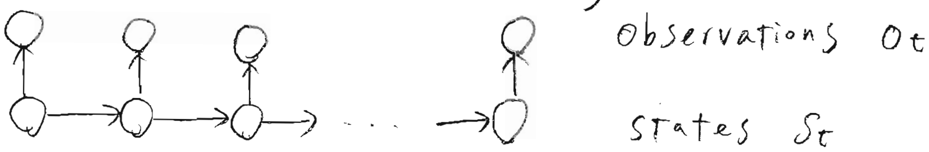
Review

* Hidden Markov models (HMMs)



* Parameters

$a_{ij} = P(S_{t+1}=j \mid S_t=i)$ transition matrix

$b_{ik} = P(O_t=k \mid S_t=i)$ emission matrix

$\pi_i = P(S_1=i)$ initial state distribution

* EM algorithm

Choose parameters $\{a_{ij}, b_{ik}, \pi_i\}$ to maximize log-likelihood $\log P(O_1, O_2, \ldots, O_T)$.

* Forward-backward procedures to compute:

$\alpha_{it} = P(O_1, O_2, \ldots, O_t, S_t=i)$

$\beta_{it} = P(O_{t+1}, \ldots, O_T \mid S_t=i)$

* E-step: Compute posterior probabilities

$$P(S_t=i, S_{t+1}=j \mid O_1, O_2, \ldots, O_T) = \frac{P(S_t=i, S_{t+1}=j, O_1, \ldots, O_T)}{P(O_1, O_2, \ldots, O_T)}$$

$$= \frac{\alpha_{it}\, a_{ij}\, b_j(O_{t+1})\, \beta_{j\,t+1}}{\sum_i \alpha_{iT}}$$

$$P(S_t=i \mid O_1, O_2, \ldots, O_T) = \sum_j P(S_t=i, S_{t+1}=j \mid O_1, O_2, \ldots, O_T)$$

* M-step: update parameters

$$\pi_i \longleftarrow P(S_1=i \mid O_1, O_2, \ldots, O_T)$$

$$a_{ij} \longleftarrow \frac{\sum_t P(S_t=i, S_{t+1}=j \mid O_1, O_2, \ldots, O_T)}{\sum_t P(S_t=i \mid O_1, O_2, \ldots, O_T)}$$

$$b_{ik} \leftarrow \frac{\sum_t p(S_t = i \mid O_1, O_2, \ldots, O_T)\, I(O_t, k)}{\sum_t p(S_t = i \mid O_1, O_2, \ldots, O_T)}$$

Complexity of inference / learning in HMMs

1) compute $p(O_1, O_2, \ldots, O_T)$

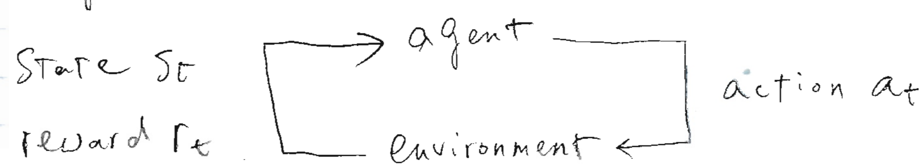
2) decode $S^* = \arg\max_S p(S_1, S_2, \ldots, S_T \mid O_1, \ldots, O_T)$

3) performing iteration of EM

All these have same complexity : $O(\underbrace{n^2}_{\#\text{ states}} \underbrace{T}_{\text{sequence length}})$

Reinforcement Learning

Q. How should embodied / embedded / situated / decision-making agents act and learn from experience in the world?



Ex: robot navigation
Ex: backgammon

* Challenges

- handling of uncertainty
- exploration vs. exploitation dilemma
- delayed vs. immediate rewards : "temporal credit assignment"
- evaluative vs. instructive feedback
- complex worlds, computational guarantees.

Markov decision processes (MDPs)

* Definition
  - state space $\mathcal{S}$ with states $s \in \mathcal{S}$
  - action space $\mathcal{A}$ with actions $a \in \mathcal{A}$
  - Transition probabilities

    for all state-action pairs (s, a),

    $$P(s'|s,a) = P(s_{t+1} = s' \mid s_t = s,\ a_t = a)$$

    probability moving from state s to state t after taking action a.

* Assumptions
  - time-independent

    $$P(s_{t+1} = s' \mid s_t = s,\ a_t = a) = P(s_t = s' \mid s_{t-1} = s,\ a_{t-1} = a)$$

  - Markov condition

    $$P(s_{t+1} \mid s_t, a_t) = P(s_{t+1} \mid s_t, a_t, s_{t-1}, a_{t-1}, s_{t-2}, a_{t-2}, \ldots)$$

    (conditional independence)

* Def (con't)
  - reward function

    $R(s, s', a)$ = real-valued reward after taking action a in state s and moving to state s'.

  - Simplifications for CSE 150
    - reward function $R(s, s', a) = R(s) = R_s$ only depends on current state.
    - bounded, deterministic rewards $\max_s |R_s| < \infty$

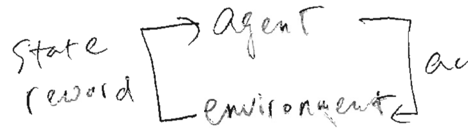
- more simplifications

  * discrete, finite state space
  * discrete, finite action space

  $\}$ vs. continuous, infinite

Example: backgammon

$\mathcal{S}$ = board position and roll of dice
$\mathcal{A}$ = set of possible moves



$p(s'|s, a)$ = how state changes due to agent's move, opponent's roll of dice, opponent's move, agent's roll of dice.

$$R(s) = \begin{cases} +1 & \text{win} \\ -1 & \text{lose} \\ 0 & \text{otherwise} \end{cases}$$

* Decision making
  - policy: deterministic mapping from states to actions
    $\pi : \mathcal{S} \to \mathcal{A}$
  - # policies: $|\mathcal{A}|^{|\mathcal{S}|}$
  - dynamics
    $p(s' | s, \pi(s))$ ← action in state s under policy π.
  - experience under policy π

$$\begin{array}{lccccc} \text{state } s_0 & \xrightarrow[\text{(action)}]{a_0 = \pi(s_0)} & s_1 & \xrightarrow{a_1 = \pi(s_1)} & s_2 & \longrightarrow \cdots \\ \text{reward } r_0 & & r_1 & & r_2 & \end{array}$$

* How to measure accumulated rewards over time?

long term discounted return

discount factor $0 \leq \gamma \leq 1$

$$\text{return} = \sum_{t=0}^{\infty} \gamma^t r_t$$

Possibilities:

$\gamma = 0 \rightarrow$ only immediate reward at $t=0$ matters

$\gamma \ll 1 \rightarrow$ near-sighted agent

$\gamma \approx 1 \rightarrow$ far-sighted agents

Intuitively: near future weighted more heavily than distant future.

Mathematically convenient: leads to recursive algorithms.

State value function

$V^\pi(s)$ = expected discounted return following policy π from initial state s.

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

↑ expectation operation (computes mean)

* Relating value function in different states:

$$V^\pi(s) = E^\pi\left[R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots \mid s_0 = s\right]$$

$$= R(s) + \gamma E^\pi\left[R(s_1) + \gamma R(s_2) + \gamma^2 R(s_3) + \cdots \mid s_0 = s\right]$$

$$= R(s) + \gamma \sum_{s'} P(s' \mid s, \pi(s))\, E^\pi\left[R(s_1) + \gamma R(s_2) + \cdots \mid s_1 = s'\right]$$

$$\boxed{V^\pi(s) = R(s) + \gamma \sum_{s'} P(s' \mid s, \pi(s))\, V^\pi(s')}$$ Bellman equation

Action-Value function

$Q^\pi(s,a)$ = expected return from initial state s, taking action a, then following policy π.

$$Q^\pi(s,a) = E^\pi\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \,\middle|\, s_0 = s, a_0 = a\right]$$

it might be that $a \neq \pi(s)$

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

Optimality in MDPs

Optimal policy

Thm: there is always at least one policy π^* for which $V^{\pi^*}(s) \geq V^\pi(s)$ for all states and policies.

Goal: how to compute π^*?