Enumeration of Self-Avoiding Walks in a Lattice

Tom Harvey

28th October 2011

Abstract

A self-avoiding walk (SAW) is a path on a lattice that does not pass through the same point more than once. We develop a method for enumerating self-avoiding walks in a lattice by decomposing them into smaller pieces called tiles, solving particular cases on the square, triangular and cubic lattices. We also show that enumeration of SAWs in a lattice is related to enumeration of edge-connected shapes, for example polyominoes.

1 Introduction

A self-avoiding walk (SAW) is a path on a lattice that does not pass through the same point more than once. This means the path may not cross the same edge or vertex twice. The problem of enumerating self-avoiding walks (SAWs) has applications in many fields of science, but unfortunately studying even small cases is difficult. The methods developed here are specifically interested in enumerating paths between two points on the boundary of a finite lattice, though they can be extended to more general cases and to other problems.

2 Building SAWs With Tiles

Consider a SAW on a $n \times m$ square lattice going from the bottom left corner to the top-right corner. Divide the walk into n horizontal strips of width m, with each division including the lower horizontal line and excluding the upper one (so the topmost horizontal edge is not contained in any strip). Each of these strips contains a finite number of edges of the underlying lattice, which are either included in or excluded from the walk itself, meaning the number of possible configurations is finite. We will call these strips *tiles* and the underlying lattice of a single strip the *base tile*.

(a) the base tile (b) an example tile (c) entrances and exits

Figure 1: Black edges are included in the tile, grey edges are excluded.

As illustrated in the figures above, each tile has *exits* at the top, and *entrances* at the base. The SAW is constructed by stacking the tiles so that the entrances of one tile correspond exactly to the exits of the tile below.

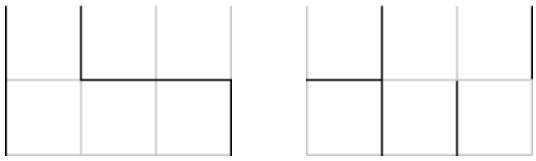
Not all permutations of the base tile can actually be used in the construction of a valid SAW. In fact a tile is admissible only if it has an odd number of entrances and exits, and the conditions of a SAW are not violated within the tile.

(a) (b)

Figure 2: (a) This tile contains a self-intersection and therefore cannot be used in the construction of a self-avoiding walk. (b) This tile is invalid because it has an even number of entrances and an even number of exits - 2 in each case.

One entrance to a tile must be used when the path first enters the tile. If a tile has more than one entrance, the second entrance would then have to be used to leave the tile from the direction it was entered, the pattern repeated for each odd and even entrance. This means that if a tile has an even number of entrances, the last one will always be used to leave the tile in the wrong direction, meaning it cannot be a part of a SAW. A similar argument demonstrates why tiles with an even number of exits can also not be part of a SAW.

In addition to the above restrictions, not every tile can be stacked on top of every other tile when forming a SAW.



(a) (b)

Figure 3: (a) Tiles that fit together can be used to construct a SAW. (b) These tiles don't fit together, even though the lower tile has the same number of exits as the upper tile has entrances.

The information on which tiles can precede each tile can be expressed as a system of recurrence relations.

Example 2.1. *Consider SAWs on the $n \times 3$ square lattice from the lower left corner to the top right corner. The tiles for this lattice will all be of width 3. Figure 4 contains a list of all possible tiles, with the admissible tiles labelled for later use.*

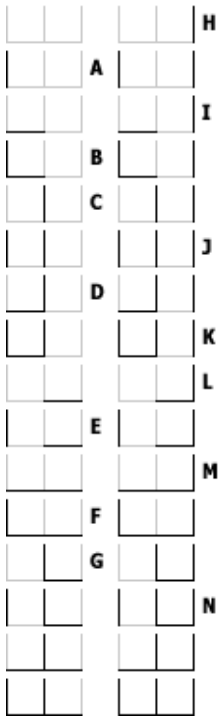


Figure 4: Tiles on the $n \times 3$ square lattice.

We see that, for example, tile A can be preceded by tiles A, B, E and F.

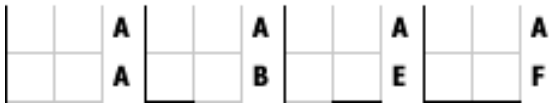


Figure 5: Predecessors of A.

This gives the equation:

$$A_{n+1} = A_n + B_n + E_n + F_n$$

where A_n is the number of SAWs in which the nth tile from the bottom is A, and similarly for the other tiles:

$$\begin{aligned}
B_{n+1} &= C_n + D_n + G_n \\
C_{n+1} &= C_n + D_n + G_n \\
D_{n+1} &= A_n + B_n + E_n + F_n \\
E_{n+1} &= J_n + K_n \\
F_{n+1} &= H_n + I_n + L_n + M_n \\
G_{n+1} &= H_n + I_n + L_n + M_n \\
H_{n+1} &= H_n + I_n + L_n + M_n \\
I_{n+1} &= J_n + N_n \\
J_{n+1} &= J_n + K_n + N_n \\
K_{n+1} &= H_n + I_n + L_n + M_n \\
L_{n+1} &= C_n + D_n + G_n \\
M_{n+1} &= A_n + B_n + E_n + F_n \\
N_{n+1} &= A_n + B_n + E_n + F_n
\end{aligned}$$

or expressed in matrix form:

$$\begin{bmatrix} A_{n+1} \\ B_{n+1} \\ C_{n+1} \\ D_{n+1} \\ E_{n+1} \\ F_{n+1} \\ G_{n+1} \\ H_{n+1} \\ I_{n+1} \\ J_{n+1} \\ K_{n+1} \\ L_{n+1} \\ M_{n+1} \\ N_{n+1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} A_n \\ B_n \\ C_n \\ D_n \\ E_n \\ F_n \\ G_n \\ H_n \\ I_n \\ J_n \\ K_n \\ L_n \\ M_n \\ N_n \end{bmatrix}.$$

3 The Init and Cap Vectors

In the same way that the predecessor matrix encodes path building information, we need two other objects. The *init vector*, denoted $\vec{T}$, encodes information about the starting point of the path. The *cap vector*, denoted $\vec{C}$, encodes information about the end point of the path.

Not all tiles can be used as the base of a SAW. Only those tiles with a single entrance, coinciding with the desired start point of the SAW can be used as the base. Each tile's entry in the init vector is the number of paths from the start point through the tile. In this case 1 if it can be used as the base, 0 if it cannot.

The cap vector is a row vector and is constructed in a similar fashion to the init vector. Each entry in the vector indicates how many ways there are of constructing a path from the tile to the end point. In this case though we include cases where some or all of the top edge of the lattice is added back in (recall that it is excluded from tiles). Like the init vector, not all tiles can be used as the final tile in a SAW and the entry for these will be 0, but unlike the init vector, the number of ways of ending a SAW is not limited to 1.

All of this information can be combined to count the total number of n-tile SAWs $P(n)$:

$$P(n) = \vec{C}\mathbf{M}^{n-1}\vec{T}.$$

Example 3.1. *Returning to the $n \times 3$ square lattice example from before, the init and cap numbers are as follows:*

Tile	*Init*	*Cap*
A	*1*	*1*
B	*0*	*1*
C	*0*	*1*
D	*1*	*1*
E	*0*	*1*
F	*0*	*1*
G	*0*	*1*
H	*0*	*1*
I	*0*	*1*
J	*0*	*1*
K	*0*	*0*
L	*0*	*1*
M	*1*	*1*
N	*1*	*1*

(a) (b)

Figure 6: (a) The cap value for tile K is 0 since adding the top left ledge produces a loop and adding the top right edge results in a path leaving the end point. (b) The cap value for tile N is 1 since adding the top left edge constructs a path from the tile's entrance to the end point in the top right.

This gives:

$$\vec{T} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \vec{C} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

So for the $n \times 3$ case the general solution is

$$P(n) = \begin{bmatrix} 1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,0\,1\,1\,1 \end{bmatrix} \begin{bmatrix} 1\,1\,0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,1\,1\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,1\,1\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0 \\ 1\,1\,0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,1\,1\,0 \\ 0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,1\,1\,0 \\ 0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,1\,1\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,1\,1\,0 \\ 0\,0\,1\,1\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0 \\ 1\,1\,0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0 \\ 1\,1\,0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0 \end{bmatrix}^{n-1} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}.$$

For example,

$$P(2) = 12.$$

4 Extensions

Using the tileset method described above allows arbitrary extension in one dimension for no additional cost once $\vec{C}$, $\mathbf{M}$ and $\vec{T}$ (essentially the information for the other dimensions) have been calculated. This same method can be applied for lattices of any shape and dimension.

Example 4.1. *Consider the* $2n \times 1$ *triangular lattice:*

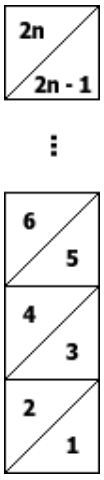


Figure 7: $2n \times 1$ triangular lattice.

When building the tileset for this lattice, we need only consider triangles 2 at a time, since we are only considering the $2n \times 1$ *case. Thus the tiles are:*

Figure 8: Tileset for the $2n \times 1$ triangular lattice.

The relevant information for each tile is:

Tile	*Pred*	*Init*	*Cap*
A	*ABCD*	*0*	*1*
B	*EF*	*1*	*1*
C	*EFG*	*1*	*1*
D	*ABCD*	*0*	*1*
E	*EFG*	*1*	*1*
F	*ABCD*	*0*	*1*
G	*ABCD*	*0*	*0*

giving

$$P(2n) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}^{n-1} \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}.$$

We can use these calculations to generate the full $n \times 1$ case by observing that $\vec{C_o} = \vec{C_e} + \vec{J}$, where $\vec{C_o}$ is the cap vector in the odd case, $\vec{C_e}$ is the cap vector in the even case, and $\vec{J}$ is the appropriately-dimensioned vector of all 1s. This is illustrated in figure 9.

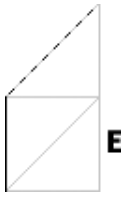


Figure 9: With an extra triangle present, all tiles have an additional path to the exit using the diagonal of the new triangle, as demonstrated here with tile E.

So we simply need to substitute the cap vectors depending on the parity of n, the number of triangles in the lattice we are studying. This gives the following general equation for the $n \times 1$ triangular lattice (relabelling C_e as C*):*

$$P(2n + a) = (\vec{C} + a\vec{J})\mathbf{M}^{n-1}\vec{T}$$

where $a \in \{0,1\}$ and $P(1) = 2$.

The upper bound on the total number of tiles is 2^e, where e is the number of edges in the tile, since each edge can be either included or excluded from the tile. Of course not every combination of edges yields a valid tile for studying SAWs.

Example 4.2. *Hexagonal lattices can be built up from the base tile in figure 10.*

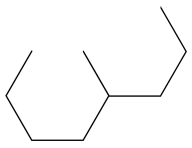


Figure 10: The base tile for the hexagonal lattice of width 2.

There are 8 edges in this tile, meaning there are at most $2^8 = 256$ *tiles in the tileset for the hexagonal lattice of width 2.*

For a square lattice of dimension n, the base tile consists of a cube of dimension n, minus a cube of dimension $n-1$ at the top (in the 2D case, the top edge of the square). Since the number of edges in an n-dimensional cube is $n2^{n-1}$, we have for $T(n)$ the number of tiles in a square lattice of dimension n,

$$T(n) \leq 2^{n2^{n-1}-(n-1)2^{n-2}}$$

Example 4.3. *Consider counting SAWs on the* $n \times 1 \times 1$ *cubic lattice from the bottom left corner to the top right corner as shown in figure 11.*

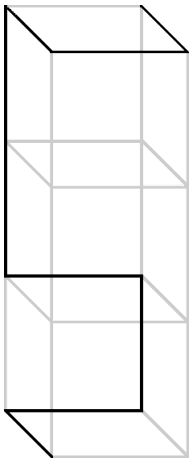


Figure 11: An example SAW on the $3 \times 1 \times 1$ cubic lattice.

This has the following tileset (of 80 tiles) with init and cap values as shown:

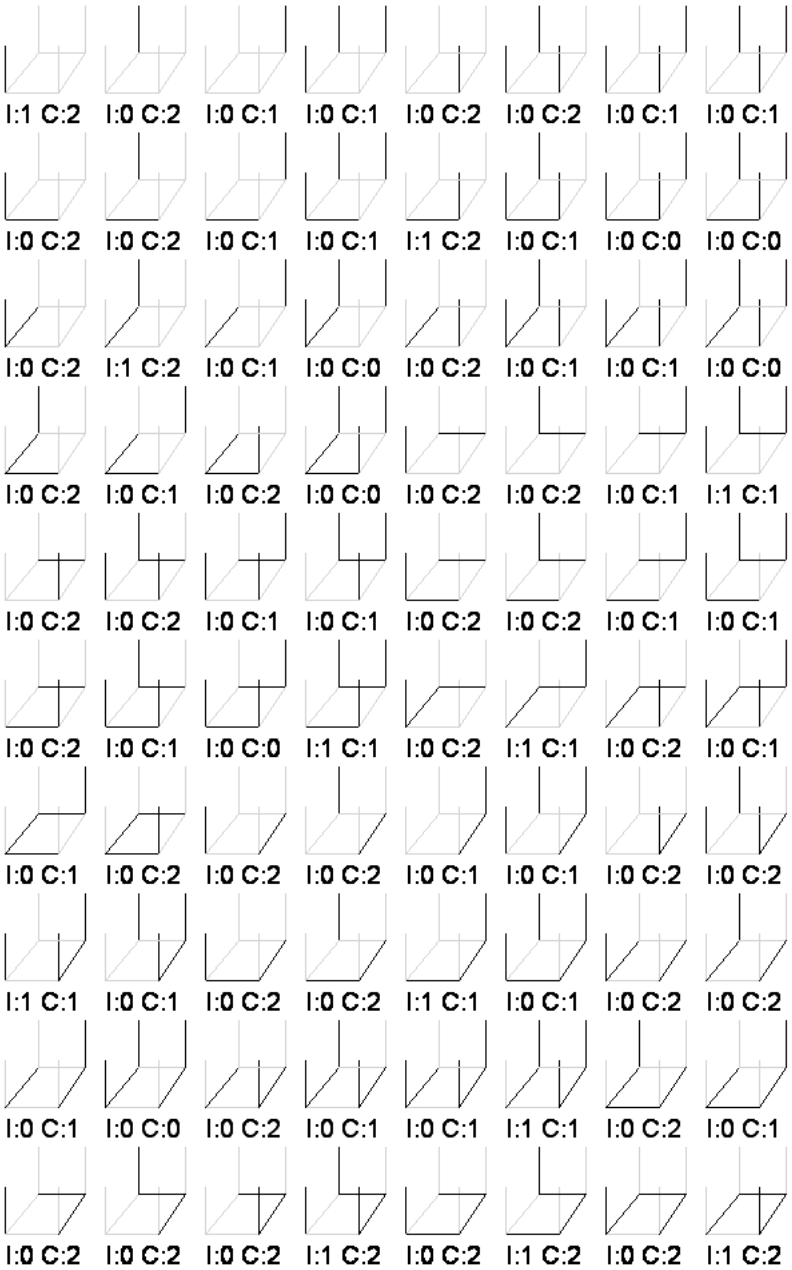


Figure 12: Tileset for the $n \times 1 \times 1$ cubic lattice, with init and cap values as indicated.

These tiles and the following sequence for P(n) were computer generated:

n	$P(n)$
1	*18*
2	*172*
3	*1806*
4	*18716*
5	*194418*
6	*2018970*
7	*20967460*
8	*217750292*
9	*2261373016*
10	*23484730792*
11	*243892800816*
12	*2532866929616*
13	*26304240502368*
14	*273173872736160*
15	*2836955689382080*

It is apparent that $P(n) \approx 18(10.38)^{n-1}$.

5 An Alternative View

Another way of viewing the problem of SAWs in a 2D lattice is to embed it in $\mathbb{R}^2$.

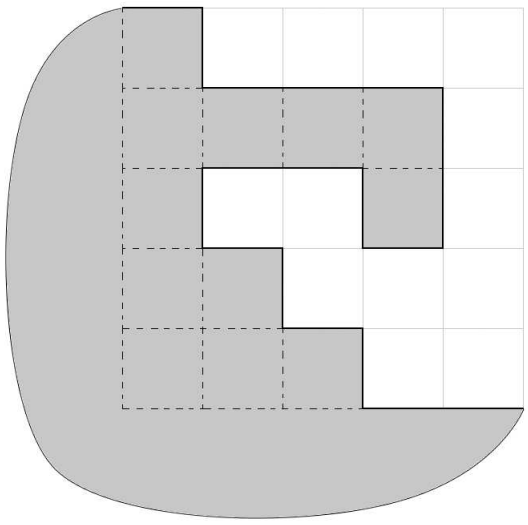


Figure 13: An alternative way of viewing SAWs in a lattice.

Using the Jordan Curve Theorem we can see that any SAW through the start and end point will separate the lattice into 2 regions. Colour one region. The enumeration of paths then becomes equivalent to the study of arrangements of connected, coloured shapes (squares in the case pictured) in the lattice - similar to polyominoes but with more restrictions.

References

[1] L. K. Williams, Enumerating Up-Side Self-Avoiding Walks on Integer Lattices, in *Electr. J. Comb. Vol. 3, No. 1* (1996)