics

University of Connecticut

May 2022

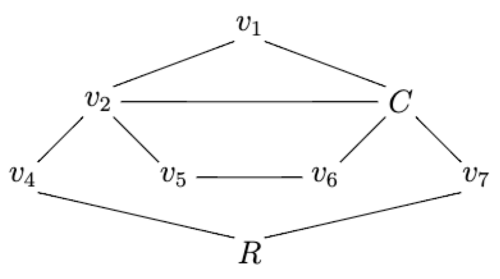
Acknowledgements

I would like to thank my thesis advisor, Dr. Reed Solomon, for his patience and guidance throughout my research process. Not only has he helped me learn more about graph and computability theory, he has served as a great role model and advisor.

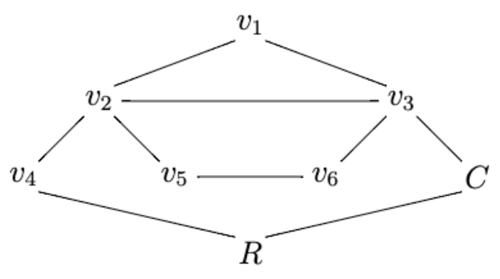
Introduction

Do you remember playing the game "cops and robbers" as a child? Now, imagine the game on graphs. Every player can only move one space for each of their turns. We can see the game being played below. We will use a R to represent the robber and a C to represent the cop.

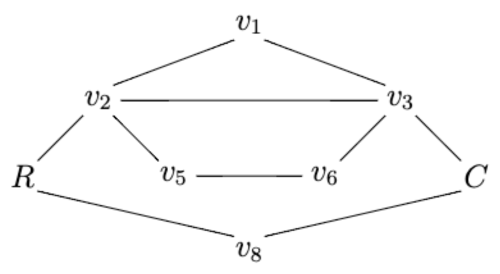
Start Positions:



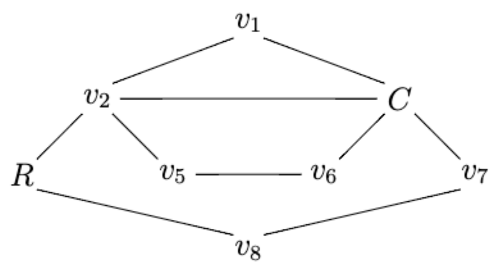
Cop's First Move:



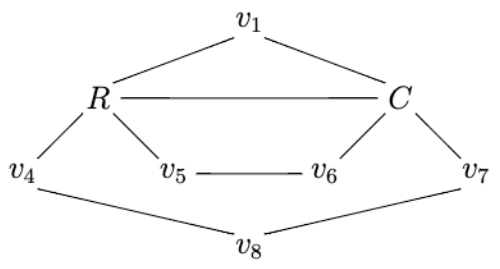
Robbers' First Move:



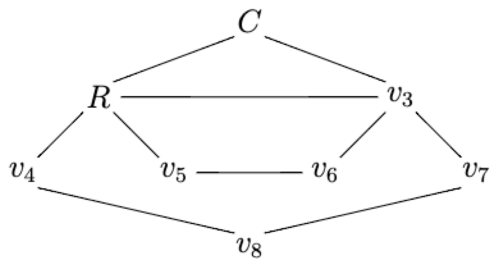
Cop's Second Move:



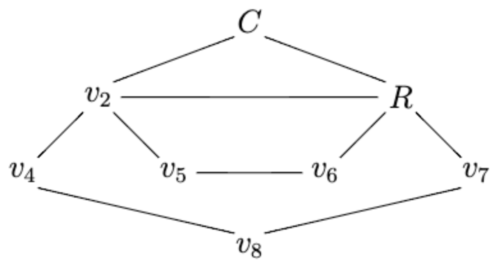
Robbers' Second Move:



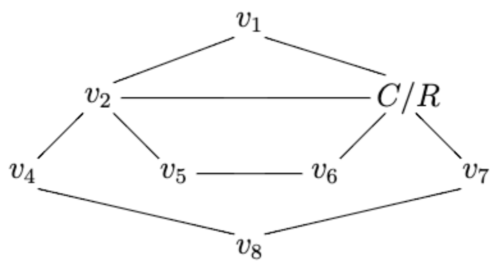
Cop's Third Move:



Robbers' Third Move:



Cop's Third Move:



We can see that the cop wins at this point because it is on the same point as the robber. The robber could have better avoided the cop by moving to v_5 instead of v_3 on his third move. In some graphs, the robber can always win if he plays with a particular strategy. We call these graphs robber-win. Graphs that can always be won by the cop if she plays correctly are called cop-win. We can also play with multiple cops. The cop number is the minimum number of cops required to win on a graph for any way the robber moves.

We can consider programs that determine the way a cop moves. These programs can be thought of as a cop's strategy. Rachel Stahl found a computable cop-win graph such that no cop strategy is computable. In this paper, we will find a computable cop-win graph such that no there is computable stategy to win for n-cops and infinitely many cops.

Introducing Graphs

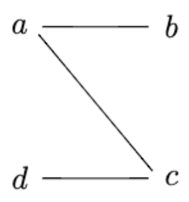
Definition: A graph G consists of

(1) A nonempty set of vertices, which we will denote $V(G)$

(2) A binary edge relation on $V(G)$, denoted by $E(G)$

We will denote an edge from vertex x_1 to vertex x_2 by (x_1, x_2). One can see from this notation then that $E(G) \subseteq V(G) \times V(G)$. A graph G is reflexive if for each vertex $x \in V(G)$, an edge from x to x exists or that $(x, x) \in E(G)$. A graph G is undirected if $E(G)$ is symmetric, meaning that if $(x, y) \in E(G)$ for $x, y \in V(G)$, then $(y, x) \in E(G)$. In this case, we can say that $(x, y) = (y, x)$. If a graph is directed, meaning that $(x, y) \in E(G)$ does not imply $(y, x) \in E(G)$, there will be an arrow on the edges showing their direction. For the purposes of this paper, we will be working with only reflexive and undirected graphs.

Example: Let G_1 be the graph shown below. It consists consists of the vertex set $V(G_1) = \{a, b, c, d\}$. For ease of notation, we will list only the nonreflexive and one from each pair of symmetric edges. Thus, the edge set is $E(G_1) = \{(a, b), (a, c), (c, d)\}$.

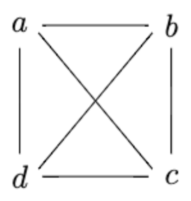


G_1

The index of a vertex x in a graph G is the number of distinct edges (x, y) where $y \in V(G)$ and $x \neq y$. For example, in the graph shown above, the index of vertex a is 2. For $x, y \in V(G)$, we say x and y are neighbors if $(x, y) \in E(G)$.

For $x_1, x_n \in V(G)$, a walk from x_1 to x_n is an ordered set of vertices $(x_1, x_2, \ldots, x_n)$ such that any two consecutive vertices are adjacent. For $x_1, x_n \in V(G)$, a path from x_1 to x_n is walk from x_1 to x_n such that no vertex appears twice. We will denote this as an (x_1, x_n)-path. We can notice that all paths are walks, but not all walks are paths.

Example: In G_2 shown below, an example of a walk is (a, c, d, c, b) and an example of a path is (a, b, c, d).

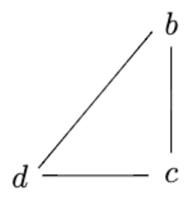


G_2

The length of a path is the number of edges it uses. In G_2, the path (a, b, c, d) has a length of 3. The distance between two points $x, y \in G$ is the length of the shortest (x, y)-path. In G_2, the distance between any two distinct vertices is 1.

Definition: A subgraph H of G is a graph such that $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. For example, G_1 is a subgraph of G_2. An induced subgraph H of G is a subgraph of G such that if $u, v \in V(H)$ and $(u, v) \in E(G)$, then $(u, v) \in H$.

For $v \in V(G)$, we define $G - \{v\}$ to be the induced subgraph of G with all points in G except for v. In other words, v and all of its edges are taken out of G. For example, $G_2 - \{a\}$ is shown below.



$G_2 - \{a\}$

Types of Graphs

We will be working with both infinite and finite graphs. If a graph G is infinite, the vertex set $V(G)$ is of infinite size. A graph G is locally finite if the index of every point is finite. We note that infinite graphs can be locally finite and that all finite graphs are locally finite.

A straight line path of length n is a finite graph with n vertices such that $n-2$ of the vertices have an index of 2 and the 2 end vertices have an index of 1. These graphs are of the form shown below.

$$v_1 — v_2 — v_3 — \;\cdots\; — v_n$$

A circle or cycle of length n is a finite graph with n vertices such that each vertex is of index 2. By relabeling, these graphs can be depicted as follows.

$$\begin{tikzcd}
v_1 \arrow[r, dash] \arrow[d, dash] & v_2 \arrow[d, dash, dotted] \\
v_n \arrow[r, dash] & v_{n-1}
\end{tikzcd}$$

Definition: If for all $x, y \in V(G)$, $(x, y) \in E(G)$, we call G a complete graph. In other words, for a complete graph G, $E(G) = V(G) \times V(G)$.

Corollary: *If a graph G is locally finite and complete, then it must be a finite graph.*

Proof. Suppose for the purpose of contradiction that G is infinite. Since G is complete, $x \in V(G)$ must share an edge with all other points in G. Since G is infinite, this would mean x shares an edge with infinitely many points. Thus, x does not have a finite index, which contradicts G being locally finite. Therefore, G must be finite.

□

An infinite ray, shown below, is an infinite graph where exactly 1 point has an index of 1 and all of the other points have an index of 2.

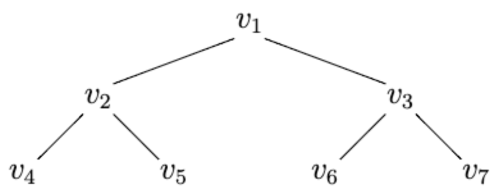
$$v_1 \text{ — } v_2 \text{ — } v_3 \text{ — } \cdots$$

An infinite 2-way ray, shown below, consists of infinitely many points, all with an index of 2.

$$\cdots \text{ — } v_1 \text{ — } v_2 \text{ — } v_3 \text{ — } \cdots$$

Definition: A graph G is connected if for all $x, y \in V(G)$, there exists a path from x to y. A cut vertex is a vertex $v \in v(G)$ such that $G - \{v\}$ is not connected.

A tree is a connected graph G where no subgraph is a cycle. These can be infinite or finite. A tree is shown below.



In the tree graph above, we can see that v_1, v_2, and v_3 are all cut vertices, but v_4, v_5, v_6, and v_7 are not cut vertices.

The Game of Cops of Robbers

The game of cops and robbers consists of $n + 1$ players for some positive integer n where there are n cops and 1 robber. To start, we will learn about the rules of the game when there is only 1 cop on a graph G.

Game Procedure:

(1) The cop picks a vertex in G to start

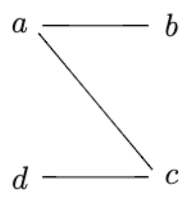
(2) The robber picks a vertex in G to start

(3) The players take turns moving from their current vertex to any neighbor of the vertex.

Since G is assumed to be undirected and reflexive, the players can move along edges in any direction and use their turn to remain at their current vertex.

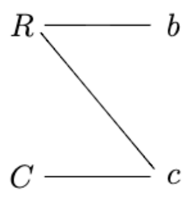
A cop wins the game if she moves onto the same vertex as the robber. The robber wins the game if the cop never moves onto the same vertex as him.

Example: We can show the game on graph G_1, shown below.

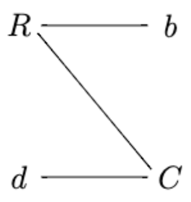


Now, we will represent the placement of the robber with R and that of the cop with C.

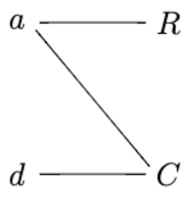
Start positions:



Cop's first move:

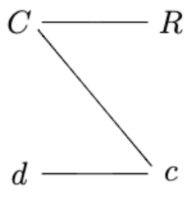


Robber's first move:



For the robber's second move, he will stay at vertex b, so we are now onto the cop's move.

Cop's second move:



From this point, the cop can win because the robber has nowhere to go and the cop is at neighbor of the robber's location.

For any graph G, either the cop can win if she plays carefully no matter what the robber does or the robber can win if he plays carefully no matter what the cop does. With this information, we classify a graph as either being cop-win or robber-win. A graph that can be won by the cop no matter how the robber plays is classified as cop-win. A graph that can be won by the robber no matter how the cop plays is classified as robber-win.

Example: We can see that the 4-cycle graph below is robber win because the robber can always move to stay 2 places further from the cop.

Start Position:

$$\begin{tikzcd}
C \arrow[r, dash] \arrow[d, dash] & b \arrow[d, dash] \\
d \arrow[r, dash] & R
\end{tikzcd}$$

Cop's First Move:

$$\begin{tikzcd}
a \arrow[r, dash] \arrow[d, dash] & C \arrow[d, dash] \\
d \arrow[r, dash] & R
\end{tikzcd}$$

Robber's First Move:

$$\begin{tikzcd}
a \arrow[r, dash] \arrow[d, dash] & C \arrow[d, dash] \\
R \arrow[r, dash] & c
\end{tikzcd}$$

Cop's Second Move:

$$\begin{tikzcd}
a \arrow[r, dash] \arrow[d, dash] & b \arrow[d, dash] \\
R \arrow[r, dash] & C
\end{tikzcd}$$

Robber's Second Move:

$$\begin{tikzcd}
R \arrow[r, dash] \arrow[d, dash] & b \arrow[d, dash] \\
d \arrow[r, dash] & C
\end{tikzcd}$$

We can see how this pattern can continue and how the robber can always move to stay 2 edges away from the cop.

Similarly, for any integer $n \geq 4$, an n-cycle graph is robber win. We can also see that if a graph is disconnected, it is automatically robber-win because the robber can start the game at a node disconnected from the cop's start position. Then, there is no path between the cop and robber, so there is no possible way for the cop to win. Since it is clear that all disconnected graphs are robber-win, we will only be focused on studying connected graphs.

Although the cop can start from any position, designating a start position does not change the nature of the game. Suppose that we designate a start position, v_0. Since we can assume the graph is connected, there must exists a (v_0, v)-path for any other vertex v in the graph. Thus, the cop can move to v in her first several moves. Thus, the game is not changed by designating a start position for the cop.

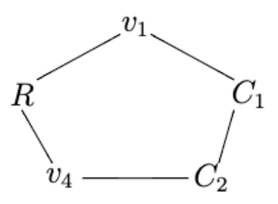
Cops of Robbers With Multiple Cops

The game can also be played with more than 1 cop. Similar to the original game, every player starts at a vertex and takes turns moving to neighboring vertices of their original vertices.

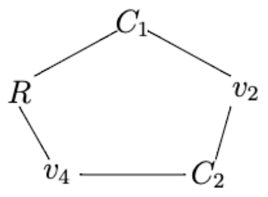
Definition: For a finite graph G, the cop number of G is the least number of cops required to make G cop-win.

Example: We can see that a 4- cycle graph has a cop number of 2.

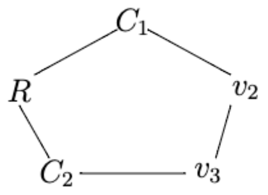
Start Position:



Cop 1's First Move:



Cop 2's First Move:



Then, we can see that the robber is cornered and Cop 1 can win on her next turn. Similarly, any n-cycle graph for any integer $n \geq 4$ has a cop number of 2 because the cops can close in on the robber from both sides. We can also see that the cops can start from a fixed vertex because they can still use their moves to close in on the robber.

Computability Theory

Computers use a program and an input of information to either give an output or to never stop running and not give any information. We can think of a computer as a function U which has the input of a computer program e and an input $x \in \mathbf{N}$. Then, the function will either give an output $y \in \mathbf{N}$ or it will diverge, meaning that it will run forever. This function is shown below.

$$U(e,x) \mapsto \begin{cases} y & \text{if the function halts} \\ \text{diverges} & \text{if the function never halts} \end{cases}$$

We can also write this function as $\Phi_e(x)$ where e represents the program code and x represents the input. Each function Φ_e is called partial computable because it might diverge on some inputs. A computable function Φ_e converges (i.e. gives an output) on all inputs.

We can assume that φ_e works on finite sequences as well as numbers. We can use the labels $V(G) = \{v_1, v_2, \ldots, v_n\}$ to describe how players in the game of cops and robbers move. For example, the sequence $\langle 3, 5, 6, 2, 4, 3 \rangle$ indicates that the cop started on v_3 while the robber started on v_5. Then, the cop moved to v_6 and the robber moved to v_2. For her third turn, the cop moved to v_4. For his third turn, the robber stayed at v_2. For the game to be played in this manner, we can see that edges (v_3, v_6), (v_4, v_6), and (v_2, v_5) must all exist.

Computability Findings

Computability With Finitely Many Cops

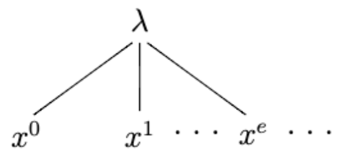
A cop strategy on a graph G is a function $f : \mathbf{N} \times \mathbf{N} \times \mathbf{N} \to \mathbf{N}$ that tells the cop how to move. We can let σ represent the sequence of moves coded by a natural number that tells us how the players have moved thus far. Then, we can say the cop's current position is at v_i and the robber is at v_j. Then, $f(\sigma, v_i, v_j)$ is the index of where the cop should move to. If the output of f describes a legal move in the game then it is a cop strategy. If f is a total computable function, then we say it is a computable cop strategy.

We can also define a cop strategy for n-cops similarly. We can have a function $f : \mathbf{N}^{n+2} \to \mathbf{N}$. Let σ represent the history of moves of the players thus far,

$c_1, \ldots, c_n$ be the locations of the cops and let r be the location of the robber. Then, $f(\sigma, c_1, \ldots, c_n, r) = (\hat{c_1}, \ldots, \hat{c_n})$ tells us that cop i should move from c_i to $\hat{c_i}$ on this turn. If the output of f describes legal moves in the game, then it is a n-cop strategy. If f is a total computable function, then we say it is a computable n-cop strategy.

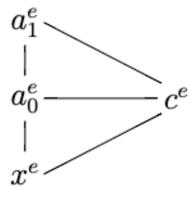
Theorem: *There exists a computable graph G that can be won with n computable cops, such that no winning strategy exists with $n-1$ computable cops and 1 non-computable cop can win.*

Proof. To start, G will consist of a root node λ and have infinitely many countable sections for the graph. We can assume that the cops start on the λ node. We will defeat each computable function Φ_e as a computable $n-1$ cop strategy.



We will expand G by building subgraphs off of each node x^e. These subgraphs will be called the e-sections of graph and are disjoint. The e-section of the graph can only be accessed through the x^e node, so each x^e is a cut vertex.

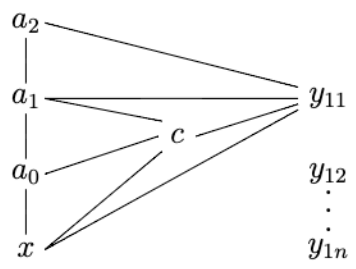
Each e-section of the graph should start with the following configuration. We can let the robber start at the node a_1^e for some e-section of the graph.



We will build the e-section of the graph by simulating how the partial computable function Φ_e controls the $n-1$ cops when the robber starts at a_1^e. The function Φ_e uses the positions of the robber and the cops to give an output telling where to move the $n-1$ cops. If Φ_e moves the $n-1$ cops into a position from which they could win in the next round, meaning that the cops are moved to a node which is a neighbor of the node that robber is at, we will expand the e-section to give the robber an escape route. This escape route will be built in such a way that n computable cops can win and 1 non-computable cop can win.

Expanding the E-Section

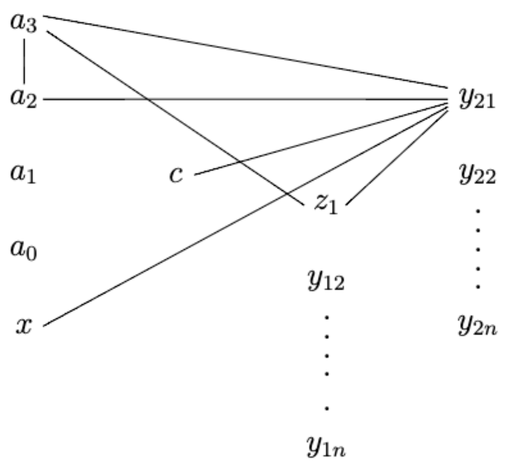
For ease of notation, we will drop the superscript on all points in the "e-section" of the graph throughout the construction of the graph. Consider Φ_e acting on the initial cop position λ and the initial robber position at a_1. We check if Φ_e moves a cop to a neighbor of a_1. Thus, we continue to watch Φ_e while keeping the robber at a_1 until a cop is moved to either a_0 or c. If not, we take no action for Φ_e. If so, we add vertices $a_2, y_{11}, y_{12}, \ldots y_{1n}$ with connections from a_2 and y_{11} shown below.



We add edges from each y_{1i} in a symmetric way. For all $i \in \{1, \ldots, n\}$, y_{1i} is connected to a_1, a_2, x_e, and c, but we will only show the connections to y_{11} for simplification. If we have $j \neq i$, then y_{1i} is not connected to y_{1j}.

After expanding G, we return to computing Φ_e to simulate how it would move the cops. At least one of the cops just moved to either a_0 or c while the robber is at a_1. The robber will move to a_2. Continue to compute the trajectory of the the cops. If the cops do not move within one space of the robber, the robber should stay at a_2.

If a cop does move within one space of the robber, to either a_1, $y_{11}, y_{12}, \dots, y_{1n}$, then we will expand the graph by adding vertices $a_3, y_{21}, y_{22}, \dots y_{2n}$. Since Φ_e only controls $n-1$ cops, we know that at least one y_{1i} does not have a cop at it. We will name this point z_1. Since y_{1i} is symmetric, we can assume without loss of generality that $y_{11} = z_1$. Then, the graph will be expanded with edges shown for y_{21}. For clarity, we will only show the new edges.

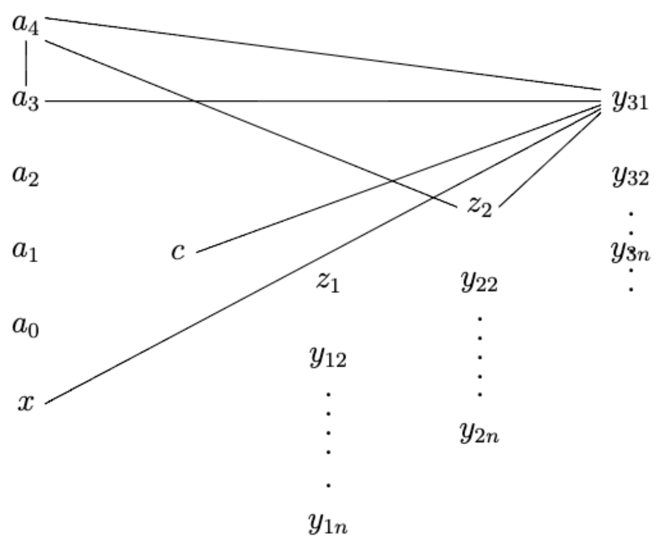


Once again, we add edges so each y_{2i} is symmetric. For all $i \in \{1, \dots, n\}$, y_{2i} is connected to a_2, a_3, c, z_1, and x_e. We also have that if $y_{1i} = z_1$, y_{1i} is connected to a_3 and if $i \neq j$, then y_{1j} is not connected to a_3.

After expanding G, we continue computing Φ_e to simulate the game. Recall that Φ_e moved at least one cop to either some $y_{1i} \neq z_1$ or a_1. We move the robber to

a_3 and continue to compute Φ_e. Then, the closest cop must be two spaces from the robber because a_3 is only connected to a_2 and $z_1 \neq y_{1i}$, if a cop is at y_{1i}.

We leave the robber at a_3 until a cop moves within one space of the robber. Suppose that a cop does move within one space of the robber. This would mean that a cop moved to either z_1, a_2, or y_{2i} for $i = 1, 2, \ldots, n$. There must be at least one y_{2i} that does not have a cop at it. We will call this point z_2. Without loss of generality, assume that $y_{21} = z_2$. We will expand the graph by adding $y_{31}, \ldots, y_{3n}$, and a_4 as follows. Once again, we will only show the new edges and the connections to y_{31} represent all the connections to every y_{3i}.



For all $i \in \{1, \ldots, n\}$, y_{3i} is connected to a_3, a_4, c, z_2, and x. We also have that z_2 is connected to a_4, but no other node of the form y_{2j} is connected to a_4.

In general, for $k \in \mathbf{N}$, a_{k+1} is connected to a_k, z_{k-1}, and y_{ki} for all $i = 1, 2 \ldots, n$. The robber stays at a_{k+1}. If Φ_e moves a cop to a_k, z_{k-1}, or y_{ki}, the graph is expanded by adding a_{k+2} and $y_{k+1,i}$. We set z_k to be the a node of the form y_{ki} that does not have a cop. We add edges from a_{k+1} and z_k to a_{k+2}. We build an edge from each $y_{k+1,i}$ to a_{k+1}, a_{k+2}, c, x, and z_k.

In the end, the e-section will either be finite or infinite. If the e-section is finite, it will consist of x, c, a_0, a_1, a_{i+1}, and $y_{i1}, \ldots y_{in}$ for $1 \leq i \leq k$. If the e-section is infinite, it will consist of x, c, b, a_0, a_1, a_{i+1}, and $y_{i1}, \ldots y_{in}$ for $i \geq 1$.

This completes the description of the construction of the graph. Now, we will check that G cannot be won with $n-1$ computable, but can be won with n computable cops or 1 non-computable cop. Moving forward, we will refer to the points $a_1, \ldots, a_i$ as the vertical segment of the e-section and the points c, and y_{jk} for $j \in [1, i-1]$ and $k \in [1, n]$ as the horizontal section. Note that x is connected to every point in the horizontal section.

Why Can't n-1 Cops Following a Computable Strategy Win?

If $n-1$ cops follow the strategy given by Φ_e, they will be unable to win if the robber starts at a_1^e and moves up the vertical segment whenever possible.

Why Can n Cops Following a Computable Strategy Win?

Suppose that n cops start at λ. We can assume the robbers start in an e-section because if the robber starts at λ, he loses.

All cops move to the x_e node. If the robber is in the horizontal section (c_e, or y_{ij}), then one of the cops can move directly there and win.

Now, assume that the robber is on the vertical segment, meaning that he is at a_i for $i \geq 1$. Then, the cops can move to $y_{(i-1)1}, \ldots y_{(i-1)n}$. Then, one of the cops is at z_{i-1}. We know a_i is connected to a_{i-1}, a_{i+1}, z_{i-2}, $y_{(i-1)1}, \ldots y_{(i-1)n}$, $y_{i1}, \ldots y_{in}$. We know that z_{i-1} is also connected to a_{i+1}, a_{i-1}, z_{i-2}, $y_{i1}, \ldots y_{in}$, so if the robber moves to any spot that doesn't already have a cop, the cop at z_{i-1} can win.

Why Can One Non-Computable Cop Win?

The cop can start at λ. Suppose the robber is in the e-section of the graph. The cop moves to x_e.

If the robber moves to one of the nodes in the horizontal section (c_e, or y_{ij}), then the cop can move directly there and win on her next turn.

If the robber is at some a_i, the cop can move to z_{i-1}. If the robber moves to a_{i+1}, a_{i-1}, z_{i-2}, $y_{i1}, \ldots y_{in}$, the cop can move directly there and win, as previously stated. The only other possible move is for the cop to go to $y_{(i-1)n}$. Then, the cop can move to z_{i-2}. The robber can either go to x, z_{i-2}, a_{i-1}, or a_i. In any of those cases, the cop can win on its next turn, so one non-computable cop can win.

$\square$

Computability With Infinitely Many Cops

We can see that if a graph is countably infinite, infinitely many cops can always win by just having a cop start at each vertex. The robber, then, is forced to share a vertex with one of the cops, so the cops win. Thus, we will require all infinitely many cops to start at a single vertex (which we will denote λ).

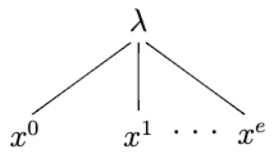
Recall that in the finite version of the game with n-cops, a strategy was a function $\Phi : \mathbf{N}^{n+2} \to \mathbf{N}$ that computed how to move the cops as a function of previous player moves and current player positions. However, we have to treat a strategy for infinitely many cops differently because we cannot code the location of infinitely many cops by a single number. A strategy in the infinite case is a function $\Phi : \mathbf{N}^3 \to \mathbf{N}$. We should let σ represent the positions of the robber in previous rounds. Let r represent the current position of the robber. Then, for each vertex $v \in V(G)$,

$$\Phi : (\sigma, r, v) \mapsto \begin{cases} 1 & \text{if there is a cop at } v \\ 0 & \text{if there is no cop at } v \end{cases}$$

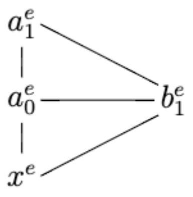
Once again, Φ is a computable strategy if it respects the rules of the game. That is, if no neighbor of v contains a cop, then Φ cannot put a cop at v in the next round.

Theorem: *There exists a computable graph G that can be won by one non-computable cop, such that no winning strategy exists with infinitely many computable cops starting at a designated node λ.*

Proof. To start, G, will consist of a root node λ and have infinitely many countable sections for the graph. We can assume that the cops start on the λ node. We will defeat the computable cops by using Φ to check if the neighboring nodes of the node with the robber have a cop at them. We can note that G is the same as the graph Rachel Stahl constructed in her cop-win graph that cannot be won with a strategy up to relabeling. We will show that this graph also can deflect a strategy for infinitely many cops.

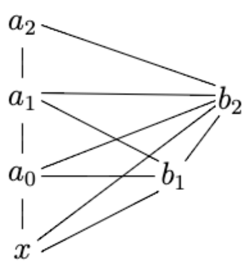


Each e-section of the graph should start consist of the following configuration. We can let the robber start at the node a_1^e for some e-section of the graph.



Expanding the E-Section

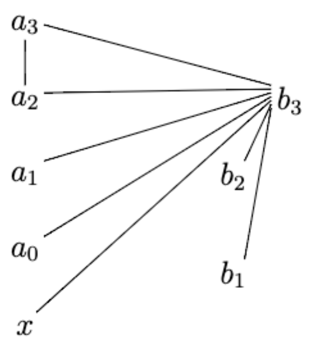
For ease of notation, we will drop the superscript on all points in the e-section of the graph throughout the construction of the graph. We use Φ_e to check if a cop moves to a neighbor of the robber's starting location at a_1. Thus, we continue to check Φ_e until a cop is moved to either a_0 or c. If not, we take no action. If so, we add vertices b_2 and a_2 as with connections shown.



We connect a_2 to a_1 and connect b_2 to all other points which are currently on the graph (x, a_0, a_1, a_2, and b_1).

After expanding G, the robber moves to a_2 and we return to computing Φ_e to check if a cop goes to a neighbor of the robber's new position. At least one of the cops just moved to either a_0 or b_1. If the cops do not move within one space of the robber, the robber should stay at a_2.

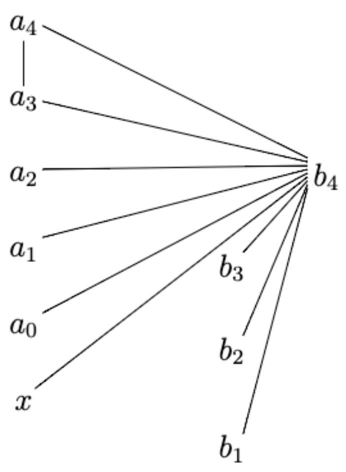
If a cop does move within one space of the robber, to either a_1 or b_2, then we will expand the graph by adding vertices a_3, and b_3 with connections as follows. For clarity, we will only show the new edges.



We connect a_3 to a_2 and connect b_3 to all other points which are currently on the graph (x, a_0, a_1, a_2, a_3, b_1 and b_2).

After expanding G, the robber moves to a_3 and we return to computing Φ_e to check if a cop goes to a neighbor of the robber's new position. At least one of the cops just moved to either a_1 or b_2. If the cops do not move within one space of the robber, the robber should stay at a_2.

If a cop does move within one space of the robber, to either a_2 or b_3, then we will expand the graph by adding vertices a_4, and b_4 with connections as follows. For clarity, we will only show the new edges.



We connect a_4 to a_3 and connect b_4 to all other points which are currently on the graph (x, a_0, a_1, a_2, a_3, a_4 b_1, b_2, and b_3).

In general, for $k \in \mathbf{N}$, a_k is connected to a_{k-1}, a_{k+1}, b_k, b_{k+1}, $b_{k+2}, \ldots$. We also know that b_k is connected to $x, a_0, \ldots a_k, b_j$ for $j \in [1, k-1]$.

In the end, the e-section will be either finite or infinite. If the e-section is infinite it will consist of $x, a_0, a_i,$ and b_i for $i \geq 1$. If the e-section is finite, it will consist of $x, a_0, a_i,$ and b_i for $1 \leq i \leq k$.

This completes the description of the construction of the graph. Now, we will check that G cannot be won with infinitely many computable cops all starting at λ, but can be won with one non-computable cop.

Why Can't Infinitely Many Computable Cops Win?

We can see that if the robber moves up the vertical segment whenever possible, the cops will never be able to catch him as they are always at least two spaces away from the robber.

Why Can One Non-Computable Cop Win?

The cop can start at λ. Suppose the robber is in the e-section of the graph and the cop moves to x_e.

If the robber moves to some b_i, then the cop can move directly there and win on her next turn.

Now suppose the robber is at a_i, then either b_{i+1} exists or there is no b_{i+1} node.

If there is a b_{i+1} node, we the cop can move to b_{i+1}. We know that a_i is connected to a_{i+1}, a_{i-1}, b_i, $b_{i+1}, \ldots$. However, b_{i+1} is also connected to all of the vertices, so the cop can win on her next turn.

If there is no b_{i+1} node, the cop can move to b_i. Since, b_{i+1} was not added, the neighbors of a_i are a_{i-1} and b_j for $j \in [1, i]$. The node b_i is connected to each of these points, so the cop can win in her next turn.

$\square$

References

(1) Bonato, Anthony and Nowakowski, Richard, *The Game Cops and Robbers on Graphs*, American Mathematical Society, (2011)./

(2) Stahl, Rachel D., *Computability Theoretic Results for the Game of Cops and Robbers on Infinite Graphs*, Archive for Math Logic 61 (2022), 373-397.