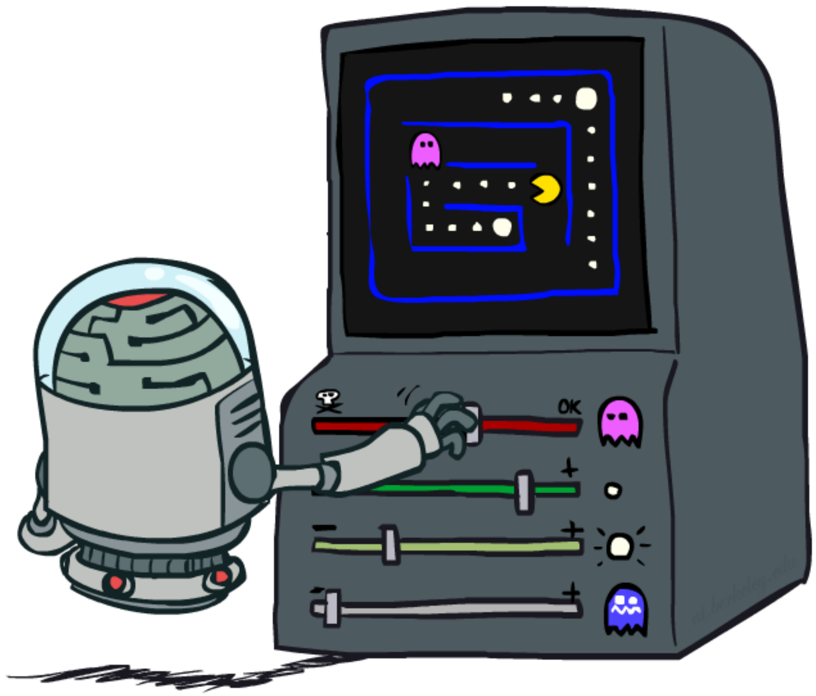
CSE 573: Artificial Intelligence

Reinforcement Learning



Dan Weld/ University of Washington

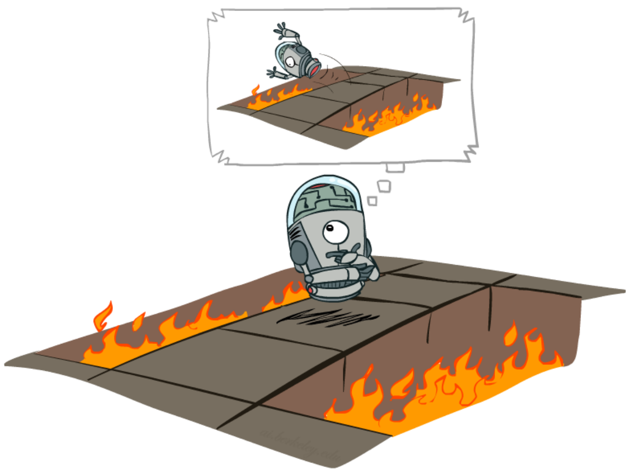
[Many slides taken from Dan Klein and Pieter Abbeel / CS188 Intro to AI at UC Berkeley – materials available at http://ai.berkeley.edu.]

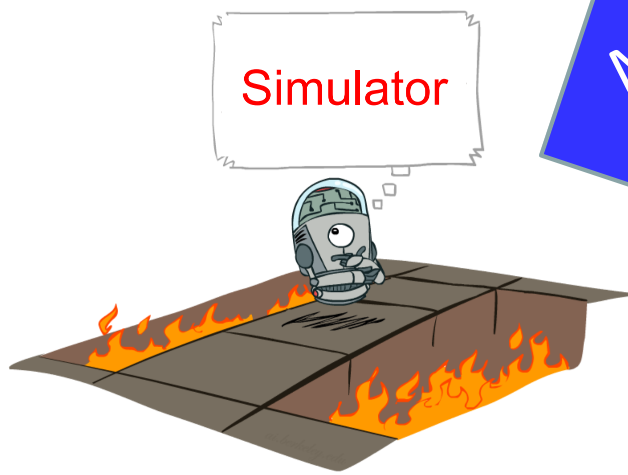
Logistics

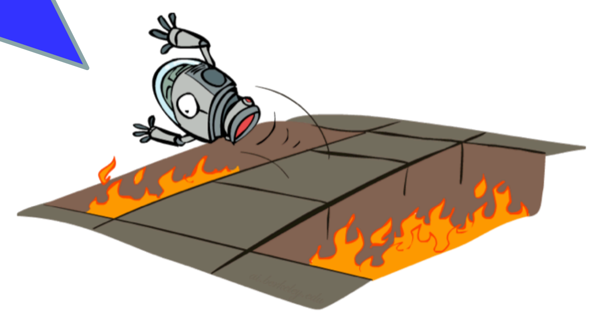
Title: Neural Question Answering over Knowledge Graphs
Speaker: Wenpeng Yin (University of Munich)
Time: Thursday, Feb 16, 10:30 am
Location: CSE 403

Offline (MDPs) vs. Online (RL)

Many people call this RL as well







Offline Solution (Planning)

Monte Carlo Planning

Online Learning (RL)

Diff: 1) dying ok; 2) (re)set button

Approximate Q Learning

$$Q(s,a) = w_1 f_1(s,a) + w_2 f_2(s,a) + \ldots + w_n f_n(s,a)$$

- **Forall *i***
  - Initialize $w_i = 0$
- **Repeat Forever**

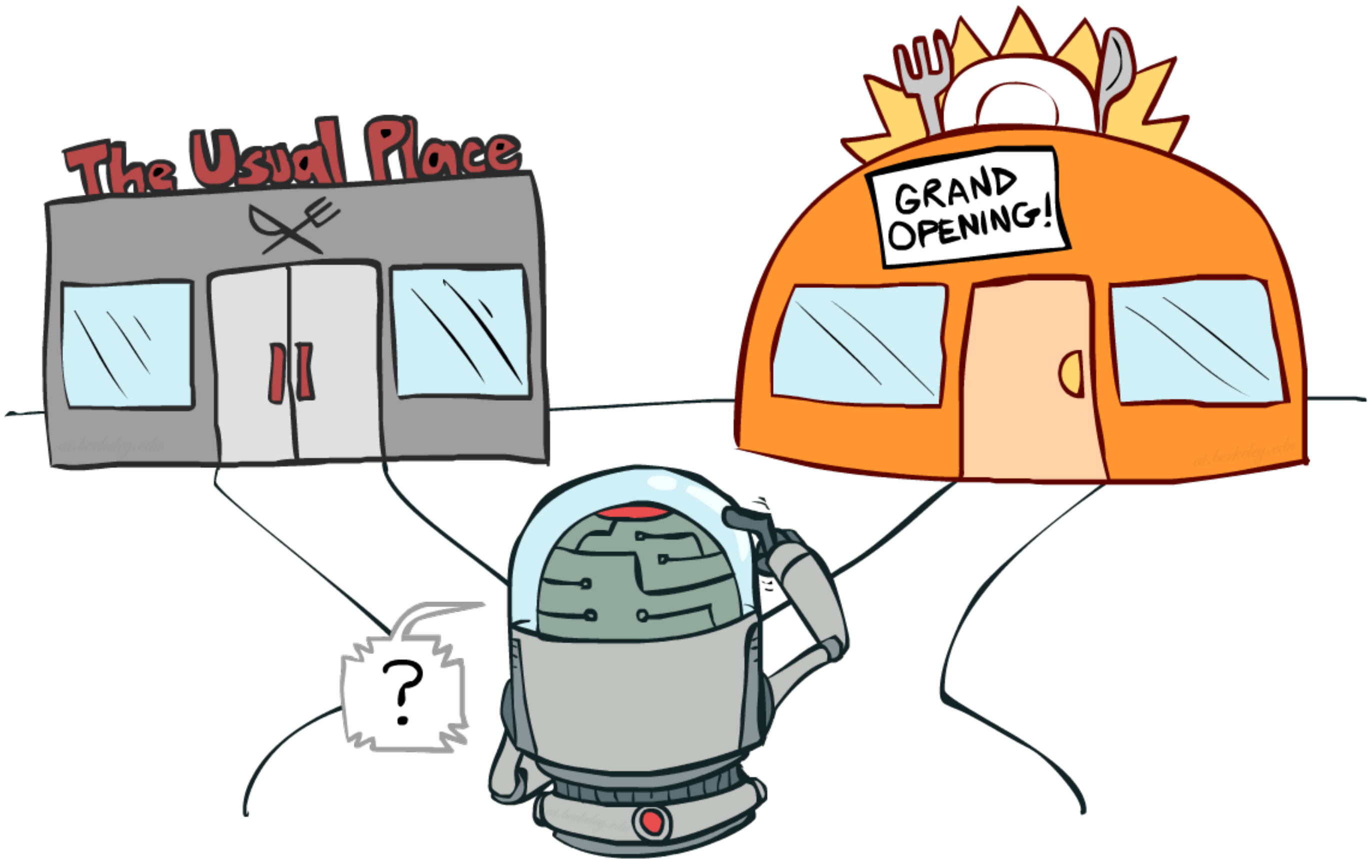
  Where are you? s.
  Choose some action a
  Execute it in real world: *(s, a, r, s')*
  Do update:

  difference ← [r + γ $\text{Max}_{a'}$ Q(s', a')] - Q(s,a)

  Forall *i* do:

$$w_i \leftarrow w_i + \alpha\,[\text{difference}]\, f_i(s,a)$$

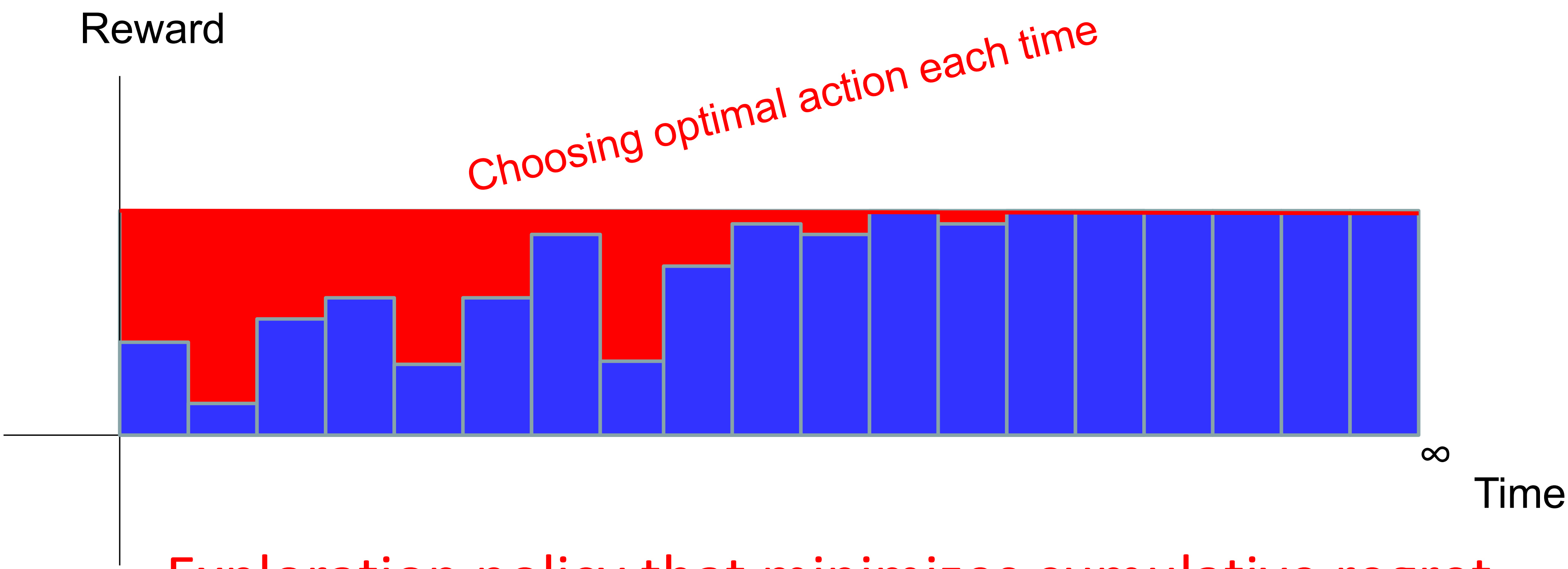
Exploration vs. Exploitation



Two KINDS of Regret

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

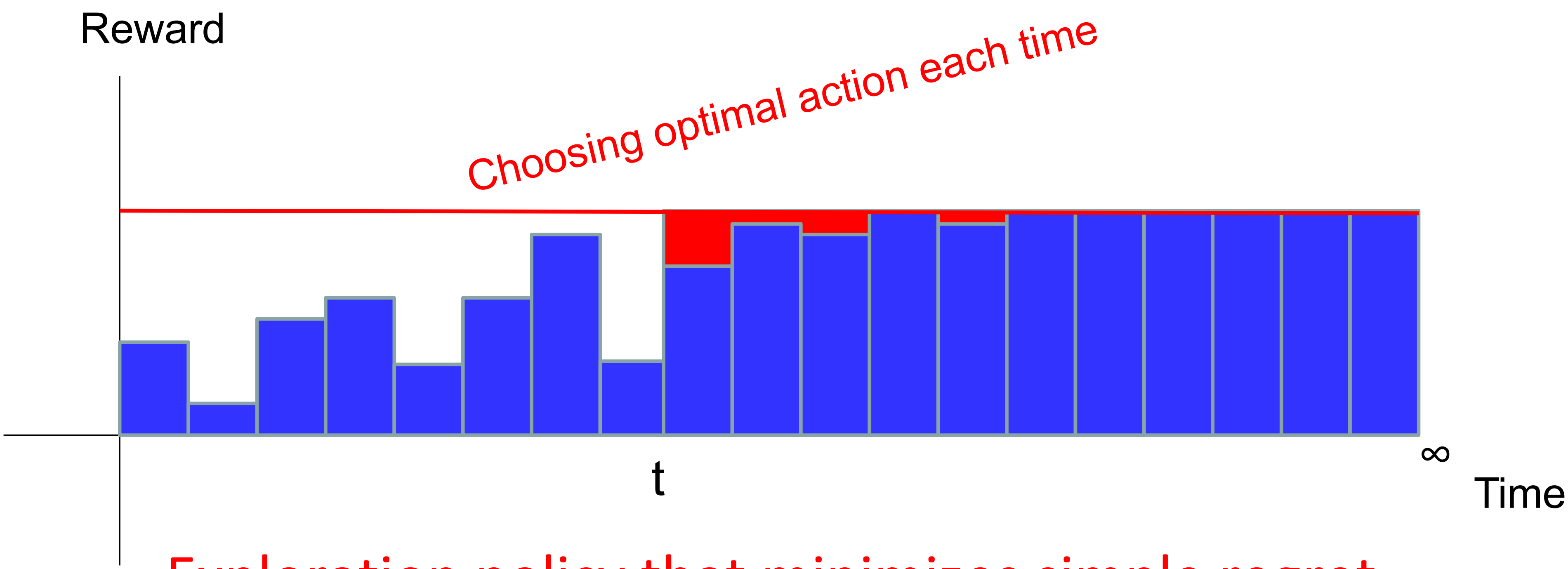
Regret



Exploration policy that minimizes cumulative regret

Minimizes red area

Regret

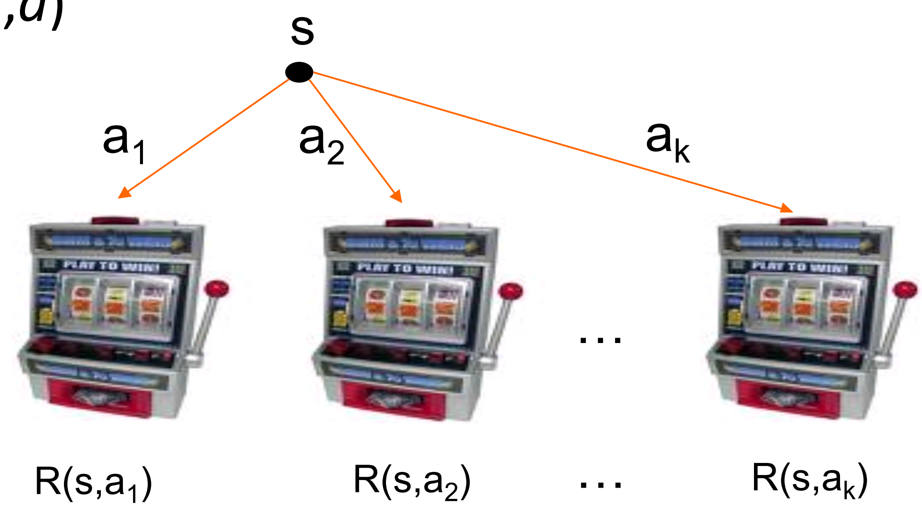


Exploration policy that minimizes simple regret…

For any time, t, minimizes red area after t

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*)

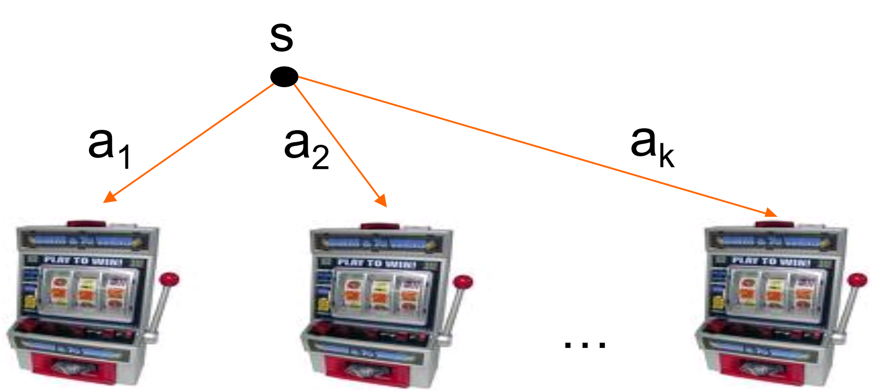


Multi-Armed Bandit Problem

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

Cumulative Regret Objective

- **Problem:** find arm-pulling strategy such that the expected total reward at time *n* is ***close*** to the best possible (one pull per time step)
  - Optimal (in expectation) is to pull optimal arm *n* times
  - UniformBandit is poor choice --- waste time on bad arms
  - Must balance **exploring** machines to find good payoffs and **exploiting** current knowledge



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

Idea

- The problem is uncertainty… How to quantify?
- Error bars

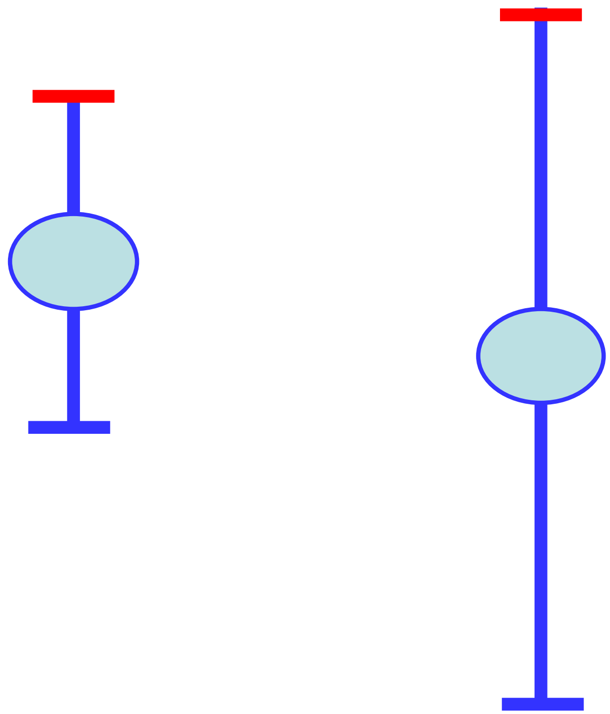
If arm has been sampled n times,
With probability at least 1- δ:

$$|\hat{\mu} - \mu| < \sqrt{\frac{\log(\frac{2}{\delta})}{2n}}$$

Slide adapted from **Travis Mandel** (UW)

Given Error bars, how do we act?

- Optimism under uncertainty!
- Why? If bad, we will soon find out!



Slide adapted from **Travis Mandel** (UW)

Upper Confidence Bound (UCB)

1. Play each arm once

2. Play arm i that maximizes:

$$\hat{\mu}_i + \sqrt{\frac{2\log(t)}{n_i}}$$

3. Repeat Step 2 forever

Slide adapted from **Travis Mandel** (UW)

UCB Performance Guarantee

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

Theorem: The expected cumulative regret of UCB $\boldsymbol{E[Reg_n]}$ 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)

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

UCB as Exploration Function in Q-Learning

Let N_{sa} be number of times one has executed a in s; let $N = \sum_{sa} N_{sa}$

Let $Q^e(s,a) = Q(s,a) + \sqrt{\log(N)/(1+n_{sa})}$

- **Forall s, a**
  - Initialize $Q(s, a) = 0$, $n_{sa} = 0$
- **Repeat Forever**

  Where are you? s.

  Choose action with highest Q^e

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

  Do update:

  N_{sa} += 1;

  difference $\leftarrow [r + \gamma \text{ Max}_{a'} Q^e(s', a')] - Q^e(s,a)$

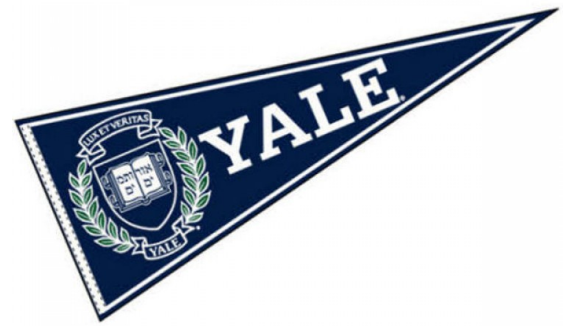
  $Q(s,a) \leftarrow Q^e(s,a) + \alpha(\text{difference})$

Video of Demo Q-learning – Epsilon-Greedy – Crawler

Video of Demo Q-learning – Exploration Function – Crawler

A little history…

William R. Thompson (1933): Was the first to examine MAB problem, proposed a method for solving them



1940s-50s: MAB problem studied intentively during WWII, Thompson was ignored

1970's-1980's: "Optimal" solution (Gittins index) found but is intractable and incomplete. Thompson ignored.

2001: UCB proposed, gains widespread use due to simplicity and "optimal" bounds. Thompson still ignored.

2011: Empricial results show Thompson's 1933 method beats UCB, but little interest since no guarantees.

2013: Optimal bounds finally shown for Thompson Sampling

Slide adapted from **Travis Mandel** (UW)

Thompson's method was fundamentally different!

Bayesian *vs.* Frequentist

- Bayesians: You have a prior, probabilities interpreted as beliefs, prefer probabilistic decisions

- Frequentists: No prior, probabilities interpreted as facts about the world, prefer hard decisions ($p<0.05$)

UCB is a frequentist technique! What if we are Bayesian?

Bayesian review: Bayes' Rule

$$p(\theta \mid data) = \frac{p(data|\theta)p(\theta)}{p(data)}$$

$$p(\theta \mid data) \propto \overbrace{\underbrace{p(data|\theta)}_{\text{Likelihood}}\underbrace{p(\theta)}_{\text{Prior}}}^{\text{Posterior}}$$

Bernoulli Case

What if distribution in the set {0,1}
instead of the range [0,1] ?

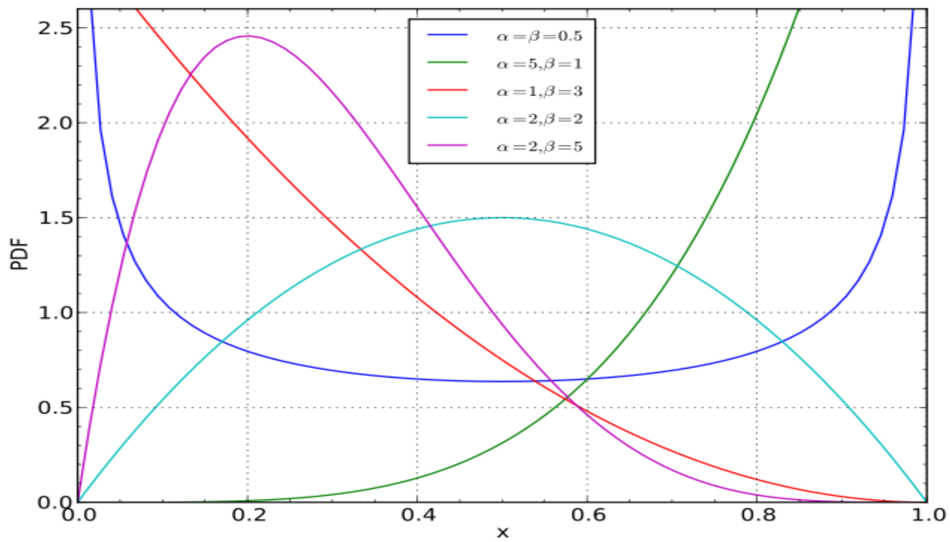
Then we flip a coin with probability p → Bernoulli distribution!

To estimate p, we count up numbers of ones and zeros

Given observed ones and zeroes, how do we calculate
the distribution of possible values of p?

Beta-Bernoulli Case

Beta(a,b) → Given a 0's and b 1's, what is the distribution over means?



Prior → pseudocounts

Likelihood → Observed counts

Posterior → pseudocounts + observed counts

How does this help us?

Thompson Sampling:

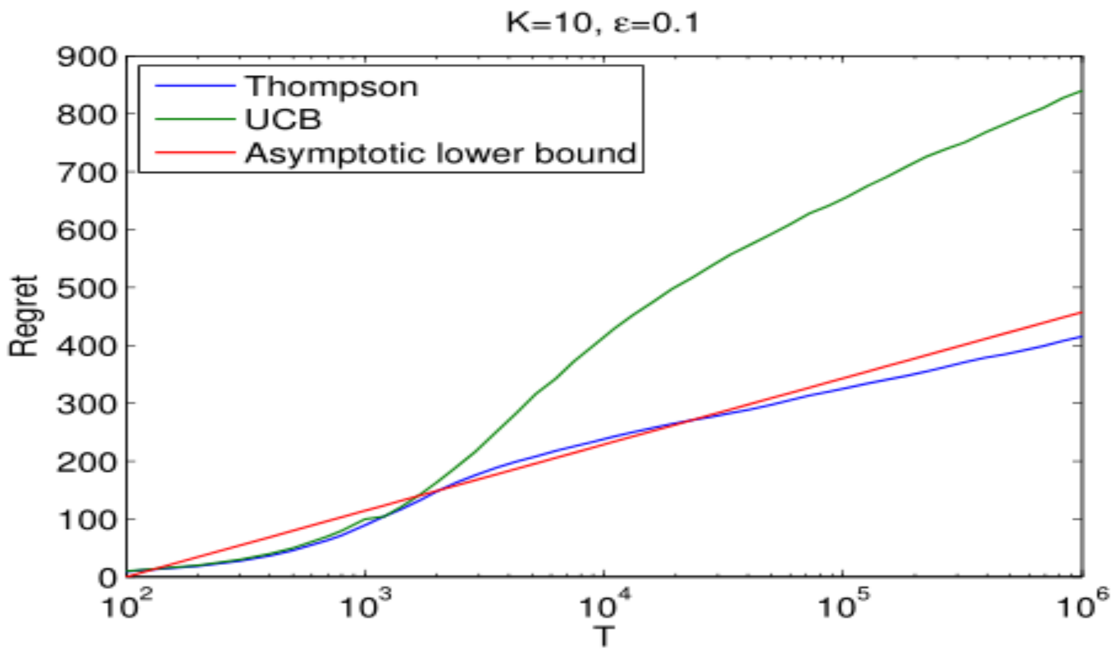
1. Specify prior (e.g., using Beta(1,1))

2. Sample from each posterior distribution to get
   estimated mean for each arm.

3. Pull arm with highest mean.

4. Repeat step 2 & 3 forever

Thompson Empirical Results



And shown to have optimal regret bounds just like
(and in some cases a little better than) UCB!

What Else ….

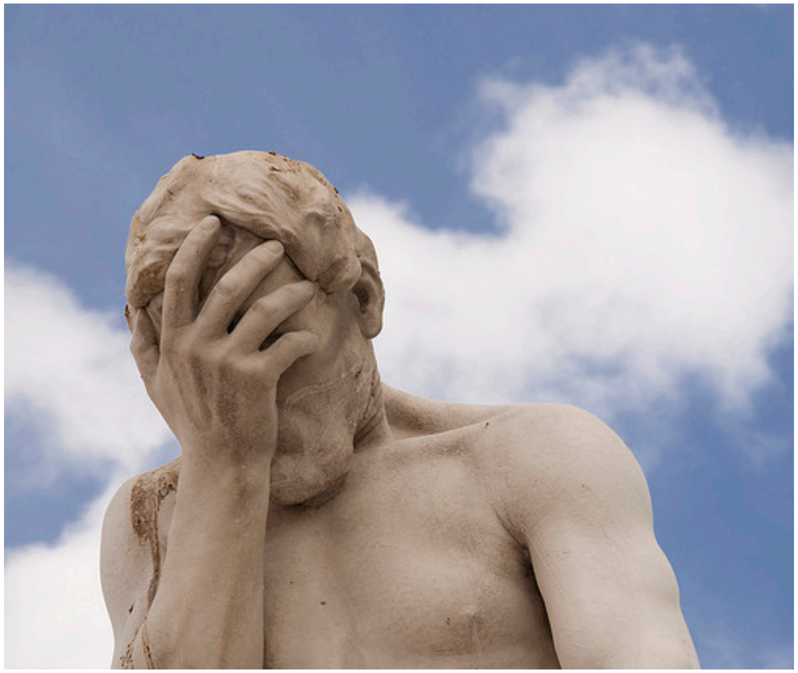
- UCB & Thompson is great when we care about cumulative regret
  - *I.e.,* when the agent is acting in the real world

- But, sometimes all we care about is ***finding a good arm quickly***
  - *E.g.,* when we are training in a simulator

- In these cases, “Simple Regret” is better objective

Two KINDS of Regret

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



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

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

How to Minimize Simple Regret?

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 $\exists$ 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}).

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).

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)
- **Thomson Sampling** also optimal; often performs better in practice

Simple Regret:

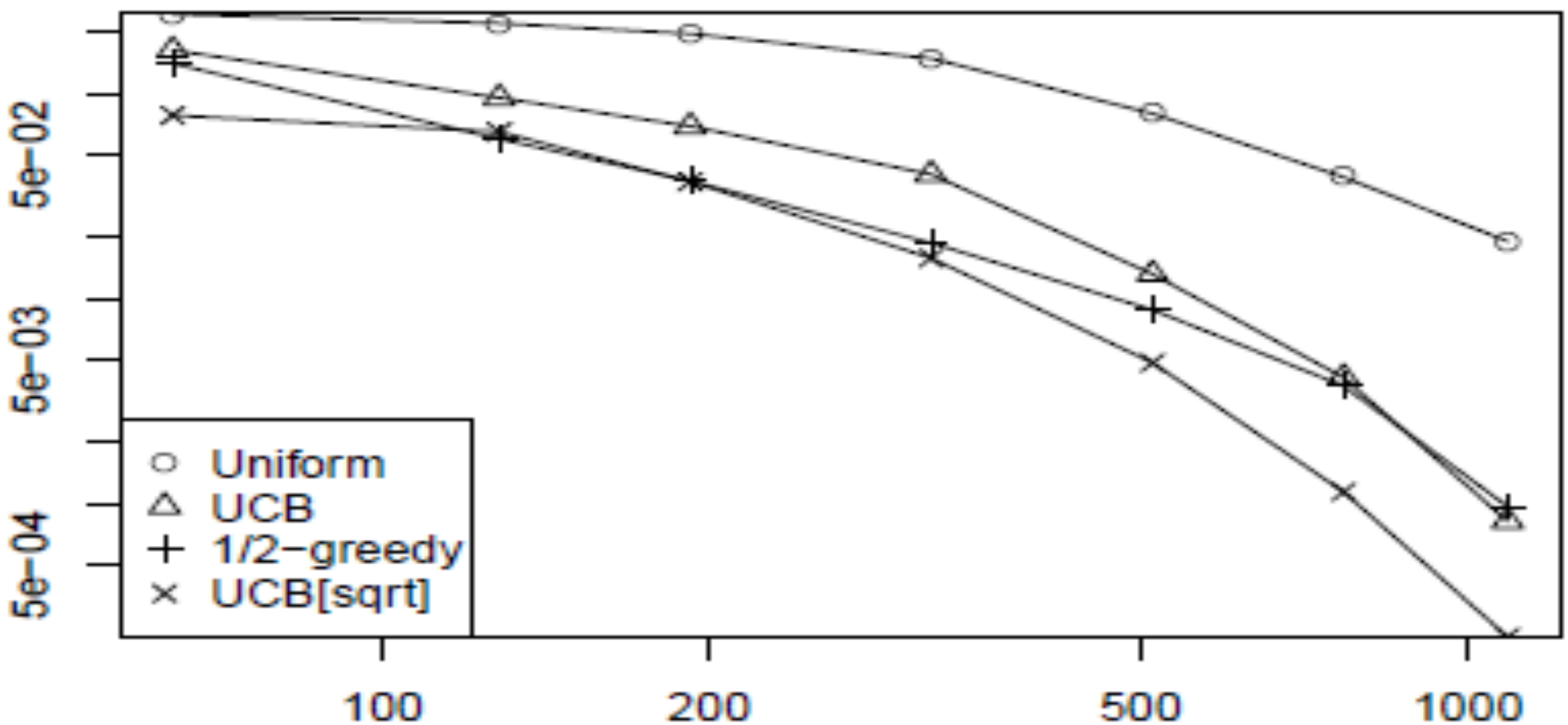
- **UCB** shown to reduce regret at polynomial rate
- **Uniform** reduces at an exponential rate
- **0.5-Greedy** may have even better exponential rate

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

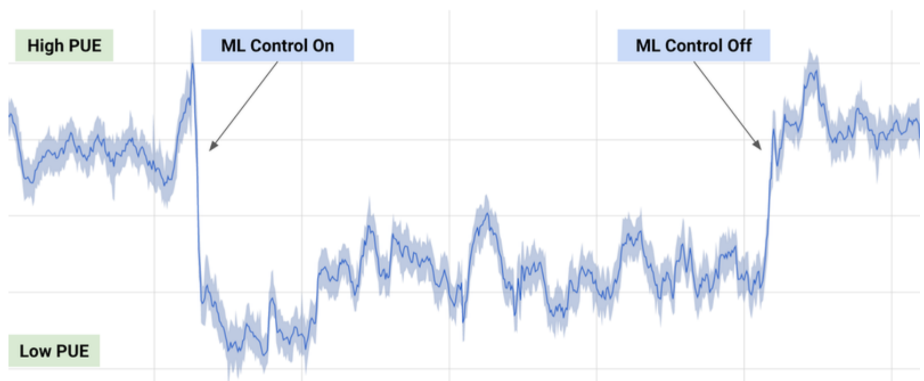


Simple regret *vs.* number of samples

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

That’s all for Reinforcement Learning!

- Very tough problem: How to perform any task well in an unknown, noisy environment!
- Traditionally used mostly for robotics, but…



Google DeepMind – RL applied to data center power usage