

Elsevier Editorial System(tm) for Discrete Applied Mathematics

Manuscript Draft

Manuscript Number:

Title: Impossible Paths in Langton's Ant

Article Type: Contribution

Keywords: Langton's Ant; ant-like automata; Highway; Path; Cell States

Corresponding Author: David Dakota Blair,

Corresponding Author's Institution: Texas A&M University

First Author: David Dakota Blair

Order of Authors: David Dakota Blair

Abstract: We develop a new characterization of a path (as a binary word) which provides a way to study the paths produced in Langton's Ant and other ant-like automata. This characterization seems to be new to the literature. Using this, one sees that there are words for which there are no possible paths. An algorithm to determine, for a given word, whether it corresponds to a possible path is presented. Surprisingly this also yields the set of cells which initially must be "on" in order to produce the path.

Dakota Blair
215 Rugen Lane
College Station, TX
(903)235-3248
dakota@tensen.net

Dear Dr. Peter L. Hammer,

I have submitted a manuscript to the journal Discrete Applied Mathematics entitled "Impossible Paths in Langton's Ant". I have not submitted this manuscript to any other journal and this is my first submission to your journal. I have seen other articles in the journal on similar topics and I think this one will add to the subject. In the language of the previous literature, this takes the focus of the search for alternate highways in Langton's ant off the initial state configuration and places it on the sequence of cell states which make up the highway. The previous articles I have read in your journal relating to this subject were of a more general computational nature, using the overall structure of this automata to prove it was sufficient to create a structure such as a universal turing machine. This article deals more with the consequences of characterizing the paths by the sequence of cell states which yields information about the automata itself. I hope you will find it worthy of publication.

Thank you for your consideration,

Dakota Blair

Impossible Paths in Langton's Ant

Dakota Blair

215 Rugen Lane

College Station, TX 77845

Abstract

We develop a new characterization of a path (as a binary word) which provides a way to study the paths produced in Langton's Ant and other ant-like automata. This characterization seems to be new to the literature. Using this, one sees that there are words for which there are no possible paths. An algorithm to determine, for a given word, whether it corresponds to a possible path is presented. Surprisingly this also yields the set of cells which initially must be "on" in order to produce the path.

Key words: Langton's Ant, ant-like automata, highway, path, cell states

1 Introduction and Preliminaries

Previous articles on this subject have dealt with a multitude of issues. Many of these have the central goal of understanding the highway, the so called 'transient symmetry' of Langton's Ant. The highway is a sequence of 104 iterations which forms a path that has the property that once the ant begins this sequence it will continue indefinitely unless an obstacle is encountered. As with many simple questions in mathematics, the answer, although elusive, has generated many levels of study within this topic. For example, the notion

of bilateral symmetry is explained by [2] once a suitable picture is developed. Therefore a suitable picture may be needed to understand the highway. This paper develops the idea of a path and produces an algorithm which, for a given path characterized as a sequence of cell states, produces the initially needed cells that are needed to create the path. We begin by giving a number of formal definitions which we will use.

The mathematical entity we are concerned with is the *ant*. It is characterized by an ordered pair of a position and a direction. A *state* is a number which determines the ant's behavior. A position on the lattice (usually $\mathbb{Z} \times i\mathbb{Z}$) which is in one of a number of states is called a *cell*. When the word 'cell' is used, it is considered to be both the position and the state which it is in. A *rule* is a string of instructions which specifies a rotation of the direction of the ant is in based on the state of the cell that it is currently on. The length of the rule determines the number of states s . For example RL, the rule for Langton's Ant, specifies a right turn for the 0(off) state and a left turn for the 1(on) state[1]. An automata where an ant moves to the next node in its current direction, increments its previous cell's state (mod s) and then changes its direction based on the instruction for the given state in the rule being considered we will refer to as an *ant-like automata*. Note that this definition encompasses automata on almost any type of lattice. The following definition is crucial for understanding the algorithm presented.

Definition 1 (Path) *The sequence of positions and directions an ant travels.*

Given a sequence of directions, the path may be considered the partial sums of the sequence, paired with the sequence itself. Since the directions are based upon the state of the cells upon which the ant lands, the sequence of cell states

uniquely determines the path. Additionally for lattices which may be viewed as lying in the complex plane, the directions may be considered numbers of the form $e^{i\theta}$. Therefore the n th direction of the ant is the product of all the direction changes specified by the sequence of cell states. The following equation succinctly describes the path, given a sequence of cell states C_k :

$$z_N = z_0 + \sum_{j=0}^N \prod_{k=0}^j \delta(C_k) \quad (1)$$

Where $\delta(n)$ gives the direction specified in the n th letter of the rule, z_N is the ant's position at the N th iteration and z_0 is where the ant began. From this point on z_0 will be assumed to be the origin due to Theorem 3.

Note 1 *It is important to make a distinction between the sequence of cell states and the path itself since there are sequences for a given rule for which no path exists.*

Given an initial state and a sequence of cell states, a *contradiction* arises when the cell state specified by the sequence is different from the cell state of the position on the lattice currently occupied by the ant. A node on the lattice the ant has already been to previously on a current set of iterations is said to have been *touched*. A path's *initially needed cells* are the cells which are necessary to produce a given path. These cells will be referred to as members of the set S_0 which may also be used to tell whether or not a cell has been touched before. An *irreparable contradiction* is a contradiction which occurs at an iteration n where the current cell has already been touched by the ant before the iteration n . Conversely, a *reparable contradiction* is when a contradiction occurs and it is not irreparable. The number of reparable contradictions a sequence of cell states has will determine how many cells must initially be in a nonzero state to

produce the sequence. An *impossible path* is a sequence of cell states $\{C_i\}_{i=0}^{n-1}$ such that there is exactly one irreparable contradiction at iteration $n - 1$. It may be more correct to call this an ‘impossible word’, but since no group structure is immediately apparent, using the term ‘word’ may be misleading. An impossible path of length n that contains another impossible path that is of a length less than n is a *trivial* impossible path. An impossible path is considered *nontrivial* if it is not trivial.

Definition 2 (Alternating Orientation Property) *Cells an even distance from the origin are labeled H and cells an odd distance from the origin are labeled V .*

This property applies to rules which only have right and left quarter turns in the rule and are on a square lattice. Since the ant turns a quarter turn every iteration, then the cells of the grid may be labeled H and V according to their distance from the origin[2]. This amounts to a chessboard coloring of the lattice. The H and V refers to the orientation (horizontal, vertical respectively) of the ant at a given iteration.

Note 2 *By convention, the ant’s position at the 0th iteration is the origin, and when given a sequence of cell states of length n , the ant’s path is defined from the 0th iteration to the $(n - 1)$ th iteration. The origin is chosen without losing generality as an H cell. This determines the coloring of the plane.*

2 A Characterization of Paths

In general, a path is simply the sequence of ant positions and directions. Defining a path of length n would require $2n$ values. For Langton’s Ant the

positions and directions can be assumed to be members of $\mathbb{Z} \times i\mathbb{Z}$, thus a path of length n in Langton's Ant requires $4n$ integer values to be represented in this format. Since we know from (1) a way of writing the path using only n integer values, it seems natural to attempt to do so. In fact, any ant-like automata which may be viewed as moving on the complex plane may be written in the form of (1). By assigning a cell state value to a direction via defining the function $\delta : \{0, 1 \dots s - 1\} \rightarrow \mathbb{C}$, where s is the length of the rule being used and each integer n is mapped to the direction $\delta(n)$ specified by the n th letter of the rule, we may write the path as a sequence of cell states.

Example 2.1 (The Principal Path)

The principal path of an ant-like automata is the path which is produced when the ant starts on an empty grid, i.e. a grid in which all the cells are in state 0. The sequence of cell states of the principal path of Langton's Ant is

000010000100001111101000010000111101000010...

The principal path for the rule RRL is

00001111200001111200001111222202000011112...

The principal path shows that similar rules can produce vastly different behaviors.

Example 2.2 (Highways)

In addition to Langton's Ant, many ant-like automata form highways. These paths have very interesting properties that are still not fully understood. For example, it is still unknown whether or not every initial state with a finite

number of cells which are in a nonzero state ends up building a highway. Some automata have multiple highways, but there is only one known for Langton's Ant. It has been documented many times that the length of this path is 104, but as far as this author knows, no representation of the exact steps has yet been presented. The first appearance of the highway occurs at iteration 9977. The rule RRL also produces a highway, but in far fewer iterations. Studying other, shorter highways such as this one may provide insight into constructing highways for different rules. It is also interesting to note that the highway for the rule RRL is short (length: 18) compared to the highway for Langton's Ant (length: 104). The sequences presented here are the highways of these rules, but this is not the way in which they first appear in the principal path. The highway of Langton's Ant:

```
0000100100001111010000100001111010000101100111100110...
...0001100101101101011000011011000011000011010000101111
```

The highway of rule RRL:

```
000011112100022212
```

The reader may note that the above representation of a path is not sufficient for ants with more than 10 states. In this case the author recommends delimiting the iterations by commas.

2.1 Impossible Paths

It is necessary first to show that there exist words which do not correspond to paths in Langton's Ant. This is not difficult to do since we may consider the sequence of cell states 00000. There are three more words of length 5, including 11111, which are impossible. To understand how the algorithm works, it is a good exercise to attempt to find the other words. If intuition fails, there are only 30 words left to check. Additionally, one may take any finite length of the principal path such that the last state is 1, change it to a 0, and thus form an impossible path. Most of the time this will be a trivial impossible path since doing this will make the last 5 iterations 00000. These paths are impossible for the same reason as 00000 is impossible. That is, the cell that needs to be changed to avoid a contradiction has already been *touched*. For the principal path, any cell in the 1 state has been touched an odd number of times and thus was not on to begin with. This is the nature of contradictions in the sense of this characterization of the path. Now that we know that impossible paths exist, we may say some things about them in general.

Theorem 3 *Nontrivial impossible paths begin and end at the origin.*

Proof: It is always assumed that the ant begins at the origin at iteration 0. Assume there is a nontrivial impossible path $(C_j)_{j=0}^{n-1}$ that has an irreparable contradiction at some place other than the origin (called z) at the iteration $n-1$. From the definition of an irreparable contradiction, we know the cell that the ant is on at the iteration of the contradiction has been touched at least once before. Call the most recent iteration where the ant was on this cell k . Since the iteration $n-1$ is a contradiction, we may assume that $(C_k + 1) \pmod{2}$

$s) \neq C_{n-1}$ where s is the total number of possible cell states. The path from k to $n - 2$ is possible since the original path included this path. The ant must now move onto the cell z which has state $(C_k + 1)(\text{mod } s)$. This is an example of an irreparable contradiction. This cell has been touched before, but the cell state is not C_{n-1} as required by the sequence. Therefore the path from k to $n - 1$ is impossible. Thus we have created an impossible path of length $n - k$ beginning and ending at z , namely $(C_j)_{j=k}^{n-1}$. This contradicts C_j being nontrivial. We may now call z the origin and note $(C_j)_{j=k}^{n-1}$ is a nontrivial impossible path which begins and ends at the origin. Q.E.D.

Corollary 4 *Nontrivial impossible paths must be of odd length and thus irreparable contradictions may only occur on even iterations.*

Proof: This is because any irreparable contradiction must occur at the origin and the ant may only be at the origin when the iteration is even due to the Alternating Orientation Property. Q.E.D.

Corollary 5 *Irreparable contradictions of nontrivial impossible paths must occur on iterations that are multiples of 4, thus the sequences of cell states must be of length $4n + 1$.*

Proof: By Corollary 4, we may consider only the even iterations. On these iterations the ant is on H cells. Therefore it only moves horizontally one cell at a time, either toward or away from the origin. Because of this, the horizontal distance from the origin is odd when the iteration $n \equiv 2(\text{mod } 4)$ and the horizontal distance from the origin is even when $n \equiv 0(\text{mod } 4)$. Therefore by Theorem 3, the fact that the distance from the origin is even the ant may only be on the origin on iterations that are multiples of 4, which proves the corollary. This also proves that there are no impossible paths of

length ≤ 4 , furthermore there are no nontrivial impossible paths of a length n where $n \not\equiv 1 \pmod{4}$. Q.E.D.

3 An Algorithm

Recall that for the impossible path 00000 we started with every cell having state 0. When the ant returns to the origin, the state of the origin is 1, but our path dictates that it be 0. We could change the state to 0, but in that case the origin should have been 1 to start with. But since the origin has already been *touched* by the ant, changing its color would change the path. This is the essence of an irreparable contradiction. Now consider if we wanted to test if the path 00011 is possible. When we proceed to the 3rd iteration, we see that we have a contradiction. The cell state is 0 since it has not yet been touched by the ant. We see that our path dictates that the cell should be in the 1 state. Since the ant has not touched this cell yet, changing the cell state will not alter the path. This is the type of contradiction which can be circumvented by altering the *original* state. Therefore, this cell is required to be in a nonzero state to produce the path so it belongs in the ‘needed initial’ set. So now we have three values to consider: the state which the path dictates, the state the current cell is in and whether or not the cell has been touched. This truth

table describes the situation thus far:

Touched	Cell	Path	Contradiction	*
0	0	0	0	
0	0	1	1	
0	1	0	1	*
0	1	1	0	*
1	0	0	0	
1	0	1	1	
1	1	0	1	
1	1	1	0	

The starred rows would not occur if we began with no cells in state 1. Under this assumption the ant would have already touched any cell with state 1, that is, a cell would have to be touched for the cell to be in state 1. Eliminating these rows, we may then consider what to do with each case. Let the set of

touched cells be denoted as T and the set of initially needed cells S_0 .

Touched	Cell	Path	Contradiction	Action
0	0	0	0	Add cell to the set T .
0	0	1	1	Add cell to the sets T and S_0 .
1	0	0	0	Do nothing.
1	0	1	1	This path is impossible.
1	1	0	1	This path is impossible.
1	1	1	0	Do nothing.

We see that there are situations where there is nothing to do, thus we may eliminate these from consideration. Now our table is

Touched	Cell	Path	Contradiction	Action
0	0	0	0	Add cell to the set T .
0	0	1	1	Add cell to the sets T and S_0 .
1	0	1	1	This path is impossible. (eg. 000010001)
1	1	0	1	This path is impossible. (eg. 00000)

Now notice that what we need to do is determined only by the touched and contradiction columns. Inspection also shows that only the set S_0 is needed, since T is the set of positions where the state must be 0 unioned with S_0 . Therefore this shows our algorithm only needs to follow this table to determine

the validity of the path:

Touched	Contradiction	Action
0	0	Add cell to the set S_0 .
0	1	Add cell to the set S_0 .
1	1	This path is impossible.

The other case, where there is no contradiction, but the cell has already been touched is the case where nothing needs to be done. Therefore we may now formulate the algorithm for any ant-like automata. In the algorithm that follows, the possibility of the path is determined in step 1.3. Therefore the truth table of step 1.3 must agree with the table above, which it does by design. The algorithm must also return, for a possible path, the cells and states which are required for the given path to be produced. The set S_0 is defined in this way, thus no further computation is necessary. Since S_0 contains the states that are initially required to make the path possible, each cell in S_0 is required for the path to be produced. Furthermore, at points in the algorithm we must be allowed to query the state of cells and properties of the ant in some way. To do this, we define $Pos(A)$ to be the position of an ant A , similarly its direction is given by $Dir(A)$. To determine the state of a node P on the lattice we use $State(P)$. The length of the rule is s as above. The set S is the set containing the lattice states and S_0 is the set of cells and their respective states which are necessary in order to produce the given path. Any line labeled ‘Output’ ends the program.

3.1 The Algorithm

Input:	$(C_j)_0^{n-1}$, Rule, $\delta : \{0 \dots s - 1\} \rightarrow \mathbb{C}$
0:	Define $S = S_0 = \{((0, 0), (C_0 + 1)(\text{mod } s))\}$ Define $A = ((0, 0), (0, 1))$
1:	For $j = 0$ to $n - 1$ do
1.1:	Set $State(Pos(A)) = (State(Pos(A)) + 1)(\text{mod } s)$
1.2:	Set $Pos(A) = Pos(A) + Dir(A)$
1.3:	If $(State(Pos(A)) \neq C_j)$
1.3.1:	If $Pos(A) \in \{P : (P, C_P) \in S_0\}$ Output: This path is impossible
1.3.2:	Else Set $State(Pos(A)) = C_j$
1.4:	$Dir(A) = Dir(A) * \delta(State(Pos(A)), Rule)$
1.5:	If $Pos(A) \notin \{P : (P, C_P) \in S_0\}$
1.5.1:	$S_0 = S_0 \cup (Pos(A), State(Pos(A)))$
Output:	S_0

3.2 *Immediate Consequences*

We now know how to determine whether or not a sequence of cell states correspond to a possible path and in doing so we also determine which cells are necessary to produce the given (possible) sequence. Therefore we may examine the number of cells which are required to start in a nonzero state to produce a given path. This number shall be called the number of initial cells. Note the principal path is the only path (infinite or finite) which requires no initial cells. It is easy to see that a finite path will require only a finite number of initial cells. Since the path of an ant is unbounded it is trivial that the number of paths which require a finite number of initial cells is infinite, but it is nontrivial that there is an infinite path whose sequence of cell states is a repeating pattern for which there is a finite number of initially required cells. Even though the highway is a well known example of this, its existence is certainly nontrivial as it requires 10185 applications of the algorithm for one to become aware of this fact. Using the algorithm above we can apply a brute force check on the highway and determine the minimum amount of initial cells required to produce it. The following two conjugates (Figures 1 and 2) tie with 11 required initial cells. In the figures, the ant is represented by an arrow and the required initial state is the top picture and the bottom picture is the state after 104 iterations. The outlined cells in the bottom picture reveal once again the initial configuration required for the given path. The cell with the white dot is the origin of the ant.

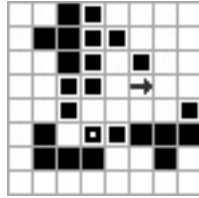
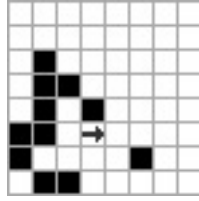


Fig. 1. 00001001000011110100001000011110100001011001111001100001100101101101011000011011000011000011010000101111

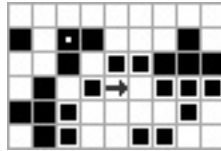
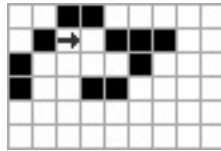


Fig. 2. 00001101100001100001101000010111100001001000011110100001000011110100001000011110100001011001111001100001100101101101011

4 Further Applications

Many programs implementing Langton's Ant exist already and so implement steps 1.1, 1.2 and 1.4 already in some form or fashion. Therefore the inclusion of path testing would be only an extra few lines of code. The advantage of this is that instead of studying the initial state of the lattice, one may study the paths themselves. The highway for example stays in a diagonal region thus the path will not be disturbed unless there is a cell in that specific region.

When it is disturbed, the outcome has more to do with where in the path the cell occurred as opposed to where on the lattice it is.

There is no reason to restrict the algorithm to just two-dimensional ants. By defining the positions and directions as n -dimensional vectors, the range of the function δ may be defined as $n \times n$ rotation matrices. Care must be taken in defining the rules however, since for n dimensions there must be n linearly independent directions specified in the rule, otherwise the ant will be stuck in a subspace. Therefore (1) is still valid and as a result so is the algorithm.

The main application this algorithm has is that of a step in the direction of a proof of the popular unsolved question concerning the highway. The path of the ant is always unbounded given any initial state. For a finite starting state it seems that the highway is always generated, and the ant settles into this repeating path. This may always happen, but there still is the possibility of alternate highways. It is unclear in the literature[3] whether or not “eventually build a highway” means ‘eventually build the known highway of Langton’s Ant’, or if it means ‘never settling into a repetitive pattern of a certain length’. The second problem certainly has more depth than the first. On consideration of the first problem, it seems given enough free space the ant will produce the original highway, consequently if another highway were found it would almost certainly be longer than the original highway which would explain why the original highway is the only one that is known. The algorithm provides a way to search systematically for alternate highways. In addition to this, there are only 2^{104} binary words of length 104, and even fewer possible paths. Therefore a word of length 104 may only be of certain types. A dumb search of all possible paths would classify all these paths. This would reveal if there were any candidates for alternate highways. However, as mentioned before, by

careful study of highways in other rules (especially ones with shorter highways such as RRL), it may be possible to find their common properties and produce a highway in Langton's Ant without a dumb search. One such already apparent property has to do with the initial state. When the path is run once, a certain number of nonzero states will be required. When run twice consecutively there should be no more required initial nonzero states. This reveals another search method employing the algorithm to search for alternate highways.

References

- [1] C.G. Langton, "Studying Artificial Life with Cellular Automata". *Physica D* vol. 2 (1986) pp 120–149.
- [2] D. Gale, "Further Travels with My Ant". *Mathematical Intelligencer*, vol. 17, no. 3 (1995) pp 48–56.
- [3] Eric W. Weisstein. "Langton's Ant." From *MathWorld*—A Wolfram Web Resource. <http://mathworld.wolfram.com/LangtonsAnt.html>