Markov Chain in a Dice Game

James Kobayashi
Florida State University (USA)
jak10h@my.fsu.edu

Abstract

A Markov Chain (or a Markov Process) is a stochastic process that satisfies the Markov Property on a fine or countable state space. The Markov Property refers to the property of a process that can make predictions for the future of the process based solely on its present state, not the sequence of events that preceded it. We apply the Markov decision process to a stochastic, discrete environment described by Roters (1998) as a dice game. Using the Markov process derives an optimal strategy to go about maximizing a player's score.

Introduction

The dice game Roters (1998) described a competition where a player would continuously roll a fair 6-sided die, adding the numbers rolled to the player's total score until either (1) the player rolls the number "1", which will reset the player's score to 0 and end his turn, or (2) the player decides to end his turn early by not rolling and keeping the total number of points accumulated. A player's turn will always end with some positive score if he chooses to end early or a score of 0. On the end of a player's turn, the opponent will then take his turn, and will repeat the steps of the game. The winner is decided by whoever first reaches a certain number score.

The dice game is considered a stochastic game (Shapley, 1953), such that the play proceeds by steps from position to position, according to transition probabilities controlled jointly by the two players. Each step involves making the choice of rolling the die or ending your turn. Each transition involves either the player choosing to end his turn or rolling a 1, or rolling some other number to move to the next step.

Different strategies were developed using varying approaches. Roters (1998) derives the strategy that maximizes the expected score over one turn. Roters & Haigh (2000) found the strategy that minimises the expected number of turns required to reach the target, while Crocce & Mordecki (2009) provides an algorithm to compute the optimal minimax strategy for the dice game.

Background

Stochastic Games

As introduced by Shapley (1953), a stochastic game is a dynamic game played by one or more players. Stochastic games involve probabilistic transitions played in a sequence of states, each of which are unconcerned by the previous state. On a given state, a player will choose an action which transitions the game into a new state, dependent on the previous state and the actions chosen by the player. These transitions are continued for either a finite or infinite amount of states, either which the game ends if a goal is reached or, in the case of finite states, the maximum number of states is reached.

In Roters' (1998) dice game, it is clear that with a goal of a certain number of points, we are involved in a stochastic game continued for an finite number of states until the goal is reached.

Markov Decision Process

Markov decision processes (MDPs) model contains a finite set of states s, a finite set of actions a, a reward $R_a(s,s') = \mathrm{Rr}(s_{t+1} = s' | s_t = s | a_{t=a})$ received after the transition from initial state at time s_t to the new state s_{t+1}, and lastly the probability $P_a(s,s')$ that state s leads to new state s' given action a. Crocce & Mordecki (2009) defines our game of interest as a competitive Markov decision process (also known as a stochastic game) and states the use of two-players, finite-state and finite-action, zero-sum games, driven by payoffs and transition probabilities.

The difference between a MDP and a Markov Chain comes from the addition of actions and rewards. While Markov Chains only have one action per state, a MDP states may contain more than one action, with each action giving its unique rewards. MDPs with only one action for each state with identical rewards would be comparable to a Markov Chain.

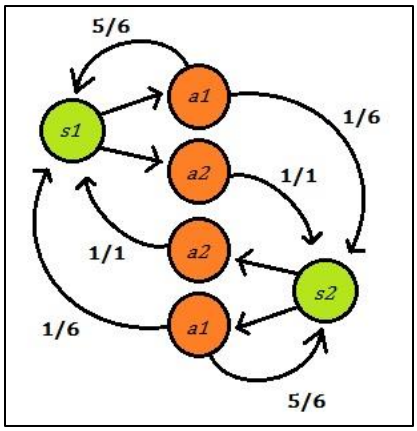

Figure 1: MDP of Roters' (1998) Dice Game containing a state for each player's turn and their actions.

Markov Chain

In this paper we'll be using a Markov Chain by assuming the player will only take the action to roll until the probability of rolling a 1 becomes a greater risk than rolling a number not equal to 1 or ending the turn early. By finding the optimal number of rolls by reducing the risk of losing points, the optimal strategy to maximize the number of points can be found.

A Markov chain has the Markov property, meaning the present state is independent to the past and future states,

$$P(X_{n+1} = x | X_1 = x_1, X_2 = x_2, \dots, X_n = x_n) = P(X_{n+1} = x | X_n = x_n)$$

The transition matrix *P* will remain the same for each state, and each state *x* can be written as a stochastic row vector with the relation $x^{n+1} = x^n P$. By using a Markov Chain, a stationary distribution can be found which will represent the optimal number of rolls to take per turn in the form of a steady state vector.

The Dice Game

With the start of the game, the player is assumed to always take the first roll over ending his turn since it would be meaningless to end the turn with no accumulated points.

The transition matrix *P* is defined as

$$P = \begin{bmatrix} \frac{5}{6} & \frac{1}{6} \\ \frac{1}{1} & \frac{0}{1} \end{bmatrix}$$

and the initial state x^1 = [5/6 1/6] due to the player always taking the action to roll over on the first turn. With just these two values, the probability of rolling a number not equal to 1 equates to 83.33%, while the probability of rolling a 1 equates to 16.67%. To calculate the next state, we multiply the initial state with the transition matrix *P* to get the second state $\boldsymbol{x^2}$

$$x^2 = x^1 P = \begin{bmatrix} \frac{5}{6} & \frac{1}{6} \end{bmatrix} \begin{bmatrix} \frac{5}{6} & \frac{1}{6} \\ \frac{1}{1} & \frac{0}{1} \end{bmatrix}$$

$$\boldsymbol{x^2} = \begin{bmatrix} \boldsymbol{\frac{31}{36}} & \boldsymbol{\frac{5}{36}} \end{bmatrix}$$

For the following state $\boldsymbol{x^3}$ simplified

$$\boldsymbol{x^3} = \begin{bmatrix} \boldsymbol{\frac{31}{36}} & \boldsymbol{\frac{1}{3}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\frac{5}{6}} & \boldsymbol{\frac{1}{6}} \\ \boldsymbol{\frac{1}{1}} & \boldsymbol{\frac{0}{1}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\frac{20}{27}} & \boldsymbol{\frac{7}{27}} \end{bmatrix}$$

Each with their decimal approximations:

$$\boldsymbol{x^1} = [\mathbf{0.8333\ \ 0.1667}]$$

$$\boldsymbol{x^2} = [\mathbf{0.8611\ \ 0.1389}]$$

$$\boldsymbol{x^3} = [\mathbf{0.7407\ \ 0.2593}]$$

With an increase in the number of states predicted, the estimates become more and more inaccurate. To represent a state that is independent of the previous state's probability, a steady state vector can be calculated. The steady state vector *q* is independent of the initial conditions and cannot be changed by the transition matrix *P*. The steady state vector is which is defined as:

$$q = \lim_{n \to \infty} x^{(n)}$$

Since *q* is independent of initial conditions, we make q unchanged by the transition matrix *P* by subtracting it by its identity matrix.

$$q \begin{bmatrix} \frac{5}{6} & \frac{1}{6} \\ \frac{1}{1} & \frac{0}{1} \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = q \begin{bmatrix} -\frac{1}{6} & \frac{1}{6} \\ \frac{1}{1} & -\frac{1}{1} \end{bmatrix}$$

$$[q_1\ q_2] \begin{bmatrix} -\frac{1}{6} & \frac{1}{6} \\ \frac{1}{1} & -\frac{1}{1} \end{bmatrix} = [0\ \ 0]$$

$$-\frac{1}{6}q_1 + q_2 = 0$$

Since we know q is a probability vector, $q_1 + q_2 = 1$, and after solving the simultaneous equations, we get the steady state distribution:

$$[q_1\ q_2] = \left[\frac{6}{7}\ \frac{1}{7}\right] = [0.8571\quad 0.1429]$$

With this steady state distribution, the probability of rolling a 1 averages to every 7th roll, and any roll after the 6th roll is not worth the risk of losing the accumulated points. This suggests that the optimal number of rolls is the 6th roll, and that rolling attempting to roll 6 times before ending your turn will maximize your score during the course of the game, regardless of the goal.

C++ Prototype

To test the accuracy of the results of the steady state vector, the C++ prototype program will mimic a player playing the dice game. The player will decide on a set number of rolls, which he will then attempt to roll that number of rolls every turn. The program will then average the total score every turn for each set of rolls. The program demonstrates what happens when a player goes through each set of rolls for every number of rolls between 1 and 12.

For instances, the beginning roll set has the program roll once for every turn where each roll will simulate a fair 6-sided die. The program will roll one die once per turn for 100,000 turns. Then second set of rolls begin, which the program will roll two rolls per turn for 100,000 turns. This goes on for 12 sets of rolls, where the last set of rolls will attempt to roll 12 times for 100,000 turns. The average is computed at the end and displayed to show how one number of rolls compares to the next.

Results of the Prototype

The results of the program initially showed that the optimal number of rolls per turn was 6, as the steady state vector suggested. However even after averaging 100,000 turns for each set of rolls, the top average score per set of rolls varies between 5 rolls per turn and 6 rolls per turn with an average score nearing 8 points per turn.

The average score for the numbers lower and greater than 5 and 6 stayed consistently lower, which can be seen illustrated by Figure 2. The average score peaks at 5 or 6 rolls per turn and gradually falls back down. Upon further testing, the average score scales back down to 3 points per turn at 17 rolls per turn. At 26 rolls per turn, the average score per turn is consistently less than 1.

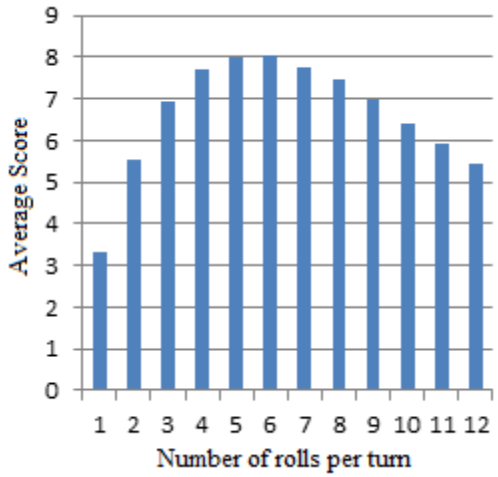

Figure 2: Sample game generated by C++ prototype program. The results show the optimal number of turns to take as suggested by the steady state vector was accurate.

For the set of 6 rolls per turn, the mean score was consistently averaging out to 8, varying in the hundredth place. The mean score of successful turns, turns where points were scored, was 23.98 and the probability of having a successful turn where points averaged to 33%.

For the set of 5 rolls per turn, the mean score was always close to the mean score of 6 rolls per turn, consistently averaging out to 8, varying in the hundredth place. The mean score of successful turns was 19.98 and the probability of having a successful turn where points averaged to 40%.

Roters (1998) showed that the optimal strategy was for the player to stop his turn early when his score had reached either 20 or 21, both of which achieved the maximum expected score on a maximum turn. This is comparable to the mean score for 5 rolls per turn at 19.98 points scored per successful turn.

Haigh & Roters' (2000) results showed that initial strategies as diverse as scoring 14 or 25 points can be optimal and for games with larger goals to win suggested scoring 20 to 21 points to be optimal. The average score of successful turns for both 5 and 6 rolls per turn both fell in Haigh & Roters' optimal number of points per turn, which confirms that the steady state vector stating taking 6 rolls out of every 7 can be optimal.

The Conclusion

In this paper we get a steady state vector from a transition matrix which showed the probabilities of the results of a single roll of a die. The steady state vector suggested that taking 6 rolls per turn was the optimal strategy. Results from the prototype program would partially support the

steady state vector, showing that the optimal number of rolls per turn was 5 or 6.

Future Work

The study done by this paper did not address certain important aspects of this game. A different, optimal strategy may exist if the target player is falling behind the opponent. Again, another optimal strategy may exist if the target player is initially ahead and would seek to win taking a safer number of rolls per turn. Further testing can include different rules and elements. Crocce & Mordecki (2009) tests three different variants of the dice game to test their strategy.

References

Haigh, J. & Roters, M. (2000). Optimal Strategy in a Dice Game. Journal of Applied Probability 37, 1110-1116.

Roters, M. (1998). Optimal Stopping in a Dice Game. Journal of Applied Probability 35, 229-235.

Shapley, L.S. (1953). Stochastic games. Proc. of the Nat. Acad. Sciences 39, 1095-1100.

Crocce, F. & Mordecki, E. (2009). Optimal Minimax Strategy in a Dice Game.

Puterman, M. (2005). *Markov Decision Processes Discrete Stochastic Dynamix Programming.* University of British Columbia.

Russell, S. & Norvig, P. (2009). *Artificial Intelligence, A Modern Approach 3e*