Code Jam 2011 - World Finals

Analysis: Program within a Program

Here at the Google Code Jam, we have always tried to encourage a variety of programming languages. But in this problem, we took things a little further by forcing you to program in the language that started them all -- the abstract Turing Machine! Unfortunately, abstract Turing Machines are pretty unwieldy, and that makes even the simple task here quite difficult.

With so few rules allowed, it is important to take advantage of the numbers you can mark onto the lampposts. The obvious approach is to write down, perhaps in binary, the number of steps you need to make, and then have the robot makes decisions based on that. Unfortunately, there is also a rather tight limit on the number of moves, which means you cannot afford to keep going back to the start to check on your number. You will need to take the data with you as you move.

Here is one effective way of doing this:

- First write down the distance you want to go forward in binary, using the values 1 and 2 to represent the binary digits. This information will now be available on the starting lampposts.
- Now repeat the following:
  - Trace through the number from right to left, and subtract 1 as you go.
  - If the number was 0, then drop the cake right now.
  - Otherwise trace through the number from left to right, copying everything one position to the right.

Once you have the high level algorithm, each piece is relatively straightforward to implement. For example, subtracting 1 comes down to formalizing the subtraction algorithm you learned in grade school:

- Start in state 1, which we'll you use to mean you have not yet done the subtraction. If the last digit is 1, you can replace it with 0 and the subtraction is done, so switch to state 2. If the last digit is 0, replace it with 1 but you now need to borrow 1 from the previous digit, so stay in state 1. Either way, move to the previous digit.
- Once you are in state 2, you are just moving left through the number without changing anything.
- If you reach the left end of the number in state 1, then the number must have been 0, so you should drop the cake. Otherwise, you should move onto the copy phase.

And the copy phase is similar but conceptually simpler.

Almost any implementation of this algorithm should be good enough to solve the problem, but there is room for more optimization if you want a challenge! With the 30-rule limit, we were able to bring the number of steps down to about 95,000. Here is the full program to move 5,000 lampposts:

```
0 0 -> E 1 1
1 0 -> E 2 3
2 0 -> E 3 1
3 0 -> E 4 3
4 0 -> E 5 2
5 0 -> E 6 3
6 0 -> E 7 1
7 0 -> E 8 3
8 0 -> W 9 3
```

```
9 0 -> R
10 0 -> E 11 0
9 1 -> W 9 3
10 1 -> W 10 1
9 2 -> W 10 1
10 2 -> W 10 2
9 3 -> W 10 2
10 3 -> W 10 3
11 1 -> E 12 0
12 0 -> W 9 3
12 1 -> E 12 1
12 2 -> E 13 1
12 3 -> E 14 1
13 0 -> W 10 1
13 1 -> E 12 2
13 2 -> E 13 2
13 3 -> E 14 2
14 0 -> W 10 2
14 1 -> E 12 3
14 2 -> E 13 3
14 3 -> E 14 3
```

Can you do better? We'd love to hear about it if so!