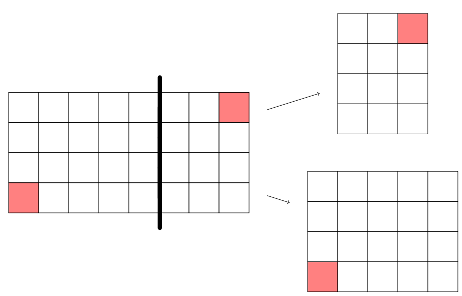
An Analysis of the Game of DiviNim

Exploring combinatorial game theory through an extension of the game of Bad Chocolate

Daniel Kinsey



University of Ottawa — CSI 4900
Supervisor: Professor Jean-Lou De Carufel
Revised: December 17, 2024

Abstract

The game of “Bad Chocolate” is put forth by James Tanton in his book “Solve This”[1]. The game involves a chocolate bar made of a grid of squares, where one square is poisoned. Players take turns slicing either horizontally or vertically across the whole bar until some player is left with the poisoned square.

In this report, we will examine an extension of this game, in which there are multiple poisoned squares. Named *DiviNim*, for its relation to the game of Nim and the division of the chocolate bar, we will analyze the game's properties and find optimal strategies for various configurations of the game.

Contents

Chapter 1

Introduction

> The Lord giveth and the Lord [nimm]eth away
>
> *Job 1:21*

1.1 DiviNim

In his book “Solve This”[1], James Tanton describes the game of *Bad Chocolate*. The game is presented like so.

> Dan and James are presented with a rectangular 4×8 chocolate bar with score marks for breaking it into 32 individual square pieces. They note that the bottom right square of chocolate is spoiled and cannot be eaten.
>
> These gentlemen decide to play the following game: Dan will break the bar along one entire score line, hand the piece containing the bad square to James, and place the remaining piece aside. James will then break the piece handed to him into two sections, again along an entire score line, and hand the portion containing the bad square to Dan, placing the other piece aside; and so on. They will do this until someone is handed a lone square of bad chocolate. That person will then be declared the loser and will keep only the single rotten piece of chocolate, while the other person gets all the rest to enjoy.
>
> If you were to play this game, what strategy would you employ? [1]

For this research project, we examine a generalized form of the game, one where any (positive non-zero) number of arbitrary unit squares on the bar can be spoiled / poisonous. If a bar has been split in two, and each one contains at least one poisonous square, the following player can take a turn on any bar of their choosing. Any bar without a single poisoned square is discarded, as in the original game. This set of rules shall be referred to in general as the game of *DiviNim* going forward. We will examine two particular interpretations of the game, each with a different *win-condition*.

- *Last-One-Standing* / *Last DiviNim* — The game is played until a single player is left with a single square. The player who is handed the last square, and thus has no moves available to them, is the loser. This is very similar to other impartial games, such as Nim, chomp, hackenbush, etc.
- *Score-Keeping* / *Scored DiviNim*— The game is played until a single player is left with a single square. Over the course of a game, any player who is handed a poisoned square is “poisoned” and increments their score by one. The player with the lowest score at the end of the game is the winner.

For this project we focused on Scored DiviNim, since it is comparable to other games while the addition of a score makes the game more complex and interesting. Last DiviNim is also interesting, but it is more similar to other games such as Chomp, and thus is less unique.

An example of a DiviNim turn is shown in figure 1.1.

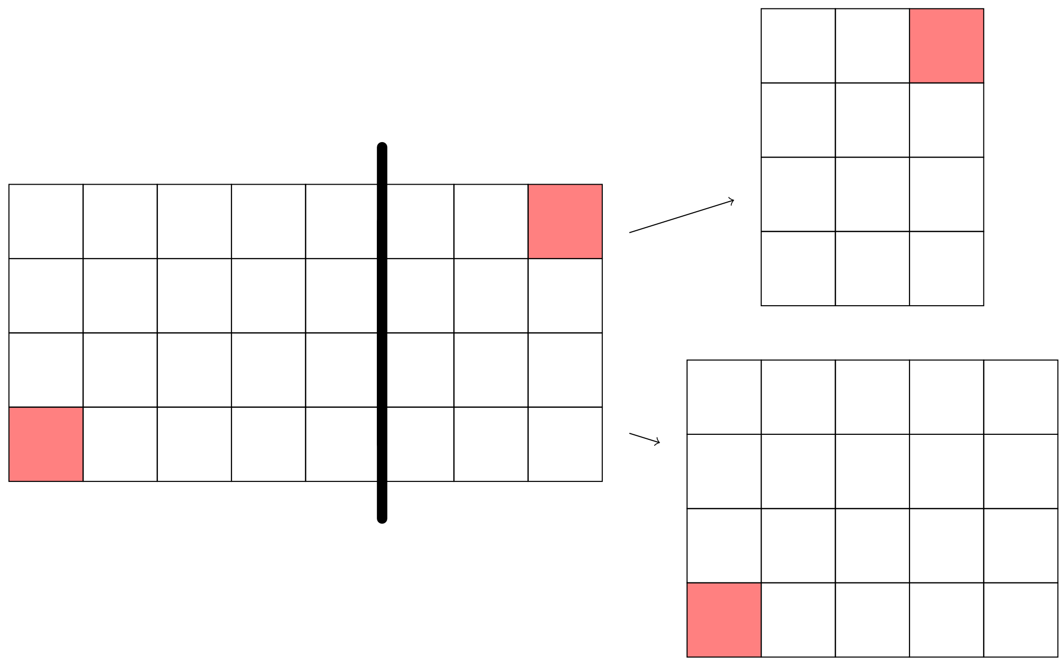


Figure 1.1: An example of a DiviNim turn. The first player slices vertically between two poisoned pieces, creating two bars for the next player.

So in summary, a *game* of *DiviNim* has one or more *bars*, each bar having a number of *squares* arranged in a grid. Some of these squares are *poisoned*. A *turn* consists of a player slicing a bar into two pieces along a single line, and discarding any piece that contains no poisoned squares. The game is played until a single player is left with a single square. The set of *possible moves* for a player is the set of all possible slices that can be made on any bar in the game.

1.2 Preliminary Knowledge

1.2.1 Introduction to Combinatorial Game Theory

Over the course of the 20th century, mathematicians developed the field of *combinatorial game theory*, which studies games as mathematical objects. In combinatorial game theory, a game is defined as a set of possible *positions* or *game states*, and a set of possible *moves* that can be made from each position. In this way, a game can be thought of as a directed graph, with the positions as nodes and the moves as edges. A game then is a *combination* of its child games, begetting the name combinatorial game theory.

In this section, we briefly introduce the concepts of combinatorial game theory, as far as they are necessary to understand the game of DiviNim. Ideas will be presented in an unfolding manner, mostly following the chronological development of the field. We will be sacrificing some formality when presenting these ideas for the sake of understandability. For a more formal treatment, we recommend *On Numbers and Games* by John Conway [2], which cemented many of the ideas presented here.

Nim

It is helpful to start with the game of Nim, perhaps the simplest combinatorial game. Nim is a game with many variants, but the form most commonly known is simple: two players take turns taking ("nimming") objects from a set. The set is arranged in rows, or "heaps", and a player can take any non-zero amount of objects, but only from a single heap. The last player to take an object is the winner (or sometimes the loser, depending on the variant). An example of a game of Nim is shown in figure 1.2. In this game, the first player takes two objects from the second heap, then player two takes two objects from the third heap, player one takes one object from the first heap, and player two takes the last object from the third heap. Player two took the last object, so they are the winner.

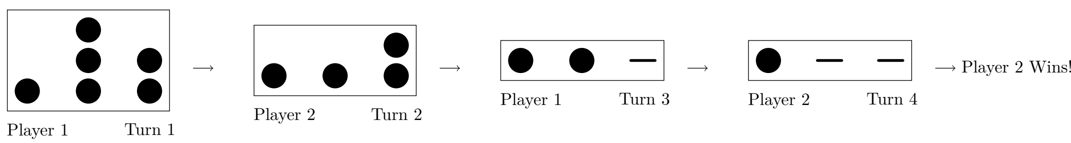


Figure 1.2: An example of a game of Nim.

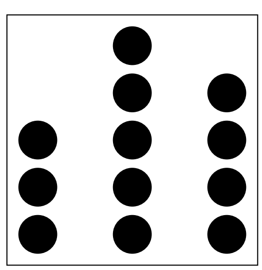
It can be seen that when playing the game with only a single heap, the strategy is simple. If the last player to take an object is the winner, then simply take all the objects on your turn. If the last player to take an object is the loser, then take all but one. The second player will be forced to take the singular remaining object, making them the loser. Thus, the game is only interesting when there are multiple heaps.

Nim-sums and Winning Positions

One of the first steps into combinatorial game theory was made by Charles Bouton in 1901 [3], when he formulated the strategy for the game of Nim. Bouton shows that, in the game of Nim, a position can be said to be a *winning position* if the bitwise XOR of the heap sizes is non-zero, and a *losing position* if it is zero. A game state / position is said to be a winning position if it is possible for the current player to force a win, and a losing position if the following player can force a win. We call the bitwise XOR of heap sizes the *Nim-sum*. This is a fundamental result in the theory of combinatorial games, often referred to as just Bouton's Theorem.

Theorem 1.2.1 (Bouton's Theorem)**.** If the Nim-sum of a game of Nim is zero, then the current player is in a losing position. Otherwise, they are in a winning position.

For example, the Nim-sum of the heaps in figure 1.3 is 2, since $3 \oplus 5 \oplus 4 = 2$.



$$\longrightarrow \quad \begin{array}{ccccccc} 3_{10} & \oplus & 5_{10} & \oplus & 4_{10} & = & 2_{10} \\ 11_2 & \oplus & 101_2 & \oplus & 100_2 & = & 10_2 \end{array}$$

Figure 1.3: An example of calculating game's Nim-sum (decimal and binary).

As a further demonstration of the result, consider the games in figure 1.4. The Nim-sum of each game is shown to the top left as $N(G)$. The games can be completed in a few turns, so one can try playing them out mentally and observe Bouton's theorem in action. Any game with a Nim-sum of zero cannot be won by the current player (against an optimal player). For any game with a non-zero Nim-sum, there exists a move that will allow the current player to win and simultaneously prevent the opponent from having any winning opportunity. That move will be to make the Nim-sum zero for the following player (putting them in a losing position).

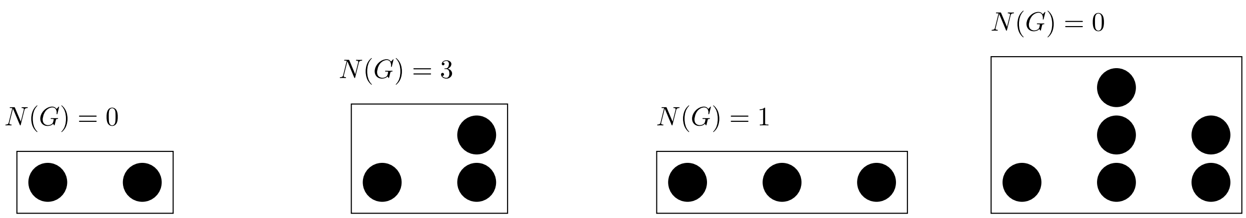


Figure 1.4: Examples of Nim games with their Nim-sums.

Impartiality

Bouton's theorem was expanded upon later, with the next major advancement coming from Roland Sprague and Patrick Michael Grundy, who independently released articles in 1935 [4] and 1939 [5] respectively that showed that every impartial game can be equated to a Nim game. This will be formalized shortly in the Sprague-Grundy theorem (theorem 1.2.2).

First, let us cover what impartiality means. The game of Nim is perhaps the quintessential example of an impartial game. An impartial game is a game with the following properties [2]:

Definition 1.2.1 (Impartial Game)**.** An impartial game is a game that satisfies the following properties.

- The game must involve two players, that strictly alternate turns.
- The game must be finished when no moves are possible.
- The game must have a finite number of operations that can be made.
- The two players must have the same set of moves available to them for each position.
- The game must not have any random elements.

The above definition is a simplification of the definition given in *On Numbers and Games* [**conway**]. The original definition formalizes the bullet points above further, but for the purposes of this paper, the simplified definition is sufficient.

We can see that Nim satisfies all of these properties. Other examples of impartial games include Chomp and Hackenbush. As an opposing example, the game of chess is not an impartial game, as the two players have different sets of moves available to them (one player moves white pieces, the other moves black).

This brings us to the Sprague-Grundy theorem, which states that every impartial game can be equated to a Nim game.

Theorem 1.2.2 (Sprague-Grundy Theorem)**.** Every impartial game is *equivalent* to a one-heap Nim game. We say that two games are equivalent if their outcomes are the same.

It is clear that the Nim-sum of a game of Nim with one heap is simply the number of objects in the heap, and we know that the Nim-sum determines the outcome of a game (Bouton's theorem, Theorem 1.2.1). A game with some Nim-sum will have the same outcome as any other with the same Nim-sum, no matter how many heaps it has. This is the esssece of the Sprague-Grundy theorem — a complex game can be thought of as a simple one (a game of Nim with one heap), and so as a single number, and the outcome of the game can be determined by that number.

To demonstrate this theorem, consider the games in figure 1.5. The Nim-sum for every game is zero, and so the games have the same outcome (a guaranteed win for the second player).

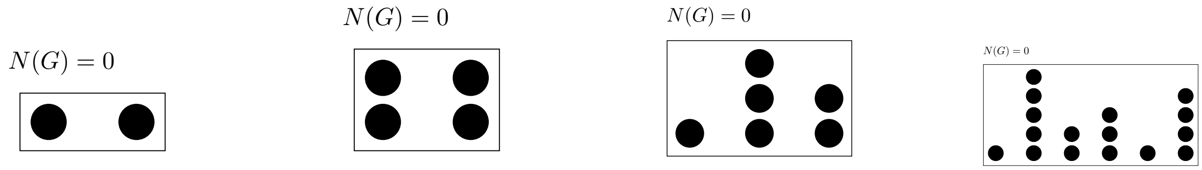


Figure 1.5: Examples of equivalent Nim games.

Numbers, Surreals, and Nimbers

Later in 1974, John Horton Conway and Donald Knuth introduced the concept of *surreal numbers* to represent game states for any combinatorial game [6]. This was later formalized in 1976 when Conway published *On Numbers and Games* [2], which expanded upon the work of Sprague and Grundy. In the same way that we can think of Nim-games via their Nim-sum, we can think of any combinatorial game in terms of its surreal number. In essence, combinatorial games can be thought of as a recursive structure of smaller games, and every game state can be represented by a *surreal* number. Surreal numbers are a superset of the reals, ordinals, and infinitesimals, and the way that a game state can be transformed into a surreal number can vary depending on the game's properties. The important necessary outcome is that the surreal numbers can be used to represent the game state, and then we can do operations on these numbers to determine the outcome of the game.

Nimbers are the subset of surreal numbers that represent impartial games. Specifically, the nimbers are a field defined on the ordinals, with the special new operations of nimber-addition and nimber-multiplication defined. In *On Numbers and Games*, Conway defines Nim-addition as the following [2].

Definition 1.2.2 (Nim-addition)**.** Nim-addition ($\oplus$) is defined as follows:

$$a \oplus b = \begin{cases} \operatorname{mex}\left(\left\{a' \oplus b \mid a' < a\right\} \cup \left\{a \oplus b' \mid b' < b\right\}\right) & \text{if } a, b \neq 0 \\ a & \text{if } b = 0 \\ b & \text{if } a = 0 \end{cases}$$

where a and b are ordinals and the function $\operatorname{mex}(A)$ is the minimum excluded value (the smallest ordinal not in the set) for a set of ordinals A .

As an example, let us consider the Nim-sum of 1 and 2. We have the following.

$$\begin{aligned} 1 \oplus 2 &= \operatorname{mex}\left(\left\{a' \oplus 2 \mid a' < 1\right\} \cup \left\{1 \oplus b' \mid b' < 2\right\}\right) \\ &= \operatorname{mex}\left(\{0 \oplus 2\} \cup \{1 \oplus 1, 1 \oplus 0\}\right) \\ &= \operatorname{mex}\left(\{2\} \cup \left\{\operatorname{mex}\left(\{0 \oplus 1, 1 \oplus 0\}\right), 1\right\}\right) \\ &= \operatorname{mex}\left(\{2\} \cup \left\{\operatorname{mex}\left(\{1\}\right), 1\right\}\right) \\ &= \operatorname{mex}\left(\{2\} \cup \{0, 1\}\right) \\ &= \operatorname{mex}\left(\{0, 1, 2\}\right) \\ &= 3 \end{aligned}$$

The way to think about this definition in plain, non-mathematical terms is that the combined nimber of two games is dependent on the nimbers of every possible game state you would get from a turn that could be made in either game. The set $\left\{a' \oplus b \mid a' < a\right\}$ represents the nimbers of all possible game states that could be reached by taking objects from the first game, and $\{a \oplus b' \mid b' < b\}$ represents the nimbers of all possible game states that could be reached by taking objects from the second game.

Nimber multiplication is defined in a similar way, but is not necessary for the purposes of this paper.

As an aside, it is worth noting that Nim-addition is commutative, associative, non-distributive, and (intuitively) has an identity element of 0. This means that nimbers can be added in any order or grouping, and that adding 0 to a nimber will not change the nimber. Tying everything together, we have the following definition.

Definition 1.2.3 (Nimbers)**.** The nimbers are an Abelian group defined on the ordinals with Nim-addition and Nim-multiplication as the group operations, 0 as the identity element, and all elements being their own additive inverse.

For further reading, see *On Numbers and Games*, where this group is identified using the term On_2 [2].

Nimbers are Nim-sums

For most games, even the definition of nimbers given above (definition 1.2.3) is more complex than necessary. Games generally don't involve infinities, and so using the subset of natural numbers ($\mathcal{N} = \{0, 1, 2, 3, \ldots\}$) in place of the ordinals in our definition is sufficient for our purposes. For this subset of nimbers, it can be seen that the previous analysis of Nim games by Bouton (Theorem 1.2.1) fits in perfectly — that is, Nim-addition is equivalent to the bitwise XOR. We can see that the amount of possible turns on a heap of size n is simply $n - 1$, since a player must take at least one. So the minmal excluded value in the set of resulting game state will be n. So, intuitively, the nimber of a game of Nim is the size of the heap if it is a single heap, and the Nim-sum of the nimbers of the heaps if there are multiple heaps. This perfectly mirrors the original theorization which used the bitwise XOR operation.

So, we can adapt Bouton's theorem (Theorem 1.2.1), and say that a game is *Nim-losing* if its nimber is zero ($N(G) = 0$), and *Nim-winning* if its nimber is non-zero ($N(G) \neq 0$). Existing literature often uses the terms $\mathcal{N}$-position for Nim-winning positions — positions where the *next* person to play has a guaranteed win — and $\mathcal{P}$-position for Nim-losing positions — positions where the *previous* person to play has a guaranteed win. For the sake of not confusing the *next* player and the *following* player, we will use the terms *Nim-winning* and *Nim-losing*, and a position can be a $\mathcal{W}$-position or $\mathcal{L}$-position respectively.

We can rework the nimber addition definition (definition **??**) to define a general way to calculate nimbers for impartial games. Instead of being an operation on two ordinals, we shall define a function on a game state that calculates the nimber of that game state. This recursive definition is essentially the same as the original definition, but illustrates the concept in a more intuitive way.

Definition 1.2.4 (Recursive Definition of Nimbers)**.** The nimber of an impartial game $N(G)$ is defined recursively as follows.

$$N(G) = \begin{cases} 0 & \text{G is finished} \\ \operatorname{mex}\Big(\big\{N(G') \mid G' \text{ is a possible child game state from G}\big\}\Big) & \text{otherwise} \end{cases}$$

If we were to attempt to calculate a nimber using this recursive definition (definition 1.2.4), we would end up calculating a nimber for every possible move (child game state), and every possible subsequent move, and so forth. This leads to a factorial running time, so ideally we want to find a way to calculate the nimber of a game state without having to recurse whenever possible. Of course, for games of Nim, we know from Bouton's theorem (theorem 1.2.1) that we can take the bitwise XOR to get a game's nimber, and from Sprague-Grundy theorem (Theorem 1.2.2) we know that any impartial game is reducible to a Nim game. So, for impartial games, we can calculate the nimber without having to recurse, so long as we can find a way to map the game to Nim. The goal in our analysis of DiviNim will be to find a way to calculate the nimber of DiviNim in a similar way — cutting down on the number of recursive calls we need to make.

1.2.2 The Minimax Algorithm

The minimax algorithm (first devised by John von Neumann in 1928 in German [7], and later in English [8]) is an algorithm that can determine the optimal move in a game. In actuality, it finds the move that corresponds with the best *guaranteed* outcome, according to some heuristic function. In essence, it finds the optimal move for the current player, assuming that the opposing player will also make the optimal move. The algorithm can be used in many places, not just combinatorial games — for instance it is used often in artificial intelligence for decision making.

Suppose we have a heuristic function $H(G)$, that takes a *completed* game state and outputs a number representing how desirable that is for the current player. For DiviNim, and in fact most games, we can define it like so.

$$H(G) = \begin{cases} 1 & \text{player 1 has won} \\ -1 & \text{player 1 has lost} \\ 0 & \text{player 1 has tied} \end{cases}$$

Then, we can define the minimax algorithm with the pseudocode in algorithm 1. At each recursive call, the algorithm will alternate between maximizing and minimizing the heuristic function, until it reaches a terminal state.

Algorithm 1 Minimax algorithm

```
function MINIMAX(G, depth, maximizingPlayer)
    if depth = 0 or G is terminal then
        return H(G)
    if maximizingPlayer then
        v ← −∞
        for child of G do
            v ← max(v, MINIMAX(child, depth − 1, false))
        return v
    else
        v ← ∞
        for child of G do
            v ← min(v, MINIMAX(child, depth − 1, true))
        return v
```

It can be helpful to think of the minimax algorithm as an algorithm that finds a path on the game's tree. Recall that we say the root of the tree is the current game state, and each child node is the result of a possible move that can be made. The algorithm finds a path from the root to a leaf node that has the best possible outcome for the player and simultaneously gives the opponent the fewest possible chances to stop it. This is shown in figure 1.6. In the figure, the highlighted path represents one possible optimal strategy for the current player, the one found by minimax. Green arrows represent the first player's maximizing choices, the red arrows represent the second player's [attempted] minimizing choices. Notice that the algorithm selected a move that not only leads to a win, but also leaves no way for the opposing player to win. Also note that after the first turn, the game is already decided — the Nim-sum of the heaps is zero ($1 \oplus 1 = 0$). These facts are why Nim-losing and Nim-winning positions are so important in the game of Nim, and by extension, in impartial games in general.

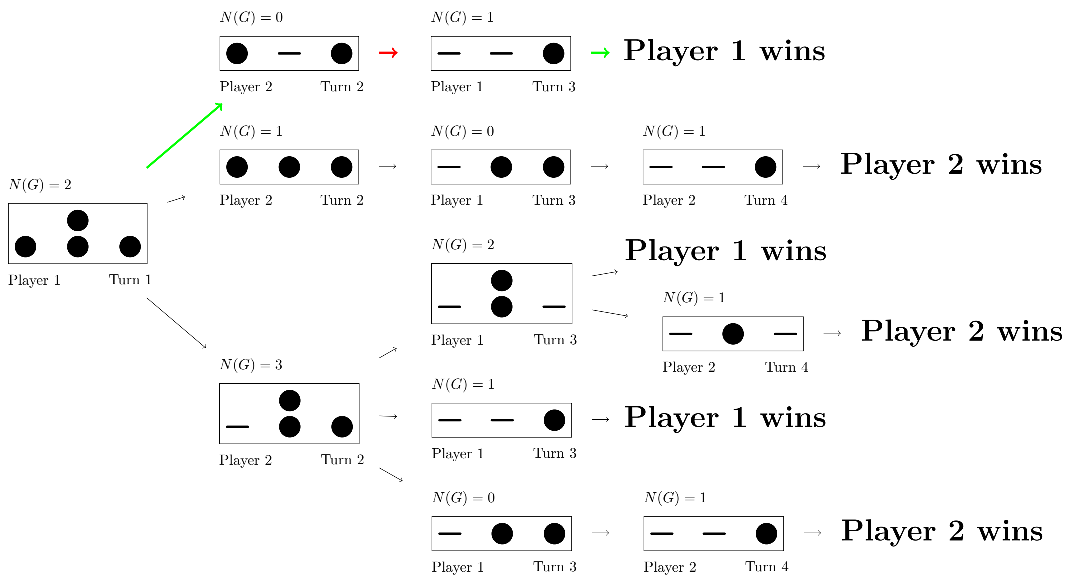


Figure 1.6: A tree representing the possible moves in a game of Nim.

Chapter 2

DiviNim

Worry is the stomach's worst *poison.*

Alfred Nobel

2.1 General Analysis

We will begin by examining the general properties of DiviNim, no matter the win condition or setup.

2.1.1 Impartiality

Theorem 2.1.1. DiviNim is an impartial game.

Proof. To be an impartial game, a game must satisfy five properties:

- The game must involve two players, that strictly alternate turns.
- The game must be finished when no moves are possible.
- The game must have a finite number of operations that can be made.
- The two players must have the same set of moves available to them for each position.
- The game must not have any random elements.

We can see that DiviNim satisfies all of these properties. The game has two players, who alternate turns. The game is finished when no moves are possible (when the last remaining bar is a single square). The dimensions of the bar determine the amount of moves that can be made, are finite, and decrease with each move. The players have the same set of moves available to them, and the game has no random elements. □

Since DiviNim is impartial (Theorem 2.1.1) we can use the minimax algorithm (Algorithm 1) to analyze it. However, the runtime will be exceptionally large. Given a bar of size $m \times n$, there are $m + n - 2$ possible turns that a player can take, and so there are $m + n - 2$ recursions to do. At the next call, there will be $m + n - 3$, and so forth, giving a factorial structure. Given a max depth d and a single poisoned square, we can say then that the running time is $\mathcal{O}\left(\frac{(m+n)!}{(m+n-d)!}\right)$. Note that to fully examine the game state, reaching every possible move, the value of d must be $d = m + n$, giving a simple, yet exceedingly large, runtime of $\mathcal{O}\big((m + n)!\big)$. If there are multiple poisoned squares, the running time will be even longer, as a split can cause a second bar to be playable, with its own dimensions and possible moves. In the worst case, every single square is poisoned, and so a slice never removes any un-poisoned squares from play. Poisoned squares are only removed from play when the dimensions reach 1×1 for a given bar, which is our base case. So for games with multiple squares, the runtime is $\mathcal{O}\Big((m+n)^{(m+n)}\Big)$, which is unwieldy to say the least. This illustrates why a defined optimal strategy for DiviNim (that is easier to compute) is desirable. For example, running the algorithm on the original version of Bad Chocolate (an 8×4 grid with one poisoned square) will mean a number of recursions near $12! = 479,001,600$. A lot of the nodes in this hypothetical recursion tree are redundant, as a position can be reached in multiple ways, so caching and pruning the tree is essential to use the minimax algorithm effectively. The size of the recursion tree makes it impractical to run the algorithm on larger games, but it can be used as a simple demonstration of any optimal strategy we define.

2.1.2 Equivalences

To start with, it can be seen that for games of DiviNim with one poisoned square the game win-condition does not matter.

Theorem 2.1.2. For a game of DiviNim with a single poisoned square, the win-condition does not matter.

Proof. For games of DiviNim with a single poisoned square, the game is finished when the last square is played. For Last DiviNim, the player who plays the last turn is the winner. For Scored DiviNim, the player who is handed the last square is poisoned and increments their score by one, making the score of the game one to zero, and the player who plays the last turn is the winner just the same. So, both win-conditions will have the same outcome. □

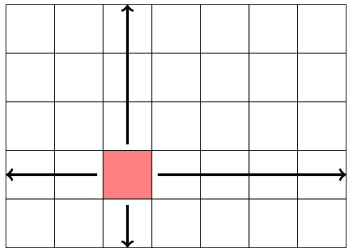
Going forward, we will refer to games of DiviNim with a single poisoned square as *1-square games*. Similarly, we will refer to games of DiviNim with two poisoned squares as *2-square games*, and so on.

We can notice something else about 1-square games of DiviNim: the game is equivalent to a game of Nim with four heaps. Intuitively, this comes from the observation that slicing is an equivalent operation to nimming. We can treat the squares in each direction away from a poisoned square as a Nim heap. Since a slice can only be on one side of a poisoned square, a slice is guaranteed to only take from one "heap" / direction away from the square. This brings us to the following theorem.

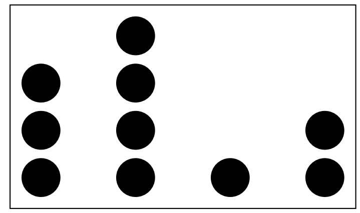
Theorem 2.1.3. A 1-square bar of DiviNim is equivalent to a game of Nim with four heaps.

Proof. Slicing is an operation that works on one axis of the bar. This means that for any given bar, a slice will be on only one side of the poisoned square. Thus, we can treat a slice as an operation equivalent to a Nim move on a single heap. The heaps themselves will be the number of squares in each direction from the poisoned square to the edge of the bar. The amount taken from the heap is the amount of squares between the slice and the edge of the bar in the heap's direction. A slice in DiviNim can only ever be on exactly one side of a poisoned square, so the directional heaps are independent. So, we can construct a game of Nim with four heaps, each one equivalent to the number of squares in each direction from the poisoned square to the edge of the bar. The heaps / directions will be fully independent, satisfying the conditions for a game of Nim. □

For example, consider the games in Figure 2.1. Figure 2.1a is a 1-square game of DiviNim, and Figure 2.1b is a game of Nim with four heaps. The arrows show the what the heaps represent in the game of Nim — the number of squares in each direction from the poisoned square to the edge of the bar. It is important to note that if there are multiple poisoned squares then this mapping does not work, because the directions from one poisoned square to the edge of the bar will overlap at some point. This will mean the heaps are no longer independent, and so the game is not directly equivalent to a game of Nim.



(a) A 5×7 chocolate bar with a single poisoned square.



(b) A game of Nim with four heaps.

Figure 2.1: A 1-square game of DiviNim and its equivalent Nim game.

2.2 Scored DiviNim

2.2.1 1-Corner Games

To begin, we will analyze the simplest case of DiviNim: a chocolate bar with a single spoiled square in a corner. After all, the original game of Bad Chocolate, as described by James Tanton in his book "Solve This"[1], was set up this way. Going forward, we will refer to games of DiviNim with a single spoiled square in a corner as *1-corner games*. Further, games with two spoiled squares in corners will be referred to as *2-corner games*, and so on. Take a look at the bar in figure 2.2.

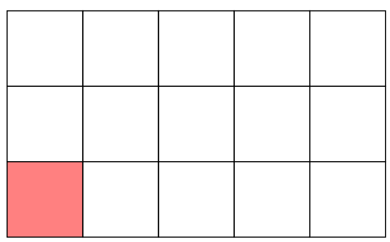


Figure 2.2: A 3×5 chocolate bar with a single poisoned square in a corner.

Now, notice what happens when the bar becomes a square, after some number of turns (Figure 2.3). Using the Nim-representation given in theorem 2.1.3 the Nim-sum of the poisoned squares in the bar is now $2 \oplus 2 = 0$

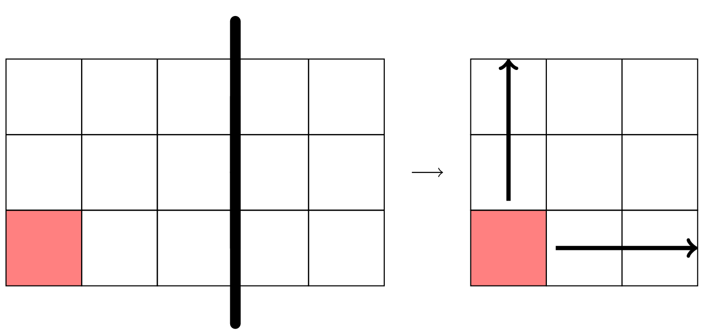


Figure 2.3: A 3×5 chocolate bar split into a square.

The Nim-sum for the poisoned square is easy to check. The amount of squares from the poisoned square to the edge is going to be the same in both directions — since it's a square. Since Nim-addition (on games of independent components) is done with the bitwise XOR operation, the values cancel out. This gives us the following theorems, which allow us to define a winning and losing position for a 1-corner game of DiviNim.

Theorem 2.2.1. A game of DiviNim with a single 1-square bar with a Nim sum of 0 is a guranteed losing position for the current player.

Proof. Follows naturally from theorems 2.1.3 and 1.2.1. □

Theorem 2.2.2. A game of DiviNim with a single 1-corner bar is in a losing position for the current player if and only if the bar is a square.

Proof. First, we prove that a square bar is a losing position. A square bar necessarily has dimensions $n \times n$, where n is a positive integer. The poisoned square is in a corner, so it has only two sides for which the nimber is 0. The other two sides have nimbers of $n - 1$, which cancel each other out when Nim-added. Thus, the Nim-sum must be 0. By theorem 2.2.1, this is a losing position.

Next, we prove that any losing position, where there is one poisoned square which is in a corner, must be a square. Since the bar is Nim-losing, we know the Nim-sum is 0 (Theorem 2.2.1). Since the poisoned square is in a corner, we know that at least two sides of the square have nimbers of 0. This leaves us with two sides, and since the Nim-sum is 0, the nimbers on these sides must cancel each other out. The additive inverse of a nimber is itself, so they must be the same. Thus, the side lengths must be equal, and so the bar is a square. □

We can now see that a 1-square bar can be in one of three states:

- Finished ($\mathcal{F}$) — a finished game / bar that is a single poisoned square. A Nim-sum of 0, with the nimbers in all directions being 0.
- Nim-losing ($\mathcal{L}$) — a bar with a Nim-sum of 0, with at least one side having a non-zero nimber. A guaranteed loss for the player who has to play on it, even with optimal play (theorem 2.2.1).
- Nim-winning ($\mathcal{W}$) — a bar with a Nim-sum greater than 0. A guaranteed win for the player who plays on it, assuming they use optimal play (Theorem 2.2.1).

We can then determine what transitions between these states are possible. This will allow us to create a strategy for DiviNim that mirrors Bouton's strategy for Nim.

Theorem 2.2.3. A bar in a Nim-losing position can always be sliced into a Nim-winning position, and never into another Nim-losing position or a finished state.

Proof. To begin, we can see that a Nim-losing position can never be sliced into a finished state. A Nim-losing position is defined as a position where the Nim-sum is 0 and the game is not finished. This means that the heap sizes cancel each other out, and so there must be multiple heaps that are non-zero. Any slice can only affect the bar on one side of the poisoned square, and there are multiple directions away from the poisoned square with non-zero nimbers, so the game cannot be finished by a single slice.

To show that a Nim-losing position cannot be sliced into another Nim-losing position, we can consider the Nim-sum of the bar. By theorem 2.1.3, the Nim-sum of the bar is

$$N = 0 = n_1 \oplus n_2 \oplus n_3 \oplus n_4$$

for the four directions away from the poisoned square, n_1, n_2, n_3, n_4. Order does not matter, so let us choose n_1 to be the part of the bar that is sliced off. The Nim-sum of the new bar is then

$$N' = n_1' \oplus n_2 \oplus n_3 \oplus n_4, \ n_1' < n_1$$

Recall that any number XOR'd with itself is 0, anything XOR'd with 0 is itself, and XORing is a non-distributive operation, so we can find that the following is true:

$$\begin{aligned} N' &= (n_1' \oplus n_1 \oplus n_1) \oplus n_2 \oplus n_3 \oplus n_4 \\ &= n_1' \oplus n_1 \oplus (n_1 \oplus n_2 \oplus n_3 \oplus n_4) \\ &= n_1' \oplus n_1 \oplus 0 \\ &= n_1' \oplus n_1 \end{aligned}$$

Since by definition $n_1' < n_1$, the XOR of n_1' and n_1 will be non-zero. Thus, the Nim-sum of the new bar will be non-zero, and the bar will not be in a Nim-losing position. □

This theorem has a simpler formulation for the case where the poisoned square is in a corner. We have the following corollary.

Corollary 2.2.3.1. A square 1-corner bar is a Nim-losing position that can always be sliced into a Nim-winning position, and never into a Nim-losing position or a finished state.

Proof. We know from Theorem 2.2.2 that a Nim-losing position for a bar with a single poisoned square in a corner can only be a square. If the square is of dimensions $n \times n$ then we can see that any cut made will result in the bar no longer being a square, instead being $n \times n'$ with $n' < n$. Thus, the bar will no longer be a square, and so it will no longer be Nim-losing. □

We can also consider the converse of theorem 2.2.3, the transitions possible from a Nim-winning position.

Theorem 2.2.4. A bar in a Nim-winning position can always be won immediately, or be sliced into a Nim-losing position.

Proof. Similarly to Theorem 2.2.3, we can consider the Nim-sum of the bar. By Theorem 2.1.3, the Nim-sum of the bar is

$$N = n_1 \oplus n_2 \oplus n_3 \oplus n_4$$

where n_1, n_2, n_3, n_4 are the amount of bars between the poisoned square and the bar edge in each direction, and the Nim-sum $N > 0$. We can then consider two particular cases:

- If only one nimber is non-zero, then the bar extends in only one direction away from the poisoned sqare, and so the bar square can be won in a single move.
- If at least two nimbers are non-zero, then the bar extends in at least two directions away from the poisoned square. We would like to show that we can always find a slice that will result in a Nim-losing position.

  Since the Nim-sum is non-zero, we know that there will be at least one bit in the sum that is 1. Let us pick the leftmost bit that is 1, and call its position k. Since the final sum has a 1 in this position, we know that either one or three of the nimbers have a 1 in this position as well (if two or four of the nimbers had a 1 in position k, then the final sum would have a 0 in that position). Let us select one of the nimbers that has a 1 in position k, and call it n_1. We can then construct the new value of n_1 to be n_1' such that $n_1' = n_1 \oplus N$. Another way to think about this is that we are going through every bit in n_1 and flipping the bit if the corresponding bit in N is 1. Since the bit at position k is the leftmost 1 (and all 1s will be flipped), we know that $n_1' < 2^k \leq n_1 < 2^{k+1}$, and so it is possible to slice n_1 to n_1'. The new Nim-sum of the bar will then be $N' = n_1' \oplus n_2 \oplus n_3 \oplus n_4$.

$$\begin{aligned} N' &= n_1' \oplus n_2 \oplus n_3 \oplus n_4 \\ &= (n_1 \oplus N) \oplus n_2 \oplus n_3 \oplus n_4 \\ &= N \oplus (n_1 \oplus n_2 \oplus n_3 \oplus n_4) \end{aligned}$$

$$= N \oplus N$$
$$= 0$$

Thus, the Nim-sum of the new bar will be 0 after performing the slice on n_1 to n_1', and the bar will be in a Nim-losing position.

□

This theorem also has a simpler formulation for the case where the poisoned square is in a corner. We have the following corollary.

Corollary 2.2.4.1. A 1-corner bar in a Nim-winning position can always be won immediately, or be sliced into a Nim-losing position.

Proof. For a bar of dimensions $n \times m$ with $n \leq m$, we can examine two simple cases for the value of n: $n = 1$ and $n > 1$. If $n = 1$, the bar can always be sliced into a finished state — the poisoned square is in a corner and, since $n = 1$, has only one side to slice from — simply slice at the poisoned square. If $n > 1$, the bar can always be sliced into a bar with square dimensions (which is a Nim-losing position by theorem 2.2.2). We know this because for any set of dimensions with $n < m$ there will exist some $m' < m$ such that $m' = n$. Thus, the height and widths will be the same after the slice, and the Nim-sum will be 0. □

With theorems 2.2.3 and 2.2.4 in mind, we can construct a state diagram for 1-square games as shown in figure 2.4. The state of the diagram represents whether there can be guaranteed a loss or win for the *current* player. Edges are turns that lead from the current state to the state that the following player will inherit. There is always at least one state change possible from the Nim-winning position, and there are always two if the bar dimensions are greater than two. The if-statements in the diagram represent when the move is available to the player. Recall that by definition we have $n \leq m$.

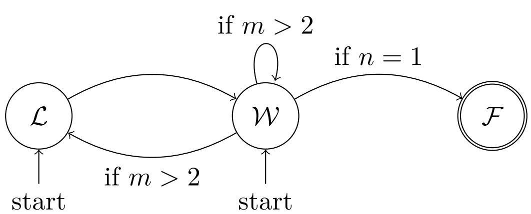


Figure 2.4: State diagram showing possible moves for a 1-square game of DiviNim with dimensions $n \times m$, $n < m$.

From the state diagram (Figure 2.4), we can see that there is a simple optimal strategy. A Nim-losing bar has no way to win, and a Nim-winning bar can always either win immediately or put the following player in a losing position. Thus, the optimal strategy is to always cut a Nim-winning bar into a Nim-losing or finished position, and to forefit when in a Nim-losing position. Although we devised this strategy using the 1-corner setup, we can see that it is generalizable to any 1-square bar of DiviNim.

Theorem 2.2.5 (1-Square Optimal Strategy)**.** For a 1-square game of DiviNim with dimensions $n \times m$, $n \leq m$, a strategy to guarantee a win for the current player exists if and only if the current position is Nim-winning. The strategy is as follows:

1.) If the bar is Nim-winning:

   a.) If $n = 1$, cut the bar into a finished position.

   b.) If $n > 1$, cut the bar into a Nim-losing position.

2.) If the bar is Nim-losing, play any arbitrary move or forefit.

Proof. By Theorem 2.2.3, a Nim-losing bar can only be sliced into a Nim-winning position, and never won. By theorem 2.2.4, a Nim-winning bar can always be sliced into a Nim-losing position or a finished position. So, a player in a Nim-winning position can always either win immediately or put the following player in a losing position, then the following player will have no choice but to put the original player back into the same position they were in before, and the logic repeats. Thus, the strategy is optimal. □

Empirically, we can verify the correctness of the state diagram and strategy using the minimax algorithm (Algorithm 1). For example, consider the game in Figure 2.5. The game starts in a Nim-winning position, and the highlighted path shows the optimal moves for the current player. Note that equivalent sibling nodes (i.e. nodes that are rotations of each other) are combined into a single node. Notice that the path found by the minimax algorithm involves slicing the bar into a square (Nim-losing) position any time it is possible to do so.

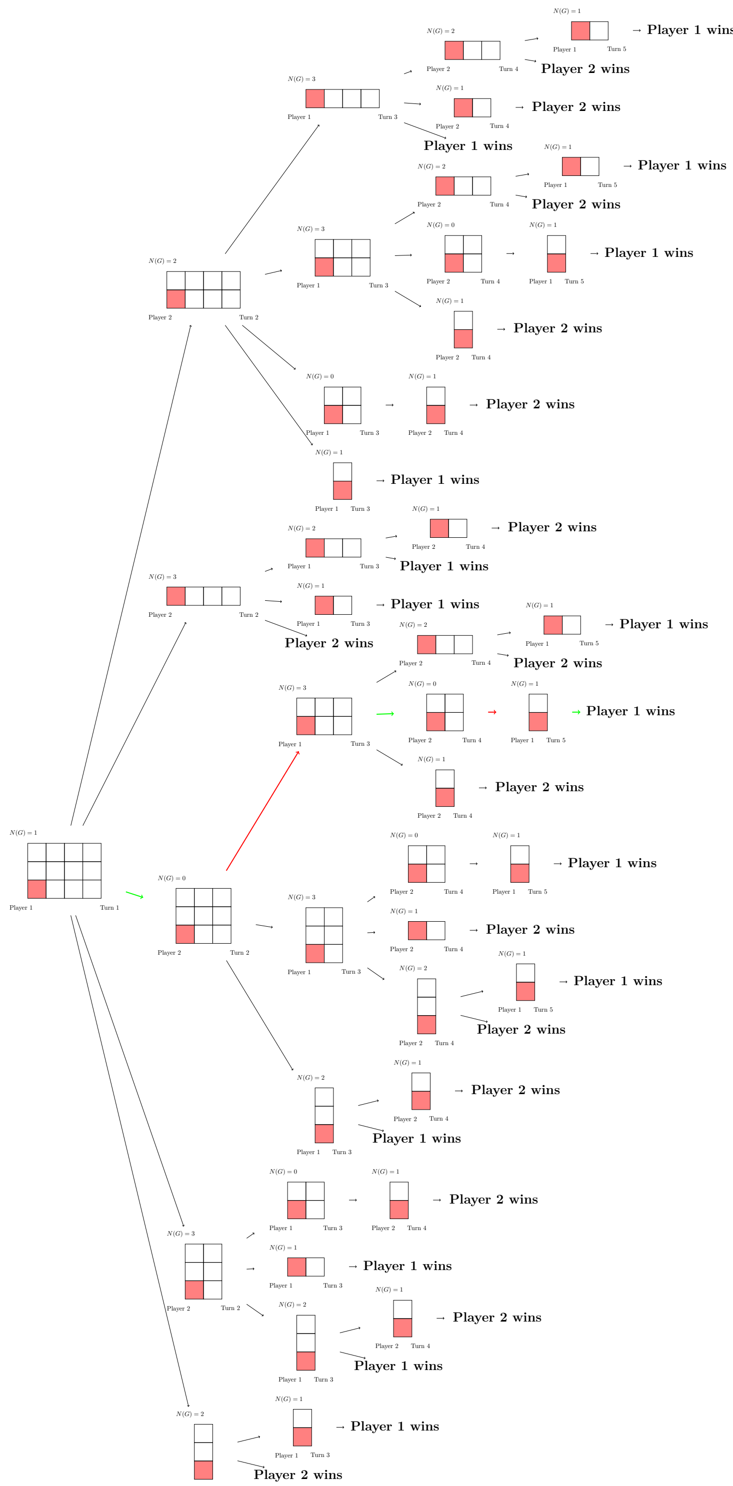


Figure 2.5: The tree resulting from the minimax algorithm for a 3×4 1-corner game.

2.2.2 Opposing 2-Corner Games

We will now examine how the game works when there are two poisoned squares in the bar, each in an opposite corner.

We can see that, after any single move, the bar will be split into two bars, each with a single poisoned square. Most of the previous theorems still apply — but take note of when "bar"is used within a theorem as opposed to "game".

Games with two poisoned squares in a corner can be thought of in terms of 1-corner games. We can see that since the poisoned squares are in opposite corners, any slice will cause the game to be split into two 1-corner bars after a single move. Thus, the states of the game can be thought of in terms of the states of the two bars. We have the following states for a 2-corner game:

- Finished ($\mathcal{F}$) — a finished game / two bars that are singular poisoned squares.
- Indeterminate ($\mathcal{I}$) — a game with a single bar with two poisoned squares.
- $2 \times$ Nim-losing ($\mathcal{LL}$) — a game with two bars, each with a Nim-sum of 0 and having at least one side with a non-zero nimber.
- $1 \times$ Nim-losing $+ 1 \times$ Nim-winning ($\mathcal{LW}$) — a game with one bar with a Nim-sum of 0 (and having at least one side with a non-zero nimber), and one bar with a Nim-sum greater than 0.
- $2 \times$ Nim-winning ($\mathcal{WW}$) — a game with two bars, each with a Nim-sum greater than 0.
- $1 \times$ Nim-winning $+ 1 \times$ Finished ($\mathcal{WF}$) — a game with one bar with a Nim-sum greater than 0, and one bar that is a single poisoned square.
- $1 \times$ Nim-losing $+ 1 \times$ Finished ($\mathcal{LF}$) — a game with one bar with a Nim-sum of 0 (and at least one side with a non-zero nimber), and one bar that is a single square.

We can now apply the analysis for 1-corner bars to 2-corner games. By theorem 2.2.3, we know that any Nim-losing bar can only be sliced into a Nim-winning bar. So, we know that a $\mathcal{LL}$ state can only be sliced into a $\mathcal{LW}$ state. By theorem 2.2.4, we know that any Nim-winning bar can always be sliced into a Nim-losing bar or a finished state. So, we know that a $\mathcal{LW}$ state can always be sliced into a $\mathcal{LL}$ state for the following player to inherit. After some player goes from a winning state to a finished state for that bar, the remainder of the game plays out as a 1-corner game. So, the goal for an optimal strategy for 2-corner games is to win one of the bars, and ensure that the other bar is in a Nim-losing state for the following player. This will result in the other player "starting" the 1-corner game in a losing position, giving the preceding player both wins.

So, from the above, we can see that an $\mathcal{LL}$ state can only be sliced into some $\mathcal{LW}$ state. A $\mathcal{LW}$ state can always be sliced into a $\mathcal{LL}$ or into a $\mathcal{LF}$ state. A $\mathcal{WW}$ state can always be sliced into a $\mathcal{WF}$ state or into a $\mathcal{LW}$ state. This all means that for optimal play, a $\mathcal{LL}$ state will always result in a loss, a $\mathcal{LW}$ state will always result in a win, and a WW state will always result in a tie.

Finally, we can see that the optimal strategy for a 2-corner game of DiviNim is given by Theorem 2.2.6.

Theorem 2.2.6 (2-Corner Opposing Optimal Strategy)**.** For a 2-corner game of DiviNim with starting dimensions $n \times m,\ n \leq m,\ n \geq 1,\ m \geq 2$, the optimal strategy is the following:

1.) If the state is indeterminate (one bar with two poisoned squares):

  a.) If the bar is of dimensions $n \times m$ with $m = 2 \cdot n$, slice the bar in half to create two $n \times n$ bars.

  b.) Otherwise:

    i.) If the bar is of dimensions $n \times m$ with $n = 1$, slice off a single square.

    ii.) Otherwise, slice the bar into two bars, one of size $m \times 1$, and one of size $(n-1) \times m$.

2.) If the state of the game is $2 \times$ Nim-losing, forefit or make any move.

3.) If the state of the game is $1 \times$ Nim-losing $+ 1 \times$ Nim-winning:

a.) If the Nim-winning bar is of dimensions $n \times m$ with $n = 1$, slice it into a finished state.

b.) Otherwise, slice the Nim-winning bar (dimensions $n \times m$) to create a square ($n \times n$) bar.

4.) If the state of the game is $2 \times$ Nim-winning:

a.) If either bar is of dimensions $n \times m$ with $n = 1$, slice that bar into a finished state.

b.) Otherwise, slice either bar into dimensions $n \times n - 1$.

5.) If the state of the game is $1 \times$ Nim-winning $+ 1 \times$ Finished or $1 \times$ Nim-losing $+ 1 \times$ Finished, play optimally for a 1-corner game of DiviNim as described in theorem 2.2.5.

Proof. There are multiple cases to consider. For each case, we will consider the state of the game, and the possible moves that can be made. We will then show that the suggested move is always possible, and that it will result in a particular state for the following player. At the end of the proof, we will have shown that every case has a known behaviour, and that following the suggested move will always result in a win or a tie for the current player.

We shall consider the cases in a different order than they are presented in the theorem. In this way, successive cases will build upon the previous cases.

- If the game state is $1 \times$ Nim-winning $+ 1 \times$ Finished or $1 \times$ Nim-losing $+ 1 \times$ Finished the game is equivalent to a 1-corner game of DiviNim. The optimal strategy for these games is described in Theorem 2.2.5.

- If the game state is $2 \times$ Nim-losing, any move made will result in a $1 \times$ Nim-losing $+ 1 \times$ Nim-winning state for the following player. We know this because by theorem 2.2.3, a Nim-losing bar can only be sliced into a Nim-winning bar, and never won.

- If the game state is $1 \times$ Nim-losing $+ 1 \times$ Nim-winning and the Nim-winning bar is of dimensions $n \times 1$, then the move given in part 3.a is optimal. We know that $m \geq 2$, so the Nim-winning bar can always be sliced into a finished (1×1 bar) state, giving us one point. This will result in a $1 \times$ Nim-losing $+ 1 \times$ Finished state for the following player, which we know means another point. This gives us both points, and so a win overall.

- If the game state is $1 \times$ Nim-losing $+ 1 \times$ Nim-winning and the Nim-winning bar is not of dimensions $n \times 1$, then the move given in part 3.b is optimal. We know that $n \geq 2$ and $n < m$, since the bar is Nim-winning and Nim-winning bars are not squares (Theorem 2.2.2). Thus, the Nim-winning bar can be sliced into a square ($n \times n$) bar. This will result in a $2 \times$ Nim-losing state for the following player, which we know means a win overall.

- If the game state is $2 \times$ Nim-winning and either bar is of dimensions $n \times 1$, then the move given in part 4.a is optimal. For the Nim-winning bar with the dimensions $n \times 1$, we know that $m \geq 2$, so the bar can always be sliced into a finished (1×1 bar) state, giving us one point. This will result in a $1 \times$ Nim-winning $+ 1 \times$ Finished state for the following player, which we know means the opponent will get a point. This gives us a tie overall.

- If the game state is $2 \times$ Nim-winning and neither bar is of dimensions $n \times 1$, then the move given in part 4.b is optimal. We know that for either bar, $n \geq 2$ and $n < m$, since the bar is Nim-winning and Nim-winning bars are not squares (theorem 2.2.2). Thus, the Nim-winning bar can be sliced into a bar of dimensions $n \times n - 1$. This will result in another $2 \times$ Nim-winning state for the following player. This means we will repeatedly stay in a $2 \times$ Nim-winning until $n = 1$ for some bar, at which point we can use the strategy given in the previous case, which we know means a tie overall.

- If the game state is indeterminate and the bar is of dimensions $m = 2 \cdot n$, then the move given in part 1.a is optimal. We know that the bar can be sliced in half to create two $n \times n$ bars. This will result in a $2 \times$ Nim-losing state for the following player, which we know means a win overall.

- If the game state is indeterminate and the bar is not of dimensions $m = 2 \cdot n$, and the bar is of dimensions $n = 1$, then the move given in part 1.b.i is optimal. If $m = 2$, then we have the trivial case where the entire chocolate bar is two squares, and we can win immediately in one slice. If $m > 2$, then we can slice off a single square, giving us one point. This will result in a $1 \times$ Nim-winning $+ 1 \times$ Finished state for the following player, which we know means the following player will gian a point. This gives us a tie overall.

- If the game state is indeterminate and the bar is not of dimensions $m = 2 \cdot n$, and the bar is of dimensions $n > 1$, then the move given in part 1.b.ii is optimal. We know that $n \geq 2$ and $n < m$, so the bar can be sliced into two bars, one of size $m \times 1$, and one of size $(n-1) \times m$. This will result in a $2 \times$ Nim-winning state for the following player, which we know means a tie overall.

Thus, we have shown that the optimal strategy for a 2-corner game of DiviNim can always guarantee a win or a tie for the current player. □

Empirically, we can verify the correctness of the optimal strategy for 2-corner games using the minimax algorithm (Algorithm 1). For example, consider the game in Figure 2.6. The game is of starting dimensions 2×4, the smallest non-trivial 2-corner game for which there exists a way to slice the bar into a $2 \times$ Nim-losing state. The highlighted path shows the optimal moves for the current player. Note that equivalent sibling nodes (i.e. nodes that are rotations of each other) are combined into a single node.

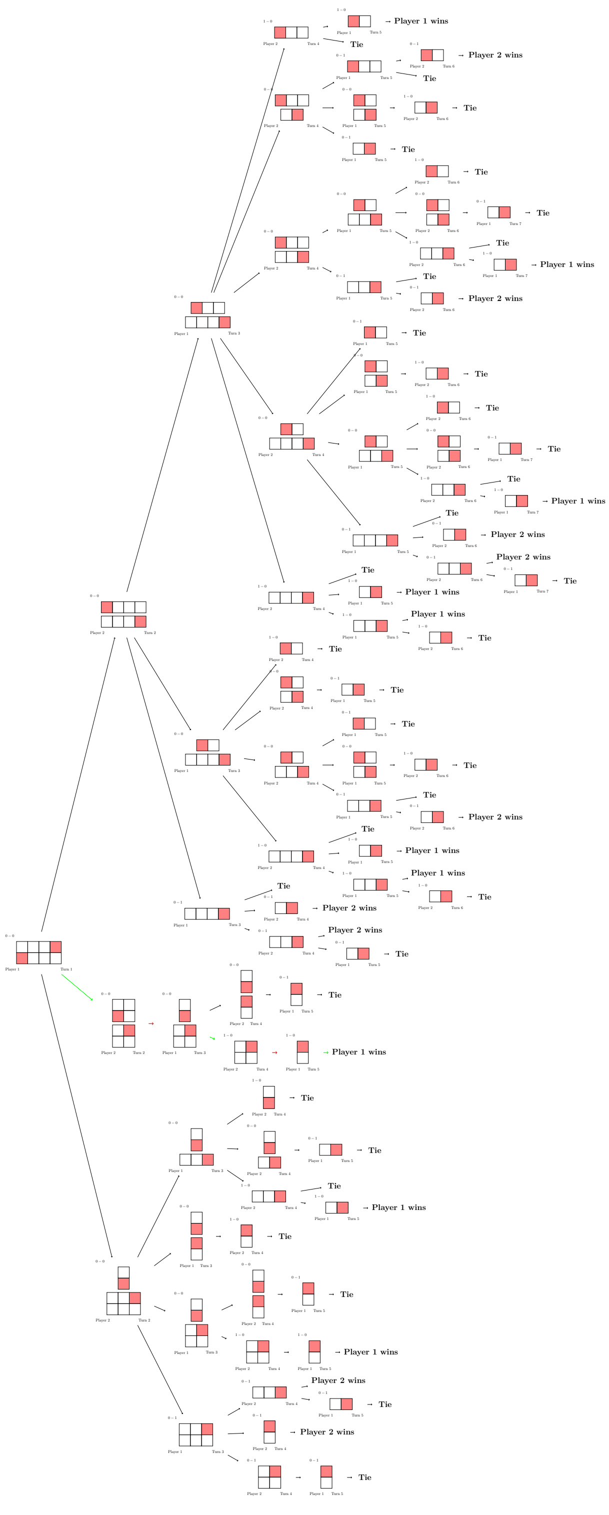


Figure 2.6: The tree resulting from the minimax algorithm for a 2×4 2-corner game.

2.2.3 2-Corner Games in General

In the previous section, we examined the case of 2-corner games where the poisoned squares are in opposite corners. The fact that the poisoned squares are in opposite corners means that any initial slice will result in two bars, each with a single poisoned square. This is not the case when the poisoned squares are in adjacent corners.

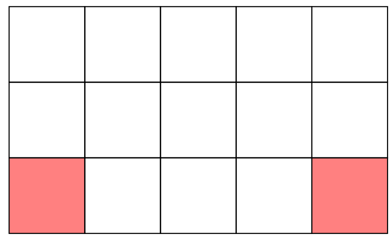


Figure 2.7: A 3×5 chocolate bar with two poisoned squares in adjacent corners.

The goal is still to create two bars that are Nim-losing if possible, and to avoid giving the opponent two Nim-winning bars. The question now becomes: “When is it appropriate to perform a slice that makes the bar into two bars, each with a single poisoned square?” To attempt to answer this question, we will examine the possible outcomes of each way the bar can be sliced.

Let the dimensions of the bar be $n \times m$ with $n \leq m$ as usual. It is also necessary to specify which edge of the bar the two poisoned squares share. We say that the two poisoned squares (x_1, y_1) and (x_2, y_2) share an edge if $x_1 = x_2$ or $y_1 = y_2$. If the two poisoned squares do not share an edge, then Theorem 2.2.6 applies.

If the two poisoned squares share one of the two larger edges (i.e. $x_1 = x_2$ and $y_2 - y_1 = m - 1$ or $y_1 = y_2$ and $x_2 - x_1 = m - 1$), we can utilize similar logic to the previous section: if $2 \cdot n = m$ then the bar can be split into two square bars, each with a single poisoned square. This will result in a $2 \times$ Nim-losing state and therefore a loss as seen in theorem 2.2.6.

Otherwise, the bar can be split into one row, dimensions $1 \times m$. This will guarantee a tie, since the following player can break off a single poisoned square, and so can player one in the next turn. This is shown in figure 2.8.

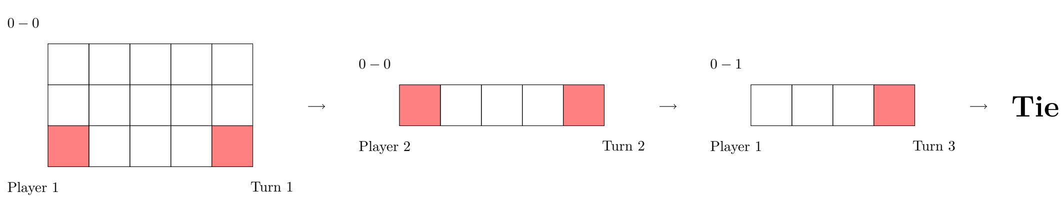


Figure 2.8: Progression of a game with two poisoned squares in adjacent corners.

This is not a complete analysis of the game, but it is a start. More research is necessary to determine and fully formalize the optimal strategy for these games.

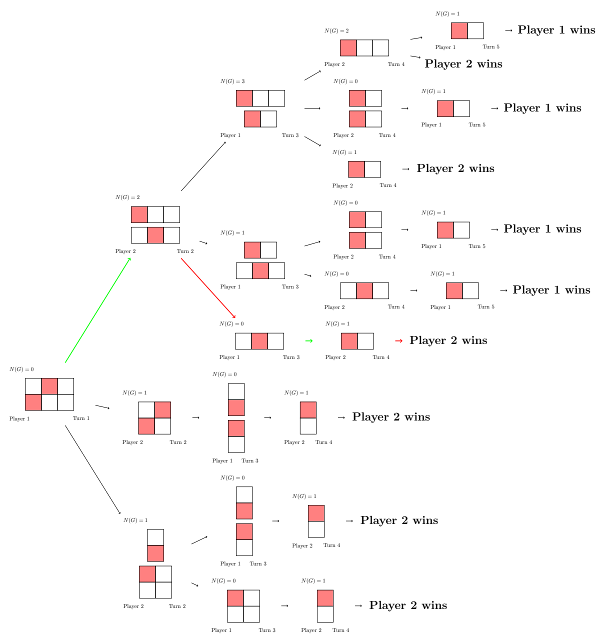
2.3 Recursive DiviNimbers

In an ideal world, we would be able to define a nimber for every game of DiviNim. In general, the optimal strategy for a game will always by definition end up being a series of slices that result in the opponent being in a losing position — a Nim-sum of zero. The only thing changing is the way that Nim-sums are calculated. If it is possible to define a valid nimber construction for DiviNim, and then also find a way to calculate this nimber in less time than it takes to calculate the minimax tree, then we will have a powerful tool for analyzing the game.

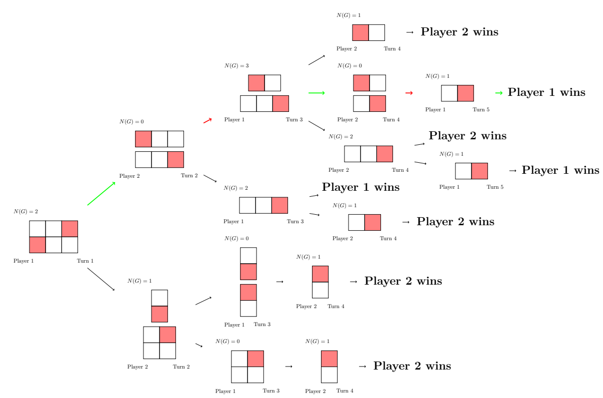
2.3.1 Last One Standing DiviNim

When examining the Last One Standing variant of DiviNim, we can see that the game is simpler to analyze. We can see the standard theorems involving nimbers extend beyond the simple 1-square case — the nimber of a bar, no matter the amount of poisoned squares, determines the outcome of the game. We can use the recursive definition of nimbers (definition 1.2.4) to determine the nimber of a bar with multiple poisoned squares.

For example, consider the games in Figure 2.9. The game in Figure 2.9a is a guaranteed loss for the player who has to play on it, we can see that the minimax algorithm cannot find a way to win the game. The game in Figure 2.9b is a guaranteed win for the player who has to play on it, as shown by the path found by the minimax algorithm. The Nim-sum of the game is 2, and the player can always slice the bar into a Nim-losing position.



(a) A 2×3 chocolate bar with a Nim-sum of 0.



(b) A 2×3 chocolate bar with a Nim-sum of 2.

Figure 2.9: Game trees for two configurations of Last One Standing DiviNim

So, we can create a function for finding the nimber of a bar in Last One Standing DiviNim with an additional base case. This means we can stop recursing earlier than the previous, more general version. Specifically, we have the following:

$$N(G) = \begin{cases} 0 & \text{if } n = 1, m = 1 \\ n_1 \oplus n_2 \oplus n_3 \oplus n_4 & \text{if } n \geq 1, m > 1, G \text{ has one poisoned square} \\ \operatorname{mex}\left(N(G_1), N(G_2), \ldots\right) & \textit{otherwise} \end{cases}$$

Where $n_1 \ldots n_4$ are the squares in each direction from the poisoned square (from 2.1.3), and $G_1, G_2, \ldots$ are the games that can be sliced from G, by slicing on any bar.

For last-one-standing DiviNim, a way to determine the nimber faster than the above has not been found. More research is necessary to determine if such a formula exists. In the current form, nimbers are not especially useful because calculating one requires searching almost every possible game state to determine the nimber of a game — just as the ordinary minimax algorithm does. We are only able to do nimber calculations of one-square bars in constant time, anything else rerquires recursion. If a non-recursive formula for the nimber of a bar with multiple poisoned squares exists, it would be a powerful tool for analyzing DiviNim. Additionally, the nimber of a bar is not necessarily useful information on its own unless it can lead you to the optimal move in a reasonable amount of time. In simpler games like Nim that can be clear, but for multi-square DiviNim games it can be exceedingly complex to calculate.

2.3.2 Scored DiviNim

On the whole, scoring-play combinatorial games are somewhat understudied in the literature. The work of Fraser Stewart [9] is one of the few sources that discusses scoring-play combinatorial games. In his work, Stewart discusses scoring play games in a very abstract way, including impartial scoring play games (which DiviNim is an example of). Unfortunately there does not yet seem to be any theorem as powerful as the Sprague-Grundy theorem (theorem 1.2.2) for scoring-play games.

Unfortunately, we find that Scored DiviNim is an even more complex game than Last DiviNim. With the addition of a score, the combination of two games is no longer equivalent to the sum of their component states, and so Nim-addition of games is no longer a valid operation in the general case. Also noteworthy is that ties are possible in Scored DiviNim, and the existing literature on impartial games defines only the $\mathcal{P}$-position and $\mathcal{N}$-position states. In order to analyze Scored DiviNim in a similar way, we would need to define a new operation for combining games that is valid for scoring play impartial games.

We know from Theorem 2.1.1 that DiviNim is an impartial game. If it is possible to define a combination operation for Scored DiviNim that is valid, then we can define a nimber for Scored DiviNim. It would be extremely useful if such a construction exists, as it would allow us to analyze the game in a similar way to how we analyze ordinary impartial games.

Conjecture 2.3.1. There exists a way to define nimbers for Scored DiviNim that is valid — a construction such that a nimber of 0 implies a losing position, and a nimber greater than 0 implies a winning position.

Chapter 3

Conclusion

> "The only winning move is not to play."
>
> WOPR, *WarGames*

In this report, we have examined the game of DiviNim. We have found that the game is an example of an impartial combinatorial game, and optionally a scored impartial combinatorial game. We have found that the game can under certain conditions be analyzed using the Sprague-Grundy theorem, and utilized this analysis to find optimal strategies for some configurations of the game — specifically, the one-square (Theorem 2.2.5), and two-corner opposing (Theorem 2.2.6) configurations. We have also found that under last-move scoring, the game is equivalent to the game of Nim, and thus can be analyzed using Nim addition and regular nimbers.

There are many open questions remaining about DiviNim. A short list of some of the most pressing questions includes:

- Is there a general strategy for DiviNim that can be applied to any configuration of the game? If so, what is it, and is it feasible for human players to use on non-trivial games?
- Is there a way to describe Nim addition for Scored DiviNim? If so, how can it be used to find optimal strategies for Scored DiviNim games?
- The original rule additions are somewhat arbitrary; there are alternative ways the game could be run. For example, after slicing, the following player might be forced to play on the smaller of the two bars. Additionally, we could say the first player to be handed a poisonous square is immediately the loser. We might also say that a player's score is the amount of squares they remove.
- Is there a way to generalize even further? For example, what if the grid is made of an arbitrary tesselation of the plane, such as with triangles that can be sliced in one of three ways? Similarly, what if the grid were three-dimensional? What if there could be more than two players? How much of the analysis done in this report can be extended to these new games?

More research is needed to answer these questions, and to further explore the properties of DiviNim. DiviNim is an interesting game with relations to many others, and we hope that this report has inspired further research into the game, its properties and strategies, and into scoring-play impartial combinatorial games in general.

Appendix A

DiviNimity

A web application has been developed to allow users to play DiviNim for themselves. The application allows users to play against various AI strategies, while easily examining the various results in this paper, such as the nimbers of boards and squares, the minimax results for a game state, and the game tree for the possible moves for any given game state. The app also allows users to quickly simulate large numbers of games and view statistics on the outcomes.

The application is available at kinseyda.github.io/divinimity

Bibliography

[1] J. Tanton, *Solve This: Math Activities for Students and Clubs*. 2001, ISBN: 0-88385-717-0.

[2] J. H. Conway, *On Numbers and Games*. Academic Press, Inc., 1976, ISBN: 0-12-186350-6.

[3] C. L. Bouton, "Nim, a game with a complete mathematical theory," *Annals of Mathematics*, vol. 3, no. 1/4, pp. 35–39, 1901, ISSN: 0003486X, 19398980.

[4] R. Sprague, "Über mathematische kampfspiele," *Tohoku Mathematical Journal*, vol. 41, pp. 438–444, 1935.

[5] P. M. Grundy, "Mathematics and games," *Eureka*, vol. 2, pp. 6–8, 1939.

[6] D. Knuth, *Surreal Numbers, How Two Ex-Students Turned on to Pure Mathematics and Found Total Happiness*. Addison-Wesley Professional, 1974, ISBN: 978-0201038125.

[7] J. von Neumann, "Zur theorie der gesellschaftsspiele," *Mathematische Annalen*, vol. 100, pp. 295–320, 1928.

[8] J. von Neumann, S. Bargmann, R. Aumann, *et al.*, "On the theory of games of strategy," in *Contributions to the Theory of Games (AM-40), Volume IV*. Princeton University Press, 1959, pp. 13–42, ISBN: 9780691079370.

[9] F. Stewart, "Scoring play combinatorial games," in *Games of No Chance 5* (Mathematical Sciences Research Institute Publications v. 5), U. Larsson and C. G. T. Workshop, Eds., Mathematical Sciences Research Institute Publications v. 5. Cambridge University Press, 2019, ISBN: 9781108485807.