Rectangular Hardinian Arrays

George Spahn and Robert Dougherty-Bliss

May 2024

1 Introduction

In [1], Dougherty-Bliss and Kauers introduce a combinatorial object known as a *Hardinian array*. A Hardinian array with positive integer parameter r is an $n \times k$ array which obeys the following rules:

1. The top left entry is 0, the bottom right entry is the king distance minus r.
2. Each king step down, right, and down-right must increase the value by or leave the value the same.
3. Each value must be within r of its king distance.

The family of bivariate sequences $H_r(n, k)$ counts the number of Hardinian arrays.

Sequence AXXXXXX in the OEIS [2] contains the following conjectures about $H_2(n, k)$ for fixed k:

$$
\begin{aligned}
H_2(n,1) &= \frac{1}{2}n^2 - \frac{3}{2}n + 1 \\
H_2(n,2) &= 4n^2 - 20n + 25 \\
H_2(n,3) &= 40n^2 - 279n + 497 \\
H_2(n,3) &= 480n^2 - 4354n + 10098 \\
H_2(n,4) &= 6400n^2 - 71990n + 206573 \\
H_2(n,5) &= 90112n^2 - 1212288n + 4150790 \\
H_2(n,6) &= 1306624n^2 - 20460244n + 81385043.
\end{aligned}
$$

We can confirm that these conjectures are correct. In fact, we can prove the following theorem.

Theorem 1. *$H_r(n, k)$ is a polynomial of degree r for sufficiently large n (or k). There is an algorithm to determine this polynomial.*

The technique to prove this theorem is an application of the transfer matrix method. Given a fixed k, there exists a finite state machine which accepts only valid Hardinian arrays. This state machine has an adjacency matrix which is lower triangular and at least a single 1 on its diagonal. This implies that the number of paths of length n on this state machine which end in accepting states are polynomials.

The brilliant idea to make the state machine *finite* is the following change of variables.

Definition 1. *Given a Hardinian array, M, define a new array $T(M)$ by*

$$T(M)_{ij} = M_{ij} - KD(i, j) + r,$$

where $KD(i, j) = \max(i, j)$ is the king distance of (i, j) from $(0, 0)$. (Note that our matrices are 0-indexed.)

Now suppose we fix the number of rows n and count the number of $n \times k$ arrays as k increases. Can we construct a finite state machine that reads the columns as symbols and determines whether the columns read so far form a valid array? At first this seems tricky: since the values of the matrix can be arbitrarily large, we cannot simply use the contents of the previous column as a state. We can't store the king distance either because it also is allowed to grow without bound.

Note that the entries of $T(M)$ are strictly between 0 and r, so it will now be convenient to use the contents of the previous column as a state in our state machine. We must now replace the 3 conditions on Hardinian arrays with 3 equivalent conditions on modified arrays.

Since each row must be a non-increasing sequence of entries between 0 and r, the number of possibilities for a row of length k is upper bounded by a polynomial in k of degree r. Later we will show that indeed the number of valid arrays eventually satisfies a polynomial in k.

2 Our code

We have written a small Maple package which implements the main theorem. Its main procedure is `hardinPoly(n, r, k)`, which produces the polynomial which $H_r(n, k)$ equals for sufficiently large n. Here is a brief demo:

```
> hardinPoly(n, 2, 1);
                       2
                  1/2 n  - 3/2 n + 1
> hardinPoly(n, 2, 2);
                      2
                   4 n  - 20 n + 25
> hardinPoly(n, 2, 3);
                      2
                  40 n  - 279 n + 497
> hardinPoly(n, 2, 4);
                     2
                480 n  - 4354 n + 10098

> hardinPoly(n, 3, 6);
```

```
                  3                  2
5242880/3 n  - 41275392 n  + 991610656/3 n - 897487301
```

As a very crude overestimate, the state machine for parameter r with k rows contains r^k vertices, so the adjancency matrix is $r^k \times r^k$. We need to compute the inverse of an $r^k \times r^k$ symbolic matrix. There are a few other computational problems involved. If we put some effort into it, we could probably make our programs much much faster, perhaps by borrowing some techniques from computer algebra.

3 Random facts

Here are some random facts that we need to write down.

Theorem 2. *For any square matrix M, the sequences*

$$a_{ij}(n) := (M^n)_{ij}$$

are all C-finite with characteristic polynomial dividing the characteristic polynomial of M. In particular, the eigenvalues of a_{ij} are eigenvalues of M.

Proof. By the Caley–Hamilton theorem, M is annihilated by its own characteristic polynomial. Multiplying this by M^n and extracting the ijth entry shows that $a_{ij}(n)$ is also annihilated by the characteristic polynomial, and therefore its characteristic polynomial divides M's. □

Corollary 1. *If a square matrix M has only eigenvalues 0 and 1, then $a_{ij}(n) = (M^n)_{ij}$ is either zero for all but finitely many values or a polynomial.*

Proof. By the previous theorem $a_{ij}(n)$ is C-finite with eigenvalues 0 and 1. □

Theorem 3. *The adjacency matrix for the "Hardinian state machine" can be made lower triangular.*

Proof. A vertex in the state machine is labeled by a vector of values. The edge $v \to w$ exists only if (but not necessarily *if*) each entry in w is $\leq$ its corresponding entry in v. Therefore, if we order the vertices by their sum, then the adjacency matrix will be lower triangular. □

Theorem 4. *If M is a square matrix, then the ijth entry of $(I - xM)^{-1}$ is the generating function of $a_{ij}(n) = (M^n)_{ij}$.*

Proof. I believe that this theorem is true, but I have never seen a careful proof. I would be interested in finding one / writing one. □

References

[1] Robert Dougherty-Bliss and Manuel Kauers. "Hardinian Arrays". In: *El. J. Combinat.* 31 (2 2024).

[2] OEIS Foundation Inc. *The On-Line Encyclopedia of Integer Sequences*. 2024. URL: http://oeis.org.