Kindergarten Line

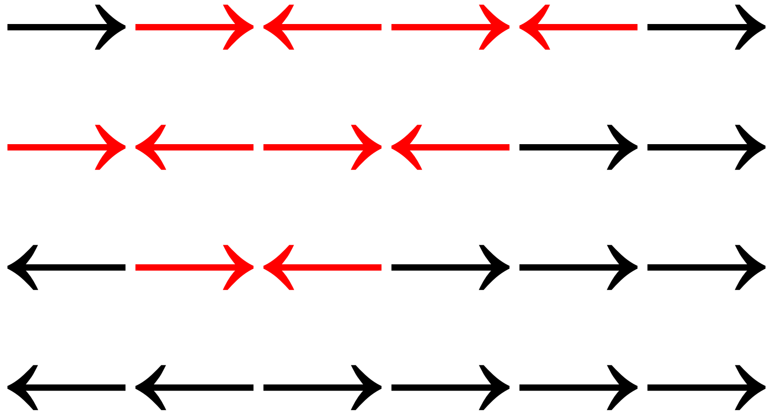
By Andrew Rogers

https://github.com/andrewerogers/Kindergarten-Line-Algorithm

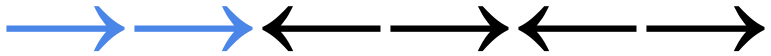
Problem Statement

A kindergarten teacher is helping students learn the directions LEFT and RIGHT. The students line up shoulder-to-shoulder, facing forward. “Turn left,” says the teacher. Of course, there are mistakes. Some turn left and some turn right. So the teacher gives a correction command, “If you turned the wrong way, fix yourself,” several times. At each command, if a kindergartener is facing a neighbor, both assume they’ve made a mistake and turn around. Does the line become stable after a while?

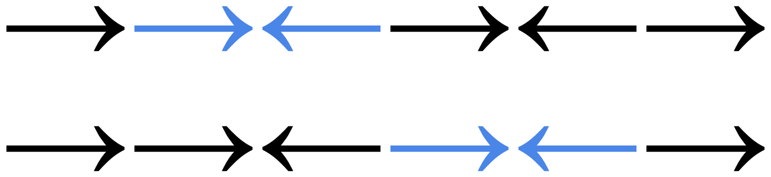
Example



Modeling Behavior



```
if: students are facing toward each other
    students turn around
    compare the next group of two students
else: compare the last student and the next student
```

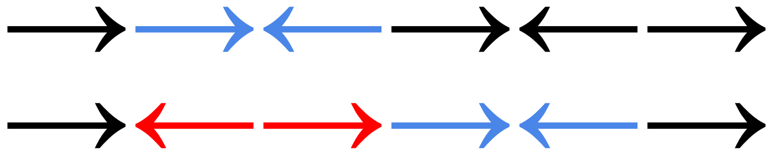


Developed Algorithm

- This algorithm simulates n rounds of students “turning around”

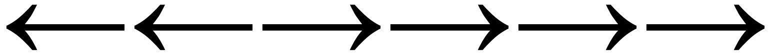
```
for(integers j less than the length of the line):
    for(integers i less than the length of the line):
        if(students i and i + 1 are facing each other):
            students i and i + 1 turn around
            i++
```

- Notice that we must increment j in the “if block.” We define a rule that each student can turn at most once per pass of the algorithm.

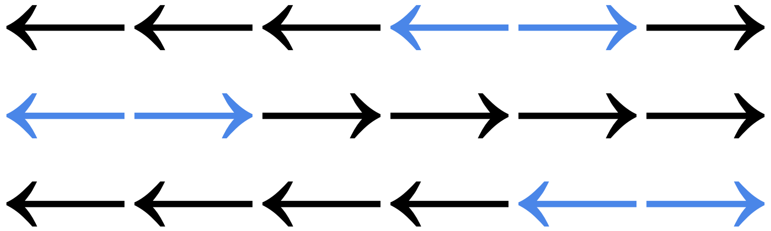


What Happens?

- All lines of length n where n is a positive integer converge to a predictable structure.
- What is the structure? Recall from our previous example



- More examples



Description of Structure

- We need to define our structure more formally.
- The structure the unstable lines converge to has exactly one “bifurcation point” in which the two students at this point in the line face away from each other.
- All other students face in the same direction.
- If this bifurcation point is not present in the line, we are guaranteed that the line was already stable.

Proof of Correctness

- We will prove the correctness of our solution using the Principle of Mathematical Induction.
- First, consider the base case (n = 2). The possible combinations are

$$\longleftarrow\longleftarrow \qquad \longrightarrow\longrightarrow \qquad \longrightarrow\longleftarrow \qquad \longleftarrow\longrightarrow$$

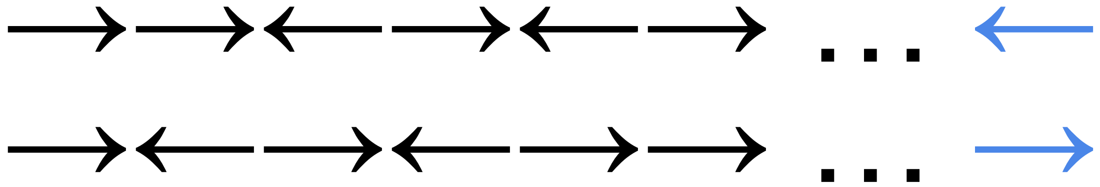
- Each state is stable except one but after one iteration we have

$$\longrightarrow\longleftarrow$$

$$\longleftarrow\longrightarrow$$

Proof of Correctness

- Now, suppose that any unstable line of length k converges to the structure we predicted. Consider a line of length k+1.



- The k+1th (in blue) element in the line **eventually faces outward** unless the line is already stable. From our induction hypothesis, we are guaranteed the remaining k elements eventually converge to the structure that we predicted.
- If we add an outward facing element to the structure, the new structure retains the properties of our predicted structure.

Time Complexity Crash Course

- Let f and g be functions from R to R. We say that “f(x) is O(g(x))” provided that there exists positive constants C and k such that

  $$|f(x)| \leq C|g(x)|$$

  whenever $x > k$.

- We can evaluate the complexity of our algorithm by developing a function that approximates the number of comparisons performed by the algorithm as n gets large.

Worst Case Complexity

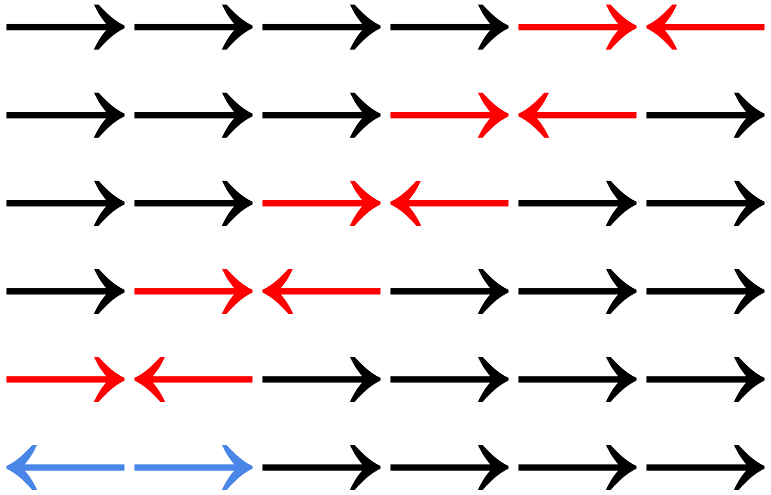
- Recall our algorithm

```
for(integers j less than the length of the line):
    for(integers i less than the length of the line):
        if(students i and i + 1 are facing each other):
            students i and i + 1 turn around
            i++
```

- Because we have two loops with n operations, our comparison function is f(n) ≈ n^2 so our algorithm is $O(n^2)$. Can we do any better?
- We must sequentially search through each pair in the line, but do we need n passes of the algorithm to arrive at a stable structure?

Worst Case Complexity

- Consider the following example



- Notice that we require n passes of our algorithm. So, we can’t do any better than O(n^2) in the worst case.

Average Case Complexity

- As you may have realized, the probability of observing such structures is very low.
- For a line of length n, we have 2^n different possible combinations.
- In the average case, we can reduce our time complexity to $O(Cn^2)$ where C is some real number between 0 and 1.
- In experimentation, $C \approx 0.6$ (so we would require .6n passes of our algorithm).
- We still consider this $O(n^2)$ time complexity.

Questions

- Thank you to Dr. Long and Dr. Glendowne for assisting me on this particular problem.
- Thank you to all the professors and classmates who taught me so much here at SFA!
- Python code is at https://github.com/andrewerogers/Kindergarten-Line-Algorithm if you would like to view it.