Langton’s Ant Complexity Order

The goal of this project was to write a program to display the progress of Langton’s Ant using P5.hs + Javascript with an HTML display page given the following main constraints:

- • Ant’s movements are ordered as follows:
  - Notices the color of its current cell, then changes direction accordingly
  - Increments the color of the cell underneath it, then finally moving forward to a new cell
- Grid colors correspond to the appropriate direction changes (red/blue turns the ant left, black/yellow/green turns the ant right)
- Grid colors increment: black(0) – red(1) – yellow(2) – blue(3) – green(4) – black(0)
- Ant must take 1000+ steps (n)

To solve this problem, a brute force algorithm based on the above constraints was appropriate. There are 3 steps to this algorithm based on the 4 movements the Ant makes. The input (n) is the number of steps the ant must take, not to be confused with the number of steps in the algorithm. Other inputs that could have been considered for analysis include the number of colors, and the size of the grid. These inputs have constant run time O(1), and were therefore omitted. The steps and their respective run timeevaluations are as follows:

1. Notices the color of current cell and changes direction accordingly
   A. g[x][y]==? && direction ++/-- → O(n)
2. Increments the color of the cell underneath it (wraps around if applicable)
   A. g[x][y]++ → O(n)
3. Moves forward to a new cell (increments x or y position accordingly)
   A. myAnt.move() → O(n)

The overall run time evaluation of the algorithm is the sum of all the algorithm’s steps, so we have the following:

$$O(n+n+n) = O(3n)$$

Result: O(n)

We remove the constant 3 as we are looking for long term growth rate. Therefore the big O notation for our algorithm is linear O(n). The run time of this algorithm increases linearly with respect to the input (n) steps that the ant must take. You can confirm this result through observation by altering the number of steps the Ant must take in the program source code and timing the program execution. You will see a perfectly linear pattern.