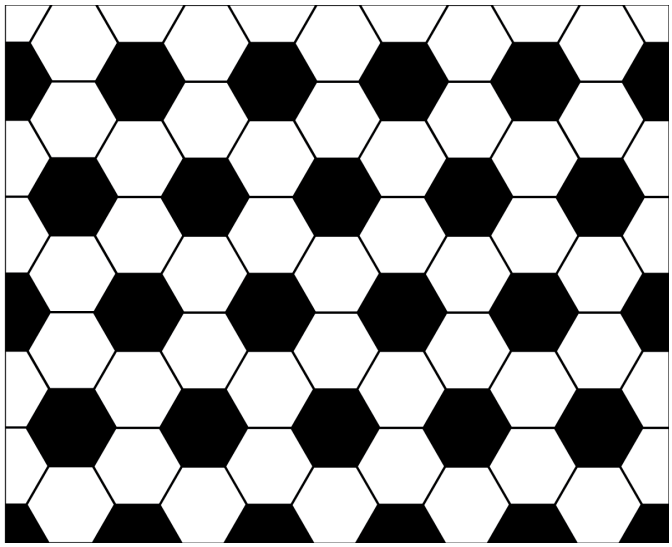
Jane Street Puzzle July 2022 Solution

Leuven Wang

2022-07-29

The Problem



Andy the ant has spent most of his days living on a strange land consisting of white hexagons that are surrounded by alternating black pentagons and white hexagons (three of each), and black pentagons surrounded by five white hexagons. To us this land is familiar as the classic soccer ball we see above on the left. Due to Andy's tiny size and terrible eyesight, he doesn't notice the curvature of the land and avoids the black pentagons because he suspects they may be bottomless pits.

Every morning he wakes up on a white hexagon, leaves some pheromones to mark it as his special home space, and starts his random morning stroll. Every step on this stroll takes him to one of the three neighboring white hexagons with equal probability. He ends his stroll as soon as he first returns to his home space. As an example, on exactly 1/3 of mornings Andy's stroll is 2 steps long, as he randomly visits one of the three neighbors, and then has a 1/3 probability of returning immediately to the home hexagon.

This morning, his soccer ball bounced through a kitchen with an infinite (at least practically speaking...) regular hexagonal floor tiling consisting of black and white hexagons, a small part of which is shown above on the right. In this tiling every white hexagon is surrounded by alternating black and white hexagons, and black hexagons are surrounded by six white hexagons. Andy fell off the ball and woke up on a white hexagon. He didn't notice any change in his surroundings, and goes about his normal morning routine.

Let p be the probability that his morning stroll on this new land is strictly more steps than the expected number of steps his strolls on the soccer ball took. Find p, rounded to seven significant digits.

My Solution

Let $I_{football}$ be a discrete random variable representing the number of steps on Andy's walk each morning on the football. Let I_{floor} be another discrete random variable representing the number of steps on Andy's walk on the floor. The question essentially asks:

$$P(I_{floor} > E[I_{football}])$$

So a natural first step in solving this problem would be to find $E[I_{football}]$.

Using Graphs

Recognize very early on that one can conceptualize the football pattern above as a graph by representing the white hexagon with nodes, and the edges that connect them as... well, edges. In doing so, we ignore the black pentagons and achieve a result that can be graphically represented with something similar to:

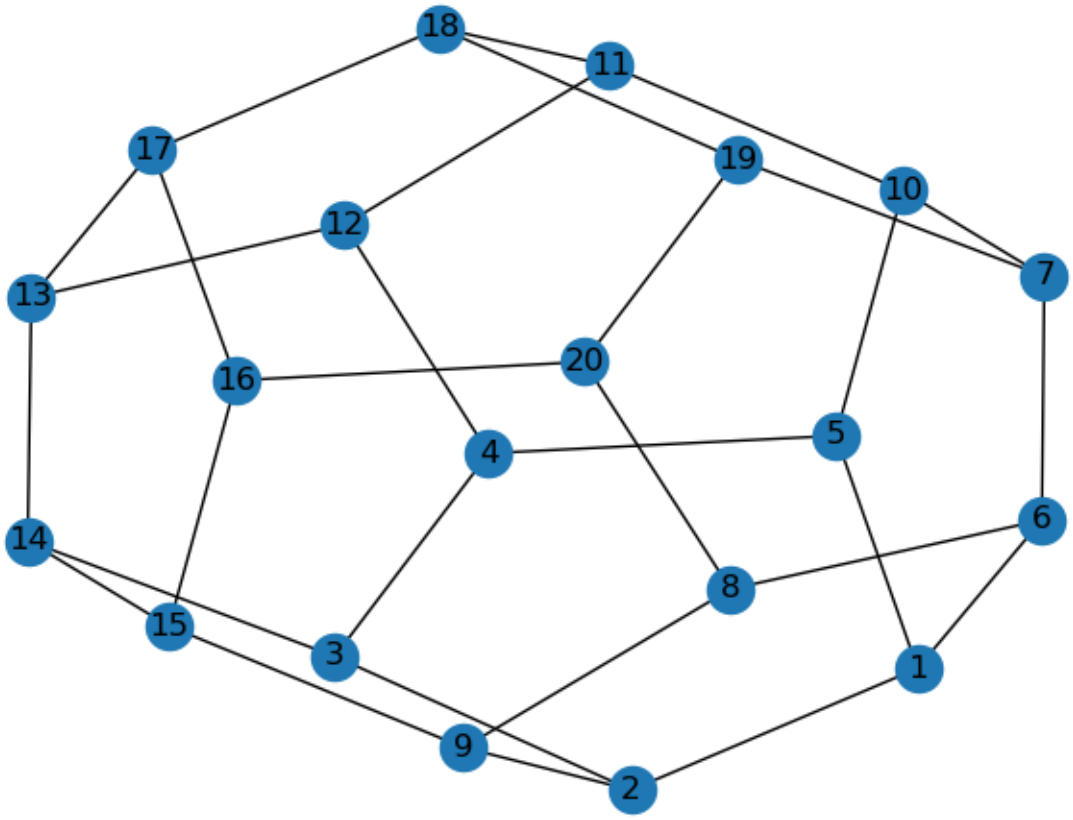


Figure 1: *Graph representation of the football. It can also be more intuitively drawn as a pentagon, surounded by a upright pentagon, which itself is surounded by an even larger pentagon.*

Thinking of the pattern as a graph such has two benefits - firstly, we now have a concrete mathematical conception which we can apply functions (through code) towards. Secondly, we can further simplify the problem. Simplifying the problem itself has two benefits: it can reduce the amount of computational resources needed to run simulations and it opens up new avenues by which we can reach our answer of $E[I_{football}]$.

Reduction to Markov Chain

Notice that the graph above is an undirected graph. Notice also, that each node is connected to 3 others and that you can pick whichever starting node and every single other node would be connected to it in 5 steps or less. Let us designate nodes which are immediately connected to the origin node "H" as "1st neighbours". There will thus be "2nd neighbours", "3rd neighbours", "4th neighbours", and "5th neighbours" (only one of each for each origin node).

At the start of Andy's random walk, he will definitely take at least one step onto a "1st neighbour" node. There will then be a $\frac{1}{3}$ chance he steps back to "H" and a $\frac{2}{3}$ chance he will step onto a "2nd neighbour" node. This goes for every single "H" and every single following "1st neighbour" node on the graph. Using this method, and extending it to the "3rd, 4th, and 5th neighbours", we can create both a tree diagram or a Markov Chain that can be represented as such:

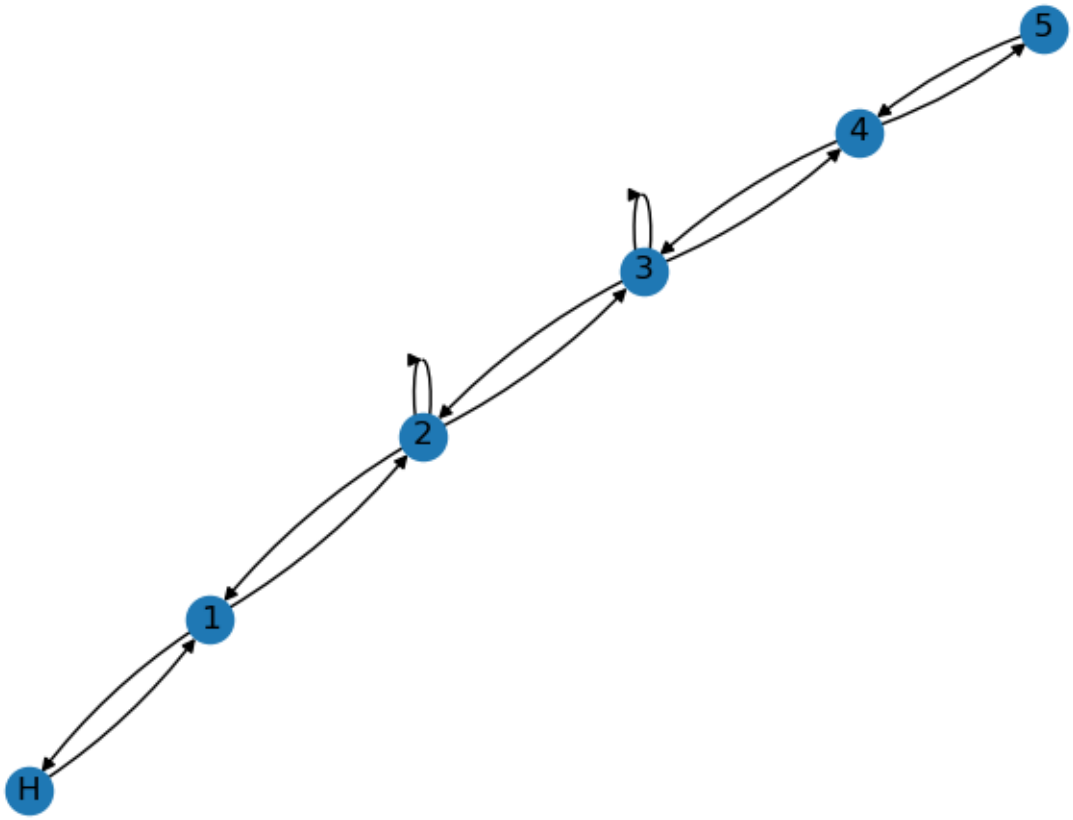


Figure 2: *Markov Chain representation of the football. Integers now refer to the nth neighbour of the origin node. (Note that the probablities of reaching each subsequent node from a preceding node aren't labelled here.)*

Notice that sometimes, you can move from an "ith neighbour" to another "ith neighbour" node so the Markov Chain has self loops. This Markov Chain can be represented using a probability transition matrix. In this case, the matrix looks something likes this:

	H	1	2	3	4	5
H	0	1	0	0	0	0
1	1/3	0	2/3	0	0	0
2	0	1/3	1/3	1/3	0	0
3	0	0	1/3	1/3	1/3	0
4	0	0	0	2/3	0	1/3
5	0	0	0	0	1	0

We can now calculate $E[I_{football}]$ using the probability transition matrix and a technique known as "first-step analysis". Because our Markov Chain represents all the nodes/hexagons on our original football and the probability of reaching them through the transition matrix, working on the Markov Chain is as good as working on the original, more complex graph.

Let μ_{iH} be the expected number of steps to reach node "H" from the node "i". Our random walk begins on "H" but we have to take at least one step off onto a "1st neighbour" node so

$$\mu_{HH} = 1 + \mu_{1H}$$

Because "1st neighbour" nodes are only one step away from "H", we can calculate μ_{1H} based on the expected number of steps to reach "H" from the other nodes and from the probability of reaching those other nodes from "1st neibhgour" nodes. This is because of the law of total expectation. This means that we can generalize our μ_{iH} function as such:

$$\mu_{iH} = 1 + \sum_{l \neq H} p_{il}\mu_{lH}$$

where p_{il} is the probability of going from node i to node l, something that we can reference in our probability transition matrix. We can apply this general formula to each of the five orders of neighbour nodes and we achieve a system of equations like this:

$$\begin{aligned}
\mu_{HH} &= 1 + \mu_{1H} \\
\mu_{1H} &= 1 + \frac{2}{3}\mu_{2H} \\
\mu_{2H} &= 1 + \frac{1}{3}\mu_{1H} + \frac{1}{3}\mu_{2H} + \frac{1}{3}\mu_{3H} \\
\mu_{3H} &= 1 + \frac{1}{3}\mu_{2H} + \frac{1}{3}\mu_{3H} + \frac{1}{3}\mu_{4H} \\
\mu_{4H} &= 1 + \frac{2}{3}\mu_{3H} + \frac{1}{3}\mu_{5H} \\
\mu_{5H} &= 1 + \mu_{4H}
\end{aligned}$$

Solving this system of equations (throwback to 8th grade maths!) gives us:

$$\mu_{HH} = 20$$

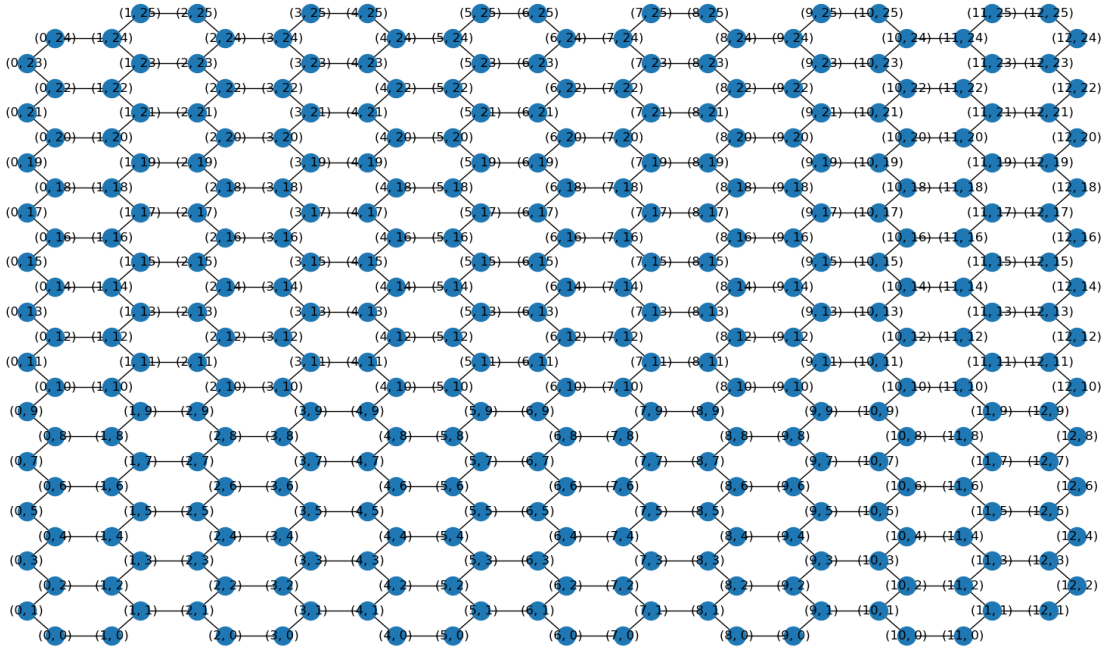
Therefore,

$$E[I_{football}] = \mu_{HH} = 20$$

and we are looking for $P(I_{floor} > 20)$

Graph Simulations

We now know more concretely what we are looking for. Now let us divert our attention to Andy's new random walk on the floor. This too, can be conceptualized with graphs, again by representing the white hexagons as nodes and their edges this time resulting in the pattern of an infinite hexagonal lattice as partially represented below.



A random walk on this floor could be infinite (as could one on the football for that matter). Rather then finding the probability distribution for each n steps from 21 to infinity, it would be a lot easier to find

$$P(I_{floor} > 20) = 1 - P(I_{floor} \leq 20)$$

This limits what we are calculating to a small range of easy-to-compute values. For each step n, we can easily calculate the probability of randomly walking such a path with exactly n steps from origin H back home by using the formula:

$$P(I_{floor} = n) = (\frac{1}{3})^n \times \text{No. of possible routes returning to origin for the first time on step n}$$

This is because each node is connected to 3 other nodes so there is a $\frac{1}{3}$ chance of choosing each node and the choice is made n times. Therefore, to compute this probability, all we need is the number of possible routes on a infinite hexagonal lattice returning to origin for the first time on step n. We can find this using computational simulation.

I'm doing most of this in Python but I'm sure most of the code I'm writing is adaptable to other languages and their respective graph theory packages. I've never worked with graphs in code before but a quick Google search revealed that `networkx` is a very popular Python package for graph handling. Installation and the setup of my various graphs proved to be a very simple affair.

`networkx` even comes with a preset function to quickly create hexagonal lattices:

```
nx.hexagonal_lattice_graph(m=12, n = 12)
```

Because the greatest n we'll have to deal with is 20, and to come back by step n the furthest node we can go from the origin is $\frac{n}{2}$ steps away, we don't have to generate an infinite graph, but only one which captures all possible nodes as far as $\frac{n}{2} + 1$ steps away from the origin.

The function to find all possible paths from origin "H" of steps n is relatively simple recursive function using inbuilt `networkx` functions:

```
#G is the networkx graph object.
#u is the origin node.
#n is the number of steps
#.neighbors() is an inbuilt networkx function which finds the neighbouring nodes to the input node.
def find_paths_1(G,u,n):
      if n==0:
        return [[u]]
    paths = [[u] + path for neighbour in G.neighbors(u) for path in find_paths_1(G,neighbour,n-1)]
    return paths
```

Afterwards you'll have to take out all the paths which prematurely returned to the origin or which didn't end at the origin:

```
def find_paths_2(paths,u):
  paths = [path for path in paths if path[-1]==u]
  paths = [path for path in paths if not u in path[1:-1]]
  return paths
```

Or, you can build these conditions within the recursive function to save yourself a bit of time. Using this method and applying a simple `len()` at the end will get you the number of possible routes on a infinite hexagonal lattice returning to origin for the first time on step n. The results (along with their calculated probabilities) up until $n = 16$ are as shown below.

Theoretically, this method is foolproof. The only problem is that practically, the complexity of the recursive functions increases exponentially with n and very soon you are computing problems that take up too much time or even exceed your computer's memory. This happened to me after $n > 16$, meaning I couldn't get $P(I_{floor} = 18)$ and $P(I_{floor} = 20)$. This was a significant problem.

Reducing Computational Complexity

There are at least two methods we can utilize to get the results for $P(I_{floor} = 18)$ and $P(I_{floor} = 20)$ without exceeding my computer's available memory. Firstly, I had noticed earlier in my exploratory analysis of the football graph that the number of possible routes returning to the origin for the first time at the nth step is equal to the number of possible routes from the origin to a "2nd neighbour" at the $n-2$th step (without returning to the origin).

Reflecting on this from a theoretical perspective, this is because for any path of length n which at the $n-2$ step ends up at a "2nd neighbour" node, the remaining 2 steps needed to return home will be the same for that particular "2nd neighbour" node. This pattern also holds for the hexagonal lattice, which we confirmed through computation. Anyway, we can use this pattern to our advantage by finding the values of $P(I_{floor} = 18)$ and $P(I_{floor} = 20)$ while only computing the paths for $n = 16$ and $n = 18$ respectively and finding the number of such paths which start from "H" and end at a "2nd neighbour". This method yielded to us the value of $P(I_{floor} = 18)$ but could not find $P(I_{floor} = 20)$ as an $n = 18$ recursive function through the hexagonal lattice graph was still too complex for my computer to handle.

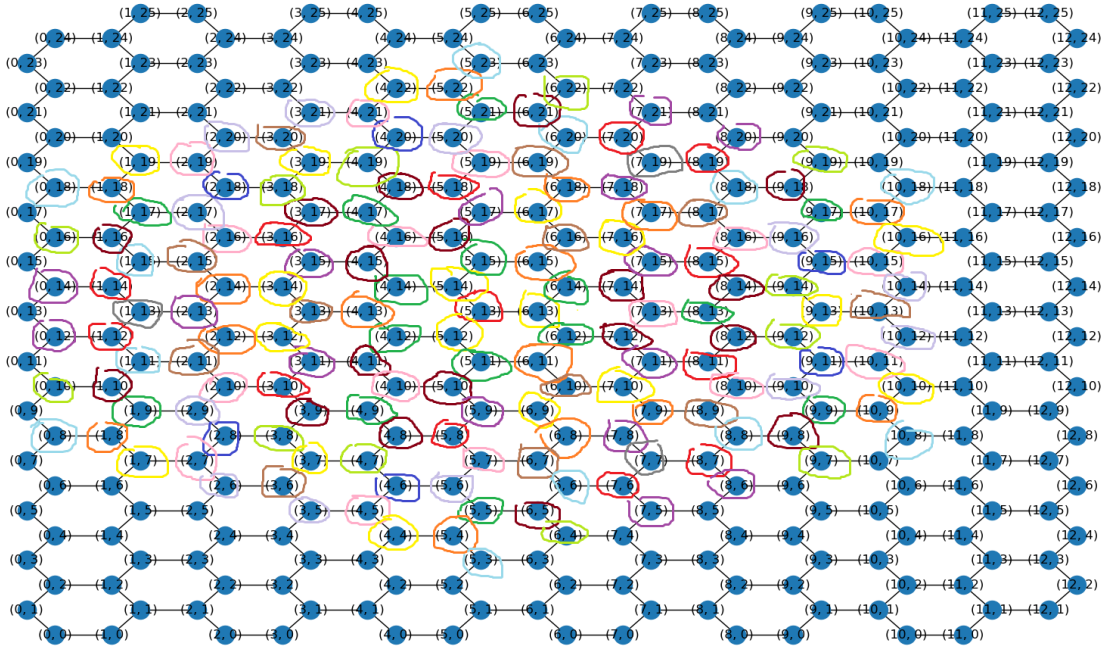
Steps	Options	Probability
1	0	0
2	3	0.333333333
3	0	0
4	6	0.074074074
5	0	0
6	30	0.041152263
7	0	0
8	180	0.027434842
9	0	0
10	1188	0.020118884
11	0	0
12	8322	0.015659311
13	0	0
14	60714	0.012693789
15	0	0
16	456174	0.010597183

Figure 3: *Number of routes returning to origin for the first time depending on number of steps. Notice that there is no way to return to the origin for the first time on an odd-numbered step. Therefore, we are only interested for even numbers of n.*

nth Step	0th	1th	2th
1	0	3	
2	3	0	6
3	0	6	6
4	6	6	24
5	6	24	48
6	24	48	144
7	48	144	360
8	144	360	1044
9	360	1044	2832
10	1044	2832	8220
11	2832	8220	23088
12	8220	23088	67116
13	23088	67116	191592
14	67116	191592	556908
15	191592	556908	1602240
16	556908	1602240	4654284

Figure 4: *Number of possible routes of nth step on the football without passing the origin that end at the origin, 1st and 2nd neighbours*

The second method I used to reduce the complexity of my computer's calculations was to reduce my hexagonal lattice to a Markov Chain using a similar technique to the one I had applied to the football graph. Unfortunately, while the theory remained the same, the complexity of the hexagonal lattice meant that I couldn't simply categorize all nodes as belonging to a "jth order of neighbour" as I had done with the football. This is because some "3rd neighbours" were connected to 2 "4th neighbours" while others were connected to only 1 "4th neighbours" and this problem cascaded throughout each level. However, we could still simplify the graph, albeit with more categories than the 11 order of neighbours that we have for each origin (an "11th order" of neighbours is needed because there needed to be an "exit" option for the "10th order neighbours" that accurately reflected the probability of the walk turning back at the 10th step). Simplifying the graph in this format created 34 distinct nodes and was a painstaking process that took me at least 2 hours. I categorized each individual node on the hexagonal lattice up until the "11th order" of neighbours by studying a picture of the hexagonal lattice and using colour-coding. The end result was this monstrosity:



Once again, reducing the hexagonal lattice to find nodes which fit into distinct "categories" and then identifying the probability relations between each category and building a Markov Chain off of it, greatly simplfied the problem. We can run our random walk simulation on this new Markov Chain and my computer arrived at the answer for $P(I_{floor} = 20)$ in a little over an hour. The only caveat is that for each path it identifies on each step, you have to multiply the connections by the number of possibilities that they represent and aggregate all of the connections. For example, if there are 2 ways a "4th order neighbour" goes to a "3rd order neighbour", every time you see a walk taking such a route you have to multiply by 2. This has to be done for each path. Then, the number of all paths for step n is added together to reach the final answer for that step n.

If we combine this method with the first one, we actually can arrive at the same result in around 5 minutes. The result is we are able to calculate the number of possible routes of n steps for each $n \leq 20$ that makes its first return to the origin at the end of the walk. Taking these numbers, feeding it into our earlier formula and aggregating the probabilities before subtracting it from 1 gives us our final answer.

Steps	Options	Probability	
1	0	0	
2	3	0.333333333	
3	0	0	
4	6	0.074074074	
5	0	0	
6	30	0.041152263	
7	0	0	
8	180	0.027434842	
9	0	0	
10	1188	0.020118884	
11	0	0	
12	8322	0.015659311	
13	0	0	
14	60714	0.012693789	
15	0	0	
16	456174	0.010597183	
17	0	0	
18	3504630	0.009046063	
19		0	
20	27398106	0.0078577	
21		0	
		0.551967444	0.4480326

$$P(I_{floor} > E[I_{football}]) = 0.4480326$$