Monty Hall Problem & Birthday Paradox

Hanqiu Peng

Abstract

There are many situations that our intuitions lead us to the wrong direction, especially when we are solving some probability problems. In this presentation, I'll talk about 2 famous counterintuitive math problem: Monty Hall Problem and Birthday Paradox. I will also expand these problems and discuss some variations of them, as well as illustrate how these math questions are related to other fields.

1. Introduction

Monty Hall problem is a probability puzzle, based on the American television game show Let's Make a Deal and named after its celebrated host, Monty Hall. The rules of classical Monty Hall problem are as follows:

1. The number of doors in this game is three. At the beginning of the game, a prize is placed behind each door. Behind one of the doors is a new car. Others two doors are goats.
2. The player will choose one of the doors, if he chooses the door having a car, he wins the car.
3. The host of the game will open one of the two doors that the player didn't select which hiding a goat.
4. After opening one door, the host asks if the player would like to keep his initial selection or switch to the remaining unopened door.
5. The player should decide to either stay or switch.

In the different versions of this show that showed up irregularly on TV from 1963 until 2003, diverse solutions to deal with this were presented, with extensions such as adding the fourth doors in 1984. Given the intensive debates regarding the counter-intuitive nature of the solution, the problem's structure and the solution of the Monty Hall problem have been intensively discussed academically. On the other hand, birthday paradox considers a group of n people, and asks the probability that two or more people share the same birthday (same month and day). The purposes of this paper are to explore different ways of solving these two problems and also to elaborate on several variations and applications of them.

2. Methods and Results

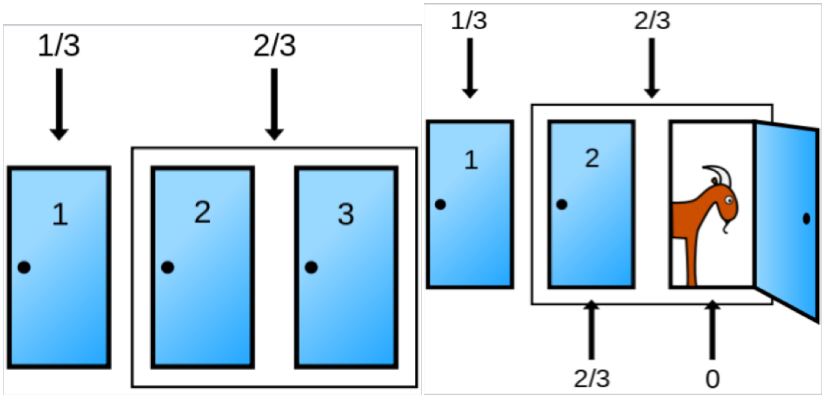
For Monty Hall problem, most people come to the conclusion that switching does not matter

because there are two unopened doors and one car and that it is a 50/50 choice. This would be true if the host opens a door randomly, but that is not the case; the door opened depends on the player's initial choice, so the assumption of independence does not hold. Here I will describe three ways to solve it.

The first method:

Car has a 1/3 chance of being behind the player's pick, and a 2/3 chance of being behind one of the other two doors. The host opens a door, the odds for the two sets don't change but the odds become 0 for the open door and 2/3 for the closed door. So the player should change in order to have a bigger chance of winning the car.

The following two figures can help understand this approach.



The second method:

If the player chooses to switch, there are three possible scenarios, each has equal possibility (1/3):

1. The player chooses the door containing car, and the host opens one of the two doors containing a sheep. Then if player switches, he will not win the car.
2. The player chooses the door containing sheep A, and the host must open the door containing sheep B. Then if player switches, he will win the car.
3. The player chooses the door containing sheep B, and the host must open the door containing sheep A. Then if player switches, he will win the car.

So the probability that the player win is 2/3 if he chooses to switch. But if he doesn't, the probability does not change, and is still 1/3. Hence the player should switch to get a bigger chance of winning the car.

The third method:

I wrote a program in python to play this game repeatedly. The program uses choice function from random library. The program was set to generate samples of 100000 games. In ten runs, the average percentage of times the switching strategy proved successful was 66.67%; in a separate set of ten runs, the strategy of always staying with the original choice succeeded 33.32%. The result also proves that if the player switches, he'll have a probability of almost 2/3 to win the car.

Things can be more intuitive if we just change the number of doors from 3 to 1000, and the host need to open 998 doors that don't have goats behind. Keep all the other rules the same as before, the player would definitely switch in order to have higher odds of winning the car.

As for Birthday Paradox, it is easy to see that the we need a group of 366 people in order to ensure two or more people share the same birthday. (If we don't consider leap year, there are only 365 possible birthdays). But if we want the probability that two or more people share the same birthday reaches 50%, do we need 366/2 people? The answer is no. The reasons are stated as follows.

Let p(n) denote the probability of at least two of the n people sharing a birthday. It is easier to first calculate the probability $\bar{p}(n)$ that all n birthdays are different.

$$\bar{p}(n) = 1\times (1-\frac{1}{365}) \times (1-\frac{2}{365}) \times \ldots \times (1-\frac{n-1}{365})$$

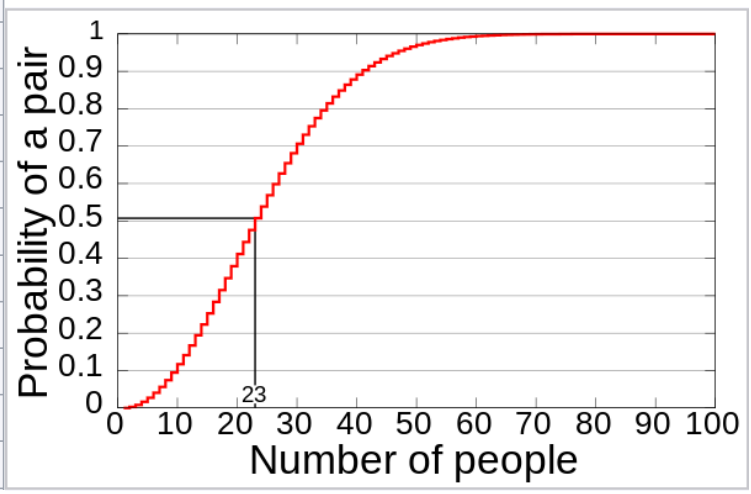
$$= (\frac{365\times364\times363\times\ldots\times(365-n+1)}{365})$$

$$= 365! / [365^n(365-n)!]$$

$$p(n) = 1 - \bar{p}(n)$$

The picture at the bottom left shows p(n)s for different n(s). When n is only 23, p(n) reaches 50%; when n is only 60, p(n) reaches 99%, which is far less than I expected.

n	*p*(*n*)
1	0.0%
5	2.7%
10	11.7%
20	41.1%
23	50.7%
30	70.6%
40	89.1%
50	97.0%
60	99.4%
70	99.9%
75	99.97%
100	99.999 97%
200	99.999 999 999 999 999 999 999 999 9998%
300	$(100 - 6\times10^{-80})$%
350	$(100 - 3\times10^{-129})$%
365	$(100 - 1.45\times10^{-155})$%
366	100%



The picture at the top right shows relation between n and p(n), we can generate it using R.

3. Variations of Monty Hall Problem

Rules are revised:

1. There are 3 doors, behind one of the doors is a new car. Others two doors are goats.
2. 3 players each picks one of the doors.
3. The host tells one player that he has chosen the one with goat, and his game is over.
4. For one of the two remaining players, should he exchange boxes with the other's in order to increase his chance of winning the car?

The answer is no, because these two players have the same probability of wining the car.

Reason:

In the Monty Hall problem, it is determined that the host will not announce the player has chosen the door with goat, and end his game, but the host will allow the player to change anyway.

However, in the revised problem, for certain player, the host may announce that he has chosen the door with goat, and end his game. So for a player, if he has the chance to exchange, he is lucky. But once such lucky event (coincidence) happens, we need to recalculate the probability of the original events using conditional probability formula.

So for the 2 remaining players, mark them as player1 and player2, and let player3 denote the player whose game is over.

P (player1 initially select the car | player3 didn't select the car)

$$= \frac{P(player1\ initially \text{ select the car} \cup \text{player3 didn't select the car})}{\text{player3 didn't select the car}}$$

$$= (\frac{1}{3} \times \frac{2}{3}) / \frac{2}{3}$$

$$= \frac{1}{2}$$

P (player2 initially select the car | player3 didn't select the car)

$$= \frac{\text{P(player2 initially select the car} \cup \text{player3 didn't select the car)}}{\text{player3 didn't select the car}}$$

$$= (\frac{1}{3} \times \frac{2}{3}) / \frac{2}{3}$$

$$= \frac{1}{2}$$

So these 2 player both have $\frac{1}{2}$ chance of winning the car, either of them needs to switch.

We could also expand the question to N doors:

There are N doors, two of which have money behind. You and your friend each chose a door. If the door a person choose has money, then the money is given to him. Your friend got the money from the door he chose. At this point, you are told that you can re-choose the door. So should you re-choose?

The case that your friend gets the money does not always happen, once it happens, we need to recalculate the probability of the original events (I choose the door with money).

P (your friend gets the money) = $\frac{2}{N}$

P (your friend and you both get the money) = $\frac{2}{N} \times \frac{1}{N-1} = \frac{2}{N(N-1)}$

P (you chose the door with money initially | your friend gets the money)

$$= \frac{\text{P(your friend and you both get the money)}}{\text{P(your friend gets the money)}}$$

$$= \frac{2}{\text{N}(\text{N}-1)} \; / \; \frac{2}{\text{N}}$$

$$= \frac{1}{\text{N}-1}$$

For the doors which were not selected by your friend, they have equal probabilities of having money behind. Since there are N-1 doors, the probability that each door has money is $\frac{1}{\text{N}-1}$.

Because the probabilities are the same ($\frac{1}{\text{N}-1}$), you don't need to re-choose.

After our calculation, we do find that the probability changes from $\frac{1}{\text{N}}$ (the probability that you get the money before you know that your friend gets the money) to $\frac{1}{\text{N}-1}$.

4. Application of Birthday Paradox

The mathematics of birthday paradox is used in the birthday attack, a brute-force cryptographic attack against hash function problems. A hash function is used to convert large amounts of data into a small, single-integer datum, called a hash value, which speeds up the looking up of items in a database. However, since there are limited number of hash values, there can be many collisions of hashes, which mean for two inputs $x_1 \neq x_2$, $f(x_1) = f(x_2)$. Birthday attack can be used to abuse communication between two or more parties, which depends on the higher likelihood of collisions found between random attack attempts and a fixed degree of permutations. With a birthday attack, it is possible to find a collision of a hash function in $2^{n/2}$, with 2^n being fixed degree of permutations + 1 (366 in our birthday paradox). The good news is birthday attack can be made unfeasible by increasing the hash value output size until it is unfeasible to find a collision. Just as if we want to find the probability that 2 or more people sharing the same birthday (same year, month and day instead of only month and day), we need a group of much more than 23 people to reach 50%. An ideal cryptographic hash function must be easy to compute a hash value for a message, infeasible to create a message with a given hash, infeasible to modify a message without changing the hash, and infeasible to find different messages with the same hash. SHA-256 (Secure Hash Algorithm 256) is the most secure hash algorithm that we have nowadays,

and it is widely used in many areas such as web browser and block chain.

5. Connection and Thought

I think my presentation and Charlie Strausser's both apply conditional probability to solve questions. Mine elaborates on the circumstances that people should use condition probability, whereas his expanded Rule of Bayes and talked about Bayesian inference. However, from a broader perspective, Charlie, Joe and I all illustrated using probability methods to uncover, predict, and model real world problems.

Before this presentation, I felt it was really challenging to do research in a short period of time and have a 30-40 minutes' presentation. But after I completed this task, I feel that my research and expression skills are improved a lot, because I have to search a large amount of information, and extract the essence; besides, I need to organize a long academic English presentation that I never did before, considering the content depth, interaction with audiences, clear explanation, etc. In summary, it is a process of self-improvement.

Therefore, I will keep what I did well and show more original ideas for my next presentation.

6. Reference

[1] Mazen Alrahili, *Simulation of the Monty Hall Problem*, October 2016, Clark Atlanta University.

[2] Michael Mitzenmacher, *The Monty Hall Problem: A Study*, 1986, Massachusetts Institute of Technology.

[3] Mmile Seitlheko, *The Birthday Paradox*, August 2011, Stellenbosch University.

[4] Shay Gueron, Simon Johnson, Jesse Walker, *SHA-512/256*, 2010.