The Maths of Snakes and Ladders

J Campbell

December 12, 2013

1 Introduction

Snakes and Ladders is an ancient Indian board game regarded today as a worldwide classic. It is played between two or more players on a gameboard having numbered, gridded squares. A number of 'ladders' and 'snakes' are pictured on the board, each connecting two specific board squares. The object of the game is to navigate one's game piece, according to die rolls, from the start (off the board) to the finish (square 100), helped or hindered by ladders and snakes respectively.[1]

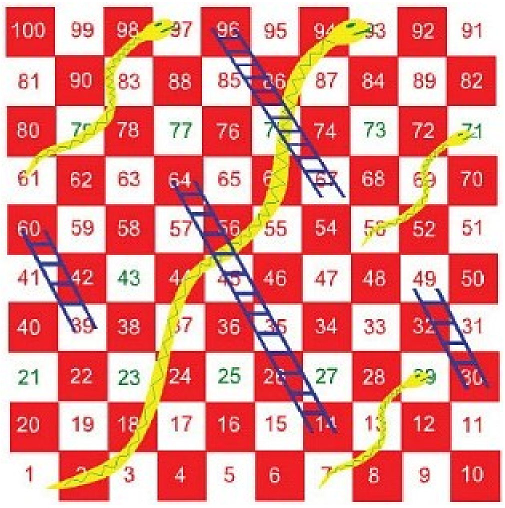


Figure 1: The board that we will be studying.

2 Markov Chains

A Markov chain is a mathematical system that undergoes changes from one state to another. It is a random process in which the next state depends only on the current state and not on the sequence of events that preceded it. This is called the Markov property. This applies to Snakes and Ladders where a state is the position of a player on the board. To get to the next state you need to roll the die again and this is called a Transition. A Transition Matrix describes the probability distribution of all transitions.

Snakes and Ladders is actually an example of an Absorbing Markov Chain. This is because it has an absorbing state, a state which once entered, cannot be left. In this case it is when a player reaches 100, the game ends.

3 The Transition Matrix

In this Transition Matrixix, the value of $M_{x,y}$ is the probability of moving from x to y. Below is the code used in sage to create the matrix for Figure 1.

```
a = matrix(QQ, 101) # creates a square matrix populated with 0's
SaLdic = {29: 7, # a dictionary of all the snakes and ladders
          71: 53,
          93: 1,
          97: 61,
          14: 64,
          30: 49,
          39: 60,
          67: 96,
          }

def rolldie(N):
    """
    Simulates the rolling of a die
    Inputs: an integer N, the index of the row in the matrix
    Outputs: alters the matrix a to reflect the outcome of the die
    """
    for i in range(1,7):
        if N + i <= 100: # player must land on '100' exactly
            a[N, N + i] = 1/6
        else: # otherwise, player does not move
            a[N, N] += 1/6

for k in range(100): # does rolldie over the entire matrix except for [100,100].
    if k not in SaLdic: # except for snakes and ladders
        rolldie(k)

for p, q in SaLdic.iteritems(): # changes probabilities of moving along snake/ladder
    a[p, q] = 1                 # to 1 because they are certain.
```

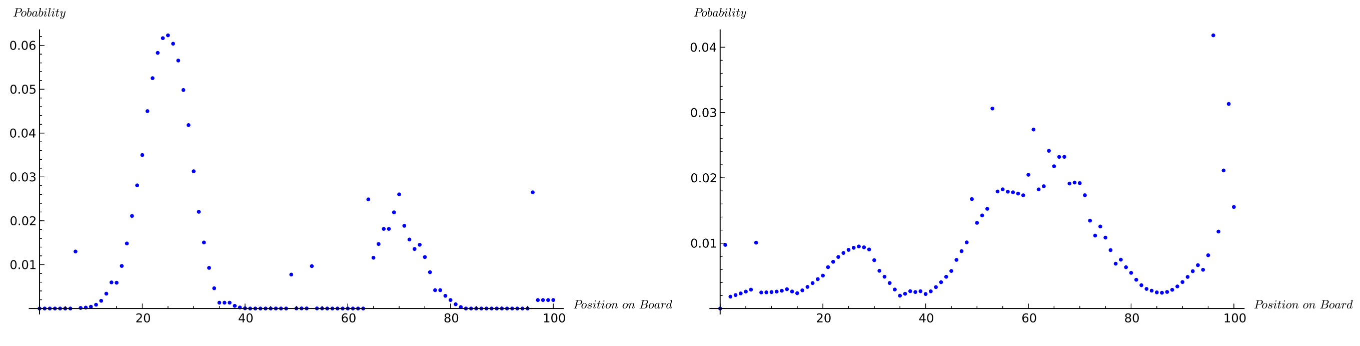
Then we can create a vector to represent the initial state. It begins with just 1 in Row_0 because the probability of being off the board at the start of the game is certain. By multiplying this vector by the transition matrix, it gets populated by more and more probabilities.

4 Limitations

There is one big assumption made in this model though. It assumes that moving up/down a ladder/snake counts as a roll of the die instead of being instantaneous. This is not how the game is normally played but the overall effect is small.

However, this does make a large difference in computational speed. Matrix multiplication requires n^3 calculations and this particular board has a total of 8 ladders and snakes. If their corresponding rows and columns were removed from the transition matrix it would reduce the number of calculations required by 21.9%.

```
x = 101^3
y = (101-8)^3
(x - y)/float(x)*100
21.92990203833637
```



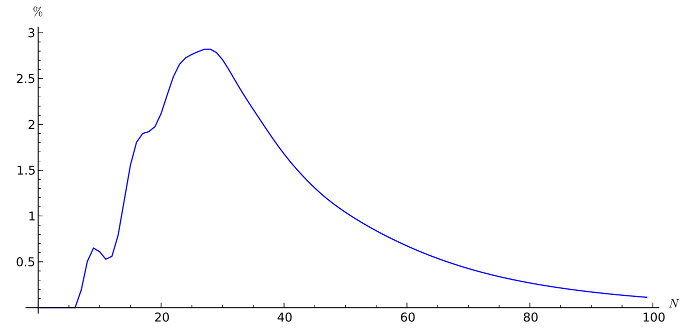
(a) 7 rolls

(b) 15 rolls.

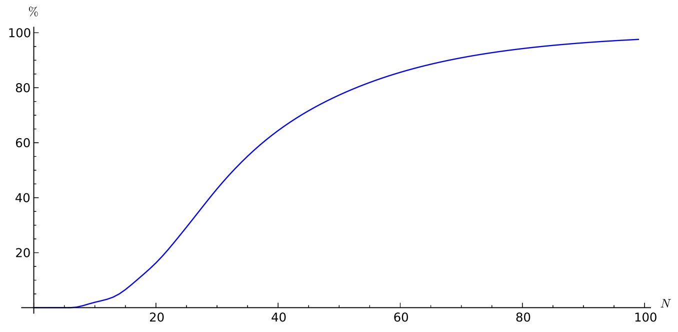
Figure 2: The Probability Distributions after different numbers of rolls.

5 The Results

In Figure 2 we can see the effect the snakes and ladders have on the probability distributions. We end up with several peaks in our graph due to players skipping large areas of the board with a ladder and then repeating areas due to a snake. Figure 2a shows us the first time a player might reach square 100. This is after only 7 rolls of the die. Figure 2b shows the probability distribution 8 rolls later and as you would expect it has a much more even spread. An interesting point is that square 1 has a relatively large probability, this is due to the long snake running across the board.



(a) The Percentage of games that end on turn N.



(b) The Percentage of games that end by turn N.

Figure 3: Behaviour compared to N rolls.

Figure 3 is produced by repeatedly mutlipying by the Transition Matrix (rolling the die), taking the value at Row_{100} and then appending it to a list. Figure 3a shows us that the modal value of N is 28, ie. more games will be completed in 28 rolls than any other number of rolls. Figure 3b is the same process done cumulatively. It tells us that the median is 33, ie. half of all games are completed in less than 33 rolls and half of all games are completed in 33 or more rolls.

6 The Future

If this project were to continue there are several new things could be done:

- Improve the Transition Matrix so that it gave a more accurate representation of the game and therefore reduced calculation times.
- Create a function that generated different game boards with randomly placed snakes and ladders.
- Using a large set of random boards, look for patterns between game length and number of snakes/ladders. Also look at the properties of a more/less frequently used snake/ladder.
- Using the points previously mentioned, create a function that when given specific parameters (size of board, game length, number of snakes/ladders,...) will produce a random board with those properties.

References

[1] Wikipedia. Snakes and ladders — wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Snakes_and_Ladders&oldid=581047421`, 2013. [Online; accessed 9-December-2013].