More Games of No Chance
MSRI Publications
Volume **42**, 2002

A Simple FSM-Based Proof of the Additive Periodicity of the Sprague-Grundy Function of Wythoff's Game

HOWARD A. LANDMAN

ABSTRACT. Dress et al.[4] recently proved the additive periodicity of rows of the Sprague–Grundy function[7,5] of a class of Nim-like games including Wythoff's Game[8]. This implies that the function for each row minus its saltus is periodic and can be computed by a finite state machine.

The proof presented here, which was developed independently and in ignorance of the above result, proceeds in exactly the opposite direction by explicitly constructing a finite state machine with $o(n^2)$ bits of state to compute $\mathcal{H}(m,n) = \mathcal{G}(m,n) - m + 2n$, for all m and fixed n. From this, the periodicity of $\mathcal{H}$ and the additive periodicity of $\mathcal{G}$ follow trivially.

1. Introduction

Note: The original lecture on this material is available on streaming video at http://www.msri.org/publications/ln/msri/2000/gametheory/landman/1/.

Wythoff's Game (also called Wythoff's Nim or simply Wyt) is an impartial 2-player game played with 2 piles of counters. Each player may, on their turn, remove any number of counters from either pile, or remove the same number of counters from both piles. The player removing the last counter wins. The set of winning positions (those for which the Sprague–Grundy value $\mathcal{G}(m,n)$ equals 0) is well-known[3,1]. However, no direct formula is known for computing the Sprague–Grundy value $\mathcal{G}(m,n)$ of an arbitrary position with m counters in one pile and n in the other; it appears necessary to compute all the $\mathcal{G}(i,j)$-values for smaller games ($i \leq m,\ j \leq n,\ i+j < m+n$) first.

It is clearly impossible to compute the n-th row of $\mathcal{G}$ (that is, the sequence $G_n(m) = \mathcal{G}(m,n)$) directly with a finite state machine. One reason is that the values grow without bound, and thus the number of bits needed to represent the value will eventually exceed the capacity of any FSM. It also appears at first that the FSM would have to remember an ever-growing number of values which have already been used. The following lemmas provide bounds on $\mathcal{G}$ and allow us to overcome these obstacles.

2. Bounds on $\mathcal{G}(m, n)$

Lemma 1. $\mathcal{G}(m, n) \leq m + n$.

Proof: by induction on $m + n$. First observe that $\mathcal{G}(0, 0) = 0 \leq 0 + 0$. Assume that, for all i, j such that $i + j \leq m + n$, we have $\mathcal{G}(i, j) \leq i + j$. When calculating $\mathcal{G}(m, n)$, the set A of values to be excluded contains only values which are $< m + n$. So $\mathcal{G}(m, n) = \text{mex}(A) \leq \max(A) + 1 \leq (m + n - 1) + 1 = m + n$.

Lemma 2. $\mathcal{G}(m, n) \geq m - 2n$.

Proof: Assume $g = \mathcal{G}(m, n) < m - 2n$. This implies that g did not appear as any $\mathcal{G}(k, n)$ for $k < m$ (m times in all). Now there are only three reasons why g would not appear for any particular k. Either $\mathcal{G}(k, n) < g$, which can only happen at most g times, or g cannot appear because some $\mathcal{G}(k, j)$ for $j < n$ equals g, or g cannot appear because some $\mathcal{G}(k - i, n - i)$ for $0 \leq i \leq \min(k, n)$ equals g. But since g can appear at most once in each row, the second and third reasons can only happen at most n times each. So the total number of times g does not occur must be no more than $g + 2n < (m - 2n) + 2n = m$, which is a contradiction.

We note in passing that every non-negative integer g must eventually appear in each row, that is, for all g and n there exists m such that $g = \mathcal{G}(m, n)$. Not only must each g appear, but the above lemmas bound its position m: $g - n \leq m \leq g + 2n$. Since we also have (by definition of mex) that no integer can appear more than once, each row is a permutation of the non-negative integers.

3. Main Theorem

If we now define $\mathcal{H}(m, n) = \mathcal{G}(m, n) - m + 2n$, we can see from the above lemmas that $0 \leq \mathcal{H}(m, n) \leq 3n$ for all m. (Other saltus-adjusted functions besides $\mathcal{H}$ would also work for what follows, but this $\mathcal{H}$ seems one of the more natural choices.) If we know $\mathcal{H}$ we can compute $\mathcal{G}(m, n) = \mathcal{H}(m, n) + m - 2n$ quite easily. $\mathcal{G}$ is additively periodic iff $\mathcal{H}$ is periodic. And the values of $\mathcal{H}$ do not grow without bound, so the first obstacle is obviously overcome. What is less obvious but nonetheless true is that the second obstacle is also overcome.

Define the Left, Slanting, and Down sets: Let $L(m, n)$ be the set of integers which appear to the left of $\mathcal{G}(m, n)$, i.e. $L(m, n) = \{\mathcal{G}(m - k, n) : 1 \leq k \leq m\}$. Similarly let $S(m, n) = \{\mathcal{G}(m - k, n - k) : 1 \leq k \leq \min(m, n)\}$ and $D(m, n) = \mathcal{G}(m, n - k) : 1 \leq k \leq n$. L contains the $\mathcal{G}$-values corresponding to moves which remove counters from the pile of m, D to those which remove conters from the pile of n, and S to those which remove counters from both piles. We can calculate

$\mathcal{G}$ or $\mathcal{H}$ from L, S, and D, since $A(m,n) = L(m,n) \cup S(m,n) \cup D(m,n)$ is the set of elements to be excluded when calculating $\mathcal{G}$, that is, $\mathcal{G} = \text{mex}(A)$. And both S and D are bounded in size; they each never have more than n elements. Unfortunately $L(m,n)$ grows arbitrarily large as m increases, so it cannot be directly held in a FSM. We need to find a more clever and compact way to represent these sets.

Let $L'(m,n) = ((m-2n)\dots(m+n)) - L(m,n)$, which, by lemmas 1 and 2, is the set of all numbers $\leq m+n$ which are *not* in L. $L'(m,n)$ can be represented as a bit-array of $3n+1$ bits for all m, indicating which of the numbers from $m-2n$ to $m+n$ it contains. We similarly construct $S'(m,n)$ and $D'(m,n)$, which have the same size. By definition we have $\mathcal{G}(m,n) = \text{mex}(L(m,n) \cup S(m,n) \cup D(m,n)) = \min(L'(m,n) \cap S'(m,n) \cap D'(m,n))$.

To compute $\mathcal{H}(m,n)$ for all m, it is sufficient to keep track of the sets $L'(m,0)$, $L'(m,1), \dots, L'(m,n-1), L'(m,n), S'(m,0), S'(m,1), \dots, S'(m,n-1), S(m,n)$, and $D'(m,n)$. ($D'(m,0)$, $D'(m,1)$, $\dots$, $D'(m,n-1)$ are all subsets of $D'(m,n)$ and do not need to be stored separately if the bit-array representation is used.) This gives $o(n)$ sets which are each $o(n)$ bits, for a total number of bits which is $o(n^2)$. Only the sets $L'(m,n)$, $S'(m,n)$, and $D'(m,n)$ are needed directly to compute $\mathcal{H}(m,n)$ and $L'(m+1,n)$, but the other sets are needed to compute e.g. $S'(m+1,n)$ and $D'(m+1,n)$.

For example, to calculate $L'(m+1,k)$, we take $L'(m,k)$, unset the bit corresponding to $\mathcal{H}(m,k)$, shift over 1, and set the end bit corresponding to $m+k+1$. This is guaranteed never to shift a set bit off the other end (due to lemma 2). so the number of set bits in $L'(m,k)$ is constant (equal to $k+1$) as m varies. Similarly, $S'(m+1,k)$ can be calculated from $S'(m,k-1)$ and $\mathcal{H}(m,k)$. The $D'(m+1,k)$ can be computed "bottom up" in sequence starting from $k=0$.

In addition to the above storage requirements, a program to calculate $\mathcal{H}$ by this method might also need one or more variables whose value is bounded by n (for example, to do the counting from $k=0$ to n while computing D'). However, a finite number of these suffice, and each of them requires only $o(\log(n))$ bits. Therefore the n-th row can be computed by a finite state machine with $b = o(n^2)$ total bits of state.

This result tells us immediately that $\mathcal{H}(m,n)$ is eventually periodic for fixed n, since after no more than 2^b steps it must reenter a state previously visited and thereafter loop. Therefore $G_n(m)$ must be additively periodic.

4. Conclusion

The key point in the above proof was the finiteness of storage requirements. It was critical to show that the storage required did not depend on m, which increases, but rather only on n, which is constant as m varies. The precise amount of storage, or even its being $o(n^2)$, was not really important.

From the computational point of view, it is much simpler to prove periodicity of a bounded function than additive periodicity of an unbounded function, even though the two may be logically equivalent. This is because periodicity is equivalent to computability by a finite state machine; while additive periodicity requires a machine with unbounded storage, placing it in a different and harder computational complexity class.

The technique presented here should be generalizable to many other impartial games, such as the various proposed extensions of Wythoff's Game to more than 2 piles[2,6].

5. Bibliography

[1] U. Blass and A. S. Fraenkel [1990], The Sprague–Grundy function of Wythoff's game, *Theoret. Comput. Sci.* (Math Games) **75**, 311–333.

[2] I. G. Connell [1959], A generalization of Wythoff's game, *Canad. Math. Bull.* **2**, 181–190.

[3] H. S. M. Coxeter [1953], The golden section, phyllotaxis and Wythoff's game, *Scripta Math.* **19**, 135–143.

[4] A. Dress, A. Flammenkamp and N. Pink [1999], Additive periodicity of the Sprague–Grundy function of certain Nim games, *Adv. in Appl. Math.* **22**, 249–270.

[5] P. M. Grundy [1964], Mathematics and Games, *Eureka* **27**, 9–11, originally published: *ibid.* **2** (1939), 6–8.

[6] A. S. Fraenkel and I. Borosh [1973], A generalization of Wythoff's game, *J. Combin. Theory* (Ser. A) **15**, 175–191.

[7] R. Sprague [1935–36], Über mathematische Kampfspiele, *Tôhoku Math. J.* **41**, 438–444.

[8] W. A. Wythoff [1907], A modification of the game of Nim, *Nieuw Arch. Wisk.* **7**, 199–202.

Howard A. Landman
Fort Collins, CO 80521
howard@riverrock.org