Markov Decision Problems (MDP) [17.1]

- **State**
  - -> general representation
  - -> S_0 Initial State
  - -> IS-TERMINAL(s) - Terminal state test

- ACTIONS(s) : state $\longrightarrow$ {actions}
  - -> get legal actions from s

- **Transition Model**
  - Deterministic: RESULT(s, a) $\rightarrow$ s'
    STATE X ACTION $\longrightarrow$ STATE
  - Stochastic: $P(s' \mid s, a)$ PROBABILITY DISTRIBUTION
    STATE X ACTION x STATE $\longrightarrow$ p $\quad 0 \le p \le 1$

MARKOV ASSUMPTION: Transitions only depend on the current state & no other history.

- **Reward Function**
  - -> per action reward
  - $R(s, a, s')$. STATE X ACTION x STATE $\longrightarrow$ r $\quad -R_{MAX} \le r \le R_{MAX}$
    Reward for taking action a in state s to reach state s'

Sequential Decision Problem: Utility is determined by a sequence of decisions.

From Rewards To Utility [17.1.1]

Goal: $U(s)$ Utility value for state s.

Sequential Decision Utility: Sequence of state, action $\rightarrow$ u

$$U_h([\underbrace{s_0, a_0}_{\text{Next Decision}}, s_1, a_1, \ldots, \underbrace{s_n, a_n}_{\text{Future Decision}}])$$ Utility of decision history.

$\rightarrow$ In state s_i took action a_i

Additive Discount Rewards

Utility is the sum of rewards, weighed for their age, with more recent rewards carrying more weight than potential future rewards

- Discount Factor γ with $0 \leq \gamma \leq 1$

$$U_h([\ldots s_i, a_i \ldots]) = \sum_{i=0}^{n} \underbrace{\gamma^i}_{\text{Discount}} \times \underbrace{R(s_i, a_i, s_{i+1})}_{\text{Reward for } i^{th} \text{ Decision}}$$

$\gamma = 1 \longrightarrow$ No discount. Past $\equiv$ Future

$1^n = 1$ for all $n \in \mathbb{N} \longrightarrow \{0, 1, 2, 3, \ldots\}$

$\gamma = 0 \longrightarrow$ Ignore future. Only s_0, a_0 count.

$0^0 = 1$, $0^n = 0$ for $n > 0$.

From U_h to Utility of States [17.1.1 - 17.1.2]

Do the "first & rest" trick to isolate one decision

$$U_h([\cdots s_i, a_i \cdots]) = \sum_{i=0}^{n} \gamma^i R(s_i, a_i, s_{i+1})$$

$$= \underbrace{R(s_0, a_0, s_1)}_{\text{current reward}} + \underset{\substack{\uparrow \\ \text{Discount}}}{\gamma} \times \underbrace{U_h([s_1, a_1, \cdots s_n, a_n])}_{\text{reward-to-go}}$$

But how should we isolate the state from its associated action?

↳ State utility ⟶ Utility of best action in state!

Bellman Equation (With Deterministic Transitions)

$$U(s) = \max_{a \in \text{Action}(s)} \left\{ \underbrace{R(s, a, \text{Result}(s,a))}_{\text{Reward for } s} + \gamma \underbrace{U(\text{Result}(s,a))}_{\text{reward-to-go}} \right\}$$

* This is really a SYSTEM OF EQUATIONS:
  - One equation per state
  - Per-State equations refering to other states.

* Methods exist that let us solve the system (see 17.2). They generally work through iterative refinement.

We're intrested in LEARNING from experience.

From Utility to Policy [17.1.2]

A policy $\pi(s)$ determines what action to take in state s.

Given $U(s)$,

$$\pi(s) = \underset{a \in \text{Action}(s)}{\text{argmax}} \{ U(\text{Result}(s,a)) \}$$

⇒ action that results in the state with the highest utility.

$Q(s,a)$: The Action-Utility Function.

Utility of action a in state s. More fine-grained than $U(s)$.

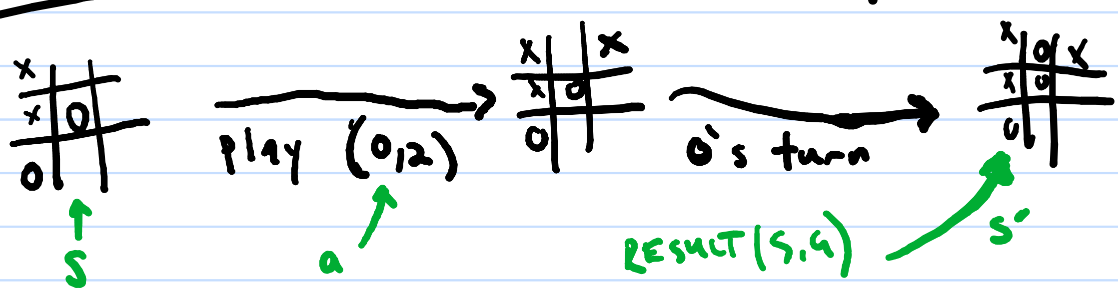
$$U(s) = \max_{a \in \text{Action}(s)} Q(s,a)$$

State utility ⇓ Best action utility

$$U(s) = \max_{a \in \text{Action}(s)} \{ R(s,a,s') + \gamma \max_{a'} Q(s',a') \}$$

Can be observed. No Result needed!

Why?



⇒ No need to predict/model opponent. Just observe

Temporal Difference Learning [22.3.3]

Say we have an estimate for the Action-Utility function Q. We then make action a in state s to result in state s' and receive reward $R(s,a,s')$. We can calculate $Q(s,a)$ as.

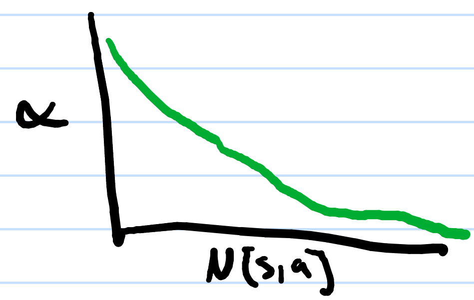
This should be equal to our estimate for $Q(s,a)$. If not we should adjust our estimate based on the error.

$$Q(s,a) \leftarrow Q(s,a) + \alpha \underbrace{\Big[\underbrace{R(s,a,s') + \gamma \max_{a'} Q(s',a')}_{\text{Calculated } Q(s,a)} - \underbrace{Q(s,a)}_{\text{expected } Q(s,a)} \Big]}_{\text{Error of expected \& observed values}}$$

Learning rate parameter

Do this enough and our Q values will converge to the actual values. The learning rate parameter α determines the size of the adjustment. Ideally, it's a function of time and decreases the more you update the value for (s,a)

$$\alpha(\underbrace{N[s,a]}_{\text{number of updates to } Q(s,a)})$$



Active Learning [17.3, 22.3 - 22.3.1]

But ... actions taken are based on the policy π, which is ideally based on the utility function Q.

⇒ While learning, we want to balance EXPLOITING moves we think are good (high utility) while EXPLORING moves of unknown quality. After sufficient exploration, we can stick to "good" moves.

The EXPLORATION FUNCTION f determines our preference for Exploration vs exploitation.

$$f : \underbrace{\text{STATE} \times \text{ACTION}}_{\text{given state action}} \longrightarrow \underbrace{V}_{\text{optimistic utility or "RANK"}}$$

○ TRY AT LEAST N_e

$$f(s,a) = \begin{cases} R_{max} & N[s,a] < N_e \\ Q(s,a) & \text{otherwise} \end{cases}$$

□ UCB1 [5.4]

$$f(s,a) = Q(s,a) + C \times \sqrt{\frac{\log\left(\sum_{a'} N[s,a']\right)}{N[s,a]}}$$ ← log(Times in s)

Temporal Difference Q-Learning

- ☐ Specify Game as MDP. [Transition Model Optional!]
- ☐ Specify Reward Function R
- ☐ Specify learning rate function α $(\mathbb{N} \rightarrow r,\ 0 \le r \le 1)$
- ☐ Specify discount rate γ
- ☐ Specify Exploration function f

While Learning

Maintain

$Q[s,a] \rightarrow$ Action-Utility Estimates. Initially 0.

$N[s,a] \rightarrow$ Frequency Counts for state s, action a. Init. 0.

→ Update Q values

Given states s and s' and action a such that taking action a in s resulted in s'.

$$Q[s,a] \mathrel{+}= \alpha(N[s,a])\left[R(s,a,s') + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$$

→ Choose Action

In state s,

$$a \leftarrow \operatorname*{argmax}_{a' \in Actions(s)} f(s,a')$$