1

Nim and Combinatorial Games

Combinatorial game theory is about perfect-information two-player games, such as Checkers, Go, Chess, or Nim, which are analyzed using their rules. It tries to answer who will win in a game position (assuming optimal play on both sides), and to quantify who is ahead and by how much. The topic has a rich mathematical theory that relates to discrete mathematics, algebra, and (not touched here) computational complexity, and highly original ideas specific to these games.

Combinatorial games are not part of "classical" game theory as used in economics. However, they nicely demonstrate that game theory is about rigorous, and often unfamiliar, mathematical *concepts* rather than complex techniques.

This chapter is only an introduction to combinatorial games. It presents the theory of *impartial games* where in any game position both players have the same allowed moves. We show the powerful and surprisingly simple result (Theorem 1.14), independently found by Sprague (1935) and Grundy (1939), that every impartial game is equivalent to a "Nim heap" of suitable size.

In Section 1.8 we give a short glimpse into the more general theory of partizan games, where the allowed moves may depend on the player (e.g., one player can move the white pieces on the game board and the other player the black pieces).

For a deeper treatment, the final Section 1.9 of this chapter lists some excellent textbooks on combinatorial games. They treat impartial games as a special case of general combinatorial games. In contrast, we first treat the simpler impartial games in full.

1.1 Prerequisites and Learning Outcomes

The combinatorial games that we consider are finite, and dealing with them uses a type of mathematical induction that we explain further in Section 1.3. It is helpful to have seen induction about the natural numbers before. Although not essential, is is useful to know the algebraic concept of an abelian group, in particular addition modulo 2. We assume familiarity with the *binary* system where numbers are

written in base 2, using only the two digits 0 and 1, rather than in the familiar base 10.

After studying this chapter, you should be able to:

- play Nim optimally;
- explain the different concepts of options, game *sums*, equivalent games, Nim values, and the mex rule;
- apply these concepts to play other impartial games like those described in the exercises;
- understand how some (not all) game positions in partizan games can be expressed as *numbers* that state how many moves the Left player is safely ahead, and why these numbers may be fractions with powers of two in the denominator.

1.2 Nim

Combinatorial games are two-player win-lose games of *perfect information*, that is, every player is perfectly informed about the state of play (unlike, for example, the card games Bridge or Poker that have hidden information). The games do *not* have chance moves like rolling dice or shuffling cards. When playing the game, the two players always alternate in making a move. Every play of the game ends with a win for one player and a loss for the other player (some games like Chess allow for a draw as an outcome, but not the games we consider here).

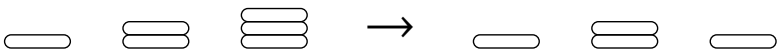
The game has a (typically finite) number of *positions*, with well-defined rules that define the allowed moves to reach the next position. The rules are such that play will always come to an end because some player is unable to move. This is called the *ending condition*. We assume the *normal play* convention that a player unable to move loses. The alternative to normal play is *misère play*, where a player who is unable to move wins (so the previous player who has made the last move loses).

We study *impartial* games where the available moves in a game position do not depend on whose turn it is to move. If that is not the case, as in Chess where one player can only move the white pieces and the other player the black pieces, the game is called *partizan*.

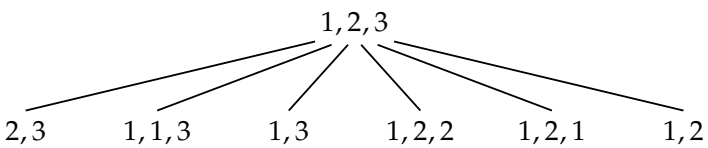
For impartial games, the game *Nim* plays a central role. A game position in Nim is given by some *heaps* of *tokens*, and a move is to remove some (at least one, possibly all) tokens from *one* of the heaps. The last player able to move wins the game, according to the normal play convention.

We analyze the Nim position 1, 2, 3, which means three heaps with one, two, and three tokens, respectively. One possible move is to remove two tokens from

the heap of size three, like here:



which we write as a move from $1, 2, 3$ to $1, 2, 1$. Because the move can be made in any one heap, the order of the heap sizes does not matter, so the position $1, 2, 1$ could also be written as $1, 1, 2$. The *options* of a game position are the positions that can be reached by a single legal move (according to the game rules) from the player to move. We draw them with moves shown as downward lines, like here,



(1.1)

where the first option $2, 3$ is obtained by removing from $1, 2, 3$ the entire heap of size 1, the second option $1, 1, 3$ by removing one token from the heap of size 2, and so on. The *game tree* is obtained by continuing to draw all possible moves in this way until play ends (game trees are studied in much more detail in Chapter 4). We may conflate options with obvious equal meaning, such as the positions $1, 1, 2$ and $1, 2, 1$ that can be reached from $1, 2, 2$. However, we do not draw moves to the same position from two different predecessors, such as $1, 1, 2$ that can be reached from $1, 1, 3$ and $1, 2, 2$. Instead, such a position like $1, 1, 2$ will be repeated in the game tree, so that every position has a unique history of moves.

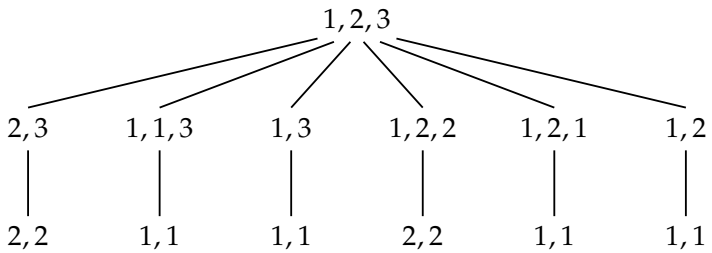
In an impartial game, the available moves in a game position are by definition independent of the player to move. A game position belongs therefore to exactly one of two possible *outcome classes*, namely it is either a *winning* or a *losing position*. "Winning" or "losing" applies to the player whose turn it is to move, assuming optimal play. A winning position means that the player can force a win with a suitable first "winning" move (and subsequent winning moves at all later positions). A losing position means that *every* move from the current position leads to a winning position of the other player, who can then force a win, so that the current player will lose.

When drawn in full with all possible subsequent moves, the Nim position $1, 2, 3$ has already a rather large game tree. However, we can tell from its options in (1.1) that $1, 2, 3$ is a losing position, using the following simple observations of how to play optimally in a Nim position with *at most two heaps*:

- If there are no Nim heaps left, this defines a losing position, by the normal play convention.
- A single Nim heap is a winning position; the winning move is to remove the entire heap.

- Two Nim heaps are a winning position if and only if the heaps have different sizes; the winning move is to equalize their sizes by suitably reducing the larger heap, which creates a losing position of two equal heaps. If the two heaps are equal, then a Nim move means reducing one of the heaps, so that the two heaps become unequal, which is a winning position; hence, two equal heaps define a losing position, as claimed.

Every option of $1, 2, 3$ shown in (1.1) is a winning position, because it allows moving to a two-heap losing position, as follows:



(1.2)

This picture shows sequences of two moves from position $1, 2, 3$, but not all of them, so this is not the top part of the full game tree. Only from the first position $1, 2, 3$, which is a losing position, all moves are shown. For the second move, only a single, winning move is shown, which leads again to a losing position. This suffices to prove that $1, 2, 3$ is a losing position in Nim, by the following observation.

Lemma 1.1. *In an impartial game, a game position is losing if and only if all its options are winning positions. A game position is winning if and only if at least one of its options is a losing position; moving there is a winning move.*

This lemma applies also to the game position that has no options at all, so trivially ("vacuously") all these options are winning (because there aren't any) and hence the position is losing. The lemma can also be taken as a definition of losing and winning positions, which is not circular but *recursive* because the options of a game position are *simpler* in the sense of being closer to the end of any play in the game. In Section 1.3 we will make this precise, and will give a formal proof of Lemma 1.1.

The concepts of winning and losing position assume optimal play. Binmore (2007, p. 49) writes: "A dull art movie called *Last Year in Marienbad* consists largely of the characters playing Nim very badly." In that movie (from 1961, directed by Alain Resnais), they play several times misère Nim, always from the starting position $1, 3, 5, 7$. There is a scene where three successive positions are

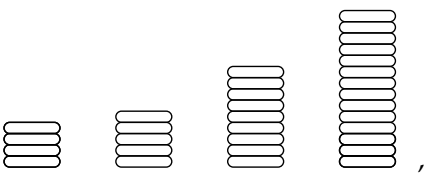
$$\begin{array}{ccccc} & \text{player I} & & \text{player II} & \\ 2,3,6 & \longrightarrow & 2,3,5 & \longrightarrow & 2,3,1 \end{array}$$

and then player I, whose turn it is, eventually loses. We can conclude from this that player I indeed played badly. Namely, either $2, 3, 1$ is a winning position, in

which case player I failed to force a win. Or 2, 3, 1 is a losing position (which is in fact the case), in which case player I failed to move there from position 2, 3, 6.

On the other hand, a game is more entertaining if it is not known how to play it optimally. In Chess, for example, which allows for a draw, one can show similarly to Lemma 1.1 that either White can force a win, or Black can force a win, or both players can force a draw. However, Chess is not yet "solved" in the sense that we know which outcome results from optimal play, which is why it is still an interesting game because players do not play perfectly.

Nim, however, has been solved by Bouton (1901). We demonstrate his winning strategy for the already formidable Nim position



which is 4, 5, 9, 14 and a winning position. The trick is to write the heap sizes in the *binary* system. The following tables show that one winning move is to reduce the Nim heap of size 5 to size 3, which leads to the losing position 4, 3, 9, 14:

		8	4	2	1
4	=	0	1	0	0
5	=	0	1	0	1
9	=	1	0	0	1
14	=	1	1	1	0
		0	1	1	0

$\rightarrow$

		8	4	2	1
4	=	0	1	0	0
$\boxed{3}$	=	0	$\boxed{0}$	$\boxed{1}$	1
9	=	1	0	0	1
14	=	1	1	1	0
		0	0	0	0

(1.3)

In (1.3), the top row represents the powers of 2 used to represent each heap size with binary digits, such as $5 = 0 \cdot 8 + 1 \cdot 4 + 0 \cdot 2 + 1 \cdot 1$. The bottom row is called the *Nim sum* of the heap sizes and is obtained by "addition modulo 2 without carry" for each column; that is, the Nim sum has a 0 if the number of 1's in that column is even, and a 1 if that number is odd.

We claim that a Nim position is losing if and only if that Nim sum is zero for all columns, like on the right in (1.3); we call such a position a *zero position*. In order to show this, we have to show, in line with Lemma 1.1, that

(a) every move from a zero position leads to a position which is not zero and therefore winning, and

(b) from every winning position (with a Nim sum that is not zero) there is a move to a zero position.

It easy to see (a): A move changes exactly one Nim heap, which corresponds to one row in the table of binary numbers, and changes at least one binary digit of

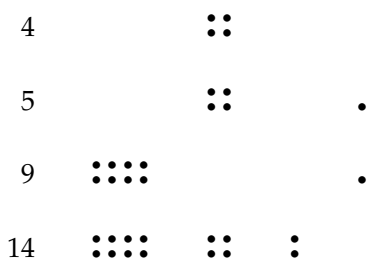
the binary representation of the heap size. Therefore, the resulting Nim sum is also changed in the corresponding digits and becomes nonzero.

Condition (b) amounts to finding at least one winning move for a position that is not zero, like on the left in (1.3). Such a winning move is found by these steps:

- Choose the leftmost "1" column in the Nim sum (which exists because not all columns are 0), which represents the highest power of two used in the binary representation of the Nim sum, here $4 = 2^2$.
- By definition, there is an odd number of Nim heaps which use that power of two in their binary representation. In (1.3), these are the heaps of sizes 4, 5, and 14. Choose one of the corresponding rows; we choose the heap of size 5.
- In the chosen row, "flip" the binary digits for *each* 1 in the Nim-sum, here for the columns 4 and 2; the resulting changes are shown with boxes on the right in (1.3). The largest of the changed powers of two in that row has digit 1, which is changed to 0. Even if all subsequent digits were changed from 0 to 1, the resulting binary number gives a new, *smaller* integer (possibly zero, which will mean removing the entire heap). The winning move is to reduce the Nim heap to that smaller size, here from 5 to 3.

As a result of these steps, the Nim sum is changed from 1 to 0 exactly in the columns where there was a 1 before, with the digits 0 untouched, so the new Nim sum has a 0 in each column. The resulting Nim position is therefore a zero position, as required in (b).

This winning strategy in Nim can be done with pen and paper, but is probably difficult to perform in one's head in actual play. For small heap sizes, a helpful mental image is to group the tokens in each Nim heap (now shown as dots) into distinct powers of two:



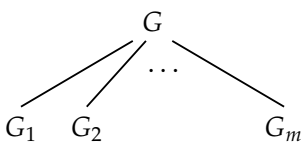
(1.4)

This is an equivalent representation to the left table in (1.3), but without binary digits. It shows that the powers of two that occur an odd number of times are 4 and 2. A winning move has to change one heap so that afterwards they occur an even number of times. This can be done by changing the four dots in the heap of size 5 to two dots, as in ⁘ → ⁚. This is the winning move in (1.3) that removes two tokens from the heap of size 5; the same winning move can be made in the heap of size 4. The third possible winning move is to remove six tokens

(⁘ and ⁚) from the heap of size 14. Caution: The resulting powers of two in a heap size must be *distinct*. The change ⁘ → ⁚ is not allowed in the heap of size 14, because the intended result ⁙ ⁚ ⁚ is not a binary representation because 2 appears twice; rather, the result of removing two tokens from the heap of size 14 is ⁙ ⁘, and then the overall created Nim position is not zero. Once the heap for the winning move is chosen, the number of tokens that have to be removed is unique.

1.3 Top-down Induction

When talking about combinatorial games, we will often use for brevity the word *game* for "game position". Every game G has finitely many *options* $G_1, \ldots, G_m$ that are reached from G by one of the allowed moves in G, as in this picture:



(1.5)

If $m = 0$ then G has no options. We denote the game with no options by 0, which by the normal play convention is a losing game. Otherwise the options of G are themselves games, defined by their respective options according to the rules of the game. In that way, any game is completely defined by its options. In short, the starting position defines the game completely.

We introduce a certain type of mathematical induction for games, which is applied to a *partial order* (see the background material text box on the next page).

Consider a set S of games, defined, for example, by a starting game and all the games that can reached from it via any sequence of moves of the players. For two games G and H in S, call H *simpler* than G if there is a sequence of moves that leads from G to H. We allow $G = H$ where this sequence is empty. The relation of being "simpler than" defines a partial order which for the moment we denote by $\leq$. Note that $\leq$ is antisymmetric because it is not possible to reach G from G by a nonempty sequence of moves because this would violate the ending condition. The ending condition for games implies the following property:

Every nonempty subset of S has a minimal element. (1.8)

If there was a nonempty subset T of S without a minimal element, then we could produce an infinite play as follows: Start with some G in T. Because G is not minimal, there is some H in T with $H < G$, so there is some sequence of moves from G to H. Similarly, H is not minimal, so another game in T is reached from H. Continuing in this manner creates an infinite sequence of moves, which contradicts the ending condition.

Background material: Partial orders

Definition 1.2 (Partial order, total order)**.** A binary relation $\leq$ on a set S is called a *partial order* if the following hold for all x, y, z in S:

$$x \leq y \text{ and } y \leq z \quad \Rightarrow \quad x \leq z \qquad (\leq \text{ is } \textit{transitive}),$$
$$x \leq x \qquad (\leq \text{ is } \textit{reflexive}), \text{ and}$$
$$x \leq y \text{ and } y \leq x \quad \Rightarrow \quad x = y \qquad (\leq \text{ is } \textit{antisymmetric}).$$

If, in addition, for all x, y in S

$$x \leq y \text{ or } y \leq x \qquad (\leq \text{ is } \textit{total})$$

then $\leq$ is called a *total order*. □

For a given partial order $\leq$, we often say "x is less than or equal to y" if $x \leq y$. We then define $x < y$ ("x is less than y") as follows:

$$x < y \quad \Leftrightarrow \quad x \leq y \text{ and } x \neq y\,. \tag{1.6}$$

This relation $<$ is also called the *strict order* that corresponds to $\leq$. Exercise 1.1 asks you to show how a partial order $\leq$ can be defined from a relation $<$ on S with suitable properties (transitivity and "irreflexivity") that define a strict order. Then $\leq$ is obtained from $<$ according to

$$x \leq y \quad \Leftrightarrow \quad x < y \text{ or } x = y\,. \tag{1.7}$$

An element x of S is called *minimal* if there is no y in S with $y < x$.

Any set S of rational or real numbers has the familiar relation $\leq$, which is a total order. The most important partial order that is usually not total is the *set inclusion* $\subseteq$ on a set S of sets. Then elements x, y of S as used in Definition 1.2 are usually written with capital letters like A, B. The partial order is written with the symbol $\subseteq$ where $A \subseteq B$ holds if A is a subset of B (which allows for $A = B$). The inclusion order is not total because two sets may be *incomparable*, that is, neither $A \subseteq B$ nor $B \subseteq A$ hold, for example if $A = \{1, 2\}$ and $B = \{2, 3, 4\}$. If S is the set of *all* subsets of, say, the finite set $\{1, \ldots, n\}$ for some positive integer n, then S, partially ordered by $\subseteq$, has the empty set $\emptyset$ as a unique minimal element. If S is the set of *nonempty* subsets of $\{1, \ldots, n\}$, then its minimal elements are the singletons $\{1\}, \{2\}, \ldots, \{n\}$.

For any partial order with the property (1.8), the following theorem applies.

Theorem 1.3 (Top-down induction)**.** *Consider a set S with a partial order $\leq$ that fulfills (1.8). Let $P(x)$ be a statement that may be true or false for an element x of S, and assume that $P(x)$ holds whenever $P(y)$ holds for all y of S with $y < x$. Then $P(x)$ is true for all x*

in S. In symbols, where "$\forall\, x$" means "for all elements x of S":

$$\big(\forall\, x \;:\; (\forall\, y < x \;:\; P(y)) \;\Rightarrow\; P(x)\big) \quad\Rightarrow\quad \big(\forall\, x \;:\; P(x)\big)\,. \tag{1.9}$$

Proof. Suppose, as assumed, that condition $P(x)$ holds whenever it holds for all elements y of S with $y < x$. Consider the set $T = \{\, z \in S \mid P(z) \text{ is false} \,\}$. Then $P(x)$ is true for all x in S if T is the empty set, so assume T is not empty. Let x be a minimal element of T, which exists by (1.8). Then all y in S with $y < x$ do not belong to T, that is, $P(y)$ is true. But, by assumption (1.9), this implies that $P(x)$ is also true, which contradicts $x \in T$. So T is empty, as claimed. □

Theorem 1.3 is called a method of *induction* because it assumes that a property $P(x)$ can be inferred from the fact that $P(y)$ is true for all y that are smaller than x. Then, assuming (1.8), the property holds for all x in S. An important application of this induction method is for proving the prime number decomposition theorem $P(x)$, which states that if x is an integer and $x \geq 2$, then x can be written as a product of prime numbers. In this induction proof, one first shows that x has a prime factor p, where either $x = p$ (in which case we have shown $P(x)$ and are done) or $x = p \cdot y$ for some integer y with $2 \leq y < x$. In the latter case, $P(y)$ holds by the "inductive hypothesis", that is, $y = p_1\, p_2 \cdots p_k$ for some prime numbers $p_1, p_2, \ldots, p_k$. Hence, $x = p\, p_1\, p_2 \cdots p_k$, which also shows $P(x)$. By Theorem 1.3, $P(x)$ holds throughout. The induction can be applied because any nonempty set of positive integers has a minimal element.

The method is called "top-down" induction because we start from the "top" (here the element x) and use the property $P(y)$ in some way for elements y that are smaller than x, where we typically encounter y without knowing *how much smaller* y is compared to x, as in the prime number decomposition theorem. In contrast, ordinary induction for natural numbers n proceeds typically by proving $P(1)$ and then the implication $P(n) \;\Rightarrow\; P(n+1)$, that is, one step at a time, which then implies that $P(n)$ holds for all natural numbers n. Top-down induction is more powerful; it also applies to partial orders and not just to the totally ordered natural numbers (where it is often called "strong induction").

Where did the base case go, which is normally proved separately in an induction proof? The base case states that $P(x)$ holds for any minimal element x (if $\leq$ is total, then there is only one minimal element). The answer is that the base case is covered by the assumption in (1.9): The left-hand side of the implication (1.9) says that

$$(\forall\, y < x \;:\; P(y)) \;\Rightarrow\; P(x) \tag{1.10}$$

holds for all x, in particular for all minimal x. But then $(\forall\, y < x \;:\; P(y))$ is vacuously true because there is no y with $y < x$. Therefore, for minimal x, condition (1.10) means that $P(x)$ is true, which *is* the base case.

As a proof technique, we can often use Theorem 1.3 generally without any case distinctions about whether the considered elements are minimal or not. It also allows using a property about games *recursively* by referring to the same property applied to *simpler* games.

Using top-down induction, we can now prove Lemma 1.1, which states that any game G is either winning or losing. Assume that this holds for all games that are simpler than G, in particular the options of G. If all of these options are winning, then G is a losing position, because the other player can force a win no matter which move the player makes in G. If not all options of G are winning, then one of them, say H, is losing, and by moving to H the player forces a win, hence G is a winning position. This argument is not in any way deeper than what we said before. Theorem 1.3 merely asserts that it is rigorous (and that it uses the ending condition).

1.4 Game Sums and Equivalence of Games

Combinatorial games often "decompose" into parts in which players can move independently, and the players then have to decide in which part to make their move. This is captured by the important concept of a *sum* of games.

Definition 1.4. Suppose that G and H are game positions with options (positions reached by one move) $G_1, \ldots, G_k$ and $H_1, \ldots, H_m$, respectively. Then the options of the *game sum* $G + H$ are

$$G_1 + H, \ldots, G_k + H, \qquad G + H_1, \ldots, G + H_m\,. \tag{1.11}$$

□

The first list of options $G_1 + H, \ldots, G_k + H$ in (1.11) simply means that the player makes his move in G, the second list $G + H_1, \ldots, G + H_m$ that he makes his move in H; the other part of the game sum remains untouched. As an example, a Nim position is simply the game sum of its individual Nim heaps, because the player moves in exactly one of the heaps.

Definition 1.4 is a recursive definition, because the game sum is defined in terms of its options, which are themselves game sums (but they are simpler games).

The sum of games turns out to define an *abelian group* on the (appropriately defined) set of games. It is a commutative and associative operation: for any games G, H, J,

$$G + H = H + G \qquad \text{and} \qquad (G + H) + J = G + (H + J)\,. \tag{1.12}$$

The first condition (commutativity) holds because the order of the options of a game, used in (1.11), does not matter. The second condition (associativity) holds