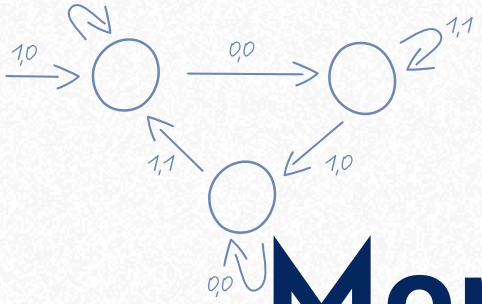
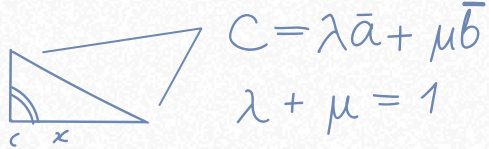


Monty Hall problem

Group 3

Yuxin Hu, Yufei Huang, Jiameng Ji,
Yiqing Jiang, Chengxu Lan



$$n\bar{c} - n\bar{a} = m\bar{b} - m\bar{c} \Rightarrow$$
$$\bar{c}(n+m) = m\bar{b} + n\bar{a}$$

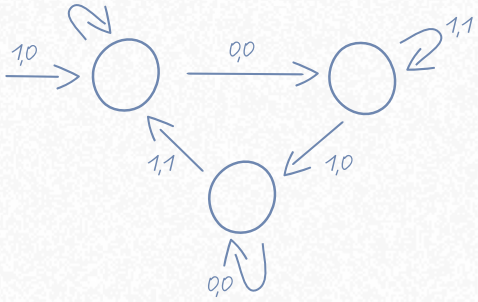
Table of contents

$$\lambda + \mu = 1$$

$$A = \frac{cd + ad}{n+m}$$

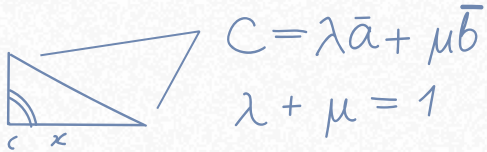
$$\frac{AC}{CB} = \frac{m}{n}$$

$$y = B$$
$$x = a$$

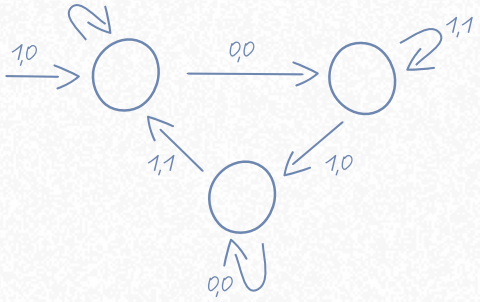


01

Introduction

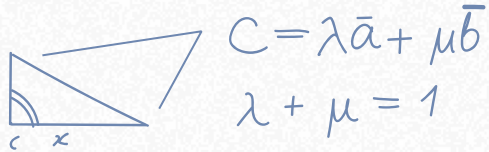


Suppose you're on a game show, and you're given the choice of three doors: Behind one door is a car; behind the others, goats. You pick a door, say No. 1, and the host, who knows what's behind the doors, opens another door, say No. 3, which has a goat. He then says to you, "Do you want to pick door No. 2?" Is it to your advantage to switch your choice?



02

Game



$$C = \lambda \bar{a} + \mu \bar{b}$$

$$\lambda + \mu = 1$$

Implement with React

Attribute

- Door
- carBehindDoor
- chosenDoorIndex
- revealedDoorIndex
- gameStage

Step 1: Initialization/reset

Randomly choose an index as the door behind which there's a car

```
setCarBehindDoor(Math.floor(Math.random() * 3));
setChosenDoorIndex(null);
setRevealedDoorIndex(null);
setGameStage(1);
setGameOver(false);
```

Step 2: Handle Player's Choice(2 choices)

Click

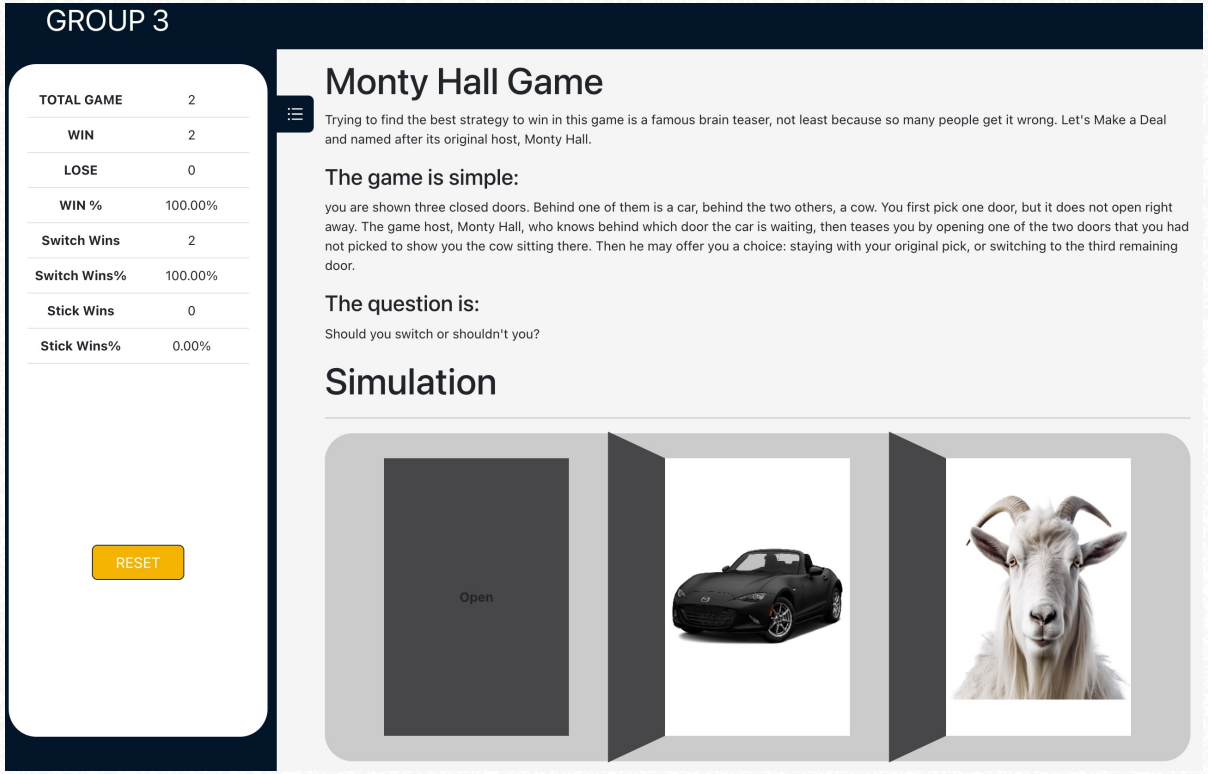
First click → set chosen index, reveal goat door, change game state to 2

Second click (player make the final decision) → reveal the right door, check win state

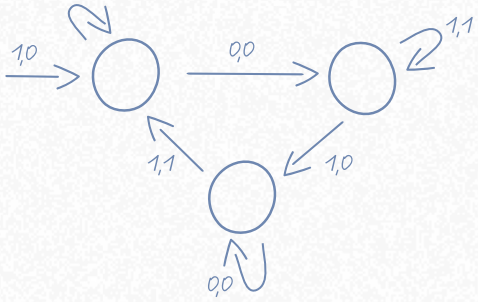
Front-End

```
  return (
    <div>
      <h1>Three Door Game</h1>
      {doors.map((door, index) => (
        <div
          key={door}
          className={`door ${index === chosenDoorIndex ? 'selected' : ''} ${index === revealedDoorIndex ? 'revealed' : ''} ${gameOver && index ===
          onClick={() => handleDoorClick(index)}
        >
          {getDoorContent(index)}
        </div>
      ))}
      <button onClick={resetGame}>Restart</button>
    </div>
  );
};


export default ThreeDoorsGame;
```

Play The Game Now!

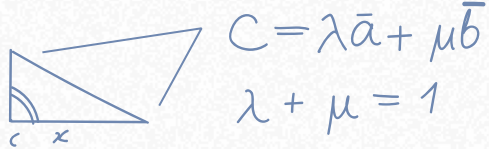


SCAN ME



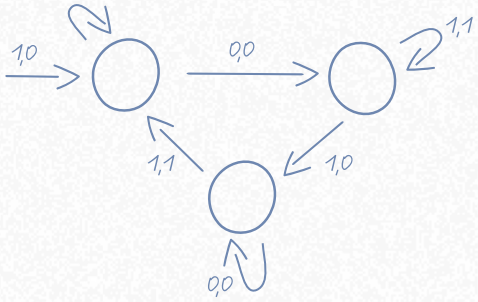
03

Probability Computations



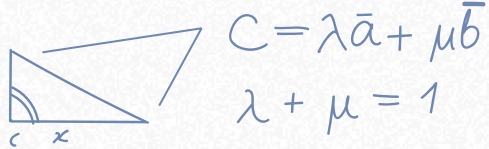
$$C = \lambda \bar{a} + \mu \bar{b}$$

$$\lambda + \mu = 1$$



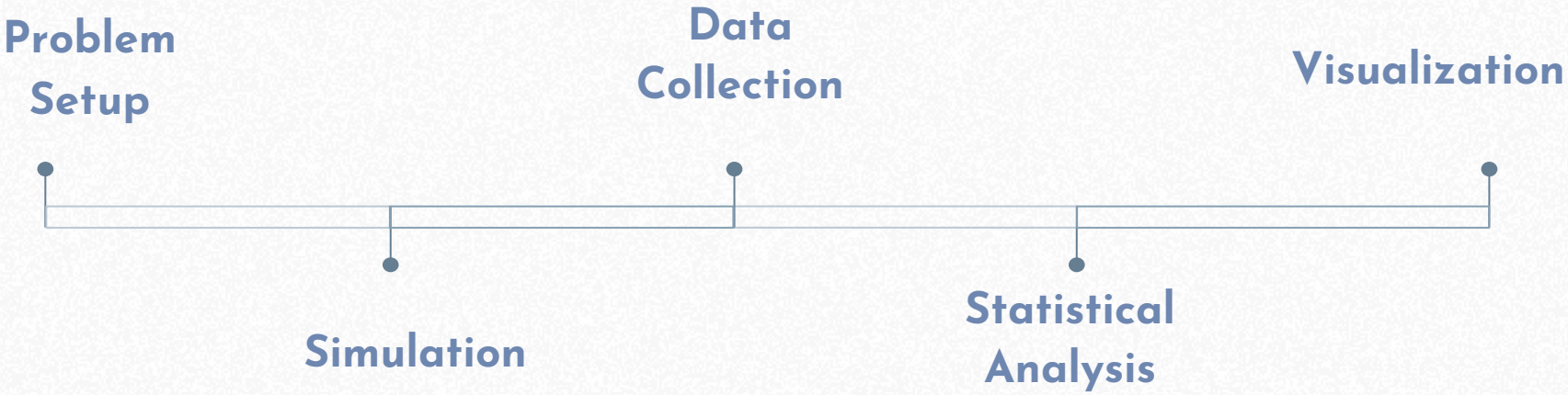
3.1

Random Simulation



$m(x - x_1)$

3.1 Random Simulation



$C = \lambda \bar{a} + \mu \bar{b}$

$\lambda + \mu = 1$

Problem Setup and Simulation

- **Doors:** Assume three doors, one of which hides a car (1), and the other two hide goats (0).
- **Guest's Initial Choice:** Randomly simulate the guest's (user's) initial choice of a door.
- **Host's Action:** Host will open one of the remaining doors there's no gift hidden.
- **Guest's Final Decision:** Simulate the user's decision to either **stick with the initial choice** or **switch to the remaining unopened door.**

```
doors = [0,0,0]
gift_i = random.randint(0,2)
doors[gift_i] = 1

#users choose a door
choose_i = random.randint(0,2)

# host open the door
host_i = -1
for i in range(len(doors)):
    if doors[i] != 1 and i != choose_i:
        host_i = i
        break
# user1 -- unchange
if change == 'unchange':
    win_unchange_nums = 0
    if choose_i == gift_i:
        win_unchange_nums =1
    change_choose_i = choose_i
    return win_unchange_nums,choose_i,gift_i,change_choose_i,host_i,doors

if change == 'change':
    win_change_nums = 0
    change_choose_i = choose_i
    for i in range(len(doors)):
        if i != host_i and i != choose_i:
            change_choose_i = i
    if  change_choose_i == gift_i:
        win_change_nums =1
    return win_change_nums,choose_i,gift_i,change_choose_i,host_i,doors
```

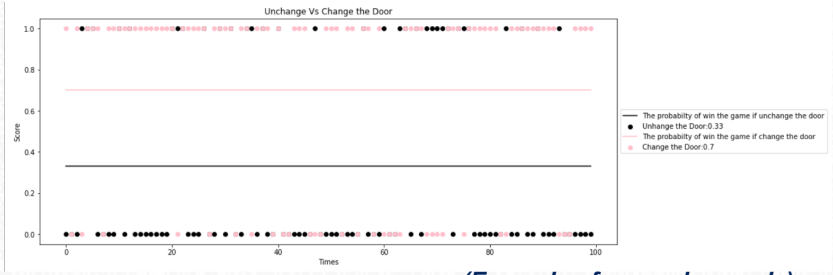
Data Collection and Visualization

- Run 100 simulations as one sub sample.
- Keep track the outcomes of each simulation.
- Calculate the probability of winning the gift based on two strategies.

```python
# get the results
def total_results(times,change_or_not,doors_array,win_loose):
    times_num = times

    # Behind Door
    total_door = []

    # if the car(1) behind the door, get 1 point; if the sheep(0) behind the door, get 0 point
    win_or_lose = []

    # initial choice of the door
    open_door = []

    # the gift behind this door
    gift_door = []

    # the door opened by the host
    host_door = []

    # final decision of the user
    final_open = []
    for i in range(times):
        total_door.append(doors_array()[0])
        doors_result = win_loose(doors_array,change_or_not)
        win_or_lose.append(doors_result[0])
        open_door.append(doors_result[1])
        gift_door.append(doors_result[2])
        final_open.append(doors_result[3])
        host_door.append(doors_result[4])

    #the probability of win the game (the car behind the door)
    prob_win = sum(win_or_lose)/times

    return total_door,win_or_lose,open_door,gift_door,host_door,final_open, prob_win
```

	Behind Door	Unchange: Initial Choice	Unchange: Host Choice	Unchange: Final Decision	Unchange: Gift	Unchange: Win or Lose
0	[1, 0, 0]	2	1	2	0	0
1	[1, 0, 0]	1	2	1	0	0
2	[0, 1, 0]	0	2	0	1	0
3	[0, 1, 0]	0	2	0	1	0
4	[0, 0, 1]	0	1	0	2	0
...	...	...	...	...	...	...

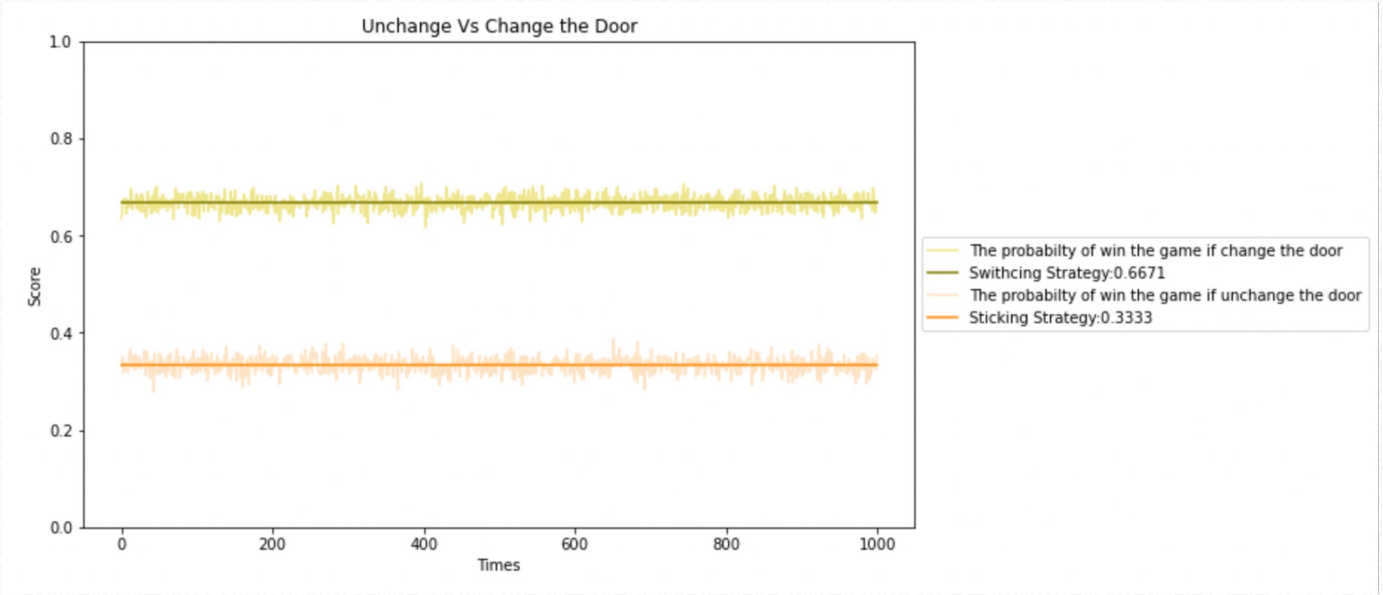
	Behind Door	Change: Initial Choice	Change: Host Choice	Change: Final Decision	Change: Gift	Change: Win or Lose
0	[1, 0, 0]	1	2	0	0	1
1	[0, 1, 0]	1	0	2	1	0
2	[0, 0, 1]	0	1	2	2	1
3	[1, 0, 0]	2	1	0	0	1
4	[0, 1, 0]	1	0	2	1	0
...	...	...	...	...	...	...

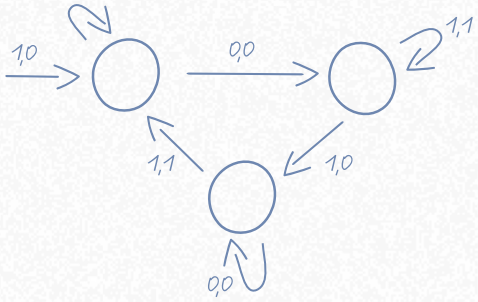


(Example of one sub sample)

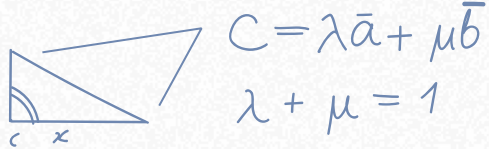
What We Found?

- We collected 1000 sub samples and found that
  - The probability of winning the gift based on the Sticking Strategy is around 33%.
  - The probability of winning the gift based on the Switching Strategy is around 66%.



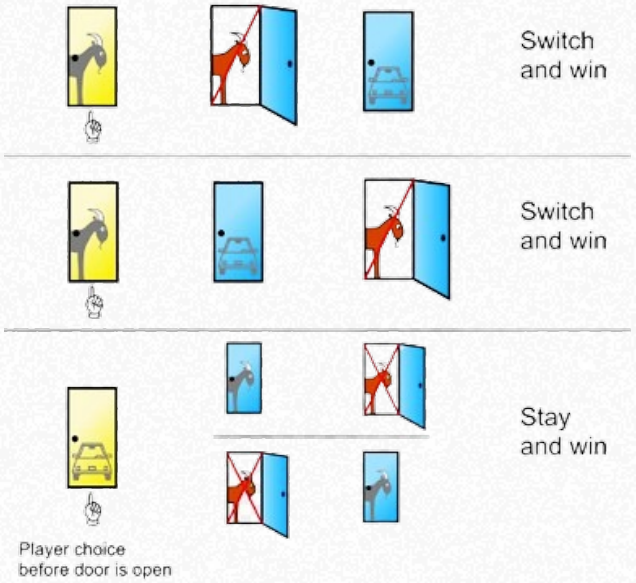


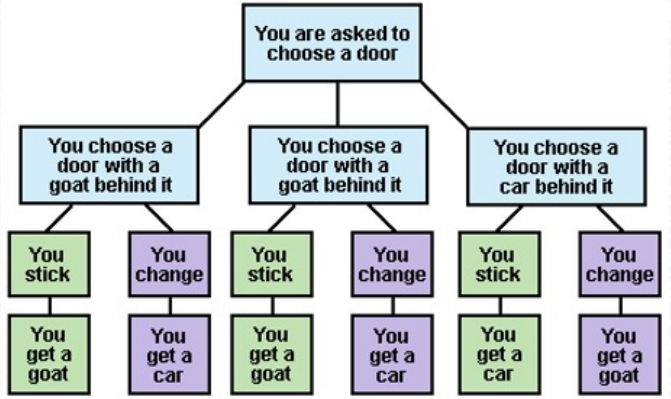
3.2
Bayes' theorem



All Possible Scenarios

If the contestant chooses door 1 and Monty opens one of the remaining doors to reveal a goat.





Door 1	Door 2	Door 3	Stick	Switch
Goat	Goat	Car	Lose	Win
Goat	Car	Goat	Lose	Win
Car	Goat	Goat	Win	Lose

The probability of winning by switching doors is 2/3,
while the probability of winning by sticking to the initial choice is only 1/3.

Bayes' theorem

The conditional probability of an event H given event E is the probability that event H occurs given that event E occurs and can be defined as:

$$P[H|E] = \frac{P[H \cap E]}{P[E]}$$

Bayes Theorem relates one conditional probability (e.g., the probability of a hypothesis H given an observation E) with its inverse (the probability of an observation given a hypothesis).

$$P[H|E] = \frac{P[H \cap E]}{P[E]} = \frac{P[E|H] \cdot P[H]}{P[E]}$$

Bayesian calculation

- H ---- the chosen door has a car behind it
- not H ---- the chosen door has a goat behind it.
- E ---- Monty has revealed a door with a goat behind it
- P(H|E) -- the probability of winning by sticking
- P(not H|E) -- the probability of winning by switching

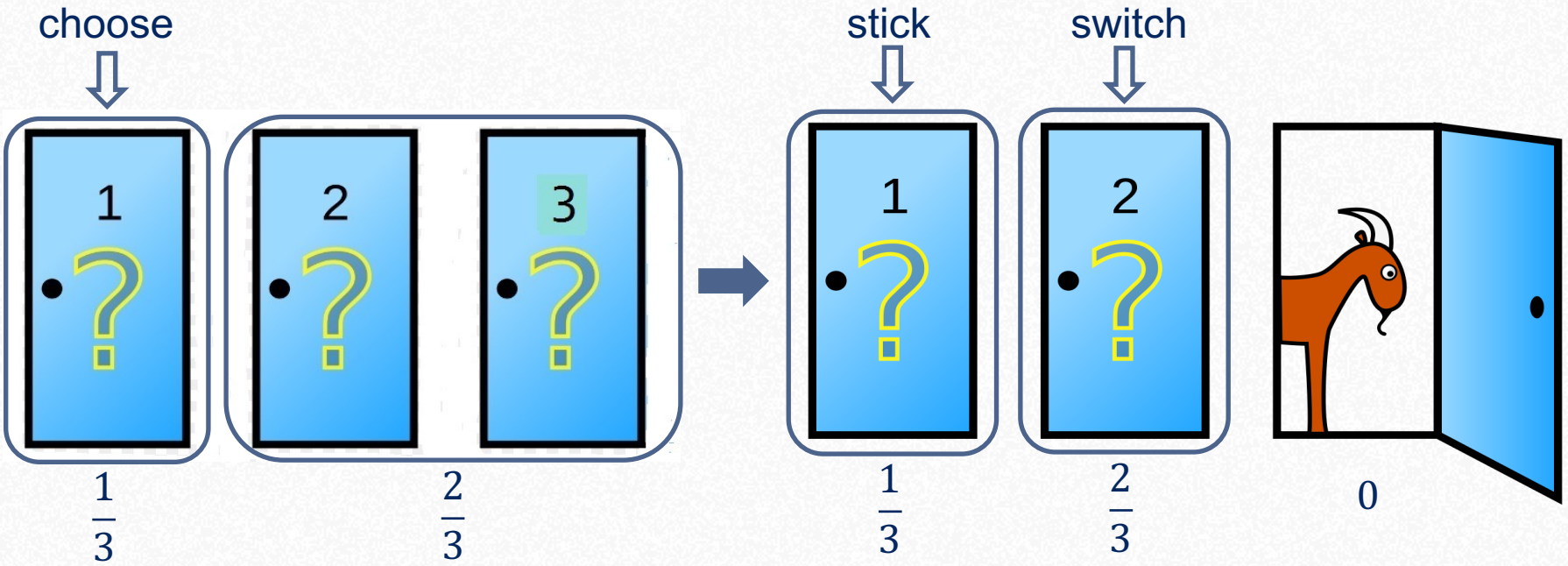
$$P[H|E] = \frac{P[E|H] \cdot P[H]}{P[E]} = \frac{P[E|H] \cdot P[H]}{P[E|H] \cdot P[H] + P[E|not\ H] \cdot P[not\ H]}$$

- P(H) = 1/3
- P(not H) = 1−P(H) = 2/3
- P(E|H) = 1
- P(E|not H) = 1

$$P(H|E) = \frac{1 \times \frac{1}{3}}{1 \times \frac{1}{3} + 1 \times \frac{2}{3}} = \frac{1}{3}$$

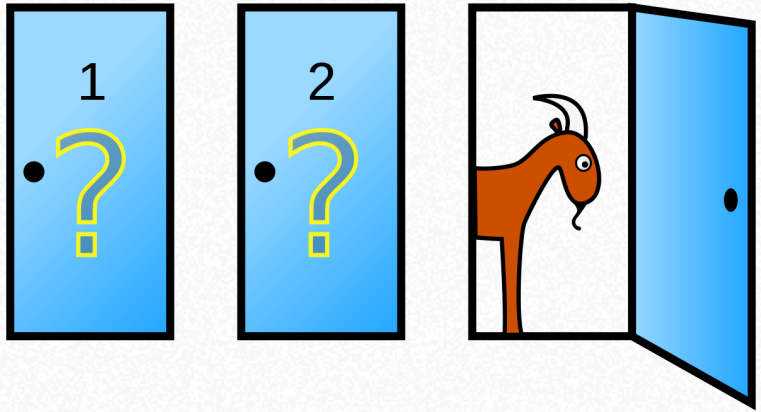
The probability of winning by sticking is only 1/3.

Probability of Winning



Similar Question

If, after you have chosen a door, **another guest** who doesn't know the door behind which the prize is located randomly selects one of the two remaining doors that you did not choose and reveals a goat behind it.



stick or switch?

It doesn't matter.

Similar Question

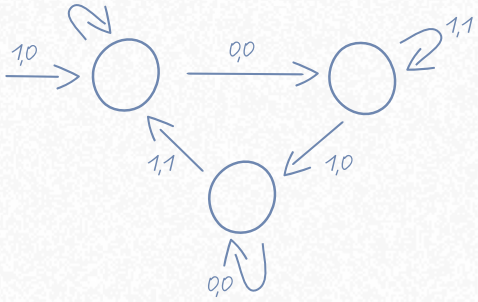
- H ---- the chosen door has a car behind it
- E ---- the guest has revealed a door with a goat behind it
- P(H|E) -- the probability of winning by sticking

$$P[H|E] = \frac{P[E|H] \cdot P[H]}{P[E]} = \frac{P[E|H] \cdot P[H]}{P[E|H] \cdot P[H] + P[E|not\ H] \cdot P[not\ H]}$$

- P(H) = 1/3
- P(not H) = 1−P(H) = 2/3
- P(E|H) = 1
- P(E|not H) = 1/2

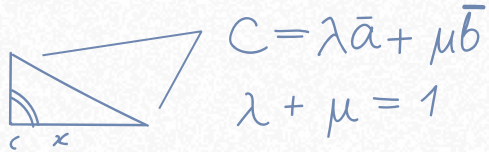
$$P(H|E) = \frac{1 \times \frac{1}{3}}{1 \times \frac{1}{3} + \frac{1}{2} \times \frac{2}{3}} = \frac{1}{2}$$

Whether you switch your choice or not, the probability of winning is **1/2**.



04

Conclusion



Conclusion

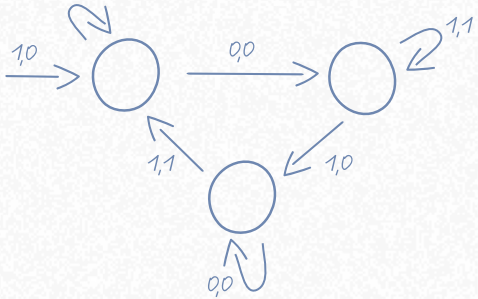
Through an in-depth exploration and analysis of the Monty Hall problem, we have arrived at an answer to the question and gained valuable insights. Our research confirms that switching doors is a statistically advantageous strategy. This conclusion not only validates statistical principles in theory but is also substantiated through simulated experiments and numerical calculations.

Limitations:

- Rely on simulated experiments and numerical computations, may not cover all variables present in real-life scenarios.
- More variants and extensions of the Monty Hall problem need to be investigated.

Reference

- https://www.cnblogs.com/bugxch/p/16177280.html#%E8%B4%9D%E5%8F%B6%E6%96%AF%E5%AE%9A%E7%90%86
- https://brilliant.org/wiki/monty-hall-problem/
- https://python.plainenglish.io/using-python-to-simulate-the-famous-monty-hall-problem-b4a9697894ba?gi=3039ff7cc691
- https://www.kaggle.com/code/ashimaabhaya/monty-hall-game-simulation



Thanks you!

Contact Information
Yuxin Hu: hu.yuxin3@northeastern.edu
Yufei Huang: huang,yufe@northeasyern.edu
Jiameng Ji: ji.jiam@northeastern.edu
Yiqing Jiang: jiang.yiqi@northeastern.edu
Chengxu Lan: lan.che@northeastern.edu