Methods and Solutions for Non-Exact Index-2 Nim

Omar Afifi and Matthew Gherman
UCLA Department of Mathematics

December 12, 2022

Abstract

In this work, we build on previous work exploring NIM-variants and formalize Non-Exact k-NIM. It seems to have been initially presented by Berlekamp et al. in "Winning Ways" as an open problem [1]. The rules of the game involve some significant modifications to the rules of Nim, while still preserving many of the essential attributes of generic Nim variants. We discuss the origins of the problem and discuess previous work attempting to solve the problem numerically, as well as the difficulty of finding a recursive solution via the Sprague-Grundy Theorem. Lastly, Our main result it to prove solutions for $n \leq 5$.

Contents

1 Nim and Index-k Nim

Nim was famously introduced and solved by Bouton in [2]. In Nim, there are n heaps, each with a certain number of tacks/stones/matchsticks, and players take turns picking arbitrarily many tacks from only one heap. In the normal form of the game, the winning player is the player who removes the last tack. In the mísere form, the objective is to avoid removing the last tack. Bouton showed that the game could be solved by taking the binary representation of the count of each heap, and then considering the column-wise sum, modulo 2, of all the heaps. He showed that the losing positions of the game were those for which the sum was 0, and that a player could win by repeatedly leaving their opponent in such a position. Moore pointed out in [5] that the game could be generalized so that players could choose arbitrarily many tacks from $1 \leq k \leq n$ heaps. With these relaxed rules, Moore showed that Bouton's original strategy of analyzing column-wise sums still works, with the caveat that the sums need to be considered modulo $k+1$. This version of the game is referred to as index k Nim, or simply k-Nim.

2 Berlekamp's Princess and Roses

The version of NIM we consider was introduced by Berlekamp et al. in [1] in the following setting: A princess has a rose garden with n rose bushes. The princess has two suitors, p_1 and p_2, who take turns each morning bringing roses to the princess, with p_1 having the first pick. They obey the following three rules.

- If at least one rose is left on any bush, at least one rose must be picked.
- No more than two roses may be picked in the same turn.
- If two roses are picked, they must be taken from different bushes.

The objective of the game is to pick the last rose. With n bushes, a position in the game can be represented by an ordered n-tuple of non-negative integers representing the number of roses on each bush. Let k be the number of bushes from which a player can take up to one rose. Each suitor will subtract one from at least one and at most k coordinates in the ordered n-tuple. We restrict ourselves to $k = 2$. We refer to the game as Non-Exact k-NIM, because it is similar to Moore's version in that we allow moves involving k heaps, but we restrict the number of items players may remove from each heap. It also is similar to another version dubbed "Exact k-NIM", analyzed by Gurvich and Bao Ho in [4], except that they require exactly k-heaps be utilized in a move, whereas we relax that condition to admit any number $\leq k$. They were able to find deterministic formulas for Sprague Grundy values when $n \leq k+2$ and $n = k+3 \leq 6$.

While the game seems similar enough to Moore's version of Nim, the Moore/Bouton strategy of column-wise sums modulo $k+1 = 3$ does not work. The ability to take arbitrarily many tacks allows the player to modify a row substantially enough to return the sums to zero after it has been altered. However, when the number of tacks that can be removed from a single bush is limited to one, players do not have sufficient control over the binary representation of the heaps to enforce this.

Example 2.1. Consider the position (3,5,6,7) when $n = 4$ and $k = 2$. We write the number of tacks in each heap in binary and sum the digits in each column modulo $k+1 = 3$.

0	1	1
1	0	1
1	1	0
1	1	1
0	0	0

Based on Moore's strategy, the column-wise sum, modulo 3, is zero, and so the position should be losing. However, we will later prove in Proposition 5.7 that this is in fact a winning position. Moreover, it should be noted that there is no valid move that will allow the player to maintain the 0-sum state.

3 Definitions and Notation

We adopt the following notational conventions:

For simplicity, G_n denotes an instance of Non-Exact 2-Nim with n heaps. We represent a position in an n-heap game, G_n, by a vector, $\overline{x} = (x_1, ..., x_n) \in \mathbb{N}^n$. (We allow 0 to be a natural number for convenience.) However, we also note that we sometimes consider $\overline{x}$ to be a position in a smaller game, G_{n-m} where m is the number of coordinates of $\overline{x}$ equal to 0.

Let $\overline{x} \in \mathbb{N}^n$ be a position in an n heap game, G_n. Denote $[n] = \{1, 2, \ldots, n\}$ and $e_i \in \mathbb{N}^n$ the vector with 1 in the ith coordinate and 0 in the other coordinates for $i \in [n]$. In accordance with the rules of the game, we say $\mathrm{P}(\overline{x})$ is a valid move from position $\overline{x}$ if $P(\overline{x}) = \overline{x} - e_i$ or $P(\overline{x}) = \overline{x} - (e_i + e_j)$ for distinct $i, j \in [n]$ provided each coordinate of $P(\overline{x})$ is non-negative.

We denote the set of possible moves from a given position by $\mathbb{P}(\overline{x}) = \{P(\overline{x}) : P(\overline{x}) \text{ is valid}\}$.

For a position $\overline{x}$, we let $\mathbb{E}(\overline{x})$ be the number of coordinates x_i of $\overline{x}$ such that $x_i \equiv 0 \pmod 2$. Likewise, $\mathbb{O}(\overline{x})$ is the number of coordinates x_i of $\overline{x}$ such that $x_i \equiv 1 \pmod 2$.

Let $\overline{y} \in \mathbb{N}^n$. We say $\overline{y}$ is *reachable* from $\overline{x}$ or $\overline{x}$ *points* to $\overline{y}$ if $y = P(\overline{x})$ for some $P(\overline{x}) \in \mathbb{P}(\overline{x})$.

As players take turns, we will commonly denote the position after k turns of either player from some initial position, $\overline{x}$, by $P^k(\overline{x})$. When the current round of the game is not relevant, we drop this notation.

Given an initial position $\overline{x}$, we adopt the convention that p_1 plays first and p_2, the opponent of p_1, plays second.

4 Sprague-Grundy for slow 2-Nim

The Sprague-Grundy Theorem has proven to be a very useful tool for solving combinatorial games of this nature. However, because the purpose of this work is not to deal with this strategy in general, we refer the reader to sources such as Ferguson [3] for a detailed treatment of this material. In this section, we show how to apply the theorem to Non-Exact k-Nim and explain the difficulties involved in using Sprague-Grundy to solve Non-Exact k-NIM.

Calculating the Grundy-values of a game is essentially providing a mapping onto Nim. As such, every impartial combinatorial, two-player game is equivalent to Nim. However, because Grundy values are calculated recursively, closed-form Grundy functions for complicated games can be difficult to find. Gurvich and Bao Ho attempted to find such a function for slow k-Nim, and their paper is a testament to the difficulty of the task [4].

Let $\mathcal{S}$ be the set of all possible game states. The Sprague-Grundy theorem gives a recursively defined function $\mathcal{G} : \mathcal{S} \to \mathbb{N}$. For $S \subset \mathbb{N}$ a subset of the natural numbers, the mex operator is

$$\text{mex}(S) = \min\{x \in \mathbb{N} : x \notin S\}.$$

In our context, the Grundy function satisfies $\mathcal{G}(\overline{0}) = 0$ and can be defined recursively as

$$\mathcal{G}(\overline{x}) = \text{mex}\{\mathcal{G}(\overline{y}) : \overline{y} \in \mathbb{P}(\overline{x})\}.$$

The Sprague-Grundy theorem classifies the losing positions of the game as those for which $\mathcal{G}(\overline{x}) = 0$. However, the recursive nature of the function does not offer a deterministic solution to the game, beyond the possibility of a dynamic programming approach. Even then, especially when the number of choices gets very large, time and memory efficiency become problematic. The real power of the Sprague-Grundy theorem comes from the following well-known result:

Theorem 4.1. If $G_1, \ldots, G_n$ are a collection of independent, impartial, combinatorial, two-player games, then we can consider a product of games $G = G_1 \times \cdots \times G_n$. If $x_1, \ldots, x_n$ are a collection of positions in each game, respectively, then:

$$\mathcal{G}(\overline{x}) = \mathcal{G}_1(x_1) \oplus (\mathcal{G}_2(x_2) \oplus (\cdots \oplus \mathcal{G}_n(x_n) \cdots))$$

where $\mathcal{G}$ denotes the Grundy function of the product game, and $\mathcal{G}_i$ gives the Grundy function of each G_i. $\oplus$ denotes a bit-wise XOR operation.

In other words, the theorem states that if we can decompose our combinatorial game into many small, simple games, then we can find the Grundy function of the larger game quite easily. It seems natural to try and simplify our problem using this result, and in fact, calculating the Grundy values for a 1-heap version of the game is simple.

$$\begin{aligned}
&\mathcal{G}(0) = 0 \\
&\mathcal{G}(1) = \text{mex}\{0\} = 1 \\
&\mathcal{G}(2) = \text{mex}\{1\} = 0 \\
&\mathcal{G}(3) = \text{mex}\{0\} = 1 \\
&\vdots
\end{aligned}$$

We obtain a simple binary sequence $(0, 1, 0, 1, 0, 1, \ldots)$. In fact $\mathcal{G}(x) = 0$ is equivalent to $x \equiv 0 \pmod 2$. Theorem 4.1 suggests that we should be able to find the Grundy values of any position $(x_1, \ldots, x_n)$ by simply taking the XOR sum of the Grundy values of each 1-heap sub-game. Since their Grundy values are 1 or 0 in accordance to their equivalence class modulo 2, then the XOR sum is entirely determined by whether or not the number of odd heaps is itself an even or odd number.

The above process does not work for Non-Exact k-Nim. We will show in Proposition 5.7 that the position (2,3,5,6) is losing, whereas the (3,4,5,6) is winning, even though they both have the same number of odd heaps. The reason this fails is that there is an implicit condition of independence. Non-Exact k-Nim allows us to pick from multiple heaps, so there is no way to make the sub-games independent. No matter how we break them up, there will always be a move in more than one sub-game. Hence, finding a neat, non-recursive formula for the Grundy function of Non-Exact k-Nim or proving that such a function does not exist is not a simple task and remains a possible future research endeavor.

5 Algebraic Solutions

For any number of heaps, a solution for the game remains open. Our main result here is to classify winning and losing positions for $n \leq 5$ and discuss observations when $n = 6$. The case of $n = 1$ is trivial and left as a thought exercise, so we begin with the following lemma.

Lemma 5.1. Let $\overline{x} = (x_1, ..., x_n)$ be a position in a game with n heaps. Suppose that for each i, $x_i \equiv 0 \pmod 2$. Then $\overline{x}$ is a losing position.

Proof. The proof is by strong induction on n. Solving the game when $n = 1$ solves the base case. Suppose that the result holds for all $1 \leq m \leq n$. Assume $\overline{x}$ satisfies $x_i \equiv 0 \pmod 2$ for all $1 \leq i \leq n$. If $\overline{x}$ is the zero vector, then p_1 loses the game. If not, p_1 is forced to leave some heap with an odd number of tacks. Then p_2 mimics the move to return the game to an all-even state. Repeat this procedure until the first heap is emptied. If p_2 plays correctly, it must be p_1's turn. By the induction hypothesis, p_1 loses. □

Corollary 5.2. Let $\overline{x}$ be a position in an n heap game, and suppose $\mathbb{O}(\overline{x}) \in \{1, 2\}$. Then $\overline{x}$ is a winning position.

Proof. By taking from odd heaps, p_1 is able to move so that $\mathbb{E}(P(\overline{x})) = n$. By Lemma 5.1, p_2 is in a losing position, which implies that $\overline{x}_0$ is a winning position. □

Together, these two results allow us to classify games G_n when $n \leq 4$. As the reader will notice, the conditions and proofs become increasingly more difficult.

5.1 The case of two heaps

Proposition 5.3. Let (x, y) be a position in a two-heap game. Then (x, y) is a winning position if and only if x or y is an odd number.

Proof. If $x_i \equiv 0 \pmod 2$ for $i \in \{1, 2\}$, then (x, y) is a losing position by Lemma 5.1. Otherwise, $\mathbb{O}(\overline{x}) \in \{1, 2\}$. The position is winning by Corollary 5.2. □

We can illustrate the two heap game by a directed graph. (See Figure 1.) Each node is a position. A directed edge between node i and node j represents a valid move from the position represented by node i to that of node j. We color the node red if it is a losing position and green if it is a winning position. The digraph has some recognizable and unsurprising features that highlight some fundamental properties of combinatorial games: Each green node has an edge to a red node while each red node points only to green nodes.

5.2 The case of three heaps

Lemma 5.4. Let $\overline{x}$ be a position in a 3-heap game. If $\mathbb{O}(\overline{x}) = 3$, then $\overline{x}$ is a losing position.

Proof. p_1 can either take one tack from an odd heap or one tack from each of two odd heaps. In either case, $\mathbb{O}(P(\overline{x})) \in \{1, 2\}$. By Corollary 5.2, $P(\overline{x})$ is a winning position for p_2. Therefore, $\overline{x}$ is a losing position for p_1. □

Remark. Note that the conclusion of Lemma 5.4 is not necessarily true if $\mathbb{O}(\overline{x}) = 3$ and $n \geq 4$. By taking a tack from an odd heap and an even heap, p_1 preserves $\mathbb{O}(P(\overline{x})) = 3$.

Together with Lemma 5.1 and Corollary 5.2, we can classify all winning positions in 3-heap games by considering the possible values of $\mathbb{O}(\overline{x})$:

Proposition 5.5. A position $\overline{x}$ in a 3-heap game is losing if and only if $\mathbb{O}(\overline{x}) = 0$ or $\mathbb{O}(\overline{x}) = 3$.

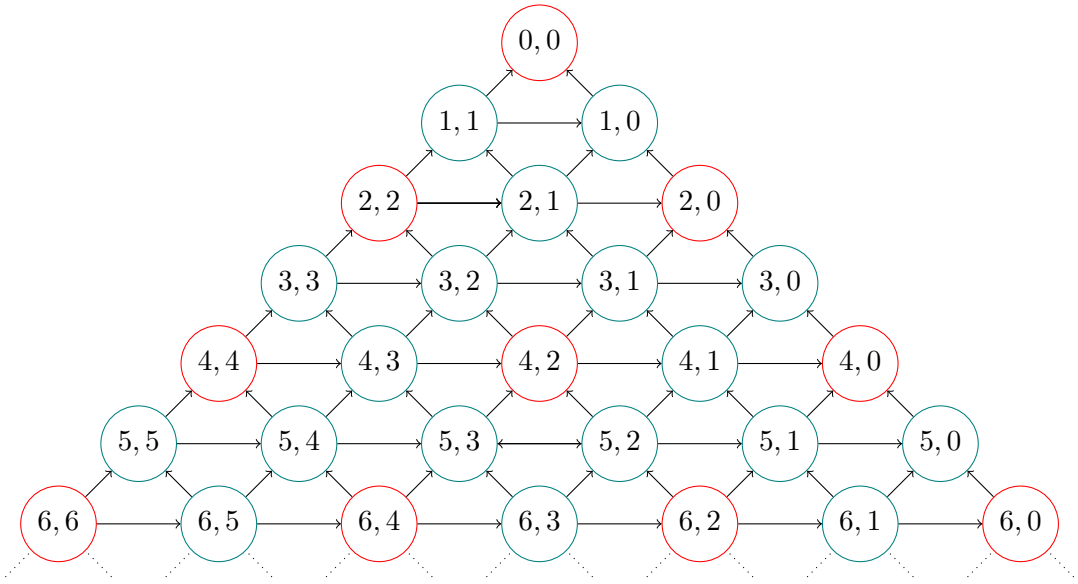

Figure 1: The digraph of G_2.

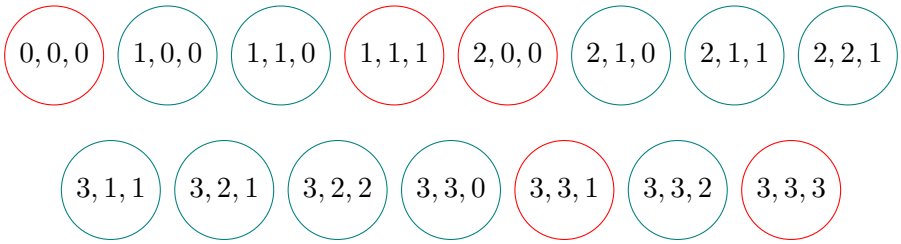

Figure 2: Some $n = 3$ positions.

5.3 The case of four heaps

Let $\overline{x}$ be a position in a 4-heap game. The following scenarios are determined by Lemma 5.1 and Corollary 5.2.

$$\mathbb{O}(\overline{x}) = 0 \text{ is a losing position}$$
$$\mathbb{O}(\overline{x}) = 1 \text{ is a winning position}$$
$$\mathbb{O}(\overline{x}) = 2 \text{ is a winning position}$$

We will classify $\mathbb{O}(\overline{x}) = 3$ and $\mathbb{O}(\overline{x}) = 4$ as winning or losing with the following results.

Lemma 5.6. Let $\overline{x}$ be a position in a 4-heap game with $\mathbb{O}(\overline{x}) \geq 3$. If $\min_{1 \leq i \leq 4}\{x_i\} = 1$, then $\overline{x}$ is a winning position.

Proof. Without loss of generality, assume $x_1 = 1$. At most one of x_2, x_3, and x_4 is even. Then p_1 removes one tack from the first heap and one tack from the even heap. We note that $P(\overline{x})$ is equivalent to a 3-heap game with $\mathbb{O}(P(\overline{x})) = 3$. By Lemma 5.4, $P(\overline{x})$ is a losing position so $\overline{x}$ is a winning position. □

Proposition 5.7. Let $\overline{x}$ be a position in a 4-heap game with $\mathbb{O}(\overline{x}) \geq 3$. If $\min_{1\leq i\leq 4}\{x_i\}$ is even, then $\overline{x}$ is a losing position. If $\min_{1\leq i\leq 4}\{x_i\}$ is odd, then $\overline{x}$ is a winning position.

Proof. We will proceed by double induction on $\min_{1\leq i\leq 4}\{x_i\}$. If $\min_{1\leq i\leq 4}\{x_i\} = 0$, then $\overline{x}$ is a 3-heap position with $\mathbb{O}(\overline{x}) = 3$, and a losing position by lemma 5.4. The base case $\min_{1\leq i\leq 4}\{x_i\} = 1$ is provided by Lemma 5.6. Assume that when $\min_{1\leq i\leq 4}\{x_i\} < k$ is even, $\overline{x}$ is a losing position. Further, assume that when $\min_{1\leq i\leq 4}\{x_i\} < k$ is odd, $\overline{x}$ is a winning position for p_1.

Let $\min_{1\leq i\leq 4}\{x_i\} = k$ be even. If p_1 takes tacks only from the odd heaps, p_2 will be left with one or two odd heaps. By Corollary 5.2, p_2 would be in a winning position. If p_1 takes a tack from the even heap, there will be either three or four odd heaps with odd minimum less than k. By the inductive hypothesis, p_2 is in a winning position. Therefore, $\overline{x}$ is a losing position for p_1.

Let $\min_{1\leq i\leq 4}\{x_i\} = k$ be odd. We must check two cases:

- Case 1: Assume $\mathbb{O}(\overline{x}) = 4$. Then p_1 takes one tack from a minimal odd heap. The new position has even minimum with three odd heaps. By the inductive hypothesis, p_2 is in a losing position.
- Case 2: Assume $\mathbb{O}(\overline{x}) = 3$. Then p_1 takes one tack from a minimal odd heap and one tack from the even heap. The resulting position has even minimum with 3 odd heaps. By the inductive hypothesis, p_2 is in a losing position.

In either case, $\overline{x}$ is a winning position for p_1. □

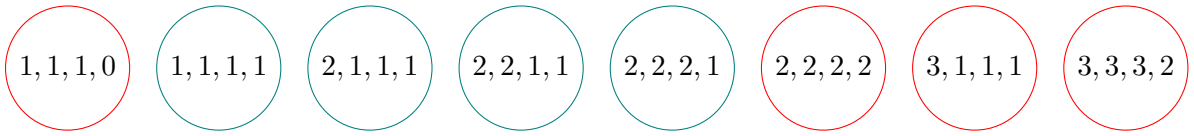

Figure 3: Some $n = 4$ positions.

5.4 The case of five heaps

Let $\overline{x}$ be a position in a 5-heap game. The following scenarios are determined by Lemma 5.1 and Corollary 5.2.

$$\mathbb{O}(\overline{x}) = 0 \text{ is a losing position}$$
$$\mathbb{O}(\overline{x}) = 1 \text{ is a winning position}$$
$$\mathbb{O}(\overline{x}) = 2 \text{ is a winning position}$$

When $\mathbb{O}(\overline{x}) > 2$, things are less obvious, but we have classified the positions when $n = 5$ in Proposition 5.8. However, the proof is dry and doesn't offer much insight into the problem, but can be found in the appendix. Here, we simply state the result before briefly discussing the $n = 6$ case and offering some concluding results.

Proposition 5.8. Let $\overline{x} = (x_1, x_2, x_3, x_4, x_5)$ be a position in a five-heap game. Without loss of generality, assume that $x_i \leq x_j$ when $i \leq j$: The position $\overline{x}$ is a losing position if and only if one of the following scenarios occurs:

(1) x_i is even for all $1 \leq i \leq 5$

(2) x_1, x_2 are even and x_3, x_4, x_5 are odd

(3) x_1, x_2 are odd and $x_3 \equiv x_4 \pmod 2$, $x_3 \not\equiv x_5 \pmod 2$

5.5 The case of six heaps

For the four and five heap cases, a position is classified by the parity and relative size of each pile. The six heap positions $(1,1,1,1,1,1)$ and $(1,1,1,1,1,3)$ each have six odd heaps. The next examples prove that the former is a losing position while the latter is a winning position, and as such, the classification for six heaps will be more complicated than that of four and five heaps. No significant progress has been made here to the best of our knowledge.

Example 5.9. The position $\overline{x} = (1,1,1,1,1,1)$ is a losing position. The two possible moves from $\overline{x}$ are $(0,0,1,1,1,1)$ and $(0,1,1,1,1,1)$. The former is a winning position by Proposition 5.7 and the latter is a winning position by Lemma 7.1. Therefore, $\overline{x}$ is a losing position.

Example 5.10. The position $\overline{x} = (1,1,1,1,1,3)$ is a winning position. If p_1 takes from the first and last heaps, p_2 is left with the position $\overline{y} = (0,1,1,1,1,2)$. By Proposition 5.8, $\overline{y}$ is losing so $\overline{x}$ is a winning position.

6 Conclusion

As we see, a solution for Non-Exact index-2 Nim presents a lot of challenges. Even for relatively small n, it is tricky, classification is messy, and an elegant technique is not evident. We believe that the best approach to a complete solution is a framework that allows for the construction of a decision-procedure in the form of an algorithm, or furthering the work towards a closed-form Grundy function of the game. The authors have explored reinforcement learning algorithms to learn the $n = 6$ strategy using the preceding results in default policy decisions, and hope to follow-up with these results. Helpful directions in the immediate future include expanding on machine-learning solutions, implementing dynamic-programming to discover patterns for $n = 6$, or expanding on the theoretic work using the algebraic methods we have presented here. Eventually, relaxing the $k = 2$ constraint would complete the solution.

7 Appendix: The Proof of Proposition 5.8

In this section, we prove 5.8. We warn the reader that this section is incredibly dry mathematically, and is something of a brute-force proof, but we include it for completeness. In order to prove 5.8, we need to establish a few preliminary results detailing the behavior of the games on 5-heap games. We formalize some observations for easy reference:

Lemma 7.1. Let $\overline{x} = (1, 1, x, y, z)$ be a position in a 5-heap game. If $\mathbb{O}(\overline{x}) = 5$, then $\overline{x}$ is a winning position.

Proof. By taking from the first and second heaps, p_1 produces a 3-heap position with three odd heaps. Apply Lemma 5.4. □

Lemma 7.2. The 5-heap position $(1, 1, x, x + 2k, x + (2\ell + 1))$ for $0 \leq x$ and $0 \leq k \leq \ell$ is losing.

Proof. We will proceed via induction on x. As a base case, take $x = 0$ and apply Lemma 5.4 or Proposition 5.7. Assume that $(1, 1, x, x + 2k, x + (2\ell + 1))$ is a losing position. We will show that

$$(1, 1, x + 1, (x + 1) + 2k, (x + 1) + (2\ell + 1))$$

is a losing position. If p_1 takes a tack from the first or second heap, then $\mathbb{O}(P(\overline{x})) \in \{1, 2, 3, 4\}$. In any case, p_2 has a winning position in a 3-heap or 4-heap game by Corollary 5.2 or Proposition 5.7. (since x, $x + 2k, x + 2(\ell + 1)$ cannot all have the same parity). Considering Lemma 7.1 as well, p_1 is left with the following four moves.

$$\begin{aligned}
&(1, 1, x, (x + 1) + 2k, (x + 1) + (2\ell + 1))\\
&(1, 1, x, (x + 1) + 2k, x + (2\ell + 1))\\
&(1, 1, x + 1, x + 2k, (x + 1) + (2\ell + 1))\\
&(1, 1, x + 1, x + 2k, x + (2\ell + 1))
\end{aligned}$$

By complementing p_1's move, p_2 produces the position

$$(1, 1, x, x + 2k, x + (2\ell + 1))$$

in all scenarios. By the inductive hypothesis, p_1 is in a losing position. Therefore, the original position is losing for p_1 since all possible moves produce a winning position. □

Remark. Note that in the first, third, and fourth scenarios of the proof of Lemma 7.2, p_2 could mimic the p_1 move. The issue with mimicry in the second scenario is that the fifth heap could end up with fewer tacks than that of the fourth heap.

Corollary 7.3. The 5-heap position $(1, 1, x, x, x + (2\ell + 1))$ for $1 \leq x$ and $0 \leq \ell$ is losing for p_1.

Proof. Take $k = 0$ in Lemma 7.2. □

Corollary 7.4. $(1, 1, 1, 2y + 1, (2y + 1) + (2z + 1))$ for $y \geq 0$ and $z \geq 0$ is losing for p_1.

Proof. Take $x = 1$, $k = 2y$, and $\ell = y + z$ in Lemma 7.2. □

Remark. We can rephrase the position in Lemma 7.2 as $(1, 1, x, y, z)$ where $1 \leq x \leq y \leq z$, $x \equiv y \pmod 2$, and $x \not\equiv z \pmod 2$. In essence, this result just says that when the first two coordinates are 1, and the remaining three do not have equal parity, then the position is losing.

Proposition 7.5. The position $(1, 1, x, y, z)$ for $1 \leq x \leq y \leq z$ is losing if and only if $x \equiv y \pmod 2$ and $x \not\equiv z \pmod 2$.

Proof. For the backwards direction, apply Lemma 7.2.

For the forward direction, we prove the contrapositive: Assume that $x \not\equiv y \pmod 2$ or $x \equiv z \pmod 2$. If $x \not\equiv y \pmod 2$ and $x \equiv z \pmod 2$, then $1 \leq x < y < z$ and p_1 takes a tack from the y and z heaps. If $x \not\equiv y \pmod 2$ and $x \not\equiv z \pmod 2$, then $1 \leq x < y \leq z$ and p_1 takes from the y heap. In each scenario, p_2 is in a losing position by Lemma 7.2. If $x \equiv y \pmod 2$ and $x \equiv z \pmod 2$, then $\mathbb{O}(\overline{x}) = 2$ or $\mathbb{O}(\overline{x}) = 5$. By Corollary 5.2 or Lemma 7.1 respectively, p_1 has a winning position. □

Lemma 7.6. Assume $(1, 2n+1, x, y, z)$ for $2n+1 \leq x \leq y \leq z$, $x \equiv y \pmod 2$, and $x \not\equiv z \pmod 2$ is a losing position. Then $(1, 2n+2, x', y', z')$ for $2n+2 \leq x' \leq y' \leq z'$ is a winning position.

Proof. We enumerate the possible cases:

- If $x' \equiv y' \pmod 2$ and $x' \not\equiv z' \pmod 2$, then p_1 takes from the second heap.
- If $x' \not\equiv y' \pmod 2$ and $x' \not\equiv z' \pmod 2$, then p_1 takes from the second and fourth heaps.
- If $x' \not\equiv y' \pmod 2$ and $x' \equiv z' \pmod 2$, then there are either two or three odd heaps.

In the first case, apply Corollary 5.2. In the second case, x' and z' are odd so $2n+3 \leq x' < y' < z'$. Then p_1 takes from the second and third heaps.

If $x' \equiv y' \pmod 2$ and $x' \equiv z' \pmod 2$, then there are either one or four odd heaps. In the first case, apply Corollary 5.2. In the second case, x', y', and z' are odd. Then p_1 takes from the first heap. Apply Theorem 5.7. □

Lemma 7.7. The position $(1, x_2, x_3, x_4, x_5)$ for $x_i \leq x_j$ when $i \leq j$ and all x_i odd is winning.

Remark. This generalizes 7.1, relaxing the constraint that $x_2 = 1$.

Proof. If p_1 takes from the the first and second heaps, the remaining position is equivalent to 4-heap position with three odd heaps and a minimal even heap. Apply Theorem 5.7. □

Lemma 7.8. The position $(1, 2n+1, x, y, z)$ for $2n+1 \leq x \leq y \leq z$, $x \equiv y \pmod 2$, and $x \not\equiv z \pmod 2$ is losing.

Proof. We will proceed via induction on n. As a base case, take $n = 0$ and apply Lemma 7.2. Assume that $(1, 2n+1, x, y, z)$ for $2n+1 \leq x \leq y \leq z$, $x \equiv y \pmod 2$, and $x \not\equiv z \pmod 2$ is a losing position. We will show that $(1, 2n+3, x', y', z')$ for $2n+3 \leq x' \leq y' \leq z'$, $x' \equiv y' \pmod 2$, and $x' \not\equiv z' \pmod 2$ is a losing position.

As a first move, p_1 should not take from the first and second heaps by Theorem 5.7. By Lemma 7.6 and the inductive hypothesis, $(1, 2n+2, x'', y'', z'')$ when $2n+2 \leq x'' \leq y'' \leq z''$ is winning so p_1 should not take from any heap with $2n+3$ tacks, which always includes the second heap. Further, p_1 should not take solely from the first heap since this would leave a 4-heap game with the odd minimum $2n+3$.

Thus p_1 must take from only the third, fourth, or fifth heaps. If $2n+3 < x'$, then p_2 can complement each move by p_1 to reduce the third, fourth, and fifth heaps by 1 tack while the third and fourth heaps retain the same parity and the third and fifth heaps retain opposite parity. Continuing the process, the third heap will eventually have $2n+3$ tacks. By the argument above,

p_1 should not take from the third heap at that point. Further, p_1 should not take from only the fifth heap by Lemma 7.7. Thus p_2 can mimic each move by p_1 to keep the fourth heap odd and the fifth heap even. Eventually, the fourth heap will have $2n+3$ elements and the fifth heap will have an even number of elements on p_1's turn. By the argument above, p_1 should not take from heaps 1 through 4. Taking from heap 5 puts p_2 in a winning position by Lemma 7.7. We conclude that any move for p_1 puts p_2 in a winning position. □

Proposition 7.9. Assume that $1 \leq x_i \leq x_j$ when $i \leq j$. The position $(1, x_2, x_3, x_4, x_5)$ is losing if and only if x_2 is odd, $x_3 \equiv x_4 \pmod 2$, and $x_3 \not\equiv x_5 \pmod 2$.

Proof. The only if direction is a simple application of Lemma 7.8. The if direction is proved by contrapositive: Assume that x_2 is even, $x_3 \not\equiv x_4 \pmod 2$, or $x_3 \equiv x_5 \pmod 2$. If x_2 is even, apply Lemma 7.6. We may assume that x_2 is odd for the remainder of the proof, which considers the possible cases:

- If $x_3 \not\equiv x_4 \pmod 2$ and $x_3 \equiv x_5 \pmod 2$, then $x_3 < x_4 < x_5$. Thus p_1 should take from the fourth and fifth heaps. Apply Lemma 7.8.
- If $x_3 \not\equiv x_4 \pmod 2$ and $x_3 \not\equiv x_5 \pmod 2$, then $x_3 < x_4 \leq x_5$. Thus p_1 can take from the fourth heap. Apply Lemma 7.8.
- If $x_3 \equiv x_4 \pmod 2$ and $x_3 \equiv x_5 \pmod 2$, then there are either two or five odd heaps. In the first case, apply Corollary 5.2. In the second case, apply Lemma 7.7.

In all scenarios, p_1 is in a winning position. □

This is all the machinery needed to proof proposition 5.8. The rest of the appendix applies these facts towards that objective.

Proof of Proposition 5.8. We will proceed via induction on x_1. The even base case $x_1 = 0$ is Proposition 5.7. The odd base case $x_1 = 1$ is Lemma 7.2. Assume the above result holds for any game with minimal heap x_1, and denote $\overline{x} = (x_1 + 1, x_2, x_3, x_4, x_5)$ and $\overline{y} = (x_3, x_4, x_5)$.

For the remainder of the proof we consider the possible relations between $x_1 + 1$ and $x_2 \pmod 2$, and there are three cases we must consider:

Assume first that $x_1 + 1 \not\equiv x_2 \pmod 2$. We consider all possible combinations for $\bar{y}$:

- $\mathbb{O}(\overline{y}) \in \{0, 1\}$: By corollary 5.2, p_1 has a winning position.
- $\mathbb{O}(\overline{y}) = 2 : p_1$ If x_1 is even, p_1 takes from If x_1. If $x_3 \equiv x_5$ or $x_4 \equiv x_5$, then she also picks one of them to ensure that condition (3) in the statement holds.
- $\mathbb{O}(\overline{y}) = 3 : \overline{y}$ has three odd heaps, p_1 can take from just the second heap, assuming x_1 is even. If x_1 is not even, p_1 takes from x_2 and x_5.

In the first case, p_1 is winning independent of the inductive hypothesis. In the second two, the reader can verify that these moves result in positions that are losing under the inductive hypothesis.

Assume that x_1+1 and x_2 are even. If $\overline{y}$ has zero odd heaps, apply Lemma 5.1. If $\overline{y}$ has one or two odd heaps, apply Corollary 5.2. Otherwise, suppose that $\overline{y}$ has three odd heaps. If p_1 takes from the first heap without taking from the second heap, p_2 is in a winning position by the inductive hypothesis. Further, if p_1 takes from the first and second heaps, then p_2 is in a winning position by the inductive hypothesis. If p_1 only removes from $\overline{y}$, then p_2 complements the move. The heaps in $\overline{y}$ will be all even so p_1 would be in a losing position. Thus p_1 must take from the second heap, but p_2 can mimic the move. The process repeats until the first and second heaps have the same number of tacks. The above argument implies that p_1 can no longer take from the second heap. Thus p_1 only has moves that put p_2 in a winning position.

The final case is that x_1+1 and x_2 are odd. If $\overline{y}$ has zero odd heaps, apply Lemma 5.1. If $\overline{y}$ has three odd heaps, then p_1 takes one tack from each of the first and second heaps. By the inductive hypothesis, p_2 has a losing position. In the other cases, p_1 cannot take from either or both of the two minimal heaps by the inductive hypothesis and the $x_1+1 \not\equiv x_2 \pmod 2$ case above. Thus p_1 must take from $\overline{y}$.

We consider all possible cases of the relationship between x_3 and x_4 and x_3 and $x_5 \pmod 2$.

- The hardest case is $x_3 \equiv x_4 \pmod 2$ and $x_3 \not\equiv x_5 \pmod 2$: We will show that this is a losing position for p_1. If $x_2 < x_3$, then p_2 can complement each move by p_1 to reduce the third, fourth, and fifth heaps by 1 tack each while keeping the third and fourth heaps at the same parity and the third and fifth heaps different parity. Eventually, the third heap will have x_2 tacks so p_1 should not take from the third heap. (Note that we can now drop the assumption that $x_2 < x_3$.) If p_1 takes only from the fifth heap, then p_2 can respond by taking from the first and second heaps, and p_1 loses by the inductive hypothesis. Thus p_1 does not take only from the fifth heap, and p_2 can mimic each move by p_1 to keep the fourth heap odd and the fifth heap even. Eventually, the fourth heap will have x_2 elements and the fifth heap will have an even number of elements on p_1's turn. The position $(x_1+1, x_2, x_2, x_2, x_5)$ is losing for x_1+1 odd, x_2 odd, and x_5 even. The argument above proves that p_1 should not take from any heap with x_1 or x_2 tacks so p_1 must take from only the fifth heap, making it odd. Then p_2 can take from the first and second heap, and p_1 is losing again by the inductive hypothesis.
- If $x_3 \equiv x_4 \pmod 2$ and $x_3 \equiv x_5 \pmod 2$, then there are either two or five odd heaps. In the first case, apply Corollary 5.2. In the second case, p_1 can take from the first and second heaps. Apply the inductive hypothesis.
- If $x_3 \not\equiv x_4 \pmod 2$ or $x_3 \equiv x_5 \pmod 2$, then we will show that p_1 is in a winning position.
- If $x_3 \not\equiv x_4 \pmod 2$ and $x_3 \not\equiv x_5 \pmod 2$, then $x_3 < x_4 \leq x_5$. Note that p_1 can take from the fourth heap to put p_2 in a losing position.

This concludes the proof.

□

References

[1] Elwyn Berlekamp, John Conway, and Richard Guy. *Winning ways for your mathematical plays*. 88 Worcester Street, Suite 230, Wellesley MA, 2482: A.K. Peters, 2004.

[2] Charles L. Bouton. "Nim, a Game with a Complete Mathematical Theory". In: *Annals of Mathematics* 3 (1901-1902), pp. 35–39.

[3] Thomas S Ferhuson. *Game Theory*. URL: `https://www.cs.cmu.edu/afs/cs/academic/class/15859-s05/www/ferguson/comb.pdf`.

[4] Vladimir Gurvich and Nhan Bao Ho. *Slow k-Nim*. 2015. arXiv: `1508.05777 [math.CO]`.

[5] Eliakim H. Moore. "A Generalization of the Game Called Nim". In: *Annals of Mathematics* 11 (1910), pp. 93–94.