# Modeling and Analysis of Board Games

Vishva Sundarapandian Raani
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands.
v.sundarapandianraani@student.utwente.nl
Supervised by- Mr. Matthias Volk
and Dipl. math. Christina Kolb.

## ABSTRACT

Playing board games is always entertaining. Since most board games are played with more than one player, they make individuals come closer, relieve stress and also improve cognitive brain activity [10]. As most board games involve a card, a die or any similar playing tool, which is dictated by luck, it lends to the question, to what degree, and how much of the game is influenced by this uncertainty. Interesting questions arise like, would a game run infinitely? Will the starting player have an edge over the other players? This research project aims to model and analyse board games with probabilistic model checking and answer those questions for the games- Snake and Ladders and the Game of the Goose.

## Keywords

Probabilistic model, Modeling, Board games, Snake and ladders, Game of the Goose, Probabilities, Model checking, Prism.

## 1. INTRODUCTION

Games and puzzles are mental exercises which train and improve the planning capabilities of the human brain [13]. Moreover, winning a game or solving a puzzle triggers a reward in the neural system, thus giving one a sense of joy. Many board games require a certain degree of both skill and luck. Even in board games such as Chess, people could think it is purely skill based, as there are no outside factors like a card shuffle or a die involved. But there still exists a very slight element of luck, which could be the edge of the white player's first move or the contribution of human emotions of the players during the game [12]. So, though board games like Chess involve predominantly skill and very little luck, on the other hand, board games like Snake and Ladders, Game of the goose, involve almost zero skill or strategy and are mainly based on luck. This research is focused on the analysis of board games involving very less to no skills and strategy, starting with Snake and ladders and then the Game of the Goose. For those who are not familiar with the games, a short description of both the games- Snake and Ladders and The Game of the Goose can be found below at Section 3.1 and 4.1.

### 1.1 Model Checking

Model checking is a technique for the verification of properties that are enclosed in a system, through the exploration of its states and their probabilities [4]. Model checking is the fundamental phenomenon used in this research to analyse board games, allowing one to explore the amusing aspects of them and come up with interesting observations. Though some of the aspects can be answered by simply applying basic probabilities, by model checking them, one can get a much deeper understanding about the game itself and might possibly help the player to use the aspects of the game to his/her advantage.

The DTMC- discrete-time Markov chain is the structure used here to model the board games. The Markov chain is a model representing a series of events in which the probability of each event depends only on the state of the previous event [11]. The DTMC(discrete-time Markov chain) as the name implies, is a Markov chain on a measurable or a countable state space [11]. This acts as a good choice to model the chosen board games because the games are based on the same fundamental idea as well, where the probability of moving to the next state depends only on the current state, and not on any states of the past.

Though similarly motivated studies have been performed in the past, that can be found in Section 2, in this research, a different approach will be taken in analysing the board games by model checking them through Prism files. The approach involves modelling the game first as a discrete-time Markov chain within a model checking software tool called Prism. Prism is a probabilistic model checker used to model, analyse and simulate systems that have probabilistic behaviour [8]. The research then utilises Prism's property specifications feature to analyse the constructed board game model. The property specification is a tool integrated within Prism which is used to analyse a probabilistic model [1]. Once the model is analysed using the framed properties, the research questions will be answered using the findings and possibly come up with quite surprising conclusions that might change certain views on board games.

### 1.2 Goal

The ultimate aim of this project is to model a board game as a probabilistic model and analyse it by probabilistic model checking. To achieve this, the goal has been segregated into the following parts:

**Goal 1:** Framing the research questions to analyse in relation to the chosen games. Examples could be the ones discussed earlier like how many steps on average could the game take to complete, or would it even complete etc.

**Goal 2:** To model the game as a Markov chain within Prism, with all relevant information and all necessary probabilities.

**Goal 3:** Specify properties to analyse the model, to answer the research questions framed earlier.

**Goal 4:** Use the model checking tool, Prism, for the analysis of the model made in goal 2 and the properties from goal 3.

The analysis is done on the Snake and ladders first, and then the goals 1 to 4 will be repeated for the Game of the Goose.


*28th Twente Student Conference on IT* Febr. 2nd, 2018, Enschede, The Netherlands.

## 1.3 Research question

The research questions that would be analysed on the Snake and ladder probabilistic model are:

1. How large are the probabilities to reach particular fields on the board ? Do some fields have a bigger probability than others?

2. Can the game end within 30 moves? What is the average number of moves taken to complete the game?

3. Does player 1 have an edge over other players by starting first?

The Game of the Goose being a relatively more complex game with more rules than Snake and ladders, more questions can be analysed:

4. What is the probability of the game ending up being a draw?

5. Can the game be running infinitely long? What will the average number of moves be to complete the game?

6. What is the probability of a player dying in the game and having to restart? (i.e. a player is considered dead when landing on the 58th field and has to restart the game again from field zero.)

# 2. RELATED WORK

In this section, some of the previous research done related to probabilities in board games are portrayed. Though there has been quite a significant amount of research done on strategy based olden board games such as chess, over the past decades, very less research has been done on non-skill, luck based games such as Snake and ladders [6].

In 1997, Truman Collins structured a paper Probabilities in the Game of Monopoly which explains the difference in probabilities within different elements of the game Monopoly [5]. It discusses how the long term probability of reaching each square is very different from one another, the expected incomes per roll and a lot of other elements. Through these data, the author also concludes a set of things for players to play monopoly in a better way. Examples of those conclusions include, railroads being a better investment in general than the others, the best return over investment was found to be buying a third house at New York Avenue etc.

A research paper was framed in 2014, by J. F. Groote et al. on the Game of Goose, analysing the probabilities of different players winning the game based on the order of their start [7]. The research explains how always the starting player has the highest probability to win the game, no matter how many players were involved. The observed trend always stayed consistent too, where the first player to begin the game always had the highest probability and the last player to begin had the lowest probability of winning. The paper also discussed the case where the game could end up in a draw. According to the findings of the researchers, the game could end in a draw with a probability of 0.226.

Further, one research article in specific, namely 'How long is a game of snakes and ladders', from the book The Mathematical Gazette, discusses in detail how many moves it would take on average to complete a snake and ladders game [2]. The journal takes the Milton Bradley version of Snake and ladders, which has 100 squares, with 19 snakes and ladders and concludes that it will take 39.2 moves on average for one player to complete the game. The article also has explained how adding ladders and snakes on various sizes influences the number of moves to finish that variant of the Snake and ladders game.

In 2015, a research article was published by P. Milazzo et al. where probabilistic model checking has been done on a few board games [9]. The aim of the study is to show the usefulness of model checking in game design. The study has involved three different case studies dealing with, a non-strategy game, a strategy game, and a collaborative game and model checking them all. The research has successfully showed that on all the three selected case studies, the probabilistic and statistical model checking can have a role during game design, thus profiting the development of future games.

# 3. SNAKE AND LADDERS MODEL CHECKING

## 3.1 Game description

Snake and Ladders, being an ancient Indian board game, is a simple race between two or more players based on sheer luck, dictated by the roll of a dice by each player to make their moves. The objective of the game is to reach the finish from the start, being helped by the ladders and hindered by the snakes [14].

In principle, the game has no upper bound on the number of moves to be played and thus could go on infinitely. However, the probability of such long duration games rapidly asymptotes to insignificance. In an analysis done by an organisation called DataGentics, where a billion games of Snake and ladders were played, the longest recorded game took only 394 moves to complete [3]. Another observation is that the probability of the moves on the board by the dice is fixed and independent from the previous moves that already happened. Thus the game itself can be represented as a Markov chain, making it an ideal candidate for Markov chain analysis [2].

## 3.2 Model Design

As mentioned earlier, the Snake and Ladders game has been modelled in the Prism. The Snake and Ladders board that is used here is a smaller version of a board containing 20 states with 2 ladders and 3 snakes. The ladders running from state 2 to 5, 4 to 11 and the snakes running from 8 to 5, 18 to 3 and 19 to 9. The prism code used to model the game is provided below.

```
dtmc
  // board fields
  global f1 : [1..20] init 1;
  global f2 : [1..20] init 1;
  //Ladders - 2 to 5, 4 to 11
  //Snakes - 8 to 5, 18 to 3, 19 to 9;

  // winner of the game
  global w : [0..2] init 0;
  const int w1 = 1;
  const int w2 = 2;
  //which player's chance to play
  global chance : [1..2] init 1;
  const int c1 = 1;
  const int c2 = 2;
  //total number of moves
  global m : [0..1000] init 0;
    rewards "steps"
  (f1!=2 & f1!=4 & f1!=8 & f1!=18 & f1!=19 & f1!=20) &
     (f2!=2 & f2!=4 & f2!=8 & f2!=18 & f2!=19 & f2!=20)
     : 1;
    endrewards

module player1
  [] f1=1  & chance=c1 & w=0 & m<1000-> 1/6: (f1'=2) +
     1/6: (f1'=3)&(chance'=c2)&(m'=m+1) + 1/6: (f1'=4)
      + 1/6: (f1'=5)&(chance'=c2)&(m'=m+1) + 1/6: (f1
     '=6)&(chance'=c2)&(m'=m+1) + 1/6: (f1'=7)&(chance
     '=c2)&(m'=m+1);
  [] f1=2  & chance=c1 & w=0 & m<1000-> (f1'=5)&(chance
     '=c2)&(m'=m+1); //--- Ladder 2 to 5
  [] f1=3  & chance=c1 & w=0 & m<1000-> 1/6: (f1'=4) +
     1/6: (f1'=5)&(chance'=c2)&(m'=m+1)  + 1/6: (f1'=6)
     &(chance'=c2)&(m'=m+1)
      + 1/6: (f1'=7)&(chance'=c2)&(m'=m+1)  + 1/6: (f1
     '=8)  + 1/6: (f1'=9)&(chance'=c2)&(m'=m+1);
  [] f1=4  & chance=c1 & w=0 & m<1000-> (f1'=11)&(
     chance'=c2)&(m'=m+1); //--- Ladder 5 to 11
  [] f1=5  & chance=c1 & w=0 & m<1000-> 1/6: (f1'=6)&(
     chance'=c2)&(m'=m+1)  + 1/6: (f1'=7)&(chance'=c2)
     &(m'=m+1)  + 1/6: (f1'=8)
      + 1/6: (f1'=9)&(chance'=c2)&(m'=m+1)  + 1/6: (f1
     '=10)&(chance'=c2)&(m'=m+1) + 1/6: (f1'=11)&(
     chance'=c2)&(m'=m+1);
```

```
  [] f1=6  & chance=c1 & w=0 & m<1000-> 1/6: (f1'=7)&(
     chance'=c2)&(m'=m+1)  + 1/6: (f1'=8) + 1/6: (f1
     '=9)&(chance'=c2)&(m'=m+1)
      + 1/6: (f1'=10)&(chance'=c2)&(m'=m+1) + 1/6: (f1
     '=11)&(chance'=c2)&(m'=m+1) + 1/6: (f1'=12)&(
     chance'=c2)&(m'=m+1);
  [] f1=7  & chance=c1 & w=0 & m<1000-> 1/6: (f1'=8) +
     1/6: (f1'=9)&(chance'=c2)&(m'=m+1)  + 1/6: (f1
     '=10)&(chance'=c2)&(m'=m+1)
      + 1/6: (f1'=11)&(chance'=c2)&(m'=m+1) + 1/6: (f1
     '=12)&(chance'=c2)&(m'=m+1) + 1/6: (f1'=13)&(
     chance'=c2)&(m'=m+1);
  [] f1=8  & chance=c1 & w=0 & m<1000-> (f1'=5)&(chance
     '=c2)&(m'=m+1); //--- Snake 8 to 5
  [] f1=9  & chance=c1 & w=0 & m<1000-> 1/6: (f1'=10)&(
     chance'=c2)&(m'=m+1) + 1/6: (f1'=11)&(chance'=c2)
     &(m'=m+1) + 1/6: (f1'=12)&(chance'=c2)&(m'=m+1)
       + 1/6: (f1'=13)&(chance'=c2)&(m'=m+1) + 1/6: (f1
     '=14)&(chance'=c2)&(m'=m+1) + 1/6: (f1'=15)&(
     chance'=c2)&(m'=m+1);
  [] f1=10 & chance=c1 & w=0 & m<1000-> 1/6: (f1'=11)&(
     chance'=c2)&(m'=m+1) + 1/6: (f1'=12)&(chance'=c2)
     &(m'=m+1) + 1/6: (f1'=13)&(chance'=c2)&(m'=m+1)
       + 1/6: (f1'=14)&(chance'=c2)&(m'=m+1) + 1/6: (f1
     '=15)&(chance'=c2)&(m'=m+1) + 1/6: (f1'=16)&(
     chance'=c2)&(m'=m+1);
  [] f1=11 & chance=c1 & w=0 & m<1000-> 1/6: (f1'=12)&(
     chance'=c2)&(m'=m+1) + 1/6: (f1'=13)&(chance'=c2)
     &(m'=m+1) + 1/6: (f1'=14)&(chance'=c2)&(m'=m+1)
       + 1/6: (f1'=15)&(chance'=c2)&(m'=m+1) + 1/6: (f1
     '=16)&(chance'=c2)&(m'=m+1) + 1/6: (f1'=17)&(
     chance'=c2)&(m'=m+1);
  [] f1=12 & chance=c1 & w=0 & m<1000-> 1/6: (f1'=13)&(
     chance'=c2)&(m'=m+1) + 1/6: (f1'=14)&(chance'=c2)
     &(m'=m+1) + 1/6: (f1'=15)&(chance'=c2)&(m'=m+1)
       + 1/6: (f1'=16)&(chance'=c2)&(m'=m+1) + 1/6: (f1
     '=17)&(chance'=c2)&(m'=m+1) + 1/6: (f1'=18);
  [] f1=13 & chance=c1 & w=0 & m<1000-> 1/6: (f1'=14)&(
     chance'=c2)&(m'=m+1) + 1/6: (f1'=15)&(chance'=c2)
     &(m'=m+1)
       + 1/6: (f1'=16)&(chance'=c2)&(m'=m+1) + 1/6: (f1
     '=17)&(chance'=c2)&(m'=m+1) + 1/6: (f1'=18) + 1/6:
      (f1'=19);
  [] f1=14 & chance=c1 & w=0 & m<1000-> 1/6: (f1'=15)&(
     chance'=c2)&(m'=m+1) + 1/6: (f1'=16)&(chance'=c2)
     &(m'=m+1) + 1/6: (f1'=17)&(chance'=c2)&(m'=m+1)
      + 1/6: (f1'=18) + 1/6: (f1'=19) + 1/6: (f1'=20)&(
     chance'=c2)&(m'=m+1);
  [] f1=15 & chance=c1 & w=0 & m<1000-> 1/6: (f1'=16)&(
     chance'=c2)&(m'=m+1) + 1/6: (f1'=17)&(chance'=c2)
     &(m'=m+1) + 1/6: (f1'=18) + 1/6: (f1'=19)
      + 1/6: (f1'=20)&(chance'=c2)&(m'=m+1) + 1/6: (f1
     '=20)&(chance'=c2)&(m'=m+1);
  [] f1=16 & chance=c1 & w=0 & m<1000-> 1/6: (f1'=17)&(
     chance'=c2)&(m'=m+1) + 1/6: (f1'=18) + 1/6: (f1
     '=19) + 1/6: (f1'=20)&(chance'=c2)&(m'=m+1)
      + 1/6: (f1'=20)&(chance'=c2)&(m'=m+1) + 1/6: (f1
     '=20)&(chance'=c2)&(m'=m+1);
  [] f1=17 & chance=c1 & w=0 & m<1000-> 1/6: (f1'=18) +
      1/6: (f1'=19)&(m'=m+1) + 1/6: (f1'=20)&(chance'=
     c2)&(m'=m+1) + 1/6: (f1'=20)&(chance'=c2)&(m'=m+1)
       + 1/6: (f1'=20)&(chance'=c2)&(m'=m+1) + 1/6: (f1
     '=20)&(chance'=c2)&(m'=m+1);
  [] f1=18 & chance=c1 & w=0 & m<1000-> (f1'=3)&(chance
     '=c2)&(m'=m+1); //--- Snake 18 to 3
  [] f1=19 & chance=c1 & w=0 & m<1000-> (f1'=9)&(chance
     '=c2)&(m'=m+1); //--- Snake 19 to 9
  [] f1=20 & w=0 -> (f1'=20) & (w'=w1);
endmodule

module player2 = player1 [ f1=f2, c1=c2, c2=c1, w1=w2 ]
      endmodule
```

The model consists of a series of variables and constants that represent the fields/state of the players on the board, the player's turn, who the winner of the game is and the total number of moves. Further, the model is composed of two modules called 'player1' and 'player2', representing the two players playing the game. The module contains the complete procedural state space of the player, denoting all the changes happening within the system.

Taking a deeper look into the 'player1' module, each command inside the module denotes a state of the actions of the player. Within each current state, the probability of moving to the next state would be decided by the throw of a dice, represented by the six probabilities to move forward from the state, each with a probability of 1/6. In the cases where the player encounters a ladder or a snake, the system will make the player jump states and put the player at the top edge of the ladder or the tail of the snake.

The module of 'player2' is made from utilising the module of player 1 by just swapping the variables needed as shown in the final part of the code. Proper synchronisation is something that has to be considered as well, meaning, the game proceeds with only one move by a player at a time and the other player has to wait until he completes. This is accomplished using the 'chance' variable, where, as soon as a player makes a move, the value of the chance variable is changed, thus acting as a switch to allow one player to move at a time in the synchronised order. Finally, when a player makes it to the end of the board, the player wins, the game terminates and the variable 'w' stores the winner of the game.

## 3.3 Measurement and Results

In this section, the research questions are revisited and answered by analysing the constructed model. As mentioned earlier, Prism's properties specification feature is used here to analyse the model in order to answer the research questions. Prism's property specification language includes several well known probabilistic temporal logics. PCTL(Probabilistic Computation Tree Logic) is the language used for specifying properties of discrete-time models. Thus, it is used in this research as well to analyse the DTMC models. More detailed information about Prism's property specification feature could be found in the following reference [1].

**Research Question 1**

**How large are the probabilities to reach particular fields on the board ? Do some fields have a bigger probability than others ?** In order to answer the above question, the property- P=? [ F f1=x ] can be used to analyse the model. The 'P=?' part of the statement denotes the P operator, which here will generate the probability of the event specified. The event here being 'f1=x' denotes the case where player 1 reaches the 'x' field in the board. Thus the command generates the probabilities of player 1 for reaching the desired fields in the game. The resulting probabilities of all the fields have been listed below in Table 1. Moreover, through the analysis it was found that the resulting DTMC was quite large having 564879 states and taking an average duration of 0.915 seconds to analyse.

| Field | Probability | Field | Probability |
|---|---|---|---|
| 1 | 1 | 11 | 0.549 |
| 2 | 0.166 | 12 | 0.331 |
| 3 | 0.403 | 13 | 0.323 |
| 4 | 0.232 | 14 | 0.310 |
| 5 | 0.522 | 15 | 0.365 |
| 6 | 0.329 | 16 | 0.352 |
| 7 | 0.377 | 17 | 0.359 |
| 8 | 0.241 | 18 | 0.287 |
| 9 | 0.454 | 19 | 0.234 |
| 10 | 0.309 | 20 | 0.538 |

**. Table 1- Probability of player 1 to get to a field in the game. .**

As seen from the above data, it's clear that definitely some fields have a greater probability to be visited than others. The field with the highest attainable probability by a player is 11 with a probability of 0.549. This might be the case because field 11 is the field where one of the 2 ladders lands and combined with the fact that it is located in the centre of the board. On the other hand, the field with the least attainable probability by a player is 2 with a probability of 0.166. Logically this makes sense as well, as field 2 has just one possible way to be landed on, which is rolling a 1 on the die from field 1.

The experiment was repeated for player 2 using the property- P=? [ F f2=x ], which is similar to the previous property but generates all the probabilities of the fields for player 2 instead. The observed result was consistent with player 1 too, with the highest probability being for field 11 and the lowest probability for field 2. The resulting probabilities have been listed below in Table 2. Another observation is that, the probabilities computed to reach the fields are slightly higher for the starting player than the other player. Thus giving the indication that the first player indeed has a first mover advantage. A more detailed overlook on this is covered in Research question 3 below.

| Field | Probability | Field | Probability |
|---|---|---|---|
| 1 | 1 | 11 | 0.527 |
| 2 | 0.166 | 12 | 0.310 |
| 3 | 0.374 | 13 | 0.298 |
| 4 | 0.226 | 14 | 0.288 |
| 5 | 0.512 | 15 | 0.331 |
| 6 | 0.320 | 16 | 0.317 |
| 7 | 0.367 | 17 | 0.321 |
| 8 | 0.231 | 18 | 0.253 |
| 9 | 0.420 | 19 | 0.206 |
| 10 | 0.292 | 20 | 0.462 |

**. Table 2- Probability of player 2 to get to a field in the game**

**Research Question 2**

**Does the game end within 30 moves? What will the average number of moves be to complete the game?** When the property P=? [F (w=1 | w=2) m<30] is model checked, the obtained probability was 0.9772, taking a duration of 0.601 seconds to analyse. The above property generates the probability of the game finishing within 30 moves(either player 1 wins or player 2 wins). Thus, with the probability of 0.9772, the game finishing within 30 moves is very likely to happen. The obtained result is not quite surprising as being a smaller version of the board, with 20 states, it is expected to complete within a few moves.

In order to compute the average number of moves to finish the game, the reward based properties will be utilised. The tool can analyse properties which relate to the expected values of the rewards. This is achieved using the R operator, which works similar to the P operator, but returns the computed rewards instead of the probability of the given event. A more detailed overview about Prism's reward based properties could be found in the following reference [1]. When the property- R"steps"=? [ F w=1|w=2 ] is used, the system generates the expected reward of the model, which in this case is the total number of moves taken to complete the game (R "steps"). When tested, the average number of total moves was found to be 8.26. As mentioned earlier, the Milton Bradley version of Snake and ladders, which has 100 fields, with 19 snakes and ladders took an average of 39.2 moves to complete the game[2]. The obtained result is on expected lines due to the relatively smaller board.

**Research Question 3**

**Does player 1 have an edge over other players by starting first?** It was found in Research Question 1 that a consistent trend was observed where player 1 had slightly higher probabilities to reach the fields in the board than player 2. In order to check how this affects the outcome of the game, the model is checked using the property- P=? [ F w=1]. The property generates the computed probability of w=1 happening, which denotes player 1 winning the game. The obtained result was 0.538, taking a duration of 0.513 seconds to analyse.

When the property was repeated for player 2 using P=? [ F w=2], which generates the probability of player 2 winning the game, the result was 0.462. Thus, based on the obtained results, it is clear that the first player does have a significant advantage over the other player. In a board game like Snake and ladders it is quite expected to achieve the result where the player 1 does have an edge over the other players due to the first mover advantage. When more players were added to the game, the trend continued too, where still the first player had a significant winning advantage over the other players. A combined overlook on all the probabilities of the players winning in a 'n' player game is listed below in Table 3.

| Number of players | Player 1 winning | Player 2 winning | player 3 winning |
|---|---|---|---|
| 1 | 1 | - | - |
| 2 | 0.538 | 0.462 | - |
| 3 | 0.382 | 0.331 | 0.287 |

**. Table 3- Probabilities of the different players winning in a 'n' player game.**

# 4. GAME OF THE GOOSE MODEL CHECKING

## 4.1 Game description

The Game of the Goose is a traditional European board game, involving multiple players racing against each other to the end by rolling a couple of die. The game consists of 64 fields (0 to 63) and the players will be encountered with various obstacles along the way in order to complete the game. In the fields 5, 9, 14, 18, 23, 27, 32, 36, 41, 45, 50, 54, and 59 there is a goose, such that, if a player arrives at such a position, the player will move the same count of the roll again in the same direction. If this is again a position with a goose, this will be repeated. In fields 31 and 52, there will be a well and a prison where, when landed on, the player will be stuck there until being released by the other player. In field 58 there will be a death obstacle, where when landed on, the player will be dead in the game and have to restart from field zero. Other obstacles present throughout the game that have been implemented in this research, such as the bridge, maze, etc, can be found more in detail in the following research paper [7].

Similar to the snake and ladders, the game of the goose, does not have any upper bound limit to the number of moves a player can take, and thus the game can go infinitely until a player eventually reaches the last field 63 and completes the game. Similar to the Snake and ladders, the probability of the moves on the board by the die are fixed and independent from the previous moves that already happened. Thus the game can be represented as a Markov chain, making it ideal for Markov chain analysis [2].

As the two player game will be used in this research, there will be 3 possible ways the game can end. The first case where player 1 finishes by reaching the field 63 (final state), or the other case where player 2 reaches first and he/she wins, or the final case where no one wins and the game ends up in a draw. This final scenario takes place when both the players end up being stuck in the well and the prison. This makes this game much different from the Snake and ladders game analysed earlier and might end up having quite unexpected results.

## 4.2 Model Design

In this research, the single dice variant of the Game of the Goose will be used. Instead of the players rolling 2 dice at their turns, it has been reduced to just 1 die, also making it easier to model check, but preserving the important aspects of the game obstacles within the gameplay. Since the code of the Game of the goose model is too large and would not be much relevant to discuss in this research, just the important and relevant snippets have been attached and discussed below.

```
// board fields
  global f1 : [0..63] init 0;
  global f2 : [0..63] init 0;
```

```
// winner of the game
global w : [0..2] init 0;
const int w1 = 1;
const int w2 = 2;
// which player's chance to play
global chance : [1..2] init 1;
const int c1 = 1;
const int c2 = 2;

global d1 : [1..6] init 1;      //value of the dice
   roll
global d2 : [1..6] init 1;
global inv : [0..1000] init 0; // rejected invalid
   moves as other player has occupied the state
global temp1 : [0..63] init 0; //temp value to store
   previous state to go back
global temp2 : [0..63] init 0;
global check1 : [0..1] init 1; // check flags to
   verify the move
global check2 : [0..1] init 1;
global trapped1 :[0..63] init 0; // trap variables to
    lock and unlock a player
global trapped2 :[0..63] init 0;
```

The Game of the Goose board contains 64 fields, indexed from 0 to 63. Similar to the Snake and ladders model, the key variables that make up the structure of the game are- the fields/states in the board, the winner of the game and the chance switch. An overview of the the variables can be found in the above code snippet. Moreover, the Game of the Goose model required some additional variables that hold the other properties of the game such as the variables denoting if the players are being trapped(and at which state), the number of invalid moves being made, etc. The code snippet provided above contains all the variable declarations of the designed model.

The model comprises two modules called 'player1' and 'player2', representing the two players playing the game. 'player1' module contains the full state space of the player in sequence, denoting every move happening within the system. Likewise, the module of 'player2' is made by just utilising the module of player 1 by just swapping the variables as done for Snake and Ladders.

Taking a closer look into the commands of the module, the probability of moving to the next state would be decided by the throw of a die, which has been represented by the six probabilities moving forward in the board with a probability of 1/6 each. An example of the commands for the first 3 states of the board is given below.

```
[] f1=0 & chance=c1  & w=0-> 1/6:(temp1'=f1)&(f1'=1)&(
     d1'=1)&(check1'=0) +
    1/6:(temp1'=f1)&(f1'=2)&(d1'=2)&(check1'=0) + 1/6:(
     temp1'=f1)&(f1'=3)&(d1'=3)&(check1'=0) + 1/6:(
     temp1'=f1)&(d1'=4)&(f1'=4)&(check1'=0) + 1/6:(
     temp1'=f1)&(d1'=5)&(f1'=5)&(check1'=0) + 1/6:(
     temp1'=f1)&(f1'=6)&(d1'=6)&(check1'=0);
[] f1=1 & chance=c1  & w=0 & check1=1-> 1/6:(temp1'=f1)
     &(f1'=2)&(d1'=1)&(check1'=0) +
    1/6:(temp1'=f1)&(f1'=3)&(d1'=2)&(check1'=0) + 1/6:(
     temp1'=f1)&(d1'=3)&(f1'=4)&(check1'=0) + 1/6:(
     temp1'=f1)&(d1'=4)&(f1'=5)&(check1'=0) + 1/6: (
     temp1'=f1)&(d1'=5)&(f1'=6)&(check1'=0) + 1/6:(
     temp1'=f1)&(f1'=7)&(d1'=6)&(check1'=0);
[] f1=2 & chance=c1  & w=0 & check1=1-> 1/6:(temp1'=f1)
     &(f1'=3)&(d1'=1)&(check1'=0) +
    1/6:(temp1'=f1)&(f1'=4)&(d1'=2)&(check1'=0) + 1/6:(
     temp1'=f1)&(d1'=3)&(f1'=5)&(check1'=0) + 1/6:(
     temp1'=f1)&(d1'=4)&(f1'=6)&(check1'=0) + 1/6: (
     temp1'=f1)&(d1'=5)&(f1'=7)&(check1'=0) + 1/6:(
     temp1'=f1)&(f1'=8)&(d1'=6)&(check1'=0);
```

The variable 'temp1' from the above code snippet stores the field value of the current state, which will be later required to traverse back, when the player makes an invalid move (i.e. the move of a player is considered invalid, and will be traversed back to the previous existing state, if the resulting state is already occupied by another player). The variable 'd1' shown in the code stores the value of the die roll, as it would later be used in cases where the player encounters a goose.

When a player lands on a goose, the player will move the same count of the roll, again in the same direction. This has been implemented using the below statement in the code snippet, which just updates the next state by adding the same count of the dice roll.

```
[] f1=5 & chance=c1 & w=0 & temp1<f1 & g1<100-> (f1'=f1
     +d1)&(g1'=g1+1);  //Goose
```

Proper synchronisation is something that has to be strongly considered as well, meaning, the game proceeds with only one move by a player at a time and the other player has to wait until he/she completes, unless the other player is stuck (i.e. in a prison for example). Similar to the Snake and ladders, this is accomplished using the 'chance' variable, which denotes which player's move it is and thus allowing only one player to move at a time in the right order. Moreover, in the Game of the Goose model, something that had to be taken into account is, unlike Snake and ladders, the next state of the player actually depends on the current state of both the players. This is due to the fact that the player's move would be considered invalid if the other player is already present in the spot. This has been tackled using the 'check1' flags, by not allowing the players to overlap in the same spot. In the potential case of an overlap, the player will be traversed back into the previous position. The implementation has been provided as a code snippet below.

```
[] f1=f2 & f1!=0 & chance=c1 & w=0 & trapped1=0-> 1:(f1
     '=temp1)&(chance'=c2)&(check1'=1);
[] f1!=f2 & f1!=0 & chance=c1 & w=0 & trapped1=0-> 1:(
     chance'=c2)&(check1'=1);
```

When the player encounters special cases such as landing in the prison, the player must have to be locked in the same field and have to skip his/her turns until another player lands in the same spot and releases him/her. The snippet responsible for the process has been given below, where the system will make the player trapped in the state and stop the player from further participating in the game until another player lands in.

```
[] f1=52 & chance=c1 & w=0-> (trapped1'=52); //prison
[] trapped1=52 & trapped2=0 & w=0-> (chance'=c2); //
     skip turn and wait
[] trapped1=52 & trapped2=1 & w=52-> 1/6:(trapped2'=0)
     &(f2'=53)
    +1/6:(trapped2'=0)&(f2'=54) +1/6:(trapped2'=0)&(f2
     '=55)
    +1/6:(trapped2'=0)&(f2'=56) +1/6:(trapped2'=0)&(f2
     '=57)
    + 1/6:(trapped2'=0)&(f2'=58); //releasing the other
      player
```

Finally, when a player makes it to the end, the player wins, the game terminates with the variable 'w' storing the winner of the game, or if both players end up being trapped, the game terminates with a draw.

## 4.3 Measurement and Results

In this section, the research questions are revisited and answered by analysing the constructed model. Similarly, Prism properties specification is used to analyse the model and answer the research questions. PCTL(Probabilistic Computation Tree Logic) is the language used here as well for specifying properties of the constructed DTMC models.

**Research Question 4**

**What is the probability of the game ending up being a draw?** The game can result in a draw only when two players play the game and both of them get trapped in the well and the prison. Though this seems very unlikely to happen, the resulting probability is quite surprising. The following property- P=? [ F

trapped!=0 trapped2!=0 trapped1!=trapped2] generates the probability of the required case where both the players get trapped in the prison and the well.

When the game is model checked with the property, the obtained probability was 0.162, with a duration of 2.43 seconds to analyse. Though the probability seems higher, what's more interesting is, in the research paper [7] published by J. F. Groote et al, their computed probability of the 2 dice variant Game of the Goose, ending up in a draw using linear equations was 0.226, which is a lot higher than 0.162. Thus, also giving the indication that there seems to be a much higher probability that the game ends in a draw in the 2 dice variant, than the analysed single die variant of the game.

**Research Question 5**

**Can the game be running infinitely long? What will the average number of moves be to complete the game?**

It seems to be theoretically possible for a game to run infinitely long without concluding to any of those 3 scenarios discussed earlier, but it is very highly unlikely to happen. This was tested by the property P=? [F w=1 | w=2 | (trapped!=0 trapped2!=0 trapped1!=trapped2)] which generates the combined probability of the game concluding in either of the 3 scenarios- played 1 winning or player 2 winning or a draw where both players are trapped.

When the property was tested, quite a surprising result was obtained again, where the probability was 1, taking a duration of 1.972 seconds to analyse. Meaning that, the game will definitely conclude with either of the above 3 scenarios happening and that the game will not be likely to run infinitely long.

In addition, the created model was model checked in Prism for the average number of moves required to complete the game. Similar to Snake and Ladders, this has been checked through the reward based properties with the help of the R operator. When the property- R "steps"=? [ F w=1 | w=2 | (trapped!=0 trapped2!=0 trapped1!=trapped2) ] is used, the system generates the expected reward of the model when the game ends, which in this case is the total number of moves taken in order to complete the game. The observed result was 32.87 moves. Moreover, as mentioned earlier, when DataGentics performed an analysis where a billion games of Snake and ladders were played, the longest recorded game took only 394 moves to complete [3]. Thus considering all the information, the results give the strong indication that the Game of the Goose running to infinity is not likely to happen.

**Research Question 6**

**What is the probability of a player dying in the game and having to restart?**

In the game, the player dies when he/she moves to the 58th field and must restart the game from field 0. Thus, losing all the progress made in the game thus far.

To answer this question, the property- P=? [F f1=58 | f2=58] will be used to generate the probability of either of the players dies in the game. When model checked, the resulting probability of either of the players dying in the game was 0.383, with a duration of 2.789 seconds to analyse. This is an observation which is quite surprising too as this is a pretty significant probability. Thus one can say that, there is quite a significant chance that a player dies in a game and has to restart from zero.

When the experiment was repeated and this time, tested for the individual probabilities of the players dying- P=? [F f1=58] for player 1 and P=? [F f2=58] for player 2, to analyse if there is any difference in the probabilities of dying within the players. When tested, the observed result was almost the same, each having a probability of 0.241 and 0.238. Thus giving an indication that there is not much of a difference in between the probability of the players dying in the game with respect to who starts first.

# 5. FINDINGS

To summarise, the research has analysed board games by model checking them. This has been done through Prism and its properties. Two board games Snake and Ladders and the Game of the Goose have been modelled and analysed, solving one's intriguing questions that arise while playing board games. Though some of the findings turned out to be as expected and predictable, some of them were quite surprising.

The key findings from the research are listed below:

- The probabilities to reach a particular field on the Snake and Ladders board are significantly different from one field to another and the starting player has a slightly higher probability to reach the fields than the other player.
- The probability that the Snake and Ladders game ends within 30 moves was found to be quite high, giving a strong indication that the game terminates with one player finishing the game within 30 moves. Additionally, the average moves required to complete this Snake and Ladders game was found to be 8.26.
- The starting player in the Snake and Ladders does have a significant advantage in completing the game first over the other players.
- The probability of the two player Game of the Goose ending in a draw is found to be quite significant, having a probability of 0.162. Additionally, the probability of the game ending in a draw is much higher in the 2 dice variant than the single die variant.
- The computed probability that the Game of the Goose running infinitely long is found to be zero, thus giving the indication that a game running to infinity is not likely to happen.
- With the computed probability of 0.383, there is a significant chance that in the Game of the Goose, either of the players dies and has to restart from zero.

Though some of the findings were similar to the ones been answered in previous researches, all of them have been attained here through model checking via Prism, making it distinct. Moreover, since the Game of the Goose has been analysed for the single die variant, that makes the analysis quite different from the research [7] and allowing one to compare the variants with one another. Findings such as the Game of the Goose will not likely run infinitely long, the likelihood of either one of the players dying in the game and how much of an advantage the starting player has in Snake and Ladders, were new and quite surprising to discover.

# 6. CONCLUSIONS

Playing board games has always been fun. By modeling and analysing board games, the research allows people to now deeply understand the probabilities behind them and might possibly help one to use the aspects of the game to his/her advantage. In this research, two different games Snake and Ladders and The Game of the Goose have been analysed. Since only non-strategy based games have been chosen in this research, as a future work, the development of board games that involve both luck and some strategy to it like 'The Life', 'The Monopoly' can be modelled and analysed. Thus, making it more helpful for the ones playing it by potentially allowing them to choose the best pathway in the game that has the highest likelihood of success.

## 7. ACKNOWLEDGEMENT

A sincere thanks to Mr. Matthias Volk and Dipl. math. Christina Kolb for being the supervisors for the research and for providing a guideline through out the thesis. I would also like to thank the Track chair and the University of Twente for this opportunity.

.

## 8. REFERENCES

[1] Prism manual- https://www.prismmodelchecker.org/manual/propertyspecification/introduction.

[2] S. C. Althoen, L. King, and K. Schilling. How long is a game of snakes and ladders? *The Mathematical Gazette*, 77(478):71–76, March 1993.

[3] N. Berry. Mathematical analysis of chutes and ladders. 2013. https://datagenetics.com/blog/november12011/.

[4] E. M. Clarke, O. Grumberg, and D. A. Peled. Model checking. 1999.

[5] T. Collins. Probabilities in the game of monopoly. May 1997. http://www.tkcs-collins.com/truman/monopoly/monopoly.shtml.

[6] A. de Voogt, F. Gobet, and J. Retschitzki. *Moves in mind: The psychology of board games.* Psychology Press, 2004.

[7] J. F. Groote, F. Wiedijk, and H. Zantema. A probabilistic analysis of the game of goose. *SIAM Review*, 58(1):143–155, August 2014.

[8] M. Kwiatkowska, G. Norman, and D. Parker. Prism 4.0: Verification of probabilistic real-time systems. *23rd International Conference on Computer Aided Verification*, 6806:585–591, February 2011. ISBN 978-1-119-38755-8.

[9] P. Milazzo, G. Pardini, D. Sestini, and P. Bove. Case studies of application of probabilistic and statistical model checking in game design. *ACM 4th International Workshop on Games and Software Engineering*, May 2015.

[10] M. Nakao. Special series on "effects of board games on health education and promotion. *BioPsychoSocial Medicine volume*, 13(5), February 2019.

[11] G. Paul. Markov chains: From theory to implementation and experimentation. *John Wiley  Sons*, 13:1–235, February 2017. ISBN 978-1-119-38755-8.

[12] J. Rowson. What chess can teach you about luck. *Forge*, November 2015. https://forge.medium.com/what-chess-can-teach-you-about-luck-e5b741dfe6ec.

[13] B.-H. Schlingloff. Teaching model checking via games and puzzles. *Springer International Publishing*, November 2019. https://fmfun.github.io/Papers-2019/Schlingloff.pdf.

[14] Wikipedia. Snake and ladders. https://en.m.wikipedia.org/wiki/snakes$_a$nd$_l$adders.