

# ปัญญาประดิษฐ์ (Artificial Intelligence)

บทบรรยายที่ 9: การเรียนรู้เสริมกำลัง  
(Reinforcement Learning)

ผศ. ดร. อิทธิพล พองแก้ว  
[ittipon@g.sut.ac.th](mailto:ittipon@g.sut.ac.th)

## วันนี้

จะตัดสินใจภายใต้ความไม่แน่นอน ไปพร้อมกับการเรียนรู้ เกี่ยวกับสภาพแวดล้อมได้อย่างไร?

- การเรียนรู้เสริมกำลัง (Reinforcement learning: RL)
- Passive RL (RL แบบพาสซีฟ)
  - การประมาณแบบมีแบบจำลอง (Model-based estimation)
  - การประมาณแบบไม่จำลอง (Model-free estimation)
  - การประมาณอัตราประโยชน์โดยตรง (Direct utility estimation)
  - การเรียนรู้แบบความต่างตามเวลา (Temporal-difference learning)
- Active RL (RL แบบแอคทีฟ)
  - การเรียนรู้แบบมีแบบจำลอง (Model-based learning)
  - Q-Learning
  - การทำให้หัวไปข้ามสถานะ (Generalizing across states)



## Known MDP: Offline Solution

LEC. 8

Goal	Technique
------	-----------

Compute  $V^*$ ,  $Q^*$ ,  $\pi^*$  Value / policy iteration

Evaluate a fixed policy  $\pi$  Policy evaluation

## Unknown MDP: Model-Based

LEC. 9

Goal	*use features to generalize	Technique
------	-----------------------------	-----------

Compute  $V^*$ ,  $Q^*$ ,  $\pi^*$  VI/PI on approx. MDP

Evaluate a fixed policy  $\pi$  PE on approx. MDP

## Unknown MDP: Model-Free

Goal	*use features to generalize	Technique
------	-----------------------------	-----------

Compute  $V^*$ ,  $Q^*$ ,  $\pi^*$  Q-learning

Evaluate a fixed policy  $\pi$  Value Learning

PASSIVE RL  
①

ACTIVE RL  
②

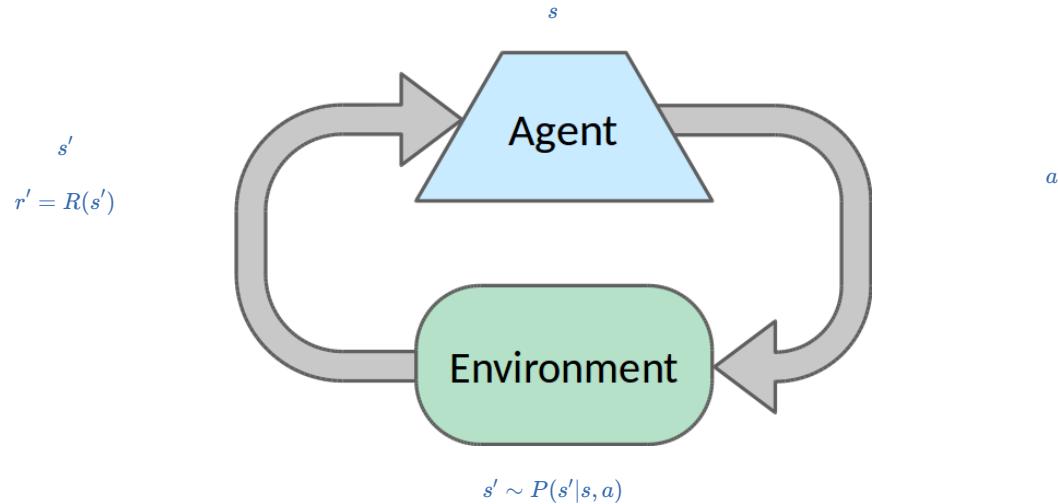
## MDPs

สรุปสั้นๆ

## MDPs

กระบวนการตัดสินใจของมาร์คอฟ (Markov decision process: MDP) คือทูเพิล  $(\mathcal{S}, \mathcal{A}, P, R)$  ที่ประกอบด้วย:

- $\mathcal{S}$  คือเซตของสถานะ  $s$ ;
- $\mathcal{A}$  คือเซตของการกระทำ  $a$ ;
- $P$  คือแบบจำลองการเปลี่ยนผ่าน (stationary) โดย  $P(s'|s, a)$  แทนความน่าจะเป็นในการไปถึงสถานะ  $s'$  เมื่อทำการกระทำ  $a$  ที่สถานะ  $s$ ;
- $R$  คือฟังก์ชันรางวัลที่ให้ค่ารางวัลทันที (จักรัด)  $R(s)$  สำหรับสถานะ  $s$ ;
- $(0 < \gamma \leq 1)$  คือตัวคูณลดค่า หรือ discount factor.)



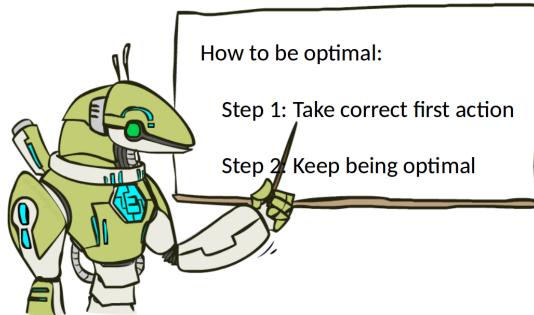
## ข้อสังเกต

- แม้ว่า MDP จะทั่วไปไปยังปริภูมิสถานะ/การกระทำต่อเนื่องได้ แต่ในบทนี้เราจะสมมติว่า  $\mathcal{S}$  และ  $\mathcal{A}$  เป็นแบบไม่ต่อเนื่องและมีจำนวนจำกัด
- รูปแบบนิยามของ MDP ไม่ได้มีแบบเดียว โดยอีกแบบที่ใช้กันแพร่หลายคือกำหนดฟังก์ชันรางวัลตามการเปลี่ยนผ่าน  $(s, a, s')$  คือ  $R(s, a, s')$  ซึ่งจะทำให้ได้รูปแบบอัลกอริทึมที่ดีกว่า (แต่เทียบเท่า) กับบทบรรยายที่ 8

## สมการเบลล์แมน (The Bellman equation)

อրรถประโภชน์ของสถานะคือรังวัลทันทีของสถานะนั้น บวกด้วยอรรถประโภชน์ที่คาดหวังแบบมีการลดค่า (discounted) ของสถานะถัดไป โดยสมมติว่าเอเจนต์เลือกการกระทำที่เหมาะสมที่สุด:

$$V(s) = R(s) + \gamma \max_a \sum_{s'} P(s'|s, a)V(s').$$



## การวนค่ามูลค่า (Value iteration)

อัลกอริทึม **Value iteration** ให้กระบวนการวนครองที่ (fixed-point iteration) เพื่อคำนวณอրรถประโยชน์ของสถานะ  $V(s)$ :

- ให้  $V_i(s)$  เป็นค่าประมาณอรรถประโยชน์ของ  $s$  ณ รอบที่  $i$ ;
- Bellman update คือการอัปเดตค่าประมาณทั้งหมดพร้อมกัน ให้สอดคล้องกับสมการเบลล์แมนในเชิงเฉพาะที่:

$$V_{i+1}(s) = R(s) + \gamma \max_a \sum_{s'} P(s'|s, a) V_i(s').$$

- ทำซ้ำจนบรรจบ

## การวนค่านโยบาย (Policy iteration)

อัลกอริทึม Policy iteration คำนวณนโยบายโดยตรง (แทนที่จะคำนวณค่าสถานะ) โดยลับสองขั้นตอนต่อไปนี้:

- ประเมินนโยบาย: เมื่อกำหนด  $\pi_i$  ให้คำนวณ  $V_i = V^{\pi_i}$  คืออรรถประโยชน์ของแต่ละสถานะเมื่อรัน  $\pi_i$ :

$$V_i(s) = R(s) + \gamma \sum_{s'} P(s'|s, \pi_i(s)) V_i(s').$$

- ปรับปรุงนโยบาย: คำนวณนโยบายใหม่  $\pi_{i+1}$  ด้วยการมองล่วงหน้า 1 ก้าวจาก  $V_i$ :

$$\pi_{i+1}(s) = \arg \max_a \sum_{s'} P(s'|s, a) V_i(s').$$

## การเรียนรู้เสริมกำลัง (Reinforcement learning)



เครดิตวิดีโอ: Megan Hayes, @YAWScience, 2020.

12 / 58



เครดิตวีดีโอ: Megan Hayes, @YAWScience, 2020.

13 / 58

## เกิดอะไรขึ้น?

- ไม่ใช่การวางแผน (planning) แต่มันคือการเรียนรู้เสริมกำลัง!
- มี MDP อญี่ แต่ไม่สามารถแก้ด้วยการคำนวณอย่างเดียว
- ไก่ต้องลงมือกระทำจริงเพื่อหาคำตอบ

## แนวคิดสำคัญใน RL ที่ pollama

- การสำรวจ (Exploration): ต้องลองกระทำการที่ไม่รู้จักเพื่อเก็บข้อมูล
- การแสวงหาผล (Exploitation): ในที่สุดต้องใช้ความรู้ที่มีให้เกิดประโยชน์
- ความเสียใจ (Regret): แม้เรียนรู้อย่างชาญฉลาด ก็อาจทำพลาดได้
- การสุ่มตัวอย่าง (Sampling): เพราะความบังเอิญ ต้องลองซ้ำๆ หลายครั้ง
- ความยก: การเรียนรู้อาจยากกว่าการแก้ MDP ที่รู้จักดีอยู่แล้วมาก

## Reinforcement learning

เรายังคงสมมติ MDP  $(\mathcal{S}, \mathcal{A}, P, R)$  ที่

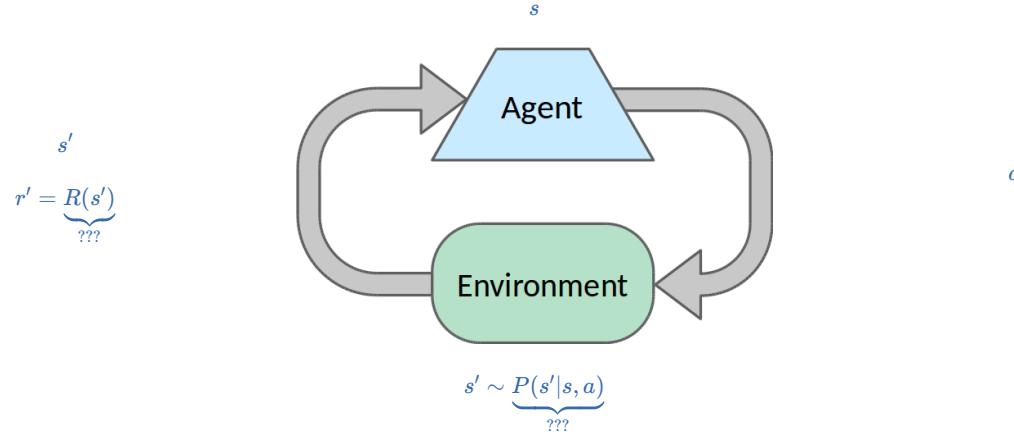
- $\mathcal{S}$  คือเซตของสถานะ  $s$ ;
- $\mathcal{A}$  คือเซตของการกระทำ  $a$ ;
- $P$  คือแบบจำลองการเปลี่ยนผ่าน (stationary) โดย  $P(s'|s, a)$  แทนความน่าจะเป็นในการไปถึง  $s'$  เมื่อทำ  $a$  ที่  $s$ ;
- $R$  คือฟังก์ชันรางวัลที่ให้รางวัลทันที  $R(s)$  ในสถานะ  $s$ .

เป้าหมายของเราคือหาโนยบายที่เหมาะสมที่สุด  $\pi^*(s)$

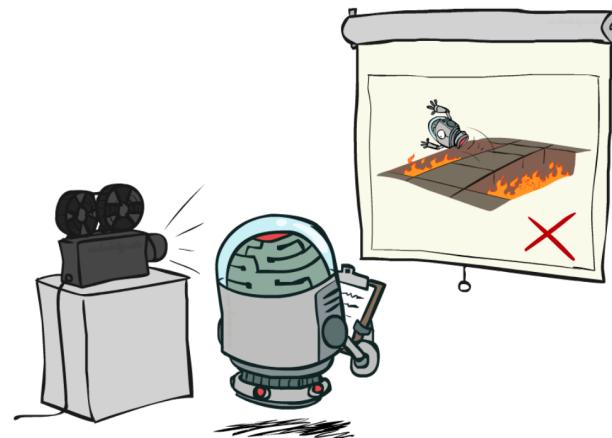
## ลูกเล่นใหม่ (New twist)

แบบจำลองการเปลี่ยนผ่าน  $P(s'|s, a)$  และพังก์ชันรางวัล  $R(s)$  เป็น สิ่งที่ไม่รู้จัก

- เราไม่รู้ว่าสถานะไหนดีและการกระทำได้ทำอะไร!
- ต้องสังเกตหรือโต้ตอบกับสภาพแวดล้อมเพื่อ เรียนรู้ ในการมิกเหล่านี้ พร้อมกับลงมือกระทำ



## Passive RL

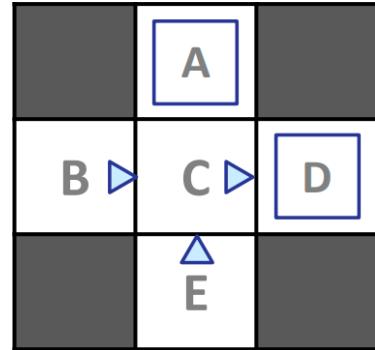


### เป้าหมาย: ประเมินนโยบาย (policy evaluation)

- นโยบายของเอเจนต์  $\pi$  ถูกตรวจด้วยว่า
- เป้าหมายคือเรียนรู้ค่าอรรถประโยชน์  $V^\pi(s)$

— ผู้เรียนไม่มีสิทธิเลือกการกระทำ แค่รับนโยบายและเรียนรู้จากประสบการณ์

เครดิตภาพ: CS188, UC Berkeley.



ເອເຈນຕີວັນ ກາຣທດລອງ (trials/episodes) ລາຍກວັງດ້ວຍນິໂຍບາຍ  $\pi$ . ລຳດັບກາຣເດີນ (trajectory)  $(s, r, a, s'), (s', r', a', s'')$ , ... ອາຈເປັນເຂັ້ນນີ້:

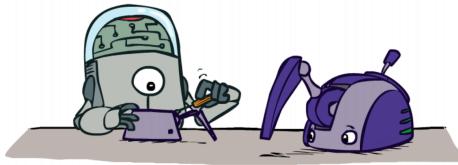
- Trial 1:  $(B, -1, \text{east}, C), (C, -1, \text{east}, D), (D, +10, \text{exit}, \perp)$
- Trial 2:  $(B, -1, \text{east}, C), (C, -1, \text{east}, D), (D, +10, \text{exit}, \perp)$
- Trial 3:  $(E, -1, \text{north}, C), (C, -1, \text{east}, D), (D, +10, \text{exit}, \perp)$
- Trial 4:  $(E, -1, \text{north}, C), (C, -1, \text{east}, A), (A, -10, \text{exit}, \perp)$

## การประมาณแบบมีแบบจำลอง (Model-based estimation)

เอเจนต์แบบ Model-based จะประมาณแบบจำลองการเปลี่ยนผ่านและรางวัล  $\hat{P}$  และ  $\hat{R}$  จากประสบการณ์ แล้วประเมิน MDP เชิงประจักษ์ที่ได้

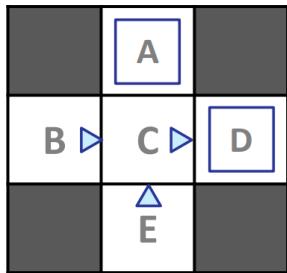
- ขั้นที่ 1: เรียนรู้ MDP เชิงประจักษ์
  - ประมาณ  $\hat{P}(s'|s, a)$  จากตัวอย่างเชิงประจักษ์  $(s, a, s')$  หรือด้วยการเรียนรู้แบบมีผู้สอน
  - ค้นหารางวัล  $\hat{R}(s)$  ของแต่ละ  $s$
- ขั้นที่ 2: ประเมิน  $\pi$  ด้วย  $\hat{P}$  และ  $\hat{R}$  เช่น

$$V(s) = \hat{R}(s) + \gamma \sum_{s'} \hat{P}(s'|s, \pi(s)) V(s').$$



## ตัวอย่าง

นโยบาย  $\pi$ :



Trajectories:

- $(B, -1, \text{east}, C), (C, -1, \text{east}, D), (D, +10, \text{exit}, \perp)$
- $(B, -1, \text{east}, C), (C, -1, \text{east}, D), (D, +10, \text{exit}, \perp)$
- $(E, -1, \text{north}, C), (C, -1, \text{east}, D), (D, +10, \text{exit}, \perp)$
- $(E, -1, \text{north}, C), (C, -1, \text{east}, A), (A, -10, \text{exit}, \perp)$

แบบจำลองการเปลี่ยนผ่านที่เรียนรู้ได้  $\hat{P}$ :

$$\hat{P}(C|B, \text{east}) = 1$$

$$\hat{P}(D|C, \text{east}) = 0.75$$

$$\hat{P}(A|C, \text{east}) = 0.25$$

(...)

รางวัลที่เรียนรู้ได้  $\hat{R}$ :

$$\hat{R}(B) = -1$$

$$\hat{R}(C) = -1$$

$$\hat{R}(D) = +10$$

(...)

## การประมาณแบบไร้แบบจำลอง (Model-free estimation)

เราจะเรียนรู้  $V^\pi$  แบบ ไร้แบบจำลอง ได้ไหม โดยไม่ต้องสร้างแบบจำลองสิ่งแวดล้อมชัดๆ เช่น ไม่ต้องเรียนรู้  $P$  และ  $R$ ?

## การประมาณอัตราประโยชน์โดยตรง (Direct utility estimation)

(หรือ Monte Carlo evaluation)

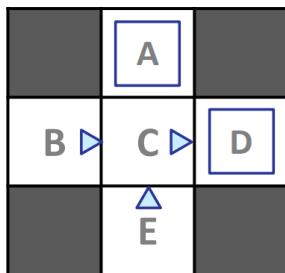
- อัตราประโยชน์  $V^\pi(s)$  ของสถานะ  $s$  คือร่วงรวมที่คาดหวังจากสถานะนั้นไปข้างหน้า (เรียกว่า reward-to-go ที่คาดหวัง)

$$V^\pi(s) = \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t) \right] \Big|_{s_0=s}$$

- แต่ละการทดลองให้ตัวอย่างของปริมาณนี้สำหรับสถานะที่ถูกเกี่ยม
- ดังนั้น เมื่อจบลำดับหนึ่งๆ เราสามารถอัปเดตค่าเฉลี่ยตัวอย่าง  $\hat{V}^\pi(s)$  โดย:
  - คำนวณ reward-to-go ที่สังเกตสำหรับแต่ละสถานะ
  - อัปเดตค่าประมาณอัตราประโยชน์ของสถานะนั้นด้วยค่าเฉลี่ยสะสม
- เมื่อจำนวนการทดลองไปสู่อนันต์ ค่าเฉลี่ยตัวอย่างจะบรรจบสู่ค่าคาดหมายจริง

## ตัวอย่าง ( $\gamma = 1$ )

นโยบาย  $\pi$ :



Trajectories:

- $(B, -1, \text{east}, C), (C, -1, \text{east}, D), (D, +10, \text{exit}, \perp)$
- $(B, -1, \text{east}, C), (C, -1, \text{east}, D), (D, +10, \text{exit}, \perp)$
- $(E, -1, \text{north}, C), (C, -1, \text{east}, D), (D, +10, \text{exit}, \perp)$
- $(E, -1, \text{north}, C), (C, -1, \text{east}, A), (A, -10, \text{exit}, \perp)$

ค่าเอาเด็พดูต  $\hat{V}^\pi(s)$ :

	-10 A	
+8 B	+4 C	+10 D
	-2 E	

ถ้า  $B$  และ  $E$  ไป  $C$  ภายใน  $\pi$  ทำไม่ค่าของพากมันถึงต่างกันได้?

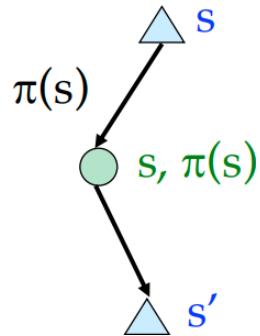
น่าเลียดายที่การประมาณโดยตรงจะเสียเวลาและไม่ได้อิสระต่อ กัน เพราะต้องเป็นไปตามสมการเบลล์แมนเมื่อกำหนดนโยบายคงที่:

$$V^\pi(s) = R(s) + \gamma \sum_{s'} P(s'|s, \pi(s)) V^\pi(s').$$

ดังนั้นจึงพลาดโอกาสในการเรียนรู้และต้องใช้เวลานานกว่าจะเรียนรู้ได้ดี

## การเรียนรู้แบบความต่างตามเวลา (Temporal-difference learning)

TD learning อัปเดต  $V^\pi(s)$  ทุกครั้งที่ເອເຈນຕໍ່ປະສົບກາຮປ່ລືຍັນຜ່ານ  $(s, r = R(s), a = \pi(s), s')$



ເນື່ອເກີດກາຮປ່ລືຍັນຈາກ  $s$  ໃປ  $s'$  ກາຮອັບແດຕແບບ TD ຈະພຍາຍາມດັນ  $V^\pi(s)$  ໄທສອດຄລ້ອງກັບສມກາຮບ່ລື້ແມນຂອງນ ໂຍບາຍຄົງທີ່ດີ  
ຂຶ້ນ ອື່ອ

$$V^\pi(s) \leftarrow V^\pi(s) + \alpha \underbrace{(r + \gamma V^\pi(s') - V^\pi(s))}_{\text{temporal difference error}}$$

ໂດຍທີ່  $\alpha$  ດີວຽກຮຸມເມີເຕອຮ໌ ອັດຮາກາຮເຮືອນຮູ້ (learning rate)

อีกมุมหนึ่ง การอัปเดต TD มองได้ว่าเป็นก้าวของเกรเดียนต์ดิเชนต์บนความคลาดเคลื่อนกำลังสองระหว่างเป้า  $r + \gamma V^\pi(s')$  กับค่าพยากรณ์  $V^\pi(s)$  (รายละเอียดภายหลัง)

## ค่าเฉลี่ยเคลื่อนที่แบบอัปเดต TD

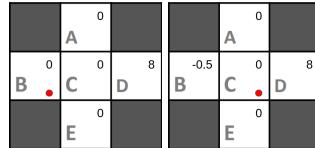
การอัปเดต TD เชียนใหม่ได้เป็นค่าเฉลี่ยเคลื่อนที่แบบอัปเดต TD

$$V^\pi(s) \leftarrow (1 - \alpha)V^\pi(s) + \alpha(r + \gamma V^\pi(s')).$$

โดยลัญชาตญาณ

- ให้ความสำคัญกับตัวอย่างล่าสุดมากขึ้น
- ค่อนข้างล้าหลัง (ซึ่งมักประเมินผิดอยู่ดี)

ตัวอย่าง ( $\gamma = 1$ ,  $\alpha = 0.5$ )



การเปลี่ยนผ่าน:  $(B, -1, \text{east}, C)$

TD-update:

$$V^\pi(B) \leftarrow V^\pi(B) + \alpha(R(B) + \gamma V^\pi(C) - V^\pi(B)) \quad \leftarrow 0 + 0.5(-1 + 0 - 0) \leftarrow -0.5$$

	A	0	
-0.5	C	0	D
B		8	
E	0		

	A	0	
-0.5	C	3.5	D
B		8	
E	0		

การเปลี่ยนผ่าน:  $(C, -1, \text{east}, D)$

TD-update:

$$V^\pi(C) \leftarrow V^\pi(C) + \alpha(R(C) + \gamma V^\pi(D) - V^\pi(C)) \quad \leftarrow 0 + 0.5(-1 + 8 - 0) \leftarrow 3.5$$

## การบรรจบ (Convergence)

- สังเกตว่าการอัปเดต TD ใช้เพียงผู้สืบทอดที่สังเกตได้  $s'$  ขณะที่สมการเบลล์แมนจริงสำหรับนโยบายคงที่พิจารณาสถานะถัดไปทุกตัว แต่ค่าเฉลี่ยของ  $V^\pi(s)$  จะบรรจบสู่ค่าที่ถูกต้อง
- หากเปลี่ยน  $\alpha$  จากค่าคงที่เป็นฟังก์ชันที่ลดลงตามจำนวนครั้งที่สถานะนั้นถูกเยี่ยม  $V^\pi(s)$  เองก็จะบรรจบสู่ค่าที่ถูกต้อง

## Active RL



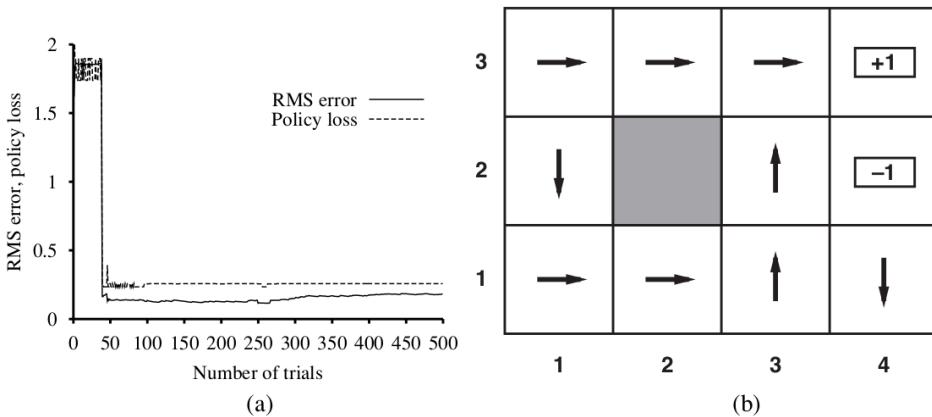
เป้าหมาย: เรียนรู้นโยบายที่เหมาะสมที่สุด

- นโยบายของเอเจนต์ไม่ถูกตรึงอีกต่อไป
- เป้าหมายคือเรียนรู้  $\pi^*$  หรือค่าอรรถประโยชน์สถานะ  $V(s)$
- ผู้เรียนเป็นคนเลือก!
- สังคมรุ่มพันธุ์: การสำรวจ vs การแสวงหาผล

## การเรียนรู้แบบมีแบบจำลอง (Model-based learning)

เอเจนต์แบบ model-based ในโหมด passive ทำให้เป็น active ได้โดยการหานโยบายเหมาะสมที่สุด  $\pi^*$  ของ MDP เชิงประจักษ์แทน เช่น เมื่อได้ฟังก์ชันอรรถประโยชน์  $V$  ที่เหมาะสมที่สุดสำหรับแบบจำลองที่เรียนรู้มา (เช่น ด้วย Value Iteration) การกระทำที่เหมาะสมที่สุดจากการมองล่วงหน้า 1 ก้าว คือ

$$\pi^*(s) = \arg \max_a \sum_{s'} \hat{P}(s'|s, a)V(s').$$



**Figure 21.6** Performance of a greedy ADP agent that executes the action recommended by the optimal policy for the learned model. (a) RMS error in the utility estimates averaged over the nine nonterminal squares. (b) The suboptimal policy to which the greedy agent converges in this particular sequence of trials.

เอเจนต์ **ไม่ได้** เรียนรู้ค่าอัตราผลประโยชน์ที่แท้จริงหรือ policy ที่แท้จริงที่เหมาะสมที่สุด!

นโยบายที่ได้จะเป็น **โลก (greedy)** และ **ต่ำกว่าที่สุด (suboptimal)**:

- แบบจำลองการเปลี่ยนผ่านและรางวัลที่เรียนรู้  $\hat{P}$  และ  $\hat{R}$  ไม่ตรงกับโลกริง เพราะอาศัยตัวอย่างที่ได้จากนโยบายของเอเจนต์เอง ซึ่งมีอคติ
- ดังนั้นสิ่งที่หมายหัวใจแบบจำลองที่เรียนรู้อาจต่ำกว่าที่สุดในโลกริง

## การสำรวจ (Exploration)

การกระทำไม่ได้แค่ให้รางวัลตามแบบจำลองที่เรียนรู้ในปัจจุบัน แต่ยังช่วยให้เรียนรู้โลกจริงด้วย

นิคิอสคราม **แสวงหาผล-สำรวจ** (exploitation-exploration):

- แสวงหาผล: ทำการกระทำเพื่อเพิ่มรางวัลสูงสุดภายใต้แบบจำลองปัจจุบัน
- สำรวจ: เลือกทำการกระทำเพื่อสำรวจและเรียนรู้โลกจริง



## จะสำรวจอย่างไร?

วิธีง่ายสุดเพื่อบังคับการสำรวจ: การกระทำแบบสุ่ม ( $\epsilon$ -greedy)

- ด้วยความน่าจะเป็น (เล็ก)  $\epsilon$  กระทำการแบบสุ่ม
- ด้วยความน่าจะเป็น (ใหญ่)  $(1 - \epsilon)$  ตามนโยบายปัจจุบัน

$\epsilon$ -greedy จะสำรวจครบทั่วทุกช่อง “ขัดไปมา” แม้เรียนรู้แล้วก็ตาม

## ควรสำรวจเมื่อใด?

ไอเดียที่ดีกว่า: สำรวจริเวณที่ยังไม่แน่ชัดว่าແຍ້ แล้วหยุดสำรวจเมื่อมั่นใจ

ให้  $V^+(s)$  เป็นค่าประมาณแบบมองโลกในแง่ดีของอัตราประโยชน์สถานะ  $s$  และ  $N(s, a)$  คือจำนวนครั้งที่เคยลองการกระทำ  $a$  ใน  $s$

สำหรับ Value Iteration สมการอัปเดตกลายเป็น

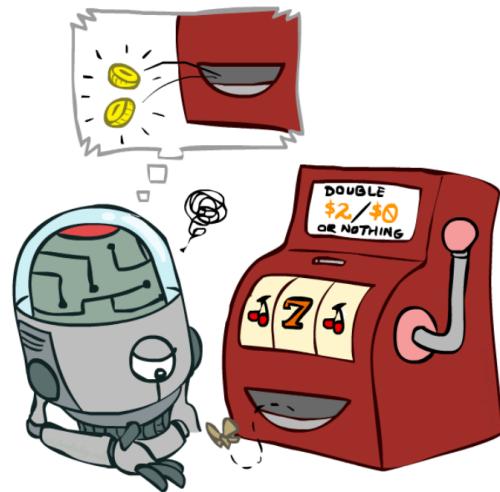
$$V_{i+1}^+(s) = R(s) + \gamma \max_a f\left(\sum_{s'} P(s'|s, a) V_i^+(s'), N(s, a)\right),$$

โดยที่  $f(v, n)$  เรียกว่า exploration function

ฟังก์ชัน  $f(v, n)$  ควรเพิ่มเมื่อ  $v$  มากขึ้น และลดเมื่อ  $n$  มากขึ้น ตัวเลือกง่ายๆ คือ  $f(v, n) = v + K/n$

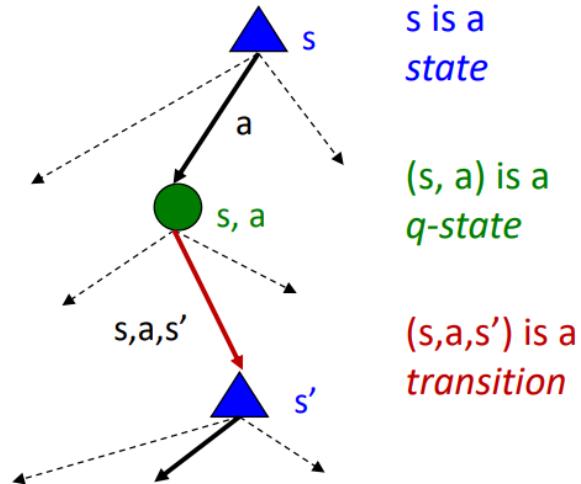
## การเรียนรู้แบบไร้แบบจำลอง (Model-free learning)

แม้ TD learning จะให้วิธีประมาณ  $V^\pi$  แบบไร้แบบจำลอง แต่ยังต้องเรียนรู้แบบจำลอง  $P(s'|s, a)$  เพื่อเลือกการกระทำด้วย one-step look-ahead



## ทางอ้อม: ค่า Q (Détour: Q-values)

- ค่าของสถานะ  $V(s)$  คือปริมาณที่คาดหวังเมื่อเริ่มที่  $s$  และกระทำอย่างเหมาะสมที่สุด
- ค่าของคู่สถานะ-การกระทำ  $Q(s, a)$  คือปริมาณที่คาดหวังเมื่อเริ่มจากทำการกระทำ  $a$  ที่  $s$  และหลังจากนั้นกระทำอย่างเหมาะสมที่สุด ดังนั้น  $V(s) = \max_a Q(s, a)$



## นโยบายที่เหมาะสมที่สุด (Optimal policy)

นิยามนโยบายที่เหมาะสมที่สุด  $\pi(s)$  ได้ทั้งในแง่ของ  $V(s)$  หรือ  $Q(s, a)$ :

$$\pi(s) = \arg \max_a \sum_{s'} P(s'|s, a)V(s') = \arg \max_a Q(s, a)$$

### สมการเบล์แมนสำหรับ $Q$

เพราะ  $V(s) = \max_a Q(s, a)$  ค่าของ  $Q(s, a)$  นิยามเวียนเกิดได้เป็น

$$Q(s, a) = R(s) + \gamma \sum_{s'} P(s'|s, a)V(s') = R(s) + \gamma \sum_{s'} P(s'|s, a) \max_{a'} Q(s', a').$$

เช่นเดียวกับ Value Iteration สมการสุดท้าย ใช้เป็นสมการอัปเดตเพื่อวนหาจุดคงที่ของ  $Q(s, a)$  ได้ แต่ยังต้องรู้  $P(s'|s, a)$  อีกดี!

## Q-Learning

ค่า  $Q(s, a)$  เรียนรู้ได้แบบไร้แบบจำลองด้วยวิธี TD ที่เรียกว่า Q-Learning

Q-Learning จะอัปเดต  $Q(s, a)$  ทุกครั้งที่เอเจนต์ประสบการณ์เปลี่ยนผ่าน  $(s, r = R(s), a, s')$

สมการอัปเดตของ TD Q-Learning คือ

$$Q(s, a) \leftarrow Q(s, a) + \alpha(r + \gamma \max_{a'} Q(s', a') - Q(s, a)).$$

เพราะ  $\pi^*(s) = \arg \max_a Q(s, a)$  ตัวแทนแบบ TD ที่เรียนรู้ค่า Q จึงไม่ต้องมีแบบจำลอง  $P(s'|s, a)$  ไม่ว่าจะเพื่อการเรียนรู้ หรือเพื่อเลือกการกระทำ!

```

function Q-LEARNING-AGENT(percept) returns an action
  inputs: percept, a percept indicating the current state  $s'$  and reward signal  $r'$ 
  persistent:  $Q$ , a table of action values indexed by state and action, initially zero
     $N_{sa}$ , a table of frequencies for state-action pairs, initially zero
     $s, a, r$ , the previous state, action, and reward, initially null

  if TERMINAL?( $s$ ) then  $Q[s, \text{None}] \leftarrow r'$ 
  if  $s$  is not null then
    increment  $N_{sa}[s, a]$ 
     $Q[s, a] \leftarrow Q[s, a] + \alpha(N_{sa}[s, a])(r + \gamma \max_{a'} Q[s', a'] - Q[s, a])$ 
     $s, a, r \leftarrow s', \text{argmax}_{a'} f(Q[s', a'], N_{sa}[s', a']), r'$ 
  return  $a$ 

```

---

**Figure 21.8** An exploratory Q-learning agent. It is an active learner that learns the value  $Q(s, a)$  of each action in each situation. It uses the same exploration function  $f$  as the exploratory ADP agent, but avoids having to learn the transition model because the Q-value of a state can be related directly to those of its neighbors.



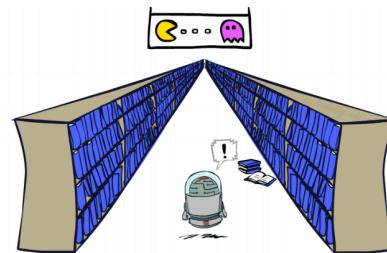
## การบรรจบ (Convergence)

Q-Learning บรรจบสู่นโยบายที่เหมาะสมที่สุด แม้ขั้นตอนจะทำแบบไม่เหมาะสมที่สุด

- สิ่งนี้เรียกว่า off-policy learning
- เนื่องจากทางเทคนิค:
  - ต้องสำรวจให้เพียงพอ
  - อัตราการเรียนรู้ต้องเล็กลงในที่สุด
  - ... แต่ยังคงเริ่มต้นไป

## การทำให้ทั่วไปข้ามสถานะ (Generalizing across states)

- Q-Learning แบบพื้นฐานเก็บตารางสำหรับทุกค่า  $Q(s, a)$
- ในสถานการณ์จริง เราไม่สามารถเรียนรู้ทุกสถานะได้!
  - สถานะมากเกินไปที่จะเขียนลงในช่วงฝึก
  - สถานะมากเกินไปที่จะเก็บตาราง Q ไว้ในหน่วยความจำ
- เราต้องการการทำให้ทั่วไป:
  - เรียนรู้จากสถานะฝึกจำนวนน้อย
  - เมย়েพรรประสนการณ์ไปยังสถานการณ์ใหม่ที่คล้ายกัน
  - นักลับไปเป็นการเรียนรู้แบบมีผู้สอนอีกครั้ง!

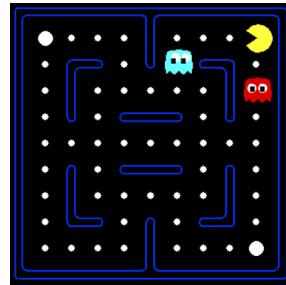


## ตัวอย่าง: Pacman

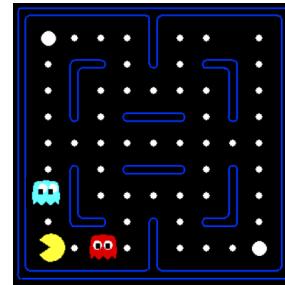
(a)



(b)



(c)

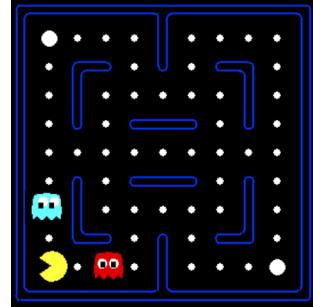


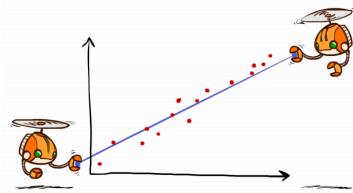
ถ้าเรารู้จักการประสบการณ์ว่า (a) แยก ใน Q-Learning แบบง่าย เราจะไม่รู้อะไรเกี่ยวกับ (b) หรือ (c) เลย!

## การแทนด้วยคุณลักษณะ (Feature-based representations)

วิธีแก้: อธิบายสถานะ  $s$  ด้วยเวกเตอร์คุณลักษณะ  $\mathbf{x} = [f_1(s), \dots, f_d(s)] \in \mathbb{R}^d$

- คุณลักษณะเป็นฟังก์ชัน  $f_k$  จากสถานะไปยังจำนวนจริงที่จับคุณสมบัติสำคัญของสถานะ
- ตัวอย่างคุณลักษณะ:
  - ระยะทางถึงผิดก้าลที่ใกล้ที่สุด
  - ระยะทางถึงจุด (dot) ที่ใกล้ที่สุด
  - จำนวนผี
  - ...
- อธิบาย q-state  $(s, a)$  ด้วยคุณลักษณะ  $f_k(s, a)$  ได้เช่นกัน





## Q-Learning แบบประมาณค่า (Approximate Q-Learning)

เมื่อใช้การแทนด้วยคุณลักษณะ ตาราง Q สามารถแทนด้วยตัวประมาณพังก์ชัน เช่น แบบจำลองเชิงเส้น

$$Q(s, a) = w_1 f_1(s, a) + w_2 f_2(s, a) + \dots + w_d f_d(s, a).$$

เมื่อเกิดการเปลี่ยนผ่าน  $(s, r, a, s')$  การอัปเดตกลายเป็น

$$w_k \leftarrow w_k + \alpha(r + \gamma \max_{a'} Q(s', a') - Q(s, a)) f_k(s, a),$$

สำหรับทุก  $w_k$

จำได้ไหมว่า TD-update มองเป็นการอัปเดต GD ออนไลน์ได้ แต่ตอนนี้เรามีปรับ  $Q(s, a)$  โดยตรง เราปรับพารามิเตอร์ของตัวประมาณพังก์ชันแทน

ในรีกรรมชั้นเชิงเส้น สมมติว่ามีจุดเดียว  $\mathbf{x}$  ที่มีคุณลักษณะ  $[f_1, \dots, f_d]$  จะได้

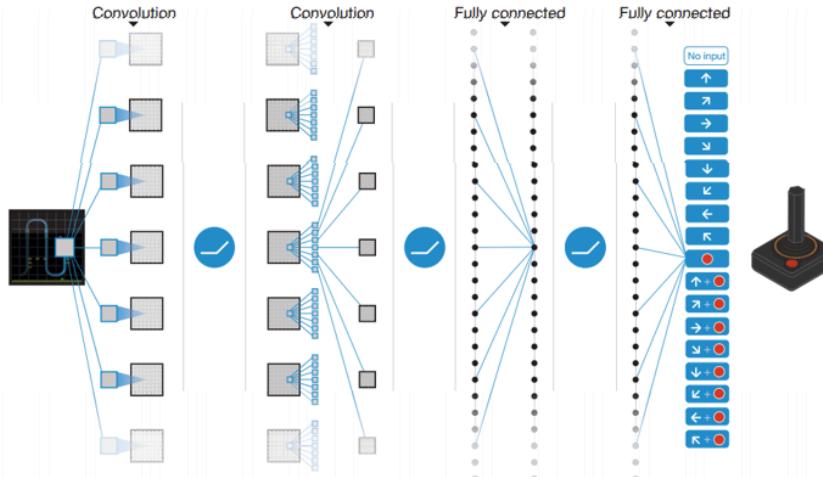
$$\ell(\mathbf{w}) = \frac{1}{2} \left( y - \sum_k w_k f_k \right)^2 \frac{\partial \ell}{\partial w_k} = - \left( y - \sum_k w_k f_k \right) f_k \quad w_k \leftarrow w_k + \alpha \left( y - \sum_k w_k f_k \right) f_k,$$

จึงได้การอัปเดต Q

$$w_k \leftarrow w_k + \alpha \left( \underbrace{r + \gamma \max_{a'} Q(s', a') - Q(s, a)}_{\text{target } y} - \underbrace{Q(s, a)}_{\text{prediction}} \right) f_k(s, a).$$

## DQN

เช่นเดียวกัน ตาราง Q สามารถแทนด้วย โครงข่ายประสาทเทียมเป็นตัวประมาณฟังก์ชันได้ กล่าวเป็นอัลกอริทึม DQN



**Figure 1 | Schematic illustration of the convolutional neural network.** The details of the architecture are explained in the Methods. The input to the neural network consists of an  $84 \times 84 \times 4$  image produced by the preprocessing map  $\phi$ , followed by three convolutional layers (note: snaking blue line

symbolizes sliding of each filter across input image) and two fully connected layers with a single output for each valid action. Each hidden layer is followed by a rectifier nonlinearity (that is,  $\max(0, x)$ ).

(ສາທິປະໄຕ)

## การประยุกต์ใช้งาน (Applications)

MarIQ -- Q-Learning Neural Network for Mario Kart – 2M Sub Special



MarIQ

Deep Q Network learning to play Video Pinball



เล่นเกม Atari (Pinball)

QT-Opt: Scalable Deep Reinforcement Learning for Vision-Based Rob...



การหยິນຈັບດ້ວຍຫຸ່ນຍົນຕໍ່ (Robotic manipulation)

Champion-level Drone Racing using Deep Reinforcement Learning (N...



การแข่งโดรน (Drone racing)

## สรุป (Summary)

Known MDP: Offline Solution *LEC. 8*

Goal	Technique
Compute $V^*$ , $Q^*$ , $\pi^*$	Value / policy iteration
Evaluate a fixed policy $\pi$	Policy evaluation

Unknown MDP: Model-Based

Goal	*use features to generalize	Technique
Compute $V^*$ , $Q^*$ , $\pi^*$		VI/PI on approx. MDP
Evaluate a fixed policy $\pi$		PE on approx. MDP

Unknown MDP: Model-Free

Goal	*use features to generalize	Technique
Compute $V^*$ , $Q^*$ , $\pi^*$		Q-learning
Evaluate a fixed policy $\pi$		Value Learning

PASSIVE RL *①* → ACTIVE RL *②*

## การกิจของฉัน ✓

เมื่อจบรายวิชานี้ คุณจะสร้างເອເຈນດົກສະໝັກທີ່ຕັດສິນ ໄຈໄດ້ວ່າຍ່າງມີປະສິກົນກົມພືບໃນສກາພແວດລ້ອມທີ່ຮູ້ຂ້ອມມຸລຄຽນຄ້ວານ ຮູ້ບາງສ່ວນ ແລະ ມີປະຫຼິບປັກຊີ່ເອເຈນດົກທີ່ຂອງคຸນຈະອຸນໜານໃນໂລກທີ່ໄມ່ແນ່ນອນແລະໄມ່ຮູ້ຈັກ ແລະເພີ່ມປະສິກົນກົມພືບໃນສກາພກຮະກະທຳສໍາຫຼັບ ໂຄງສ້າງຮາງວັດຕາມທີ່ກຳທັນດໄ້

จบการนำเสนอ