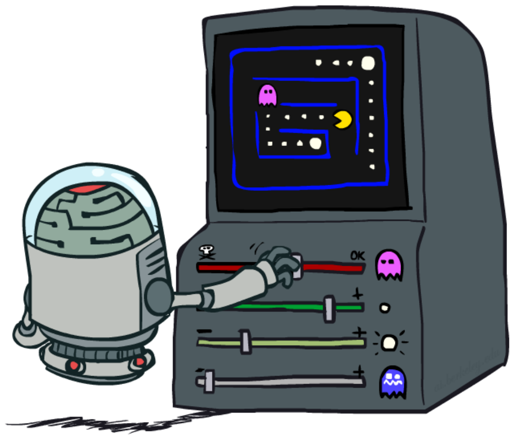
CSE 473: Artificial Intelligence

Reinforcement Learning



Dan Weld/ University of Washington

[Many slides taken from Dan Klein and Pieter Abbeel / CS188 Intro to AI at UC Berkeley – materials available at http://ai.berkeley.edu.]

Three Key Ideas for RL

- Model-based *vs* model-free learning
  - What function is being learned?
- Approximating the Value Function
  - Smaller → easier to learn & better generalization
- Exploration-exploitation tradeoff

Q Learning

- **Forall s, a**
  - **Initialize Q(s, a) = 0**
- **Repeat Forever**

  Where are you? s.

  Choose some action a

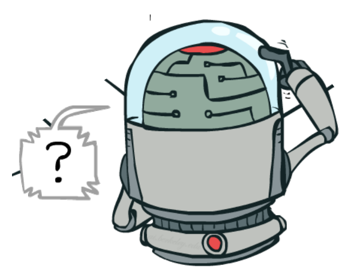
  Execute it in real world: *(s, a, r, s')*

  Do update:

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + (\alpha)\left[r + \gamma \max_{a'} Q(s',a')\right]$$

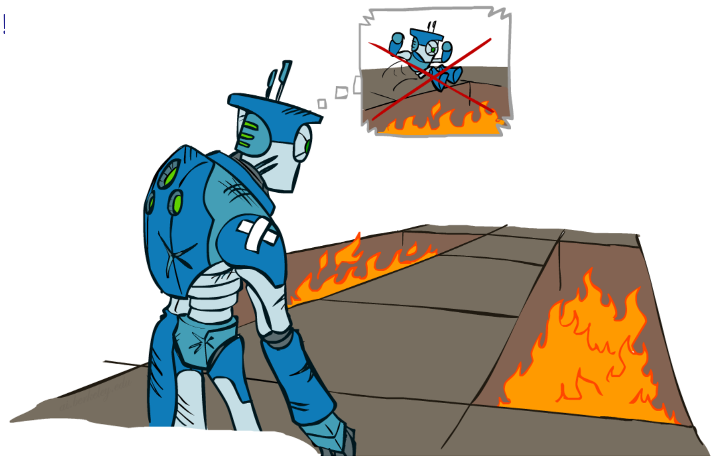
Questions

- How to explore?
  - Random Exploration
  - Uniform exploration
  - Epsilon Greedy
    - With (small) probability ε, act randomly
    - With (large) probability 1-ε, act on ***current policy***
  - Exploration Functions (such as UCB)
  - Thompson Sampling
- When to exploit?
- How to even think about this tradeoff?



Regret

- Even if you learn the optimal policy, you still make mistakes along the way!
- Regret is a measure of your total ***mistake cost***: the difference between your (expected) rewards, including youthful sub-optimality, and optimal (expected) rewards
- Minimizing regret goes beyond learning to be optimal – it requires ***optimally learning to be optimal***



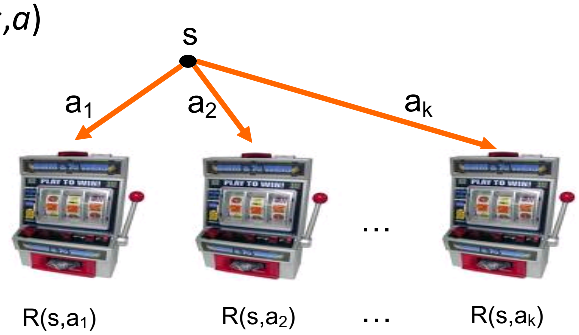
Two KINDS of Regret

- **Cumulative Regret:**
  - achieve near optimal cumulative lifetime reward (in expectation)
- **Simple Regret:**
  - quickly identify policy with high reward (in expectation)

48

RL on Single State MDP

- Suppose MDP has a single state and k actions
  - Can sample rewards of actions using call to simulator
  - Sampling action *a* is like pulling slot machine arm with random payoff function *R*(*s*,*a*)



50

Multi-Armed Bandit Problem

Slide adapted from **Alan Fern** (OSU)

UCB Algorithm for Minimizing Cumulative Regret

Auer, P., Cesa-Bianchi, N., & Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine learning*, *47*(2), 235-256.

- Q(a) : average reward for trying action a (in our single state *s*) so far
- n(a) : number of pulls of arm a so far
- Action choice by UCB after n pulls:

$$a_n = \arg\max_a Q(a) + \sqrt{\frac{2\ln n}{n(a)}}$$

- Assumes rewards in [0,1] – normalized from R_{max}.

58

Slide adapted from **Alan Fern** (OSU)

UCB Performance Guarantee

[Auer, Cesa-Bianchi, & Fischer, 2002]

Theorem: The expected cumulative regret of UCB $\boldsymbol{E}[\boldsymbol{Reg_n}]$ after *n* arm pulls is bounded by O(log *n*)

- Is this good?

Yes. The average per-step regret is O$\left(\frac{\log(n)}{n}\right)$

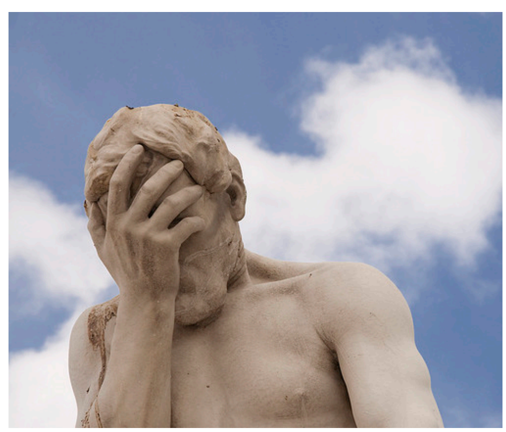
Theorem: No algorithm can achieve a better expected regret (up to constant factors)

60

Slide adapted from **Alan Fern** (OSU)

Two KINDS of Regret

- **Cumulative Regret:**
  - achieve near optimal cumulative lifetime reward (in expectation)
- **Simple Regret:**
  - quickly identify policy with high reward (in expectation)



78

Simple Regret Objective

- **Protocol:** At time step n the algorithm picks an "exploration" arm a_n to pull and observes reward r_n and also picks an arm index it thinks is best j_n *(a_n, j_n and r_n are random variables)*.
  - If interrupted at time n the algorithm returns j_n.
- **Expected Simple Regret ($\boldsymbol{E[SReg_n]}$):** difference between R^* and expected reward of arm j_n selected by our strategy at time n

$$E[SReg_n] = R^* - E[R(a_{j_n})]$$

79

Simple Regret Objective

What about UCB for simple regret?

Theorem: The expected simple regret of UCB after *n* arm pulls is upper bounded by $O(n^{-c})$ for a constant c.

Seems good, but we can do much better (at least in theory).

- Intuitively: UCB puts too much emphasis on pulling the best arm
- After an arm is looking good, maybe better to see if ∃ a better arm

Incremental Uniform (or Round Robin)

Bubeck, S., Munos, R., & Stoltz, G. (2011). Pure exploration in finitely-armed and continuous-armed bandits. Theoretical Computer Science, 412(19), 1832-1852

Algorithm:

- At round n pull arm with index (k mod n) + 1
- At round n return arm (if asked) with largest average reward

Theorem: The expected simple regret of Uniform after *n* arm pulls is upper bounded by O(e^{-cn}) for a constant c.

- This bound is exponentially decreasing in n!

  Compared to polynomially for UCB O(n^{-c}).

81

Can we do even better?

Tolpin, D. & Shimony, S, E. (2012). MCTS Based on Simple Regret. *AAAI Conference on Artificial Intelligence.*

Algorithm -Greedy **:** (parameter)

- At round n, with probability pull arm with best average reward so far, otherwise pull one of the other arms at random.
- At round n return arm (if asked) with largest average reward

Theorem: The expected simple regret of ϵ-Greedy for $\epsilon = 0.5$ after *n* arm pulls is upper bounded by O(e^{-cn}) for a constant c that is larger than the constant for Uniform (this holds for “large enough” n).

82

Summary of Bandits in Theory

PAC Objective:

- **UniformBandit** is a simple PAC algorithm
- **MedianElimination** improves by a factor of log(k) and is optimal up to constant factors

Cumulative Regret:

- **Uniform** is very bad!
- **UCB** is optimal (up to constant factors)

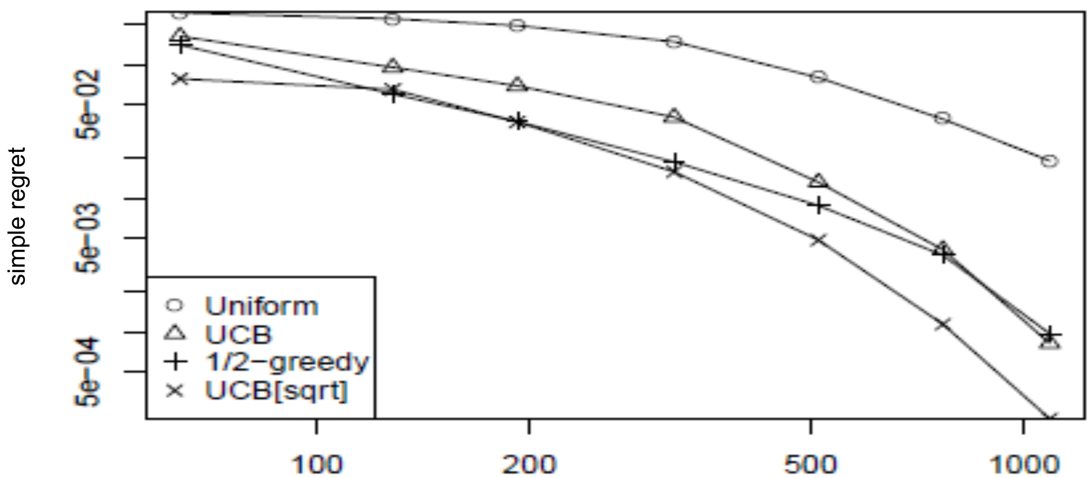
Simple Regret:

- **UCB** shown to reduce regret at polynomial rate
- **Uniform** reduces at an exponential rate
- **0.5-Greedy** may have even better exponential rate

Theory *vs*. Practice

- The established theoretical relationships among bandit algorithms have often been useful in predicting empirical relationships.
- But not always ….

Theory *vs.* Practice



Simple regret *vs.* number of samples

UCB maximizes $Q_a + \sqrt{(2 \ln(n)) / n_a}$

UCB[sqrt] maximizes $Q_a + \sqrt{(2 \sqrt{n}) / n_a}$