Bài toán: Traveler's Dilemma

"An airline loses two suitcases belonging to two different travelers. Both suitcases happen to be identical and contain identical antiques. An airline manager tasked to settle the claims of both travelers explains that the airline is liable for a maximum of $100 per suitcase—he is unable to find out directly the price of the antiques."

"To determine an honest appraised value of the antiques, the manager separates both travelers so they can't confer, and asks them to write down the amount of their value at no less than $2 and no larger than $100. He also tells them that if both write down the same number, he will treat that number as the true dollar value of both suitcases and reimburse both travelers that amount. However, if one writes down a smaller number than the other, this smaller number will be taken as the true dollar value, and both travelers will receive that amount along with a bonus/malus: $2 extra will be paid to the traveler who wrote down the lower value and a $2 deduction will be taken from the person who wrote down the higher amount. The challenge is: what strategy should both travelers follow to decide the value they should write down?"

The two players attempt to maximize their own payoff, without any concern for the other player's payoff.

[1] https://en.wikipedia.org/wiki/Traveler%27s_dilemma

Mô tả sơ lược:

Cả hai người hành khách du lịch phải độc lập đưa ra mức giá cho chiếc va li hành lý của họ, mức giá của chiếc va li phải từ 2$ đến 100$. Giả sử người A đưa ra mức giá là a và người B đưa ra mức giá là b. Ở đây sẽ xảy ra 3 trường hợp, A và B sẽ được nhận được số tiền bồi thường như sau:

- Nếu a > b => A sẽ nhận b - 2$, B sẽ nhận b + 2$
- Nếu a < b => A sẽ nhận a + 2$, B sẽ nhận a - 2$
- Nếu a = b => Cả A và B nhận a$ (=b$)

Như vậy ta sẽ tìm cách để có thể tối ưu được số tiền thưởng.

Thách thức:

Phải đạt được sự tối ưu số tiền mình nhận được. Sự tối ưu đó có thể hiểu theo hai góc độ.

Dưới hai góc độ:

- Tổng tiền nhận được của cả hai phải lớn.
- Người đưa ra mức giá chỉ cần nhận được số tiền lớn hơn người còn lại, đạt được số tiền mình đề ra.

Phải dự đoán được hành vi của các tác nhân (có theo một chiến lược nào không hay sử dụng ngẫu nhiên ?)

Thực nghiệm:

- **Ma trận số tiền nhận lại được biểu diễn:**

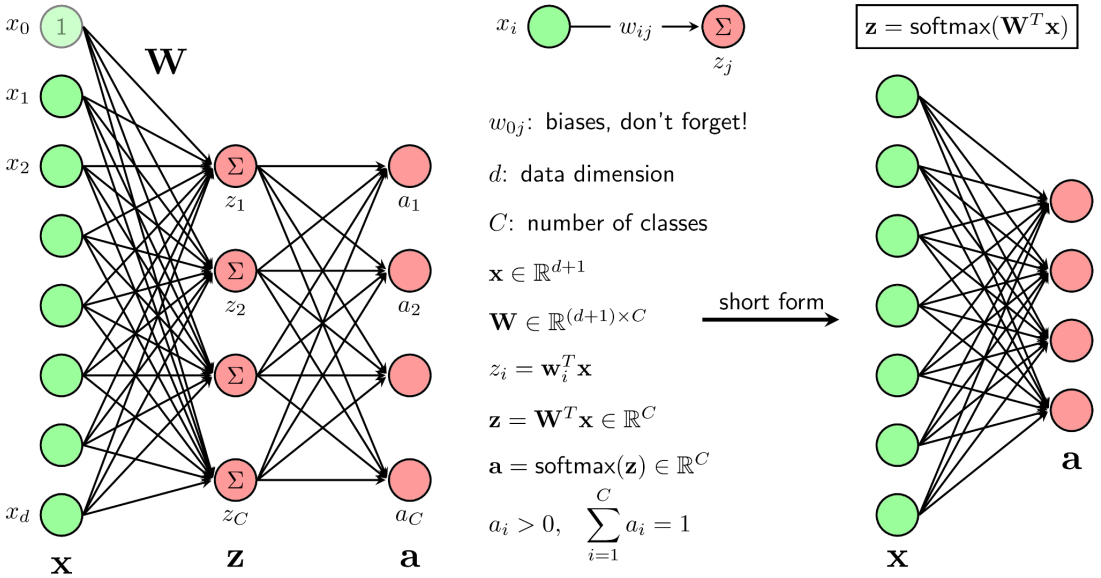
Canonical TD payoff matrix

	100	99	98	97	⋯	3	2
100	100, 100	97, 101	96, 100	95, 99	⋯	1, 5	0, 4
99	101, 97	99, 99	96, 100	95, 99	⋯	1, 5	0, 4
98	100, 96	100, 96	98, 98	95, 99	⋯	1, 5	0, 4
97	99, 95	99, 95	99, 95	97, 97	⋯	1, 5	0, 4
⋮	⋮	⋮	⋮	⋮	⋱	⋮	⋮
3	5, 1	5, 1	5, 1	5, 1	⋯	3, 3	0, 4
2	4, 0	4, 0	4, 0	4, 0	⋯	4, 0	2, 2

- **Thử nghiệm:**

- **Mô hình hóa tính toán:**

  Bài toán sẽ được áp dụng thuật toán softmax nhiều bậc để đưa ra chiến lược tối ưu tiền nhận được.



- **Phương pháp giải quyết:**
  - Thông thường đối với những bài như thế này, ta sẽ sử dụng cân bằng Nash (Nash Equilibrium)

    Tuy nhiên trong trường hợp này, sau khi sử dụng cân bằng Nash thì điểm hội tụ của chiến lược này nằm ở ô (2,2), nếu như vậy thì số tiền nhận được của cả hai tác nhân đạt được là rất nhỏ so với giá trị có thể có được của chiếc va li (100$).

- Sử dụng Hierarchical Softmax thay thế cho Nash Equilibrium bởi những đặc tính khác biệt nổi trội như sau:
    + Khi xây dựng các hệ thống ra quyết định phải tương tác với con người, việc tính toán cân bằng Nash không phải lúc nào cũng hữu ích. Con người thường không chơi chiến lược cân bằng Nash. Ngay cả khi các tác nhân có thể tính toán cân bằng Nash, họ có thể nghi ngờ rằng đối thủ của họ có thể thực hiện phép tính đó.
    + Việc tối ưu hóa số tiền bồi thường thì việc sử dụng Softmax là thuật toán được mọi người nghĩ đến. Ở đây ta sử dụng Hierarchical Softmax để tăng sự chính xác.
    + Để tối ưu hóa tiền bồi thường thì việc sử dụng Nash Equilibrium quy về điểm (2,2) là không thể chấp nhận, vì đây là mức tiền ít nhất.

- **Code:**
    - Khởi tạo các miền giá trị như actions thuộc đoạn từ 2$ đến 100$ và hàm normalize để chuẩn hóa.

Actions are the dollar amounts ($2–$100). A strategy is a distribution over these amounts, represented as a vector.

```
const Action = Integer
const Strategy = Vector{Float64}
actions = collect(2:100)
normalize(a::Vector) = a / sum(a);
```

    - Hàm mô tả cách chia bồi thường:

$$U_i(a_i, a_{-i}) = \begin{cases} a_i & \text{if } a_i = a_{-i} \\ a_i + 2 & \text{if } a_i < a_{-i} \\ a_{-i} - 2 & \text{if } a_i > a_{-i} \end{cases}$$

```
function utility(own::Action, opponent::Action)
    if own == opponent
        return own
    elseif own < opponent
        return own + 2
    else
        return opponent - 2
    end
end;
```

    - Hàm softmax_response() để mô hình hóa cách mà người dùng sẽ chọn hoạt động của họ:

$$\pi_i(a_i) \propto \exp(\lambda U_i(a, \pi_{-i}))$$

```
function softmax_response(λ::Real, opponent::Strategy)
    s = Float64[exp(λ*utility(a, opponent)) for a in actions]
    normalize(s)
end;
```

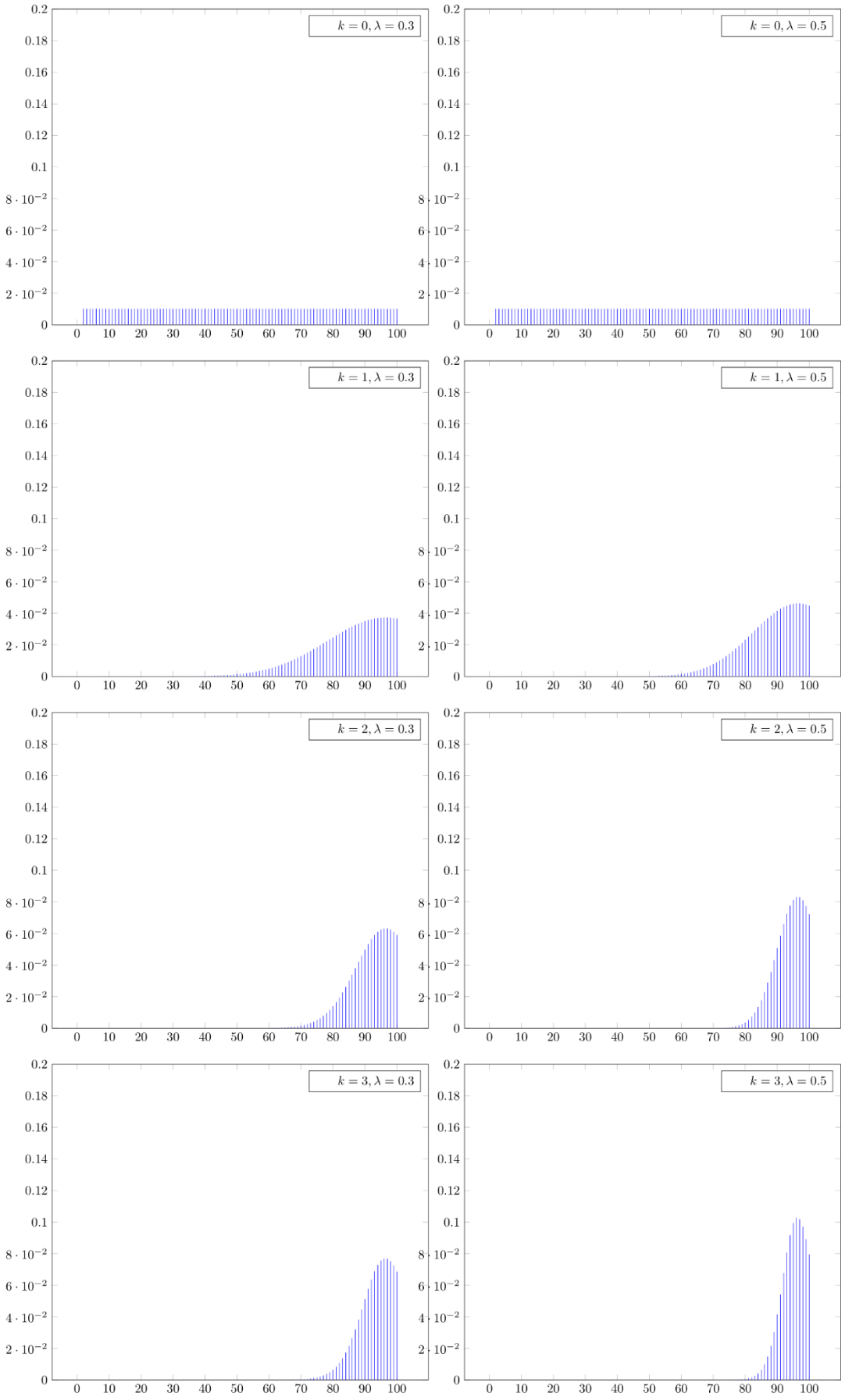
- Cuối cùng là hàm HierarchicalSoftmax() thể hiện softmax_response() theo độ sâu k, các tác nhân cấp k sẽ tiến hành chọn hành động của họ dựa trên các tác nhân k - 1 đã làm:

```
function HierarchicalSoftmax(k::Integer, λ::Real)
    if k == 0
        return normalize(ones(length(actions)))
    else
        return softmax_response(λ, HierarchicalSoftmax(k - 1, λ))
    end
end;
```

[2] https://github.com/sisl/aa228-notebook/blob/master/07-Games.ipynb

- **Phân tích:**

  Kết quả thử nghiệm:



Ý nghĩa:

Từ kết quả trên ta thấy được mọi người có sẽ có xu hướng lựa chọn các action từ vùng 97$ đến 100$ mặc dù điểm cân bằng Nash chỉ là 2$. Hệ số lamda càng cao thì biểu đồ càng hướng về 100$.

Tóm tắt kết quả:

Dựa vào tiêu chí tối ưu số tiền bồi thường va li nhận được thì thuật toán này giúp đưa ra chiến thuật được đánh giá rất tốt, số tiền dự đoán đưa ra rời vào khoảng 97-100 lớn hơn rất nhiều nếu so với 2$ ở dạng cân bằng Nash.

Điểm mạnh:

- Lựa chọn thuật toán Hierarchical Softmax phù hợp với việc tối ưu hóa tiền bồi thường nhận được.
- Kết quả thử nghiệm được trực quan hóa giúp người xem dễ hình dung về kết quả.

Điểm yếu:

- Chưa chạy được nhiều thuật toán khác nhau để có thể so sánh hiệu suất của nhiều thuật toán khác nhau.