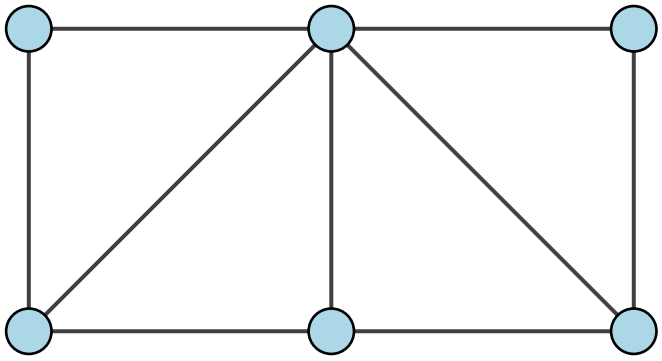
The Game of Cycles

Het Spel van Cycli

by

J.S. Zandee

to obtain the degree of Bachelor of Science
at the Delft University of Technology,
to be defended publicly on Friday July 8th, 2022 at 15:00.



Student number: 5163242
Project duration: February 28, 2022 – July 8, 2022
Thesis committee: dr. R.J. Fokkink, TU Delft, supervisor
dr. A. Bishnoi, TU Delft

TU Delft

Abstract

The Game of Cycles, invented by Francis Su (2020, p.51) is an impartial game played on a graph, where players take turns marking an edge according to a set of rules. Together with the game, there also came a conjecture that gives a condition for whether a specific position is winning or losing. Proving or disproving this conjecture is the main focus of this research, which we end up succeeding in by giving a counter-example, thus disproving the conjecture. We do this by first showcasing some relevant background knowledge from game theory in chapter 1. In chapter 2 we then introduce the Game of Cycles and its rules, as well as some of the previous results others have found. We continue in chapter 3 by creating a python script to brute-force the game for us and it is here that we find a counter-example to the main conjecture, of which we prove that it is indeed a counter-example. We close off with chapter 4 by looking at a simplification of the game, where it is played on trees instead of any graph. Here we prove that the main conjecture does hold for a special family of trees and state a conjecture for the solution of any tree.

Contents

1 Introduction

1.1. Impartial Games

In this report we will take a look at impartial combinatorial games. These games can be characterized by the following list of requirements:

- There are two players with perfect information (there are no 'hidden' moves and no randomness is involved).
- The players alternate making moves.
- There is a set of positions or configurations the game can be in, each of which has a number of possible configurations that can be reached by making a move.
- The set of possible moves in a given configuration does not depend on whose turn it is.
- The game takes a finite number of moves to finish.
- The game is finished when the current configuration has no possible moves. A winner is then decided.
- We will require the games we look at to be under the *normal play rule*, which means that the last player who made a legal move is the winner.

When the fourth requirement is not met (e.g. chess, where player 1 can only move white pieces and player 2 can only move black pieces), we instead call it a partizan game. In this report we will refer to player 1 as Alice and player 2 as Bob.

We will introduce some often-used notation using Nim as an example game. Nim is a game played with a number of stacks of coins. Alice and Bob

alternate making a move, consisting of picking one of the stacks and removing any number of coins from it. The player who removes the last coin(s) wins. Suppose we have n of these stacks and each stack has x_i coins (with $1 \leq i \leq n$), we can then represent a configuration by the set $\{x_1, x_2, \ldots, x_n\}$. One thing we could notice, is that if we have a configuration of the form $\{x, 0, \ldots, 0\}$ (with $x \neq 0$), we know that the next player can win by taking all x coins from the first pile, bringing us to the position $\{0, 0, \ldots, 0\}$. From here there are no more legal moves and therefore we call this a *terminal* position. We will use the following result to further describe certain positions.

Theorem 1.1 (Zermelo)**.** *Given a finite two-person game of perfect information that cannot end in a draw and the players take turns alternatively with no randomness involved, then one of the two players must have a winning strategy.*

In other words, every possible position is either winning for the **N**ext player to make a move, or for the **P**revious player that made a move. We call these positions N-positions and P-positions respectively.

A natural question that arises is if we can somehow tell for any given position if it is an N-position or if it is a P-position. We already know that any terminal position is a P-position and that every position from which one can reach a terminal position is an N-position. From there it is not that difficult to generalize this to an algorithm. We take the following steps:

1. Label all terminal positions as P-positions.
2. Label any unlabeled positions that can reach a P-position in one move as N-positions.
3. Label any unlabeled positions that can only reach N-positions in one move as P-positions.
4. Repeat steps 2 and 3 until all positions are labelled

One way to make this process more visual, is by noting that every impartial game can be represented by an acyclic graph, where the vertices represent the different positions and the edges show which positions are reachable from a certain position. For example:

In this example we can see that vertices H and J are terminal positions, thus they will be labeled as P-positions. in step 2, we then label E, F and I as N-positions. In step 3 we label B and G as P-positions. We continue with step 2 and find that A, C and D are all N-positions. In Figure 1.2 we can see

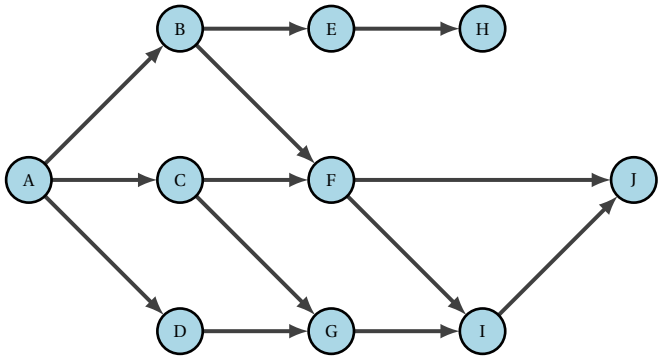


Figure 1.1: Acyclic representation of an impartial game

this represented by the coloring of the vertices. Since A is an N-position we now know that the game is winning for Alice, specifically by always moving to a P-position.

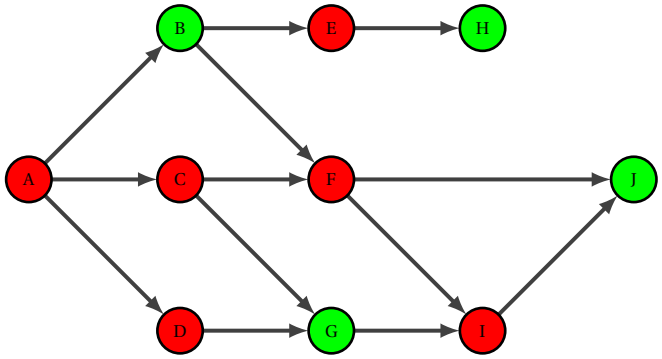


Figure 1.2: Green vertices represent P-positions, red vertices represent N-positions

For Nim we had seen that the position with 0 stacks is a P-position and any configuration with 1 stack is an N-position. What happens if we have two stacks? Suppose we have $\{1,1\}$, we then can only move to a situation with just one stack, which is an N-position, so two stacks of 1 must be a P-position. Trying some more small cases we could find a pattern rather quickly. Whenever the two stacks are of the same size, we have a P-position. Luckily it's not too difficult to prove this observation to be correct with induction.

Lemma 1.2. *In a two-stack game of Nim, if the two stacks both have c coins, we have a P-position.*

Proof. Suppose Alice makes any move on one of the two stacks, Bob will employ the copy-cat strategy and make the same move on the other stack. This ensures that Bob always has a move, and thus means Bob cannot lose. therefore this must be a P-position. □

It immediately follows that any position with two different stack sizes is

an N-position, as we can reach a P-position by making the stacks of equal size. If we look at this game for more than 2 piles it turns out that there is a really nice pattern to describe whether a configuration is a P-position or N-position. First we need to define the *nim-sum*.

Definition 1.3. *The nim-sum of two non-negative integers is their binary digital sum, meaning we first write both numbers in base 2 and then sum them without carrying over. We will write the nim-sum of x and y as $x \oplus y$.*

An example of this would be $15 \oplus 7 = (1111)_2 \oplus (0111)_2 = (1000)_2 = 8$. This nim-sum is used in the following result (Bouton, 1901-1902):

Theorem 1.4. *Given a position $\{x_1, x_2, \ldots, x_n\}$. This position is a P-position if and only if $x_1 \oplus x_2 \oplus \ldots \oplus x_n = 0$ (and an N-position otherwise)*

Let us take a look at the game $\{5, 7, 9\}$ as an example. The nim-sum is equal to

$$5 \oplus 7 \oplus 9 = (0101)_2 \oplus (0111)_2 \oplus (1001)_2 = (1011)_2 = 11 \neq 0,$$

so this is an N-position, but what is the winning move? We can deduce that by looking at $5 \oplus 7 = 2$ and therefore taking 7 coins from the third pile, reducing it to 2, so we get $5 \oplus 7 \oplus 2 = 0$.

1.2. Sprague-Grundy values

A very useful theory for analyzing impartial games is the Sprague-Grundy theory, which instead of describing configurations as just P-positions and N-positions, further subdivides N-positions, by giving all positions a non-negative integer as value, which we will call the Sprague-Grundy value of a configuration (or SG-value). We first introduce the following notion:

Definition 1.5. *The mex (**m**inimal **ex**cludant) of a set of numbers S is equal to*

$$\min\{n \geq 0 : n \notin S\}$$

Now we can properly describe

1. Assign the value 0 to all terminal positions.
2. For any position where the SG-value is known of all positions that can be reached in one move, we assign the mex of the reachable positions, to the current position
3. Repeat step 2 until all configurations have a value assigned.

Note that it is important that the game only takes a finite number of turns and thus has no loops.

Now that we can assign SG-values to all configurations of a game, we could notice that for the game of nim the SG-value of a configuration is exactly equal to the nim-sum of the stacks. A result that has to do with this is the Sprague-Grundy Theorem, which states that every impartial game under the normal play condition is equivalent to a one-heap game of nim, and thus as a natural number (the SG-value) and that these games can be added to each other (like adding multiple stacks of nim together for a larger game of nim). The SG-value of the sum of these games will then be equal to the nim-sum of the SG-values of these games.

Now that we have a basic understanding of impartial games, we will use this to analyze the Game of Cycles in the next chapters. But before we got there, we will take a look at a special type of impartial games in the next section.

1.3. Take-and-Break Games

Take-and-Break games are a type of impartial games where if we would visualize the game as a stack of coins (like Nim), a move consists of *taking* a coin from a pile and/or *breaking* a pile into two smaller piles. This is where Lasker's Nim comes in, named after Emanuel Lasker, the second world champion of chess. This version of Nim is different from standard Nim by giving the players the choice to either make a move as normal, or breaking a pile into two smaller (non-empty) piles instead. Now we would like to investigate who has a winning strategy in a given position using the Sprague-Grundy theorem introduced last section. We therefore start by looking at the SG-value of a single stack of coins.

If we have just a single coin, the only position we can reach is the terminal position, meaning a single coin has SG-value 1. We will denote this as $SG(1) = 1$. If we start with a stack of 2 coins, we can take one, both or split the pile into two single coins, those positions have SG-value 1, 0 and 0 respectively, meaning we must have $SG(2) = 2$. For a stack of 3 coins we can take any number of coins, giving SG-values up to 2, but we can also split the stack into a stack of 2 coins and a stack of 1 coin, giving SG-value $1 \oplus 2 = 3$, meaning $SG(3) = 4$. If we continue doing this for a while we find the following results:

It seems like we have a nice pattern to work with here, which we will immediately prove.

n	0	1	2	3	4	5	6	7	8	9	10
SG(n)	0	1	2	4	3	5	6	8	7	9	10

Theorem 1.6. *For every $k \geq 0$ we have $SG(4k+1) = 4k+1$, $SG(4k+2) = 4k+2$, $SG(4k+3) = 4k+4$ and $SG(4k+4) = 4k+3$ in Lasker's Nim.*

Proof. We will prove this by induction. We have already seen this claim to be true for $k = 0$. Now if we look at a stack of $4c+1$ coins, we can take any number of coins to reach positions of SG-values from 0 to $4c$. We can also split the stack in piles with size of the form $4a+4$ and $4b+1$ or $4a+2$ and $4b+3$. The nim-sum of these piles are in both cases even. This means that the SG-value $4c+1$ (which is odd) can never be reached, meaning that $SG(4c+1) = 4c+1$.

In the case of a stack with $4c+2$ coins, 0 to $4c+1$ can be reached. By breaking the pile we can reach piles with an odd SG-value or an SG-value divisible by 4. This means $4c+2$ never gets reached and we get $SG(4c+2) = 4c+2$.

Next up in the case of $4c+3$ we can reach 0 to $4c+2$ by taking coins and by splitting we can reach only odd SG-values, including $4c+3$ by splitting in a pile of $4c+2$ and a pile of 1 which means the smallest value we cannot reach is $4c+4$, meaning $SG(4c+3) = 4c+4$.

Lastly we can have a pile of $4c+4$ where we can reach 0 to $4c+2$ by taking coins as well as even values and values of the form $4a+1$ by breaking the stack, meaning $4c+3$ can never be reached, leaving us with $SG(4c+4) = 4c+3$ as $4c+3$ is the smallest value we cannot reach.

We conclude that our hypothesis is correct for all k. □

Another example of a Take-and-Break game is Grundy's game, which is played with a number of stacks of coins, where a move consists of splitting a stack of coins into 2 smaller stacks of unequal size. The game ends when only stacks of size 1 and 2 remain. We won't get into detail with this game right now, but it is recommended to the reader to have fun with it.

2

The Game of Cycles

The Game of Cycles is an impartial game invented by Francis Su (2020, p.51). In collaboration with others he wrote an article discussing some of the mathematical properties of the game. In this article the following was conjectured:

Conjecture 2.1 (Alvarado et al., 2020)**.** *A starting position of the Game of Cycles is winning for player 2 if and only if the number of edges is even.*

We will refer to this conjecture as Su's conjecture in this report. This chapter will first discuss the rules and game play of the Game of Cycles. Then in the second section some of the previous results will be discussed.

2.1. The rules of the Game of Cycles

The Game of Cycles is an impartial game played on a simple connected planar graph. A simple graph is a graph for which every edge has two distinct endpoints and is the only edge with these two endpoints. A graph is connected if for any two vertices we can walk from one to the other by some path consisting of one or more edges. Lastly, a graph is planar if we can draw it on the plane without any edges crossing each other. These constraints are further illustrated in Figure 2.1.

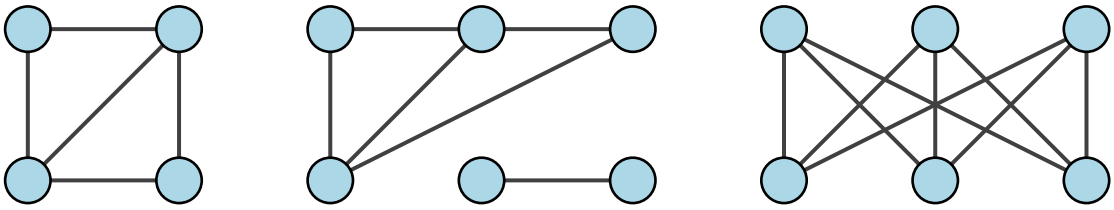


Figure 2.1: The graph on the left follows all the rules, the middle graph is disconnected and therefore not allowed and the graph on the right is not planar.

Now that we have properly defined our 'game board', we can continue by looking at how the game is played. Making a move consists of choosing a currently unmarked edge on the board and giving it a direction. These moves need to follow the so-called sink-source rule, which states that for any vertex, it is not allowed that all adjacent edges face away from the vertex or that all of them face towards the vertex, thus creating a 'source' or 'sink' respectively. An example can be seen in Figure 2.2.

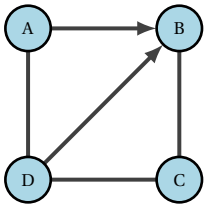


Figure 2.2: Playing ad or cb is not allowed as it would create a source on A and sink on B respectively.

The goal of the game is to complete a cycle cell, which immediately finishes the game and names the player who made the last move as the winner. It is also possible for a game to continue up till the point that no more moves are possible but no cycles have been made. In this case we also name the player who made the last move as the winner, following the normal play convention. These winning conditions are further illustrated in Figure 2.3.

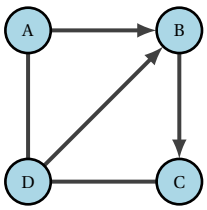


Figure 2.3: Here the current player can move cd to complete the cycle to win the game instantly!

2.1.1. Example case

In this section we will look at an example game on the board in Figure 2.4:

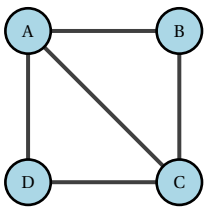
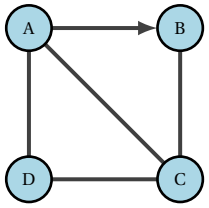


Figure 2.4: Example Game on the so-called 'diamond' graph

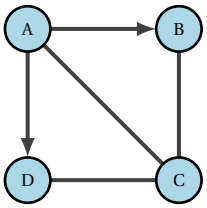
Now we would like to figure out if this example is a P-position or an N-position. Let us start with looking at what moves Alice can make. It looks like

Alice has 10 possible starting moves, which would be quite a lot to check. But luckily we can considerably reduce this amount by making 2 observations. First we can see that if we were to take any game board with some sides already played and invert all the marked arrows' direction, we end up with the exact same game. This holds because the head and tail of the arrows follow the exact same rules. The second observation has to do with graph isomorphism. We can see that $ABCD$ is isomorphic to $ADCB$, and $ABCD$ is also isomorphic to $CBAD$. Through this isomorphism we can see that the edge ab is equivalent to the edge ad through the first isomorphism and equivalent to cb through the second isomorphism. Lastly ab is also equivalent to cd. With these 2 observations combined we see that there is all moves on the sides of the square are equivalent and both diagonal moves are also equivalent, leaving us with only 2 unique moves. Alice can either make a move on the diagonal or on the side.

Suppose Alice starts by playing on the sides, marking ab. What will Bob do now?



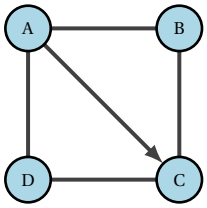
Bob now has 7 possible moves, both bc and ca are no good, because Alice would then be able to complete the cycle abc and win immediately. We call such a move that allows the other player to complete a cycle a *death-move*, as it results in your loss (assuming the other player completes the cycle). Bob decides to go with ad.



Alice can now only move dc, bc and ca, as the other unmarked moves result in a sink or source. But dc, bc and ca are all death-moves! So Bob can win on the next move regardless of what Alice does. So in this configuration we know that Bob has a winning strategy, which also means that Bob had a winning stategy after the first move of Alice. In that configuration, playing ad resulted in a winning position after all for player 2. In other words,

the first move Alice made was losing. But Alice could have picked another starting move!

Having learned from the last game, Alice decides to try her luck with a diagonal move this time around, playing ac:



Bob now has 8 moves to choose from. he immediately discards ba, cb, cd and da as those are all death-moves. This leaves us with ab, ad, bc and dc. Note that due to symmetries, the first two (ab and ad) are equivalent and the same goes for the second two. This leaves us with 2 options for Bob to play, without loss of generality: ab and dc, but if we use one of the observations we found earlier, namely inverting the directions of all arrows, we find that these two moves are actually also equivalent, meaning Bob only has one move! So Bob plays ab, something one may note is that if Alice now plays dc, there are no legal moves left because of the sink-source rule. This means that dc is a winning move for Alice, thus ab must have been a losing move for Bob. So what we see here is that regardless of the move Bob made after Alice played ac, Alice has a winning strategy. In other words, playing ac is winning, and thus Alice has an overall winning strategy on this board.

This example shows us a couple of things. One of these is that already at the very first move Alice had to make the correct moves to win, while one slip up was enough for Bob to take advantage and have a winning strategy for himself, which is typical for most impartial games. Another important observation this example showed is, is how much effort it takes to work out a game board with only 5 playable edges, and this gets exponentially larger as the number of edges increases. We definitely do not want to keep doing this by hand for larger game boards and therefore we will take an algorithmic approach in the next chapter. Before we go there, we will first look at results that have already been discovered by previous research into this game.

2.2. Previous results

Now that we understand the basics of the game and have some background knowledge on potential approaches to analyze the game, we will take a look at some of the previous research and results regarding this game.

2.2.1. Families of Boards

For some simple families of boards, such as lines or cycles the game has been shown to follow Su's conjecture with the addition that we only count markable edges. We make the distinction of *markable* edges for the case that there is a vertex with only one incident edge, making that edge unmarkable as it would turn the vertex into a sink or source right away. Here follow some of these previous results:

We denote a board consisting of a single cycle of n edges by C_c, we then have

Theorem 2.2 (Alvarado et al. 2020)**.** *The play on a C_n board is entirely determined by parity. If n is odd, Player 1 wins. If n is even, Player 2 wins.*

Theorem 2.3 (Lin, 2021)**.** *For a board that is a line L_n with n markable edges, Player 1 has a winning strategy if n is odd and Player 2 has a winning strategy if n is even.*

Another useful theorem is about the isomorphism we discussed earlier. This theorem makes use of *involutive symmetry*, which is when there is a non-trivial isomorphism of the board which is its own inverse. This isomorphism is then called an *involution*. This involution assigns each vertex, edge and cycle to a 'partner'. If the partner of a vertex, edge or cycle is itself, we call it *self-involutive*. We call a cell C *nowhere-involutive* if for every edge of C, its partner is not part C. We can now state the following theorem:

Theorem 2.4 (Alvarado et al. 2020)**.** *Let G be a board with an involution such that each cell is either self-involutive or nowhere-involutive. If there is no self-involutive edge, then Player 2 has a winning strategy. If there is exactly one self-involutive edge whose vertices are not fixed by the involution, then Player 1 has a winning strategy.*

Lastly the game has already been solved for 3-legged spiders of which all legs have an even number of markable edges:

Theorem 2.5 (Mathews. 2022)**.** *For any 3-legged spider for which the number of markable edges in each leg is even, we have that the empty board has SG-value* 0.

2.3. The line-graph

In this section we will analyze the line-graph and deduce its Sprague-Grundy value. Note that the line-graph is also covered in Theorem 2.4, but by working the line-graph out in the following way, we gain more insight into the

Sprague-Grundy values of certain boards, which Theorem 2.4 does not cover. The following results were also independently discovered by Mathews (2022).

Lemma 2.6. *For any L_n of which both outer edges are played, we have that it has Sprague-Grundy value $P(n)$ if both edges face the same direction and $1-P(n)$ if both edges face different directions.*

Proof. We will denote the line with n unmarked edges with the outer edges facing the opposite direction as O_n and the one with edges facing the same direction as S_n. We can quickly verify that $SG(O_1) = 1$ and $SG(S_1) = 0$. Now suppose the claim holds for all n up to $k-1$. If we make a move on O_k, that leaves us with O_a and S_b of which we can add the SG-values using the Sprague-Grundy theory, meaning the SG-value of O_k with one more move made is $SG(O_a) \oplus SG(S_b) = 1-P(a) \oplus P(b) = P(a+1) \oplus P(b)$ with $a+b = k-1$, so this equals $P(k)$, meaning that $SG(O_k) = 1-P(k)$. Now suppose we make a move on S_k, that leaves us with either S_a and S_b or with O_a and O_b. In either case we find that the SG-value of this new configuration is equal to $SG(S_a) \oplus SG(S_b) = SG(O_a) \oplus SG(O_b) = P(a) + P(b) = 1 - P(k)$, meaning that $SG(S_n) = P(k)$. By induction it now follows that the claim holds for all n. □

With the help of this lemma, we can prove the following theorem.

Theorem 2.7. *For any L_n of which only one outermost edge is played, we have that $SG(L_n) = n-1$*

Proof. For $n \leq 2$ the claim can easily be verified by hand. Now suppose the claim is true for all L_n with $n \leq k-1$. We will look at L_k. Suppose we make a move, this leaves us with L_a (with the outermost edge played), meaning it has SG-value $a-1$ and either O_b or S_b depending on what direction we faced our move. the SG-value of this new configuration is equal to $a-1 \oplus SG(O_b)$ or $a-1 \oplus SG(S_b)$, which equals $a-1$ or $a-1 \oplus 1$. Since we could have chosen any value for a smaller than $k-2$, this means we could reach any SG-value smaller than $k-2$. In the case we chose the $k-1$-th edge we could only face it in the same direction as the previously marked edge, giving us an SG-value of $k-2$ by the induction hypothesis. We conclude we can reach all SG-values up to and including $k-2$, meaning that our starting configuration must have SG-value $k-1$. By induction it now follows that the claim must hold for all n □

Now that we have proven this, the following result follows quickly

Corollary 2.8. *For any L_n with no played edges, we have that the Sprague-Grundy value equals the parity of n*

Proof. Suppose we play any move, it breaks up the line in two parts of lengths a and b with $a, b \geq 0$ and $a + b = n - 1$. By Theorem 2.7 we have that these parts have SG-value a and b respectively. Thus the SG-value of this configuration equals $a \oplus b$. Suppose n is even, then $P(a) = 1 - P(b)$, so $P(a \oplus b) = 1$. If n is odd we have $P(a \oplus b) = 0$ instead. So if n is even, we can only reach configurations with odd SG-value, meaning our original configuration must have SG-value $0 = P(n)$. If n is odd, the configurations we can reach all have even SG-values, meaning we cannot reach 1. By Theorem 2.4 we can always reach 0, meaning 1 is always the lowest value we cannot reach, meaning the SG-value of a line of odd length is 1, proving the claim. □

3

Acquiring data

As we have seen in the previous chapter, finding out if a particular board is winning or losing is a lengthy process, and while there are some tricks we can use to quickly gain results for specific cases, we still know very little about the general case. We could keep trying to look at more complex boards by hand, but this would take very long. Therefore, we are going to use the algorithmic approach outlined in chapter 1 and write a Python script to do the rough work for us.

3.1. Algorithmic approach

In chapter 1 we saw the following algorithm for finding the Sprague-Grundy value of a particular board:

1. Assign the value 0 to all terminal positions.
2. For any position where the SG-value is known of all positions that can be reached in one move, we assign the lowest value that is not reachable from here. This is referred to as the mex (**m**inimal **ex**cludant) of the positions reachable in one move.
3. Repeat step 2 until all configurations have a value assigned.

Using this principle in a python script allows us to gain data on larger boards. We do have to adjust the rules of our game slightly for the addition of multiple games using the Sprague-Grundy theory to make sense. Currently if we find ourselves with a disconnected board (either because it started that way, or because all edges connecting the part have been marked) we cannot use the SG-sum because of the current win-condition. If we complete a cycle on one of the sub-boards the game is immediately over on all sub-boards. An easy way to circumvent this is by making death-moves illegal

moves. Since these moves are always losing, no matter the circumstance, it does not interfere with any winning strategies. It does however make sure the game is never won by completion of a cycle, and thus we can add games together using the Sprague-Grundy theory.

We will now look at the specifics of the used Python script, which can be found in the appendices. The approach starts with a given game board, which is represented by its vertices, edges and cycle cells. The algorithm then proceeds to play, storing all intermediate positions in a memory list. When a terminal position is reached, it gets the value 0 assigned which is stored in a list of the position from which the terminal position was reached. The terminal position is then removed from the memory and the algorithm proceeds with the next possible move from the deepest non-terminal position it had reached. Once all possible moves of this position have been exhausted and thus added to the list of stored values, the code computes the mex of this list and assigns that value to the relevant position, which is then added to the list of the previous position, until all moves from there are exhausted and so on. This way all possible positions are evaluated and step-wise returned to the starting position.

3.2. Counter-examples to Su's conjecture

The first thing computed using this code was boards consisting solely of triangles. In this way the following counter-example to Su's conjecture was found.

3.2.1. The inverted hourglass graph

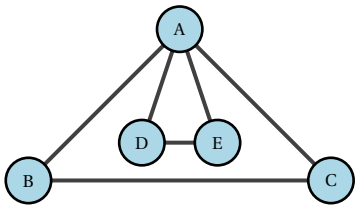


Figure 3.1: A counter-example to Su's conjecture

On this board player 1 has a winning strategy while the number of markable edges is even.

Proof. Alice will play the move ad. Bob can now either play on the outer triangle, or play ae (other moves on the inner triangle are illegal or death moves). In the case that Bob plays ae, Alice will reply with bc, forcing Bob to either play ab or ca after which Alice will play the other one leaving Bob

with no moves to play. Suppose Bob played on the outer triangle instead. Whichever move he chooses, Alice will reply by playing the move on the outer triangle such that the head of Bob's move is the tail of Alice's move. This way Bob only has 2 moves to choose from remaining. Whichever he picks, Alice will pick the last remaining move and thus win the game. □

Technically this board does not consist purely of triangles, but of a hexagon and a triangle, but since this hexagon uses all sides of the triangle it can never be finished before the triangle and is therefore irrelevant. We can 'repair' the conjecture by only allowing bi-connected graphs, that is connected graphs that remain connected if we take any one vertex and its incident edges away. This would get rid of constructions like the graph we just looked at.

3.2.2. The double-diamond graph

Continuing our search we found that for many different boards the conjecture holds. But unfortunately this new conjecture is also faulty, due to the following counter-example in Figure 3.2.

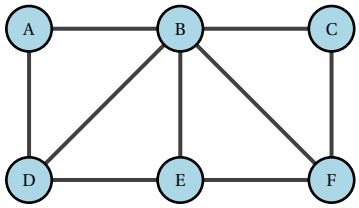


Figure 3.2: A bi-connected counter-example to Su's conjecture, we call this the double-diamond graph as it looks like two diamonds glued together

The above board has nine markable edges, but it turns out Bob has a winning strategy, which we will prove in the next section. Now that we have a proper counter-example to the conjecture, we would like to know why it does not hold and whether it is somehow repairable. Another proper counter-example can be found in Figure 3.3.

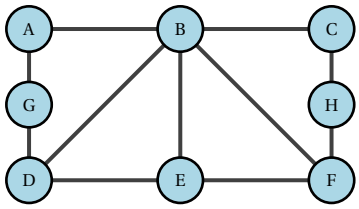


Figure 3.3: An extenstion to the last counter-example

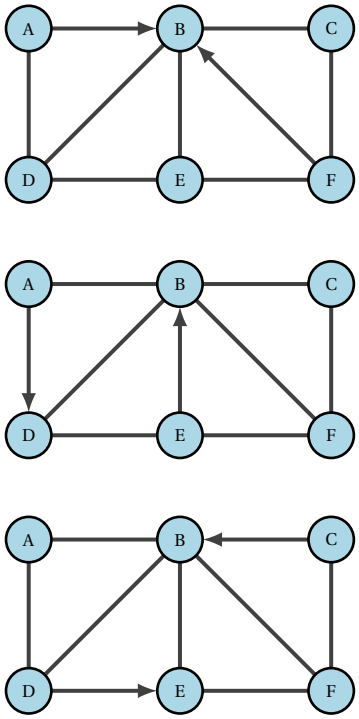
This graph is of course very similar to the earlier counter-example, only with a point on both the outer edges inserted. Unfortunately limitations

in computing power made it impossible to verify for larger graphs, but one could conjecture that any graph of this form (with n vertices inserted in both the outer edges) is a counter-example to Su's conjecture.

3.3. Proof that the double-diamond graph is a counter-example

We will prove that the game board in Figure 3.2 is winning for Bob by first noting that Bob can force the game to one of three unique positions after 2 moves, which we will then prove are winning for Bob.

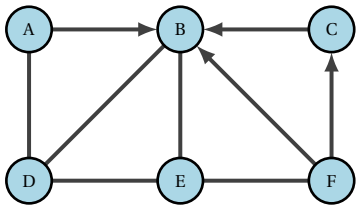
Lemma 3.1. *Independent of Alice's first move, Bob can force one of three positions illustrated below which we will call board A, B and C from top to bottom.*



Proof. This is easily verified by using the observations we used in section 2.1.1. through the first observation we made we note that for the first move the moves xy and yx are equivalent for all x and y. Through the second observation we also find that each move is equivalent with its mirrored partner due to the reflective symmetry through be. This reduces Alice's options for her first move to only ab, ad, bd, be and de. In the case of ab and bd Bob can play fb and bc to arrive in the first of the three positions (or an equivalent position) we want to be able to force. In the case of ad and be Bob can similarly force the second of the positions and in the case of de Bob can force the third. □

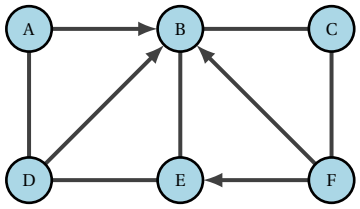
Now that Bob can always force the game to one of these three positions, we just need to prove that each of these is winning for him. We will start with the first of the three, with ab and fb played. In this position Alice has 13 legal moves, reduced to 7 when we eliminate death moves, which would obviously result in Bob's victory. These 7 moves are cb, db, de, eb, ed, fc and fe:

- cb. Bob replies with fc giving us the following position.



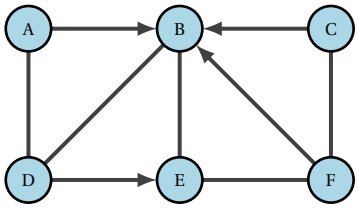
  Here Alice only has 4 legal non-death moves, being db, de, eb and ed. In the case of eb and ed, Bob can reply with the other move and the resulting position's legal moves are all death-moves making it winning for Bob. in the case of db and de, Bob can also reply with the other move, once again resulting in a position where only death moves are legal. We conclude cb was a losing move for Alice.

- db. Bob replies with fe giving us the following position.



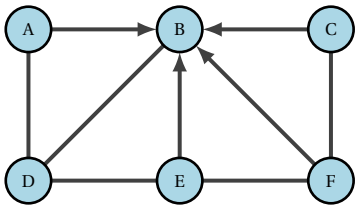
  Here Alice has 4 legal non-death moves, being cb, da, de and eb. If Alice plays either cb or de, Bob will react by playing the other move, instantly reaching a terminal position. If instead Alice plays da, Bob replies with cb resulting in a position where the 2 remaining legal moves are death moves. Lastly, if Alice plays eb Bob replies with de which also results in a position where the 2 remaining legal moves are death moves, confirming that db is a losing move for Alice

- de. Bob replies with cb giving us the following position.



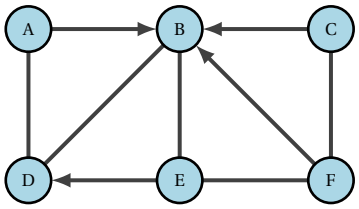
Alice has only 3 legal non-death moves in this position, namely db, fc and fe. If she plays fe or db Bob can reply with the other move instantly reaching a terminal position. If Alice plays fc instead Bob replies with db reaching a position with only death moves. We conclude de is losing for Alice as well.

- eb. Bob replies with cb leading us to the following position.



The only 4 legal non-death moves for Alice are ed, ef ,fc and fe. If she plays fc or ef, Bob will reply with the other move, leading to a position with only death moves. In the case that Alice plays ed or fe instead, Bob can reply by playing the other of the two moves resulting in another position with all legal moves being death moves.

- ed. Bob replies with cb giving us the following.



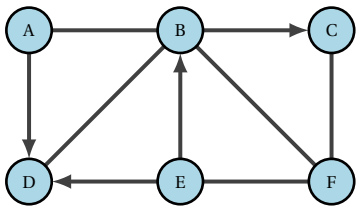
In this position, the only legal non-death moves for Alice are eb, fc and fe. In the case Alice plays eb or fc, Bob replies with the other and the resulting position only offers death moves to Alice. If Alice instead plays fe, Bob replies with eb resulting in a position with only death moves, meaning ed is also a losing move for Alice

- For fc and fe, Bob replies with cb and db respectively, giving us the first cases that we already showed were losing for Alice.

We can conclude that the position with only ab and fb is winning for Bob.

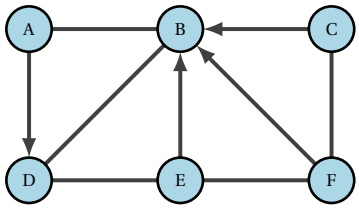
We continue by looking at the position with only ad and eb played. From here there are 13 legal moves, of which 7 are not death moves. These are bc, cb, cf, ed, ef, fb and fc. Now suppose Alice plays

- bc. Bob replies with ed, giving us the following.



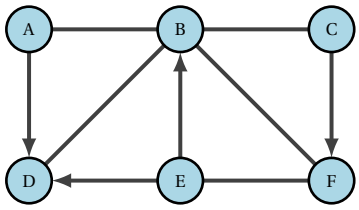
  In this position all legal moves for Alice are death moves, meaning this position is winning for Bob.

- cb. Bob replies with fb, bringing us to the following position.



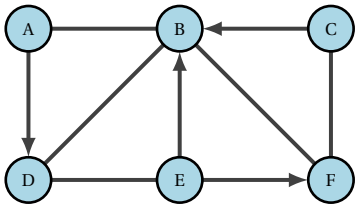
  In this position there are 4 non-death moves for Alice, being ed, ef, fc and fe. if Alice plays ef or fc, Bob replies by playing the other move resulting in a position with only death moves. If Alice instead played fe or ed, Bob can reply with the other move resulting in a position with only death moves.

- cf. Bob replies with ed, giving the following.



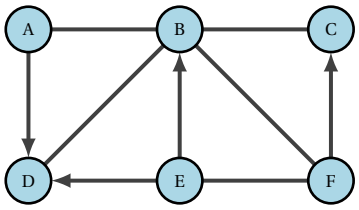
  Here all legal moves are death moves, meaning this position is winning for Bob

- ef. bob replies with cb, giving us the following.



In this position Alice only has one non-death move, being fb to which Bob can reply with fc, resulting in a position with only death moves.

- fc. Bob replies with ed, resulting in this position.

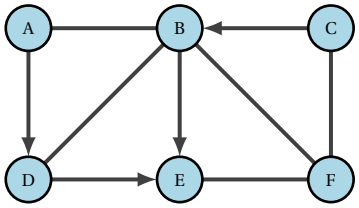


Here the only non-death move for Alice is fb, to which Bob can reply with cb, which results in a position with only death-moves, meaning Bob is winning after fc.

- For fb and ed Bob can reply with cb and bc respectively, resulting in two positions we have already seen are winning for Bob.

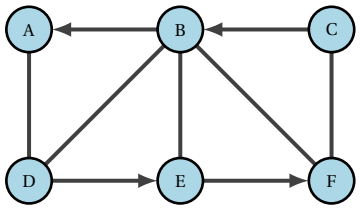
We can conclude that the position with just ad and eb played is winning for Bob. We now only need to check the position with only de and cb played. In that position Alice has 13 legal moves of which 9 are not death moves. These 9 moves are ab, ad, ba, be, da, db, ef, fb and fe. If Alice plays any of the following 5 moves ab, be, da, db and fb, Bob can make the moves fb, da, be, fb and ab respectively to arrive at one of the positions (or one equivalent to such a position) we have already shown to be winning for Bob. This leaves us with only ad, ba, ef and fe to check.

- ad. Bob replies with be, giving us the following position.



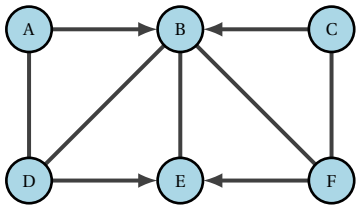
In this position there is only one non-death move for Alice, namely bd. Bob replies with ba resulting in a position with only death moves.

- ba or ef. Bob replies with the other, resulting in the following.



  In this position all moves are death moves meaning that this position is winning for Bob.

- fe. Bob replies with ab, giving us the following position.



  The only 2 non-death moves in this position are db and fb. When Alice plays one of these moves, Bob will reply with the other, resulting in a terminal position, ensuring Bob's victory.

We can conclude that in this case the game is also winning for Bob. Meaning in general that no matter how Alice plays, Bob will always have a winning strategy even though the number of markable edges in the starting position is odd. Meaning this is indeed a counter-example to Su's conjecture. Unfortunately the reason why this counter-example is a counter-example is not immediately clear. There were several attempts at finding some kind of invariant, such as the amount of markable edges at any given point. There also did not seem to be any connection between the number of vertices, edges and faces as well as their degrees and whether the starting position was winning or losing. All other boards consisting of solely 4 or less triangles do follow the conjecture.

4

Trees

In this chapter we will try to gain more insight in certain structures and aspects of the problem by looking at a special family of graphs. This family is called trees, which are graphs without cycles, which makes it look like a bunch of branches and leaves, hence the name. Vertices on the ends of the tree, meaning those with degree 1, are called leaves. Due to the absence of cycles in this chapter, the game is simpler to analyze and we conjecture that Su's conjecture does hold in the case of trees.

4.1. Z-Trees

One of the aspects that makes the Game of Cycles difficult to solve is the cycles. It may feel a bit odd to just disregard this part of the game as it is in the name of the game. Looking at boards without cycles does help by turning it into a Take-and-Break game however. At the end of chapter 1 we have already seen some of these games and how every move we make breaks the game into two smaller games. This means we can use the Sprague-Grundy theorem a lot here to sum these two smaller games and acquire the Sprague-Grundy value of the total game. In chapter 2 we already saw some previous results about lines and 3-legged spiders, which are some of the simplest trees there are. On the complete opposite side of a line which does not branch at all, we have trees that branch very often. We will now introduce a type of tree with properties we can take advantage of:

Definition 4.1. *A Z-tree is a tree for which all vertices of degree* 2 *are adjacent to a leaf*

Since we are working on trees, there are no cycles on our playing board and therefore the only rule that stops us from playing on until the entire

board is filled, is the sink-source rule. We will now prove a result involving Z-trees.

Theorem 4.2. *The Game of Cycles played on a Z-tree has a winning strategy for Bob if and only if the number of playable edges on the empty board is even.*

Proof. We shall first prove that if the number of playable edges is even, that Bob has a winning strategy. We start by choosing any vertex and considering this the parent node of the tree. The tree now has an ordering, so we can talk about parent and child vertices. Suppose every internal vertex of the Z-tree has exactly 2 child nodes. We call the edges connecting a parent vertex P with its children, the child edges of P. Suppose Alice plays any edge e with parent P. Bob will play the other child edge of P, in the opposite direction as e. This ensures that one of the two edges faces towards P and one away from P. Bob repeats this strategy for every move that Alice makes. Using this strategy we make sure that for any vertex, the 2 children face opposite directions and thus there can never be an edge unplayable because of the sink-source rule. This means that at the end of the game all moves have been played, such that Bob makes the last move (as there were an even number of playable edges), thus Bob wins, and therefore has a winning strategy.

Now suppose that not every vertex has exactly 2 children, but 2 or more. We will call a vertex *critical* if it has exactly 2 unmarked children left. Bob will apply the same strategy as above whenever Alice plays on a critical vertex. If Alice plays some edge of which the parent is not critical, Bob will also play a child of a non-critical vertex. This strategy always works since the amount of edges that can be marked on non-critical vertices is an even number (This is because the total number of playable edges is even and the number of edges of critical vertices is also even, meaning the difference also has to be even). We have now shown that this strategy ensures a winning strategy for Bob if the number of playable edges is even. Now suppose the number of playable edges is odd. Alice can now make a move on a non-critical vertex. After this move the number of playable edges left is even and it is Bob's turn. Alice can now copy Bob's winning strategy we described above to ensure that they win. Therefore Alice has a winning strategy in this case. We have thus proven that Bob has a winning strategy if and only if the number of playable edges is even. □

4.2. An attempt at solving the general case

When we looked at the Game of Cycles on a line in chapter 2, we could notice that all starting positions had SG-value 1 or 0, as well as that the reachable positions from the starting position all had the same parity as each other, which was the opposite parity of the number of markable edges in the starting position. One could conjecture this to be true for all trees. We state this as follows:

Conjecture 4.3. *Suppose T is a tree with n markable edges of which precisely* 1 *is marked, we then have*

$$P(SG(T)) = P(n-1)$$

We will try to prove this by a contradiction. Suppose there exists a counter-example to this claim, then there also exists a (possibly unique) *smallest* counter-example C (where smallest refers to the number of markable edges c in the starting position). Now if we take a look at this smallest counter-example we will deduce that its marked edge must be at one of the leaves.

If the marked edge is not adjacent to a leaf, it is instead adjacent to markable edges on either side. This means we have broken the game into two smaller games A and B with a and b remaining markable edges respectively. Since these are smaller we know that they satisfy the claim and thus that $P(SG(A)) = P(a)$ and $P(SG(B)) = P(b)$. Combining this gives us that $P(SG(C)) = P(SG(A) \oplus SG(B)) = P(P(SG(A)) + P(SG(B))) = P(a+b) = P(n-1)$. Which follows the claim, but since C was a counter-example, this is a contradiction. We thus see that the smallest counter-example to the claim must have its single marked edge at one of the leaves. This means that it would suffice to prove that the claim holds for all trees with only one of the leaves marked, since that would imply a smallest counter-example cannot exist. We conjecture this to be true, but it remains an open question for now.

5

Conclusion & Discussion

5.1. Conclusion

The main objective of this research was to either prove or disprove Su's conjecture. Thanks to the following counter-example that we found we have seen that the conjecture is false.

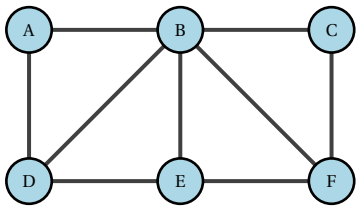


Figure 5.1: The double-diamond graph is a counter-example to Su's conjecture, because it has 9 markable edges, yet player 2 has a winning strategy.

With the main question answered, there were several other aspects of the Game of Cycles to be analyzed. Unfortunately, no good reason for why the found counter-example is a counter-example was found.
A simpler version of the Game was then analyzed where we only play on trees. For a specific family of trees introduced as Z-trees, we found that Su's conjecture does hold. Lastly an attempt at solving the Game of Cycles on any tree was made. We conjectured the following:

Suppose T is a tree with n markable edges of which precisely 1 is marked, we then have

$$P(SG(T)) = P(n-1)$$

Through this we reduced the general case to just having to prove the conjecture for trees where the singular marked edge is adjacent to a leaf.

5.2. Discussion

Despite coming across a few dead ends and having closed answers to other questions, there is still a lot to be said and done about the Game of Cycles. We list a few open questions below:

- Is there a nice reasoning as to why the double-diamond graph is a counter-example?
- Can we find more counter-examples for Su's conjecture? Is there any kind of pattern to these counter-examples?
- Is there some other way to easily check whether a position is winning or losing, now that Su's conjecture does not hold?
- Does adding more vertices to the outer edges of the double-diamond graph always give more counter-examples?
- Does the Game of Cycles played on trees follow Su's conjecture?

Some of these questions might be solvable relatively easily by improving on the Python script used in this research, or using better hardware. Having attempted a lot of different approaches towards solving the game, I personally think there is no easy or 'nice' way to check whether a given board is winning or losing. Lastly I think the game is solvable for trees and does follow Su's conjecture, but for now that remains to be seen.

Python script

```
    from copy import deepcopy

class Graph():
    def __init__(self, graph_dict = None, cycles = None):
        if graph_dict == None:
            graph_dict = {}
        self.graph_dict = graph_dict
        self.keys = self.graph_dict.keys()
        self.moves = []
        self.cycles = cycles
        for i in self.keys:
            keys = self.graph_dict[i].keys()
            for j in keys:
                self.moves.append(i+j)
        self.startNumOfMoves = len(self.moves)

    def direct(self, move):
        origin = move[0]
        destination = move[1]
        self.graph_dict[origin][destination] = 1
        self.graph_dict[destination][origin] = -1

        self.moves.remove(move)
        if destination+origin in self.moves:
            self.moves.remove(destination+origin)
        if sum(self.graph_dict[origin].values()) == len(self.graph_dict[origin]) - 1:
            for i in self.moves:
                if i[0] == origin:
                    self.moves.remove(i)
        if sum(self.graph_dict[destination].values()) == 1 - len(self.graph_dict[destination]):
            for i in self.moves:
                if i[1] == destination:
                    self.moves.remove(i)



    def check_cycles(self, move):
        for cycle in self.cycles:
            if move in cycle:
                tot = 0
                for i in range(len(cycle) -1):
                    tot += self.graph_dict[cycle[i]][cycle[i+1]]
                if  tot == len(cycle) -1:
                    return True

        return False

#####################################################################

memory = []
```

```
memory2 = {}
index = -1

def mex(lst):
    ordinal = [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25]
    for i in ordinal:
        if i in lst:
            pass
        else:
            return i

def is_legal(graph, move):
    for cycle in graph.cycles:
        if move in cycle:
            tot = 0
            for i in range(len(cycle)-1):
                tot += graph.graph_dict[cycle[i]][cycle[i+1]]
            if tot == len(cycle) -2:
                return False
    return True

def brute_force(g, c):
    test = Graph(g, c)
    memory.append([deepcopy(test.graph_dict), deepcopy(test.moves), 0, None, []])
    makemove(test)


def makemove(graph):
    while True:

        if graph.moves != [] and memory[-1][2] < len(memory[-1][1]):
            nextmove = memory[-1][1][memory[-1][2]]
            graph.direct(nextmove)
            memory[-1][2] += 1

            if is_legal(graph, nextmove):

                memory.append([deepcopy(graph.graph_dict), deepcopy(graph.moves), 0, None, []])

                if str(memory[-1][0]) in memory2.keys():
                    memory[-1][3] = memory2[str(memory[-1][0])]
                    graph.moves = []

                elif graph.check_cycles(nextmove):
                    graph.moves = []
                    memory[-1][3] = 0

            else:
                graph.graph_dict = deepcopy(memory[-1][0])
                graph.moves = deepcopy(memory[-1][1])

        elif len(memory) == 1:
            memory[-1][3] = mex(deepcopy(memory[-1][4]))
            print(memory)
            return
        else:
```

```
            if memory[-1][3] == None:
                memory[-1][3] = mex(deepcopy(memory[-1][4]))

            if memory[-2][3] == None:
                memory[-2][4].append(deepcopy(memory[-1][3]))

            memory2[str(memory[-1][0])] = memory[-1][3]

            #if len(memory) == 3 and memory[-1][3] == 0:
            #    print(memory[-1])
            memory.remove(memory[-1])
            graph.graph_dict = deepcopy(memory[-1][0])
            graph.moves = deepcopy(memory[-1][1])

brute_force(gcounter2, ccounter2)
```

Bibliography

Alvarado et al., (2020). *The Game of Cycles*. https://doi.org/10.48550/arXiv.2004.00776

Bouton, C.L. (1901-1902). Nim, A Game with a Complete Mathematical Theory. *Annals of Mathematics, 3*, 35-39.

Lasker, E., (1931). *Brettspiele der völker*. August Scherl.

Lin, K. (2021). *Exploring Winning Strategies for the Game of Cycles* [Master's thesis, Harvey Mudd College]. `https://scholarship.claremont.edu/cgi/viewcontent.cgi?article=1251&context=hmc_theses`

Mathews, B.G., (2022). *The Game of Arrows on 3-legged spider graphs*. https://doi.org/10.48550/arXiv.2110.08738

Su, F. E. (2020). *Mathematics for Human Flourishing*. Yale University Press.