COUNTING HEXAGONAL LATTICE ANIMALS

Mohamud Mohammed [1]

Abstract: We describe Maple packages for the automatic generation of generating functions (and series expansions) for counting lattice animals (fixed polyominoes),in the two-dimensional hexagonal lattice, of bounded, but *arbitrary* width. Our Maple packages (complete with *source code*) are easy-to-use and available from our website.

In [Z0], the maple packages ANIMALS and FreeANIMALS, that count fixed polyominoes for two-dimensional square lattices animals, were discussed. Here we give analogs of these for counting hexagonal lattice animals.

A *hexagonal animal* is a connected set of unit hexagons, on the hexagonal 2D lattice, up to translation-equivalence. In Neil Sloane and Simone Plouffe's Encyclopedia[SP] and Sloane's database[S], the total number of equivalence classes of animals (fixed hexagonal polyominoes) with n cells,$a(n)$, is given for $n < 25$ by **A001207** , formerly known as $M2897$ (see [S] and [R]). For example $a(1) = 1$, $a(2) = 3$, $a(3) = 11$, $a(4) = 44$, $a(5) = 186$, $a(6) = 814$, $a(7) = 3652$, $a(8) = 16684$, $a(9) = 77,359$, $a(10) = 362,671$ etc.

The generating function that enumerate 1-board hexagonal animals is given in [S] and [K], which is also computed using the maple package LEGO in [Z1] by taking $p(a,b) = a + b$, since there are a+b ways of putting a board of hexagonal animal with a cells next to a board with b cells. Here we also generate this sequence as a special case and give the first 12 terms of a new sequence that enumerates 2-board hexagonal animals which is an analog of 2-board animals for the square lattice given in [Z0]. In the Umbra version of this paper (in preparation) we expect to extend this sequence to at least 54 terms.

I) **Maple representation of hexagonal animals**.

For our purposes, it would be convenient to embed the hexagonal animal into the square lattice. We place animals in their horizontal(natural) position, i.e. the two parallel sides parallel to the x-axis. Since each vertex of a cell is at the lattice points in Z^2, to describe an animal, it suffices to give the y-coordinates of the parallel sides, which we will call the support of the animal.

Example 1 : the animal $\{\{[0,4]\}\}$ represents a hexagonal animal of two cells with the three horizontal lines on the lines $y = 0, y = 2, y = 4$ and the left vertex at $(0,1),(0,3)$ and right at $(1,1),(1,3)$. Therefore, between consecutive horizontal lines we have distance of two and in between this we have vertices.

[1] Department of Mathematics, Rutgers University, Piscataway, NJ 08854, USA. `mohamudm@math.rutgers.edu` `http://www.math.rutgers.edu/~mohamudm/`. March 1, 2002. Accompanied by the Maple packages `HexANIMALS` and `HexaFreeANIMALS`, available from Mohamud's homepage.

Example 2. The three animals with 2 cells, shown below, are represented respectively by

$$\{\{\{[0,4]\}\}\}, \{\{\{[1,3]\}, \{[0,2]\}\}\}, \{\{\{[0,2]\}, \{[1,3]\}\}\}.$$

II) **Globally skinny animals**.

A globally skinny animal is animal in which the entire animal has to fit into a prescribed bounded region.

That is: if we define, as in [Z0], for $x = 0, 1, \dots$

$$M(x) := max\{y|(x,y) \in S\} \quad , \quad m(x) := min\{y|(x,y) \in S\} \quad .$$

then, here we count , for a given n, the number of animals such that

$max_x(M(x)) - Min_x((m(x)) <= n - 1$

A User's Manual for `HexANIMALS`

First download `HexANIMALS` to your directory (either directly from INTEGERS or from my website). Then go into Maple by typing: `maple` (or, if you prefer, `xmaple`) followed by `Enter`, or click on the Maple icon. Then, once in Maple, type: `read HexANIMALS`, assuming you are still in the same directory, or, e.g. `read 'research/HexANIMALS/HexANIMALS' ;` (i.e. the full path-name of the file `HexANIMALS`).

Then follow the on-line help. To see the names of the main procedures type: `ezra();` . To get help on a specific procedure type `ezra(ProcName);` . For example, to get help on `GF1` type `ezra(GF1);`

For example, to find the generating function that enumerates animals of width ≤ 5, type `GF1(5,s)`.

Explanation of `HexANIMALS`

The main procedures are **gf, GF1,Animal**. The representation in (I) above enables as to adopt this new animals to the old one in ANIMALS. All we have to do is make the appropriate changes in the individual procedures. The part that needs modification are: *Support(PreLet), Weight(Let), PreLeftLetters(a,b) and PreLetToLet(Let,PreLet)*. The first two are straight forward and *PreLeftLetters(a,b)* now outputs all set-partitions of $\{a, a+1, \dots, b\}$ (written in interval notation)with one interval per block where the gaps from one point to the next is counted by two.

In `PreLetToLet(Letter,PreLetter)` now we test Support(Letter) intersection Support(PreLeftLetter) to be the empty set and in that case it check if each component of Letter is touched by at least one component of PreLeftLetter as in ANIMALS.

All the remaining part of HexANIMALS package carry over to HexaFreeANIMALS.

If you are interested to generate the animals with n cells, then use *Animal*(n), which gives all the animals with n cells as a list read from left to right.

Animal(n) uses the output from *MarCha*(n) and starting from Left(starters), if Weight(Let) is n then it checks if it is a legal right most letter of an animal by testing if it belongs to Right in MarCha(n). In that case it Normalizes by calling *NormalizeLetters* and keep it. If $wieght(Let) < n$ then it follows checking all its followers and keep repeating this to all elements in Left(starters). By the time it finishes checking, it knows all the animals with n-cells by their representation.

For example *Animal(2)* should outputs $\{[\{\{[0,2]\}\},\{\{[1,3]\}\}],[\{\{1,3]\}\},\{\{[0,2]\}\}],[\{\{[0,4]\}\}]\}$.

If you are interested in the generating function for animals with width *exactly* n, then use `Gf(n,s)`, which is `GF1(n,s)-GF1(n-1,s)`.

Once $n \geq 13$, it takes too long to compute `GF1(n,s)` exactly, but one can go much further with `GFseries(n,L)` which uses `SolveMC1series` to find the series expansion up to L terms. `GfseriesS(n,L,s)` uses `SolveMC1seriesS` to find the series-expansion where we also keep track of the length of the animal.

III) **Locally-Skinny Animals**

With $M(x)$ and $m(x)$ as in (II) above,

the Maple package `HexaFreeANIMALS`, to be described in this section, counts animals such that $M(x) - m(x) \leq n-1$ for *each* x.

For example the 'staircase' animal: $\{\{\{[0,2]\}\},\{\{[1,3]\}\},\{\{[2,4]\}\},\{\{[3,5]\}\},,,\{\{[n-1,n+1]\}\},\{\{[n,n+2]\}\}\ldots\}$ is not counted by `GF(n,s)` of `HexANIMALS`, but is already counted by `gf(3,s)` of `HexaFreeANIMALS`, since each vertical cross-section, individually, had width ≤ 2.

A User's Manual for the Maple Package HexaFreeANIMALS

The main procedures are: `gf, gfSeries, gfList, gfSeriesList` .

For a positive integer n, and a variable s, typing `gf(n,s)` would give the generating function

$$f_n(s) := \sum_{i=1}^{\infty} a_n(i) s^i \quad ,$$

where $a_n(i)$ is the number of (2D site) animals such that for each vertical cross-section $x = x_0$ the difference between the biggest y such that (x_0, y) belongs to the animal and the smallest such y is $\leq n-1$.

`gf(n,s)` works, on my computers, up to $n = 13$. Beyond that, you may wish to use `gfSeries(n,L)`, where L is also a positive integer,in order to get the first L coefficients of $f_n(s)$.

Both `gf(n,s)` and `gfSeries(n,L)` enumerate animals where each vertical cross-section has width $\leq n$.

Suppose that whenever the "letter" (i.e. vertical cross-section) has i boards, then it is allowed to have width $\leq R_i$, for a list $[R_1, R_2, \ldots, R_m]$, say. The corresponding generating functions are given by typing `gfList(List,s);` and `gfSeriesList(List,s);`. For example `gfList([7,5],s);` would give the generating function for animals whose vertical cross-sections with one interval (board) have width ≤ 7 and vertical cross-sections with two boards have width ≤ 5.

gfSeriesList([48],24) outputs the sequence, **A059716** in the Neil Sloane and Simone Plouffe's Encyclopedia and database data 1-board hexagonal lattice animals.[S]

gfSeriesList([25,25],12) gives the first 12 terms of the sequence

$$1, 3, 11, 44, 186, 814, 3648, 16611, 76437, 354112, 1647344, 7682237$$

that enumerates 2-board hexagonal animals(" Board-pair-pile polyominoes with n-cells), analog of the sequence **A001170** of Neil Sloane and Simon Plouffe's Encyclopedia and database.

Explanation of HexaFreeANIMALS

`Alphabet(n);` gives the smaller set of normalized letters, where the smallest integer is always 0. The modification from HexANIMALS to HexaFreeANIMALS is straightforward and careful study of the code would help.

Conclusion

We have described two Maple packages: `HexANIMALS, HexaFreeANIMALS,`. The first of these, `HexANIMALS` is a packages enumerate "skinny" hexagonal lattice-animals, where the object has to fit completely within a prescribed horizontal strip, and the second one `HexaFreeANIMALS` enumerates skinny objects, where the whole object has unbounded width, but each vertical cross-section has bounded width.

Acknowledgment I would like to thank my advisor Prof. Doron Zeilberger for suggesting the problem, making me feel at ease to walk in into his office for help and permission to use his seed package that saved time.

[K1] David A. Klarner, *Cell Growth Problems*, Canad. J. Math. **19** (1967), 851-863.

[R] Ronald C. Read, *Contributions to the Cell Growth Problem*, Canad. J. Math. **14** (1962), 1-20.

[R] Ronald C. Read, *Graph Theory and Computing*, Canad. J. Math. **14** (1962), 1-20.

[S] Neil Sloane http://www.research.att.com∼njas/sequences/

[SP] N.J.A. Sloane and S. Plouffe, " The Encyclopedia of Integer Sequences", Academic Press, 1995

[Z0] Doron Zeilberger, *Symbol-Crunching with the Transfer-Matrix Method in Order to Count Skinny Physical Creatures*,INTEGERS (`http://www.integers-ejcnt.org`), **0** A9 (29 pages)(2000)

[Z1] Doron Zeilberger, The Umbral Transfer-Matrix Method III, Counting Animals, New York J. of Mathematics,7(2001) 223-231