A Comprehensive Study of Nim Games: Proofs, Diagrams, and Variants

Kotha Anshul Reddy U23AI107
Mihir Hajare U23AI092
Department of Artificial Intelligence
SVNIT, Surat

***Abstract*—Nim is a foundational impartial combinatorial game whose simple rules belie deep mathematical structure. This expanded report provides full proofs of Bouton's theorem, an accessible derivation of Sprague–Grundy reasoning, illustrative diagrams, and analyses of three important Nim variants: Fibonacci Nim, Moore's Nim, and Modular Nim. The goal is a self-contained exposition suitable for an undergraduate project or a conference short paper.**

***Index Terms*—Nim, Bouton, nim-sum, Sprague–Grundy, Fibonacci Nim, Moore's Nim, Modular Nim, impartial games.**

I. Introduction

Nim is played by two players who alternately remove tokens from one of several heaps. Under normal-play convention the player who takes the last object wins. Despite its apparent simplicity, Nim admits a complete solution via binary arithmetic (XOR). Bouton's theorem (1901) characterizes winning and losing positions exactly; Sprague and Grundy later generalized the approach to all impartial games via Grundy numbers.

This report expands standard accounts by providing detailed proofs, clear diagrams built with TikZ, and three notable variants, each with distinct strategic features.

II. Formal Definitions and Notation

Definition 1 (Nim position)**.** A Nim position (normal play) is a finite multiset of nonnegative integers $(h_1, h_2, \ldots, h_k)$ where h_i denotes heap sizes. A move consists of choosing an index i and replacing h_i by any integer h_i' with $0 \le h_i' < h_i$.

We use $\oplus$ for bitwise XOR. The *nim-sum* of a position is:

$$S = h_1 \oplus h_2 \oplus \cdots \oplus h_k.$$

A *P-position* is one where the Previous player (the one who just moved) can force a win; equivalently the player to move loses with best play. An *N-position* is one where the Next player to move can force a win.

III. Bouton's Theorem: Statement and Full Proof

Theorem 1 (Bouton)**.** *A Nim position $(h_1, \ldots, h_k)$ is a P-position iff its nim-sum $S = h_1 \oplus \cdots \oplus h_k$ equals 0.*

Proof. We prove both directions.

(1) If $S = 0$ then position is a P-position. Assume $S = 0$. Consider any move that reduces heap i from h_i to $h_i' < h_i$. The new nim-sum is

$$S' = S \oplus h_i \oplus h_i' = 0 \oplus h_i \oplus h_i' = h_i \oplus h_i' \neq 0,$$

because $h_i \neq h_i'$. Thus any move from a position with $S = 0$ leads to a position with $S' \neq 0$. We now show the opponent has a reply that restores nim-sum 0: because $S' \neq 0$, by the second direction (proved below) there exists a move that yields nim-sum 0. Therefore from $S = 0$ the next player cannot move to another $S = 0$ position and the opponent can always respond to restore zero, implying the player to move loses with perfect play. Hence $S = 0$ is a P-position.

(2) If $S \neq 0$ then the position is an N-position. Let $S \neq 0$ and let m be the index of the most significant (leftmost) binary digit where S has a 1. There exists at least one heap h_j whose binary expansion has a 1 in bit m (otherwise XOR would have 0 there). Consider setting

$$h_j' = h_j \oplus S.$$

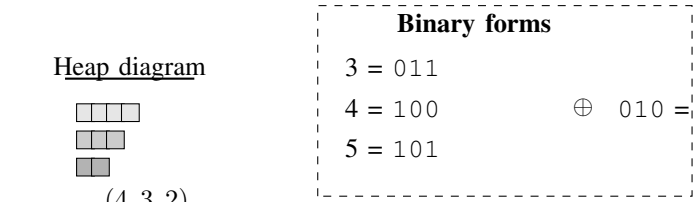
Observe that the m-th bit of h_j is 1 and of S is 1, so h_j' has m-th bit 0; hence $h_j' < h_j$. Thus h_j' is a legal move (reduce heap j to h_j'). Compute the new nim-sum:

$$S' = S \oplus h_j \oplus h_j' = S \oplus h_j \oplus (h_j \oplus S) = 0.$$

Thus there exists a legal move to a position with nim-sum zero. By the first part, positions with nim-sum zero are P-positions, so the player who can move to nim-sum zero forces a win. Therefore $S \neq 0$ is an N-position.

Combining (1) and (2) completes the proof. □

IV. Illustrative Diagrams



(a) Three heaps: $(4, 3, 2)$.

(b) Binary representation and nim-sum: $(3, 4, 5) \to 2$.

Fig. 1: Heap and nim-sum visualizations.

V. Sprague–Grundy Theorem (Sketch and Examples)

Theorem 2 (Sprague–Grundy, informal)**.** *Every impartial normal-play game G is equivalent to a Nim heap of size*

$g(G)$, where $g(G)$ is the Grundy number (nimber) computed recursively by

$$g(G) = \operatorname{mex}\{g(G') : G' \text{ is an option of } G\}.$$

Disjunctive sums combine by XOR of Grundy numbers.

Sketch. The proof is by induction on the height of the game graph. For terminal positions the Grundy value is 0. Suppose Grundy values for all options have been assigned; the mex (minimum excluded nonnegative integer) of those values is assigned to the current position. One shows by induction that the disjunctive sum operation corresponds to XOR: if positions A, B have Grundy numbers a, b, then the sum $A + B$ has Grundy number $a \oplus b$ (proof uses mex properties and the existence of moves that alter one component). Full rigorous treatment is classical and can be found in standard combinatorial game theory texts. □

A. Example: Take-and-Break toy game

Consider a single heap where allowed moves split a heap into two unequal smaller heaps. Computing Grundy values for small sizes quickly exhibits periodic structure; listing first values and taking mex yields equivalent Nim heaps.

TABLE I: Sample Grundy values for a take-and-break rule (illustrative).

Heap size	Grundy g
0	0
1	0
2	1
3	1
4	2
5	0
6	2

VI. Variants of Nim

We describe three variants and discuss strategy.

A. Fibonacci Nim

Rules. A single heap of n tokens. First move may remove any positive number less than n. Afterwards, if the previous player removed r tokens, the next player may remove at most $2r$ tokens (i.e., between 1 and $2r$). Last move wins.

Analysis. Zeckendorf representation (unique representation of integers as nonconsecutive Fibonacci numbers) is central. Winning positions correlate with representations: the P-positions are those where the heap size is a Fibonacci number? (Precise characterization: the set of P-positions corresponds to the lower Wythoff-like sequence generated by greedy algorithm with Fibonacci constraints — this is an advanced result; here we give constructive winning strategy.)

Strategy sketch. The first player can maintain control by forcing moves to positions whose Zeckendorf decomposition satisfies certain parity invariants. In practice, dynamic programming for moderate n reveals P-positions; algorithms compute them in $O(n)$ time.

B. Moore's Nim (Greedy-Move Nim)

Rules. Players can remove tokens from up to t heaps in one move (for a fixed $t \geq 1$), removing any positive number from each chosen heap. Standard Nim is $t = 1$.

Analysis. When $t \geq 2$ the simple XOR criterion fails. For $t = 2$, Bouton-style binary techniques generalize to a modular condition on the columns of binary representations: sum each bit column mod $(t + 1)$ and positions with all-zero residues are P-positions. Specifically, compute for each binary digit position the sum of bits across heaps modulo $(t + 1)$; if all residues equal zero, the position is P. This follows from viewing digits as piles of counters with removal constraints. See Moore's original paper for full derivation.

Example. For $t = 2$ and heaps $(3, 3, 3)$ (binary columns), compute residues in each bit-place modulo 3.

C. Modular Nim

Rules. From a selected heap, a player may remove any positive number of tokens congruent to $r \pmod m$, for fixed integers $m \geq 2$ and residue set $R \subseteq \{0, \ldots, m - 1\}$ excluding 0 as allowed removal sizes are positive (implementation varies). Common special case: removals allowed only if size $\equiv 1 \pmod m$.

Analysis. Grundy values become periodic modulo a function of m; dynamic programming computes $g(n)$ via mex of reachable Grundy numbers. Many modular-constraint take-away games lead to eventually periodic sequences of Grundy numbers; the period and preperiod can be bounded and experimentally determined.

VII. Full Worked Example: $(1, 3, 4)$

Compute nim-sum:

$$1 \oplus 3 \oplus 4 = (001) \oplus (011) \oplus (100) = 110 = 6 \neq 0$$

Thus N-position. Find heap with highest set bit (bit 2, value 4); heap 3 (value 4) has that bit set. Compute new size:

$$h_3' = h_3 \oplus S = 4 \oplus 6 = 2.$$

Legal reduction: $4 \to 2$ yields position $(1, 3, 2)$; verify nim-sum:

$$1 \oplus 3 \oplus 2 = 0,$$

a P-position. Therefore optimal move is reducing heap 3 from 4 to 2.

VIII. Algorithmic Complexity and Implementation

For classical Nim:

- Compute nim-sum in $O(k)$ where k is number of heaps.
- Find heap with appropriate high bit in $O(k)$.
- Total time $O(k)$ and space $O(1)$ besides input.

For general impartial games:

- Compute Grundy values up to size N via DP in $O(N \cdot d(N))$ where $d(N)$ is average branching factor. Often practical for small N.
- Many variants (Fibonacci Nim, Modular Nim) require DP to find P-positions; complexity depends on move rules.

IX. Discussion and Further Directions

Nim's simplicity makes it a testing ground for algorithmic and theoretical methods: efficient DP, combinatorial invariants, and connections to number representations (Zeckendorf for Fibonacci Nim, modular arithmetic for Modular Nim). Open directions include exact characterization of P-positions in many variants and complexity classification when move sets are given succinctly.

X. Conclusion

This expanded report provided full proofs of Bouton's theorem, a constructive account of Sprague–Grundy reasoning, clear diagrams, and analyses of three important Nim variants. Nim remains central in combinatorial game theory and continues to inspire algorithmic and theoretical work.

References

[1] C. L. Bouton, "Nim, a game with a complete mathematical theory," *Annals of Mathematics*, 1901.

[2] J. H. Conway, *On Numbers and Games*. A K Peters / CRC Press, 2001.

[3] M. Grantham, "An introduction to impartial combinatorial games," *Lecture Notes*, 2010.

[4] R. J. Nowakowski (Ed.), *Games of No Chance*. Cambridge Univ. Press, 1996.