Cops and Robbers on Planar Graphs

Aaron Maurer (Carleton College)
John McCauley (Haverford College)
and Silviya Valeva (Mount Holyoke College)

Summer 2010 Interdisciplinary Research Experience for Undergraduates
Institute for Mathematics and its Applications
University of Minnesota

Advisor: Andrew Beveridge (Macalester College)
Problem Poser: Volkan Islar (University of Minnesota)
Mentor: Vishal Saraswat (University of Minnesota)
1 Introduction

We investigate the game of Cops and Robbers on graph $G = (V, E)$ on $n$ vertices. This game was introduced by Nowakowski and Winkler [4] and by Quilliot [5].

We start with $k$ cops $C_1, C_2, \ldots, C_k$ and one robber $R$. First, the cops are placed at $k$ vertices of the graph. Then the robber is placed on a vertex. During play, the cops are robber move alternately. On the cop turn, each cop may move to a neighboring vertex or remain in place. Multiple cops may occupy the same vertex. On the robber’s turn, he moves similarly.

The cops win if there is some time at which a cop is at the same vertex as the robber. Otherwise, the robber wins. The minimum number of cops required to catch the robber (regardless of robber’s strategy) is called the cop number of $G$, and is denoted $c(G)$.

Cop numbers have been an object of study since their introduction. The most influential results for this work comes from Aigner and Fromme [1]. We briefly survey these results.

Given a vertex $v$, its neighborhood is $N(v) = \{u \in V \mid (v, u) \in E\}$. The closed neighborhood of $v$ is $\overline{N}(v) = \{v\} \cup N(v)$. A vertex $v$ is dominated by the vertex $w$ if $\overline{N}(v) \subseteq \overline{N}(w)$. The following theorem is a straightforward generalization of Lemma 1 in [1].

**Theorem 1.1** If $v$ is dominated by $w$ then $c(G - v) = c(G)$.

A graph $G$ is dismantleable if we can reduce $G$ to a single vertex by successively removing dominated vertices.

**Theorem 1.2** $G$ is 1-cop win if and only if $G$ is dismantleable.

A graph homomorphism $\phi : G_1 \to G_2$ is a mapping of $V(G_1)$ to $V(G_2)$ that preserves vertex adjacency. More specifically, if $(u, v) \in E(G_1)$ then either $\phi(u) = \phi(v)$ or $(\phi(u), \phi(v)) \in E(G_2)$. If $H \subseteq G$ is a subgraph of $G$, then a retraction is a homomorphism $\phi : G \to H$ that is the identity on $H$. In this case, we say that $H$ is a retract of $G$.

**Theorem 1.3** If $H$ is a retraction of $G$ then $c(H) \leq c(G)$.

We also have a very nice characterization of the cop number for graphs with no small cycles

**Theorem 1.4** If $G$ is a graph with minimum degree $\delta(G)$ which has no 3- or 4-cycles, then $c(G) \geq \delta(G)$. 

5
Finally, they provide the following important bound on the cop number of planar graphs.

**Theorem 1.5** If $G$ is planar then $c(G) \leq 3$.

The goal of this work is to investigate the cop number of families of planar graphs. We are particularly interested in finding graphs with cop number at most 2. One of the most useful tools in this endeavor is the following lemma of Aigner and Fromme.

**Lemma 1.6** Let $u, v$ be vertices of a graph $G$. Let $P$ be a shortest path between $u$ and $v$. After a finite number of moves, a cop may prevent the robber from entering $P$.

In other words, a cop can eventually “guard” any shortest path he chooses. If the robber ever steps on that path, he will be caught. Of course, it is not enough to simply guard shortest paths: the cops must actively pursue the robber. Many cop strategies involve guarding a series of shortest paths, successively limiting the number of vertices on which the robber may freely travel.

### 1.1 To do list

We need to standardize notation

- Cops: $C_i$ or $c_i$?
- Robber: $R$ or $r$?
- Cop Number: $c(G)$ versus $C(G)$?
- Neighborhood and closed neighborhood $N(v)$ versus $\overline{N}(v)$.
- Number of edges: $e(G)$ versus $|E(G)|$.
- Edges: $e(G)$ versus $E(G)$
- Number of vertices: $v(G)$ versus $|V(G)|$

We need to standardize language

- “We move the robber” versus “The robber moves”
- Gender: “he” or “it” for cops and robbers
- Colors: Eventually support monochromatic version
2 Platonic Solids

The Platonic solids are polyhedra whose sides are congruent polygons. The five Platonic solids are tetrahedron, cube, octahedron, dodecahedron and icosahedron. We find the cop numbers of the planar graph representation of the five Platonic solids by hand. The tetrahedron is a complete graph on four vertices $K_4$, therefore a robber at any position can always be captured by one cop. The cube is not dismantlable, so one cop is not enough to capture a robber. Two cops $C_1$ and $C_2$ however can be positioned on a dominating set, such that $N(c_1) \cup N(c_2) = V(G)$. In a similar way there exists a dominating set for the octahedron and the icosahedron such that two cops can guard all vertices. For the dodecahedron two cops are not enough to trap a robber so he always has an escape. It is also a graph with no 3- or 4-cycles and a minimum degree of 3, so by $\chi(C(G)) \geq 3$. But $G$ is also planar, so $\chi(G) \leq 3$, meaning that the cop number of $G$ is 3.
3 Marking Algorithm

The marking algorithm described in [3] uses configurations which are ordered pairs of vertices representing the position of the cop and robber on the graph.

**Algorithm Mark Full Visibility One Cop**

Mark all configurations \((v, v)\) for every vertex \(v\).

**Repeat**

Mark \((c, r)\) if for all \(r' \in N(r)\), there exists a vertex \(c' \in N(c)\) with \((c', r')\) marked.

**Until** no further marking is possible.

The Theorem says that a graph is one cop win if and only if all possible configurations are marked.

The marking algorithm can easily be adapted for two cops. This time instead of pairs the configurations will be ordered 3-tuples \((c_1, c_2, r)\) representing the position of cop 1, cop 2 and robber. In the initialization, mark all configurations \((v, v, v)\), \((v, w, v)\), and \((w, v, v)\) for all vertices \(v\) and \(w\) in \(G\).

These are the trivial cases when at least one of the cops captures the robber. In the next steps we look at unmarked configurations \((c_1, c_2, r)\) and all elements of \(N(r)\), the possible moves for the robber. If for every \(r' \in N(r)\) there exists a configuration \((c'_1, c'_2, r')\) with \(c'_1 \in N(c_1)\) and \(c'_2 \in N(c_2)\), mark the configuration. This step ensures that if the robber steps on the given vertex he will be caught by one of the cops on their next move.

**Algorithm Mark Full Visibility Two Cops**

Mark all configurations \((v, w, v)\), \((w, v, v)\), and \((v, v, v)\) for all vertices \(v\) and \(w\).

**Repeat**

Mark \((c_1, c_2, r)\) if for all \(r' \in N(r)\), there exist vertices \(c'_1 \in N(c_1)\) and \(c'_2 \in N(c_2)\) with \((c'_1, c'_2, r')\) marked.

**Until** no further marking is possible.

The graph \(G\) is at most two cop win only if all possible configurations are marked.
4 Archimedean Solids

The Archimedean solids are convex polyhedra whose sides consist of two or more regular polygons. The polygons are arranged identically at each vertex and the edge lengths are all equal. There are 13 different Archimedean solids all of which can be represented by planar Archimedean graphs.

None of the Archimedean graphs is dismantlable, meaning that their cop number is at least 2. Using the Algorithm Mark Full Visibility Two Cops we found that the only 3 cop graphs are great rhombicosidodecahedron, small rhombicosidodecahedron, truncated dodecahedron, and truncated icosahedron.

In addition to using the marking algorithm, we also developed explicit proofs of some cop numbers of Archimedean solids.
4.1 Truncated Icosahedron

The truncated icosahedron contains no 3- or 4-cycles and has minimum degree $\delta(G) = 3$. By Theorem 3 in \cite{1} it follows that $C(G) = 3$. In the small rhombicosidodecahedron every vertex is part of a 3-cycle, two 4-cycles, and a 5-cycle. The only final configuration that leaves the robber no escape is if the 2 cops occupy the opposite corners of the squares. At this point robber stays put until one of the cops makes a move leaving him an escape. Therefore robber has an escape strategy at every point of the graph, meaning that 3 cops will be needed to capture him.
4.2 Truncated Tetrahedron

**Theorem 4.1** The graph of the truncated tetrahedron is two cop win.

**Proof.** We show that the truncated tetrahedron is a two cop win graph by providing a strategy for two cops to capture a robber irrespective of both their and his starting position. This strategy consists of two different phases. The first is for the cops to move to vertices such that the shortest path between them is of maximal length, in this case 3. In the second phase, the cops move to capture the robber, who will have chosen one of the remaining vertices.

4.2.1 Placement Phase

The Cops will place themselves on two vertices with a distance of three edges apart. By symmetry, there is only one such placement. While the cops are placing themselves and then waiting for their turn, the robber will either be incidentally caught, or end up on one of the remaining vertices.

4.2.2 Capture Phase

Accordingly, with the cops’ move, the second phase begins. The graph appears as such, with the cops on vertices marked in blue and the robber on one of the remaining vertices marked in red or green.

![Diagram of truncated tetrahedron with vertices marked in blue, green, and red.]

If the robber is on a vertex marked in green, then he is within one edge of a cop, and that cop can immediately capture him.

If the robber is on one of the four vertices marked in red though, this is not the case. Instead, the cops can make the moves marked in blue to the circled vertices for the respective red vertices. In each case, the vertices the cops move to are a dominating set of the neighborhood of the robber, mainly
every vertex within an edge of the robber is also within an edge of a cop, so irrespective of what the robber does, he will be captured on the cops subsequent turn.

4.2.3 Conclusion

By this strategy, two cops can capture a robber on a truncated tetrahedron irrespective of starting position, so the graph is two cop win. □

4.3 Small Rhombicubocathedron

Theorem 4.2 The graph of the small rhombicubocathedron is two cop win.

Proof. The cops strategy involves two phases. The first of these is to move to two preselected vertices, leaving the robber either captured or on one of the remaining vertices. The second phase is to then move to capture the robber on whichever vertex the robber ended up on.

4.3.1 Placement Phase

The first phase of the strategy is for the cops to place themselves at two predetermined vertices at distance five apart. While they are doing so, the robber will choose one of the remaining vertices. After the cops reach their vertices and wait for their turn once more, the capture phase of the strategy can begin.
4.3.2 Capture Phase

At the beginning of the second phase of the strategy, the possible robber positions can be divided into three groups, as denoted by green, red, and yellow vertices, with the cops on the two blue vertices:

It is fairly obvious from the picture that the set of red vertices are all reflections of the green vertices across an axis going through the two blue vertices, and accordingly, the strategy to capture a robber on the each corresponding green vertex can be reflected so as to capture a robber on the red vertex. Moreover the two cop vertices can be transposed, and accordingly each yellow vertex, and a corresponding capture strategy for the cops, is a transposition of one of the yellow or green vertices. Accordingly, if there exist capture strategies for the six green vertices, then there are strategies by which the cops can capture robbers on any of the remaining vertices.

**Trivial Cases** If the robber is on either of the vertices which share an edge with a cop, the cop who is next to him immediately captures him.

**Non-Trivial Case 1**
Here, the robber begins on the vertex shaded in red. The cops make the moves marked “1” to the blue circled vertices. The robber either stays put, in which case he can not get captured on the next turn, or moves and is within one edge of a cop and is captured on the next turn.

If the robber stays put, the cops then make the moves marked “2” in blue to the according vertices. Unless the robber makes the move “2,” in red, to the vertex circled in red, he will get captured on the next turn, since all his other options places him neighboring a cop.

If the robber makes this move, the top cop moves to his previous location as marked by “3” and the other stays still. They are now on the purple vertices with the robber on the vertex circled in red. The purple vertices are a dominating set of the neighborhood of the robber; the robber’s vertex and all neighboring vertices are within an edge of one of the cops. Thus, whatever the robber does, he is captured on the next turn, and the cops win.

Non-Trivial Case 2
In this case, with the robber on the vertex marked in red, the two cops immediately move to the purple vertices, which are a dominating set of the neighborhood of the robber, and the robber loses on the next turn, irrespective of where he moves.

**Non-Trivial Case 3**

Here, the cops make the move marked “1” to the two vertices circled in blue. From here, the robber is either captured on the next turn or takes his move “1”. Now, the cops take their moves marked “2,” one to its starting vertex and the other to the vertex marked in solid purple. These two vertices are a dominating set of the robber’s neighborhood, so the cops win.

**Non-Trivial Case 4**
Should the robber start on the red vertex, the one cop shall move as shown. From here, the robber actually has two choices that won’t result in his immediate capture. The first is to stand still. The second is to move to the vertex circled in orange.

If the robber stays put, one cop takes the move “2” to the purple vertices, which dominate the robber’s neighborhood. Thus, the cops win.

If the robber takes his move “1” as shown, then each cop takes their move “2,” forcing the robber to take his move “2” or get captured, from here both cops take their moves “3,” placing them in a dominating set of the robber’s neighborhood, and ensuring his capture.
4.3.3 Conclusion

We can now conclude that from the beginning of phase two, the cops have a strategy to capture the robber from wherever he ends up. Accordingly, using this strategy from the start, two cops, from wherever they might have started, can capture a robber on this graph, irrespective of where he starts. Though this strategy will probably not capture a robber in the fewest number of moves in most circumstances, it does prove that his graph is two cop win.
4.4 Snub Cube

Since the graph of the snub cube (hereafter referred to as $G$) is not dismantleable, $c(G) \geq 2$. This paper serves to show that $c(G) = 2$. The proof is developed by offering strategies for $C_1$ and $C_2$ for any beginning location of $R$. No strategy requires more than three turns to “trap” $R$, with the assumption that $R$ will be captured the next turn. We say $R$ is “trapped” when $N(R) \in N(C_1) \cup N(C_2)$. As a general rule, a blue vertex will indicate the location of a cop, a green vertex will indicate the location of $R$, and a red vertex will indicate a vertex threatened by a cop.

4.4.1 Placing the cops

When looking at the graph, we wish to place the cops as follows. For all cases, assume the cop on the upper vertex is $C_1$ and the cop on the lower vertex is $C_2$. The black vertices are the ones where $R$ may be safely placed to start. Once we have eliminated potential starting vertices, they will be colored yellow for the remainder of the proof, unless occupied or threatened in a given configuration.

4.4.2 Configurations

4.4.3 Corners

Now, suppose $R$ is placed in the top left corner. We move $C_1$ straight up and $C_2$ to the lower left corner. This traps $R$. An analogous strategy will trap $R$ if it is placed in the upper right corner:

4.4.4 Middle Square

Suppose $R$ is placed on the left vertex of the middle square. We move $C_1$ down and to the left, and move $C_2$ straight up. This traps $R$. An analogous strategy will trap $R$ if it is placed on the right vertex
Suppose $R$ is placed on the bottom vertex of the middle square. Then $C_1$ moves straight down and $C_2$ moves straight up. This traps $R$.

### 4.4.5 Inner Octagon

Suppose $R$ is placed on the left vertex of the inner octagon. We move $C_1$ down and to the left, and move $C_2$ up and to the left. This traps $R$. An analogous strategy will trap $R$ if it is placed on the right vertex of the inner octagon.
4.4.6 Outer Octagon

Suppose $R$ is placed on the left vertex of the outer octagon. We move $C_1$ down and to the left, and move $C_2$ down and to the left. This traps $R$. An analogous strategy will trap $R$ if it is placed on the right vertex of the outer octagon.

4.4.7 The last vertex

Suppose $R$ is placed on the upper-right vertex of the outer octagon. We keep $C_1$ still and move $C_2$ up and to the left. This gives $R$ two options: staying on the same vertex or moving to the upper right corner. If $R$ chooses the former, we keep $C_1$ still and move $C_2$ up and to the right. If $R$ chooses the latter, we move $C_1$ straight up and move $C_2$ up and to the right.
This shows that 2 cops may always catch a robber on the graph of the snub cube. □
(k) Beginning...

(l) ...middle...

(m) ...end 1.

(n) ...end 2.
5 Johnson Solids

The Johnson solids are another family of convex polyhedra, with 92 members. Every member of this family has regular faces and equal edge lengths. The Johnson solids do not include the Platonic solids or the Archimedean solids. This family also excludes the infinite families of prisms and antiprisms. We ran the marking algorithm on the family of Johnson solids to determine their cop numbers.

Two of the Johnson solids are dismantleable: The square pyramid and the pentagonal pyramid.

Two of the Johnson solids are three-cop win: the augmented truncated dodecahedron and the parabiaugmented truncated dodecahedron.

The remaining 89 Johnson solids are all two-cop win.
6 Series Parallel Graphs

A 2-terminal graph $G$ is a graph with two specified distinct vertices $s$ and $t$, called the source and sink respectively. A series composition of the two terminal graphs $(G_1, s_1, t_1)$ and $(G_2, s_2, t_2)$ is a graph with source $s_1$ and sink $t_2$ obtained from the disjoint union of $G_1$ and $G_2$ by identifying $t_1$ and $s_2$. A parallel composition of the same 2-terminal graphs is obtained by identifying their sources and sinks to form the source and sink of the resulting 2-terminal graph. A graph is a 2-terminal series parallel graph if it can be constructed by a sequence of series and parallel compositions starting from a set of copies of a single-edge graph with two vertices. Finally, a graph $G$ is a series parallel graph (sp-graph) if there is some pair of vertices $s, t \in V(G)$ such that $(G, s, t)$ is a 2-terminal series parallel graph.

The minimum degree of a series parallel graph is $\leq 2$. Note that the only case where it can be 3 or more is if the graph has parallel edges, which in the context of pursuit-evasion games are always deleted. Cycles can be of any length and a robber can move from one series parallel subgraph to another only through the terminals.

A series parallel graph is one cop win if it is dismantlable. These are the cases when $G$ is a path, a tree, or a 3-cycle. Any series composition of paths, trees, and 3-cycles is also one cop win. In all other cases $G$ is two cop win. Since parallel composition creates cycles of any length, one cop will not be enough.
Theorem 6.1 If $G$ is a series parallel graph then $c(G) \leq 2$.

Proof. The idea of the proof is that 2 cops can always push a robber in a smaller series parallel subgraph and prevent him from escaping that subgraph. Consider a sp-graph $G$ with $m = e(G)$. If we start with $c_1$ on $s$ and $c_2$ on $t$ then $c_1$ and $c_2$ can catch the robber. Moreover, $r$ can never step to $s$ or $t$. We use induction on the number of edges $m = e(G)$. The base case is $m = 1$, where trivially two cops capture a robber. Assume it is true that two cops can capture a robber on a sp-graph graph with $e(G) < m$.

Case 1: If $G$ is a parallel composition of two sp-subgraphs $G_1$ and $G_2$, then the robber is already trapped in one of them. In any case $e(G_1) < e(G)$ and $e(G_2) < e(G)$.

Case 2: If $G$ is a series composition of two sp-subgraphs $G_1$ and $G_2$, identify the cutpoint $q$ where the sink and source of the graphs were merged. Pick the shortest paths $P_1$ from $s$ to $q$ and $P_2$ from $t$ to $q$ to be the corridors of the cops. Denote the shortest distance from a vertex to the sink $t$ as height of the vertex $h(v)$. At the start of the game if a robber is at either $s$ or $t$ he gets caught. Assuming we are not at the trivial case, then $h(c_2) < h(r) < h(c_1)$. After the robber’s move the positions change to $h(c_2) \leq h(r) \leq h(c_1)$. It is always true that $h(c_2) \leq h(q) \leq h(c_1)$ and after each turn at least one cop gets closer to $q$ with the following strategy:

Case 2.1: If $h(c_2) < h(r) < h(c_1)$, $c_1$ moves towards $q$ on $P_1$ and $c_2$ moves towards $q$ on $P_2$.

Case 2.2: If $h(c_2) = h(r) < h(c_1)$, $c_1$ moves towards $q$ on $P_1$ and $c_2$ moves back towards $t$ on $P_2$.

Case 2.3: If $h(c_2) < h(q) \leq h(r) = h(c_1)$, $c_1$ moves back towards $s$ on $P_1$ and $c_2$ moves towards $q$ on $P_2$.

If either cop encounters $r$ along the way, the game is over. We say that the robber threatens a cop $c_i$ if after the robber’s move $h(r) = h(c_i)$. If robber never threatens either cop before one of them gets to $q$, the other one can go back to his starting position terminal and they successfully trap $r$. Otherwise assume without loss of generality that $c_2$ is threatened first. Following the strategy in Case 2.2 cop $c_2$ moves back and keeps a height $h(c_2) = h(r) - 1$. From now on if $r$ steps above $q$, then $c_2$ steps to $q$. Cop $c_1$ can reset to $s$ and they apply the strategy again. Similarly if $r$ does not step above $q$, then $c_1$ gets to $q$ first and cop $c_2$ can reset to $t$. By induction, after a finite number of moves cops $c_1$ and $c_2$ capture the robber. □
In generalized series parallel graphs one additional operation is allowed. The source merge of the two 2-terminal graphs \((G_1, s_1, t_1)\) and \((G_2, s_2, t_2)\) is the 2-terminal threshold graph \(G\) created by merging their sources to become the new source. The sink is \(t_1\), so there is a component isomorphic to \(G_2\) that has the same source but has no sink.

**Corollary 6.2** The cop number of a generalized series parallel graph \(G\) is \(C(G) \leq 2\).

**Proof.** Similar to the series parallel graph strategy can be applied with two cops. Whenever robber is in a no sink component \(G'\), he can only escape to another subgraph through the source. While one cop guards the source move the other cop towards him and into \(G'\). Identify the sink of \(G'\) and position one cop there. We are now playing in a sp-graph. \(\square\)
7 Prism Graphs

Prism graphs are the planar representation of the prism polyhedra. Two of the faces are congruent polygons and all the other faces are parallelograms. Therefore the planar graphs have an inner and an outer $n$ cycle with a ladder between them.

![Prism Graphs Diagram]

**Theorem 7.1** Two cops can capture a robber on a prism graph.

**Proof.** The prism graph is the Cartesian graph product of a cycle of length $n$ with a path on 2 vertices. Therefore the vertex set of a prism graph is $V(G) = ((v_i, u_j))$ where $i = 1, \ldots, n$ and $j = 1, 2$. Place $c_1$ at $(v_1, u_1)$ and $c_2$ at $(v_1, u_2)$ and move cops in opposite directions on their cycles. After $\leq n/2$ moves we have robber at $(v_i, u_j)$ and either $c_1$ is at $(v_{i-1}, u_1)$ or $c_2$ at $(v_{i+1}, u_1)$. Without loss of generality assume $c_1$ is at $(v_{i-1}, u_1)$. Cop $c_2$ moves to $(v_{i+1}, u_2)$ along the outer cycle forcing the robber to stay put. Robber is captured on the next move. □

Another similar family of graphs is the antiprism graphs. Antiprism graphs consist of two congruent polygons connected by triangles. An $n$-antiprism graph has $2n$ vertices and $4n$ edges.

**Theorem 7.2** Two cops can capture a robber on an antiprism graph.

**Proof.** Similar to the proof for prism graphs. □

8 Möbius Ladder Graphs

Möbius Ladder graphs are constructed by introducing a twist in a Prism graph. Since two edges have to cross these graphs are non-planar.

![Möbius Ladder Graphs Diagram]
Theorem 8.1 If \( G \) is a Möbius Ladder graph, then \( c(G) = 2 \).

We give a graph homomorphism \( \phi : H \to G \) that takes vertices in a prism graph \( H \) of order \( n \) to vertices in a Möbius Ladder graph \( G \) of order \( n - 1 \). Define \( \phi \) as:

\[
\begin{align*}
\phi((v_i, u_j)) &= (v_i, u_j) \text{ for } 1 \leq i \leq n - 1 \text{ and } 1 \leq j \leq 2 \\
\phi((v_n, u_1)) &= (v_1, u_2) \\
\phi((v_n, u_2)) &= (v_1, u_1)
\end{align*}
\]

The function \( \phi \) takes vertices in \( V(H) \) to vertices in \( V(G) \) and \( ((v_i, u_j), (v_k, u_l)) \in E(H) \), implies \( (\phi(v_i, u_j), \phi(v_k, u_l)) \in E(G) \). Therefore \( \phi \) is a graph homomorphism that takes prism graphs to Möbius ladder graphs. Since \( C(H) = 2 \) it follows by Theorem 2 in [1] that \( C(G) = 2 \).
9 Cartesian Products with Paths and Cycles

Prism graphs are the Cartesian graph products of a cycle of arbitrary length with a path on 2 vertices. The prism graphs $Y_{n,m}$ are the Cartesian graph products of a cycle of length $n$ with a path of length $m$. Two cops can capture a robber on any prism graph. For the sake of the proofs, define the subgraph $G_j$ to be the induced copy of $G$ on $\{(v_i, u_j)\}$. The shadow of a vertex $(v_i, u_j)$ in $G_k$ is the vertex $(v_i, u_k)$.

**Theorem 9.1** The Cartesian graph product $C_n \square P_m$ where $C_n$ is a cycle on $n$ vertices and $P_m$ is a path on $m$ vertices has cop number 2.

**Proof.** *Base case:* The Cartesian graph product $C_n \square P_2$ is an $n$-prism graph where we know cop number is 2. Suppose that $C(C_n \square P_m) = 2$ and consider $C_n \square P_{m+1}$. Define shadow $C^k_n = \{(u_i, v_k)\}$. Two cops play on $C_n \square P_m$ where they either capture the robber or his shadow in $C^m_n$. Cop $c_1$ keeps distance 1 from the shadow preventing $r$ from stepping back in $C_n \square P_m$ and from moving backwards on the cycle $C^m_{n+1}$. Meanwhile cop $c_2$ moves in $C^m_{n+1}$ on the path to the $C^m_{n+1}$ shadow of $c_1$ containing $r$. Robber is trapped in a prism graph where in a finite number of moves two cops capture him. □

**Theorem 9.2** A Möbius grid has cop number 2.

**Proof.** A generalized homomorphism can be defined that takes prism graphs $Y_{n,m}$ to Möbius strips of order $n-1$ on length $m$ ladders. Let $\phi : H \rightarrow G$ be defined as:

$$\phi((v_i, u_j)) = (v_i, u_j) \text{ for } 1 \leq i \leq n-1 \text{ and } 1 \leq j \leq m$$

$$\phi((v_n, u_j)) = (v_1, u_{m-j+1}) \text{ for } 1 \leq j \leq m$$

The function $\phi$ takes vertices in $V(H)$ to vertices in $V(G)$ and $((v_i, u_j), (v_k, u_l)) \in E(H)$ implies $((\phi(v_i, u_j), \phi(v_k, u_l)) \in E(G)$. Therefore $\phi$ is a graph homomorphism and since $C(H) = 2$ it follows by Theorem 2 in [1] that $C(G) = 2$. □
**Theorem 9.3** The Cartesian graph product $C_m \Box C_n$ where $C_m$ is a cycle on $m$ vertices and $C_n$ is a cycle on $n$ vertices has cop number 3.

**Proof.** Cops chase a robber’s shadow in $C_m^i$. After cop $c_1$ gets to the shadow of the robber in $C_m^i$ he mirrors his movements, preventing him from stepping on that cycle. Now robber can safely move on a $C_m \Box P_{n-1}$ where 2 cops can always capture him. \qed

**Theorem 9.4** The Cartesian graph product $P_m \Box P_n \Box P_r$ has cop number 2.

**Proof.** Base case: $P_m \Box P_n \Box P_2$ consists of two copies of a $m \times n$ grid connected with edges. Two cops chase the robber or his shadow in one of the grids. When one cop $c_1$ is on the robber’s shadow he prevents him from going back and forces him to always make a move on the top grid. Second cop can now move on the top $m \times n$ grid and after a certain number of moves position himself on the shadow of $r$ in $P_m^1$. From now on cop $c_2$ can always preserve distance 1 one from the robber’s shadow and advance towards $P_m^n$. Robber can move along $P_m^i$ until he gets to the edge and is forced to move to $P_m^{i+1}$. After a finite amount of moves robber will be pushed in a corner square in $P_m^n$ with no escape. 

**Inductive step:** Suppose $P_m \Box P_n \Box P_r$ has cop number 2. Consider $P_m \Box P_n \Box P_{r+1}$. The two cops capture the robber’s shadow in $P_m \Box P_n \Box P_r$ and trap him in the top grid. Again by standing on the opposite corner of the square the second cop can push the robber into smaller grids until he has no escape. \qed
10  Co-Normal Products of Graphs

For graphs $G$ and $H$, their co-normal product, $G \times H$ is defined as the graph whose vertices are the Cartesian product of the sets of vertices of $G$ and $H$, $V(G) \times V(H)$, and whose vertices $(u, v)$ and $(u', v')$, for $u, u' \in V(G)$ and $v, v' \in V(H)$ share and edge if and only if $u$ and $u'$ share an edge in $G$ or $v$ and $v'$ share an edge in $H$. It turns out that cop number of $G \times H$, $C(G \times H)$ is either always one or two.

**Theorem 10.1** Let $G$ and $H$ be connected graphs with multiple vertices. The co-normal product of these graphs has at most cop number two

**Proof.** No matter what, two cops can always capture a robber in a co-normal product of two graphs $G$ and $H$. Let the two cops be known as $c_1$ and $c_2$. At the beginning of the cops’ turn, let the robber be at $(s, t)$, $c_1$ at $(a_1, b_1)$, and $c_2$ at $(a_2, b_2)$. Since $G$ and $H$ are connected and have multiple vertices, there must exist $x \in G$ and $y \in H$ such that $x$ and $a_1$ share an edge and $y$ and $b_2$ share an edge. Thus, in $G \times H$, $(a_1, b_1)$ and $(x, t)$ are connected as are $(a_2, b_2)$ and $(s, y)$. Thus $c_1$ can move to $(x, t)$ and $c_2$ can move to $(s, y)$. At this point, if the robber leaves $(s, t)$, then it must be to a point $(u, v)$ such that either $s$ and $u$ are neighbors in $G$, in which case the vertex will neighbor $(s, y)$ in $G \times H$, or $t$ and $v$ are neighbors in $H$, in which case the vertex will neighbor $(x, t)$ in $G \times H$. In both these situations, either $c_1$ or $c_2$ would immediately capture the robber.

There must be a path from $x$ to $s$ in $G$ since $G$ is connected. If this path is put in $G \times H$ by replacing each vertex $z$ in it with $(z, t)$, there is now a path from from $(x, t)$ to $(s, t)$ such that every vertex is of the form $(z, t)$ accordingly, if $c_2$ takes this path, either in a finite number of moves he will reach the robber’s vertex, and the robber will be captured, or the robber will move and be captured. Thus, two cops are always sufficient to defeat a robber in a co-normal product graph.

**Theorem 10.2** A co-normal product of two graphs $G$ and $H$ is one cop win if and only if both graphs are one cop win and at least one of them is dominated by a single vertex, which is to say there exists a vertex in the graph such that the one vertex shares an edge with all other vertices.

**Proof.** Let $G$ and $H$ both be one cop win and without loss of generality say that $H$ is dominated by a single vertex $x$. The cop can move in $G \times H$ to a vertex of the form $(z, x)$ for some $z \in G$. Since every vertex in $H$ shares and edge with $x$, $(z, x)$ shares an edge with every vertex not of the same form, so
after his turn either the robber is on a vertex \((a, x)\) or he is captured by the cop. Since \(G\) is one cop win, the cop has a strategy to capture a robber on \(a\) when starting on \(z\). By employing this strategy in \(G \times H\) by similarly sticking to vertices whose corresponding vertex in \(H\) is \(x\), the robber will either be captured in a finite number of moves or will move to a vertex not of the form \((a, x)\) and be captured. Thus \(G \times H\) is one cop win.

However if neither \(G\) nor \(H\) is dominated by a single vertex this is not true. Let neither \(G\) nor \(H\) be dominated by a single vertex. If the cop is going to win, he must begin his turn within one edge of the robber so as to capture him. Let this be the case with the cop on vertex \((u, v)\) and the robber on vertex \((s, t)\). Since \((u, v)\) neighbors \((s, t)\) in \(G \times H\), \(u\) neighbors \(s\) in \(G\) or \(v\) neighbors \(t\) in \(H\). Let us say without loss of generality that \(u\) neighbors \(s\) in \(G\). Since \(v\) does not dominate \(H\), there has to be a vertex \(q \in H\) such that \(q\) and \(v\) are not neighbors. Accordingly, \((u, q)\) does not neighbor \((u, v)\) in \(G \times H\) but does neighbor \((s, t)\). Thus, the robber can move to \((u, q)\) and no longer be next to the cop. Since he can do this whenever the cop gets next to him, he can never be captured and \(G \times H\) is not one cop win.

Furthermore, even if one of the graphs is dominated by a single vertex, the product will still not be one cop unless the other graph is one cop win. Without loss of generality, let \(G\) be not one cop win and \(H\) be dominated by a single vertex. Since \(G\) is not one cop win, in \(G\), where ever just one cop is on the graph, there is at least one vertex from which the robber has a strategy to avoid the cop indefinitely. This must be true since otherwise the cop could just move to the vertex from which he can capture a robber at any vertex and have a winning strategy. The robber can take advantage of this in \(G \times H\). Since showing that there is one initial position from which a cop cannot definitively capture a robber in a graph is sufficient to show the graph is not one cop win, let the cop start at vertex \((u, v)\) and the robber at vertex \((s, v)\) in \(G \times H\) such that if the cop was at \(u\) and the robber was at \(s\) in \(G\) the robber would have a winning strategy. Obviously, the cop cannot capture the robber in one move from this position, so he has to try to position himself better by moving. In moving to a vertex \((u', v')\) trying to catch the robber, either \(u\) and \(u'\) are connected in \(G\) or \(v\) and \(v'\) are connected in \(H\). If the move is such that \(u\) and \(u'\) are connected in \(G\), the robber can just employ his strategy in \(G\), moving to \((s', v')\), where \(s'\) is where he would move to in \(G\) as dictated by his strategy for when the cop moves to \(u'\). If the cops move is such that \(u\) and \(u'\) are not connected in \(G\), then \(v\) and \(v'\) must be connected in \(H\). In this case, the robber can move to vertex \((t, v')\) such that a robber starting on \(t\) in \(G\) would have a winning strategy against a cop starting on \(u'\). In either situation, the cop and robber have reset themselves to their original positions, and since the robber can repeat this indefinitely, the robber has a
winning strategy, and $G \times H$ is not one cop win.

Thus, a co-normal product of graphs is one cop if and only if one graphs is one cop and the other is dominated by a single vertex.
11 Lexicographic Product of Graphs

Let $G$ and $H$ be graphs. Their lexicographic product, $G \times H$ is defined as the graph whose vertices are the Cartesian product of the sets of vertices of $G$ and $H$, $V(G) \times V(H)$, and whose vertices $(u, v)$ and $(u', v')$, for $u, u' \in V(G)$ and $v, v' \in V(H)$ share and edge if and only if $u$ and $u'$ share an edge in $G$ or if $u = u'$ and $v$ and $v'$ share an edge in $H$. It turns out that the cop number of $G \times H$, $C(G \times H)$, is determined by $C(G)$ and $C(H)$.

**Theorem 11.1** Let $G$ and $H$ be graphs with cop numbers $C(G)$ and $C(H)$ respectively. $G \times H$ will have $C(G \times H) = C(G)$ if $C(G) > 1$ or if $C(H) = 1$. If $C(G) = 1$ and $C(H) > 1$, then $C(G \times H) = 2$

**Proof.** First, it is fairly straightforward to see that it can not be the case that $C(G \times H) < C(G)$. If there are less than $C(G)$ cops, the robber has a strategy to never get caught in $G$. This strategy will still work in $G \times H$ if the robber treats a vertex $(u, v)$ in $G \times H$ as the vertex $u$ in $G$, so the robber can escape the cops indefinitely in $G \times H$ as well.

Now, if there are $C(G)$ cops, the cops can employ the same strategy they had to win in $G$ in $G \times H$ by similarly treating a vertex $(u, v)$ in $G \times H$ as the vertex $u$ in $G$. In a finite number of terms, this will result in a cop and the robber being at vertices $(a, b)$ and $(a, c)$ respectively for some $a, b, c$. At this point, the robber is essentially stuck in one version of $H$, since he can only stay still or move to vertices $(u', v')$ such that $a = u'$ and $b$ and $v'$ share an edge in $H$. If he moves to a vertex $(u', v')$ such that $u' \neq a$, then $u'$ and $a$ share an edge in $G$, so the cop who is positioned at $(a, c)$ can immediately capture him there.

If $C(G) > 1$, a second cop other than the one at $(a, c)$ can move to a position $(x, y)$ such that $x$ and $a$ share an edge in $G$. From here, if the robber moves to a vertex $(u', v')$ such that $u' \neq a$ the first cop will immediately catch him, and if he stays put or moves to a vertex $(a, v')$ the second cop can immediately catch him, so the cops will win on the next turn, and $C(G \times H) = C(G)$.

If $C(H) = 1$ and $C(G) = 1$, then the lone cop can still employ the same strategy as he would in $G$ to corner the robber in one version of $H$ as previously mentioned. From there, he has a strategy to win in $H$, which he can employ by only making moves such that $u' = a$ until the robber is caught, so once again $C(G \times H) = C(G)$.

However, if $C(H) = 1$ and $C(G) > 1$ and there is only one cop the robber has a strategy to avoid the cop indefinitely. If the robber is at a vertex $(u, v)$ and the cop is at a vertex $(c, d)$, he will move to a
vertex \((u', v')\) such that if he were at \(v\) and the cop were at \(d\) in \(H\), the robber’s strategy would dictate he move to \(v'\), and if \(u\) and \(c\) at neighbors in \(G\), \(u' = c\), otherwise \(u' = u\). In this way, the robber will always be able to avoid the cop, since if he is at \((u, v)\), the cop will never be able to get to a vertex \((c, d)\) such that \(v = d\).

As described though, a second cop would be able to ensure the robbers capture, so if \(C(H) = 1\) and \(C(G) > 1\), then \(C(G \times H) = 2\). \(\square\)
12 Maximal Planar Graphs

A triangulated graph $G$, also known as a maximal planar graph, is a planar graph such that the addition of any more edges to $G$ would result in $G$ no longer being planar. This implies that every face has three edges (except in the case of $C_2$), since if there was a face with more than three edges, there could be an edge between two vertices on it that aren’t neighbors without making the graph non-planar, and if a face had two edges, as is only possible if $G$ is a path of length two, an edge could be added to make it a three cycle without making it non-planar.

12.1 Triangulated Graphs with Maximum Degree Five

Lemma 12.1 In a triangulated graph $G$, the vertices that share an edge with a vertex $v$ of degree $n$ are on a $n$-cycle in $G$.

Proof. Let vertex $v$ be a vertex of degree $n$ in a triangulated graph $G$. Let the set of vertices that share an edge with vertex $v$, its neighborhood, be known as $N(v)$. Choose an $a \in N(v)$. If the edge $\{a, v\}$ does not border two different faces, then $G$ is a tree; thus either $G$ is two connected vertices, with $v$ having degree one and accordingly $a$ being on a one cycle or $G$ is not triangulated. In all other cases, $\{a, v\}$ is on two different faces. Since $v$ is on both these faces, there must be two other edges $\{b, v\}$ and $\{c, v\}$ for $b, c \in N(v)$ with $b, c \neq a$ such that $\{b, v\}$ and $\{a, v\}$ are on the same face and $\{c, v\}$ and $\{a, v\}$ are on the same face. Since every face has three edges and accordingly only three vertices, it must be the case that $\{a, b\}, \{a, c\} \in E(G)$. Thus each $a \in N(v)$ must be on a path consisting only of vertices in $N(v)$, going along edges which are on faces consisting of two vertices neighboring $v$ and $v$.

Suppose that this path does not cover all $n$ vertices in $N(v)$. Let $x$ be one end of the path. It sits on two faces consisting of it, $v$, and a third vertex neighboring $v$. The third vertex in one of these faces shares an edge on the path with $x$. To the contrary, if the third vertex of the other face, which we shall call $y$, is also on the path, then there is a cycle of length less than $n$ surrounding $v$, implying there is a vertex $z$ neighboring $v$ which is not on the cycle. Then $z$ cannot be outside the cycle, since then the edge connecting it and $v$ would cross the cycle and $G$ would not be planar. However, $z$ cannot be within the cycle either, since then it would have to be on a face consisting of itself, $v$, and two vertices on the cycle, which would make $G$ non triangulated. Accordingly, if the path does not cover all of $N(v)$, it can be extended to include $y$. 

36
Thus, the path can be extended to cover all vertices in $N(v)$. At this point though, each end vertex is still connected to another vertex in $N(v)$ by an edge sharing a face with $v$. If the ends are not connected to each other, then it creates a cycle of length less then $n$ out of a segment of the path. This would force a vertex on the remainder of the path into the contradiction of vertex $z$, so we must conclude that the ends of the path are connected in $G$, and accordingly the path of length $n$ can be made into a cycle of length $n$. □

**Theorem 12.2** Let $G$ be a triangulated graph with maximal degree $\Delta(G) \leq 5$. Then $c(G) \leq 2$.

**Proof.** There must exist two paths from the respective cops to the robber, where the two paths approach the robber along different edges incident to him. It will be shown that wherever the robber moves to, these paths can be modified so that they still approach him from different edges (with every face having three edges, it must be the case that every vertex is of degree at least two). Furthermore, this can be done such that the cops can move along these new paths so that the combined distance between the cops and the robber on the two respective paths has decreased or will decrease in the next turn. Accordingly, the cops will steadily get closer and eventually capture the robber.

Let $c_1$ and $c_2$ be the starting vertices of the cops, and $r$ the starting vertex of the robber. Choose directed paths $P_1$ and $P_2$ from $c_1$ to $r$ and $c_2$ to $r$. Let $l_1$ and $l_2$ be their respective lengths, with $l = l_1 + l_2$. The two paths be of the form $P_1 = \{c_1, c'_1, \ldots, a_1, r\}$ and $P_2 = \{c_2, c'_2, \ldots, a_2, r\}$ where $a_1$ and $a_2$ are distinct neighbors of $r$. For this proof, if in the application of these paths a path’s vertices are repeated, just do away with the vertices between the repeats and condense the repetitions of a vertex to just listing it once. For instance, if $c_1 = a_1$ let $P_1 = \{c_1, c'_1, \ldots, a_1, r\} = \{c_1, a_1, r\}$.

Let the cycle of vertices neighboring $r$, whose length is equal to the degree of $r$, be called $H$. Obviously, $a_1$ and $a_2$ are in $H$. If the degree of $r$ is four or less, then for every vertex $x$ in $H$ either $a_1$ is connected to $x$ or $a_2$ is connected to $x$; this is known as $\{a_1, a_2\}$ forming a dominating set of $H$. Similarly, if $r$ had degree five, but $a_1$ and $a_2$ are not neighbors in $H$ then they too will form a dominating set of $H$.

Thus, in these cases, wherever the robber moves to on his turn, he is still neighboring at least one of $a_1$ and $a_2$. Let us say without loss of generality that this is the case for $a_1$. Accordingly after the robber moves $r \rightarrow r'$ (it can be the case that $r = r'$), let $P_1$ and $P_2$ be modified as such:

1. The modified path $P_1'$ will be the same as $P_1$ except final vertex of $P_1$ will be switched from $r$ to $r'$, since $a_1$ is connected to $r'$, and the first cop moves $c_1 \rightarrow c'_1$ so the vertex $c_1$ will be removed...
from the beginning. Thus $P'_1 = \{c'_1, \ldots, a_1, r'\}$.

2. The modified path $P'_2$ will be the same as $P_2$ except from the end point $r$ the edge to $r'$ will be added and the second cop will move $c_2 \to c'_2$, so the vertex $c_2$ will be removed from the beginning. Thus $P'_2 = \{c'_2, \ldots, a_2, r, r'\}$.

These paths still go from both cops’ positions to the robbers position by different incident edges, but $|P'_1| < |P_1|$ and $|P'_2| \leq |P_2|$ (if the robber did not move, $P'_2 < P_2$), so $l' < l$.

However, if $r$ had degree five and $a_1$ is a neighbor of $a_2$ on the five cycle $H$, there is one vertex, let us call it $r'$ on $H$ where neither $a_1$ or $a_2$ neighbor $r'$. Thus, if on the robber’s turn he moves $r \to b$, then the two cops cannot immediately shorten their paths. It turns out however that they can still always shorten their path in two turns. This is true since there must be at least two other vertices in $H$, $a'_1$ and $a'_2$ which neighbor $r'$, and due to the nature of a five cycle, these must each have one of $a_1$ or $a_2$ as a neighbor (let us say that $a'_1$ neighbors $a_1$ and $a'_2$ neighbors $a_2$ without loss of generality). If $J$ is the cycle of vertices neighboring the robbers new location $r'$, then $a'_1, a'_2$ and $r$ are on it. Unless $r'$ has degree three, it cannot be the case that $a'_1$ and $a'_2$ are neighbors in $J$, or else a contradiction of cutting off the remaining vertices as described in the proof of Lemma 12.1 would arise. Accordingly, if we can modify the paths as such:

1. The modified path $P'_1$ will be the same as $P_1$ except the edge $\{a_1, r\}$ is switched to $\{a_1, a'_1\}$, the edge $\{a'_1, r'\}$ is added at the end, and the first cop moves $c_1 \to c'_1$ so the vertex $c_1$ will be removed from the beginning. Thus $P'_1 = \{c'_1, \ldots, a_1, a'_1, r'\}$.

2. The modified path $P'_2$ will be the same as $P_2$ except the edge $\{a_2, r\}$ is switched to $\{a_2, a'_2\}$, the edge $\{a'_1, r'\}$ is added at the end, and the first cop moves $c_2 \to c'_2$ so the vertex $c_2$ will be removed from the beginning. Thus $P'_2 = \{c'_2, \ldots, a_2, a'_2, r'\}$.

After this, the new paths are of the same length as the previous paths and approach $r'$ from different edges. However, either the degree of $r'$ is not five, or $a'_1$ and $a'_2$ are not neighbors, so on the next turn, as described previously, the cops can reduce the lengths of their paths to the robber.

Thus, we can say that by modifying $P_1$ and $P_2$ as described each turn, in the worst case scenario the cops will reduce their combined distance $l$ to the robber every two turns. Since if $l < 2$ it must be the case one of the cops is on top of the robber, the original $l$ must be finite, and the distances are discrete, this means that two cops will always be able to capture a robber in $G$ and the graph is two cop win. \[\square\]
12.2 Triangulated Graphs with Maximum Degree Six

Lemma 12.3 Let $G$ be a maximal linear graph with $\Delta(G) \leq 6$. Let there be no cycle inside of $G$ where every vertex has degree six. Then $c(G) \leq 2$.

Proof. With a graph $G$ such as described, the cops can still basically use the same strategy as with $\Delta(G) \leq 5$ maximal planar graphs, since the robber will either have to stay still or regularly move on to vertices of degree five or less, where the cops can gain on him.

Let $P_1$ and $P_2$ be chosen as described earlier. If the robber is on a vertex of degree five or less, the cops can reduce $l$ as described previously. If the robber is on a vertex of degree six, it will still be the case that paths $P_1$ and $P_2$ will still be set up such that $a_1 \neq a_2$. Let the six cycle of vertices neighboring $r$ be $H$.

If $a_1$ and $a_2$ neighbor each other on this cycle, then for any choice of $r'$, $r'$ will either be a neighbor of one $a_1$ or $a_2$ such that the paths can be shortened, or $r'$ will be separated by one vertex, which shall be called $a'$, from either $a_1$ or $a_2$ on $H$. If, without loss of generality, we say that this is true for $a_1$, then the paths can be modified as such:

1. The modified path $P_1'$ will be the same as $P_1$ except the edge $\{a_1, r\}$ is switched to $\{a_1, a'_1\}$, the edge $\{a'_1, r'\}$ is added at the end, and the first cop moves $c_1 \rightarrow c'_1$ so the vertex $c_1$ will be removed from the beginning. Thus $P_1' = \{c'_1, \ldots, a_1, a'_1, r'\}$.

2. The modified path $P_2'$ will be the same as $P_2$ except the edge $\{r, r'\}$ is added at the end, and the first cop moves $c_2 \rightarrow c'_2$ so the vertex $c_2$ will be removed from the beginning. Thus $P_2' = \{c'_2, \ldots, a_2, r\}$.

Now both paths will have the same length and go to the robber’s new vertex by different incident edges. This can be repeated for as long as the robber is moving to new vertices of degree six, ensuring that at worse $l$ will stay the same.

If $r$ is of degree six but $a_1$ and $a_2$ are not neighbors on $H$, then for any choice of $r'$ in $H$, since $H$ is a six cycle, either $r'$ neighbors $a_1$ or $a_2$, so $l$ can be shortened, or $r'$ is separated from $a_1$ and $a_2$ on $H$ solely by the vertices $a'_1$ and $a'_2$ respectively. Thus, the paths can be modified as follows:

1. The modified path $P_1'$ will be the same as $P_1$ except the edge $\{a_1, r\}$ is switched to $\{a_1, a'_1\}$, the edge $\{a'_1, r'\}$ is added at the end, and the first cop moves $c_1 \rightarrow c'_1$ so the vertex $c_1$ will be removed from the beginning. Thus $P_1' = \{c'_1, \ldots, a_1, a'_1, r'\}$.
2. The modified path $P'_2$ will be the same as $P_2$ except the edge \{a_2, r\} is switched to \{a_2, a'_2\}, the edge \{a'_1, r'\} is added at the end, and the first cop moves $c_2 \rightarrow c'_2$ so the vertex $c_2$ will be removed from the beginning. Thus $P'_2 = \{c'_2, \ldots, a_2, a'_2, r'\}$.

In this case, $l$ stays the same, but if $J$ is the cycle of neighbors of $r'$, then either $r'$ is of degree three, or $a'_1$ and $a'_2$ are not neighbors, since a contradiction of the remaining vertices neighboring $r'$ being cut off as described Lemma 12.1 would occur.

Thus, in the absolute worse case, the cops will neither lose ground or position on the robber while he is on vertices of degree six, and can continue to count on gaining ground on the robber at worst every other time he moves onto a vertex not of degree six. Accordingly at the very worst, every other time that the robber moves to a vertex of degree five or less, the paths can be modified such that $l$ decreases. Since there are no cycles where all the vertices are of degree six, either the robber must leave vertices of degree six on a regular basis, or stay still or retrace his steps on a path where all the vertices are of degree six. If he stays still, both cops move along their paths and $l$ grows shorter. If he backtracks from one vertex of degree six to one he was just on, moving $r' \rightarrow r$, then both cops will still gain on him, due to the fact that the vertices their new paths approach $r'$ from still also neighbor $r$ or are $r$, allowing them to modify their paths so as to immediately shorten $l$. Accordingly, after a finite number of moves, $l < 2$, and the cops win.

□
12.3 A Maximal Planar Graph With Cop Number Three

Theorem 12.4 There exist maximal planar graphs with cop number three.

Proof. The above is a maximal planar graph with maximum degree nine. It is constructed in two parts; the subgraph of black vertices and edges is the graph of the dodecahedron, which is three cop by itself, and the red vertices and edges are additions to this, identically arranged in each pentagon of the dodecahedron, making the graph maximal planar.

The robber has a winning strategy against two cops on this graph. Whenever the robber is on a black vertex, which are all also arranged identically, there is no way the two cops can position themselves so that they are within one edge of all three other black vertices connected to it. Thus, the robber will always be able to move to another black vertex if there are cops in a position to capture him. Since this will be true on the black vertex he moves to, the robber can evade the cops indefinitely, and two cops can not always win. However, since the graph is planar, three cops do have a winning strategy. Thus, this graph is three cop win. □
13 Characterization of $k$-Outerplanar Graphs

An outerplanar graph is one in which the vertices form a cycle that can be embedded in a circle on the plane, with non-intersecting chords. A $k$-outerplanar graph is one in which the vertices form $k$ cycles embedded in $k$ successive circles, each one enclosing all circles before it. $K$-outerplanarity implies that removing the vertices and incident edges of the outermost cycle results in a $(k - 1)$-outerplanar graph. A maximal $k$-outerplanar graph is a graph such that the addition of any edge would cause it to no longer be $k$-outerplanar. This paper contains three results. It shows that any outerplanar graph $G$ satisfies $c(G) \leq 2$. It shows that any maximal outerplanar graph is 1-cop win. And it shows that any maximal 2-outerplanar graph is at most 2-cop win.

13.1 Outerplanar Graphs

**Theorem 13.1** If $G$ is outerplanar, then $c(G) \leq 2$.

**Proof.** All outerplanar graphs can be constructed as series-parallel graphs, and we have already seen, by Theorem 6.1 that series parallel graphs have cop number at most 2. Any two adjacent vertices $u$, $v$ may be labeled the source and the sink. Here, $G$ has 2 or 3 subgraphs (3 if $u$ and $v$ are joined by a chord, 2 if not), one of which is a path of length 1. If a subgraph $H$ has a chord in it, let $u_1 \in H$ be the closest vertex to $u$ that is the endpoint of a chord. Then $H$ can be broken into two subgraphs: the vertices between $u$ and $u_1$ and the vertices between $u_1$ and $v$. We may continue dividing these subgraphs until none of them have chords, at which point they are all composed of only series compositions. Therefore, all outerplanar graphs are series-parallel graphs, and so if $G$ is outerplanar, $c(G) \leq 2$. □

13.2 Maximal 1-Outerplanar Graphs

The proof that all maximal 1-outerplanar graphs are 1-cop win rests on two lemmas:

**Lemma 13.2** There are at least two degree-2 vertices in any maximal 1-outerplanar graph.

**Proof.** When $v(G) = 3$, this is obviously true. This is our base case. Let us assume this holds for any maximal outerplanar graph with up to $n$ vertices. Now consider a maximal outerplanar graph $G$ with $n + 1$ vertices. We note that in a maximal outerplanar graph $G$ with $v(G) \geq 4$, if a vertex has degree 2, its two neighbors must have degree at least 3. We identify a chord in $G$ connecting $u$ and $v$; This chord partitions $G$ into two connected components. We consider one component $G'$ (including $u$ and
where $3 \leq v(G') < n + 1$, and $G'$ is maximal outerplanar. Therefore there are at least two degree-2 vertices in $G'$. If $v(G) = 3$ then $w \notin \{u, v\}$ has degree 2 in the original graph $G$. If $v(G') \geq 4$ and $u$ has degree 2 in $G'$, then $v$ cannot, so $\exists w \notin \{u, v\}, w \in G'$ with degree 2. The same argument holds for the other connected component of $G$ separated by the chord, so $G$ has at least two degree-2 vertices! □

**Lemma 13.3** Let $u$ be a pitfall of $G$ and $G' = G \setminus \{u\}$, the graph obtained by deleting $u$ and its incident edges. Then $c(G') = k$ if and only if $c(G) = k$.

**Proof.** Aigner and Fromme prove this for 1-cop graphs; the argument generalizes to arbitrary $k$. Let $v$ be a dominating vertex of $u$; suppose that $c(G') = k$. The $k$ cops have a winning strategy in $G'$; they can catch $R$ in $G$ by assuming $R$ is on $v$ whenever $R$ is on $u$, so $c(G) \leq k$. $R$ has a strategy for avoiding $k - 1$ cops in $G'$ indefinitely; he can apply this strategy in $G$ by assuming a cop on $u$ is on $v$, so $c(G) > (k - 1)$. Therefore if $c(G') = k$, then $c(G) = k$. Now suppose $c(G) = k$. Any winning strategy for the $k$ cops does not need to involve entering $u$ unless $R$ is on $u$, since positioning a cop on $v$ achieves the same effect. Therefore in $G'$, the $k$ cops will use the same strategy to catch $R$ and so $c(G') \leq k$. Likewise, $R$ has a strategy for avoiding $k - 1$ cops indefinitely. This strategy does not need to involve entering $u$, since the same effect is obtained by entering $v$ (that is, $R$ cannot hide from the cops by moving to $!u$.) Therefore, in $G'$, $R$ will use the same strategy to avoid $k - 1$ cops, so $c(G') > (k - 1)$. Therefore, if $c(G) = k$, $c(G') = k$. □

**Theorem 13.4** If $G$ is maximal outerplanar, then $c(G) = 1$.

**Proof.** We use a proof by induction. When $v(G) = 3$, clearly $c(G) = 1$. Let us assume all maximal 1-outerplanar graphs $G'$ with at most $n$ vertices are 1-cop win. Given a maximal 1-outerplanar graph $G$ such that $v(G) = n + 1$, we use Lemma 13.3 to find a vertex $w$ with degree 2. Since $G$ is maximal outerplanar, $w$’s neighbors $x, y$ must be joined by an edge. Therefore, $\overline{N}(w) = \{x, y, w\} \in \overline{N}(x)$, so $w$ is a pitfall, which we may remove without changing the cop number of the $G$ by Lemma 13.3. Once we delete $w$ and its two incident edges, the resulting graph $G'$ is maximal 1-outerplanar with $n$ vertices, which is 1-cop win by induction. Therefore all maximal outerplanar graphs are 1-cop win. □
13.3 Maximal 2-outerplanar graphs

The octahedral graph is a maximal 2-outerplanar graph with cop number 2, so not all maximal 2-outerplanar graphs are dismantleable. Instead, the cop number for a given maximal 2-outerplanar graph is at most 2.

**Theorem 13.5** If $G$ is maximal 2-outerplanar, then $c(G) \leq 2$.

**Proof.** We start by laying out some key properties of maximal 1-and-2-outerplanar graphs:

- In a maximal 2-outerplanar graph, every vertex on the inner cycle must connect to at least one vertex on the outer cycle.
- Given two adjacent vertices $u$, $v$ in a maximal 1-outerplanar graph, there exists a vertex $w$ such that $d(u, w) = 1 = d(v, w)$.
- If a vertex on the outer cycle is not connected to any vertex on the inner cycle, it is a pitfall.

The algorithm uses the two cops to cut off $R$’s safe neighborhood by controlling paths through the middle of the graph. Begin by placing $C_1$ and $C_2$ at the endpoints of a chord in the inner cycle $u$ and $v$ (if there are no chords, place $C_1$ and $C_2$ on any two vertices on the inner cycle). Let $x$ and $y$ be vertices on the outer cycle connected to $u$ and $v$, respectively. The path $P = \{x, u, v, y\}$ splits the graph $G$ so that $R$ will be placed in a connected component of $G \setminus P$. Let $w$ be the vertex on the inner cycle (in $R$’s connected component) that forms a triangle with $u$, $v$. From the properties above, we are always in one of three scenarios.

![Figure 1: The 3 cases in a maximal 2-outerplanar graph. The blue node indicates $C_1$’s position; the green node indicates $C_2$’s position.](image)

- Case 1 ($(x, x')$ and $(u, x')$ are in $E(G)$): we may automatically update $P$ to be $\{x', u, v, y\}$. $S(R)$ is reduced.
• Case 2 \(((u, w), (x, w), \text{ and } (v, w))\) are in \(E(G)\): we move \(C_1\) to \(w\) and we update \(P\) to be \(\{x, w, v, y\}\). \(S(R)\) is reduced.

• Case 3 \(((u, w) \text{ and } (v, w))\) are chords, \((x, x')\) may or may not be in \(E(G)\): Our movement depends on \(R\).

1. If \(R\) is inside the path \(\{x', w, u, x\}\), \(C_2\) moves to \(w\) and \(C_1\) moves to (or stays on) \(u\). We update \(P\) to be \(\{x', w, u, x\}\) and \(S(R)\) is reduced.
2. If \(R\) is inside the path \(\{x', w, v, y\}\), \(C_1\) moves to \(w\) and \(C_2\) moves to (or stays on) \(v\). We update \(P\) to be \(\{x', w, v, y\}\) and \(S(R)\) is reduced.
3. If \(R\) is on \(x'\) and \((x, x') \in E(G)\), \(C_1\) moves to \(x\). This forces \(R\) to move into one of the first two scenarios and we continue. (If \(R\) moves outside the path \(\{x', w, u, x\}\), \(C_1\) moves to \(x'\) instead of \(w\), then to \(w\).)
4. If \(R\) is on \(x'\) and \((x, x') \notin E(G)\), \(C_1\) moves to \(w\). This forces \(R\) to move into one of the first two scenarios and we continue.

This process can be repeated until \(C_1\) and \(C_2\) are (again) on adjacent vertices \(u', v'\) on the inner cycle. At this point, \(S(R) = \emptyset\) since all vertices on the outer cycle must be connected to the inner cycle.

13.4 Maximal 3-Outerplanar Graphs

**Theorem 13.6** If \(G\) is maximal 3-outterplanar, then \(c(G) \leq 2\).

**Proof.** We use a similar strategy to our proof of Theorem 13.5. First, we partition \(G\) with two connected paths that radiate from the innermost cycle to the outermost. Let \(a\) and \(x\) be adjacent vertices on the innermost cycle (hereafter referred to as the 1st cycle). Let \(b\) and \(y\) be vertices on the 2nd cycle adjacent to \(a\) and \(x\), respectively. Let \(c\) and \(z\) be vertices on the 3rd cycle connected to \(b\) and \(y\), respectively. Place \(C_1\) on \(b\) and \(C_2\) on \(y\); now the cops control a two connected paths \(P_1 = \{a, b, c\}\) and \(P_2 = \{x, y, z\}\) that partition \(G\) into two connected components:

When \(R\) is placed in one of these components, we may ignore the other. Our goal is to update the paths and reduce \(S(R)\) while ensuring that \(R\) cannot slip past the two paths. Let \(w\) be a vertex on the 1st cycle (in \(R\)'s connected component) that forms a triangle with \(x, a\). For either cop, the strategy to update its path depends on the configuration of vertices it is in. There are 5 such configurations:
Figure 2: The paths controlled by the two cops. The blue node indicates $C_1$’s position; the green node indicates $C_2$’s position.

Figure 3: The 5 scenarios for the cops. The blue node indicates the position of $C_1$; the green node indicates the position of $C_2$.

The first 3 configurations have unconditional strategies:

- Case 1 ($(c, c')$ and $(b, c')$ are in $E(G)$): we update $P_1$ from $\{a, b, c\}$ to $\{a, b, c'\}$.
- Case 2 ($(a, w), (b, w)$, and $(x, w)$ are in $E(G)$): we update $P_1$ to $\{w, b, c\}$.
- Case 3 ($(a, b'), (b, b')$, and $(c, b')$ are in $E(G)$): we move $C_1$ to $b'$ and update $P_1$ to $\{a, b', c\}$.

Case 4 ($(a, w)$ and $(x, w)$ are chords, $(b, b')$ is in $E(G)$) depends on the location of $R$:

- If $R$ is not on $w$ or inside the square $\{a, b, b'w\}$ then we move $C_1$ to $b'$ and update $P_1$ to $\{w, b', c\}$.
- If $R$ is inside the square $\{a, b, b'w\}$, we move $C_2$ from $y$ to $x$ (threatening $w$), then to $w$, then to $b'$ and update $P_2$ to $\{w, b', c\}$.
If \( R \) is on \( w \), we may assume \((b, w) \notin E(G)\) (or we may capture \( R \)). We move \( C_2 \) to \( x \) (threatening \( w \)), which forces \( R \) into one of the two previous scenarios.

Case 5 \(((b, b')\) is a chord in the middle cycle, \((c, c')\) is in \( E(G)\)) requires a separate proof if \( R \) is on \( c' \) or inside the square \( \{b, c, c', b'\} \). If \( R \) is not on \( c' \) or inside the square, we move \( C_1 \) to \( b' \) and update \( P_1 \) to \( \{a, b', c'\} \).

### 13.5 Case 5

Our goal here is to show that if \( R \) is on \( c' \) or inside the square \( \{b, c, c', b'\} \), the two cops may always chase \( R \) off \( c' \) or trap \( R \) in the square. We want to use \( C_2 \) to threaten \( c' \) while making still protecting \( P_2 \). If \( d(x, c') \geq d(y, c') \) or \( d(z, c') \geq d(y, c') \), we know \( C_2 \) may safely threaten \( c' \) while protecting \( P_2 \). Suppose \( d(x, c') \geq d(y, c') = k \) and \( d(z, c') < d(y, c') \): then we move \( C_2 \) to \( z \), then along a shortest path from \( z \) to \( c' \), stopping when \( C_2 \) threatens \( c' \). Now \( R \) must move either inside the square \( \{b, c, c', b'\} \) or away from \( c' \); in the former case, we move \( C_2 \) to \( c' \), then to \( b' \) and update \( P_2 \) to \( \{a, b', c'\} \).

In the latter case, we return \( C_2 \) to \( z \) along the shortest path, and then move \( C_2 \) to \( y \), while we move \( C_1 \) to \( b' \) and update \( P_1 \) to \( \{a, b', c'\} \). \( R \) will not be able to reach any vertices in \( P_2 \) before \( C_2 \) can protect the!

We supposed that we have updated \( P_2 \) as much as possible. We know that \( d(y, c') \leq 4 \) since we may move \( C_2 \) from \( y \) to \( x \), then \( a \), then \( b' \), then \( c \). We also know that, since vertices on non-adjacent cycles must themselves be nonadjacent, \( d(x, c') \geq 2 \).

- If \( d(x, c') = 2 \), \( d(y, c') \leq 2 \) and we are done (see diagram for reference). So we suppose that \( d(x, c') \geq 3 \).
- If \( d(x, c') \geq 4 \), we are done since \( d(y, c') \leq 4 \). So we suppose that \( d(x, c') = 3 \), and that \( d(y, c') = 4 \).
- If \( d(z, c') \geq 4 \), we are done (see above). We supposed that \( d(y, c') = 4 \), so \( d(z, c') \geq 3 \); we suppose that \( d(z, c') = 3 \).
- Supposing that \( d(x, c') = 3 = d(z, c') \) and \( d(y, c') = 4 \), we move the cops accordingly:
  - We move \( C_2 \) to \( x \). If \( R \) moves outside \( \{b, c, c', b'\} \), we move \( C_1 \) to \( b' \) and update \( P_1 \) to \( \{a, b', c'\} \) and move \( C_2 \) back to \( y \) without updating \( P_2 \). OTHERWISE,
We move $C_2$ to $a$ (threatening $b'$) and move $C_1$ to $c$. If $R$ moves outside $\{b, c, c', b'\}$, we move $C_1$ to $c'$ and $C_2$ back to $x$, then move $C_1$ to $b'$ and update $P_1$ to $\{a, b', c'\}$ and move $C_2$ back to $y$ without updating $P_2$. If $R$ moves inside $\{b, c, c', b'\}$, we move $C_2$ to $b'$ and update $P_2$ to $\{a, b', c'\}$.

Thus the two paths $P_1$ and $P_2$ will be updated until $R$ is trapped in a square between adjacent paths (see diagram). At this stage, we may treat the game as maximal 2-outerplanar and use the strategy for that case.

13.6 $k$-outerplanar graphs

We can not definitively characterize $k$-outerplanar graphs for $k \geq 3$; the dodecahedron is a 3-outerplanar graph with cop number 3, the cuboctahedron is a 3-outerplanar graph with cop number 2, and we can construct a rather unwieldy 3-outerplanar graph with cop number 1:

![Diagram](https://via.placeholder.com/150)

However, for $k \leq 2$, it appears that we cannot construct a $k$-outerplanar graph with cop number 3. The proof for 1-outerplanar graphs is done; it remains to show that 2-outerplanar graphs have cop number $\leq 2$.

**Conjecture 13.1** Given a 2-outerplanar graph $G$, $c(G) \leq 2$.

[I have yet to prove this either way.]

Remaining questions: What about maximal $k$-outerplanar graphs?
14 The Minimum Size of Graphs with Cop Number 3

This paper determines the minimum number of vertices required for a 3-cop win graph. The Petersen graph gives us an upper bound of 10 vertices; we show that if $v(G) \leq 9$, then $c(G) \leq 2$.

14.1 Notation

Let

$$V_k = \{v \in V(G) \mid \deg(v) = k\}.$$

We will use this notation most often when talking about the number of vertices in a graph with degree 3. Let

$$N(u) = \{v \in V(G) \mid (u, v) \in E(G)\},$$

and let

$$\overline{N}(u) = N(u) \cup \{u\}.$$

This paper occasionally refers to $\overline{N}(C_i)$, and should be taken to mean the closed neighborhood of the vertex that $C_i$ is occupying. Given a set of vertices, $K$, let

$$N(K) = \{v \in V(G) \mid (v, k) \in E(G), k \in K\},$$

and let

$$\overline{N}(K) = \{N(K) \cup K\}.$$

We use this notation in Lemma 3.2.

$S(R)$ is defined, after $R$’s move, as the connected component of $V(G) \setminus \{\overline{N}(C_1) \cup \overline{N}(C_2)\}$ containing $R$. If $R$ moves to a vertex in $\{\overline{N}(C_1) \cup \overline{N}(C_2)\}$, then a cop will catch him on the next turn. Therefore, if $S(R) = \emptyset$ after either player’s turn, the game is over.
14.2 Using the degree to determine a lower bound

First, we may assume that there are no dominated vertices in the graphs we are considering: we can eliminate dominated vertices without changing the cop number. This means that \( V_1 = \emptyset \) for all the graphs we consider. Even when we do not consider graphs with dominated vertices, there are too many graphs of size \( \leq 9 \) to give strategies for two cops to each, so we turn instead to conditions under which 2 cops may always catch a robber. An important boundary condition is on the maximum degree of any vertex.

**Lemma 14.1** For a given graph \( G \) such that \( v(G) = n \), if \( \Delta(G) \geq n - 5 \), then \( c(G) \leq 2 \).

**Proof.** Suppose \( \exists u \in V(G) \) such that \( \text{deg}(u) \geq n - 5 \). This implies that \( |\overline{N}(u)| \geq n - 4 \). If we place \( C_1 \) on \( u \), then \( R \) has at most 4 possible starting locations \( R^0 \) before we place \( C_2 \). If \( |V(G) \setminus \overline{N}(u)| < 4 \), we know that we can keep \( C_1 \) on \( u \) while \( C_2 \) moves to these vertices and chases \( R \) into \( \overline{N}(C)1 \). If \( |V(G) \setminus \overline{N}(u)| = 4 \), let \( \{V(G) \setminus \overline{N}(u)\} \). If the graph induced by \( s_1, \ldots, s_4 \) is disconnected or 1-cop win, the strategy is clear - position \( C_1 \) on \( u \) and \( C_2 \) on one of the vertices outside \( \overline{N}(u) \). Once \( R \) is placed, have \( C_2 \) use a 1-cop strategy on \( S(R) \) until \( R \) is caught or enters \( \overline{N}(C_1) \). Thus the only troublesome case is when the four \( s_i \) are arranged in a square. Even in this case, if one of these vertices \( s_k \) does not connect to \( \overline{N}(u) \) in at least two places, we may place \( C_2 \) such that \( R \) must start on \( s_k \), then move \( C_1 \) along a shortest path to \( s_k \). If each vertex connects to \( \overline{N}(u) \) in at least 2 places, we examine the graph for two special circumstances:

![Figure 4: Two special circumstances regarding the \( s_i \).](image)

In the first case, we have a dominating set. Place \( C_1 \) at \( u \) and \( C_2 \) at \( v \) and \( S(R) = \emptyset \). In the second case, place \( C_1 \) at \( u \) and \( C_2 \) at \( v \), leaving only \( s_4 \) for \( R \) to occupy safely. On the cops’ first turn, move \( C_2 \) to \( w \); now \( R \) is trapped. If neither special circumstance exists, we proceed as follows:

1. Place \( C_1 \) at \( u \) and \( C_2 \) at \( s_1 \). This forces \( R \) to be placed at \( s_3 \).
2. Move $C_1$ to $w \in N(u)$, where $w$ connects to $s_3$. This forces $R$ to a vertex $v \in N(u)$. Note that if $v$ is connected to $s_1$, $C_2$ will capture $R$.

3. The next step depends on $v$. We know $v$ connects to $s_3$ and does not connect to $s_1$; our actions are determined by the number of $s_i$ connected to $v$.

   (a) If $v$ is connected to 1 or 2 of the $s_i$, then all these vertices can be guarded by $C_2$. We move $C_2$ so that the $s_i$ in $N(v)$ are also in $N(C_2)$. We move $C_1$ back to $u$, and $R$ will be trapped, since it can not move back to the safe square.

   (b) If $v$ is connected to $s_2, s_3$, and $s_4$, we move $C_1$ to $s_3$ and $C_2$ into $N(u)$. Now $R$ must move but cannot return to the square this turn; we say he moves to $z \in N(u)$. $z$ may connect to some $s_i$, but it cannot connect to $s_1, s_2, s_4$ or to all the $s_i$ (these are the special circumstances we check for at the beginning.) On the cops’ next move, we are assured that $C_1$ can move from $s_3$ to another $s_i$ such that if $s_k \in N(z)$, then $s_k \in N(C_1)$. $C_1$ guards the $s_i$ in the neighborhood of $z$ and $C_2$ moves to $u$. $R$ cannot move back to the $s_i$ because of $C_1$, and cannot stay in $N(u)$ because of $C_2$. So regardless of $R$’s move, $S(R) = \emptyset$ and we are done.

**Theorem 14.2** Given a graph $G$ s.t. $v(G) \leq 9$, $c(G) \leq 2$.

If $v(G) \leq 8$, we know that if $\Delta(G) \geq 3$, we are done by Lemma 14.1. If $\Delta(G) \leq 2$ then we know $G$ is either a cycle or a tree. If $G$ is a cycle, $c(G) = 2$. If $G$ is a tree, $c(G) = 1$.

To show that 9-vertex graphs are at most 2-cop win, we use the following lemma.

**Lemma 14.3** Given a graph $G$, if 2 cops can always move such that, after $R$’s move, $\Delta(S(R)) \leq 3$ and $S(R)$ contains at most two nonadjacent vertices of degree 3, then $c(G) \leq 2$.

**Proof.** For all cases, we suppose that $C_1$ and $C_2$ have moved to force the conditions above.

If $S(R)$ contains no vertices of degree 3, $S(R)$ will be a path; the two endpoints of the path will connect to vertices in the cops’ neighborhoods:

![Diagram of N(C1) U N(C2)]

WLOG, assume $N(S(R)) \cap N(C_1) \neq \emptyset$. If both endpoints of $S(R)$ are connected to vertices in $N(C_1)$, then $C_1$ holds still while $C_2$ moves to the closest endpoint of $S(R)$. $C_2$ then moves along
Figure 5: the red vertex indicates degree 3.

Figure 6: The two cases where $S(R) \cap V_3 = 2$. Red vertices are degree-3; others are degree 2.

the original $S(R)$; by the time it reaches the other endpoint, $R$ must have been caught. If only one end-vertex of $S(R)$ connects to $\overline{N}(C_1)$, then $C_1$ holds still while $C_2$ moves to the other endpoint of $S(R)$ (which must connect to a vertex in $\overline{N}(C_2)$). $C_2$ then moves through the original $S(R)$; by the time it reaches the other end-vertex, $R$ must have been caught.

If $S(R)$ contains 1 degree-3 vertex $u$, $S(R)$ will have 3 paths connected at $u$; the other endpoints of these paths will connect to vertices in the cops’ neighborhoods:

In both of these cases, there are 3 edges connecting $S(R)$ to the cops’ neighborhoods. WLOG, assume $|N(S(R)) \cap N(C_1)| \geq 2$ (the pigeonhole principle assures us this is the case for at least one cop). If $|N(S(R)) \cap N(C_1)| = 3$, $C_1$ holds still while $C_2$ moves to the closest endpoint of $S(R)$; $C_2$ then moves from that endpoint to $u$. Now $S(R)$ contains no degree-3 vertices and we have reduced the problem to a previous one. If $|N(S(R)) \cap N(C_1)| = 2$, $C_1$ holds still while $C_2$ moves through the third endpoint to $u$; again we have reduced the problem to a previous one.

If $S(R)$ contains 2 non-adjacent degree-3 vertices $u_1$ and $u_2$, $S(R)$ will look like one of the following:

In case A) there are only two endpoints connected to vertices in the cops’ neighborhoods, and we may proceed as above. WLOG, assume that $|N(S(R)) \cap N(C_1)| \geq 1$; $C_1$ stays still while $C_2$ moves to the other endpoint, then up to a degree-3 vertex. At this stage, $S(R)$ will be reduced to one degree-3 vertex and have reduced the problem to one already solved.

52
In case B), there are 4 endpoints connected to the cops’ neighborhoods. WLOG, assume $|\overline{N}(S(R)) \cap \overline{N}(C_1)| \geq 2$ (the pigeonhole principle assures us this for at least one cop). If $|\overline{N}(S(R)) \cap \overline{N}(C_1)| = 4$, $C_1$ stays still while $C_2$ moves to the nearest endpoint, then to either $u_1$ or $u_2$. Now $S(R)$ contains at most one degree-3 vertex and we follow the strategies above. If $|\overline{N}(S(R)) \cap \overline{N}(C_1)| = 3$, $C_1$ again holds still and $C_2$ moves to $u_1$ or $u_2$ via the remaining endpoint.

When $C_2$ reaches $u_1$ or $u_2$, $S(R)$ is reduced to at most one degree-3 vertex and we follow the strategies above. If $|\overline{N}(S(R)) \cap \overline{N}(C_1)| = 2$ (and therefore $|\overline{N}(S(R)) \cap \overline{N}(C_2)| = 2$), $C_1$ and $C_2$ follow the algorithm below, which diverts at the end based on the configuration of the paths.

1. Of the 4 paths connecting $u_1$ and $u_2$ to the cops’ starting points, $C_1^0$ and $C_2^0$, at least one is the shortest. Assume, WLOG, that it is a path from $C_1^0$ to $u_1$. $C_1$ controls that path. $C_2$ controls the shortest path from $C_2^0$ to $u_2$.

2. If $R$ is on the path between $u_1$ and $u_2$, or on a path marked by one of the cops (not including $u_1$ and $u_2$), both cops advance along their marked paths.

3. If $R$ is on a path between $u_j$ and $C_i^0$ (including $u_j$ and $C_i^0$) that was not marked, AND $d(R, C_i^0) - d(C_i, C_i^0) \geq 3$, both cops advance along their marked paths.

4. If $R$ is on a path between $u_j$ and $C_i^0$ (including $u_j$ and $C_i^0$) that was not marked, AND $d(R, C_i^0) - d(C_i, C_i^0) = 2$, $C_i$ does not move and the other cop advances along his marked path. (This ensures that $R$ cannot move past either of the cops toward $C_i^0$.)

5. If $R$ is on a path between $u_j$ and $C_i^0$ (including $u_j$ and $C_i^0$) that was not marked, AND $d(R, C_i^0) - d(C_i, C_i^0) \leq 1$, $C_i$ moves back toward $C_i^0$ and the other cop advances along his marked path. (This ensures that $C_i$ will be able to protect $\overline{N}(C_i^0)$ if $R$ moves toward it.)
Using this method, at least one cop will reach its goal vertex. We assume WLOG that $C_1$ does. At this point, the strategy depends on the case.

**Endgame scenario for case $B_1$)**

$C_1$ is on $u_1$; $C_2$ is somewhere on the path between $C_0^0$ and $u_2$.

- if $R$ is on a path from $u_1$ to $C_0^0$, $C_1$ can keep him on this path while $C_2$ moves to $u_1$. When $C_2$ reaches $u_1$, $C_1$ moves to $C_1^0$. When $C_1$ reaches $C_0^0$, $S(R)$ contains no degree-3 vertices and the problem has been reduced.
- if $R$ is not on the path from $u_1$ to $C_0^0$, $C_1$ moves from $u_1$ to $u_2$ while $C_2$ moves back to $C_0^0$. This forces $R$ into a path from $C_0^0$ to $u_2$. When $C_2$ reaches $C_0^0$ and $C_1$ reaches $u_2$, $S(R)$ will contain no degree-3 vertices and the problem has been reduced.

**Endgame scenario for case $B_2$)**

$C_1$ is on $u_1$; $C_2$ is somewhere on the path between $C_0^0$ and $u_2$.

- If $R$ is on the path from $u_1$ to $C_0^0$, $C_2$ returns to $C_0^0$. Now, $S(R)$ contains no degree-3 vertices and the problem has been reduced.
- If $R$ is not on the path from $u_1$ to $C_2^0$, $C_1$ can keep him off it permanently. $C_2$ continues the original strategy until it reaches $u_2$. At this point:
  - If $R$ is on the path between $u_1$ and $u_2$, $S(R)$ contains no degree-3 vertices and the problem has been reduced.
  - If $R$ is on the path from $u_2$ to $C_0^0$, $C_1$ moves back to $C_1^0$. $S(R)$ contains no degree-3 vertices and the problem is reduced.

This shows that if 2 cops can always move such that $\Delta(S(R)) \leq 3$ and $S(R)$ contains at most two nonadjacent vertices of degree 3, then $c(G) \leq 2$. Now we can prove the main theorem.

**Proof of Theorem [14.2]**

**Note:** for the purposes of this section, a graph $H$ is a 9-vertex graph with $2 \leq \delta(H) \leq \Delta(H) \leq 3$, and with no dominated vertices.

Since the vertices in a graph $H$ will have degree either 2 or 3, Euler’s handshaking lemma tells us that $|V_3|$ is even. So we may classify the graphs into five cases.
Figure 8: red vertices are degree-3; others are degree 2.

- $|V_3| = 0$. Since $2 \leq \delta(H) \leq \Delta(H) \leq 2$, $H$ must be the cycle $C_9$, which is 2-cop win.

- $|V_3| = 2$. We place the two cops on $u, v \in V_3$. Now $S(R)$ cannot contain any vertices of degree 3 and we are done by Lemma 14.3.

- $|V_3| = 4$. In this scenario, we may have adjacent degree-3 vertices. If we do, we place $C_1$ on one of the adjacent degree 3 vertices; this ensures $V_3 \setminus \overline{N}(C_1) \leq 2$. We place $C_2$ on another and $|S(R) \cap V_3| \leq 1$ and we are done by Lemma 14.3. If none of the vertices in $V_3$ are adjacent, place the cops on any two degree-3 vertices and again we are done by Lemma 14.3.

- $|V_3| = 6$. We claim that $\exists u \in V_3$ such that $\overline{N}(u) \cap V_3 \geq 3$. Indeed, suppose $\forall u \in V_3$, $\overline{N}(u) \cap V_3 \leq 2$. Note that since $|V_3| = 6$, we have $|V_2| = 3$. Now consider $u \in V_3$ in this situation:

  Note that $v$ cannot connect to $w$ or $x$ without dominating them. Since $|V_2| = 3$, one of $v$’s two other edges must connect to a vertex of degree 3. However, this means that $\overline{N}(v) \cap V_3 \geq 3$. This is a contradiction.

  Given this, we place $C_1$ on $u \in V_3$ such that $\overline{N}(u) \cap V_3 \geq 3$. Thus, before we place $C_2$, we know $|S(R) \cap V_3| \leq 3$. If two or more of the degree-3 vertices are adjacent, we place $C_2$ on one of them; this ensures $S(R) \cap V_3 \leq 1$ and we are done by Lemma 14.3. If no two degree-3 vertices in $S(R)$ are adjacent, we place $C_2$ on any degree-3 vertex in $S(R)$; now $|S(R) \cap V_3| \leq 2$ and $S(R)$ does not contain two adjacent vertices of degree 3, so we are again done by Lemma 14.3.

- $|V_3| = 8$. We can construct 9-vertex graphs with 8 degree-3 vertices by starting with an 8-vertex, 3-regular graphs and subdividing an edge. There are only 5 such graphs to consider:

  Of the five cases, the first three have diameter 3; subdividing an edge cannot shrink the diameter of the graph, so we may always place the cops on 2 vertices of degree 3 with disjoint neighborhoods. Each has a neighborhood of size 4, and $|H| = 9$. This implies $|S(R)| = |V(H) \setminus \{\overline{N}(C_1) \cup \overline{N}(C_2)\}| = 1$, so we are done by Lemma 14.3. In case 4,
given an edge $e$, \( \exists u, v \in V(G) \) such that $e$ is on the shortest path of length 2 between $u$ and $v$. Thus, subdividing any edge brings the diameter of the graph to 3 and allows us to place $C_1$ and $C_2$ on $u$ and $v$, which are now disjoint vertices of degree 3. This means we are done by Lemma 14.3. In case 5, subdividing an external edge allows us to place $C_1$ and $C_2$ on disjoint vertices of degree 3; subdividing a chord does not. However, if we subdivide an internal edge (WLOG, assume it is $(a, b)$), by placing the cops at $c$ and $d$ we ensure $S(R)$ does not contain two adjacent vertices of degree 3, and we are done by Lemma 14.3.

### 14.3 Summary

By showing that any 9-vertex graph $G$ with $\Delta(G) \leq 3$ is a 2-cop graph, and also showing that any graph with a vertex of degree $\geq n - 5$ is a 2-cop graph, we see that all 9-vertex graphs are 2-cop graphs. Thus the smallest number of vertices for a 3-cop graph is 10. A possible extension may be to examine the smallest possible 4-cop graph.
15 Conjectures and Suggestions For Further Work

15.1 Maximal Planar Graphs

We have proved that maximal planar graphs with maximum degree five or less are always two cop, and we have also found maximal planar graphs with maximum degree seven who are three cop, though only one with maximum degree nine is shown in this paper. This leaves maximal planar graphs with maximum degree six. In this paper we have shown that without a cycle of vertices all of degree six such a graph is two cop, and furthermore it can be shown that unless there exists a cycle of two very specific types, then the graph can still be shown two cop by the same algorithm as described. With such a cycle on a planar graph, two of the edges going into each vertex on the cycle will be part of the cycle, and the other four must be connected to vertices on one side of the cycle or the other, since a cycle on a planar graph partitions the vertices. The two types of cycles required for a graph such as described to possibly not be two cop are ones where every vertex is connected to two vertices on one side of the cycle and two on the other, or one where the vertices are strictly alternating between being connected to three on one side and one on the other and one on one side and three on the other. Having investigated these two conditions, it is our belief they too can be shown to be two cop, but this is impossible by the method we have described.

15.2 Maximal $k$-Outerplanar Graphs

Though we have made some headway into this problem, proving that if $k/\leq 3$ then the graph is two cop, we have not been either able to find a three cop maximal outerplanar graph or prove that the entire class is two cop. Our suspicion though is that since there are maximal planar three cop graphs, there will also be maximal outerplanar three cop graphs. The question is if this is true, and if so what is the largest $k$ such that all maximal $k$-outerplanar graphs are two cop.

15.3 2-Outerplanar Graphs

Three cop win 3-outerplanar graphs exist, but we have yet to find any 2-outerplanar three cop win graphs. We suspect that it can be shown that none exist, but were unable to do so.
15.4 Extremal Results

We conjecture that the smallest four cop win graph has nineteen vertices, and that the smallest three cop win planar graph is the dodecahedron, with twenty vertices.
A Mathematica Implementation of the Marking Algorithms

A.1 One Cop Algorithm Implementation

Algorithm Mark Full Visibility One Cop

- IsCopWin takes in vertices and edges
- IsCopWinFromName takes in the name of a graph known to Mathematica, residing in its Graph-Data dictionary.

\[
\text{IsCopWinFromName}[\text{graphName}_] := \\
\text{Module}[[\text{name} = \text{graphName}, \text{vertices}, \text{edges}], \\
\text{edges} = \text{GraphData}[\text{name}, \text{"EdgeIndices"}]; \\
\text{vertices} = \text{DeleteDuplicates}[\text{Flatten}[\text{edges}]]; \\
\text{IsCopWin}[\text{vertices}, \text{edges}] \\
]\]

\[
\text{IsCopWin}[\text{vertices}_, \text{edges}_] := \\
\text{Module}[[\text{vert} = \text{vertices}, \text{edge} = \text{edges}, \text{marked}, \text{unmarked}, \text{pairs}, \\
\text{isDominated}, \text{prevUnmarkedCount}, \text{i}, \text{retVal}], \\
\text{pairs} = \text{Tuples}[\text{vert}, 2]; \\
\text{marked} = \text{Select}[\text{pairs}, # == \text{Reverse}[#] \&]; \\
\text{unmarked} = \text{Complement}[\text{pairs}, \text{marked}]; \\
\text{isDominated} = \text{False}; \\
\text{prevUnmarkedCount} = \text{Length}[\text{pairs}]; \\
\text{While}[\text{prevUnmarkedCount} > \text{Length}[\text{unmarked}], \\
\text{prevUnmarkedCount} = \text{Length}[\text{unmarked}]; \\
\text{For}[\text{i} = \text{Length}[\text{unmarked}], \text{i} >= 1, \text{i}--, \\
\text{If}[\text{hasNoEscape}[\text{unmarked}[[\text{i}]], \text{marked}, \text{vert}, \text{edge}], \\
\text{If}[\text{DEBUG}, \\
]
Print["<<<<<<<<<<< Marking ", unmarked[[i]], "<<<<<<<<<<<<<<"];
marked = Append[marked, unmarked[[i]]];
unmarked = Delete[unmarked, i];
]
]
]
retVal = Evaluate[Length[unmarked] == 0];
retVal
]

hasNoEscape[configuration_, markedList_, vertices_, edges_] :=
Module[{config = configuration, marked = markedList, 
vert = vertices, edge = edges, robber, cop, robberNeighbors, 
copNeighbors, noEscape, i, j, robberN, copN, zzz },

robber = config[[2]]; 
cop = config[[1]]; 
robberNeighbors = getNeighbors[robber, edge]; 
copNeighbors = getNeighbors[cop, edge]; 
noEscape = True;
i = 1;

While [noEscape && i <= Length[robberNeighbors],

robberN = robberNeighbors[[i]]; 
j = 1;

If[DEBUG, Print[" robberN=", robberN]]; 
zzz = False;

60
For[j = 1, j <= Length[copNeighbors], j++,
    copN = copNeighbors[[j]];
    zzz = Evaluate[zzz || MemberQ[marked, {copN, robberN}]];
]

zzz;

noEscape = zzz;

    i = i + 1;
]

noEscape
]

copNeighbors[vertex_, edges_] :=
    DeleteDuplicates[Flatten[Select[edges, MemberQ[#, vertex] &]]]
A.2 Two Cop Marking Algorithm

Algorithm Mark Full Visibility Two Cops

- IsTwoCopWin takes in vertices and edges
- IsTwoCopWinFromName takes in the name of a graph known to Mathematica, residing in its GraphData dictionary.
- IsTwoCopWinFromPolyhedronName takes in the name of a polyhedron known to Mathematica, residing in its GraphData dictionary.

<< Combinatorica';

IsTwoCopWinFromPolyhedronName[polyName_] :=

Module[{name = polyName, vertices, edges},
  edges = PolyhedronData[name, "EdgeIndices"];  
  vertices = DeleteDuplicates[Flatten[edges]];

  Print["++++++++++++", name, "++++++++++++"];

  If[DEBUG,
    Print["vertices=", vertices];
    Print["edges=", edges];
    Print["++++++++++++++++++++++++++++++++++++++++++++"];
  
  IsTwoCopWinParallel[vertices, edges]
]

IsTwoCopWinFromName[graphName_] :=

Module[{name = graphName, vertices, edges},
  edges = GraphData[name, "EdgeIndices"];  
  vertices = DeleteDuplicates[Flatten[edges]];

  Print["+++++++++++++++++++++++++++++++", name, "+++++++++++++++++++++++++++++++"];

  If[DEBUG,
    Print["vertices=", vertices];
    Print["edges=", edges];
    Print["++++++++++++++++++++++++++++++++++++++++++++++++++"];
  
  IsTwoCopWinParallel[vertices, edges]
]
IsTwoCopWin[vertices, edges]
}

IsTwoCopWin[vertices_, edges_] :=

Module[{vert = vertices, edge = edges, marked, unmarked, copConfigs, configs, isDominated, prevUnmarkedCount, i, retVal},
copConfigs = Join[Subsets[vert, {2}], Table[{i, i}, {i, 1, Length[vert]}]];
cConfigs = Flatten[Table[{copConfigs[[i]], j}, {i, 1, Length[copConfigs]}, {j, 1, Length[vert]}], 1];
marked = Select[configs, #[[1]][[1]] == #[[2]] || #[[1]][[2]] == #[[2]] &];
unmarked = Complement[configs, marked];
isDominated = False;
prevUnmarkedCount = Length[configs];

While[prevUnmarkedCount > Length[unmarked],

  prevUnmarkedCount = Length[unmarked];

  Print["+++++++++++++++++++++Number Unmarked is ", prevUnmarkedCount, 
         "+++++++++++++++++++++"];

  For[i = Length[unmarked], i >= 1, i--,
     If[hasNoEscape[unmarked[[i]], marked, vert, edge],

       marked = Append[marked, unmarked[[i]]];
       unmarked = Delete[unmarked, i]
     ]]
}
hasNoEscape[configuration_, markedList_, vertices_, edges_] :=
    Module[{config = configuration, marked = markedList, vert = vertices,
        edge = edges, robber, cop1, cop2, robberNeighbors, cop1Neighbors,
        cop2Neighbors, noEscape, i, j, k, robberN, cop1N, cop2N, zzz,
        markedRobberN},
        robber = config[[2]];  
cop1 = config[[1]][[1]];  
cop2 = config[[1]][[2]];  
robberNeighbors = getNeighbors[robber, edge];  
cop1Neighbors = getNeighbors[cop1, edge];  
cop2Neighbors = getNeighbors[cop2, edge];  
noEscape = True;  
i = 1;  

While[noEscape && i <= Length[robberNeighbors],  
    robberN = robberNeighbors[[i]];  
    markedRobberN = Select[marked, #[[2]] == robberN &];  
    k = 1;  

    Print[" markedRobberN=", markedRobberN];  
    zzz = False;  
    While[Not[zzz] && k <= Length[cop1Neighbors],
cop1N = cop1Neighbors[[k]];  
j = 1;  
While[Not[zzz] && j <= Length[cop2Neighbors],  
cop2N = cop2Neighbors[[j]];  
zzz = If[cop1N <= cop2N,  
MemberQ[markedRobberN, {{cop1N, cop2N}, robberN}],  
MemberQ[markedRobberN, {{cop2N, cop1N}, robberN}]];  
j = j + 1;  
k = k + 1]  
(*I’m not sure why, but the next line fixes the assignment error*)  
zzz;  
noEscape = zzz;  
i = i + 1;]

getNeighbors[vertex_, edges_] :=

DeleteDuplicates[Flatten[Select[edges, MemberQ[#, vertex] &]]]

IsTwoCopWinParallel[vertices_, edges_] :=

Module[{vert = vertices, edge = edges, marked, unmarked, copConfigs, configs, isDominated, prevUnmarkedCount, i, retVal, noEscapeList},
copConfigs = 
Join[Subsets[vert, {2}], Table[{i, i}, {i, 1, Length[vert]}]];  
copConfigs = 
Flatten[Table[{copConfigs[[i]], j}, {i, 1, Length[copConfigs]}, {j, 1, Length[vert]}], 1];

marked = 
Select[configs, #[[1]][[1]] == #[[2]] || #[[1]][[2]] == #[[2]] &];
unmarked = Complement[configs, marked];

isDominated = False;

prevUnmarkedCount = Length[configs];

While[prevUnmarkedCount > Length[unmarked],

prevUnmarkedCount = Length[unmarked];

Print["+++++++++++++++Number Unmarked is ", prevUnmarkedCount,
"+++++++++++++++++++++"];

(*noEscapeList = ParallelMap[hasNoEscape[#, marked, vert, edge] &, unmarked]*)

noEscapeList =
Parallelize[Map[hasNoEscape[#, marked, vert, edge] &, unmarked]];

For[i = Length[unmarked], i >= 1, i--,
If[noEscapeList[[i]],

marked = Append[marked, unmarked[[i]]];
unmarked = Delete[unmarked, i];
];
];

retVal = Evaluate[Length[unmarked] == 0];
retVal]
References


