Graphs, Games, and Decisions:

How Does a Winner *Slither*?

Parikshit Chalise

Dr. Katherine Perry

Senior Capstone

Submitted in Partial Fulfillment of the Requirements for the Degree of

B.A. in Liberal Arts

Social and Behavioral Sciences

Applied Mathematics

2022

Acknowledgements

I thank my mathematics professors, Dr. Anna Varvak and Dr. Jonathan Merzel, for always keeping my spirit high in mathematical inquiry and analysis. Without their mentoring in my independent studies, I would not have gained mathematical sophistication, however little, I now rejoice. My massive gratitude extends to Dr. Peter Johnson at Auburn University for introducing me to the joy of mathematical research, particularly in graph theory. Dr. Dean Hoffman at Auburn University receives a special recognition for showing me how fun mathematics can be and for introducing me to the game of *Slither*.

Most of all, I would like to thank my professor and mentor, Dr. Katherine Perry, who has gone above and beyond what is expected of a supervisor, with her generosity, her advice, and her encouragement. I greatly appreciate my friends, Biplab Karki and Rodrigo Morais, for their help in coding *Slither*. Not the least, I am thoroughly thankful to my parents and my supportive friends, who have listened to me as I talked about things they could have cared less.

Abstract

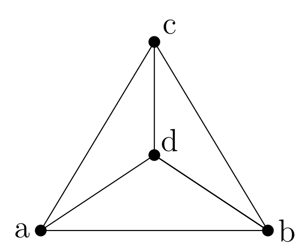
Slither is a multi-player path-making game played on a given host graph. The current project — in its essence — is a narrative of a young mathematician's exploratory journey from the inception of a given mathematical problem to its findings, generalizations, musings, applications, and directions for further study. The paper starts with descriptions of graph theoretical tools employed in *Slither* and its historical beginnings on a 5×6 grid graph. The paper continues to give general strategy for a 2-player game on any host graph, followed by attempts to explore winning strategies for 3-player games and generalization to n-player games. Towards the end, a randomized algorithm is developed for computer-versus-computer games, which is employed in a web application. The paper concludes with potential applications of ideas involved in *Slither* to problems in game theory and optimization.

Contents

1 Introduction

1.1 Graph Theory

1.1.1. Definition. A *graph*, $G(V, E)$, is a set V of *vertices* and a set E of pairs of vertices, called *edges*. Take an example of a graph where the set of vertices is $V = \{a, b, c, d\}$, and the set of edges is $E = \{(a, b), (b, c), (c, a), (a, d), (b, d), (c, d)\}$. This can be graphically represented — which is why it is called a "graph" — as follows:



Some properties of this particular graph are:

1. It is a *simple graph*, meaning no directions are associated with the edges, no vertices have self-loops as edges, and no pair of given vertices have more than one edge.

2. It is a *complete graph* on 4 vertices (denoted K_4), meaning each vertex in the graph is connected to every other vertex.

3. It is a *planar graph*, meaning it can be drawn on the plane without its edges crossing. In fact, it is the largest complete graph that is planar.

4. It has a *perfect matching*, meaning a set of distinct pairs of vertices can be constructed, which includes all vertices of the graph.

5. It has a *Hamiltonian path*, meaning one can trace a path along its edges, which includes all vertices of the graph without repeating an edge.

Primarily, graph theory is the study of such patterns, structures, and relationships present in and/or among graphs. Any important definitions pertaining to graphs will be explained as needed in the paper. For undefined unfamiliar terms, the reader may refer to Douglas B. West's *An Introduction to Graph Theory* [12].

1.2 A Game on Graph: *Slither*

The game of *Slither* first appeared in the June 1972 issue of *Scientific American* in the "Mathematical Games" series of Martin Gardner [5]. The game was conceived by David L. Silverman in his game collection *Your Move* [10]. The earliest version of the game goes like this: Two players are given a planar lattice of dimension 5×6. Each player take turns drawing an orthogonal unit segment such that the new segment is connected to the path, on either side. The player that is forced to enclose a boundary or has no further move remaining loses.

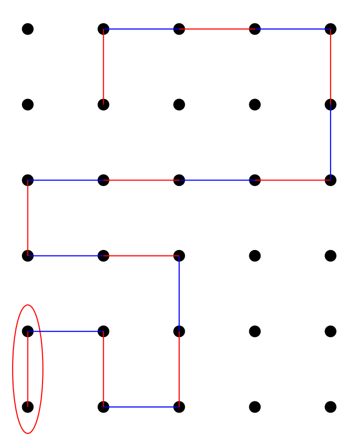


Figure 1: An example of *Slither* game on 5×6 lattice

In Figure 1 above, let us assume that the game began at the bottom left corner (circled red) with red as player 1. The game being played above is a player 1 win game. Although the game is still not over, the reader can continue a few more steps to see the eventual outcome. Whatever choice blue makes next, red can make a move, and blue has no move to make. *Can player 1 always win this game no matter what player 2 does?* This simple game can be given a graph theoretic treatment to formalize who wins and who loses the game. These insights will be treated in Section 2.1 of this paper, as will be the generalized version of the game and its winning strategy in Section 2.2.

Along the paper, we introduce another, slightly modified, version of *Slither*, which we call the 2-player progressive game: We are given an undirected host graph, i.e., a playground for the game. The players successively make their moves such that they trace a directed path. The player unable to add an edge to the path loses. Notice that the rules here are slightly different from the previous game, in that we are tracing a directed path starting from a vertex.

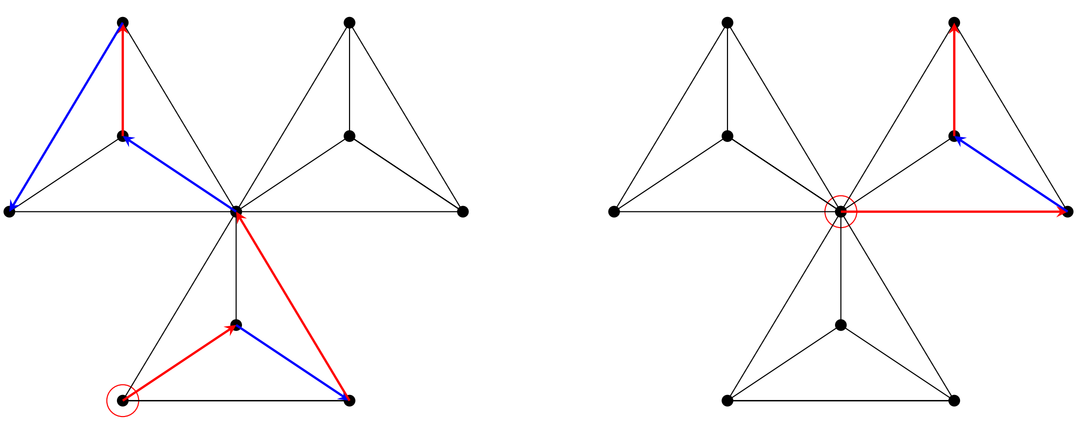


Figure 2: Two cases of 2-player progressive *Slither*

In the Figure 2 above, the game on the left, given that the red (player 1) started the game from the circled vertex at the bottom left, is a player 2 win. In the second game above (right), given that red started the game from the circled vertex at the center of the graph, the game is a player 1 win. *Was there a strategy for a particular player to always win this game?* We will be answering this question in Section 2.3 of this paper. Further generalizations for the game of *Slither* and the winning ways are described graph-theoretically in the remaining sections of Chapter 2.

As we shall see, *Slither* is an intriguing game with profound complexities than it appears. The remainder of the paper will discuss some of the associations of the game to game theory. *Slither*, being a decision making game, is a robust tool to make analogies with game theory. The paper will end with a presentation of the online version of *Slither* that we created for anyone to play against one-another or against the computer.

2 Winning Strategies

2.1 The Classic Game

By the classic game, we refer to the earliest version of the game described by Gardner in the June 1972 issue of *Scientific American* [5]. Gardner's follow-up article in the October 1972 issue highlights some of the results [6]. He points out that, given the 5-by-6 field as a sample playing field, the first player has an easy win by taking the central edge and thereafter making his moves symmetrically opposite to his opponent's moves. One such game is played below. (Note: Unless otherwise stated, the first player is always red, the first move is circled — ellipsed, to be precise — and the latest move in the game is dotted.) The reader can continue a few more steps to see player 1 (red) wins the game. This strategy is always employable because of the particular dimension of the graph. The reader may experiment with a few more games for verification.

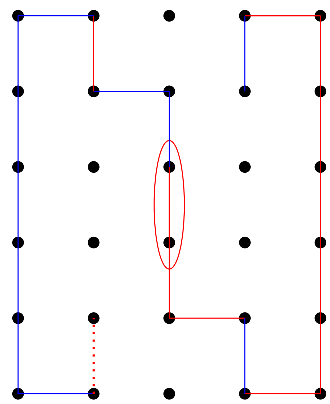


Figure 3: The classic *Slither* played with the winning strategy in Gardner's article

Here, we propose a different winning strategy for player 1. Start at the bottom left with a vertical move. Now, always follow the move that player 2 makes, meaning if player 2 makes a horizontal move, make a horizontal move, and if player 2 makes a vertical move, make a vertical move. This works most of the time because of the following logic: The only way player 2 can try to win the game on a board with an even number of vertices is by creating a path with all but an odd number of vertices, but if player 1 replicates player 2's moves, player 2 cannot usually skip an odd number of vertices. One such game is played below:

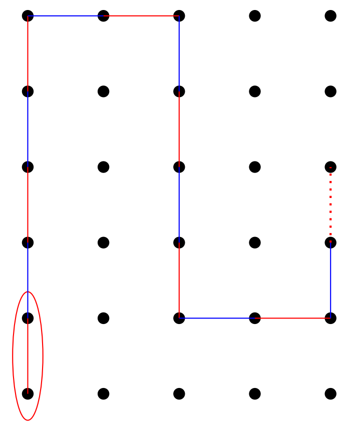


Figure 4: The classic *Slither* played with the replication strategy

There may be obvious instances when player 1 should not replicate the move, such as an instance given below where continuing the move forms a cycle. Player 1 would never want to willingly lose, so at those instances, player 1 should choose to divert the move. Player 1 continues replicating player 2's moves again. A game using such strategy is played below:

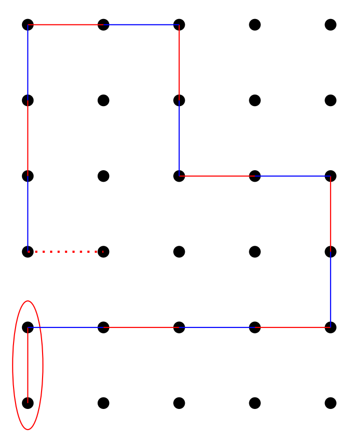


Figure 5: The classic *Slither* played with the replicate-divert-replicate strategy

Another instance where player 1 is unable to copy player 2 is when player 2 is able to skip odd number of vertices. At all these instances, the field of remaining vertices is partitioned to two grounds with either odd or even number vertices. Player 1 should play towards the field with odd number of vertices. one such game is played below:

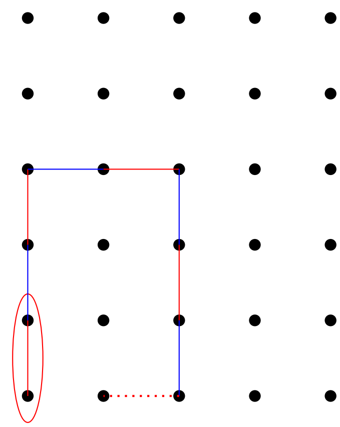


Figure 6: The classic *Slither* played with the odd-even partition strategy

2.2 2-Player Classic Game on a $p \times q$ Lattice

Gardner's follow-up article addresses some generalization of the classic *Slither*, due to R. Read [6]. Primarily, if the lattice is made of an even number of dots, it is possible to draw a Hamiltonian path with alternating colors, starting and ending with the same color, meaning player 1 can win. Recall that a Hamiltonian path is a path that consists of all vertices of the graph. If the lattice is made of an odd number if dots, player 2 can leverage the same strategy by tracing a Hamiltonian path after player 1 makes the first move. *But can a player always achieve such goal?* Let us proceed with the simple case of the 2×2 lattice.

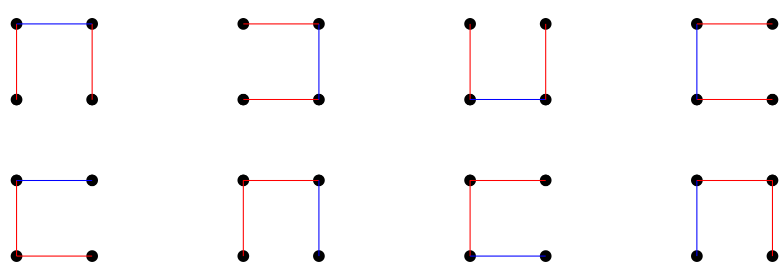


Figure 7: All 8 possible games on a 2×2 grid

The only possibilities for a 2×2 lattice are the eight games shown in Figure 7 above. In all cases, it is a player 1, i.e., red, win game. The above eight games are essentially only two games (the first column) under rotational and reflective symmetry. Precisely stated, all above possibilities are *isomorphic* to each other, shape-wise.

The game on a 3×3 lattice always takes one of the following shapes (Figure 8). Here, we do not draw all possible games, for there are far too many. ***For the reader****: What is the number of all possible games that can be played on a* 3×3 *lattice?* It is nonetheless apparent it is a player 2 (blue) win game. It is such because player 2 has no incentive to make a path shorter than the one consisting all vertices and is never forced to do so. Therefore, as the players make moves, the final path is always a Hamiltonian path.

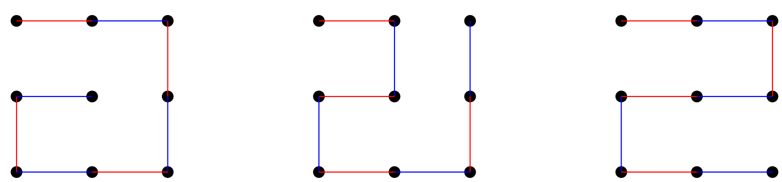


Figure 8: All possible path shapes for a game on 3×3 grid

Considering the three cases presented above — 2×2 and 3×3 — we proceed to hypothesize that player 1 wins all games played on a field with an even number of vertices, and player 2 wins all games played on a field with an odd number of vertices. We need to introduce the concept of graph factorization to proceed to verify such a hypothesis.

2.2.1. Definition. A graph is said to have a *1-factor* if it is possible to join all the vertices in pairs so that every vertex belongs to one and only one of the disjoint edges. Precisely, *1-factor* is a spanning 1-regular subgraph of G. Figure 10 shows an example of a 1-factor of the graph in Figure 9.

2.2.2. Proposition. The classic *Slither* is a player 1 win game if the number of vertices is even.

Proof. As the players are allowed orthogonal moves, the game can be thought of as being played on the graph shown in Figure 9 below. The graph so formed is called a *grid graph*. We will call grid graphs with an even number of vertices an *even grid graph*, and grid graphs with an odd number of vertices an *odd grid graph*.

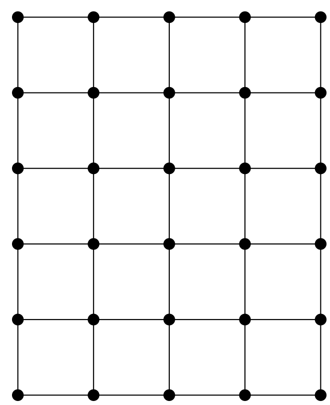


Figure 9: A 5×6 grid graph

2.2.3. Remark. All even grid graphs can be 1-factored, by construction.

As such, player 1 can always use the edges in the 1-factor since each vertex has an edge in the 1-factor, and player 2 is forced to make a choose edges not in the 1-factor. And it is always possible for the player 1 to use the edges of the 1-factor for the first and the last move as there is an odd number of them in an even grid graph. Therefore, player 1 always wins classic *Slither* played on an even grid graph. $\square$

There may be several possible ways of 1-factoring a particular even grid graph. But player 1 simply needs to choose one of them. One of the possibilities of 1-factoring a 5×6 grid graph is shown in Figure 10 below as an example. The reader can see that player 1 is always able to choose one of the edges of the following graph as the game proceeds. ***For the reader***: *In how many distinct ways can a given even grid graph with dimension $p \times q$ be 1-factored?*

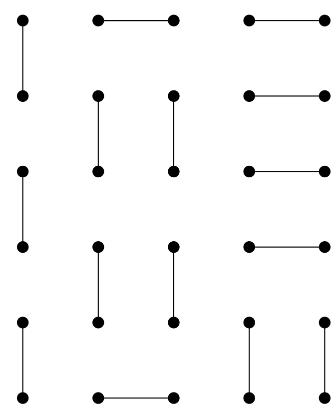


Figure 10: One possible 1-factor of a 5×6 grid graph

2.2.4. Corollary The classic *Slither* is a player 2 win game if the number of vertices is odd.

Proof. After player 1 makes the move, one of the ends of the player 1's move and all remaining vertices total to an even number of vertices. Player 2 can 1-factor these vertices and always choose to follow the edges as in Proposition 2.2.2. $\square$

2.3 2-Player Progressive Game

By the 2-player progressive game, we refer to the second game introduced in Section 1.2 of this paper. Here, the players continue making a directed path simultaneously. One who cannot continue making a path loses the game. The strategy for a progressive game requires us to introduce the concept of matching, which, as we shall see, is closely related to 1-factorization.

2.3.1. Definition. A *matching* M in a graph G is a set of pairwise non-adjacent edges, such that no edges share common vertices. A *maximum matching* is a matching with the maximum possible number of edges. A *perfect matching* is a matching that contains all edges.

2.3.2. Remark. A graph with a perfect matching can be 1-factored.

This leads to conclusion that, given a host graph with a perfect matching, player 1 always wins the 2-player game. We, therefore, look at graphs without perfect matchings.

2.3.3. Definition. Let M be a matching. A vertex $v \in V$ is *M-saturated* if v is an endpoint of an edge in M; otherwise, v is called *M-unsaturated*.

2.3.4. Definition. A path P is called an *M-alternating path* if the edges of P are alternately in M and not in M. An M-alternating path that begins and ends at unsaturated vertices is called an *M-augmenting path*.

2.3.5. Theorem. [3] A matching M of a graph G is a maximum matching if and only if G has no M-augmenting path.

Proof. Let us suppose there exists a matching M that is not maximum and has no M-augmenting path. Then there exists a matching M', which is a maximum matching. Let H be a subgraph with the vertex set $V(H) = V(G)$ and edge set $E(H) = M \cup M'$. Since each vertex of H has a maximum degree of 2, each component of H is either a cycle of even length with edges alternately in M and M', a path with edges alternately in M and M', or a path consisting of one edge e with $e \in M \cap M'$. Since $|M'| > |M|$, there is at a component of H which is a path P such that $|M'| > |M|$. This means P is an M-augmenting path. Therefore, by contradiction, if G has no M-augmenting path, a matching M of a graph G is a maximum matching.

To prove the other direction of the proof, we prove the contrapositive, i.e., if G has an

M-augmenting path, the matching M is not a maximum. If G has an M-augmenting path, switching edges along the augmented path will result in a matching M' with a cardinality larger than that of M. Hence, if G has no M-augmenting path, the matching M is a maximum matching. □

2.3.6. Definition. A vertex is called a *universal vertex* if it is present in all maximum matching sets.

In Figure 11, we see that the vertex at the center (circled green) is present in all sets of maximum matchings, meaning it is a universal vertex. Figure 11 is the same graph as in Figure 2 with its universal vertex labeled. The two sample games presented in Figure 2, therefore, show that starting from the universal vertex leads to player 1 winning and not doing so leads to player 2 winning. *Is that always true?*

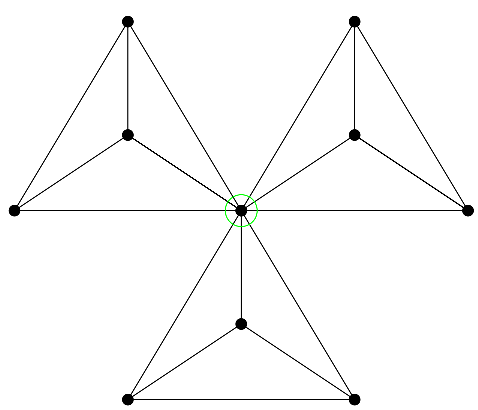


Figure 11: Graph in Figure 2 with the universal vertex labeled

2.3.7. Lemma. An M-alternating path that starts at a universal vertex of a maximum matching M has both end vertices of the path *saturated*.

Proof. Let P be an M-alternating path that starts at a universal vertex of a maximum matching M. Theorem 2.3.5 implies at least one end vertex of P is M-saturated. If the number of edges in a path is even, the number of vertices is odd, and the end vertices cannot be universal vertices. This means, if an M-alternating path starts at a universal vertex, the number of edges is odd. It is clear that a path with odd number of edges has both ends of the path M-saturated. $\square$

2.3.8. Theorem. [1] The 2-player progressive *Slither* is a player 1 win game if player 1 starts at a universal vertex of the maximum matching.

Proof. Let M be a maximum matching, and r be a universal vertex of the maximum matching. Let P be a path containing M that starts at r. Lemma 2.3.7 implies P has an odd number of edges and that the edges at the ends are in the maximum matching. Player 1 can, therefore, take an M-alternating path and be able to make the starting and the ending move. $\square$

2.3.9. Corollary. The 2-player progressive *Slither* is a player 2 win game if the host graph does not have a universal vertex.

2.3.10. Remark. For a graph with a perfect matching, which can always be 1-factored, each vertex is a universal vertex.

If we consider the 5×6 grid graph now, we now see why it is a player 1 win game no matter where player 1 starts. Further, even if diagonal moves were allowed (or for that matter, whatsoever legal move be introduced), it would still be a player 1 win game, for the perfect matching is already present in an even grid graph.

2.3.11. Remark. A odd grid graph does not have an universal vertex.

2.3.11. Remark. The corollary 2.3.9 implies player 2 always wins in an odd grid, which overlaps with the conclusion in corollary 2.2.4.

2.4 3-player Classic Game on $p \times q$ Lattice

Now that we know when and how a particular player wins a given game of *Slither* with two players, for it is a mathematician's muse to indulge in generalizations, we ask: *what happens if we have 3 players?* First, it should be noted that the game of *Slither* transforms to trying not to lose rather than trying to win because, in a game of 3 players, there are 2 winners. The one player who is forced to make a cycle or has no move to make loses the game.

In a 2 $\times$2 lattice, the longest path is a path on 4 vertices (denoted P_4) with 3 edges. After player 1, player 2, and player 3 successively make their moves, player 1 has no choice but to make a cycle and hence always loses. This simplicity suddenly disappears as we move to a 3×3 lattice.

For a 3×3 lattice, unlike in a 2-player game, it is not apparent if the players are always forced to form a Hamiltonian path. For example, see three games played in Figure 12 below. The edges are labeled with the move numbers, and the colors represent players (red = player 1, blue = player 2, green = player 3).

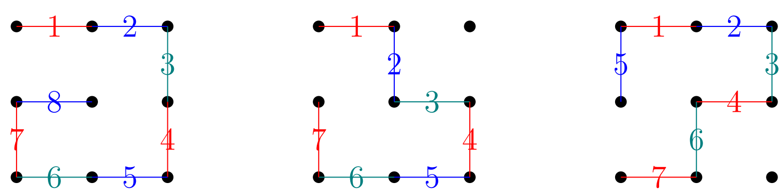


Figure 12: Three possibilities of a 3-player *Slither* on a 3×3 lattice

The first game (on the left) is a game on Hamiltonian path. The maximum length of a path on a graph with 9 vertices is 8. If all 8 edges are traversed, player 3 (green) does not have a move at the end and hence loses. If the game forms a Hamiltonian path, who loses a

3-player game played on a $p \times q$ grid can be figured as follows: Let $m = p.q - 1$, where m denotes the length of the Hamiltonian path. Let $s = m \bmod 3$, where s denotes the player who makes the last legal move in the game. Then player $(s+1)$ loses the game. As we saw, in a 3×3 grid, $m = 3.3 - 1 = 8$, $s = 8 \bmod 3 = 2$, so player $(2+1) = 3$ loses the game.

The second game (in the middle), however, is not a game on Hamiltonian path. As we skipped a vertex, the number of edges in the path is 7. $7 \bmod 3 = 1$. So player 2 (blue) loses the game. It appears that blue's second move was a bad move, and if it meant losing, it would not have made that move.

In the third game (on the right), blue makes its second move same as in the first game. But red and green can make moves such that blue (player 2) loses again.

The major conclusion of three experiments above is that player 1 can decide who wins the game. Player 1 never loses a 3×3 game as it is not possible for other players to force skipping two vertices. The simple conclusion here has a fundamental implication to group dynamics. Although that direction is not a primary goal of the discussions in this section, it is apparent how player 1 can cooperate with either player 2 or player 3 to decide their fate in the game. And it is no surprise that we see similar phenomena in circumstances outside of mathematical games.

One can anticipate how nuanced a 3-player game on higher dimensions can become. Here, we shall introduce the notion of cooperation. We will call the game C_{12} if player 1 and player 2 cooperate, C_{13} if player 1 and player 3 cooperate, and C_{23} if player 2 and player 3 cooperate. The cooperations are obviously commutative. And these three are all possible cooperations in a 3-player game. Using the discussions we have made hitherto in Section 2.4, we will develop the following lemma.

2.4.1. Lemma. Let P be the path taken by players in a game of *Slither* on a $p \times q$ grid graph. Let n be the length of P. Let s denote the player who makes the last legal move in the game. Then

1. $s = n \bmod 3$
2. If $s = 0$, player 1 loses the game
3. If $s = 1$, player 2 loses the game
4. If $s = 2$, player 3 loses the game

The longest possible path that can be formed in a $p \times q$ grid graph is a Hamiltonian path of length $p.q - 1$. If we now let C_{ij}, where $i, j = \{1, 2, 3\}, i \neq j$, denote the strategy where players i and j coordinate against the remaining player, the number of vertices to skip such that players i and j win against the remaining player is t, where $s = (p.q - 1 - t) \bmod 3$. *But how do we figure t?* The values of t are developed in discussions below and given in the C_{ij} columns of succeeding tables. Let us proceed by splitting $p \times q$ into following sub-cases.

Case I : $p \times 1$

For all C_{12}, C_{13}, and C_{23}, the classic game on $p \times 1$ grid must be a Hamiltonian path. As such, player $(s + 1)$ always loses the game, where $s = (p - 1) \bmod 3$, as defined earlier in lemma 2.4.1. Example:

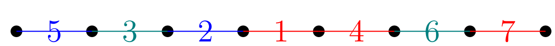


Figure 13: A 3-player game on 8×1 grid, where s = 7 mod 3 = 1, meaning player 2 loses.

Case II : $p \times 2$

We have already dealt with the simplest case where $p = 2$, i.e., a 2×2 lattice. If the path formed for the game is a Hamiltonian path, who loses is figured through rows where $C_{ij} = 0$. If $C_{12} = 0$, player 3 loses; if $C_{13} = 0$, player 2 loses; if $C_{23} = 0$, player 1 loses. A Hamiltonian path is the desired strategy in C_{12} when $(2p-1)$ mod $3 = 2$, in C_{13} when $(2p-1)$ mod $3 = 1$, and in C_{23} when $(2p-1)$ mod $3 = 0$. However, two of the players may be able to manipulate the game, forcing a certain number of vertices to be skipped. For example, in $p = 3$ case, player 3 loses a Hamiltonian game; C_{12} do not need to skip any vertices to win; if player 1 and player 3 want to win, they need to try to skip 1 vertex; if player 2 and player 3 want to win, they need to try to skip 2 vertices. It is not necessary to skip exactly the number of vertices in the columns C_{ij}. The number of vertices to skip can be, for $k \in \mathbb{Z}$, $3k + t$, where t is the number in the columns C_{ij}. That is to say, the number of vertices to skip is congruent to numbers in the column C_{ij} (mod 3).

p	$(2p-1)$	$mod\, 3$	C_{12}	C_{13}	C_{23}
3	5	2	0	1	2
4	7	1	2	0	1
5	9	0	1	2	0
6	11	2	0	1	2
7	13	1	2	0	1
8	15	0	1	2	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

We have stated fundamental strategies to employ. But it is not still clear if these strategies can be employed. For example, with a few experiments, we can see a 3×2 game must always form a Hamilton path, so C_{13} or C_{23} is not possible, and player 3 always loses. Player 1 never loses as long as $C_{23} \neq 0$ because it can always force a Hamiltonian path in a

$p \times 2$ lattice. Indeed, player 1 never wins a $C_{23} = 0 \pmod 3$ on a $p \times 2$ grid graph because player 2 and 3 can always collaborate to form a Hamiltonian path should they wish. We will first observe what happens in higher dimensions before we dive into the question of whether skipping certain number of vertices (mod 3) is possible.

Case III : $p \times 3$

The table here simply lists the number of vertices to skip (mod 3). All denominations for this and the subsections that follow are same as explained in discussions hitherto.

p	$(3p-1)$	$mod\,3$	C_{12}	C_{13}	C_{23}
3	8	2	0	1	2
4	11	2	0	1	2
5	14	2	0	1	2
6	17	2	0	1	2
7	20	2	0	1	2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Again, we have already seen the case where the game C_{23} is not possible for $p = 3$, as player 1 never loses in 3×3 lattice. *Are all other strategies employable?*

Case IV : $p \times 4$

We will start with $p = 4$ because $p = 3$ in Case IV is given by $p = 4$ in Case III.

p	$(4p-1)$	$mod\, 3$	C_{12}	C_{13}	C_{23}
4	15	0	1	2	0
5	19	1	2	0	1
6	23	2	0	1	2
7	27	0	1	2	0
8	31	1	2	0	1
9	35	2	0	1	2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Case V : $p \times q$

p	q	$(pq-1)$	$mod\, 3$	C_{12}	C_{13}	C_{23}
p	q	$(pq-1)$	0	1	2	0
p	q	$(pq-1)$	1	2	0	1
p	q	$(pq-1)$	2	0	1	2

The general table here corresponds to the specific cases developed sequentially in earlier cases I – IV. Discerning whether employing the tabulated strategy appears to be a computationally intense question. As such, we will leave readers with a conjecture that may be addressed in future works.

2.4.2. Conjecture. For a large enough grid, i.e. $\min(p, q) > 3$, it is always possible for any 2 players to collude and win against the 3^{rd} player.

2.5 n-player Classic Game on $p \times q$ Lattice

With slight change of notation, $C_{\overline{x}}, x \in \{1, 2, \ldots, n\}$, means that all other players except the player x collaborate to force player x to lose.

p	q	$(pq-1)$	$mod\, n$	$C_{\overline{n}}$	$C_{\overline{n-1}}$	$C_{\overline{n-2}}$	$\cdots$	$C_{\overline{2}}$	$C_{\overline{1}}$
p	q	$(pq-1)$	0	1	2	3	$\cdots$	$(n-1)$	0
p	q	$(pq-1)$	1	2	3	4	$\cdots$	0	1
p	q	$(pq-1)$	2	3	4	5	$\cdots$	1	2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
p	q	$(pq-1)$	$(n-2)$	$(n-1)$	0	1	$\cdots$	$(n-3)$	$(n-2)$
p	q	$(pq-1)$	$(n-1)$	0	1	2	$\cdots$	$(n-2)$	$(n-1)$

When $n = 3$, the general table here corresponds to the specific cases in subsections of Section 2.4. The generalized conjecture for n-player *Slither* is as follows:

2.5.1 Conjecture. Given a n-player game on a large enough grid, i.e., $\min(p, q) > n$ there always exists a strategy for $(n-1)$ players to collude and make the remaining player lose.

3 Developing the Classic *Slither* Bot

To gain more insights into the nature of 3-player games, we created a randomized algorithm. Given a pair (m, n), the algorithm plays a random game of *Slither* on $m \times n$ grid. The game begins with a random choice of an edge, creates a list of untraversed neighboring edges, and proceeds to form a path by choosing a random neighbor. This process iterates until neither of the first or the last vertex of the path has a untraversed neighboring edge. Counting the path length at the end helps us figure out who won/lost the game. The Python script below is an example of a 2-player game. In the specific example below, the game repeats 100 times. But a trivial addition of code at the end generates winners/loser of an n-player game for any number of repeats.

```
import random

for i in range(100):

    #Creating the possible array of vertices. All vertices have a value
    ↪  from 1 to m*n.
    def grid(m,n):
        value_list = []
        j = 1
        for item in range(n):
            row_list = []
            for item in range(m):
                row_list.append(j)
                j = j + 1
            value_list.append(row_list)
        return value_list
        #print(value_list)

    #Finding the neighbor list of all vertices. The output is a list of
    ↪  dictionaries with index, value, and neighbor list.
    #Neighbors are horizontally and vertically adjacent vertices.
    #credits to GitHub: MaxRudometkin
    def find_neighbours(arr):

        neighbors = []

        for i in range(len(arr)):
            for j, value in enumerate(arr[i]):

                if i == 0 or i == len(arr) - 1 or j == 0 or j ==
                ↪  len(arr[i]) - 1:
```

```
                # corners
                new_neighbors = []
                if i != 0:
                    new_neighbors.append(arr[i - 1][j])  # top
                     ↪  neighbor
                if j != len(arr[i]) - 1:
                    new_neighbors.append(arr[i][j + 1])  # right
                     ↪  neighbor
                if i != len(arr) - 1:
                    new_neighbors.append(arr[i + 1][j])  # bottom
                     ↪  neighbor
                if j != 0:
                    new_neighbors.append(arr[i][j - 1])  # left
                     ↪  neighbor

            else:
                # add neighbors
                new_neighbors = [
                    arr[i - 1][j],  # top neighbor
                    arr[i][j + 1],  # right neighbor
                    arr[i + 1][j],  # bottom neighbor
                    arr[i][j - 1]   # left neighbor
                ]

            neighbors.append({
                "index": i * len(arr[i]) + j,
                "value": value,
                "neighbors": new_neighbors})

    #print(neighbors)
    return neighbors


arr = grid(5,4)
list_of_dict = find_neighbours(arr)
list_of_dict_for_removal = list_of_dict
```

```python
#print(list_of_dict)
#print(list_of_dict_for_removal)

#all_nodes = [i["value"] for i in list_of_dict]
#print(all_nodes)

#selecting one of the values to start the game
random_index = random.randint(0,len(list_of_dict)-1)
first_node_dict = list_of_dict[random_index]
first_node = first_node_dict["value"]
#print("Printing first node *********")
#print(first_node)
path_list = []
path_list.append(first_node)

first_node_neighbor_list = (first_node_dict["neighbors"])

random_index_again = random.randint(0,len(first_node_neighbor_list)-1)
selected_node = first_node_neighbor_list[random_index_again]
#print("Printing second node *********")
#print(selected_node)
path_list.append(selected_node)
#print("Printing the starting edge *********")
#print(path_list)

def removal(node):

    #remove the first two nodes from all neighbor lists
    for i in list_of_dict:
        try:
            i["neighbors"].remove(node)
        except ValueError:
            #print("could not find {} in
            ↪  {}".format(j,i["neighbors"]))
            continue
```

```
removal(first_node)
removal(selected_node)

#print(list_of_dict)

while True:

    #randomly choosing one of the vertex from the existing path
    next_node = random.choice([path_list[0],
     ↪  path_list[len(path_list)-1]])
    #print("Printing next node ********* ")
    #print(next_node)

    #removing the selected node from all neighbor lists
    removal(next_node)

    if next_node == path_list[0]:

        next_node_dict = list_of_dict[next_node-1]
        next_neighbors = next_node_dict["neighbors"]
        #print("Printing next node neighbors *******")
        #print(next_neighbors)
        if len(next_neighbors) == 0:
            break

        next_next_neighbor = random.choice(next_neighbors)
        #print("Printing chosen neighbor *******")
        #print(next_next_neighbor)

        #removing the selected node from all neighbor lists
        removal(next_next_neighbor)

        path_list.insert(0, next_next_neighbor)

    elif next_node == path_list[len(path_list)-1]:
```

```
            next_node_dict = list_of_dict[next_node-1]
            next_neighbors = next_node_dict["neighbors"]
            #print("Printing next node neighbors *******")
            #print(next_neighbors)
            if len(next_neighbors) == 0:
                break

            next_next_neighbor = random.choice(next_neighbors)
            #print("Printing chosen neighbor *******")
            #print(next_next_neighbor)

            #removing the selected node from all neighbor lists
            removal(next_next_neighbor)

            path_list.append(next_next_neighbor)

    print(path_list)

    path_length = len(path_list)

    if path_length%2 == 0:
        print("Player 1 won.")
    else:
        print("Player 2 won.")

print("Played 100 games and printed all path lists and outcomes.")
```

4 Developing the Classic *Slither* Online

Using the algorithm developed in the preceding section, a web application for *Slither* was created using JavaScript. For the reader who has read and understood the winning strategy, the online game is a way to impress the crowds. Reminder: Play a 2-player game, play as player 1, and play on an even grid graph. If your opponent wants to be player 1, say you want to decide the grid size, and play on an odd grid graph!

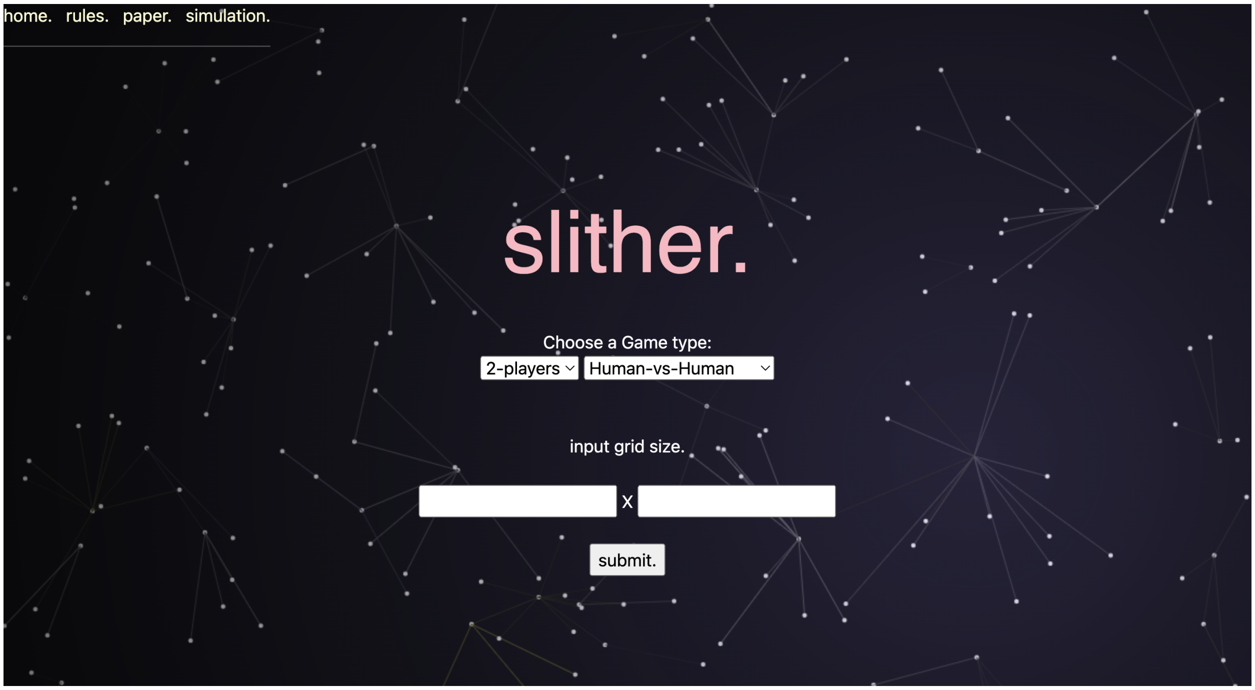


Figure 14: Web application home page for *Slither*

The web application allows one to play a 2-player game or a 3-player game. Both games can be played against another human or against a computer. The color scheme is preserved as in the paper. Player 1 is blue, player 2 is red, and player 3 is green.

The web application also runs randomized simulations for 2-player and 3-player games, where computer plays against itself, using the algorithm defined in the Section 3. The simulation can be run 50, 100, 500, or 1000 times. The simulation also shows the proportions of *losses* beared by each player.

The simulation was created with the hypothesis that several repetition of random games helps us predict the outcome of a given game. For example, 1000 simulation of a

3-player game in a 3×3 grid graph (Figure 13) shows that it possible for player 2 and 3 to lose the game, but player 1 never loses the game, exactly as we explained in the Section 2.4. The proportion of times player 3 loses is the highest in the figure, but the relevance of these proportions is not yet clear as the algorithm is random.

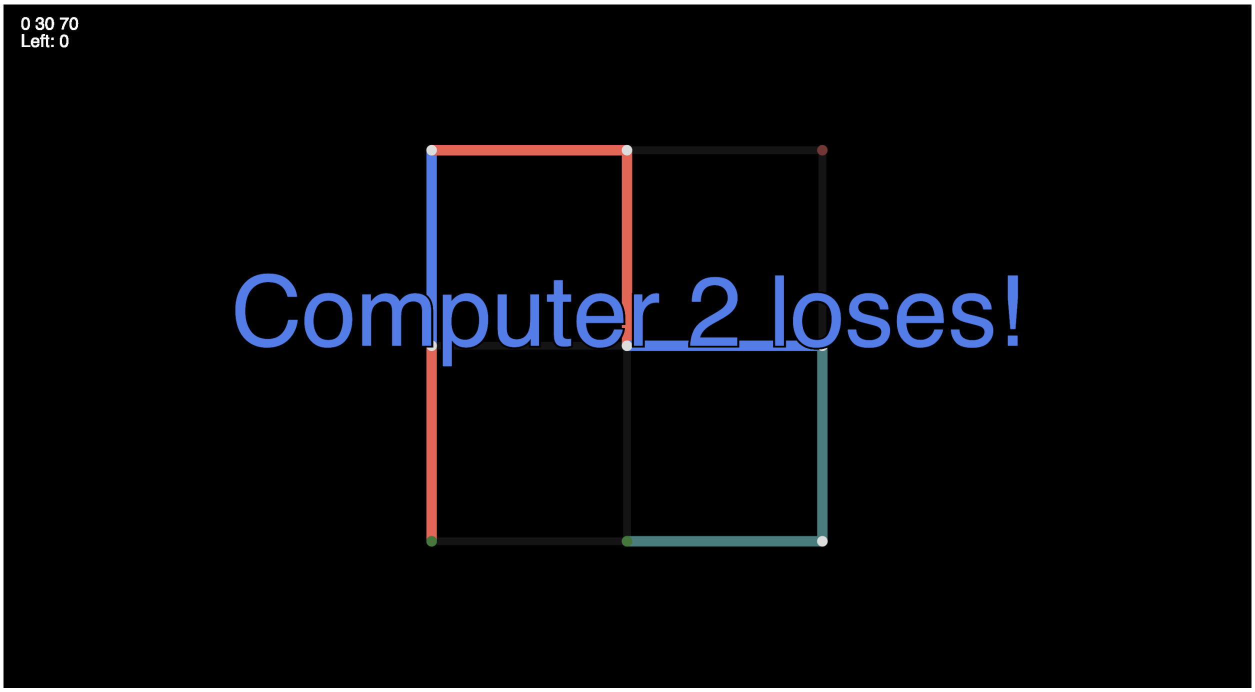


Figure 15: 3 player game on 3×3 lattice

Figure 14 below shows 1000 simulations of a 3-player game on a 6×4 lattice. We see that each player loses certain proportions of games. The existence of losses for each player implies each pair of 2 players must have a strategy to win against the 3^{rd} player. After experimenting with several games on several grids on the simulated game, we have stronger inclination to believe Conjecture 2.4.2 for *Slither*, i.e., any two players can always win against a 3^{rd} player, given a large enough lattice.

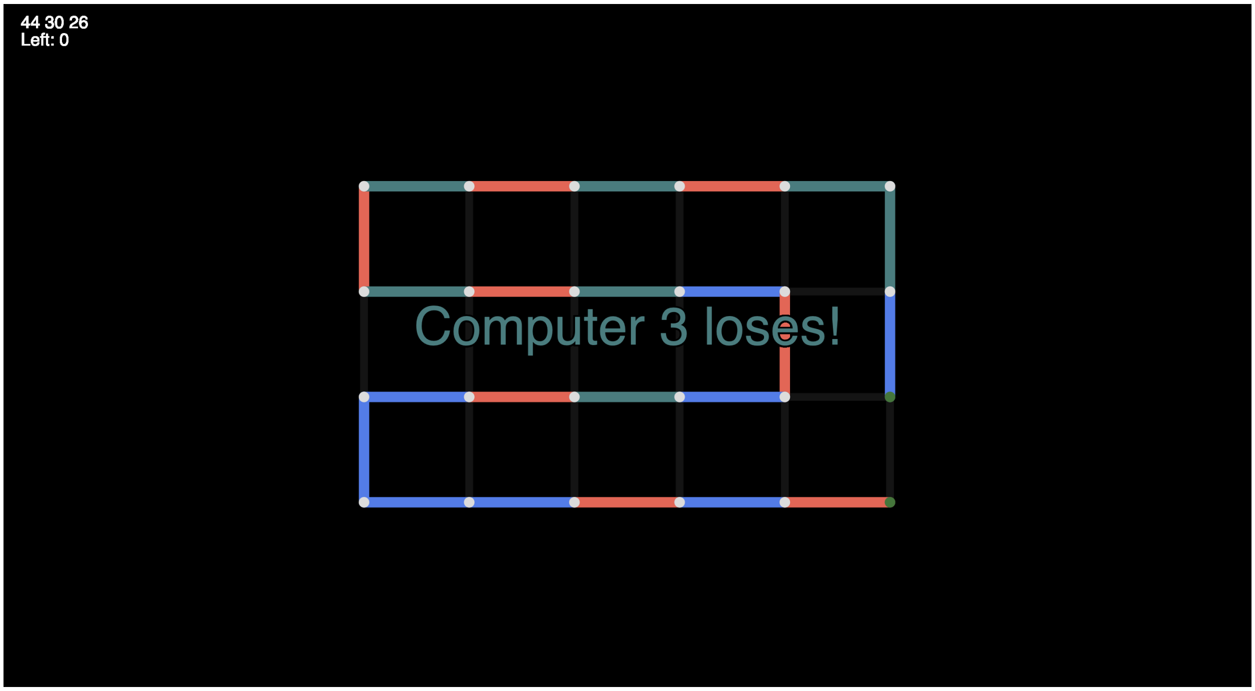


Figure 16: 3 player game on 6×4 lattice

Although we already have known theorems on 2-player games, we can check how randomized algorithm predicts outcomes. Figure 15 below shows 1000 simulations of a 2-player game on a 7×6 lattice. As expected player 1 loses lesser proportions of times. We must be careful, however, that finite experimentation of a randomized algorithm does not establish anything as true. It can nonetheless give us motivation for further questioning. The online page contains all the information pertaining to rules of the game. The link to the page is: *pchalise.github.io/Slither/*.

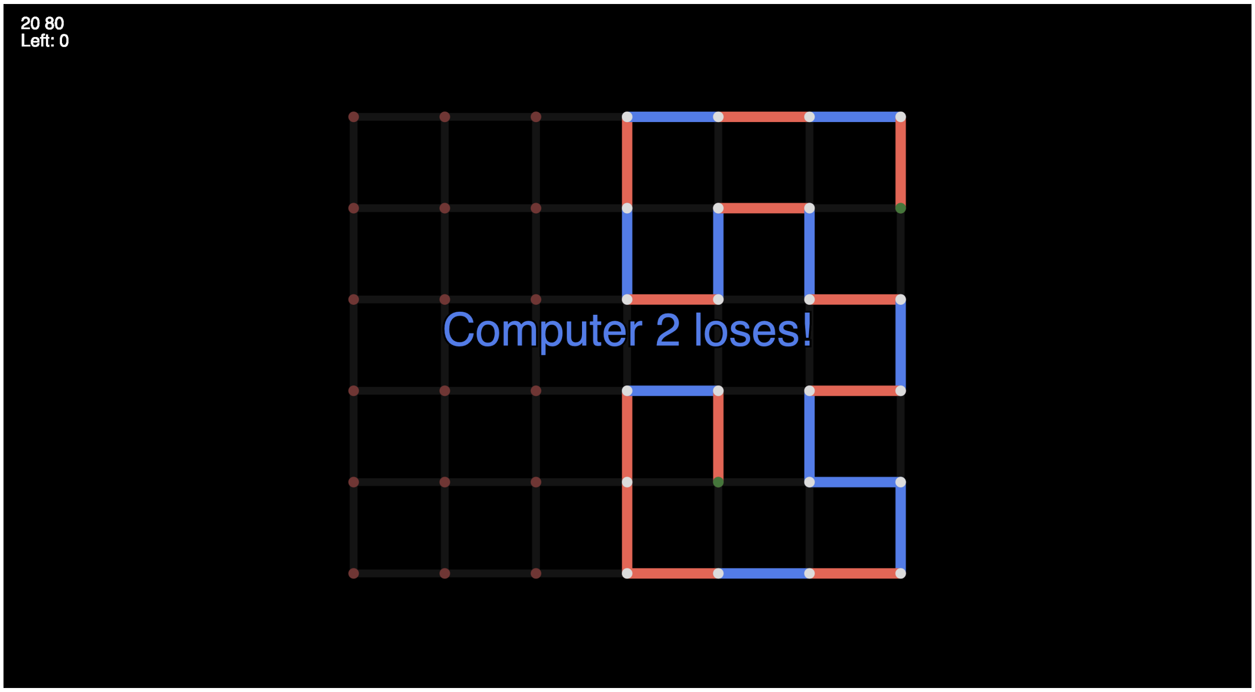


Figure 17: 2 player game on 7×6 lattice

5 Applications

> "How can it be that mathematics, being after all a product of human thought which is independent of experience, is so admirably appropriate to the objects of reality?"
>
> ---
>
> Albert Einstein

The formal questions in winning strategies in mathematical games rests on humorous beginnings, in passion of human beings for joy but a possessive hunger for truths. The current paper continues to extrapolate these ideas to several applications of ideas involved in *Slither*.

5.1 Game Theory

The discipline of game theory is based on the foundation that rules of any game unambiguously prescribe the following: i) Each player is fully cognizant of the game in extensive form, i.e., one is fully aware of the rules of the game and the utility functions of each of the players; ii) Of two or more alternatives which give rise to outcomes, a player will choose the one which yields the more preferred outcome, or, more precisely, in terms of the utility function, one will attempt to maximize expected utility [8]. The game of *Slither* translates to game theory, for the rules and winning strategies of the game are well-defined.

Any market of good and services has decision makers, finite resources, and the notion of competition and cooperation, all of which are present in the game of *Slither.* Each player's goal is to win the game and utility is maximized when one is able to follow the winning strategy at each step of the game. A player is always subject to constraints brought about by someone else's decision in any market and also in *Slither.* Most importantly, *Slither* is a game on a networked environment. As the world gets increasingly connected, decision-making too becomes a phenomena in a networked environment where one's choices are immensely influenced by others. As such, further study into variations of *Slither* in diverse sets of host graphs is likely to lead to important conclusions in the discipline of game theory.

5.2 Network Science

All models are wrong, but some are useful.

George E. P. Box

The use of graphs and graph theory as a tool to theorize physical, biological, or socio-political problems can be termed network science. Networks exist everywhere, for most structures and systems emerge from relationships between various attributes, i.e., vertices (nodes, items) connected by edges (links, connections, relationships). A simple example of a network is the social network in Facebook, where each profile is a vertex, and an edge exists between the vertices if two profiles are *friends* with each other. But one can imagine networks in internet, classrooms, financial system, transportation, ecosystem, and the list continues [2].

The applicability of graph theory also arises from its ability to be expressed visually. As such, graph theory is accessible to most audience. Overall, the versatility and applicability of graph theory in understanding not only mathematical structures but also natural and social aspects makes it a germane tool for progress in sciences and social sciences.

With regard to *Slither*, as briefly mentioned earlier, it demonstrates decision making constraints in a networked environment. In the game, one's decision is influenced by someone else's previous decision. Unlike classical notion of decision making where a rational decision-maker acts to maximize one's utility, decision-making in a networked environment is a dynamic process and one that is closet to our socio-political reality. Especially as the world becomes connected by online medias and information diffusion accelerates, decision-making evolves from an individual's choice to a network dynamic [7].

5.3 Matchings

The notion of matching was central to our theorems developed for winning strategy in classic *Slither.* Theoretical developments in matching have found numerous applications in allocation problems. The class of problems involving matchings are also called marriage problems. The theory of matchings is at the core of online dating sites, marriage portals, kidney donations, college applications, and so on.

The popular Gale and Shapley algorithm [4], also called deferred acceptance algorithm has been able to rectify public school assignment problems in segregated communities of San Francisco and Boston [9]. With regard to *Slither*, the subgraph that guarantees player 1 win game is also a maximum matching of the graph. As it is often in mathematics, there might be cases when finding the winning strategy is easier than finding the maximum matching. This hints at the versatility of *Slither* in contributing to useful theories applicable to society.

6 Conclusion

The game of *Slither* possesses sophisticated characteristics. As we saw, the 2-player game on a given $p \times q$ grid graph had an astounding simplicity. To revisit the winning strategy, player 1 always wins *Slither* on an even grid graph given one follows the M-alternating path, and player 2 always wins on an odd grid graph with the same strategy. In a game where players draw a directed path, one wins by starting at the universal vertex of the graph.

The paper followed with experimentations on 3-player *Slither*. We had no particular theorems for 3-player games. Hence, we conjectured that any two players could collaborate to win against the 3^{rd} player on a large enough graph. Generalizing that to n-players, we conjectured that any $(n-1)$ players could collaborate to win against the remaining player on a large enough graph.

Following the conjectures, we developed a randomized algorithm such that the computer plays *Slither* against itself. We employed algorithmic framework developed in Python to JavaScript to develop a web application so that anyone can play *Slither* online.

Towards the end of our paper, we presented introductory notions of several applications of ideas involved in *Slither* to various graph theoretical and game theoretical scenarios. As we see in the current paper, a simple mathematical wondering encompasses profound ideas across several inquiries in diverse disciplines. The study into other variations of *Slither* under several other constraints will lead to many more conclusions.

7 Future Works

Several directions are open to further the study of *Slither*, primarily in its computational solutions. For instance, Figure 16 is a 2-player game on 7×6 grid. Although Theorem 2.1 states player 1 never loses the particular game, the randomized game predicts the opposite. Hence the implications from the randomized game do not seem to be conclusive.

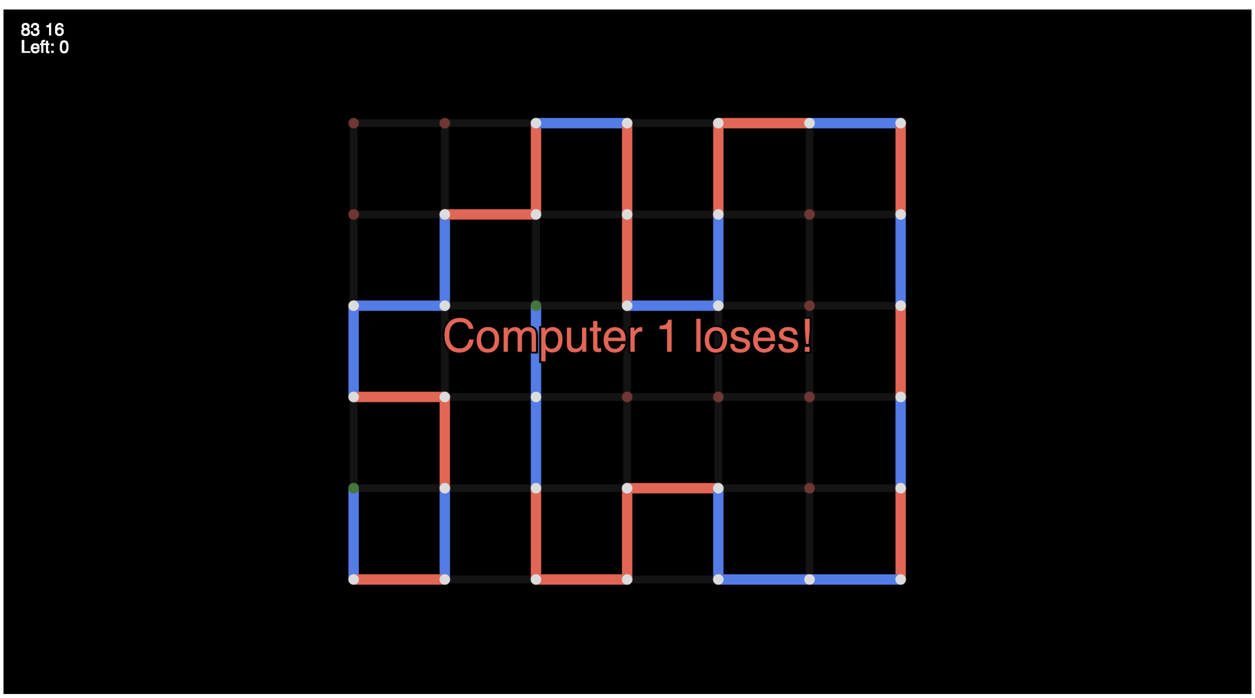


Figure 18: 2 player game on 7×6 lattice

The most important next step, therefore, is to test our conjectures in ≥ 3 players scenarios by building a machine learning enabled computer-versus-computer game. An unsupervised machine learning algorithm, after certain (large) number of iterations, finds the path an assigned player needs to take. The end goal can be conveniently defined using the matrix built in Section 2.5.

Studying other variations of *Slither* is another avenue to take. Although we briefly looked at the progressive game, we did not generalize it to more than 2 players. For a game of progressive *Slither*, given a large enough host graph, does there always exist a strategy for $(n-1)$ players to collaborate and win over the remaining player? If so, can such strategy be mathematically well-defined?

We suggest a new form of game called *Independent Slither*. Here, the players are not restricted to one single path. Rather, given n players, all players construct their own paths. Whoever cannot continue constructing the path, either because they no longer have any vertices or because they would collide against another player's path, loses the game. Independent *Slither* is applicable to game theoretic scenarios when one does not have to strictly depend on another's move, i.e., the freedom to construct your own path, until a competition arises for the scarce available resource, i.e., the finite grid graph [11].

Of course, other eclectic sets of rules can be applied to make the game more dynamic, such as a given player has the choice of either drawing a new edge or deleting the previous one. All types of games can be studied on other classes of host graphs. The grid graph is a rather uniform type of graph, so games on other host graphs may lead to some interesting conclusions. In essence, as long as the insatiable human pondering continues, any set of dots and lines can lead to the creation of knowledge never thought to exist.

References

[1] William N. Anderson Jr. "Maximum matching and the game of Slither". In: *J. Combinatorial Theory Ser. B* 17 (1974), pp. 234–239. ISSN: 0095-8956. DOI: 10.1016/0095-8956(74)90029-x. URL: https://doi.org/10.1016/0095-8956(74)90029-x.

[2] Albert-László Barabási and Márton Pósfai. *Network science*. Cambridge: Cambridge University Press, 2016. ISBN: 9781107076266 1107076269. URL: http://barabasi.com/networksciencebook/.

[3] Claude Berge. "Two Theorems in Graph Theory". In: *Proceedings of the National Academy of Sciences of the United States of America* 43.9 (1957), pp. 842–844. ISSN: 00278424. URL: http://www.jstor.org/stable/89875 (visited on 04/29/2022).

[4] D. Gale and L. S. Shapley. "College Admissions and the Stability of Marriage". In: *The American Mathematical Monthly* 69.1 (1962), pp. 9–15. ISSN: 00029890, 19300972. URL: http://www.jstor.org/stable/2312726 (visited on 04/29/2022).

[5] Martin Gardner. "MATHEMATICAL GAMES". In: *Scientific American* 202.2 (1960), pp. 150–156. ISSN: 00368733, 19467087. URL: http://www.jstor.org/stable/24941266 (visited on 04/29/2022).

[6] Martin Gardner. "MATHEMATICAL GAMES". In: *Scientific American* 227.4 (1972), pp. 110–113. ISSN: 00368733, 19467087. URL: http://www.jstor.org/stable/24922896 (visited on 04/29/2022).

[7] Pawan Kumar and Adwitiya Sinha. "Information diffusion modeling and analysis for socially interacting networks". In: *Social Network Analysis and Mining* 11.1 (2021),

p. 11. DOI: 10.1007/s13278-020-00719-7. URL: https://doi.org/10.1007/s13278-020-00719-7.

[8] R. Duncan Luce and Howard Raiffa. *Games and decisions : introduction and critical survey / R. Duncan Luce and Howard Raiffa*. English. Wiley New York, 1957, 509 p. :

[9] Parag A. Pathak. "The Mechanism Design Approach to Student Assignment". In: *Annual Review of Economics* 3.1 (2011), pp. 513–536. DOI: 10.1146/annurev-economics-061109-080213. eprint: https://doi.org/10.1146/annurev-economics-061109-080213. URL: https://doi.org/10.1146/annurev-economics-061109-080213.

[10] Gerald R. Rising. In: *The Arithmetic Teacher* 19.1 (1972), pp. 47–47. ISSN: 0004136X. URL: http://www.jstor.org/stable/41187893 (visited on 04/29/2022).

[11] Aaron N. Siegel. *Combinatorial game theory*. Vol. 146. Graduate Studies in Mathematics. American Mathematical Society, Providence, RI, 2013, pp. xiv+523. ISBN: 978-0-8218-5190-6. DOI: 10.1090/gsm/146. URL: https://doi.org/10.1090/gsm/146.

[12] Douglas B. West. *Introduction to graph theory*. Prentice Hall, Inc., Upper Saddle River, NJ, 1996, pp. xvi+512. ISBN: 0-13-227828-6.