Paradoks Parrondo

Gabriela Godek, Fizyka Techniczna III

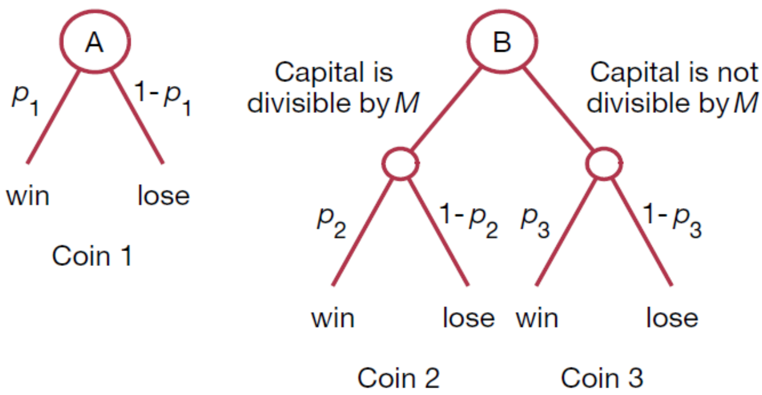
1. Przedstawienie problemu

Paradoks Parrondo to sprzeczne z intuicją zjawisko, w którym indywidualne strategie przegrywają za każdym razem, jednak ich kombinacja prowadzi do wygranej. Paradoksalny efekt można wyjaśnić poprzez umieszczenie wszystkich parametrów w jednej przestrzeni prawdopodobieństwa. Odtworzyłam ten efekt za pomocą prostej gry zależnej od kapitału.

2. Krótkie wyjaśnienie

Paradoks Parrondo zwrócił uwagę, ponieważ ma ogromny potencjał w przekształcaniu dwóch niekorzystnych sytuacji w korzystną. Koncepcja została dokładnie przeanalizowana od czasu jej pierwszego pojawienia się w 1996 i rozszerzona na inne potencjalne zastosowania.

Paradoks opiera się na dwóch grach: A i B. Gra B różni się od gry A lekką złożonością, która polega na dodatkowym mechanizmie:



Rysunek 1: Rozrysowane prawdopodobieństwo przegranej lub wygranej

Przy założeniach, że: $P_1 = \frac{1}{2} - \varepsilon$ ($\varepsilon = 0.005$ jest to bardzo mała liczba, której możemy przypisać złe wywarzenie monety którą rzucamy, bądź przysłowiowy pech).

$P_2 = \frac{1}{10} - \varepsilon$, $P_3 = \frac{3}{4} - \varepsilon$ oraz kapitał M = 3 możemy dojść do wniosku, że każda próba skazana jest na porażkę, ponieważ ten niewielki epsilon wpływa na wynik.

2.1. Jak więc za ich pomocą można wygrać?

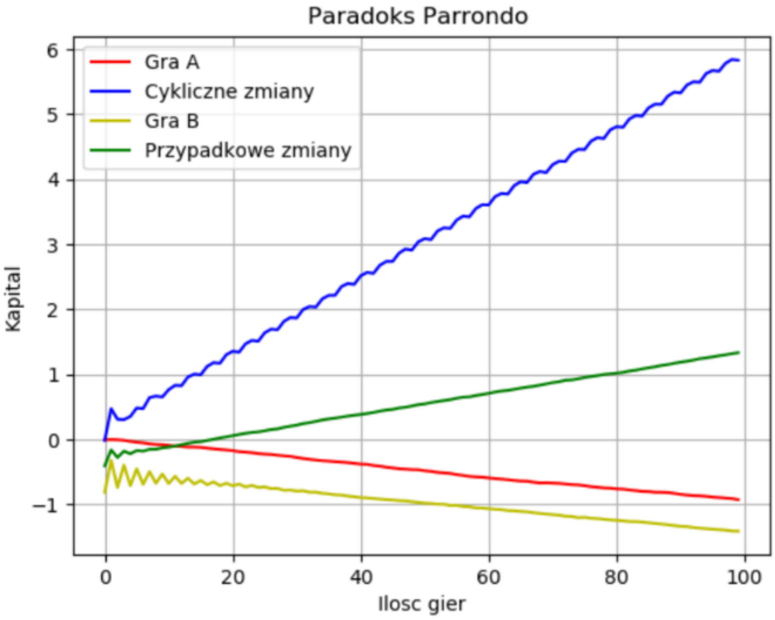
Wyobraźmy sobie, że przy grze A tracimy 1 zł za każdym razem gdy w nią gramy. Gra B zależy od naszego uzbieranego kapitału i tego czy jest podzielny przez M, jeśli kapitał jest wielokrotnością M to wygrywamy 3 zł, w przeciwnym wypadku tracimy 5 zł. Kapitał początkowy wynosi 100 zł, jeśli będziemy grać w grę A to przy każdym podejściu będziemy tracić po złotówce i tak w ciągu stu rund nasz kapitał osiągnie 0 zł. Dlatego naszą grę zaczniemy od gry A, celowo stracimy i nasz kapitał wynosić będzie 99 zł, ta liczba jest podzielna przez 3, tak więc wygramy 3 zł. Teraz nasz kapitał wynosi 102 zł. Jak można zobaczyć najpierw należało przegrać by później móc wygrać. Taka świadoma strategia pozwala nam wygrywać co przedstawia Wykres 1.

Paradoks ujawnia się, przy założeniach, że wybór gry będzie losowy.

3. Implementacja numeryczna

```python
# wariacja dla gry A, z jedna moneta
def gameA(outcome):
    if random.random() < (p1 - eps):
        return outcome + 1  # w przypadku wygranej (ktora jest mniej prawdopodobna nasz kapital wzrasta o jeden
    else:
        return outcome - 1  # w przypadku przegranej tracimy

# wariacja dla gry B, z dwiema monetami
def gameB(outcome):
    if outcome % 3 == 0:  # jesli nasz kapital jest wielokrotnoscia M = 3 to rozwazamy przypadek z moneta 3
        if random.random() < (p3 - eps):
            return outcome + 1   # wygrana
        else:
            return outcome - 1  # przegrana
    else: # jesli kapital nie jest wielokrotnoscia M rozwazamy monete 2
        if random.random() < (p2 - eps):
            return outcome + 1
        else:
            return outcome - 1

def randomSwitchGame(outcome):
    if random.random() < 0.5:  # funckja random() losuje liczbe z zakresu 0:1, dlatego warunek "<0.5"
        return gameA(outcome)  # sprawia, ze jest rowna "szansa" na wylosowanie gry A i gry B
    else:
        return gameB(outcome)
outcome = []
for i in range(100):  # ta petla wylczy nam srednia
    outcome.append(0)
for _ in range(50000):  # w petli wykonujemy dla samej gry A 50 000 powtorzen
    capital = 0
    for i in range(100):
        capital = gameA(capital)
        outcome[i] += capital  # powiekszamy nasz poczatkowy budzet o kapital
for i in range(100):
    outcome[i] /= 50000.0

outcome1 = []
for i in range(100):  # srednia
    outcome1.append(0)
for _ in range(50000):  # w petli wykonujemy dla samej gry B 50 000 powtorzen
    capital = 0
    for i in range(100):
        capital = gameB(capital)
        outcome1[i] += capital
for i in range(100):
    outcome1[i] /= 50000.0
outcome2 = []
for i in range(100):
    outcome2.append(0)
for _ in range(50000):
    capital = 0
    for i in range(100):
        capital = randomSwitchGame(capital)
        outcome2[i] += capital
for i in range(100):
    outcome2[i] /= 50000.0

outcome3 = []
for i in range(100):
    outcome3.append(0)
for _ in range(50000):
    capital = 0
    for i in range(100):
        if i % 3 == 0:
            capital = gameA(capital)
        else:
            capital = gameB(capital)
        outcome3[i] += capital
for i in range(100):
    outcome3[i] /= 50000.0
```

Rysunek 2: Program liczący 50 000 prób i wyliczający średnią dla 100 gier



Wykres 1: Graficzne przedstawienie prawdopodobieństwa wygranych i przegranych gier

4. Bibliografia

https://www.nature.com/articles/srep04244

https://en.wikipedia.org/wiki/Parrondo%27s_paradox