Games of No Chance 3
MSRI Publications
Volume **56**, 2009

More on the Sprague–Grundy function for Wythoff's game

GABRIEL NIVASCH

ABSTRACT. We present two new results on Wythoff's Grundy function $\mathcal{G}$. The first one is a proof that for every integer $g \geq 0$, the g-values of $\mathcal{G}$ are within a bounded distance to their corresponding 0-values. Since the 0-values are located roughly along two diagonals, of slopes ϕ and ϕ^{-1}, the g-values are contained within two strips of bounded width around those diagonals. This is a generalization of a previous result by Blass and Fraenkel regarding the 1-values.

Our second result is a *convergence* conjecture and an accompanying recursive algorithm. We show that for every g for which a certain conjecture is true, there exists a recursive algorithm for finding the n-th g-value in $O(\log n)$ arithmetic operations. Our algorithm and conjecture are modifications of a similar result by Blass and Fraenkel for the 1-values. We also present experimental evidence for our conjecture for small g.

1. Introduction

The game of Wythoff [10] is a two-player impartial game played with two piles of tokens. On each turn, a player removes either an arbitrary number of tokens from one pile (between one token and the entire pile), or the same number of tokens from both piles. The game ends when both piles become empty. The last player to move is the winner.

Wythoff's game can be represented graphically with a quarter-infinite chessboard, extending to infinity upwards and to the right (Figure 1). We number the rows and columns sequentially $0, 1, 2, \ldots$. A chess queen is placed in some cell

This work was done when the author was at the Weizmann Institute of Science in Rehovot, Israel. The author was partially supported by a grant from the German-Israeli Foundation for Scientific Research and Development.

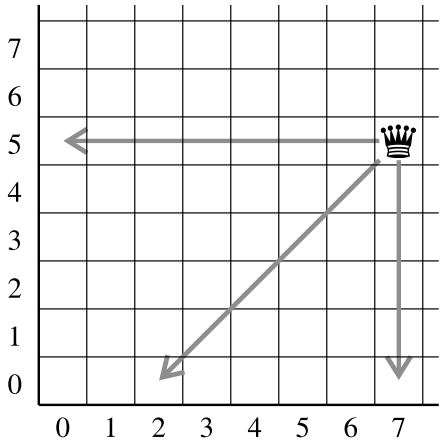

Figure 1. Graphic representation of Wythoff's game.

of the board. On each turn, a player moves the queen to some other cell, except that the queen can only move left, down, or diagonally down-left. The player who takes the queen to the corner wins.

Wythoff found a simple, closed formula for the P-positions of his game. Let $\phi = (1 + \sqrt{5})/2$ be the golden ratio. Then:

THEOREM 1.1 (WYTHOFF [10]). *The P-positions of Wythoff's game are given by*

$$(\lfloor \phi n \rfloor, \lfloor \phi^2 n \rfloor) \quad \text{and} \quad (\lfloor \phi^2 n \rfloor, \lfloor \phi n \rfloor), \tag{1-1}$$

for $n = 0, 1, 2, \ldots$.

1.1. Impartial games and the Sprague–Grundy function. We briefly review the Sprague–Grundy theory of impartial games [1].

An impartial game can be represented by a directed acyclic graph $G = (V, E)$. Each position in the game corresponds to a vertex in G, and edges join vertices according to the game's legal moves. A token is initially placed on some vertex $v \in V$. Two players take turns moving the token from its current vertex to one of its direct followers. The player who moves the token into a sink wins.

Given two games G_1 and G_2, their *sum* $G_1 + G_2$ is played as follows: On each turn, a player chooses one of G_1, G_2, and moves on it, leaving the other game untouched. The game ends when no moves are possible on G_1 nor on G_2.

Let $\mathbb{N} = \{0, 1, 2, \ldots\}$ be the set of natural numbers. Given a finite subset $S \subset \mathbb{N}$, let $\operatorname{mex} S = \min(\mathbb{N} \setminus S)$ denote the smallest natural number not in S. Then, given a game $G = (V, E)$, its *Sprague–Grundy function* (or just *Grundy function*) $\mathcal{G} : V \to \mathbb{N}$ is defined recursively by

$$\mathcal{G}(u) = \operatorname{mex}\{\mathcal{G}(v) \mid (u, v) \in E\}, \qquad \text{for } u \in V. \tag{1-2}$$

This recursion starts by assigning sinks the value 0.

The Grundy function $\mathcal{G}$ satisfies the following two important properties:

1. A vertex $v \in V$ is a P-position if and only if $\mathcal{G}(v) = 0$.
2. If $G = G_1 + G_2$ and $v = (v_1, v_2) \in V_1 \times V_2$, then $\mathcal{G}(v)$ is the bitwise XOR, or *nim-sum*, of the binary representations of $\mathcal{G}(v_1)$ and $\mathcal{G}(v_2)$.

Clearly, the sum of games is an associative operation, as is the nim-sum operation. Therefore, knowledge of the Grundy function provides a winning strategy for the sum of any number of games.

The following lemma gives some basic bounds on the Grundy function.

LEMMA 1.2. *Given a vertex v, let n_v be the number of direct followers of v, and let p_v be the number of edges in the longest path from v to a leaf. Then $\mathcal{G}(v) \le n_v$ and $\mathcal{G}(v) \le p_v$.*

PROOF. The first bound follows trivially from equation (1-2). The second bound follows by induction. □

The following lemma shows that the Grundy function of a game-graph can be calculated up to a certain value g using the mex property.

LEMMA 1.3. *Given a game-graph $G = (V, E)$ and an integer $g \ge 0$, let $\mathcal{H}$ be a function $\mathcal{H} : V \to \{0, \ldots, g, \infty\}$ such that for all $v \in V$,*

1. *if $\mathcal{H}(v) \le g$ then*

$$\mathcal{H}(v) = \operatorname{mex}\{\mathcal{H}(w) \mid (v, w) \in E\};$$

2. *if $\mathcal{H}(v) = \infty$ then*

$$\operatorname{mex}\{\mathcal{H}(w) \mid (v, w) \in E\} > g.$$

Then $\mathcal{G}(v) = \mathcal{H}(v)$ whenever $\mathcal{H}(v) \le g$, and $\mathcal{G}(v) > g$ whenever $\mathcal{H}(v) = \infty$.

In the function $\mathcal{H}$, the labels ∞ are placeholders that indicate values larger than g. Theorem 1.3 is easily proven by induction; see [6].

1.2. Notation and terminology. From now on, $\mathcal{G}$ will denote specifically the Grundy function of Wythoff's game. Thus, $\mathcal{G} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is given by

$$\begin{aligned}\mathcal{G}(x, y) = \operatorname{mex}\big(&\{\mathcal{G}(x', y) \mid 0 \le x' < x\} \cup \\ &\{\mathcal{G}(x, y') \mid 0 \le y' < y\} \cup \\ &\{\mathcal{G}(x-k, y-k) \mid 1 \le k \le \min(x, y)\}\big).\end{aligned} \tag{1-3}$$

Table 1 shows the value of $\mathcal{G}$ for small x and y. This matrix looks quite chaotic at first glance, as has been pointed out before [1]. And indeed, it is still an open problem to compute $\mathcal{G}$ in time polynomial in the size of the input (meaning, to compute $\mathcal{G}(x, y)$ in time $O((\log x + \log y)^c)$ for some constant c).

15	15	16	17	18	10	13	12	19	14	0	3	21	22	8	23	20
14	14	12	13	16	15	17	18	10	9	1	2	20	21	7	11	23
13	13	14	12	11	16	15	17	2	0	5	6	19	20	9	7	8
12	12	13	14	15	11	9	16	17	18	19	7	8	10	20	21	22
11	11	9	10	7	12	14	2	13	17	6	18	15	8	19	20	21
10	10	11	9	8	13	12	0	15	16	17	14	18	7	6	2	3
9	9	10	11	12	8	7	13	14	15	16	17	6	19	5	1	0
8	8	6	7	10	1	2	5	3	4	15	16	17	18	0	9	14
7	7	8	6	9	0	1	4	5	3	14	15	13	17	2	10	19
6	6	7	8	1	9	10	3	4	5	13	0	2	16	17	18	12
5	5	3	4	0	6	8	10	1	2	7	12	14	9	15	17	13
4	4	5	3	2	7	6	9	0	1	8	13	12	11	16	15	10
3	3	4	5	6	2	0	1	9	10	12	8	7	15	11	16	18
2	2	0	1	5	3	4	8	6	7	11	9	10	14	12	13	17
1	1	2	0	4	5	3	7	8	6	10	11	9	13	14	12	16
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Table 1. The Grundy function of Wythoff's game.

Nevertheless, as we will see now, several results on $\mathcal{G}$ have been established. But let us first introduce some notation.

A pair (x, y) is also called a *point* or a *cell*. If $\mathcal{G}(x, y) = g$, we call (x, y) a *g-point* or a *g-value*.

Note that by symmetry, $\mathcal{G}(x, y) = \mathcal{G}(y, x)$ for all x, y. We refer to this property as *diagonal symmetry*.

We will consistently use the following graphical representation of $\mathcal{G}$: The first coordinate of $\mathcal{G}$ is plotted vertically, increasing upwards, and the second coordinate is plotted horizontally, increasing to the right.

Thus, we call *row r* the set of points (r, x) for all $x \geq 0$, and *column c* the set of points (x, c) for all $x \geq 0$. Also, *diagonal e* is the set of points $(x, x + e)$ for all x if $e \geq 0$, or the set of points $(x - e, x)$ for all x if $e < 0$. (Note that we only consider diagonals parallel to the movement of the queen.)

We also define, for every integer $g \geq 0$, the sequence of g-values that lie on or to the right of the main diagonal, sorted by increasing row. Formally, we let

$$T_g = \left((a_0^g, b_0^g), (a_1^g, b_1^g), (a_2^g, b_2^g), \ldots\right) \tag{1-4}$$

be the sequence of g-values having $a_n^g \leq b_n^g$, ordered by increasing a_n^g. We also

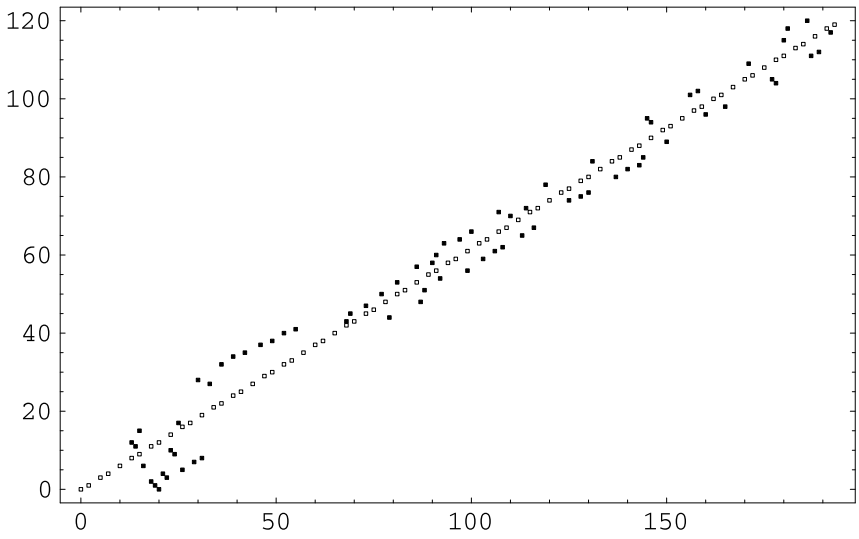

Figure 2. First terms of the sequences T_0 (framed squares) and T_{20} (filled squares).

let $p_n^g = (a_n^g, b_n^g)$. Note that, by Theorem 1.1, we have

$$p_n^0 = (\lfloor \phi n \rfloor, \lfloor \phi^2 n \rfloor).$$

For example, Figure 2 plots the first points in T_0 and T_{20}. A pattern is immediately evident: The 20-values seem to lie within a strip of constant width around the 0-values. This is true in general. In fact, we will prove something stronger as one of the main results of this paper.

We also let $d_n^g = b_n^g - a_n^g$ be the diagonal occupied by the point p_n^g.

Finally, we place the a, b, and d coordinates of g-points into sets, as follows:

$$A_g = \{a_i^g \mid i \geq 0\}, \quad B_g = \{b_i^g \mid i \geq 0\}, \quad D_g = \{d_i^g \mid i \geq 0\}. \tag{1-5}$$

1.3. Previous results on Wythoff's Grundy function. We now give a brief overview of previous results on the Grundy function of Wythoff's game.

It follows directly from formula (1-3) that no row, column, or diagonal of $\mathcal{G}$ contains any g-value more than once. In fact, it is not hard to show that every row and column of $\mathcal{G}$ contains every g-value exactly once [2; 4]. Furthermore, every diagonal contains every g-value exactly once [2], although this is somewhat harder to show. We will rederive these results in this paper.

It has also been shown that every row of $\mathcal{G}$ is *additively periodic*. This result was first proven by Norbert Pink in his doctoral thesis [8] (published in [3]), and Landman [4] later found a simpler proof. Both [3] and [4] derive an upper bound of $2^{O(x^2)}$ for the preperiod and the period of row x.

The additive periodicity of $\mathcal{G}$ has important computational implications: It means that $\mathcal{G}(x, y)$ can be computed in time

$$2^{O(x^2)} + O(x^2 \log y),$$

which is linear in $\log y$ for constant x. See [6] for details.

Blass and Fraenkel [2] obtained several results on the sequence T_1 of 1-values, as defined above (1-4). They showed that the n-th 1-value is within a bounded distance to the n-th 0-value. Specifically,

$$\begin{aligned} 8 - 6\phi \;<\; a_n^1 - \phi n \;<\; 6 - 3\phi\,, \\ -3\phi \;<\; b_n^1 - \phi^2 n \;<\; 8 - 3\phi \end{aligned} \tag{1-6}$$

(Theorem 5.6, Corollary 5.13 in [2]).

They also presented a recursive algorithm for computing the n-th 1-value given n. We will not get into the details of the algorithm, but suffice it to say that the recursion is carried out to a logarithmic number of levels. Further, if the computation done at each level were shown to be constant, then the algorithm would run in $O(\log n)$ steps altogether. For the computation at each level to be constant, certain arrangements in the sequence of 0- and 1-values must occur infinitely many times with at least constant regularity. The authors did not manage to prove this latter property, so they left the polynomiality of their algorithm as a conjecture.

1.4. Our results. In this paper we make two main contributions on the function $\mathcal{G}$. The first one is a generalization of the result for the 1-values described above. We will prove that for every g, the point p_n^g is within a bounded distance to p_n^0, where the bound depends only on g, not on n. Our theoretical bound turns out to be much worse than the actual distances seen in practice. We present experimental data and compare them to our theoretical result.

Our second contribution is a modification and generalization of Blass and Fraenkel's recursive algorithm. We present a conjecture, called the *Convergence Conjecture*, which claims a certain property about the sequences T_0 through T_g. We show that for every g for which the conjecture is true, there exists an algorithm that computes p_n^g in $O(\log n)$ arithmetic operations, where the factor implicit in the O notation depends on g.

We present experimental results that seem to support the conjecture for small g. We finally use our recursive algorithm to predict the value of several points p_n^g for small g and very large n.

1.5. Significance of our results. Suppose we are playing the sum of Wythoff's game with some other game, like a Nim pile. Our winning strategy, then, is to make the Grundy values of the two games equal. Suppose that the position in

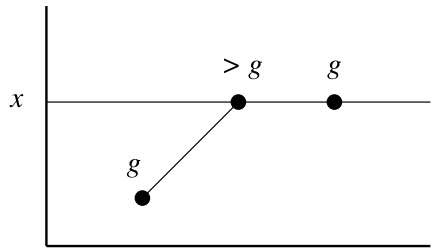

Figure 3. A supporter from a row lower than x.

Wythoff has Grundy value m, and the Nim pile is of size n. Then, if $m < n$, we should reduce the Nim pile to size m, and if $m > n$, we should move in Wythoff to a position with Grundy value n.

How do the results in this paper help us in this scenario? The first result, regarding the location of the g-values, is of no practical help: It gives us only the approximate location of the g-values, not their precise position.

The recursive algorithm, on the other hand, has much more practical significance. If the conjecture is true for small values of g, then we can play on sums where the Nim pile is of size $\leq g$. And even if there are sporadic counterexamples to the conjecture, the recursive algorithm will probably give the correct answer in most cases, so it is a good heuristic.

1.6. Organization of this paper. The rest of this paper is organized as follows. In Section 2 we prove the closeness of the g-values to the 0-values, and in Section 3 we present the Convergence Conjecture and its accompanying recursive algorithm. The Appendix (page 407) contains some proofs omitted from Section 2.5.

2. Closeness of the g-values to the 0-values

In this Section we will prove that for every g, the point p_n^g is within a bounded distance to p_n^0, where the bound depends only on g. We begin with some basic results on $\mathcal{G}$.

LEMMA 2.1 (LANDMAN [4]). *Given x and g, there exists a unique y such that $\mathcal{G}(x, y) = g$. Moreover,*

$$g - x \leq y \leq g + 2x. \tag{2-1}$$

PROOF. Uniqueness follows trivially from the definition of $\mathcal{G}$.

As for existence, for any x, y, the longest path from cell (x, y) to the corner $(0, 0)$ has length $x + y$, so by Lemma 1.2, $g \leq x + y$, implying the lower bound in (2-1).

Next, let y be an integer such that $G(x, y') \neq g$ for all $y' < y$. At most g such points (x, y') have a value smaller than g, so at least $y - g$ of them have a value larger than g. Each of the latter points must be "supported" by a g-point in a lower row, either vertically of diagonally down (see Figure 3). No two such supporters can share a row, so there are at most x of them. On the other hand, each supporter can support at most two points in row x. Therefore, $y - g \leq 2x$, yielding the upper bound of (2-1). □

The following result was also known already to Landman [5]:

LEMMA 2.2. *Given $g \geq 0$, there exists a unique integer x such that $G(x, x) = g$. Moreover*,

$$g/2 \leq x \leq 2g. \tag{2-2}$$

PROOF. As before, uniqueness follows from the definition of G. And the lower bound follows from equation (2-1).

For the upper bound, suppose x is such that $G(y, y) \neq g$ for all $y < x$. At most g of such points (y, y) have value smaller than g, so at least $x - g$ of them have value larger than g. By diagonal symmetry, for each of the latter points there exist two g-points, one to the left of (y, y) and the other one below it.

Therefore, there are at least $2(x - g)$ g-points below and to the left of point (x, x). No two of them can share a column, so $2(x - g) \leq x$, or $x \leq 2g$. □

Recall the definition of the sets A_g, B_g, and D_g (1-5). By Lemmas 2.1 and 2.2, together with diagonal symmetry, we have $|A_g \cap B_g| = 1$ and $A_g \cup B_g = \mathbb{N}$ for all g. Therefore, we could say that the sequences $\{a_i^g\}$ and $\{b_i^g\}$ are "almost complementary". We will show later on that $D_g = \mathbb{N}$ for all g.

2.1. Algorithm WSG for computing T_g. On page 385 we show a greedy algorithm that computes, given an integer $g \geq 0$, the sequences T_h and the sets A_h, B_h, D_h for $0 \leq h \leq g$. This algorithm was first described in [2].

The idea behind the algorithm is simple: We traverse the rows in increasing order, and for each row, we go through the values $h = 0, \ldots, g$ in increasing order. If the current row needs an h-point on or to the right of the main diagonal (because it contains no h-point to the left of the main diagonal), then we greedily find the first legal place for an h-point, and insert the h-point there. We also reflect the h-point with respect to the main diagonal, into a higher row (so that higher row will not receive an h-point to the right of the main diagonal).

Of course, since this algorithm works row by row, it takes exponential time.

For simplicity, we do not specify when the algorithm halts, although we could make it halt after, say, computing the first n terms of T_g.

We will rely heavily on this algorithm later on, in our analysis of T_g.

The correctness of Algorithm WSG follows easily from Lemma 1.3; see [6] for the details.

Algorithm WSG (Wythoff Sprague–Grundy)

1. Initialize the sets A_h, B_h, D_h, and the sequences T_h, to $\varnothing$, for $0 \le h \le g$.
2. For $r = 0, 1, 2, \ldots$ do:
3. For $h = 0, \ldots, g$ do:
4. If $r \notin B_h$ then:
5. • find the smallest $d = 0, 1, 2, \ldots$ for which:
6. ∘ $(r, r + d) \notin T_k$ for all $0 \le k < h$,
7. ∘ $r + d \notin B_h$, and
8. ∘ $d \notin D_h$;
9. • append $(r, r + d)$ to T_h;
10. • insert r into A_h;
11. • insert $r + d$ into B_h;
12. • insert d into D_h.

2.2. Statement of the main Theorem. Recall from Theorem 1.1 that the 0-values of Wythoff's game are given by

$$(a_n^0, b_n^0) = (\lfloor \phi n \rfloor, \lfloor \phi^2 n \rfloor).$$

Graphically, the 0-values lie close to a straight line of slope ϕ^{-1} that starts at the origin.

Our main result for this Section is the following:

THEOREM 2.3. *For every Grundy value $g \ge 0$ and every diagonal $e \ge 0$, there exists an n such that*

$$d_n^g = e$$

(*i.e., every diagonal e contains a g-value*).

Further, for every $g \ge 0$ there exist constants α_g, β_g, such that

$$|a_n^g - a_n^0| \le \alpha_g, \quad |b_n^g - b_n^0| \le \beta_g, \quad \textit{for all } n$$

(*i.e., the n-th g-value is close to the n-th 0-value*).

Our strategy for proving Theorem 2.3 is as follows. We first show that for every g there is a g-value in every diagonal, and furthermore, for every g there exists a constant δ_g such that

$$|d_n^g - n| \le \delta_g \quad \text{for all } n. \tag{2-3}$$

In other words, the g-values occupy the diagonals in roughly sequential order.

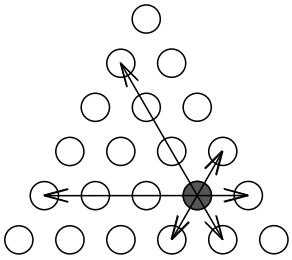

Figure 4. Queen in a triangular lattice.

Then we show how equation (2-3), together with the almost-complementarity of the sequences $\{a_n^g\}$ and $\{b_n^g\}$, implies that $|a_n^g - \phi n|$ and $|b_n^g - \phi^2 n|$ are bounded.

Note that for $g = 0$, Theorem 1.1 gives us

$$d_n^0 = b_n^0 - a_n^0 = (\lfloor \phi n \rfloor + n) - \lfloor \phi n \rfloor = n;$$

in other words, the 0-values occupy the diagonals in sequential order. This can be confirmed easily by following Algorithm WSG with $g = 0$.

2.3. Nonattacking queens on a triangle. The following is a variation on the well-known "eight queens problem". We will use its solution in proving bound (2-3).

We are given a triangular lattice of side n, as shown in Figure 4. A queen on the lattice can move along a straight line parallel to any of the sides of the triangle. How many queens can be placed on the lattice, without any two queens attacking each other?

LEMMA 2.4. *The maximum number of nonattacking queens that can be placed on a triangular lattice of side n is exactly*

$$q(n) = \left\lfloor \frac{2n+1}{3} \right\rfloor.$$

A simple proof of this fact is given by Vaderlind et al. in [9, Problem 252]. Another proof is given in [7].

2.4. d_n^g is close to n. We will now prove bound (2-3). For convenience, in this subsection we fix g, and we write $a_n = a_n^g$, $b_n = b_n^g$, $d_n = d_n^g$, $p_n = p_n^g$. Whenever we refer to h-points, $h < g$, we will say so explicitly.

Recall that the points p_n are ordered by increasing row a_n, so that $a_n > a_m$ if and only if $n > m$.

In this subsection we will make extensive use of Algorithm WSG.

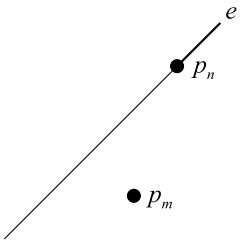

Figure 5. Point p_m skips diagonal e.

Observe that Algorithm WSG does not place a g-point on certain rows r, because it skips row r on line 4. Then such an r is not added to A_g (line 10). We call such an r a *skipped row*.

Similarly, sometimes a certain column c is never inserted into B_g (line 11), because no point p_m falls on that column. In that case, we call column c a *skipped column*.

Let us define the notion of a g-point skipping a diagonal. Intuitively, point p_m skips diagonal e if Algorithm WSG places point p_m to the right of diagonal e, while diagonal e does not yet contain a g-point (see Figure 5). Formally:

DEFINITION 2.5. Diagonal e is *empty up to row* r if there is no point p_n with $d_n = e$ and $a_n \leq r$.

DEFINITION 2.6. Point p_m is said to *skip diagonal* e if $d_m > e$ and e is empty up to row a_m.

Our goal in this subsection is to derive a bound on $|d_n - n|$. We will derive separately upper bounds for $d_n - n$ and $n - d_n$. We do this by bounding the number of diagonals that a given point can skip, and the number of points that can skip a given diagonal:

LEMMA 2.7. *If no point p_n skips more than k diagonals, then $d_n - n \leq k$ for all n. If no diagonal is skipped by more than k points, then $n - d_n \leq k$ for all n.*

PROOF. For the first claim, suppose by contradiction that $d_n - n > k$ for some n. Then, of the d_n diagonals $0, \ldots, d_n - 1$, only n can be occupied by points $p_0, \ldots, p_{n-1}$. Therefore, point p_n skips at least $k + 1$ diagonals.

For the second claim, suppose by contradiction that $n - d_n > k$ for some n. Then, of the n points $p_0, \ldots, p_{n-1}$, only d_n can fall on diagonals $0, \ldots, d_n - 1$. Therefore, diagonal d_n is skipped by at least $k + 1$ points. □

Let us inspect why a g-point skips a diagonal according to Algorithm WSG. Suppose point p_m skips diagonal e, and let $C = (a_m, a_m + e)$ be the cell on diagonal e on the row in which p_m was inserted. Then, point p_m skipped diagonal e, either because cell C was already assigned some value $k < g$ (WSG

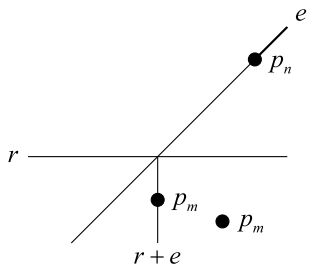

Figure 6. Points p_m active with respect to diagonal e and row r.

line 6), or because there was already a point $p_{m'}$ directly below cell C (WSG line 7).

We need a further definition: Let e be a diagonal and r be a row, such that diagonal e is empty up to row $r-1$. Draw a line from the intersection of row r and diagonal e vertically down. If a point p_m is strictly below row r, and on or to the right of the said vertical line, then we say that p_m is *active with respect to diagonal e and row r* (see Figure 6). In other words:

DEFINITION 2.8. If diagonal e is empty up to row $r-1$, then point p_m is *active with respect to diagonal e and row r* if $a_m < r$ and $b_m \geq r+e$.

We can bound the number of active g-points:

LEMMA 2.9. *The number of g-points active with respect to any diagonal e and any row r is at most g.*

PROOF. By assumption, diagonal e is empty up to row $r-1$.

We will show that for every $r' \leq r$, if diagonal e contains k h-points, $h < g$, below row r', then there can be at most k active g-points with respect to diagonal e and row r'. This implies our Lemma, since there are at most g h-points, $h < g$, on diagonal e.

We prove the above claim by induction on r'. If $r' = 0$ then clearly $k = 0$ and there are no active g-points with respect to e and r'.

Suppose our claim is true up to row r', and let us examine Algorithm WSG on row r' itself. If no point p_m is inserted on row r', then the number of active g-points does not increase when we go from row r' to row $r'+1$. And if point p_m is inserted on row r' and it skips diagonal e, it must be for one of the two reasons mentioned above. If there is an h-point, $h < g$, on the intersection of row r' with diagonal e, then the number k of our claim increases by 1 when we go to row $r'+1$. And if there is no such h-point, then there must be an earlier point $p_{m'}$ directly below the intersection of e and r'. But then $p_{m'}$ is active with respect to e and r', but not with respect to e and $r'+1$, so the number of active g-points stays the same when we go from row r' to row $r'+1$.

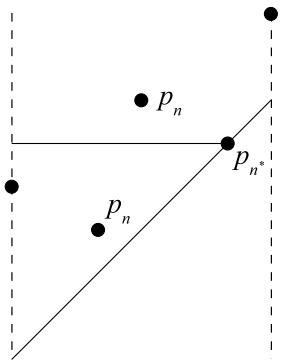

Figure 7. p_{n^*} is the point p_n with maximum d_n.

So in either case, the inductive claim is also true for row $r'+1$. □

We can now bound the number of diagonals a given g-point can skip:

LEMMA 2.10. *A point p_m can skip at most $2g$ diagonals.*

PROOF. Let e_0 be the first diagonal skipped by point p_m. For every diagonal e skipped by p_m, there must be either an active g-point with respect to diagonal e_0 and row a_m, or an h-point, $h<g$, on cell (a_m, a_m+e). There can be at most g of the latter, and by Lemma 2.9, at most g of the former. □

We proceed to bound the number of g-points that can skip a given diagonal. For this we need a lower bound on the number of skipped columns in an interval of consecutive columns:

LEMMA 2.11. *An interval of k consecutive columns contains at least $k/3-2g$ skipped columns.*

PROOF. Consider the points p_n that lie within the given interval of columns. Let p_{n^*} be the point p_n with maximum d_n (see Figure 7). The number of points p_n with $n>n^*$ is at most $2g$ by Lemma 2.10. And the points p_n with $n<n^*$ are confined to a triangular lattice; but this situation is isomorphic to the nonattacking queens of subsection 2.3!

The triangular lattice has side at most $k-2$ (since the lattice cannot reach the column of p_{n^*} nor the preceding column). Therefore, the number of points p_n with $n<n^*$ is at most

$$\left\lfloor \frac{2(k-2)+1}{3} \right\rfloor \le 2k/3-1.$$

Thus, the total number of points p_n is at most $2k/3-1+2g+1=2k/3+2g$, so the number of skipped columns is at least $k/3-2g$. □

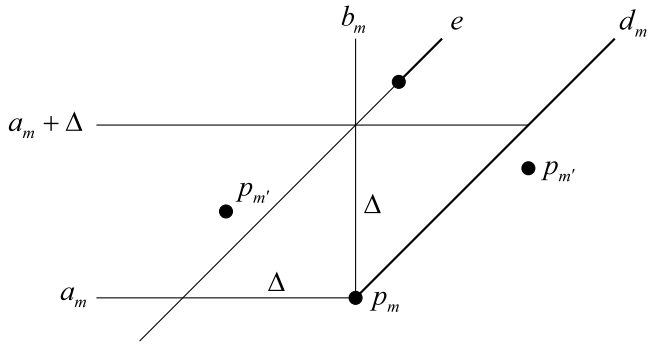

Figure 8. Points $p_{m'}$ between rows a_m and $a_m + \Delta$.

COROLLARY 2.12. *An interval of k consecutive rows contains at least $k/3-2g$ points p_n.*

PROOF. By diagonal symmetry: If column c is a skipped column, then row c contains a point p_n. □

LEMMA 2.13. *Suppose point p_m skips diagonal e, and let $\Delta = d_m - e$. Suppose diagonal e is empty up to row $a_m + \Delta$ (see Figure 8). Then $\Delta \leq 15g$.*

PROOF. Let us bound the number of points $p_{m'}$ in the interval between row $a_m + 1$ and row $a_m + \Delta$. For every such $p_{m'}$, either $d_{m'} < d_m$, or $b_{m'} > b_m$, or both (see Figure 8). In the former case, p_m skips diagonal $d_{m'}$, and in the latter case, $p_{m'}$ is active with respect to diagonal e and row $a_m + \Delta + 1$. So by Lemmas 2.9 and 2.10, there are at most $3g$ such points $p_{m'}$.

Therefore, by Corollary 2.12, we must have

$$\Delta/3 - 2g \leq 3g,$$

so $\Delta \leq 15g$. □

COROLLARY 2.14. *The number of points p_n that can skip a given diagonal e is at most $16g$.*

PROOF. Every point that skips diagonal e must lie on a different diagonal. Therefore, Lemma 2.13 already implies that diagonal e must eventually be occupied by a point p_n.

Now, consider the points p_m, $m < n$, that skip diagonal $e = d_n$. Partition these points into two sets: those having $b_m > b_n$ and those having $b_m < b_n$.

By Lemma 2.9, there are at most g points in the first set, since each such point is active with respect to the diagonal and row of p_n. And every point in the second set satisfies the assumptions of Lemma 2.13, so there are at most $15g$ such points. Therefore, diagonal e is skipped by no more than $16g$ points. □

We used somewhat messy arguments, but we have finally proven:

THEOREM 2.15. *For every Grundy value g and every diagonal e, there exists a g-point with $d_n^g = e$. Furthermore,*

$$-16g \le d_n^g - n \le 2g. \qquad \square$$

2.5. The g-values are close to the 0-values. We proceed to show the existence of the constants α_g and β_g of Theorem 2.3. In order to understand the idea behind our proof, it is helpful to look first at the following proof that the ratio between consecutive Fibonacci numbers tends to ϕ:

CLAIM 2.16. *Let F_n be the n-th Fibonacci number. Then $F_n/F_{n-1} \to \phi$.*

PROOF. Let $x_n = F_n - \phi F_{n-1}$. Then,

$$\begin{aligned} x_{n+1} = F_{n+1} - \phi F_n &= (F_n + F_{n-1}) - \phi F_n \\ &= -\phi^{-1}(F_n - \phi F_{n-1}) = -\phi^{-1} x_n. \end{aligned} \tag{2-4}$$

Therefore, $x_n \to 0$, since $|-\phi^{-1}| < 1$. Therefore,

$$\frac{F_n}{F_{n-1}} = \frac{x_n}{F_{n-1}} + \phi \to \phi. \qquad \square$$

Next, we introduce the following notation, which will help make our arguments clearer:

DEFINITION 2.17. Given sequences $\{f_n\}$ and $\{g_n\}$, we write

$$f_n \sim g_n$$

if, for some k, $|f_n - g_n| \le k$ for all n.

Note that the relation $\sim$ is transitive: If $f_n \sim g_n$ and $g_n \sim h_n$, then $f_n \sim h_n$.

In this subsection we make a few claims that are intuitively obvious. We therefore decided to defer their proofs to the Appendix, in order not to interrupt the main flow of the arguments. Our first intuitive claim is the following:

LEMMA 2.18. *Let $\{x_n\}$ be a sequence that satisfies $x_{n+1} \sim c x_n$ for some $|c| < 1$. Then $\{x_n\}$ is bounded as a sequence.*

The main result of this subsection is the following somewhat general theorem:

THEOREM 2.19. *Let $a_0 < a_1 < a_2 < \cdots$ be a sequence of increasing natural numbers, and let $b_0, b_1, b_2, \ldots$ be a sequence of distinct natural numbers. Let $A = \{a_n \mid n \ge 0\}$, $B = \{b_n \mid n \ge 0\}$. Suppose the following conditions hold*:

1. *$|A \cap B|$ is finite*;
2. *$A \cup B = \mathbb{N}$*;
3. *$b_n - a_n \sim n$.*

Then $a_n \sim \phi n$ and $b_n \sim \phi^2 n$.

Note that, in particular, our Wythoff sequences $\{a_n^g\}$ and $\{b_n^g\}$ satisfy all of the above requirements, so the above theorem yields Theorem 2.3, as desired.

PROOF OF THEOREM 2.19.

We start with the following claim, which we prove in the Appendix:

LEMMA 2.20. *Regarding the sequences $\{a_n\}$ and $\{b_n\}$ in the statement of the Theorem*:

(a) *There is a constant k such that for all n, the number of $b_m > b_n$, $m < n$, is at most k.*
(b) *There is a constant k' such that for all n, the number of $b_m < b_n$, $m > n$, is at most k'.*
(c) *$a_n \sim a_{n-1}$ and $b_n \sim b_{n-1}$.*

(Note that, for our sequences $\{a_n^g\}$ and $\{b_n^g\}$, the lemmas of Section 2.4 already give bounds on the number of m's in Lemma 2.20(a,b). But we still want to prove Theorem 2.19 in general.)

Now, for $n \geq 0$, define

$$x_n = \phi n - a_n,$$

and let

$$f(n) = \left|A \cap \{0, 1, 2, \ldots, b_n - 1\}\right|$$

be the number of a's smaller than b_n.

By Lemma 2.20(a,b), the number of b's smaller than b_n is $\sim n$, so by conditions 1 and 2 of our Theorem, the number of a's smaller than b_n is $\sim b_n - n$. And by condition 3 we have $b_n - n \sim a_n$. Therefore,

$$f(n) \sim a_n. \tag{2-5}$$

Further, $a_{f(n)}$ is the first a that is $\geq b_n$ (by the definition of $f(n)$), so $a_{f(n)} \sim b_n$ by Lemma 2.20(c). Therefore (compare with (2-4)),

$$\begin{aligned} x_{f(n)} = \phi f(n) - a_{f(n)} &\sim \phi a_n - b_n \sim \phi a_n - a_n - n \\ &= \phi^{-1} a_n - n = -\phi^{-1} x_n. \end{aligned} \tag{2-6}$$

The following lemma is proven in the Appendix:

LEMMA 2.21. *There exists an integer n_1 such that $f(n) > n$ for all $n \geq n_1$.*

Now, choose n_1 as in Lemma 2.21, and recursively define the integer sequence $n_1, n_2, n_3, \ldots$, by $n_{i+1} = f(n_i)$. This sequence, therefore, is strictly increasing. Also let $n_0 = 0$.

Define the sequence $\{y_j\}_{j=0}^{\infty}$ by

$$y_j = \max_{n_j \le i \le n_{j+1}} |x_i|, \quad \text{for } j \ge 0. \tag{2-7}$$

We want to show that $\{y_j\}$ is bounded as a sequence, which would imply that $\{|x_n|\}$ is bounded as a sequence. But it follows from equation (2-6) that:

LEMMA 2.22. *The sequence* $\{y_j\}$ *satisfies* $y_{j+1} \sim \phi^{-1} y_j$.

The full proof of Lemma 2.22 is given in the Appendix.

Lemmas 2.22 and 2.18 together imply that $\{y_j\}$ is bounded as a sequence, so $\{|x_n|\}$ is bounded as a sequence, as desired. Therefore, $a_n \sim \phi n$, and by condition 3 of our Theorem, $b_n \sim \phi^2 n$. □

This completes the proof of the existence of the constants α_g and β_g of Theorem 2.3.

2.6. Experimental results. In this subsection we present experimental results on a few aspects of the function $\mathcal{G}$.

Experimental bounds on $d_n^g - n$. Let us compare the rigorous bound of $-16g \le d_n^g - n \le 2g$ given by Theorem 2.15 with data obtained experimentally.

Figure 9 shows a histogram of $d_n^g - n$ for $g = 30$, counting points up to row $5 \cdot 10^6$.

Table 2 shows the extreme values of $d_n^g - n$ achieved for different g by points lying in rows up to $5 \cdot 10^6$. In each case we show the earliest appearance of the extremal value.

We notice an interesting phenomenon: For $g \ge 7$ the maximum is achieved by the zeroth point $p_0^g = (0, g)$. This phenomenon is due to the fact that the sequence T_g tends to start with an anomalous behavior that "smooths out" over time.

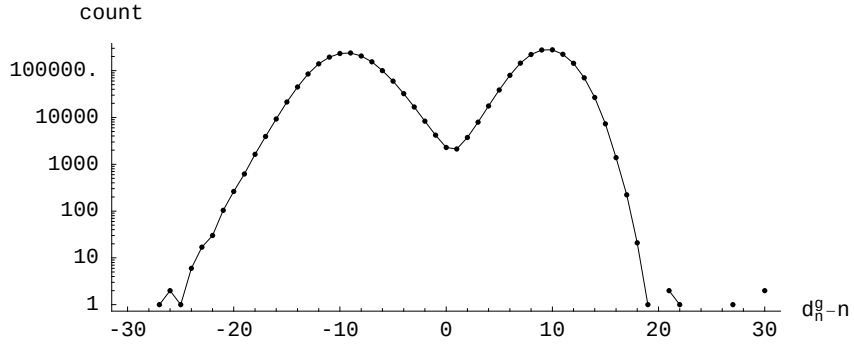

Figure 9. Histogram of $d_n^g - n$ for $g = 30$, counting points up to row $5 \cdot 10^6$.

Table 3 shows the maximum $d_n^g - n$ achieved by points having $n \geq 100$, for $g \geq 7$. We see that as g grows, the maximum achieved decreases substantially from Table 2.

g	min $d_n^g - n$	n	max $d_n^g - n$	n
0	0	0	0	0
1	−4	57	2	282
2	−6	35745	3	38814
3	−8	149804	4	2335
4	−10	569350	5	15486
5	−11	1245820	6	2638
6	−11	30165	7	1974933
7	−11	75459	7	0
8	−12	701260	8	0
9	−13	17972	9	0
10	−13	516328	10	0
11	−14	722842	11	0
12	−16	2853838	12	0
13	−17	2860809	13	0
14	−18	2814039	14	0
15	−18	2597774	15	0
16	−18	1027151	16	0
17	−18	2979529	17	0
18	−19	789978	18	0
19	−20	22347	19	0
20	−21	2548028	20	0
21	−19	277362	21	0
22	−20	30200	22	0
23	−23	1412268	23	0
24	−22	684205	24	0
25	−23	349878	25	0
26	−24	2087092	26	0
27	−24	617166	27	0
28	−24	2343474	28	0
29	−26	27	29	0
30	−27	1872274	30	0

Table 2. Extreme values of $d_n^g - n$ for given g, for points p_n^g having $a_n^g \leq 5 \cdot 10^6$.

g	max $d_n^g - n$	n	g	max $d_n^g - n$	n
7	7	131307	19	14	594141
8	8	20735	20	14	2482469
9	9	1056831	21	14	90130
10	9	258676	22	15	347510
11	10	987102	23	15	323425
12	10	1295870	24	16	129240
13	10	90426	25	17	1880006
14	11	453415	26	17	36662
15	11	61780	27	18	332552
16	12	509772	28	18	370321
17	12	86093	29	19	2425182
18	13	32439	30	18	444272

Table 3. Maximum value of $d_n^g - n$ for given g, for points p_n^g having $n \geq 100$ and $a_n^g \leq 5 \cdot 10^6$.

The conclusion from these observations is the following: The bounds observed experimentally for $d_n^g - n$ are much tighter than those given by Theorem 2.15. Therefore, either the theoretical bound is much looser than necessary, or it is a worst-case bound that is achieved very rarely in practice.

The converse of Theorem 2.3**.** Theorem 2.3 implies that if $\mathcal{G}(a, b)$ is bounded with $a \leq b$, then $|b - \phi a|$ is bounded. Is the converse also true? Namely, does a bound on $|b - \phi a|$ imply a bound on $\mathcal{G}(a, b)$? Or, on the contrary, are there arbitrarily large values very close to 0-values? We do not know the answer, but we explored this question experimentally.

We looked for the cells with the largest Grundy value lying at a given Manhattan distance from the closest 0-value. (The *Manhattan distance* between (a_1, b_1) and (a_2, b_2) is defined as $|a_2 - a_1| + |b_2 - b_1|$.) We looked up to row 10^6, calculating points up to $g = 199$. Our results are shown in Table 4.

The first column in Table 4 lists the cell with the largest Grundy value at a given Manhattan distance from the closest 0-value, for cells in rows $\leq 10^6$. Some cells are labelled "≥ 200" because they were not assigned any Grundy value ≤ 199.

The second column in Table 4 lists the cell with the largest Grundy value at a given Manhattan distance from the closest 0-value, restricted to cells between rows 600 and 10^6. (Only entries differing from the first column are shown.)

We see a very significant difference in the Grundy values between the first and second columns. This phenomenon is due to the fact that the sequences T_g

distance	cell	$\mathcal{G}$ value	row ≥ 600 cell	$\mathcal{G}$ value
1	(283432, 458601)	82		
2	(944634, 1528447)	96		
3	(44, 67)	89	(82399, 133320)	81
4	(49, 86)	115	(665224, 1076349)	82
5	(58, 86)	116	(402997, 652071)	103
6	(62, 110)	147	(538568, 871413)	99
7	(97, 168)	≥ 200	(839162, 1357804)	108
8	(95, 167)	≥ 200	(182922, 295987)	115
9	(87, 155)	≥ 200	(319656, 517229)	122
10	(85, 154)	≥ 200	(927492, 1500730)	125

Table 4. Large Grundy values at a given Manhattan distance from the closest 0-value.

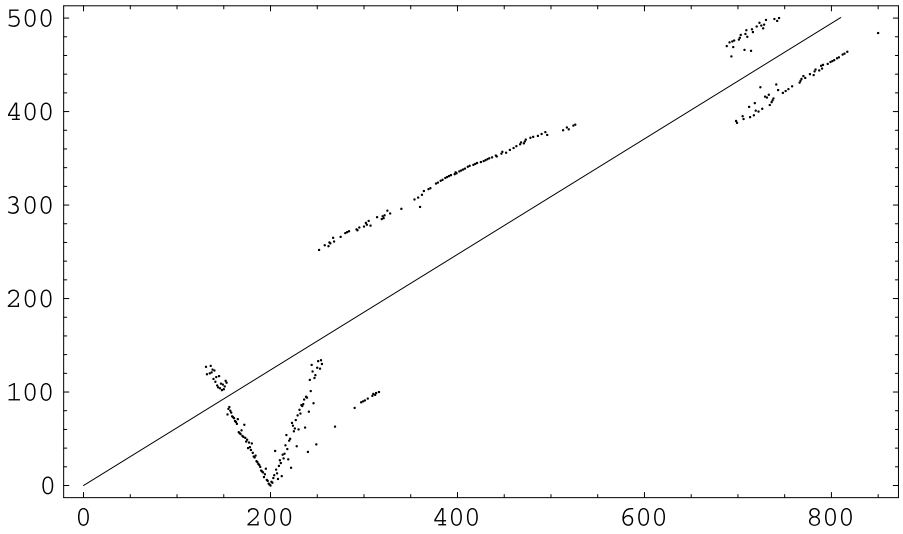

Figure 10. First terms of the sequence T_{200}.

tend to start by passing very close to the 0-points, before "smoothing out". This can be seen in Figure 10, which plots the beginning of T_{200}.

3. A recursive algorithm for the n-th g-value

In this Section we will present our so-called *Convergence Conjecture* and show how, if the conjecture is true, it leads to a recursive algorithm for find-

ing the n-th g-point in $O(f(g)\log n)$ arithmetic operations, where f is some function on g.

To do this, we will first show how Algorithm WSG (page 385) can be considered, in a certain sense, as a *finite-state automaton* that receives input symbols and jumps from state to state as it calculates rows one by one.

Consider Algorithm WSG as it is about to start inserting points in row r. For each h, $0 \le h \le g$, there are cells on row r that cannot take an h-point: some because they have an h-point below them, some because they have an h-point diagonally below, and others because they have already been assigned a value $k < h$.

We will show how we can represent this information about row r, which we will call its *state*, in such a way that there is a finite number of states among all rows. We will further consider the set of h-points to be inserted in row r as a single *symbol* out of a finite alphabet of symbols. Then, knowing the state of row r and the said symbol, we can correctly place along row r the relevant h-points, and compute the state of row $r+1$.

(Here we ignore the fact that the points to be inserted at row r are determined by the points inserted in lower rows.)

Let us develop these ideas formally.

3.1. A finite-state algorithm. Let us fix g. The variable h will always take the values $0 \le h \le g$.

In this Section, whenever we refer to an h-point, we mean a point (x, y) with $\mathcal{G}(x, y) = h$ and $x \le y$. Points with $x \ge y$ will be referred to as *reflected* h-points. A point with $x = y$ is both reflected and unreflected.

DEFINITION 3.1.

1. $index_h(r) = \left|A_h \cap \{0, \ldots, r-1\}\right|$ is the number of h-points strictly below row r. It is also the index of the first h-point on a row $\ge r$.
2. $D_h(r) = \{d_n^h \mid 0 \le n < index_h(r)\}$ is the set of diagonals occupied by h-points on rows $< r$.
3. $firstd_h(r) = \operatorname{mex} D_h(r)$ denotes the first empty diagonal on row r.
4. $ocdiag_h(r) = \{d \in D_h(r) \mid d > firstd_h(r)\}$ is the set of occupied diagonals after the first empty diagonal on row r.

   Note that

$$index_h(r) = firstd_h(r) + |ocdiag_h(r)|. \tag{3-1}$$

   This is because there are $index_h(r)$ occupied diagonals on row r: diagonals 0 through $firstd_h(r) - 1$, and $|ocdiag_h(r)|$ additional ones.
5. Similarly, $B_h(r) = \{b_n^h \mid 0 \le n < index_h(r)\}$ is the set of columns occupied by h-points on rows $< r$.

6. We let $occol_h(r) = \{b - r \mid b \in B_h(r),\ b - r \geq firstd_h(r)\}$. This set has the following interpretation: Identify each cell on row r by the diagonal it lies on, i.e., cell (r, b) is identified by $b - r$. Then $occol_h(r)$ represents the set of cells on row r on a diagonal $\geq firstd_h(r)$ that lie on a column occupied by a lower h-point.

LEMMA 3.2. *The expression*

$$\big(\max ocdiag_h(r)\big) - firstd_h(r) \tag{3-2}$$

is bounded for all r, and so is $|ocdiag_h(r)|$.

PROOF. Expression (3-2) corresponds to the maximum distance between a free diagonal and a subsequent occupied diagonal on row r; and this is bounded by Theorem 2.3. And since $ocdiag_h(r)$ only contains integers $> firstd_h(r)$, its size is also bounded. □

LEMMA 3.3. $\max occol_h(r) < \max ocdiag_h(r)$ *for all h, r.*

PROOF. Suppose a cell on row r lies on a column occupied by a lower h-point. Then a cell on row r further to the right lies on the diagonal occupied by that h-point. □

LEMMA 3.4. *For $g = 0$ we can compute explicitly the quantities of Definition* 3.1. *In particular*,

$$index_0(r) = firstd_0(r) = \lceil \phi^{-1} r \rceil, \tag{3-3}$$

$$ocdiag_0(r) = occol_0(r) = \varnothing. \tag{3-4}$$

PROOF. $index_0(r)$ is the index of the first 0-point on a row $\geq r$. Since $a_n^0 = \lfloor \phi n \rfloor$, the first n that gives $a_n^0 \geq r$ is $n = \lceil \phi^{-1} r \rceil$. (3-3) follows from the fact that the 0-points fill the diagonals in sequential order. And the second part of (3-3) follows from (3-1). □

DEFINITION 3.5. We define the following "normalized" quantities by subtracting $index_0(r)$:

$$\begin{aligned}
n_index_h(r) &= index_h(r) - index_0(r),\\
n_firstd_h(r) &= firstd_h(r) - index_0(r),\\
n_ocdiag_h(r) &= \{d - index_0(r) \mid d \in ocdiag_h(r)\},\\
n_occol_h(r) &= \{c - index_0(r) \mid c \in occol_h(r)\}.
\end{aligned}$$

LEMMA 3.6. *The quantities $n\_index_h(r)$ and $n\_firstd_h(r)$, as well as the elements of $n\_ocdiag_h(r)$ and $n\_occol_h(r)$, are bounded in absolute value for all r.*

PROOF. This follows from Theorem 2.3. □

DEFINITION 3.7. The *state* of a given row r consists of

$$n\_index_h(r),\ n\_firstd_h(r),\ n\_ocdiag_h(r),\ n\_occol_h(r),$$

for $0 \le h \le g$.

By Lemma 3.6 we have:

COROLLARY 3.8. *There is a finite number of distinct states among all rows* $r \ge 0$. □

Note that the state of a row r always satisfies

$$n\_index_h(r) = n\_firstd_h(r) + |n\_ocdiag_h(r)|. \tag{3-5}$$

Note also that if $g = 0$ there is a single state for all rows:

$$n\_index_0(r) = n\_firstd_0(r) = 0, \quad n\_ocdiag_0(r) = n\_occol_0(r) = \varnothing.$$

DEFINITION 3.9. Given row r, we denote by

$$insert(r) = \{h \mid r \in A_h, 0 \le h \le g\}$$

the set of h-points that Algorithm WSG must insert in this row.

Definitions 3.7 and 3.9 enable us to reformulate Algorithm WSG as a finite-state automaton that jumps from state to state as it reads symbols from a finite alphabet Σ. The automaton is in the state of row r when it begins to calculate row r, and after reading the symbol $insert(r)$, it goes to the state of row $r + 1$. The input alphabet is $\Sigma = 2^{\{0,\ldots,g\}}$, the set of all possible values of $insert(r)$.

Algorithm FSW (page 400) spells out in detail this finite-state formulation. (This algorithm is not equivalent to Algorithm WSG because it does not calculate the sets $insert(r)$ from previous rows, but only receives them as input.)

3.2. Convergence of states. Suppose we run Algorithm FSW starting from some row r_1, giving it as input the correct values of $insert(r_1)$, $insert(r_1 + 1)$, ..., but with a different initial state

$$n\_index'_h, n\_firstd'_h, n\_ocdiag'_h, n\_occol'_h, \quad 0 \le h \le g, \tag{3-7}$$

instead of (3-6). Then the algorithm will output 4-tuples (h, n, a_n^h, b'^h_n), where the b-coordinates of the points will not necessarily be correct.

Could it happen that at some row $r > r_1$ the algorithm reaches the correct state for row r? If that happens, then for all subsequent rows the algorithm will be in the correct state, since the state of a row depends only on the state of the previous row. Therefore, for all rows $\ge r$ the algorithm will output the correct 4-tuples (h, n, a_n^h, b_n^h).

Algorithm FSW (Finite-State Wythoff)

Input: Integer g; integers r_1, r_2 (initial and final rows); state at row r_1, given by the variables

$$n\_index_h, n\_firstd_h, n\_ocdiag_h, n\_occol_h, \quad \text{for } 0 \le h \le g; \tag{3-6}$$

sets $insert(r_1), \ldots, insert(r_2)$ (points to insert in rows $r_1, \ldots, r_2$).

Output: h-points in rows r_1 through r_2, given as 4-tuples

$$(h, n, a_n^h, b_n^h) \quad \text{for } 0 \le h \le g,\ r_1 \le a_n^h \le r_2.$$

1. For $r = r_1, \ldots, r_2$ do:
2. • Let $S \leftarrow \varnothing$ [location of points inserted in this row].
3. • For $h = 0, \ldots, g$ do:
4. ◦ If $h \in insert(r)$ then:
5. ∗ find the smallest $d \ge n\_firstd_h$ which is in none of the sets n_ocdiag_h, n_occol_h, and S;
6. ∗ output the 4-tuple

$$\big(h,\ n\_index_h + index_0(r),\ r,\ r + d + index_0(r)\big)$$

[note that $index_0(r)$ can be calculated by Lemma 3.4];

7. ∗ let $n\_index_h \leftarrow n\_index_h + 1$;
8. ∗ insert d into n_ocdiag_h, n_occol_h, and S;
9. ∗ while $n\_firstd_h \in n\_ocdiag_h$ do $n\_firstd_h \leftarrow n\_firstd_h + 1$.
10. ◦ Subtract 1 from each element of n_occol_h [since r increases by 1: see Definition 3.1–6].
11. ◦ Remove from n_ocdiag_h and n_occol_h all elements $< n\_firstd_h$.
12. • If $n\_index_0 = 1$ then [renormalize]:
13. ◦ subtract 1 from n_index_h and n_firstd_h for all h;
14. ◦ subtract 1 from each element of n_ocdiag_h and n_occol_h for all h.

Denote by

$$n\_index'_h(r), n\_firstd'_h(r), n\_ocdiag'_h(r), n\_occol'_h(r), \quad 0 \le h \le g, \quad r \ge r_1,$$

the state of the algorithm at row r when run with the initial state (3-7). We assume that the initial state (3-7) is consistent with property (3-5).

Observe that if $n\_index'_h(r_1) \ne n\_index_h(r_1)$, this difference will persist in all subsequent rows, since changes to n_index_h (at lines 7 and 13 of Algorithm

FSW) depend only on the input symbol *insert*(r). Therefore, convergence can only occur if the initial state contains the correct values of $n\_index_h(r_1)$ for all h.

Now, we make the following conjecture:

CONJECTURE 3.10 (CONVERGENCE CONJECTURE). *For every g there exists a constant R_g such that for every row r_1, if Algorithm FSW is run starting from row r_1 with the initial "dummy" state*

$$\begin{aligned} n\_index'_h = n\_firstd'_h = n\_index_h(r_1), \\ n\_ocdiag'_h = n\_occol'_h = \varnothing, \end{aligned} \qquad \text{for } 0 \le h \le g, \tag{3-8}$$

and with the correct values of insert(r_1), insert(r_1+1), ..., then the algorithm will converge to the correct state within at most R_g rows.

3.3. Experimental evidence for convergence. We tested Conjecture 3.10 experimentally as follows: For some constant $r_{\max}$ we precalculated the state of row r and the value of *insert*(r) for all r between 0 and $r_{\max}$. We then ran Algorithm FSW starting from each row r, $0 \le r \le r_{\max}$, with the dummy initial state (3-8), comparing at each step whether the algorithm's internal state converged to the correct state of the current row. We carried out this experiment for different values of g.

Our results are as follows: For $g = 0$ convergence always occurs after 0 rows; i.e., convergence is immediate.

For $g = 1$ the maximum time to convergence found was 45 rows. In fact, up to row 10^7 there are 3019 instances of convergence taking 45 rows.

For $g = 2$ the maximum found was 72 rows. Below row 10^7 there are 91 instances of convergence taking 72 rows.

For $g = 3$ the maximum of 140 rows to convergence is achieved only once below row 10^7.

For larger values of g we ran our experiment until row 10^6. Table 5 shows our findings. In each case we indicate the largest number of rows to convergence, and the starting row that achieves the maximum (or the first such starting row in case there are several).

Finally, Figure 11 shows a histogram of the time to convergence for $g = 10$ up to row 10^6. The shape of the curve suggests that there might be instances of higher convergence times that occur very rarely. However, we still find it plausible that a theoretical maximum R_g exists.

Note that Conjecture 3.10 could also be true only up to a certain value of g.

g	rows to convergence	starting row
0	0	0
1	45	2201
2	72	72058
3	140	804421
4	180	862429
5	235	732494
6	395	685531
7	395	685531
8	461	827469
9	630	59948
10	909	443109
⋮		
15	2041	8662
⋮		
20	4136	896721

Table 5. Maximum number of rows to convergence for different g up to row 10^6, and first starting row that achieves the maximum.

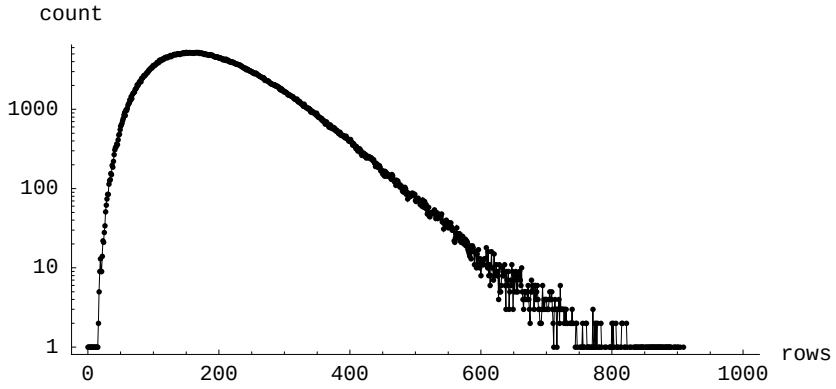

Figure 11. Histogram of the number of rows to convergence for $g = 10$, up to row 10^6.

3.4. The recursive algorithm. We now show how Conjecture 3.10 leads to an algorithm for computing point p_n^g in $O(f(g)\log n)$ arithmetic operations, where f is some function on the constant R_g of Conjecture 3.10 and the constants α_g, β_g of Theorem 2.3.

Algorithm RW below is a recursive algorithm that receives as input an integer g and an interval $[r_1, r_2]$ of rows, and calculates all h-points, $0 \le h \le g$, in that interval.

Algorithm RW (Recursive Wythoff)

Input: Integer g; integers r_1, r_2 (initial and final rows).

Output: $index_h(r_1)$ for $0 \le h \le g$; set S of h-points in rows r_1 through r_2, given as 4-tuples (h, n, a_n^h, b_n^h) for $0 \le h \le g,\ r_1 \le a_n^h \le r_2$.

1. Let $r_0 \leftarrow r_1 - R_g$.
2. Let L and H be lower and upper bounds for $a_n^h - \phi^{-1} b_n^h$ for $0 \le h \le g$ and all n, according to Theorem 2.3. [Note that $a_n^h < \phi^{-1} r + L$ implies $b_n^h < r$, and $a_n^h > \phi^{-1} r + H$ implies $b_n^h > r$.]
3. Let $r_1' \leftarrow \lceil \phi^{-1} r_0 + L \rceil$, $r_2' \leftarrow \lfloor \phi^{-1} r_2 + H \rfloor$.
4. If $r_2' \ge r_0$ or $r_1' \le 2g$ then:
5. • calculate and return the desired points by starting from row 0 using Algorithm WSG;
6. else:
7. • call Algorithm RW recursively and get $index_h(r_1')$ and the set S' of h-points in rows r_1' through r_2' for $0 \le h \le g$;
8. • calculate $insert(r_0), \ldots, insert(r_2)$ as

$$insert(r) = \left\{ h \;\middle|\; \nexists n \text{ for which } (h, n, a_n^h, r) \in S' \right\}$$

   for $r_0 \le r \le r_2$;
9. • let t_h be the number of h-points in S' with $b_n^h < r_0$, for $0 \le h \le g$;
10. • calculate $index_h(r_0)$ as $index_h(r_0) = r_0 + 1 - index_h(r_1') - t_h$, for $0 \le h \le g$ [see explanation];
11. • calculate $n\_index_h(r_0)$ as $n\_index_h(r_0) = index_h(r_0) - index_0(r_0)$, for $0 \le h \le g$;
12. • run Algorithm FSW from rows r_0 to r_2 starting from the dummy state

$$\begin{aligned} n\_index_h' = n\_firstd_h' = n\_index_h(r_0), \\ n\_ocdiag_h' = n\_occol_h' = \varnothing, \end{aligned} \qquad 0 \le h \le g,$$

   using $insert(r_0), \ldots, insert(r_2)$; get set T of 4-tuples (h, n, a_n^h, b'^h_n) for $r_0 \le a_n^h \le r_2$;
13. • return $index_h(r_1)$ for $0 \le h \le g$, and the 4-tuples in T with $r_1 \le a_n^h \le r_2$.

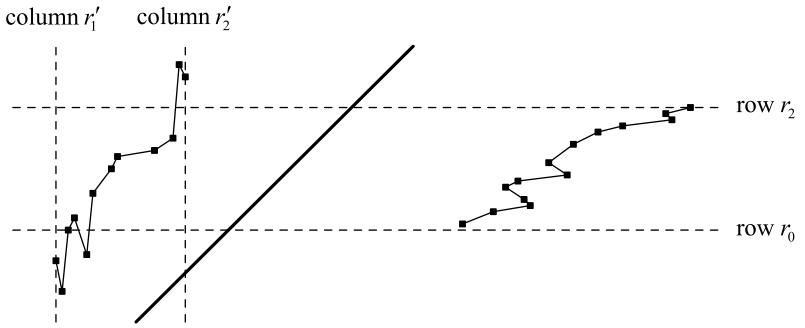

Figure 12. Points and reflected points in rows r_0 through r_2.

The idea behind Algorithm RW is the following: To calculate the h-points between rows r_1 and r_2, we run Algorithm FSW starting from row $r_0 = r_1 - R_g$ and the dummy initial state (3-8). Then, by Conjecture 3.10, the 4-tuples obtained from row r_1 on will be the correct ones (h, n, a_n^h, b_n^h).

We face two problems, however:

- To run Algorithm FSW we also need to know $insert(r_0), \ldots, insert(r_2)$, i.e., which h-points to insert between rows r_0 and r_2.
- For the dummy initial state (3-8) we need to know $index_h(r_0)$ for $0 \le h \le g$, i.e., how many h-points there are below r_0.

We solve both problems with a recursive call, in which we calculate all the *reflected* h-points that lie between rows r_0 and r_2, to the left of the main diagonal (see Figure 12). By the definition of L and H (line 2 of Algorithm RW), all these reflected h-points lie between columns r_1' and r_2' as computed in line 3. Of course, finding these reflected h-points is equivalent to finding the unreflected originals.

Once we have the reflected h-points, constructing $insert(r_0), \ldots, insert(r_2)$ is simple, since every row r, $r_0 \le r \le r_2$ that does not contain a reflected h-point must contain an h-point, and vice versa.

And computing $index_h(r_0)$ is also no problem, once we know $index_h(r_1')$ from the recursive call. Recall that $index_h(r_0)$ is the number of h-points on rows $0, \ldots, r_0 - 1$. Let k_h be the number of *reflected* h-points on rows $0, \ldots, r_0 - 1$. Then

$$index_h(r_0) + k_h = r_0 + 1,$$

since there is one h-point on the main diagonal, which is counted twice.

To calculate k_h, note that all reflected h-points before column r_1' lie below row r_0, and there are $index_h(r_1')$ such reflected h-points. Therefore,

$$k_h = index_h(r_1') + t_h,$$

where t_h is the number of reflected h-points below row r_0 that lie on or after column r_1', as in line 9. Putting all this together, we get

$$index_h(r_0) = r_0 + 1 - index_h(r_1') - t_h,$$

as in line 10.

The above calculation is only valid if the h-point on the main diagonal lies before column r_1'. That is why we check for the case $r_1' \le 2g$ at line 4 (recall Lemma 2.2).

Finally, the check $r_2' \ge r_0$ at line 4 prevents making a recursive call if the new interval $[r_1', r_2']$ is not strictly below the old interval $[r_0, r_2]$.

If we cannot make a recursive call (for either of the two possible reasons), we calculate the h-points in the standard way, using Algorithm WSG starting from row 0.

3.5. Algorithm RW's running time. If we want to use Algorithm RW to calculate a single point p_n^g, we must first estimate its row number a_n^g. By Theorem 2.3, we can bound it between $r_1 = \lceil \phi n + L' \rceil$ and $r_2 = \lfloor \phi n + H' \rfloor$ for some constants L', H' that depend on g.

Whenever Algorithm RW makes a recursive call, it goes from an interval of length $\Delta r = r_2 - r_1$ to an interval of length $\Delta r' = r_2' - r_1'$, where

$$\Delta r' = \phi^{-1} \Delta r + (H - L + \phi^{-1} R_g)$$

(ignoring the rounding to integers). After repeated application of this transformation, the interval length converges to the constant

$$\Delta r^* = \phi^2 (H - L) + \phi R_g.$$

The number of recursive calls is $O(\log n)$, since each interval is ϕ times closer to the origin than its predecessor. And in the base case of the recursion, Algorithm WSG runs for at most a bounded number of rows, taking constant time.

Therefore, altogether Algorithm RW runs in $O(f(g) \log n)$ steps, for some function f that depends on the constants R_g, α_g, and β_g, as claimed.

3.6. Application of Algorithm RW. Let us discuss how to apply Algorithm RW to the problem raised in the Introduction, namely playing the sum of Wythoff's game with a Nim pile.

Suppose we are given the sum of a game of Wythoff in position (a, b), $a \le b$, with a Nim pile of size g, where a and b are very large and g is relatively small. Suppose Conjecture 3.10 is true for this value of g, and we know the value of R_g.

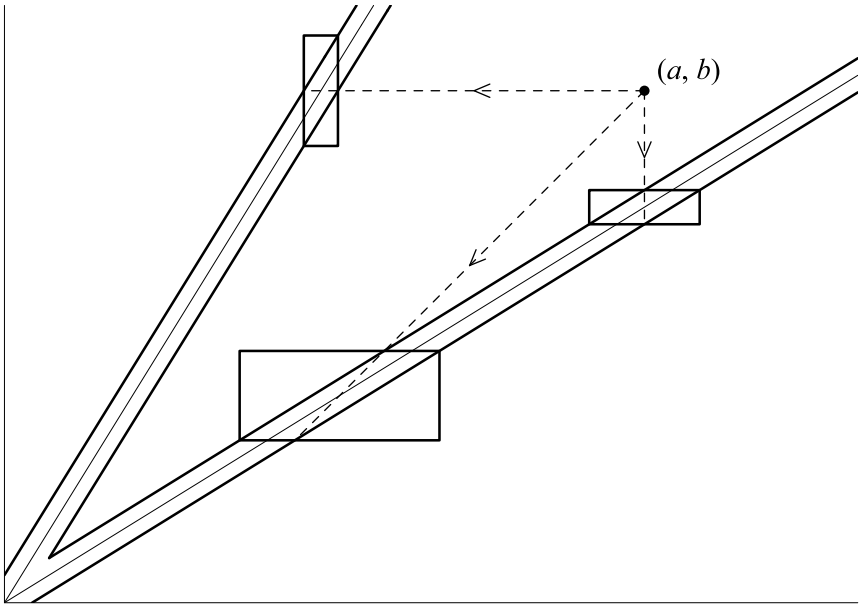

Figure 13. Intervals in which to look for a successor with Grundy value g to position (a, b).

We have to determine whether $\mathcal{G}(a, b)$ is larger than, smaller than, or equal to g. By Theorem 2.3, we can only have $\mathcal{G}(a, b) \leq g$ if $|b - \phi a| \leq k_g$ for some constant k_g that depends on α_g and β_g.

Therefore, if $|b - \phi a| > k_g$, we know right away that $\mathcal{G}(a, b) > g$. If, on the other hand, $|b - \phi a| \leq k_g$, then we use Algorithm RW to find all the h-points, $h \leq g$, in the vicinity of (a, b), and we check whether (a, b) is one of them.

If we find that $\mathcal{G}(a, b) = g$, then the overall game is in a P-position, so there is no winning move. If $\mathcal{G}(a, b) = h < g$, then our winning move is to reduce the Nim pile to size h. And if $\mathcal{G}(a, b) > g$, then our winning move consists of moving in Wythoff's game to a position with Grundy value g. There are at most three alternatives to check — moving horizontally, vertically, or diagonally. Therefore, the winning move can be found by making at most three calls to Algorithm RW with bounded-size intervals, as shown schematically in Figure 13.

g	$p^g_{10^{12}}$	g	$p^g_{10^{12}}$	g	$p^g_{10^{12}}$
0	$(a+49, b+49)$	7	$(a+50, b+46)$	14	$(a+49, b+54)$
1	$(a+50, b+50)$	8	$(a+51, b+51)$	15	$(a+47, b+51)$
2	$(a+49, b+50)$	9	$(a+51, b+56)$	16	$(a+49, b+43)$
3	$(a+50, b+49)$	10	$(a+49, b+52)$	17	$(a+53, b+51)$
4	$(a+50, b+51)$	11	$(a+51, b+49)$	18	$(a+48, b+52)$
5	$(a+50, b+52)$	12	$(a+49, b+53)$	19	$(a+52, b+61)$
6	$(a+49, b+51)$	13	$(a+50, b+55)$	20	$(a+49, b+39)$

where $a = 1\,618\,033\,988\,700$ and $b = 2\,618\,033\,988\,700$.

Table 6. Predicted value of $p^g_{10^{12}}$ for $0 \le g \le 20$.

3.7. Algorithm RW in practice. We wrote a C++ implementation of Algorithm RW. For the constants L and H we used experimental lower and upper bounds for $a^g_n - \phi^{-1} b^g_n$, to which we added safety margins. For the constant R_g we added a safety margin to the values shown in Table 5.

We checked our program's results against those produced by the nonrecursive Algorithm WSG. The results were in complete agreement as far as we tested.

We also used our recursive program to predict the trillionth g-values for g between 0 and 20. We used $L = -15.0$, $H = 15.0$ (which are a safe distance away from the experimental bounds of -12.4 and 11.3), and $R_{20} = 8000$ (almost twice the value in Table 5).

Our predictions are shown in Table 6. The program actually performed this calculation in just twenty seconds. These predictions might be verified one day with a powerful computer.

To conclude, note that if there are only sporadic counterexamples to Conjecture 3.10 for a certain value of R_g, then Algorithm RW is still likely to give correct results in most cases. Algorithm RW will only fail if one of the rows r_0 in the different recursion levels constitutes the initial row of a counterexample to Conjecture 3.10. But, as we showed earlier, the number of recursion levels is logarithmic in the magnitude of the initial parameters.

Appendix: Lemmas of Section 2.5

We relegated some proofs in Section 2.5 to this Appendix.

LEMMA 2.18. *Let $\{x_n\}$ be a sequence that satisfies $x_{n+1} \sim c x_n$ for some $|c| < 1$. Then $\{x_n\}$ is bounded as a sequence.*

PROOF. Let k be the constant such that $|x_{n+1} - c x_n| \le k$, as in Definition 2.17; and let $d = k/(1-|c|)$. Let I be the real interval $I = [-d, d]$. It can be verified

that if $x_n \in I$, then $x_{n+1} \in I$, and if $x_n \notin I$, then $|x_{n+1}| - d \leq |c|\,(|x_n| - d)$; in other words, the distance between x_n and I is multiplied by a factor of at most $|c|$. Therefore, since $|c| < 1$, the sequence $\{x_n\}$ is "attracted" towards I. □

LEMMA 2.20. *Regarding the sequences $\{a_n\}$ and $\{b_n\}$ in the statement of Theorem* 2.19:

(a) *There is a constant k such that for all n, the number of $b_m > b_n$, $m < n$, is at most k.*
(b) *There is a constant k' such that for all n, the number of $b_m < b_n$, $m > n$, is at most k'.*
(c) *$a_n \sim a_{n-1}$ and $b_n \sim b_{n-1}$.*

PROOF. According to condition 3 in the Theorem, let L and H be such that

$$n + L \leq b_n - a_n \leq n + H \quad \text{for all } n.$$

Suppose $b_m > b_n$, with $m < n$. Then

$$b_n \geq a_n + n + L, \quad b_m \leq a_m + m + H.$$

But $a_n - a_m \geq n - m$, so $0 < b_m - b_n \leq 2(m-n) + H - L$, so

$$m > n - \frac{H-L}{2}.$$

Therefore, for every n there are at most $(H-L)/2$ possible values for m. This proves claim (a). Claim (b) follows analogously.

For claim (c), let $k = a_n - a_{n-1}$. Then, by condition 2 in the Theorem, the interval

$$I = \{a_{n-1} + 1, \ldots, a_n - 1\},$$

whose size is $k-1$, is a subset of B. Let i be the smallest index and j the largest index such that both b_i and b_j are in I. Then $b_j - b_i \leq k-2$ and $j - i \geq k-2$. But we have

$$b_j \geq a_j + j + L, \quad b_i \leq a_i + i + H, \quad a_j - a_i \geq j - i,$$

so

$$k - 2 \geq b_j - b_i \geq 2(j-i) + L - H \geq 2k - 4 + L - H,$$

so

$$k \leq H - L + 2,$$

proving that $a_n \sim a_{n-1}$. Moreover

$$b_n - b_{n-1} \leq (a_n + n + H) - (a_{n-1} + n - 1 + L) \leq 2(H-L) + 3$$

and

$$b_n - b_{n-1} \geq (a_n + n + L) - (a_{n-1} + n - 1 + H) \geq L - H + 2,$$

since $a_n - a_{n-1} \geq 1$. Therefore, $b_n \sim b_{n-1}$. □

LEMMA 2.21. *There exists an integer n_1 such that $f(n) > n$ for all $n \geq n_1$.*

PROOF. The number of a's smaller than a_n is exactly n, so the number of b's smaller than a_n is $\sim a_n - n$. On the other hand, $b_n \sim b_{n-1}$, so the number of b's smaller than a_n goes to infinity as $n \to \infty$. Therefore, $a_n - n \to \infty$. But $a_n \sim f(n)$ (equation (2-5) in Section 2.5). Therefore, $f(n) - n \to \infty$, which is even stronger than our claim. □

LEMMA 2.22. *The sequence $\{y_j\}$ defined in* (2-7) *satisfies $y_{j+1} \sim \phi^{-1} y_j$.*

PROOF. First note that if $\{c_n\}$ and $\{d_n\}$ are sequences such that $c_n \sim d_n$, then by equation (2-5) and Lemma 2.20(c) we have $f(c_n) \sim f(d_n)$. We also have $x_{c_n} \sim x_{d_n}$.

For each $j \geq 0$, let $m(j)$, $n_j \leq m(j) \leq n_{j+1}$, be the index for which the maximum $y_j = |x_{m(j)}|$ is achieved.

We claim that for each $j \geq 1$ there exists an integer $p(j)$ in the range $n_j \leq p_j \leq n_{j+1}$ such that

$$f(p(j)) \sim m(j+1). \tag{A-9}$$

Indeed, at the one extreme we have $f(n_j) = n_{j+1} \leq m(j+1)$, while at the other we have $f(n_{j+1}) = n_{j+2} \geq m(j+1)$. Thus, there exists some intermediate value $p(j)$ such that $f(p_j) \leq m(j+1)$ and $f(p_j + 1) \geq m(j+1)$. This choice of $p(j)$ satisfies (A-9).

Therefore, using equation (2-6),

$$y_{j+1} = |x_{m(j+1)}| \sim |x_{f(p(j))}| \sim \phi^{-1} |x_{p(j)}| \leq \phi^{-1} |x_{m(j)}| = \phi^{-1} y_j.$$

So

$$y_{j+1} \sim h_j \leq \phi^{-1} y_j \tag{A-10}$$

for some sequence $\{h_j\}$.

Similarly, for each $j \geq 1$ there exists an integer $q(j)$ in the range $n_j \leq q(j) \leq n_{j+1}$ such that

$$f(m(j)) \sim q(j+1)$$

for all $j \geq 1$. Specifically, let

$$q(j+1) = \min\left\{\max\{f(m(j)), n_{j+1}\}, n_{j+2}\right\}.$$

(It is not hard to show that if $f(n) > f(n')$, $n < n'$, then $f(n) - f(n')$ is bounded.)

Therefore, using again equation (2-6),

$$y_{j+1} \geq |x_{q(j+1)}| \sim |x_{f(m(j))}| \sim \phi^{-1} |x_{m(j)}| = \phi^{-1} y_j;$$

so $y_{j+1} \geq h'_j \sim \phi^{-1} y_j$ for some sequence $\{h'_j\}$. This, together with (A-10), implies that $y_{j+1} \sim \phi^{-1} y_j$. □

Acknowledgements

This paper is based on the author's M.Sc. thesis work [6]. I would like to thank my advisor, Aviezri Fraenkel, for suggesting the topic for this work, and for many useful discussions we had together.

I also owe thanks to Achim Flammenkamp, who gave me extensive comments on a draft of my thesis, and who also checked independently most of the experimental data presented here.

References

[1] E. R. Berlekamp, J. H. Conway and R. K. Guy, *Winning Ways for your Mathematical Plays*, vol. 1, second edition, A K Peters, Natick, MA, 2001. (First edition: Academic Press, New York, 1982.)

[2] U. Blass and A. S. Fraenkel, The Sprague–Grundy function for Wythoff's game, *Theoret. Comput. Sci.*, vol. 75 (1990), pp. 311–333.

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

[4] H. Landman, A simple FSM-based proof of the additive periodicity of the Sprague–Grundy function of Wythoff's game, in: *More Games of No Chance*, MSRI Publications, vol. 42, Cambridge University Press, Cambridge, 2002, pp. 383–386.

[5] H. Landman, Personal communication.

[6] G. Nivasch, *The Sprague–Grundy function for Wythoff's game: On the location of the g-values*, M.Sc. thesis, Weizmann Institute of Science, 2004.

[7] G. Nivasch and E. Lev, Nonattacking queens on a triangle, *Mathematics Magazine*, vol. 78 (2005), pp. 399–403.

[8] N. Pink, *Über die Grundyfunktionen des Wythoffspiels und verwandter Spiele*, dissertation, Universität Heidelberg, 1993.

[9] P. Vaderlind, R. K. Guy and L. C. Larson, *The Inquisitive Problem Solver*, The Mathematical Association of America, 2002.

[10] W. A. Wythoff, A modification of the game of Nim, *Niew Archief voor Wiskunde*, vol. 7 (1907), pp. 199–202.

GABRIEL NIVASCH
SCHOOL OF COMPUTER SCIENCE
TEL AVIV UNIVERSITY
TEL AVIV 69978
ISRAEL
gabriel.nivasch@cs.tau.ac.il