

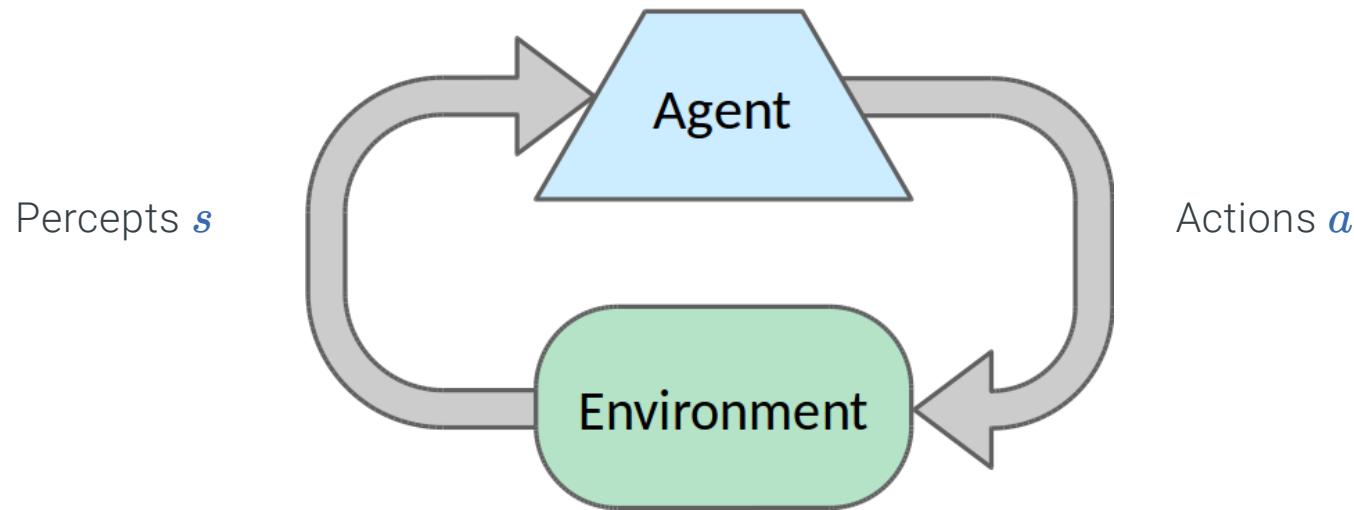
# Introduction to Artificial Intelligence

Lecture 1: Intelligent agents

Ittipon Fongkaew

# Intelligent agents

# Agents and environments



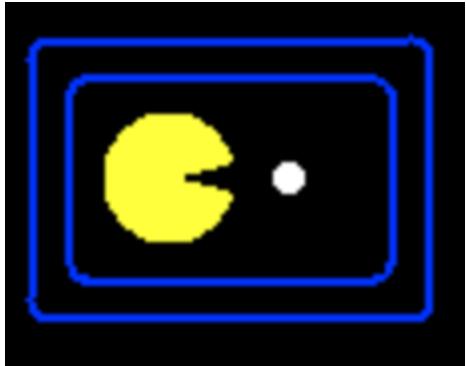
# Agents

- Agent คือสิ่งที่ รับรู้สภาพแวดล้อมของมันผ่านทาง sensor และ ทำ *action* ผ่านทาง actuator
- พฤติกรรมของ agent อธิบายได้ด้วย policy ซึ่งเป็นฟังก์ชัน

$$\pi : \mathcal{P}^* \rightarrow \mathcal{A}$$

ที่แปลงลำดับ percept ไปยัง action

## Simplified Pacman world



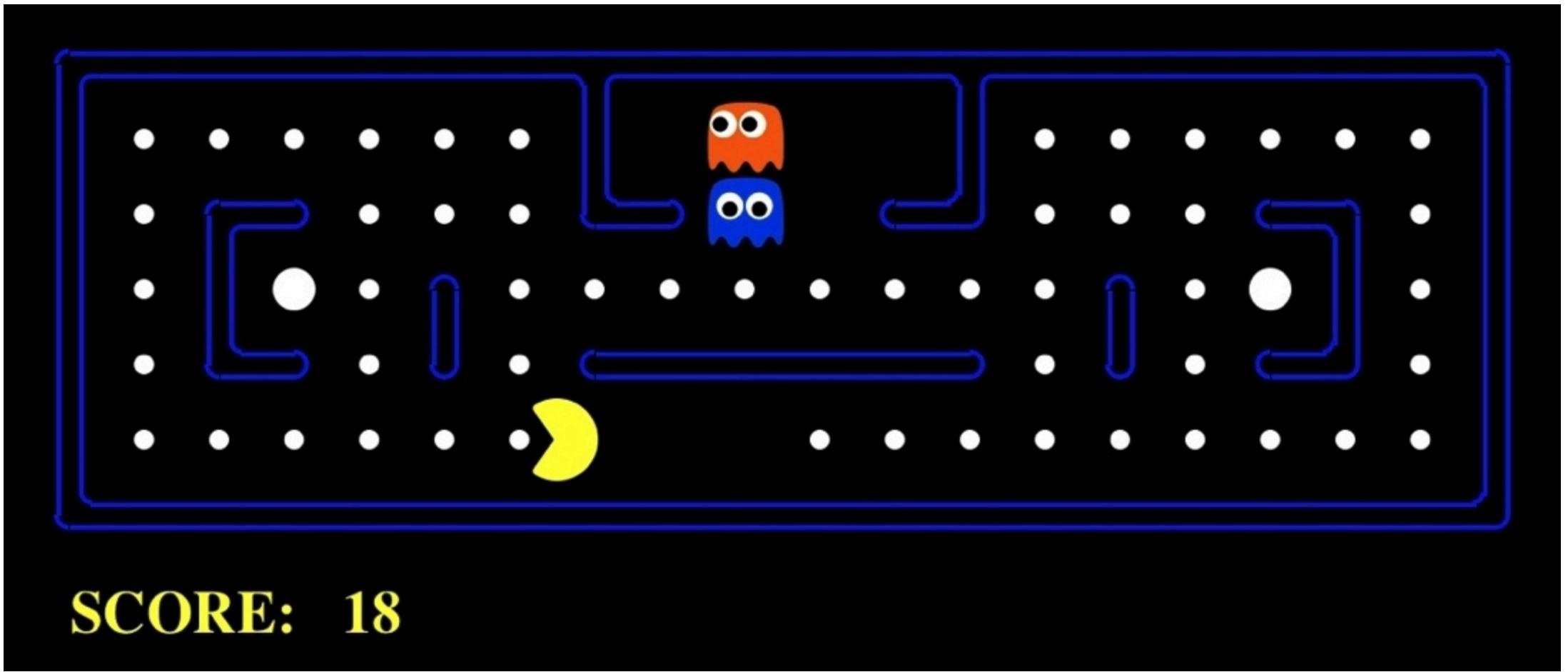
ลองพิจารณาโลก 2 ช่อง ที่มี agent Pacman

- Percept: ตำแหน่งและเนื้อหา เช่น (left cell, no food)
- Action: go left, go right, eat, do nothing

# Pacman agent

Policy ของ Pacman agent คือฟังก์ชันที่แปลงลำดับ percept ไปยัง action ซึ่งสามารถสร้างเป็นตารางได้

Percept sequence	Action
(left cell, no food)	go right
(left cell, food)	eat
(right cell, no food)	go left
(left cell, food)	eat
(left cell, no food), (left cell, no food)	go right
(left cell, no food), (left cell, food)	eat
(...)	(...)



แล้ว Pacman จริงล่ะ?

## The optimal policy?

policy ที่เหมาะสมที่สุดของ agent คืออะไร?

จะนิยามเป้าหมายของ Pacman อย่างไร?

- ได้ 1 แต้มต่อการกิน food dot แต่ละอันจนถึงเวลา  $t$ ?
- ได้ 1 แต้มต่อ food dot ที่กิน, ลบ 1 แต้มต่อ move?
- ลงโทษหากเหลือ food dot ไม่ถูกกินมากเกินไป?

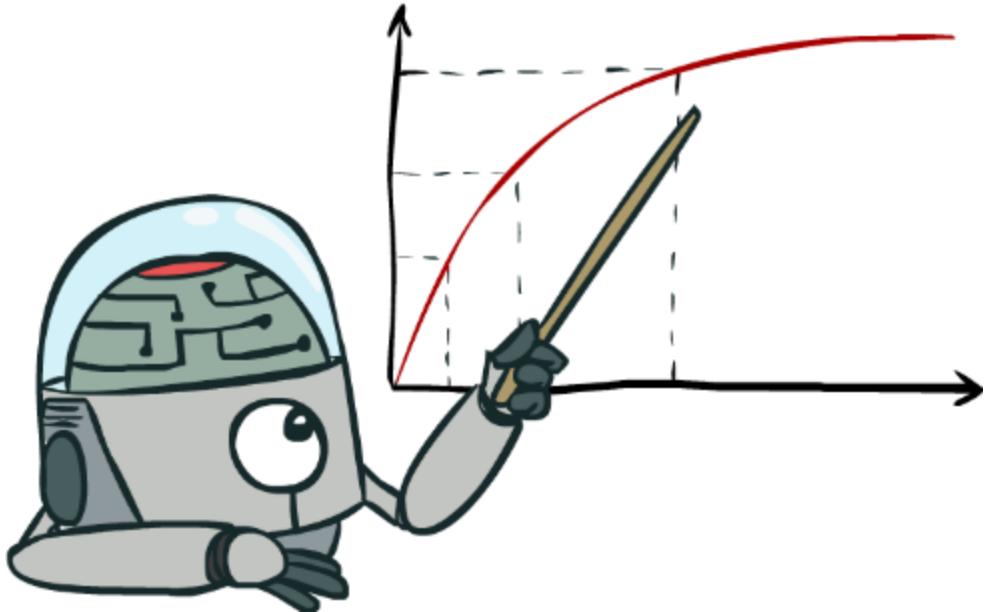
สามารถ implement ใน agent program ขนาดเล็กและมีประสิทธิภาพได้หรือไม่?

# Rational agents

- performance measure ประเมินลำดับสถานะของสภาพแวดล้อมที่เกิดจากพฤติกรรม agent
- rational agent คือ agent ที่เลือก action ที่ **maximize** expected value ของ performance measure ตาม percept sequence ที่ได้รับมาจนถึงตอนนี้

Rationality สนใจแค่ what ตัดสินใจอะไร (ไม่ใช่ขั้นตอนความคิดแบบมนุษย์)

Credits: [CS188](#), UC Berkeley.



ในคอร์สนี้ Artificial intelligence = **Maximizing expected performance**

Credits: [CS188](#), UC Berkeley.

- Rationality  $\neq$  omniscience
  - percept อาจจะไม่บอกข้อมูลที่จำเป็นทั้งหมด
- Rationality  $\neq$  clairvoyance
  - ผลลัพธ์ของ action อาจไม่เป็นตามที่คาด
- ดังนั้น rational  $\neq$  สำเร็จเสมอ
- แต่ rationality นำไปสู่ *exploration, learning* และ *autonomy*

# Performance, environment, actuators, sensors

ลักษณะของ performance measure, environment, action space และ percept  
กำหนดแนวทางในการเลือก rational action  
ทั้งหมดนี้รวมเรียกว่า **task environment**

## ตัวอย่าง 1: agent เล่นหมากรุก

- performance measure: ชนะ, เสมอ, แพ้, ...
- environment: กระดาน, คู่ต่อสู้, ...
- actuator: ขยับตัวมาก, ...
- sensor: สถานะกระดาน, การเดินของคู่ต่อสู้, ...

## ตัวอย่าง 2: รถยนต์ไร้คนขับ

- performance measure: ความปลอดภัย, ถึงจุดหมาย, ความถูกต้องตามกฎหมาย, ความสวยงาม, ...
- environment: ถนน, ทางหลวง, การจราจร, คนเดินถนน, สภาพอากาศ, ...
- actuator: พวงมาลัย, คันเร่ง, เบรก, แตร, ลำโพง, จอแสดงผล, ...
- sensor: กล้อง, accelerometer, gauge, เชนเซอร์เครื่องยนต์, GPS, ...

## ตัวอย่าง 3: ระบบวินิจฉัยทางการแพทย์

- performance measure: สุขภาพผู้ป่วย, ค่าใช้จ่าย, เวลา, ...
- environment: ผู้ป่วย, โรงพยาบาล, ประวัติแพทย์, ...
- actuator: วินิจฉัย, รักษา, ส่งต่อ, ...
- sensor: ประวัติแพทย์, ผลแลป, ...

# Environment types

## *Fully observable vs. partially observable*

sensor ของ agent ให้ข้อมูลครบถ้วนของสถานะ environment ณ เวลาหนึ่นหรือไม่

## *Deterministic vs. stochastic*

สถานะถัดไปถูกกำหนดโดยสถานะปัจจุบันและ action ของ agent หรือไม่

## *Episodic vs. sequential*

ประสบการณ์ของ agent แบ่งเป็น episode อิสระหรือไม่

## *Static vs. dynamic*

environment เปลี่ยนแปลงได้หรือไม่ หรือ performance measure เปลี่ยนตามเวลา

## *Discrete vs. continuous*

สถานะ environment, เวลา, percept หรือ action เป็นค่าต่อเนื่องหรือไม่

## *Single agent vs. multi-agent*

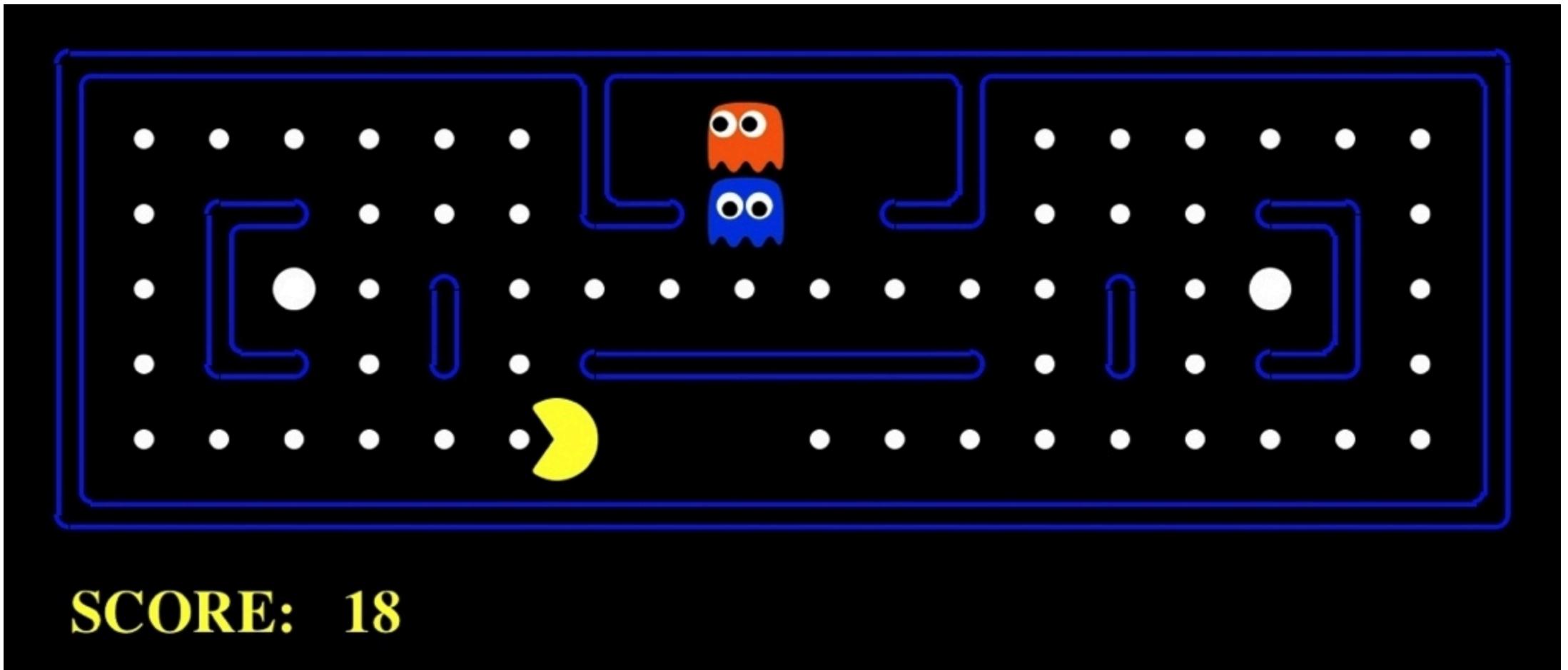
environment มี agent หลายตัวที่มีปฏิสัมพันธ์กันหรือไม่

## *Known vs unknown*

agent มีความรู้เกี่ยวกับ "กฎพิสิกส์" ของ environment หรือไม่

Task environment ต่อไปนี้ fully observable หรือไม่? deterministic? episodic?  
static? discrete? single agent? known?

- Crossword puzzle
- Chess พร้อมนาฬิกา
- Poker
- Backgammon
- ขับรถแท็กซี่
- การวินิจฉัยโรค
- วิเคราะห์ภาพ
- หุ่นยนต์หยับชิ้นส่วน
- refinery controller
- โลจิสติก



Pacman ล่ะ?

# Agent programs

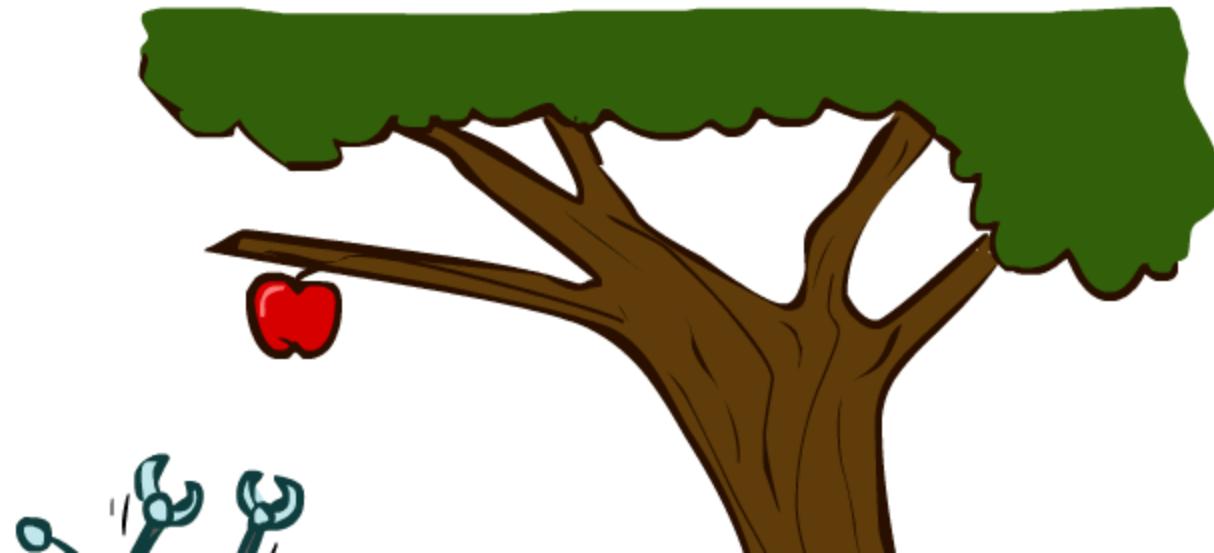
เป้าหมายของเราคือออกแบบ agent program ที่ implement agent policy  
agent program สามารถออกแบบและ implement ได้หลายแบบ เช่น

- ตาราง
- กฎ (rules)
- อัลกอริทึมค้นหา (search algorithm)
- อัลกอริทึมเรียนรู้ (learning algorithm)

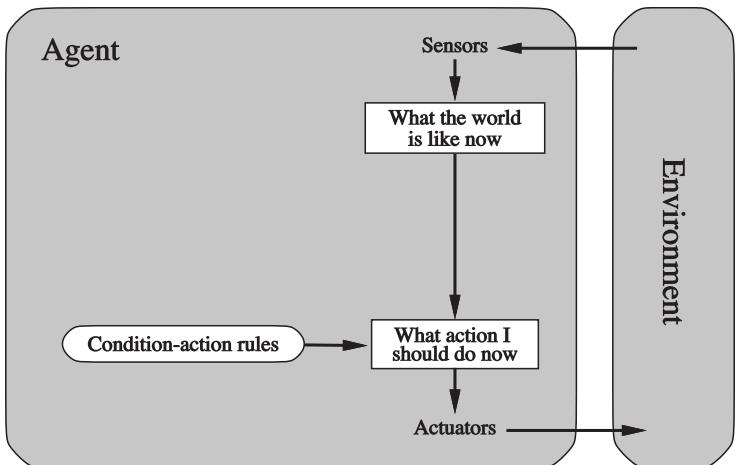
# Reflex agents

Reflex agent ...

- เลือก action จาก percept ปัจจุบัน (อาจจะใช้ memory ด้วย);
- อาจจะมี memory หรือแบบจำลอง world state;
- ไม่พิจารณาผลในอนาคตของ action ที่เลือก

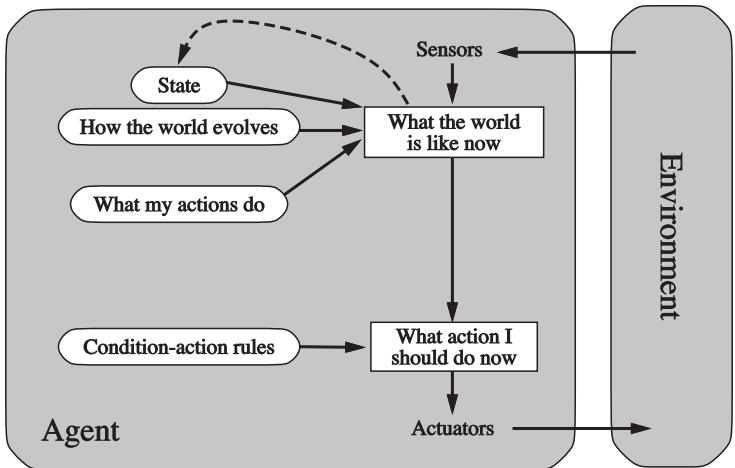


# Simple reflex agents



- *Simple reflex agent* เลือก action จาก percept ปัจจุบัน โดยไม่สนใจประวัติ percept ที่ผ่านมา
- พากเข้าใช้ **condition-action rules** แมทซ์ percept ปัจจุบันกับ action ก្នុងช่วยวัยให้ย่อขนาดฟังก์ชันลงได้
- ทำงานได้ดีเฉพาะใน *Markovian environment* គឺ តัดสินใจได้จาก percept ปัจจุบันเพียงอย่างเดียว  
หมายถึง environment ต้อง *fully observable*

# Model-based reflex agents

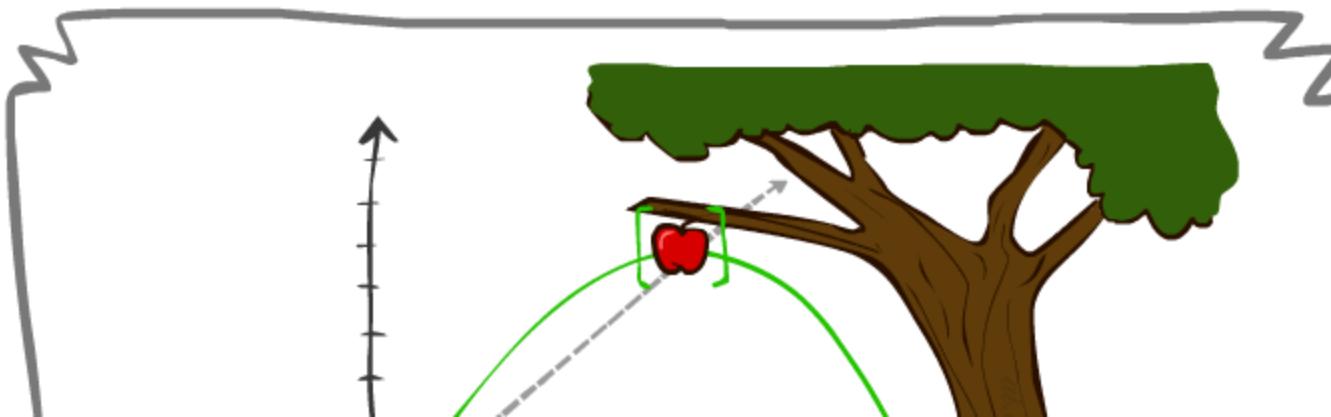


- *Model-based agent* รับมือกับ partial observability โดยเก็บ internal state เพื่อติดตามส่วนที่มองไม่เห็นใน environment
- internal state จะถูกอัปเดตตาม model ที่นักว่า
  - environment เปลี่ยนแปลงอย่างไรหากไม่มี agent
  - agent ทำ action ได้มีผลต่อโลกอย่างไร

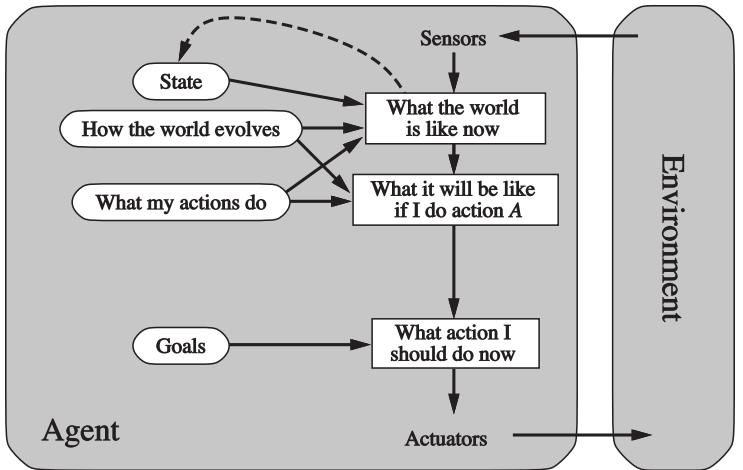
# Planning agents

Planning agent ...

- ถามว่า "what if?"
- ตัดสินใจโดยพิจารณาผล (ที่คาดการณ์) ของ action;
- ต้องมี model ของ environment ว่าเปลี่ยนแปลงอย่างไรเมื่อลงมือทำ action
- ต้องนิยามเป้าหมาย (goal)

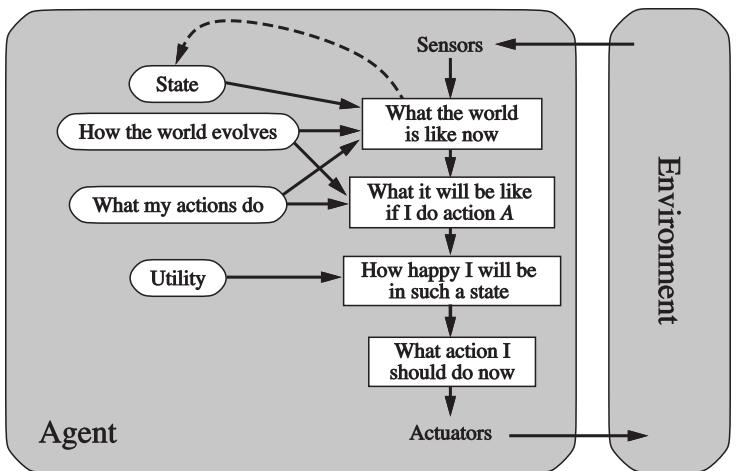


# Goal-based agents



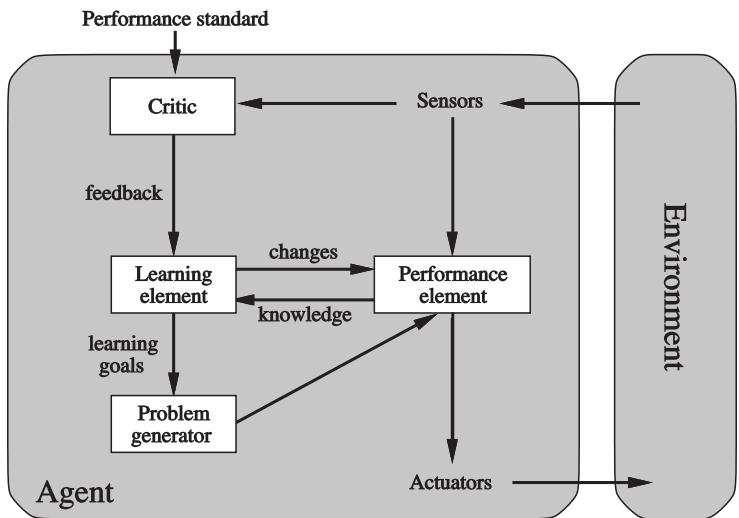
- กระบวนการตัดสินใจ:
  - i. สร้างลำดับ action ที่เป็นไปได้
  - ii. คำนายน state ที่จะตามมา
  - iii. ประเมิน goal ของแต่ละ state
- Goal-based agent เลือก action ที่จะทำให้บรรลุ goal
  - ทั่วไปกว่ากฎ (rules) เพราะ goal มักไม่แสดงในกฎ
  - การหาลำดับ action ที่จะบรรลุเป้าหมายทำได้ยาก  
ต้องใช้ search หรือ planning

# Utility-based agents



- Goal มักไม่เพียงพอที่จะสร้างพัฒนาระบบที่มีคุณภาพสูง  
goal ให้การประเมินเป็นแค่ binary (สำเร็จ/ไม่สำเร็จ)
- Utility function ให้คะแนนลำดับสถานะ environment
  - utility function คือการ internalize performance measure
- rational utility-based agent เลือก action ที่ maximize expected utility ของผลลัพธ์

# Learning agents



- *Learning agent* มีความสามารถในการ self-improvement  
สามารถเก่งขึ้นกว่าความรู้เริ่มต้นของตนเอง
- ปรับเปลี่ยน knowledge component ได้โดย:
  - เรียนรู้ว่า โลกเปลี่ยนแปลงอย่างไร
  - เรียนรู้ผลของ action
  - เรียนรู้ utility ของ action ผ่าน reward

# A learning autonomous car

- *Performance element:*
  - ระบบปัจจุบันที่เลือก action และขับรถ
- *Critic* สังเกตสภาพแวดล้อมแล้วส่งข้อมูลไปยัง *learning element*
  - เช่น รถเลี้ยวซ้ายเร็วข้าม 3 เลน critic ได้ขືนคำสอนจากผู้ขับอื่นและแจ้งว่าเป็น bad action
  - learning element พยายามปรับปรุง performance element เพื่อไม่ให้เกิดเหตุการณ์นี้ อีก
- *Problem generator* ชี้จุดพฤติกรรมที่ควรปรับปรุง และเสนอให้ทดลอง
  - เช่น ทดลองเบรกบนถนนหลากหลายสภาพอากาศ

# Summary

- **Agent** คือสิ่งที่รับรู้และกระทำใน environment
- *Performance measure* ประเมินพฤติกรรมของ agent  
**Rational agent** เลือก action เพื่อ maximize expected value ของ performance measure
- *Task environment* ประกอบด้วย performance measure, environment, actuator, sensor มีมิติสำคัญที่หลากหลาย
- **Agent program** คือสิ่งที่ implement agent function การออกแบบขึ้นกับ task environment
- *Simple reflex agent* ตอบสนองโดยตรงต่อ percept  
*Model-based reflex agent* มี internal state เพื่อติดตามโลก  
*Goal-based agent* กระทำเพื่อบรรลุเป้าหมาย  
**Utility-based agent** maximize expected performance
- ทุก agent สามารถพัฒนาตนเองได้ผ่าน learning

The end.