Robot Learning: Reinforcement Learning

Lecture 9

양정연

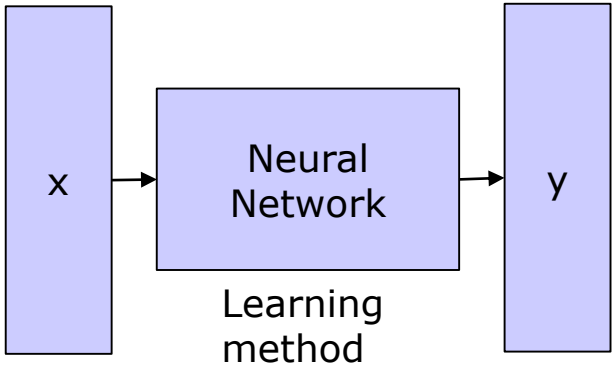
2020/12/10

1 Process Learning (Sequence Learning)

1. What learning methods try to learn?

- Think about Neural Network and Linear Regression
- Is it True? Linear Regression is Equal to Basic NN

Nonlinear Kernels works for creating Hyperplane.

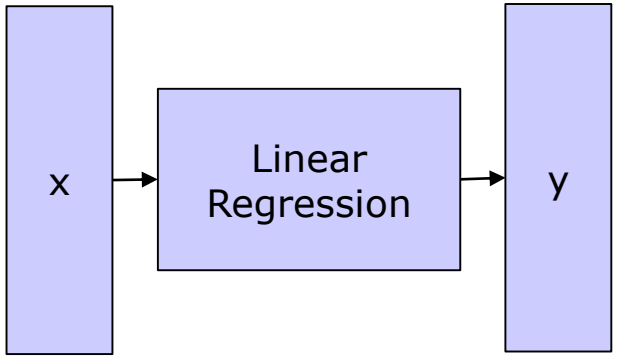


Data (x, y) are given

Goal is to find the proper function, NN

$$y = f(x) = w_2 \Phi(w_1 x)$$

Learning the function!



Data (x, y) are given

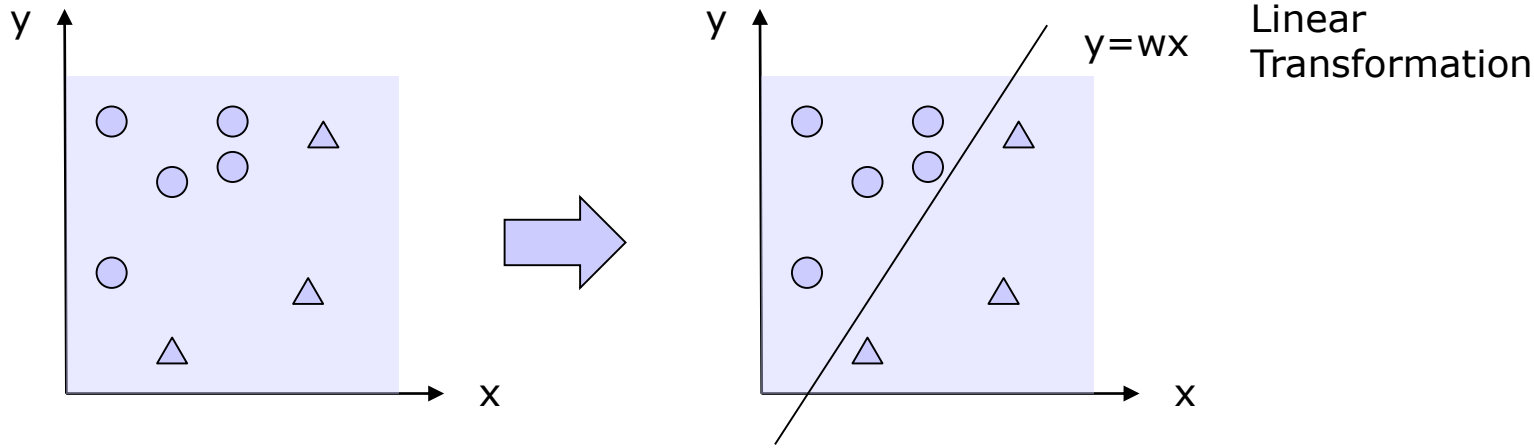
Goal is to find the proper function, y=ax+b

$$y = f(x) = wx$$

Learning the function!

Definition of Hyperplane

- Hyperplane is a subspace that
  - reduces the dimensionality of an original space.



$\bigcirc \in R^2 \quad \triangle \in R^2$

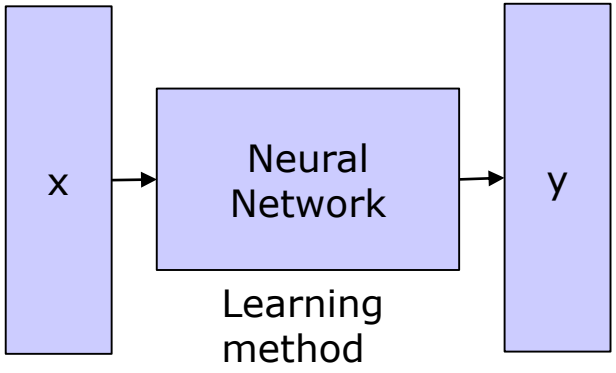
$\bigcirc \in \{x, y \mid y > wx\}$

$\triangle \in \{x, y \mid y < wx\}$

- **2 Dim. Space is projected on a New Space with y=wx.**

2. What learning method try to learn?

- Think about Neural Network and Control
- Control is the Modeling based Method.

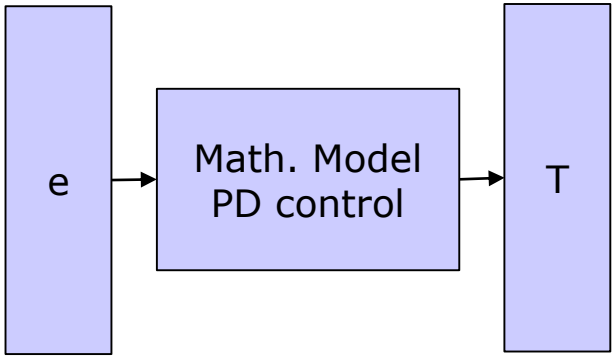


Data (x, y) are given

Goal is to find the proper function, NN

$$y = f(x)$$

Learning the function!



We don't care about the output, T

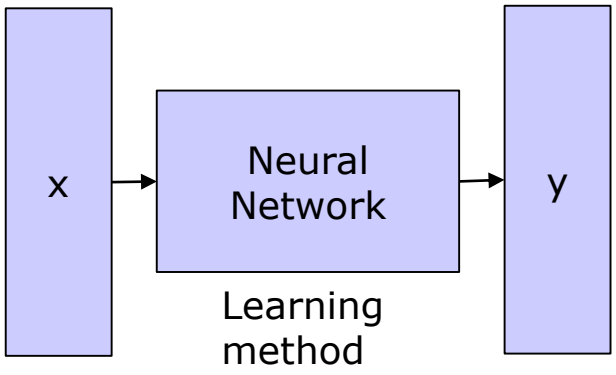
Error, e is given and we design a controller,

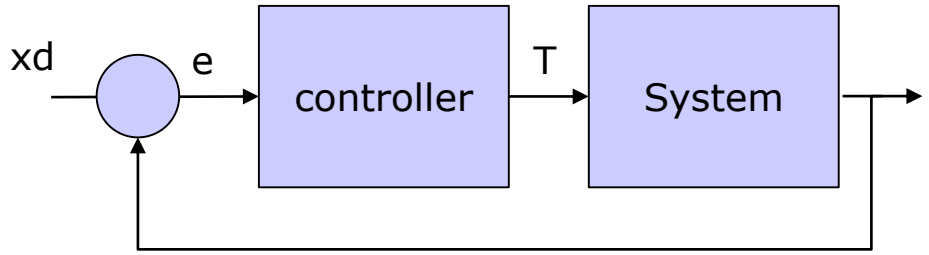
$$T = K_p e + K_d \dot{e}$$

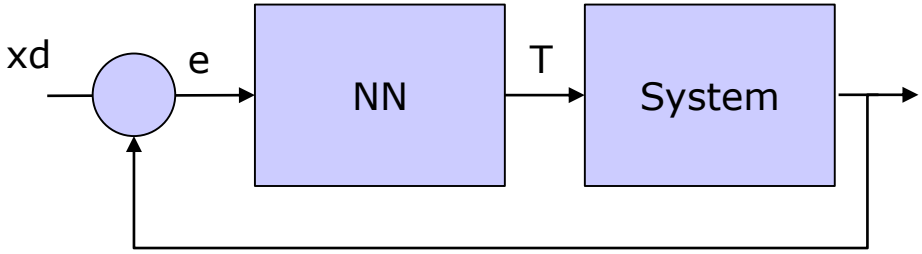
Design or Model a function

3. If we use NN in the control, What happens?

- Think Control Diagram
- → Input and output are different in every time → Failed







Data (x, y) are given

Goal is to find the proper function, NN

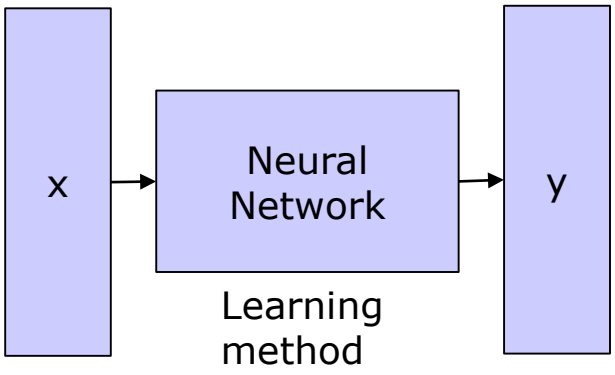
$$y = f(x)$$

Learning the function!

$$T = f(\mathrm{e})$$

4. It is different with Function Estimation. It tries to learn Dynamic System.

- Think Control Diagram
- → Input and output are different in every time → Failed



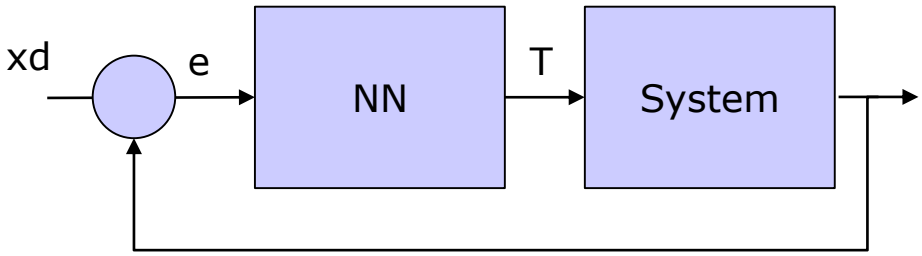
Data (x, y) are given

Goal is to find the proper function, NN

$$y = f(x)$$

Learning the function!

≠

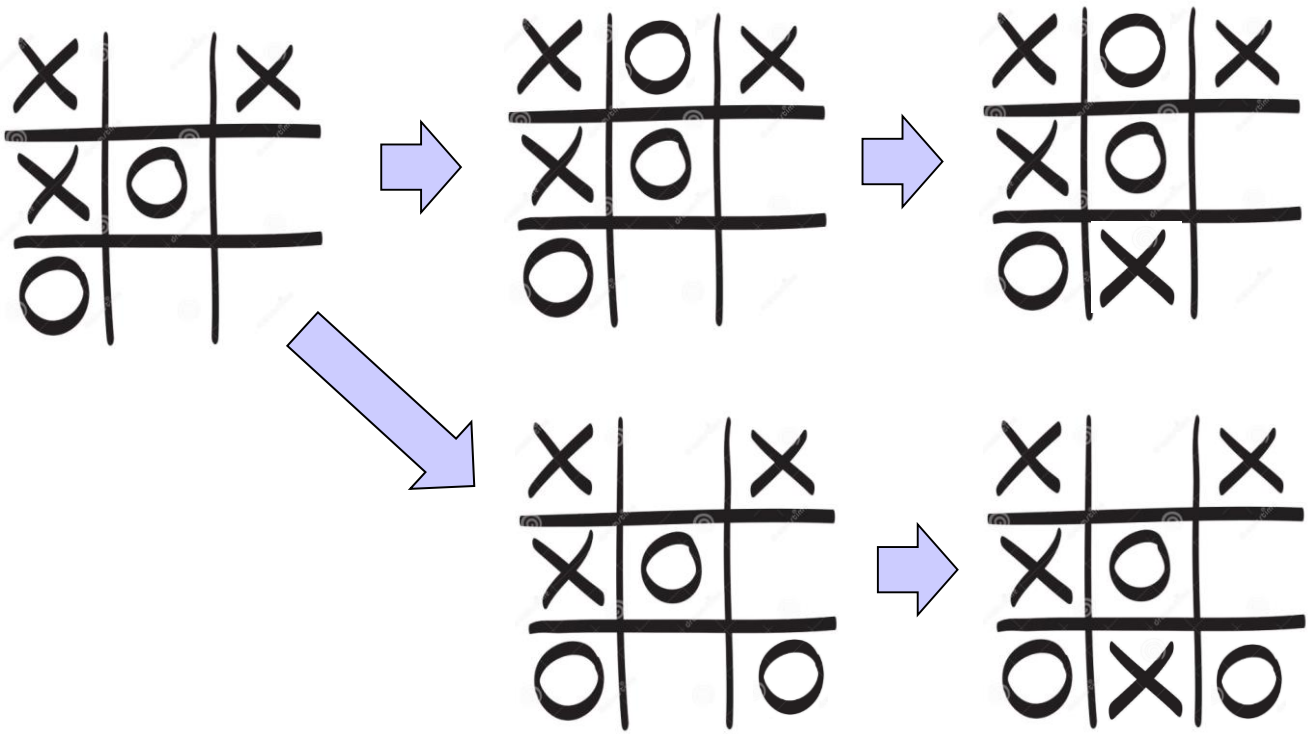


Data 'e' are given, but 'T' are NOT.

Goal is to find the NN that makes x to be xd.

→ Eg. Recurrent Neural Network (RNN)
→ Tries to solve System Dynamics.

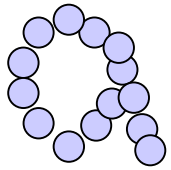
How do Learning this case?

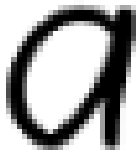


- Something is different with Neural Network…

Learning Goal is “to learn a Process”

- Process is a series of sequential events.
- Design issues
  - We can make a process like a instant data

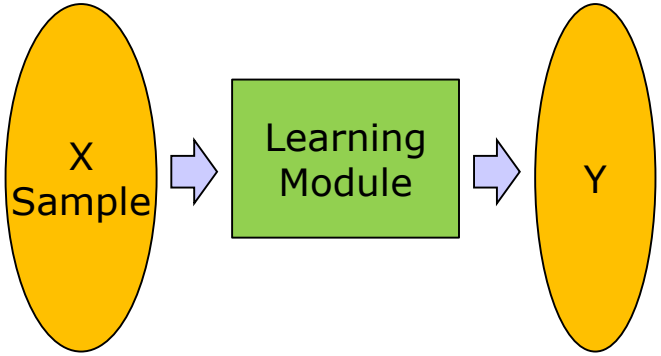




  - Each dot in a space is a process → Learning a process
  - Every dots in a space are a non process but an image.
- How to learn a process?
  - There are two types of popular methods.
  - 1. Recurrent Neural Network, RNN
  - 2. **Reinforcement Learning, RL**
  - (3. Classifier → Hidden Markov Model, HMM)

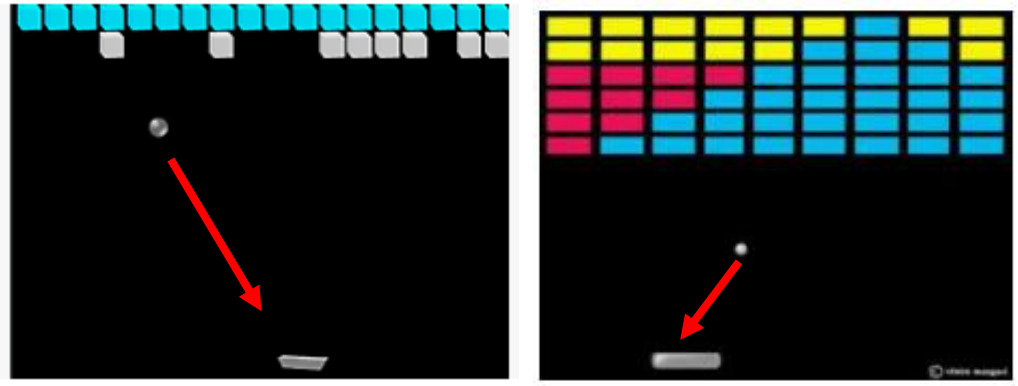
Types of Learning methods

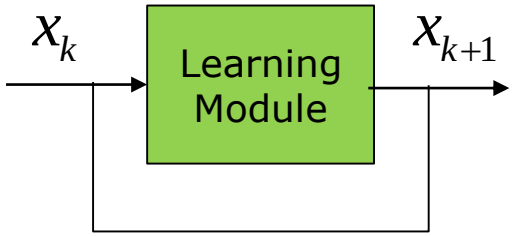
- Learning Function



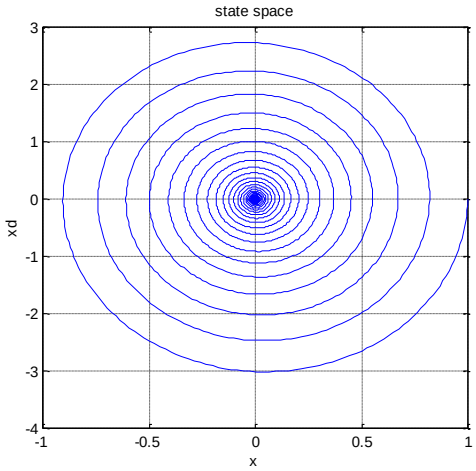
$$NN\ learns,$$
$$Y = NN(X) = f(X)$$

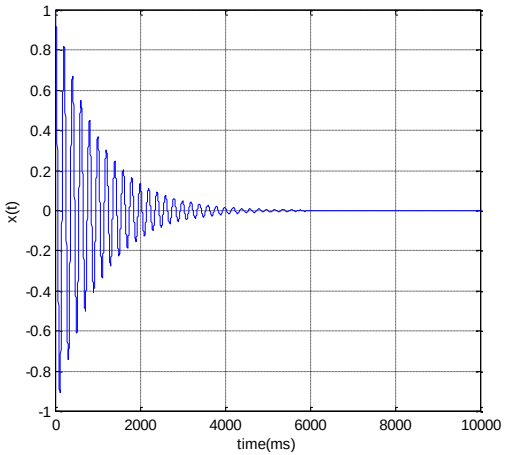
- Learning Dynamics (or Sequences)

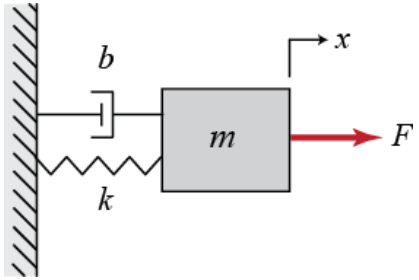




How to learn Dynamics or Sequences





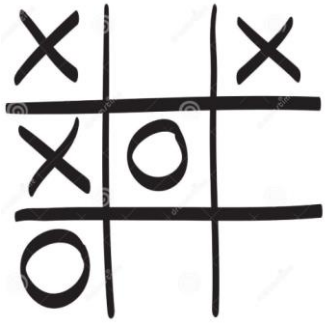
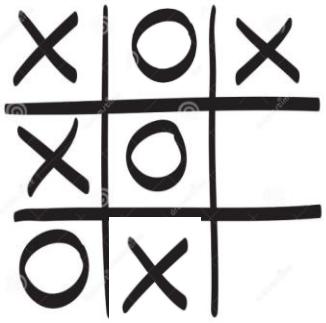


$$m\ddot{x} + c\dot{x} + kx = F(t)$$

- 1. Recurrent Neural Network (RNN)
- 2. Recursive Neural Network (Becoming Perished)

- **3. Reinforcement Learning**
  - **However, it has some different features(Stochastic)**

Remind Tic-Tac-Toe

$x_k =$  $x_{k+1} =$ 

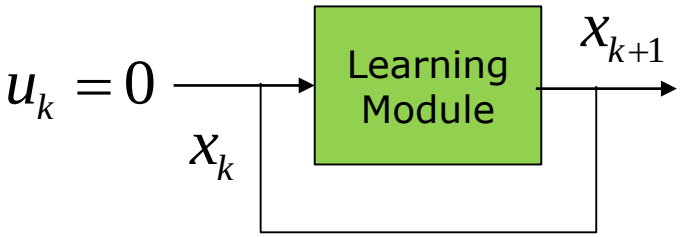
- Question 1: What is a state vector in Tic-Tac-Toe?

$$x_k = [X, \phi, X, X, O, \phi, O, \phi, \phi]^T \xrightarrow{\text{Action}} x_k = [X, O, X, X, O, \phi, O, \phi, \phi]^T$$

$$\hookrightarrow \quad x_{k+1} = [X, O, X, X, O, \phi, O, X, \phi]^T$$

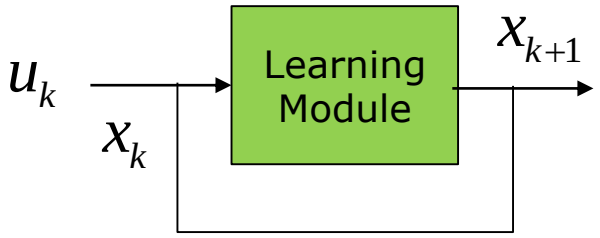
- Question 2: What is an input and an output?
  - There are No Input and Output
  - There are only States and Actions

Input / Output is Similar to Control



$$m\ddot{x} + c\dot{x} + kx = 0$$

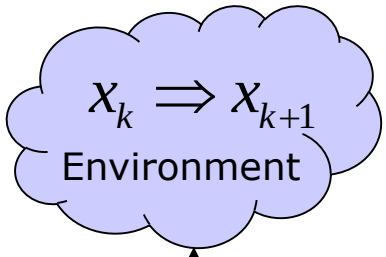
$$\dot{X} = f(X, t)$$



$$m\ddot{x} + c\dot{x} + kx = u(t)$$

$$\dot{X} = f(X, u, t)$$

$case : u_k = 0$



$case : u_k \neq 0$

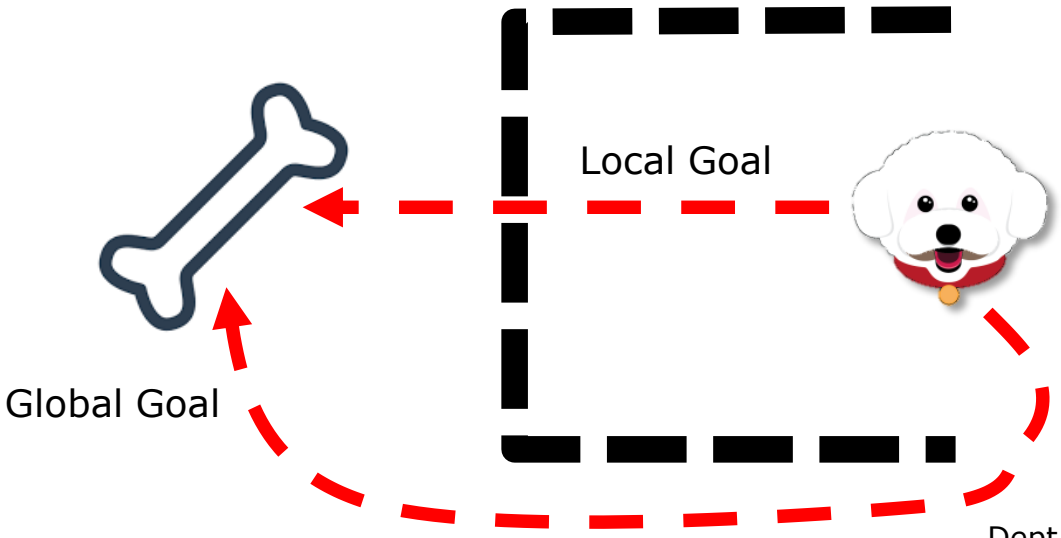
2

What we Minimize?

Or What we Maximize?

Local and Global Goal

- Minimize error is the strong issue in Learning Method
  - In other words, Optimization is the Learning.
- Local Goal
  - A dog tries to follow a bone → But failed
- Global Goal (Nearly Global Optimization)
  - A dog tries to avoid obstacles → locally weak but globally success



Example

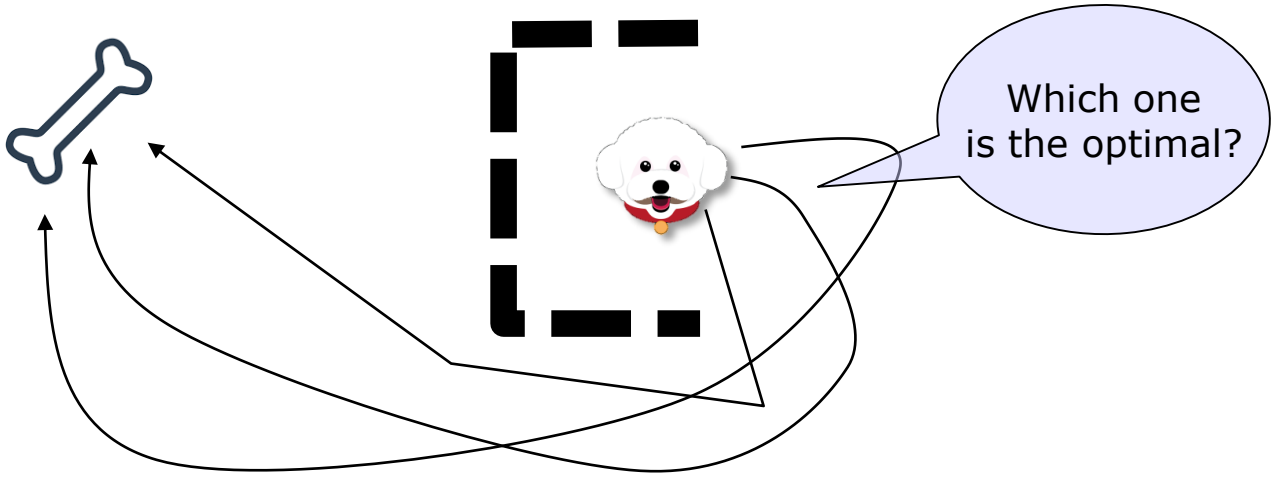
Try to follow the local goal

It is Not sure, but finally get the goal

Absolutely, success!

- Learning is achieved by several trials.
- 1st turn : follow the local goal, but get the goal eventually.
- 2nd turn: unsure but a dog remembers the solution
- 3rd turn: Fully being learned.

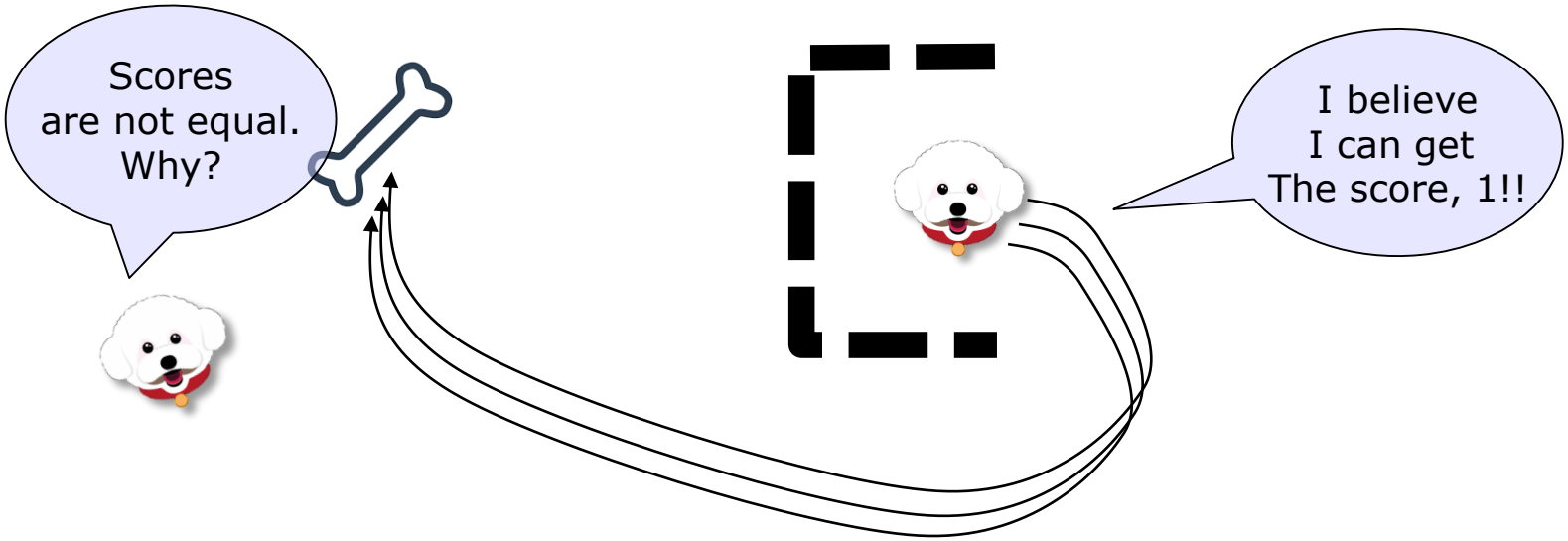
Issue. 1. There are so Many, Many Paths.



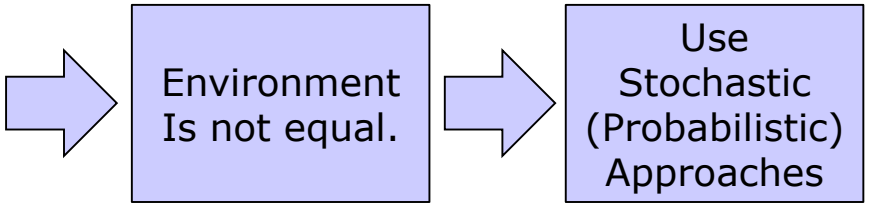
- Step 1. Each path must be scored
- Step 2. I might find the best one.
- Step 3. But, I did not pass the best one → Try it again.

- Step 4. Finally Find the Best One.

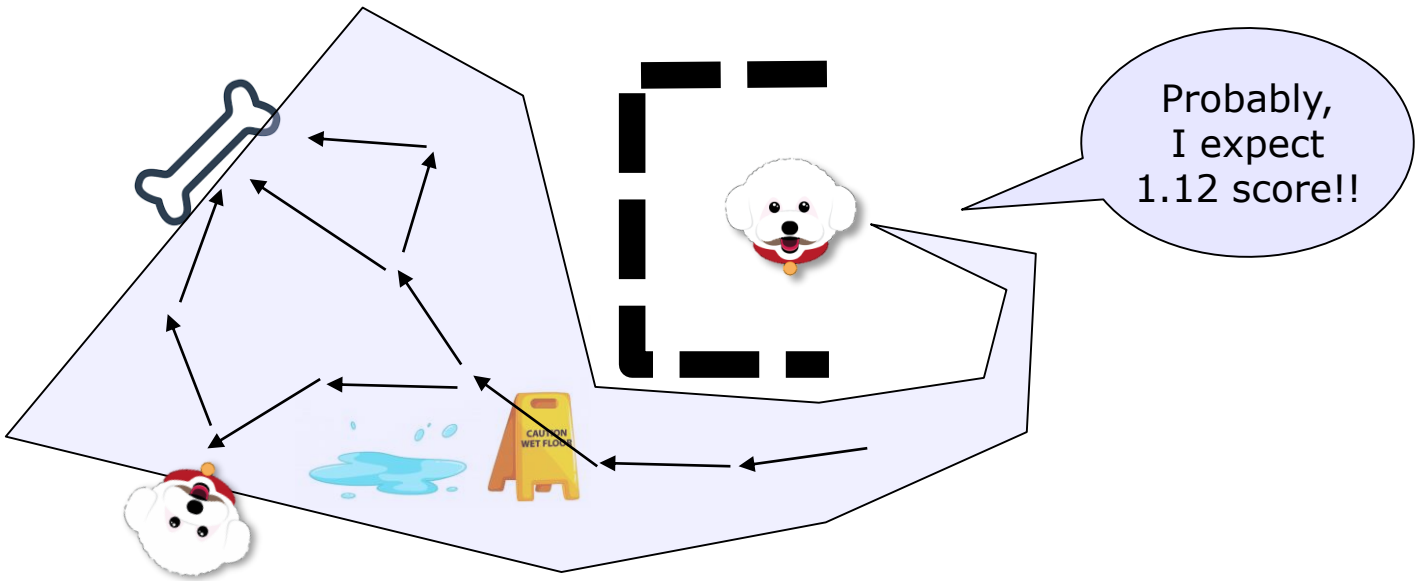
Issue. 2. **Stochastic Problem**



- An agent follows the same path.
- But Scoring is not Accurate.
  - 1st turn: the score was 1.
  - 2nd turn: the score was 1.2
  - 3rd turn: the score was 0.9



Issue. 3. **Expectation of ALL Cases**



- Expectation is the terminology in Probability
- Every events are considered as Probabilistic cases.

$$E\{R\} = \lim_{n\to\infty} \frac{1}{n} \sum_{i}^{n} R_i \left(= \sum_{i}^{\infty} R_i \, p(R_i), \; p \text{ is PDF, Not Probability} \right)$$

Expectation in Probability

- Expectation is the value that is weighted by probability

$$E(x) = \int_{-\infty}^{\infty} xp(x)dx, \quad p(x) \text{ is the Probabilistic Density Function}$$

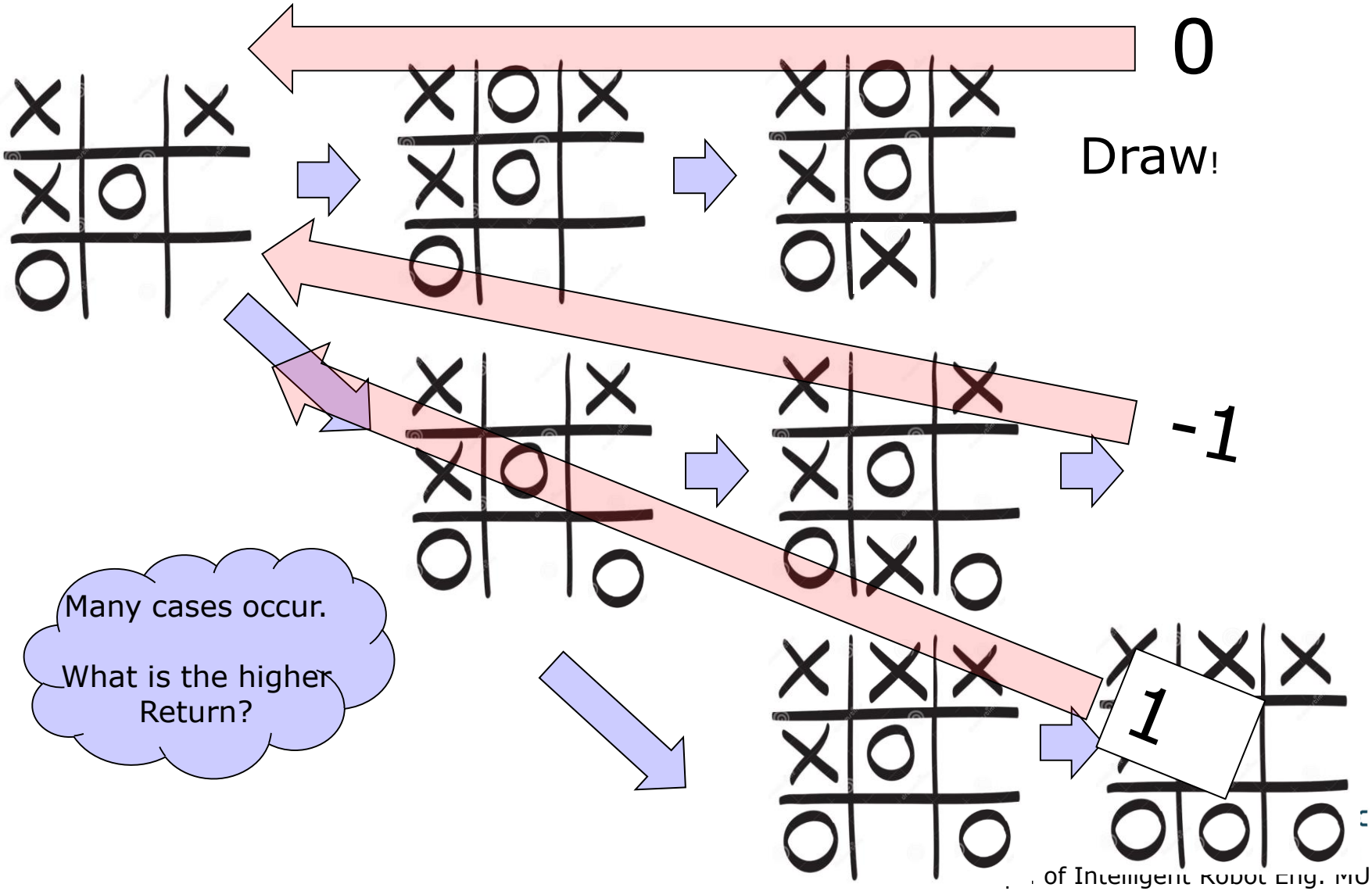
$$E(x) = \sum_i x_i p(x_i), \quad p(x) \text{ is the Probabilistic Mass Function}$$

- Expectation of rolling a dice.

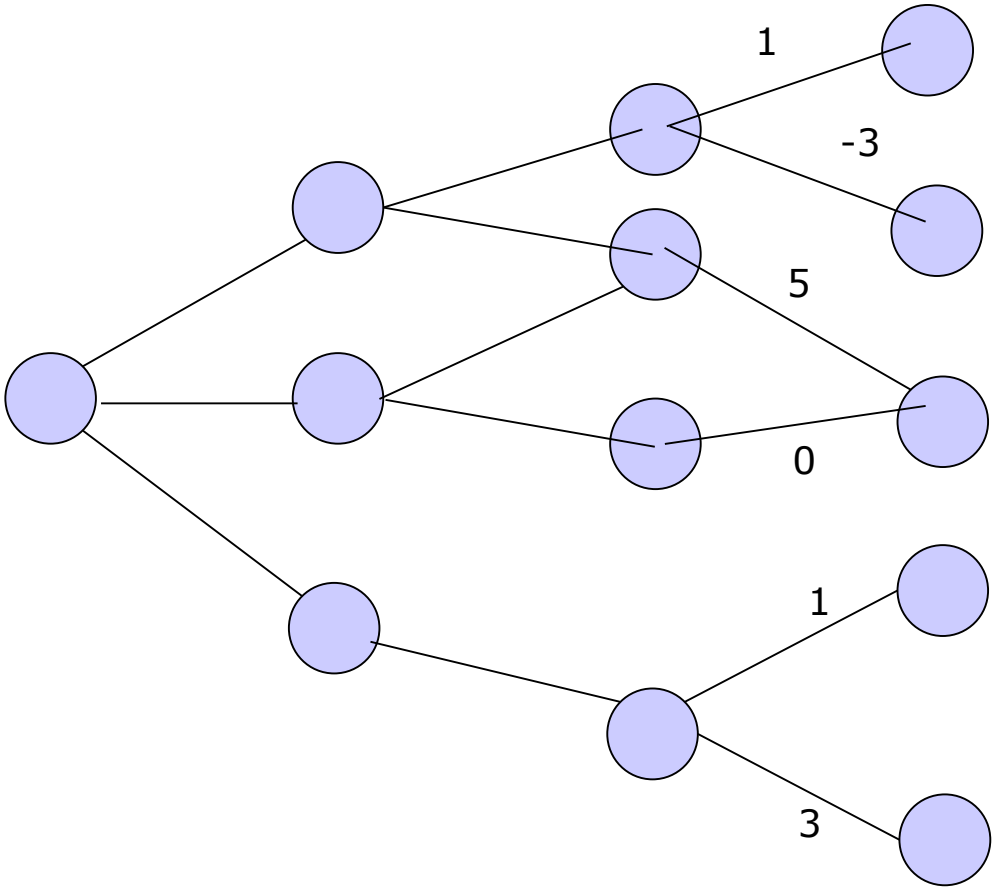
$$\text{Prob. } P(x = x_i) = p(x_i)$$

$$E\{x\} = \sum_i^6 x_i \, \mathrm{p}(x_i) = \sum_i^6 x_i \, \mathrm{P}(x = x_i) = \sum_i^6 x_i \frac{1}{6} = 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} +, \ldots + 6 \cdot \frac{1}{6} = 3.5$$

RL uses Return Value(Scores) in Future..



Scoring on every transition

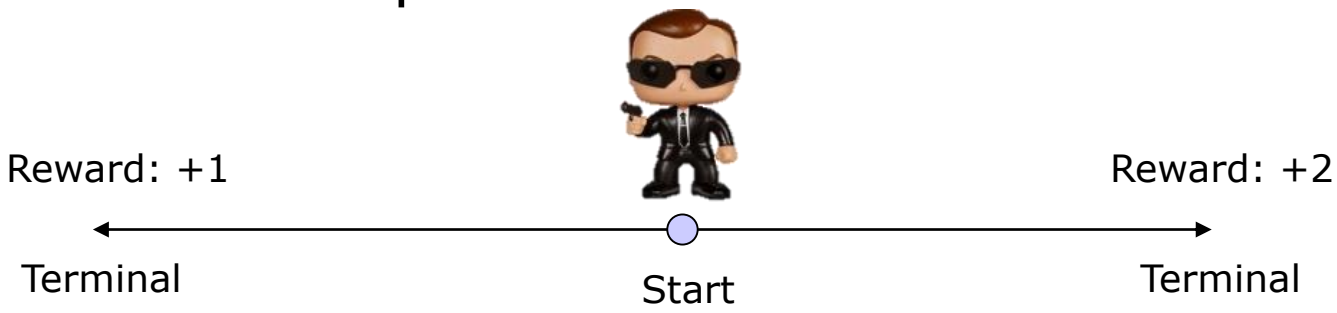


How we summarize everything?

3 Stochastic Environment

Everything is Stochastically Determined : Stochastic(Probabilistic) World

- Think the example

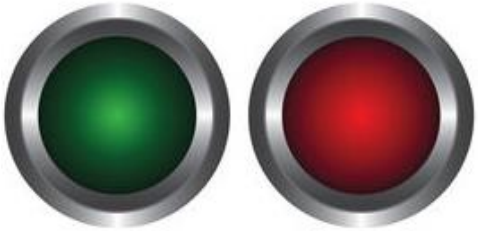


- Initial(Start) position: an agent starts its exploration.
- Terminal position: an agent ends its exploration.

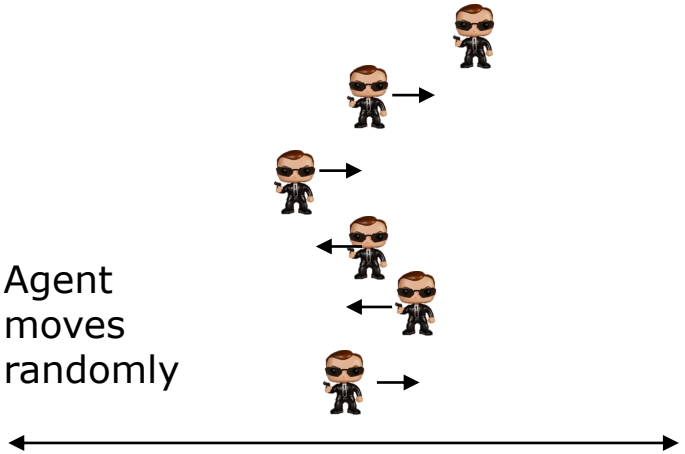
- Which way (Left or Right) is better ?
  - You answer the right turn.
  - Why? → How can you teach it to an agent

Our Agent has No Information

- An agent has two buttons



Agent does NOT know which button is left or right



- When an agent finish its exploration, it gets a Reward.

- “An agent is blind”

How it works?

Do 1000 Explorations

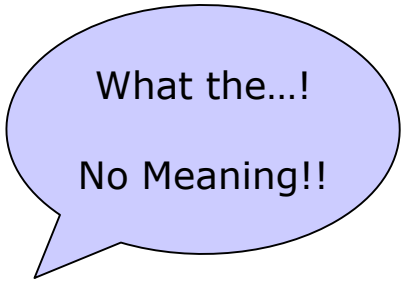
- Run l9test1.py

```
def test():
    sumLeft = 0
    sumRight= 0
    for i in range(0,1000):
        r=explore();
        if (r==1):
            sumLeft = sumLeft+1
        if (r==2):
            sumRight= sumRight+1

    print(sumLeft,sumRight)
```

```
def explore():
    global s

    # Restart exploration.
    init();

    while(True):
        # Determine action randomly
        a=randint(2)
        if (a==0):  # left
            s=s-1;
        else:
            s=s+1;  # right

        #check if s is on terminal
        if (s==0):
            r = 1;
            break;
        if (s==10):
            r=2;
            break;
    #print(s,r)
    return r;
```

Result is ..

Result	Sum of Left goals	Sum of Right goals
1	506	494
2	485	515
3	518	482
4	498	502
5	503	497
6	517	483
7	485	515



- It is just Left or Right example.
- If we do many iterations, left becomes equal to right.
  - Ex) left → 500 , right → 500, Probability is 0.5

What we Learn from this STUPID Example

- 1. if we push Red or Green button “Randomly”,
  - We can go left or right terminal
  - Thus, probability of Left and Right is

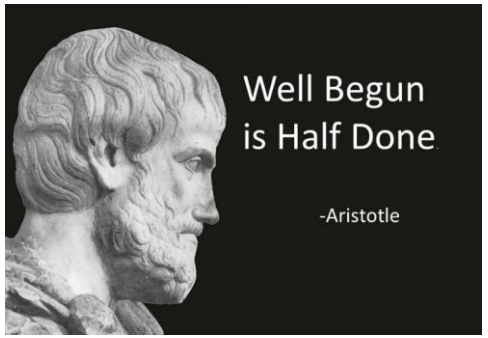
$$p(left) \cong 0.5 \quad p(right) \cong 0.5$$

- 2. However, we know that “Right” will get better reward
  - How can we express this?

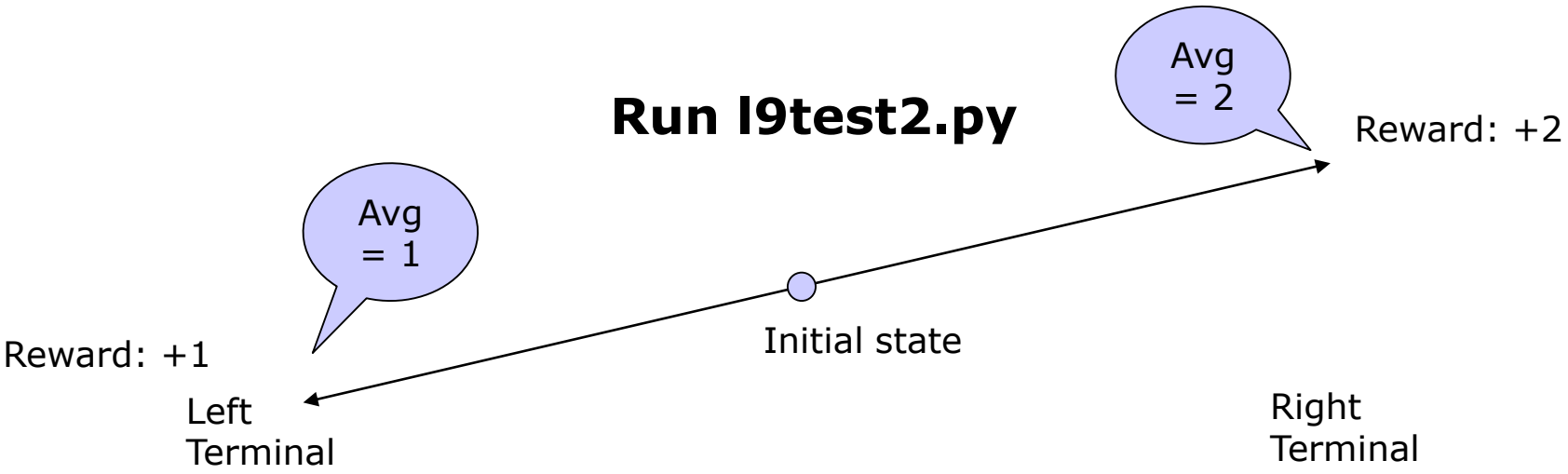
At an initial state, we expect that

$$r_{left}\, p(left) + r_{right}\, p(right)$$

$$= 1 \bullet \frac{1}{2} + 2 \bullet \frac{1}{2} = 1.5$$



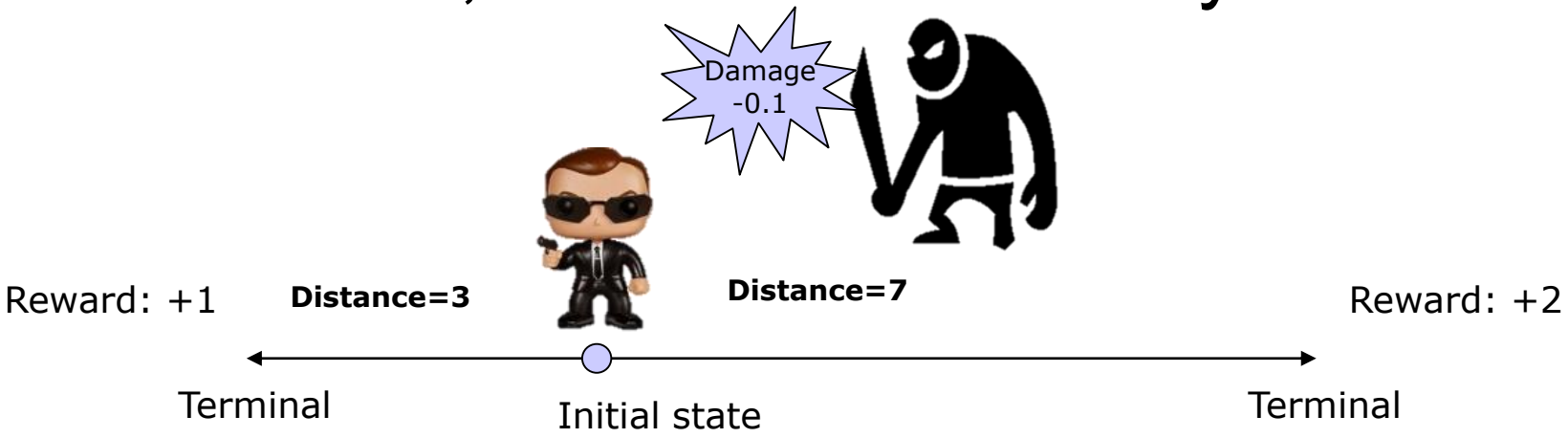
Sum of Rewards is Good for Learning



	Left Terminal	Right Terminal
Number of cases	503	497
Sum of Reward	503	994
average	503/503=1.0	994/497=2.0

Finally, I can say that Average of Reward at the RIGHT is better!

Well, World is not so easy…



- If it is a tough road, an agent receives -0.1 reward.
  - There are some Monsters as in video games.
- In each Movement, reward is -0.1…
  - Which way is better?

See l9test3.py

```
def explore():
    global s

    # Restart exploration.
    init();

    r=0;
    case =0;
    while(True):
        # Determine action randomly
        a=randint(2)
        if (a==0):  # left
            s=s-1;
        else:
            s=s+1;  # right

        r=r-0.1;

        #check if s is on terminal
        if (s==0):
            r = r+1;
            case =0;
            break;
        if (s==10):
            r= r+ 2;
            case =1;
            break;

    #print(s,r)
    return case,r;
```

Initial reward

Monster attack

Final reward at each terminal

- How it works?

```
a.test()
723 277
-0.8167358229598886 -0.8068592057761725
a.test()
711 289
-0.6769338959212378 -1.1539792387543224
```

	Left Terminal	Right Terminal
1st test	723 Avg= -0.817	277 Avg=-0.807
2nd test	711 Avg= -0.677	289 Avg=-1.154

Which one is better?

l9test3.py Test
1000 → 10000 times.

1000 times

```
a.test()
723 277
-0.8167358229598886 -0.8068592057761725
a.test()
711 289
-0.6769338959212378 -1.1539792387543224
```

- Which one is better?
- With 10000 tests,

"Left" is better.

10000 times

```
a.test()
6984 3016
-0.6861397479954281 -1.0130636604774528
a.test()
7020 2980
-0.6702564102564202 -1.0504026845637588
a.test()
6969 3031
-0.7029559477687005 -1.0480699439128982
a.test()
7013 2987
-0.6885355767859803 -1.0675929025778355
a.test()
7020 2980
-0.6943589743589857 -1.0181879194630854
a.test()
7015 2985
-0.7208410548824017 -0.9618425460636508
a.test()
6956 3044
-0.7127659574468206 -1.046911957950067
a.test()
6938 3062
-0.7210435283943618 -1.0207707380796869
a.test()
7008 2992
-0.7084189497716983 -0.9660427807486635
```

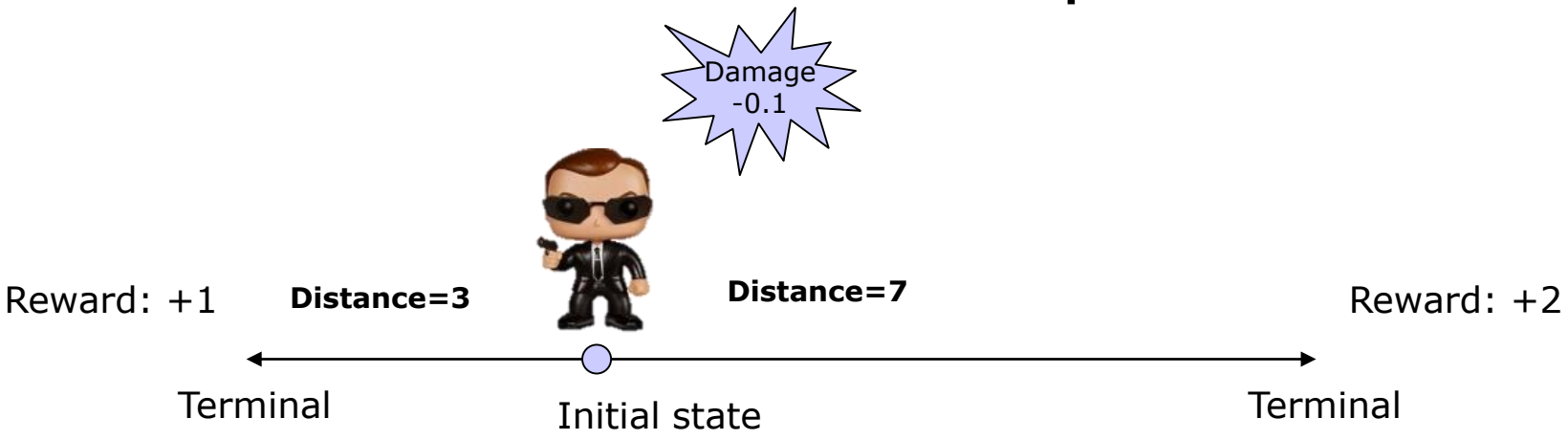
See the Result, Carefully

```
a.test()
6984 3016
-0.6861397479954281 -1.0130636604774528
a.test()
7020 2980
-0.6702564102564202 -1.0504026845637588
a.test()
6969 3031
-0.7029559477687005 -1.0480699439128982
a.test()
7013 2987
-0.6885355767859803 -1.0675929025778355
a.test()
7020 2980
-0.6943589743589857 -1.0181879194630854
a.test()
7015 2985
-0.7208410548824017 -0.9618425460636508
a.test()
6956 3044
-0.7127659574468206 -1.046911957950067
a.test()
6938 3062
-0.7210435283943618 -1.0207707380796869
a.test()
7008 2992
-0.7084189497716983 -0.9660427807486635
```

- Optimal might be
  - Left Distance is 3 and Left reward is 1.
    → 1 -3*0.1 = 0.7.
  - Right Distance is 7 and Right reward is 2.
    Then, 2- 7*0.1 = 1.3??
- Why Avg. left and right is so different?

It is a stochastic world.
An Random agent CANNOT reach at Right Terminal within 7 steps.
(Probably Not)

Extend This Concept



- Sum is Good?
  - Case 1) s=[3, 2, 1, 0] Sum of rewards= 1-3*0.1=0.7 (Best)
  - Case 2) s=[3,2,3,2,3,2,3,2,3,2,3,2,3,2,3,2,3,2,1,0] Sum = 1-19*0.1 = -0.9
- Average is Good?
  - Case 1) 3 turns → 0.7/3 = 0.233
  - Case 2) 19 turns → -0.9/19= -0.047

Summary

- Average of all rewards
  - Meaningless [3,2,3,2,3,2….] only reduces average rewards.
  - **Increasing average of all rewards is to find the optimal path**
- **Why Not Sum?**
  - **Summation is just a Cost function in deterministic ways**

$$R = J = \sum_k r_k$$

- **Why Average?**
  - **Average is an alternative expression of Expectation**
  - **Our problem is in a stochastic world → Use Probabilistic method**

$$\bar{R} = \frac{1}{N}\sum_k r_k \triangleq E\{R\} \quad \therefore optimal\ path : \max(\mathrm{E}\{\mathrm{R}\})$$