RISE
Reconfigurable and Intelligent Systems Engineering

An Introduction to Reinforcement Learning and Multi-arm Bandits

Explore-Exploit Dilemma

B. Ravindran

Reconfigurable and Intelligent Systems (RISE) Group

Department of Computer Science and Engineering

Indian Institute of Technology Madras

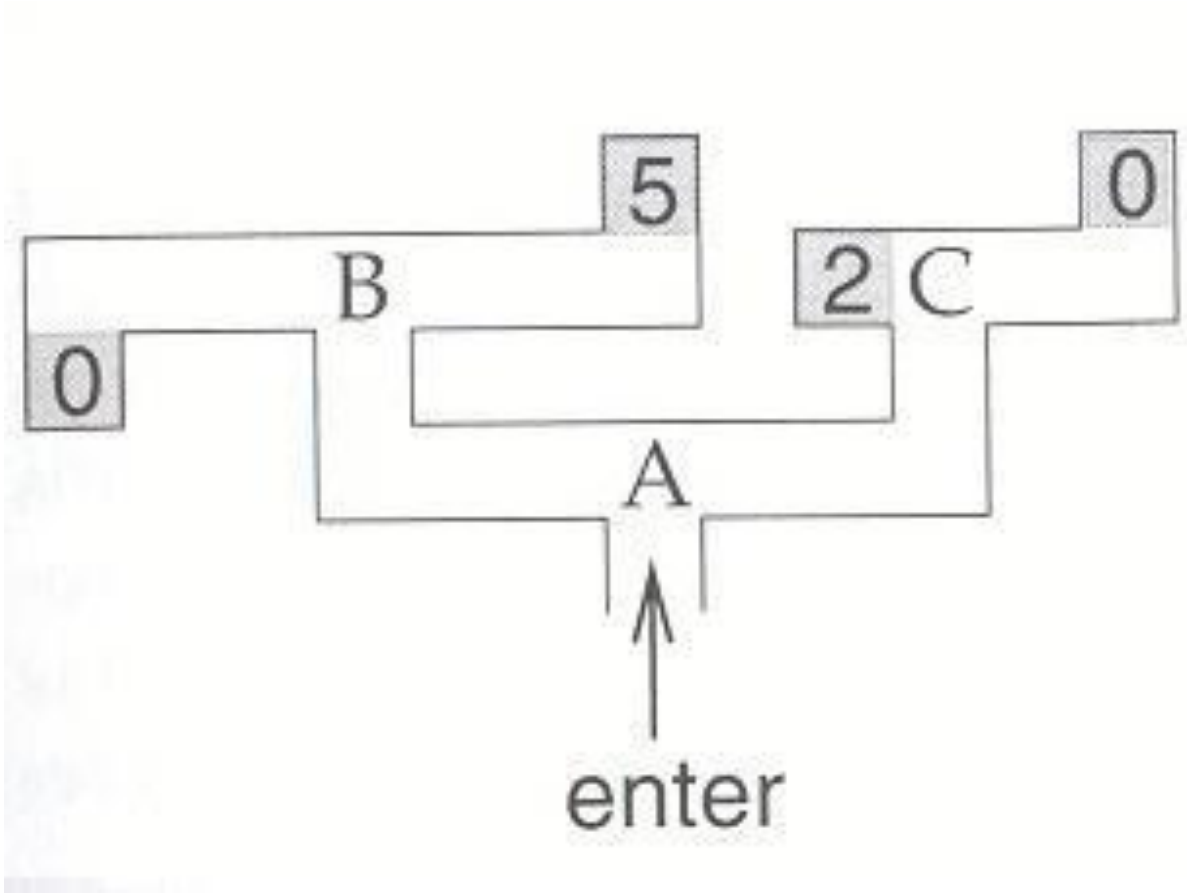
Learning to Control

- Familiar models of machine learning
  - Supervised: Classification, Regression, etc.
  - Unsupervised: Clustering, Frequent patterns, etc.

- How did you learn to cycle?
  - Neither of the above
  - Trial and error!
  - Falling down hurts!

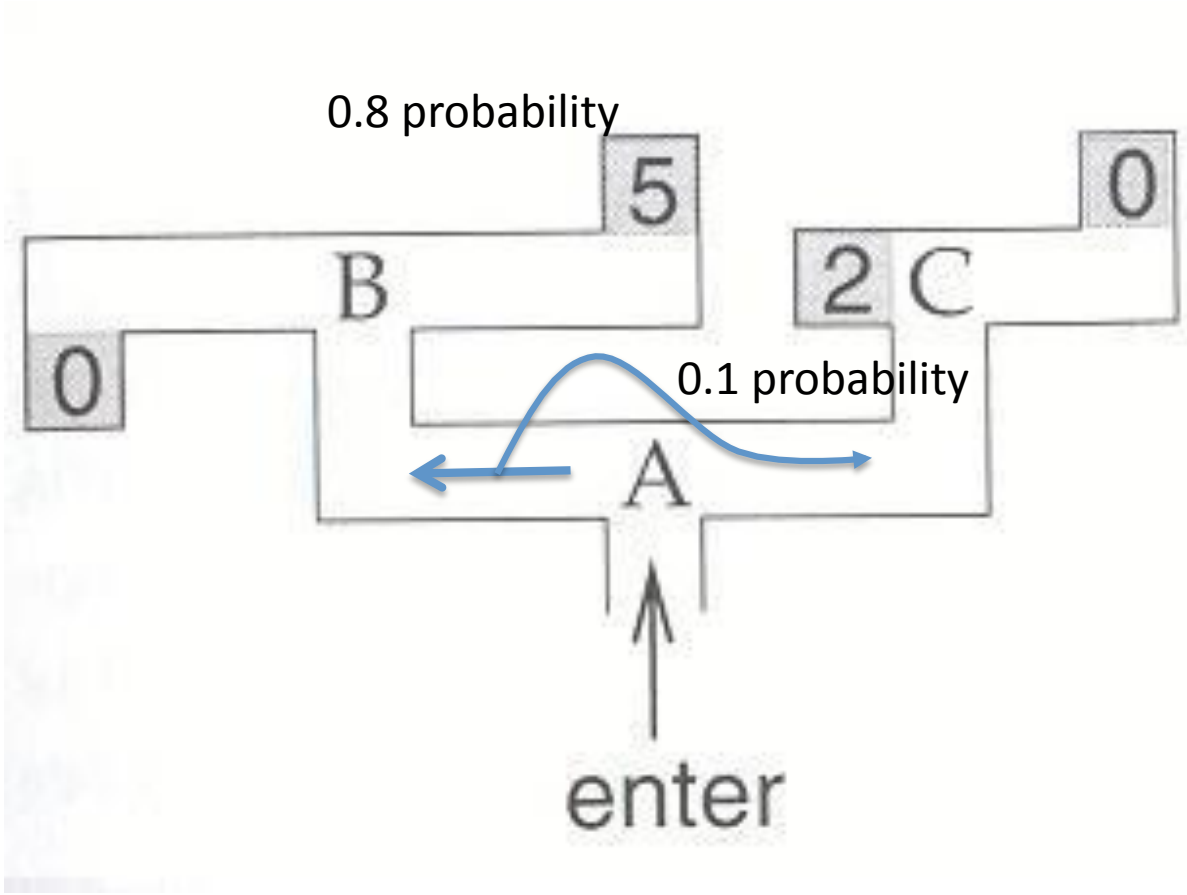
Running a Maze

Dayan and Abbott



Running a Stochastic Maze

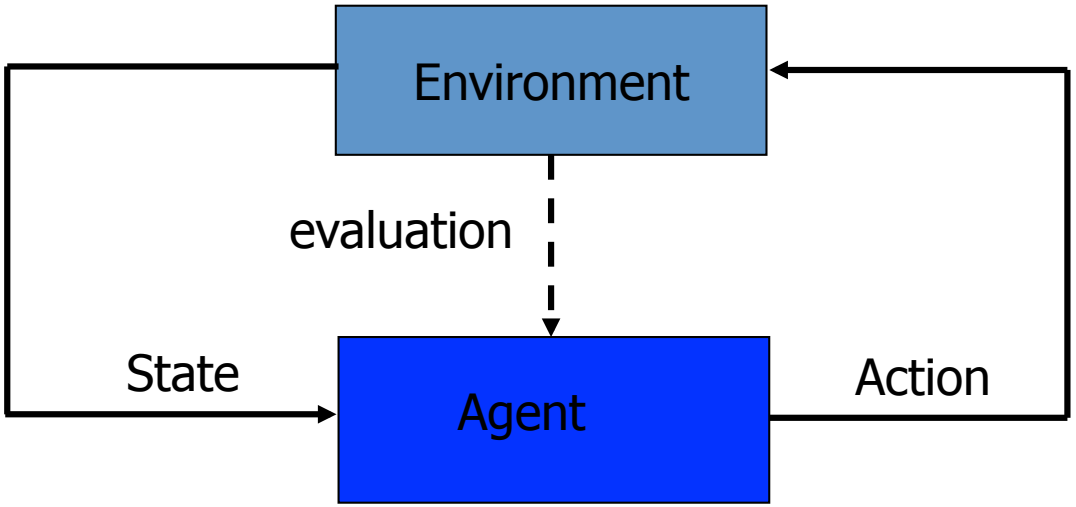
Dayan and Abbott



Reinforcement Learning

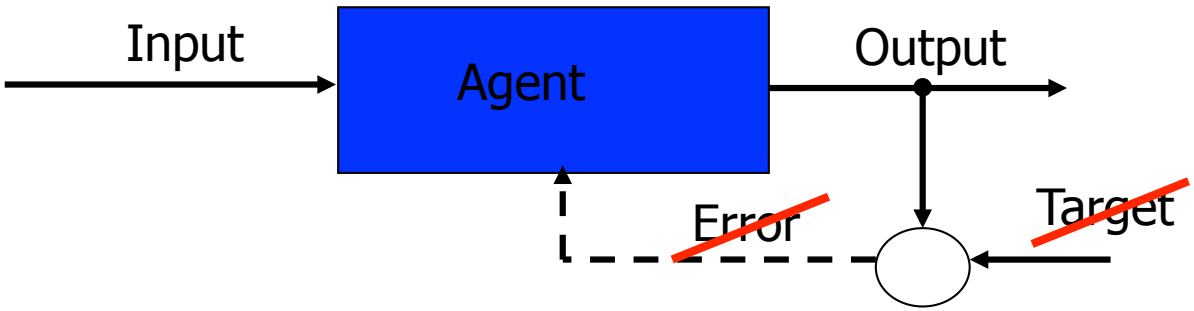
- A trial-and-error learning paradigm
  - Rewards and Punishments
- Not just an algorithm but a new paradigm in itself
- Learn about a system through interaction
- Inspired by behavioural psychology!

RL Framework



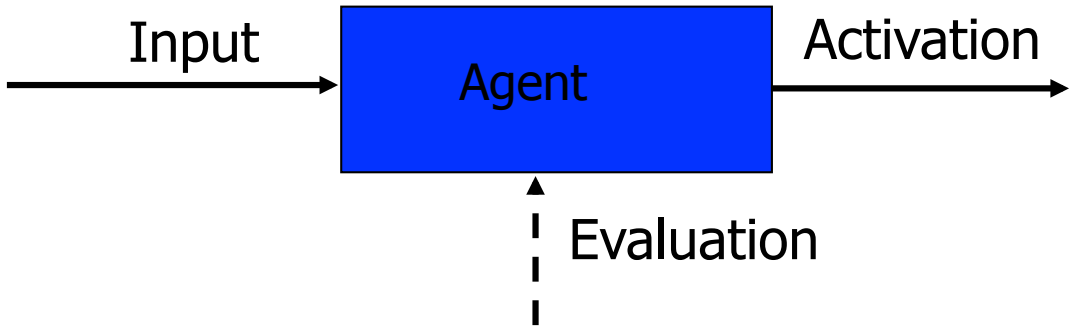
- Learn from close interaction
- Stochastic environment
- Noisy delayed scalar evaluation
- Maximize a measure of long term performance

Not Supervised Learning!



- Very sparse “supervision”
- No target output provided
- No error gradient information available
- Action chooses next state
- Explore to estimate gradient – Trail and error learning

Not Unsupervised Learning

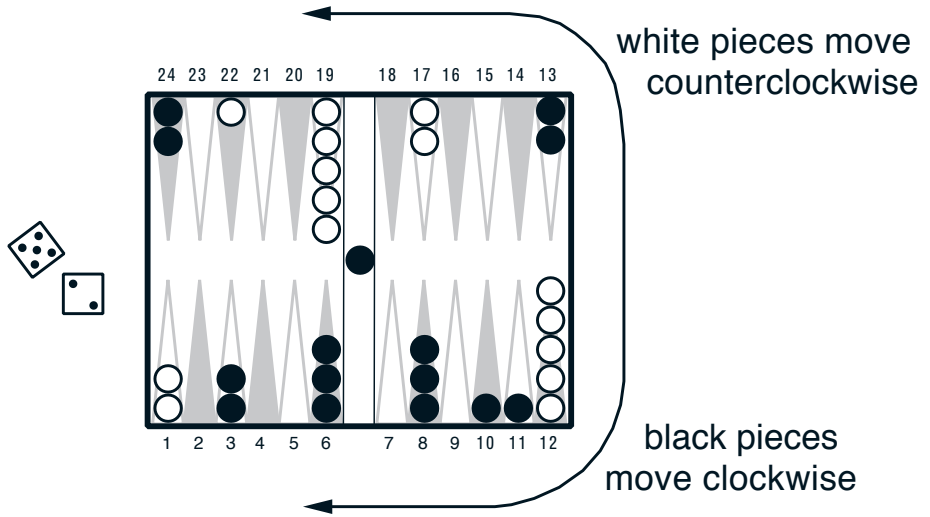


- Sparse “supervision” available
- Pattern detection not primary goal

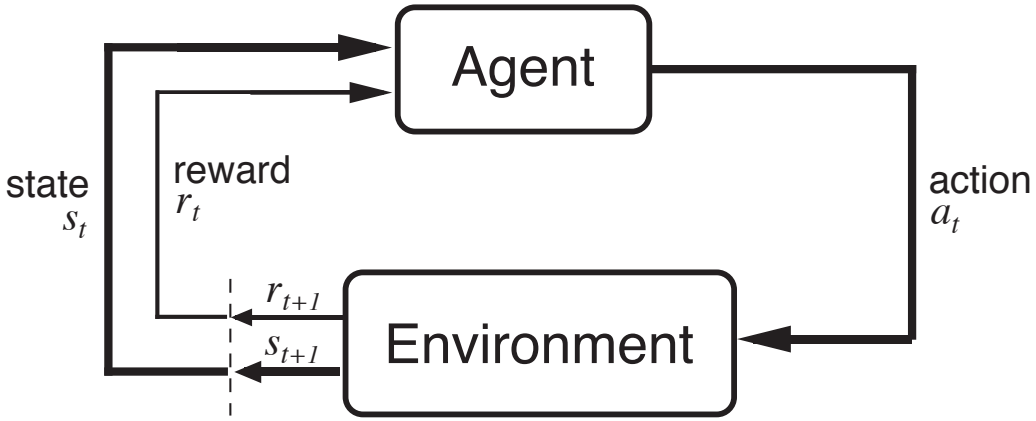
TD Gammon

Tesauro 1992, 1994, 1995, ...

- White has just rolled a 5 and a 2 so can move one of his pieces 5 and one (possibly the same) 2 steps
- Objective is to advance all pieces to points 19-24
- Hitting
- 30 pieces, 24 locations implies enormous number of configurations
- Effective branching factor of 400



The Agent-Environment Interface



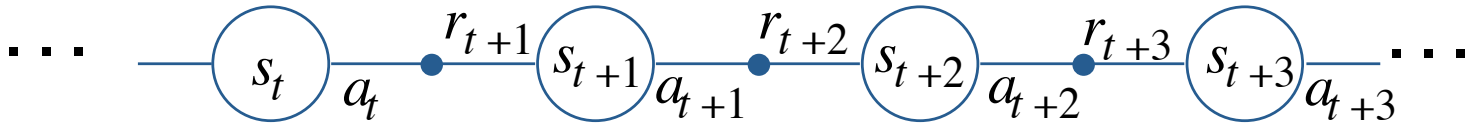
Agent and environment interact at discrete time steps: $t = 0, 1, 2, \ldots$

Agent observes state at step t: $s_t \in S$

produces action at step t : $a_t \in A(s_t)$

gets resulting reward: $r_{t+1} \in \Re$

and resulting next state: s_{t+1}



The Agent Learns a Policy

Policy at step t, π_t :

a mapping from states to action probabilities

$\pi_t(s,a) =$ probability that $a_t = a$ when $s_t = s$

- Reinforcement learning methods specify how the agent changes its policy as a result of experience.
- Roughly, the agent's goal is to get as much reward as it can over the long run.

Goals and Rewards

- Is a scalar reward signal an adequate notion of a goal?—maybe not, but it is surprisingly flexible.
- A goal should specify what we want to achieve, not how we want to achieve it.
- A goal must be outside the agent's direct control —thus outside the agent.
- The agent must be able to measure success:
  - explicitly;
  - frequently during its lifespan.

Returns

Suppose the sequence of rewards after step t is :

$$r_{t+1}, r_{t+2}, r_{t+3}, \cdots$$

What do we want to maximize?

In general,

we want to maximize the **expected return**, $E\{R_t\}$, for each step t.

Episodic tasks: interaction breaks naturally into episodes, e.g., plays of a game, trips through a maze.

$$R_t = r_{t+1} + r_{t+2} + \cdots + r_T ,$$

where T is a final time step at which a **terminal state** is reached, ending an episode.

Returns for Continuing Tasks

Continuing tasks: interaction does not have natural episodes.

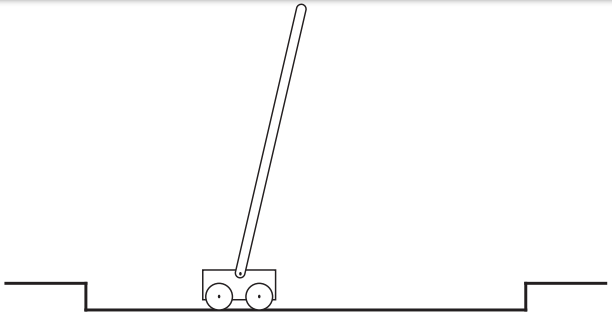
Discounted return:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1},$$

where $\gamma, 0 \leq \gamma \leq 1$, is the **discount rate**.

$$\text{shortsighted } 0 \leftarrow \gamma \rightarrow 1 \text{ farsighted}$$

An Example



Avoid **failure:** the pole falling beyond a critical angle or the cart hitting end of track.

As an **episodic task** where episode ends upon failure:

$$\text{reward} = +1 \text{ for each step before failure}$$

$$\Rightarrow \text{return} = \text{number of steps before failure}$$

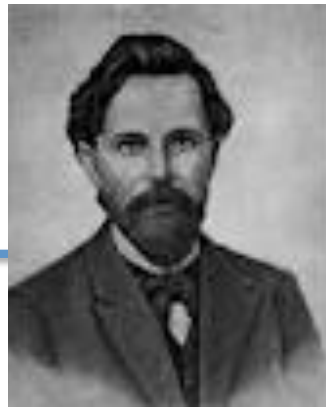
As a **continuing task** with discounted return:

$$\text{reward} = -1 \text{ upon failure; } 0 \text{ otherwise}$$

$$\Rightarrow \text{return} = -\gamma^k, \text{ for } k \text{ steps before failure}$$

In either case, return is maximized by avoiding failure for as long as possible.

The Markov Property



- "the state" at step *t*, means whatever information is available to the agent at step *t* about its environment.
- The state can include immediate "sensations", highly processed sensations, and structures built up over time from sequences of sensations.
- Ideally, a state should summarize past sensations so as to retain all "essential" information, i.e., it should have the **Markov Property**:

$$\Pr\left\{s_{t+1}=s', r_{t+1}=r \mid s_t, a_t, r_t, s_{t-1}, a_{t-1}, \ldots, r_1, s_0, a_0\right\} =$$
$$\Pr\left\{s_{t+1}=s', r_{t+1}=r \mid s_t, a_t\right\}$$

for all s', r, and histories $s_t, a_t, r_t, s_{t-1}, a_{t-1}, \ldots, r_1, s_0, a_0$.

Markov Decision Processes

- If a reinforcement learning task has the Markov Property, it is basically a Markov Decision Process (MDP).
- If state and action sets are finite, it is a finite MDP.
- To define a finite MDP, you need to give: $M = \langle S, A, P, R \rangle$
  - state and action sets
  - one-step "dynamics" defined by transition probabilities:

$$P^a_{ss'} = \Pr\{s_{t+1} = s' \mid s_t = s, a_t = a\} \quad \text{for all } s, s' \in S,\ a \in A(s).$$

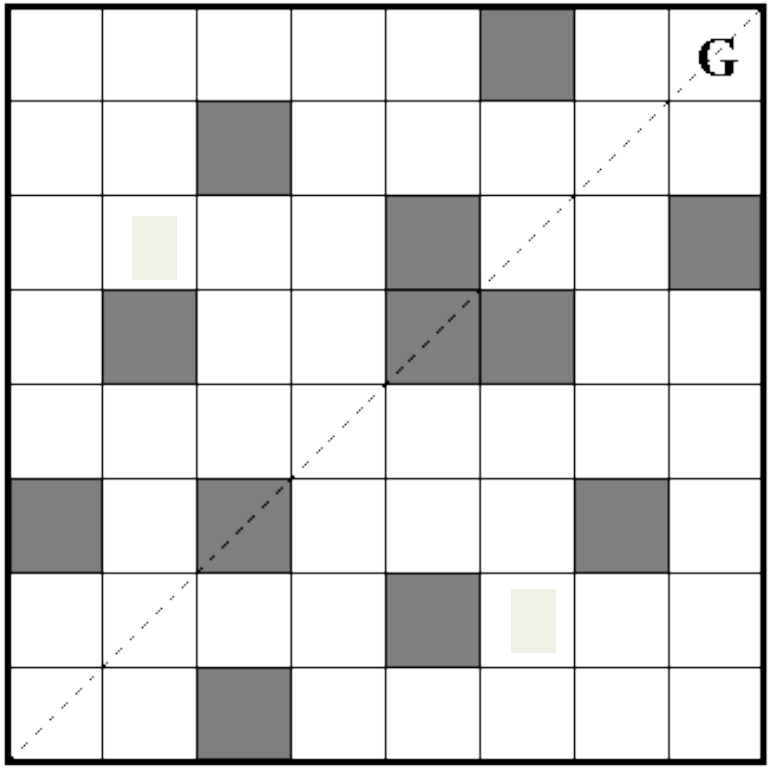
  - reward expectations:

$$R^a_{ss'} = E\{r_{t+1} \mid s_t = s, a_t = a, s_{t+1} = s'\} \quad \text{for all } s, s' \in S,\ a \in A(s).$$

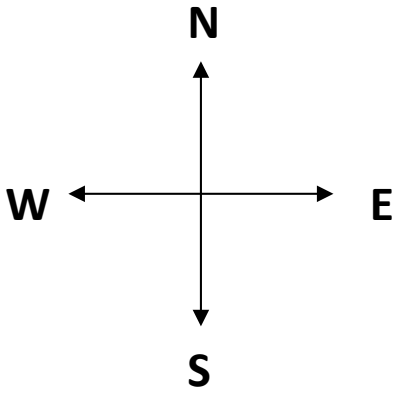
Markov Decision Processes

- MDP, *M,* is the tuple: $M = \langle S, A, \Psi, P, R \rangle$
  - S : set of states.
  - A : set of actions.
  - $\Psi \subseteq S \times A$: set of admissible state-action pairs.
  - $P : \Psi \times S \rightarrow [0,1]$: probability of transition.
  - $R : \Psi \rightarrow \Re$: expected reward.
- Policy $\pi : S \rightarrow A$ (can be stochastic)
- Maximize total expected reward.

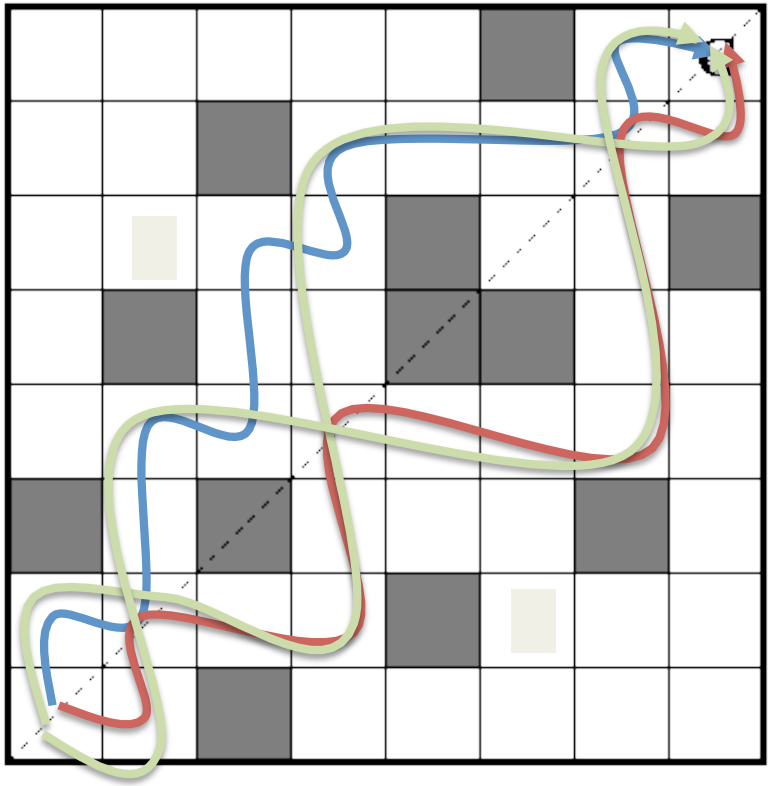
Example



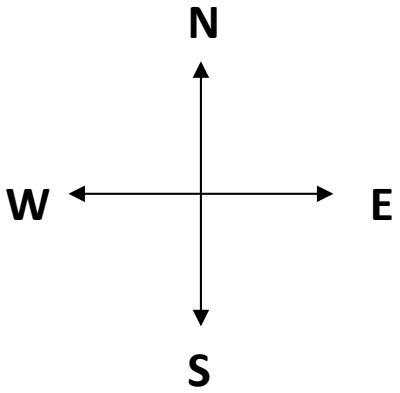
$$M = \langle S, A, \Psi, P, R \rangle$$



Optimal Policies



$$M = \langle S, A, \Psi, P, R \rangle$$



Solution Methods

- Temporal Difference Methods
  - TD(λ)
  - Q-learning
  - SARSA
  - Actor-Critic
- Policy Search
  - Policy Gradient Methods
  - Evolutionary algorithms
- Stochastic Dynamic Programming

Applications of RL

- Optimal Control
  - Robot Navigation
  - Helicopters!
  - Chemical Plants
- Combinatorial Optimization
  - Elevator Dispatching
  - VLSI placement and routing
  - Job-shop scheduling
  - Routing algorithms
  - Call admission control
- More
  - Intelligent Tutoring Systems
- Computational Neuroscience
  - Primary mechanism of learning
- Psychology
  - Behavioral and operant conditioning
  - Decision making
- Operations Research
  - Approximate Dynamic Programming
- More
  - Game Playing
  - Dialogue systems

Bandit Problems

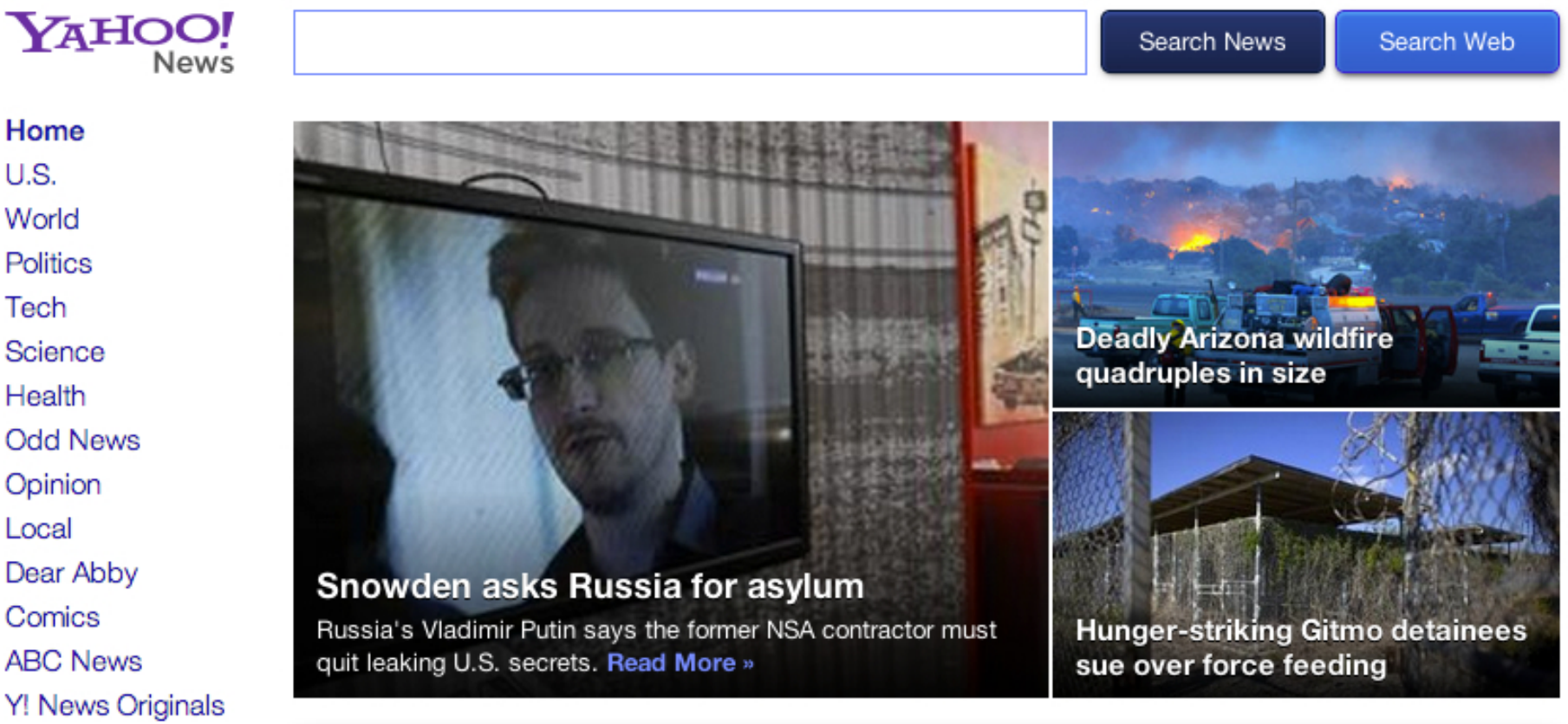
- One key question - the dilemma between exploration and exploitation
- Explore to find profitable actions
- Exploit to act according to the best observations already made
- Bandit problems encapsulate ‘Explore vs Exploit’

Problem Description

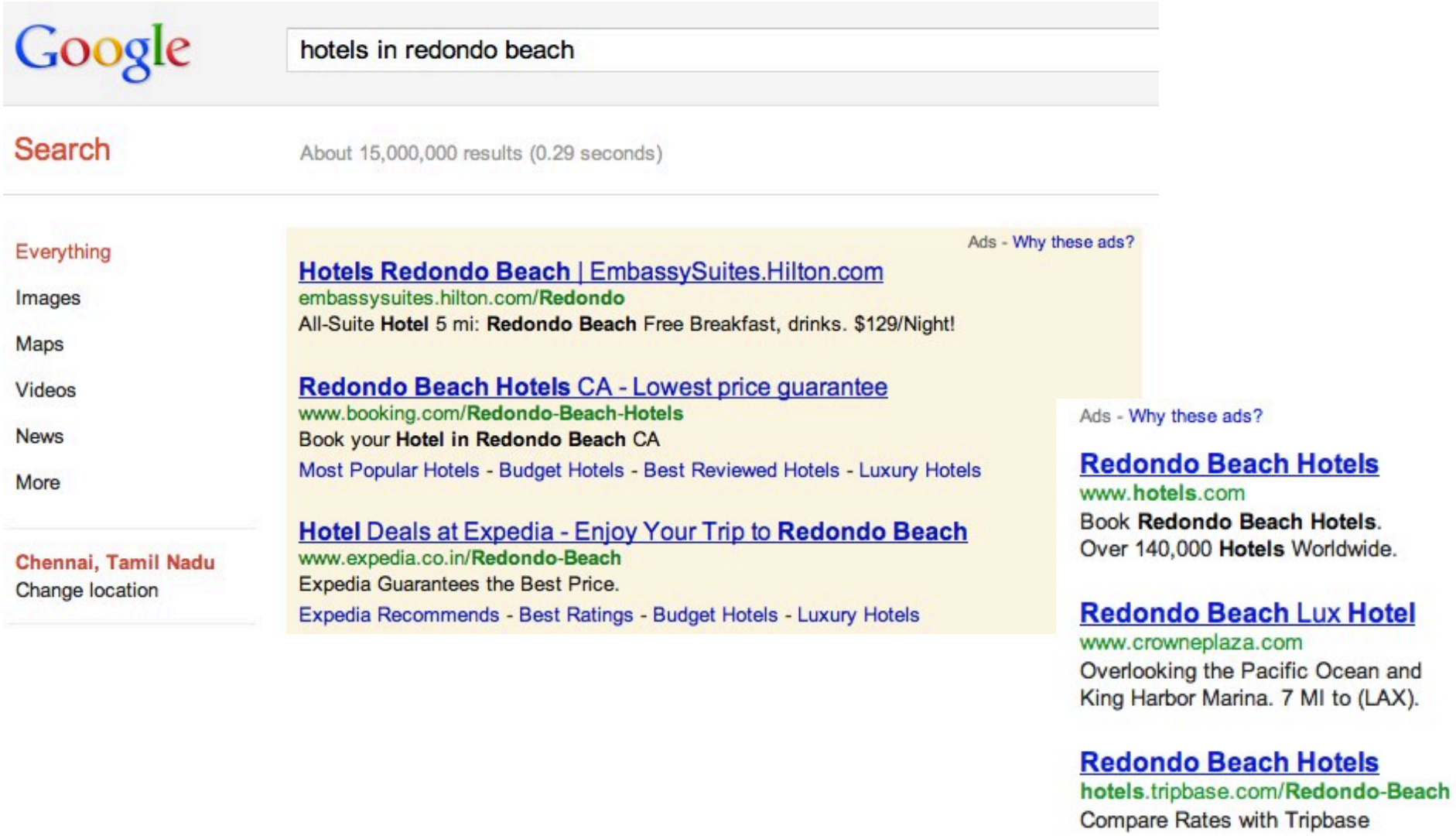
- n-arm bandit problem is to learn to preferentially select a particular action (arm) from a set of n actions $(1, 2, 3, \ldots, n)$
- Each selection results in Rewards derived from the respective probability distribution
- Arm i has a reward distribution with mean μ_i and

$$\mu^* = \max\{\mu_i\}$$

Customization



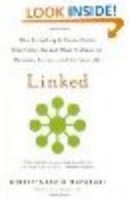
Ad Selection



Recommendation

Really?

Customers Who Bought This Item Also Bought

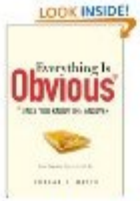


Linked: How Everything Is Connected to... by Albert-Laszlo Barabasi

★★★★☆ (116)

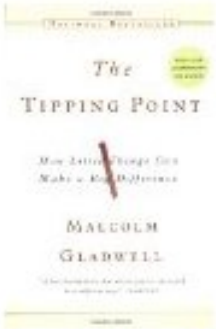
$10.38

Connected: The Surprising Power of Our... by Nicholas A. Christakis



Everything Is Obvious: *Once You Know the Answer by Duncan J. Watts

Continue Shopping: Customers Who Bought Items in Your Recent History Also Bought



The Tipping Point: How Little...

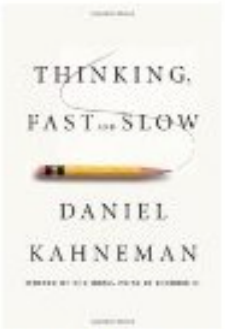
› Malcolm Gladwell

★★★★☆ (1,262)

Paperback

$9.59

Fix this recommendation



Thinking, Fast and Slow

› Daniel Kahneman

★★★★☆ (91)

Hardcover

$16.65

Fix this recommendation

Adapt: Why Success Always Starts...

› Tim Harford

★★★★½ (31)

Hardcover

$17.82

Fix this recommendation

Comment Recommendation

Bob Aboey 32 minutes ago 4 7
Too bad the waves didn't hit 100 miles south..... right into Paris!
Expand Replies (6) Reply

Anubis 32 minutes ago 8 12
so much for global warming , cold enough for ya , oh would you like colder bigger waves .
Expand Replies (5) Reply

NRAFOREVER 24 minutes ago 11 10
God is mad at the Muslims in England
Expand Replies (3) Reply

Hughes 26 minutes ago 2 4
The UK has video cameras everywhere so I hope that fat guy on his 'mobility scooter' falling into the river will be on World's Dumbest or Britain's Funniest Videos.
Expand Replies (1) Reply

Comment Recommendation

Applications

- Decision Theory, Reinforcement Learning
- Evolutionary Programming
- Multiclass prediction Kakade et al 2008.
- Google/Yahoo Ads
- Recommender Systems
- Parameter Selection: Rate control
- Wavelength Selection in WDM optical networks

Problem Description

- n-arm bandit problem is to learn to preferentially select a particular action (arm) from a set of n actions (1, 2, 3, ... , n)
- Each selection results in Rewards derived from the respective probability distribution
- Arm i has a reward distribution with mean μ_i and $\mu^* = \max\{\mu_i\}$

Objectives

- Identify the *correct* arm eventually
- Maximize the total rewards obtained
  - Minimize regret (= loss) while learning
- Probably Approximately Correct (PAC) frameworks
  - Identification of an ε -optimal arm with probability $1 - \delta$
  - ε -Optimal: Mean of the selected arm satisfies
    $$\mu > \mu^* - \varepsilon$$
  - Minimize sample complexity: Order of samples required for such an arm identification

Traditional Approaches

- Let $r_{i,k}$ be the reward sample acquired when i^{th} arm is selected for the k^{th} time
- Define:

$$\hat{\mu}_i = \frac{\sum_k r_{i,k}}{\sum_{\{k:r_{i,k}\}} 1} \qquad \hat{\mu}_* = \max_i \{\hat{\mu}_i\}$$

- Epsilon Greedy: Select arm * with probability 1- ε and select any arbitrary arm with probability ε
- Asymptotic Convergence Guarantees

Traditional Approaches - Learning Automata

- Variable structure automata
- Rich history and foundation for many algorithms
- Policy for selection of arms represented as a stochastic finite state machine
  - Modify transition/emission probabilities based on rewards obtained
- Asymptotic convergence guarantees
- Ref: Narendra and Thathachar, Learning Automata: An Introduction, 1989.

Some classical results

- Objective: Maximize total rewards
- [Lai & Robbins, 1985]
  - showed that regret should grow at least logarithmically with the no. of plays, and provided policies that attain the lower bound for specific probability distributions
- [Agrawal, 1995]
  - provided policies achieving the logarithmic bounds incorporating sample means that are computationally more efficient
- There are also many heuristics like epsilon greedy, Softmax
  - Without any guarantees on the sample complexity or regret while learning but do well in practice!

State-of-the-art (1)

- [Even-Dar et al., 2006] – Median Elimination Algorithm
  - provided another quantification of the objective measuring quickness in determining the best arm
  - A Probably Approximately Correct (PAC) framework
  - Best known sample complexity

- This is further extended to finding m arms that are epsilon-optimal with high probability in [Kalyanakrishnan & Stone, 2010]

Naïve Algorithm

Input : $\varepsilon > 0, \delta > 0$

Output : An arm

foreach Arm $a \in A$ **do**

Sample it $\ell = \frac{4}{\varepsilon^2} \ln\left(\frac{2n}{\delta}\right)$ times;

Let $\hat{p}_a$ be the average reward of arm a;

end

Output $a' = \arg\max_{a \in A} \{\hat{p}_a\}$;

- This obviously has a sample complexity of $O(n/\varepsilon^2)$ and $O(\ln(n/\delta))$
- The difficulty is to show that it achieves the desired performance guarantees

Proving Correctness

Chernoff-Hoeffding Bounds

[Hoeffding, 1963]Let $X_1, X_2, \ldots X_n$ be random variables with common range $[0, 1]$ and such that $E[X_t|X_1, \ldots, X_{t-1}] = \mu$ for $1 \leq t \leq n$. Let $S_n = \frac{X_1+\cdots+X_n}{n}$. Then for all $a \geq 0$ we have the following,

$$P\{S_n \geq \mu + a\} \leq e^{-2a^2 n}$$

$$P\{S_n \leq \mu - a\} \leq e^{-2a^2 n}$$

Proving Correctness

- Need to show that the Probability that the wrong arm is selected by the naïve algorithm is bounded, i.e. $P(\hat{p}_{a'} > \hat{p}_{a^*})$ where a' is an arm ε away from the best.

$$\begin{aligned} P(\hat{p}_{a'} > \hat{p}_{a^*}) &\leq P\left(\hat{p}_{a'} > \mathbb{E}[R(a')] + \varepsilon/2 \text{ or } \hat{p}_{a^*} < r^* - \varepsilon/2\right) \\ &\leq P\left(\hat{p}_{a'} > \mathbb{E}[R(a')] + \varepsilon/2\right) + P\left(\hat{p}_{a^*} < r^* - \varepsilon/2\right) \\ &\leq 2\exp(-2(\varepsilon/2)^2 \ell), \end{aligned}$$

- Choosing $\ell = (2/\varepsilon^2)\ln(2n/\delta)$ we get

$$P(\hat{p}_{a'} > \hat{p}_{a^*}) \leq \delta/n$$

Median Elimination Algorithm

- The problem with the Naïve algorithm is that the complexity depends on $log\ n/\delta$
- We would hope for $O(n)$ plus factors dependent on ε and δ
- Evan-Dar et al. devised an action elimination procedure
- Basis for many other algorithms
- Key Idea: Eliminate some no. of arms after each round of sampling.

Median Elimination Algorithm – MEA

- [Even-Dar et al., 2002, 2006]

Input : $\varepsilon > 0, \delta > 0$

Output : An arm

Set $S_1 = A$, $\varepsilon_1 = \varepsilon/4$, $\delta_1 = \delta/2$, $\ell = 1$. **repeat**

Sample every arm $a \in S_\ell$ for $1/(\varepsilon_\ell/2)^2 \log(3/\delta_\ell)$ times, and let $\hat{p}_a^\ell$ denote its empirical value;

Find the median of $\hat{p}_a^\ell$, denoted by m_ℓ;

$S_{\ell+1} = S_\ell \setminus \{a : \hat{p}_a^\ell < m_\ell\}$;

$\varepsilon_{\ell+1} = \frac{3}{4}\varepsilon_\ell$; $\delta_{\ell+1} = \delta_\ell/2$; $\ell = \ell + 1$;

until $|S_\ell| = 1$;

Median Elimination Algorithm

- With some work we can show that MEA is (ε,δ) PAC with sample complexity

$$O\left(\frac{n}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\right)$$

- Bound probability of eliminating optimal arm in each round

More Results

- Markov Inequality: $\Pr(X \geq a) \leq \frac{\mathrm{E}(X)}{a}.$
- Union Bound $\mathbb{P}\left(\bigcup_i A_i\right) \leq \sum_i \mathbb{P}(A_i).$

Proof

Falls by utmost ε_l

Lemma *For the* Median Elimination(ε, δ) *algorithm we have that for every phase* ℓ*:*

$$\mathbf{P}\left[\max_{j \in S_\ell} p_j \leq \max_{i \in S_{\ell+1}} p_i + \varepsilon_\ell\right] \geq 1 - \delta_\ell.$$

Proof

$$E_1 = \{\hat{p}_1 < p_1 - \varepsilon_1/2\} \qquad \mathbf{P}[E_1] \leq \delta_1/3$$

If E_1 doesn't hold, then some sub-optimal p_j could be the best estimated arm.

$$\mathbf{P}\left[\hat{p}_j \geq \hat{p}_1 \mid \hat{p}_1 \geq p_1 - \varepsilon_1/2\right] \leq \mathbf{P}\left[\hat{p}_j \geq p_j + \varepsilon_1/2 \mid \hat{p}_1 \geq p_1 - \varepsilon_1/2\right] \leq \delta_1/3$$

$$\mathbf{E}[\#\text{bad} \mid \hat{p}_1 \geq p_1 - \varepsilon_1/2] \leq n\delta_1/3$$

$$\mathbf{P}[\#\text{bad} \geq n/2 \mid \hat{p}_1 \geq p_1 - \varepsilon_1/2] \leq \frac{n\delta_1/3}{n/2} = 2\delta_1/3$$

Markov Inequality

Proof

Lemma *The sample complexity of the* Median Elimination(ε, δ) *is* $O\left((n/\varepsilon^2)\log(1/\delta)\right)$.

1. $\delta_1 = \delta/2 \;;\; \delta_\ell = \delta_{\ell-1}/2 = \delta/2^\ell$
2. $n_1 = n \;;\; n_\ell = n_{\ell-1}/2 = n/2^{\ell-1}$
3. $\varepsilon_1 = \varepsilon/4 \;;\; \varepsilon_\ell = \frac{3}{4}\varepsilon_{\ell-1} = \left(\frac{3}{4}\right)^{\ell-1}\varepsilon/4$

Therefore we have

$$\begin{aligned}
\sum_{\ell=1}^{\log_2(n)} \frac{n_\ell \log(3/\delta_\ell)}{(\varepsilon_\ell/2)^2} &= 4\sum_{\ell=1}^{\log_2(n)} \frac{n/2^{\ell-1}\log(2^\ell 3/\delta)}{((\frac{3}{4})^{\ell-1}\varepsilon/4)^2} \\
&= 64\sum_{\ell=1}^{\log_2(n)} n(\frac{8}{9})^{\ell-1}(\frac{\log(1/\delta)}{\varepsilon^2} + \frac{\log(3)}{\varepsilon^2} + \frac{\ell\log(2)}{\varepsilon^2}) \\
&\leq 64\frac{n\log(1/\delta)}{\varepsilon^2}\sum_{\ell=1}^{\infty}(\frac{8}{9})^{\ell-1}(\ell C' + C) = O(\frac{n\log(1/\delta)}{\varepsilon^2})
\end{aligned}$$

Median Elimination Algorithm

- So MEA is PAC with sample complexity

$$O\left(\frac{n}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\right)$$

- Got rid of the n in the logarithm
- But in practice PAC guarantees are weak
- Also interested in transient performance
  - *Regret analysis*

UCB

Auer et al, ICML 1998

- Upper Confidence Bounds (UCB): The arm with the best estimate r* so far serves as a benchmark, and other arms are played only if the upper bound of a suitable confidence interval is at least r*
- Sub-optimal arm *j* played fewer than $(8/\Delta_j)\ ln\ n$ times
  - Further improvents focus on reducing the constants

Deterministic policy: UCB1.
Initialization: Play each machine once.
Loop:

- Play machine j that maximizes $\bar{x}_j + \sqrt{\frac{2 \ln n}{n_j}}$, where $\bar{x}_j$ is the average reward obtained from machine j, n_j is the number of times machine j has been played so far, and n is the overall number of plays done so far.

UCB Revisited

Auer P. and Ortner R., 2010

- An action elimination procedure
- Theoretically provable improved regret bounds
- But in practice we observe that UCB performs better, especially with fewer arms
  - If reward distributions well separated even epsilon greedy works well

UCB-Revisited

Input: A set of arms A, the horizon T.

Initialization: Set $\tilde{\Delta}_0 := 1$, and $B_0 := A$.

For rounds $m = 0, 1, 2, \ldots, \left\lfloor \frac{1}{2} \log_2 \frac{T}{e} \right\rfloor$ **do:**

Arm selection:

If $|B_m| > 1$, choose each arm in B_m until the total number of times it has been chosen is $n_m := \left\lceil \frac{2 \log(T\tilde{\Delta}_m^2)}{\tilde{\Delta}_m^2} \right\rceil$. Otherwise choose the single arm in B_m until step T is reached.

Arm elimination:

Delete all arms i from B_m for which

$$\left\{ \hat{r}_i + \sqrt{\frac{\log(T\tilde{\Delta}_m^2)}{2n_m}} \right\} < \max_{j \in B} \left\{ \hat{r}_j - \sqrt{\frac{\log(T\tilde{\Delta}_m^2)}{2n_m}} \right\}$$

in order to obtain B_{m+1}. Here $\hat{r}_j$ is the average reward obtained from arm j.

Reset $\tilde{\Delta}_m$:

Set $\tilde{\Delta}_{m+1} := \frac{\tilde{\Delta}_m}{2}$.

Thomson Sampling for Bandits

- Choose an arm to play based on probability of being best arm
- Truly Bayesian approach
  - For reward support of [0,1] use a beta distribution as prior
- Achieves *log t* lower bound, but poorer dependence on Δ than UCB
- Chappelle and Li 2011, Shipra Agrawal and Navin Goyal 2012.
- Lot of recent interest

Algorithm 1 Thompson Sampling for Bernoulli bandits

For each arm $i = 1, \ldots, N$ set $S_i = 0, F_i = 0$.

foreach $t = 1, 2, \ldots,$ **do**

  For each arm $i = 1, \ldots, N$, sample $\theta_i(t)$ from the Beta$(S_i + 1, F_i + 1)$ distribution.

  Play arm $i(t) := \arg\max_i \theta_i(t)$ and observe reward r_t.

  If $r = 1$, then $S_{i(t)} = S_{i(t)} + 1$, else $F_{i(t)} = F_{i(t)} + 1$.

end

Contextual Bandits

- Different ads for different users
  - One bandit for each user!
- Hard to train
  - Need several rounds of experience with same user
  - Typically users share features
    - Demographic
    - Browsing history
    - Location, etc.
  - "Arms" share features too!

Contextual Bandits

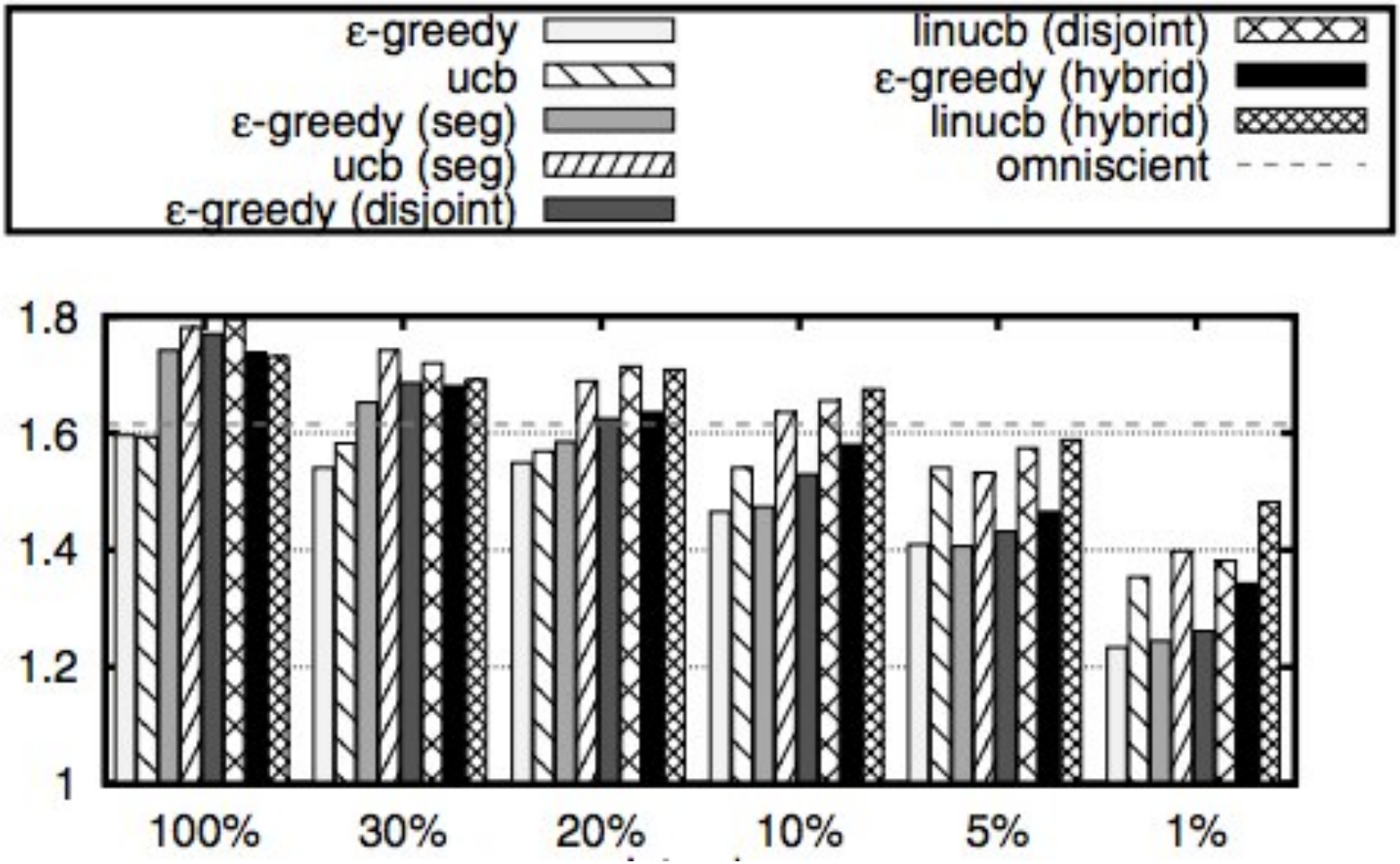
- Assume that each user is represented by a set of features
  - Can be joint features of user and arm
- The "statistic" used for choosing arms is now dependent on these features
- Simplest case is to consider that for each feature combination you have a different bandit and once a choice is made there is a stochastic transition to another bandit
  - Earlier work on associative bandits
  - Typically use a small number of *states* to encode features

LinUCB

Li et al., WWW10

- One of the more popular contextual bandit algorithms
- *Predicted expected reward* assumed to be a linear function of the features
  - Use ridge regression to fit parameters
  - Can derive upper confidence bounds for the regression fit
  - Use UCB like action selection
  - Gives better performance with lesser "training" data
- Variants: Do not share parameters across arms

Improving CTRs



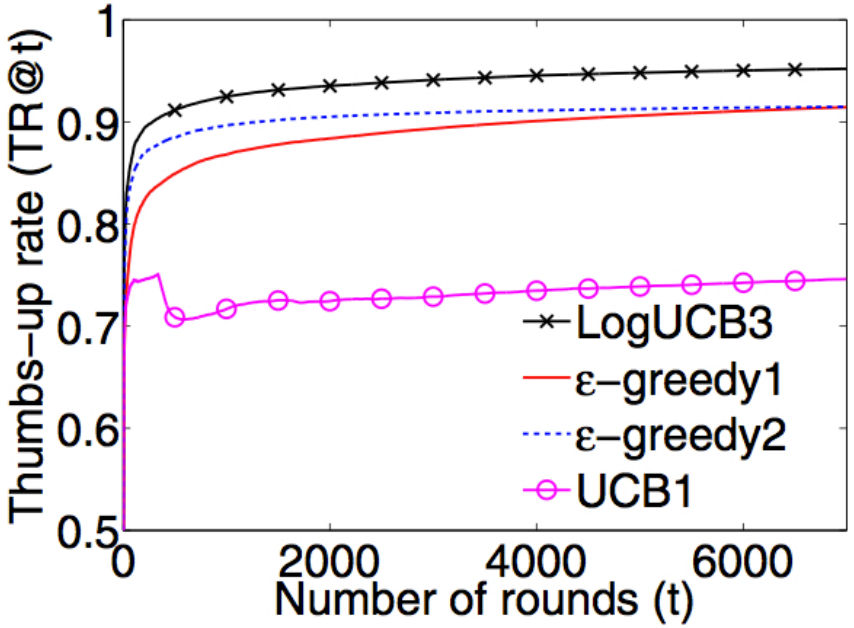
Comment Recommendation

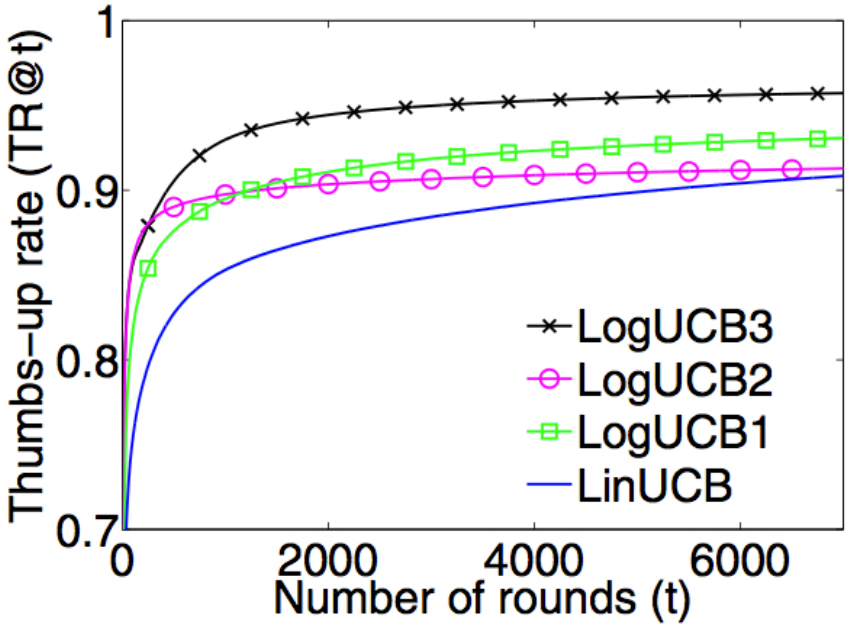
LogUCB

Mahajan et al., CIKM 2012

- Use thumbs up and down as *rewards*
- Very sparse
- Generalize using features of text
  - Topic models
  - Lexical features
  - Keywords
- Logistic regression
  - Probability of thumbs up

Results


Thumbs−up rate (TR@t)
1
0.9
0.8
0.7
0.6
0.5
0
2000
4000
6000
Number of rounds (t)
LogUCB3
ε−greedy1
ε−greedy2
UCB1


Thumbs−up rate (TR@t)
1
0.9
0.8
0.7
0
2000
4000
6000
Number of rounds (t)
LogUCB3
LogUCB2
LogUCB1
LinUCB

Fractional Moment Bandits

Anandanarayanan, and R. UAI 2011.

- Pair-wise comparison of arms
  - Use a fractional moment as the statistic
  - Family shown to possess algorithms with PAC guarantees of $O(n), O(\ln^2 \frac{1}{\delta}), O(\frac{1}{\varepsilon^{4/n}})$ complexity in finding an ε-optimal arm
  - Regret guarantees of $O(\log t)$
    - Constants not competitive with UCB
  - Can "optimize" regret for a given (ε,δ) PAC guarantee

Future

- More powerful approximations
  - GP-UCB, Srinivas et al., ICML 2010
  - Thomson Sampling for linear contextual bandits, Agrawal & Goyal, ICML 2013
- Dependencies between samples
  - Multi-slot bandits
- Extensions to *full* reinforcement learning problem
  - PAC MDP Littman et al.
  - Optimal exploration Auer et al.
- Partial extensions
  - Budgets

References

- ICML 2011 Tutorial on Bandits

  https://sites.google.com/site/banditstutorial/
- ICML 2010 and KDD 2010 Tutorial on Learning through exploration

  http://hunch.net/~exploration_learning/
- ICML 2011 Tutorial on Large Scale Recommender Systems

  http://pages.cs.wisc.edu/~beechung/icml11-tutorial/

  – Some coverage of bandits for recommender systems

Questions?

http://www.cse.iitm.ac.in/~ravi