Multi-armed bandit problem and its applications in reinforcement learning

Pietro Lovato

Ph.D. Course on Special Topics in AI: Intelligent Agents and Multi-Agent Systems

Overview

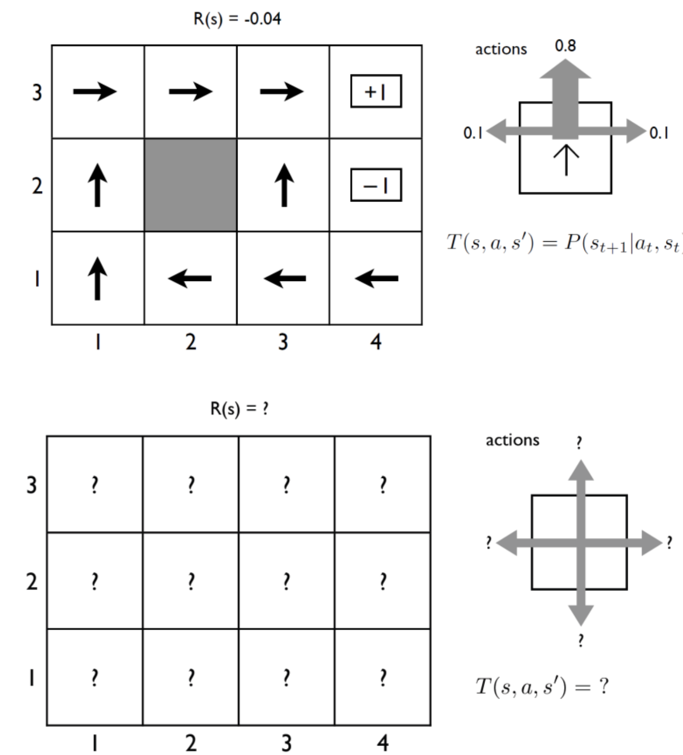
- Introduction: Reinforcement Learning
- Multi-armed bandit problem
  - Heuristic approaches
  - Index-based approaches
  - UCB algorithm
- Applications
- Conclusions

Reinforcement learning

- Reinforcement learning is learning what to do - how to map situations to actions - so as to maximize a numerical reward signal.
- The learner is not told which actions to take, as in most forms of machine learning, but instead must discover which actions yield the most reward by trying them.
- In the most interesting and challenging cases, actions may affect not only the immediate reward, but also the next situation and, through that, all subsequent rewards.

Reinforcement learning

- Supervised learning:
  - Learning from examples provided by some knowledgeable external supervisor
  - Not adequate for learning from interaction
- Reinforcement learning:
  - no teacher; the only feedback is the reward obtained after doing an action
  - Useful in cases of significant uncertainty about the environment



The multi-armed bandit problem

- Maximize the reward obtained by successively playing gamble machines (the ‘arms’ of the bandits)

- Invented in early 1950s by Robbins to model decision making under uncertainty when the environment is unknown

- The lotteries are unknown ahead of time

Reward X_1 Reward X_2 Reward X_3

Assumptions

Each machine i has a different (unknown) distribution law for rewards with (unknown) expectation μ_i :

- Successive plays of the same machine yeald rewards that are independent and identically distributed
- Independence also holds for rewards across machines

More formally

- Reward = random variable $X_{i,n}$; $1 \leq i \leq K, n \geq 1$
- i = index of the gambling machine
- n = number of plays
- μ_i = expected reward of machine i.

A *policy*, or *allocation strategy*, A is an algorithm that chooses the next machine to play based on the sequence of past plays and obtained rewards.

Some considerations

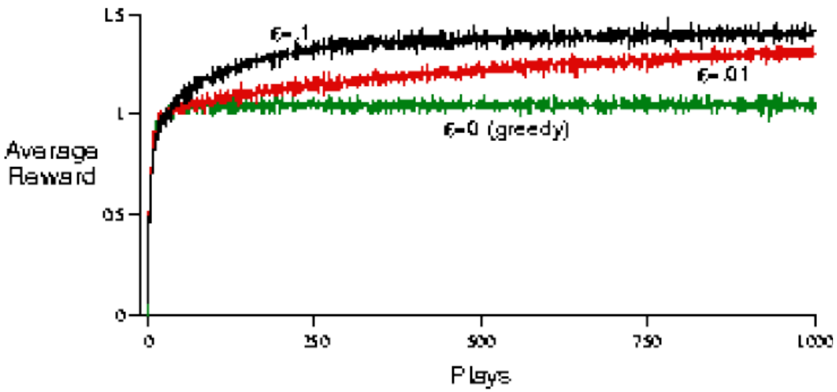
- If the expected reward is known, then it would be trivial: just pull the lever with higher expected reward.
- But what if you don't?
- Approximation of reward for a gambling machine i : average of the rewards received so far from i

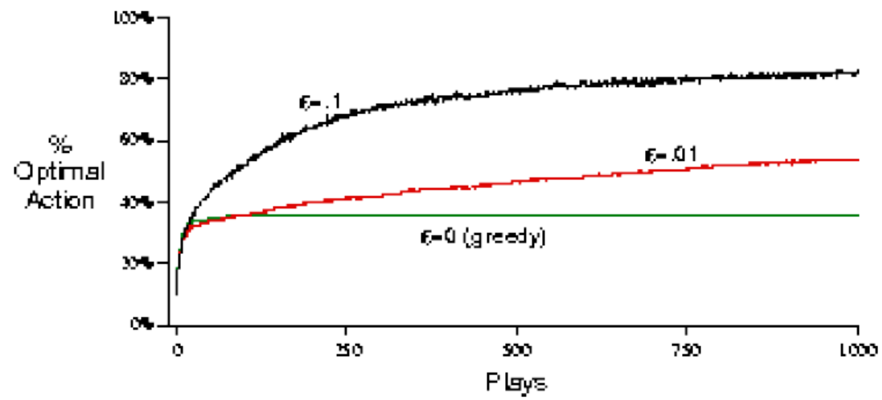
Some simple policies

- Greedy policy: always choose the machine with current best expected reward
- Exploitation vs exploration dilemma:
  - Should you exploit the information you've learned or explore new options in the hope of greater payoff?
- In the greedy case, the balance is completely towards exploitation

Some simple policies

- Slight variant: ε-greedy algorithm
  - Choose machine with current best expected reward with probability $1 - \varepsilon$
  - choose another machine randomly with probability $\varepsilon \,/\, (K - 1)$





Results on a 10-armed bandit test, averages over 2000 tasks

Performance measures of bandit algorithms

Total expected regret (after T plays):

$$R_T = \mu^* \cdot T - \sum_{j=1}^{K} \mu_j \cdot \mathbb{E}\left[T_j(T)\right]$$

μ^*: machine with highest reward expectation

$\mathbb{E}\left[T_j(T)\right]$: expectation about the number of times the policy will play machine j

Performance measures of bandit algorithms

- An algorithm is said to *solve the multi-armed bandit problem* if it can match this lower bound: $R_T = O(\log T)$.

- In other words, if it can be proved that the optimal machine is played exponentially more often (as the number of plays goes to infinity) than any other machine

The UCB algorithm

- At each time n, select an arm j s.t. $j = \operatorname*{argmax}_j B_{j,n_j,T}$

$$B_{j,n_j,T} \stackrel{\text{def}}{=} \frac{1}{n_j} \sum_{s=1}^{n_j} X_{j,s} + \sqrt{\frac{2\log(T)}{n_j}}$$

- n_j : number of times arm j has been pulled
- Sum of an exploitation term and an exploration term

The UCB algorithm

- Intuition: Select an arm that has a high probability of being the best, given what has been observed so far
- The B-values are *upper confidence bounds* on μ_j
- Assures that the optimal machine is played exponentially more often than any other machine
- Finite time-bound for regret

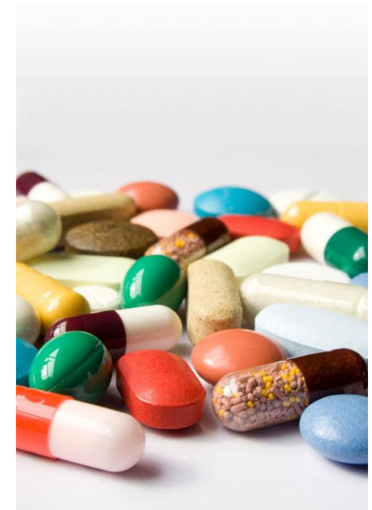
The UCB algorithm

- Many variants have been proposed:
  - Which consider the variance of the rewards obtained
  - Tuned if the distribution of rewards can be approximated as gaussian
  - Adopted if the process is non-stationary
  - ….

Some applications

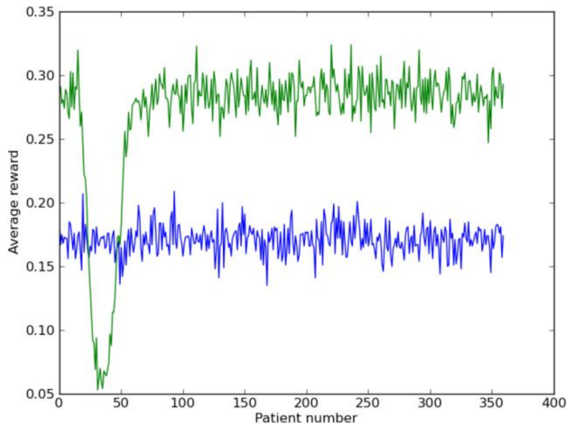
- Many applications have been studied:
  - Clinical trials
  - Adaptive routing in networks
  - Advertising: what ad to put on a web-page?
  - Economy: auctions
  - Computation of Nash equilibria

Design of ethical clinical trials

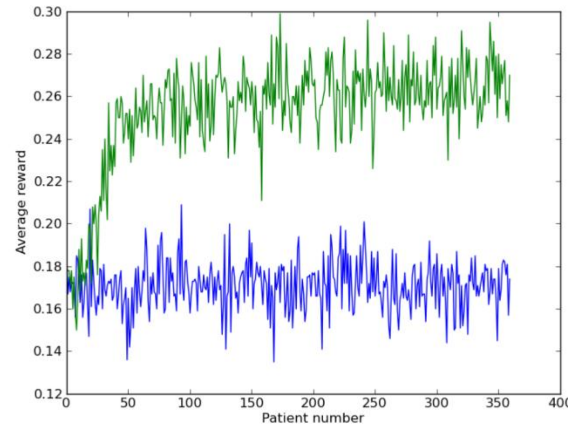


- Goal: evaluate K possible treatments for a disease
- Which one is the most effective?
  - Pool of T subjects partitioned randomly into K groups
  - Resource to allocate: partition of the subjects
    - In later stages of the trial, a greater fraction of the subjects should be assigned to treatments which have performed well during the earlier stages of the trial
  - Reward: 0-1 if the treatment is successful or not

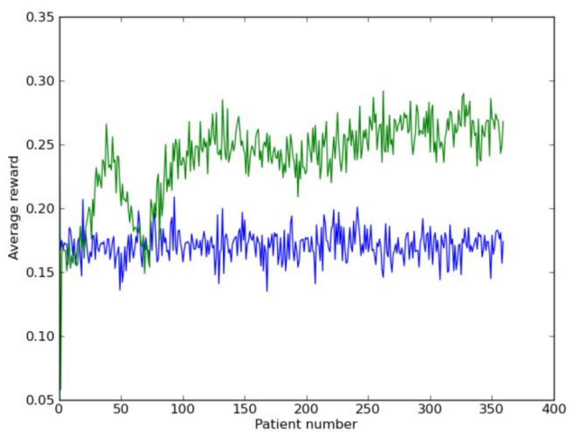
Design of ethical clinical trials



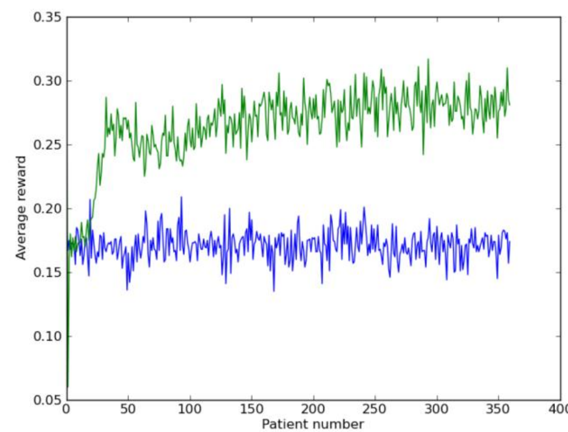
(a) ϵ-greedy



(b) Softmax



(c) UCB1



(d) UCB1-Tuned

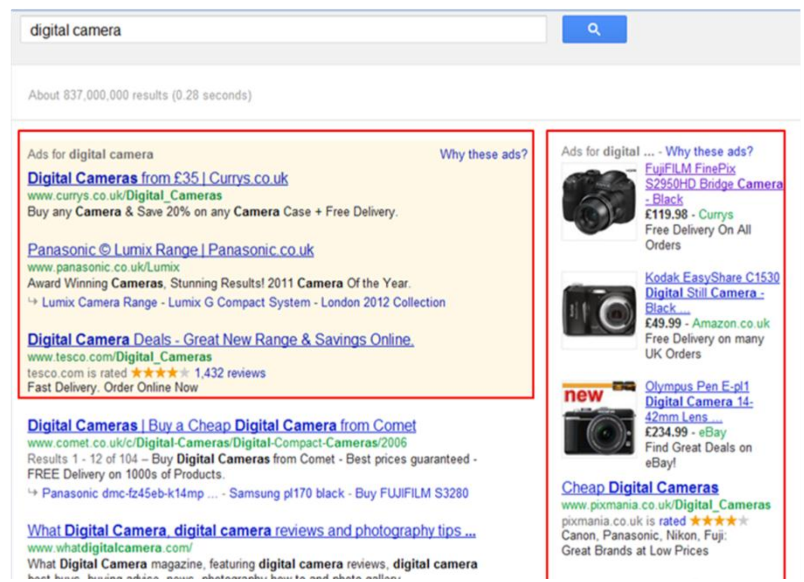
Design of ethical clinical trials

Algorithm	Average number of patients treated
Randomization	154.2
Epsilon Greedy	235.6
Softmax	239.2
UCB1	227.9
UCB-Tuned	240.7

[V. Kuleschov et al., "Algorithms for the multi-armed bandit problem", *Journal of Machine Learning Research* 2000]

Internet advertising

- Each time a user visits the site you must choose to display one of K possible advertisements
- Reward is gained if a user click on it
- No knowledge of the user, the ad content, the web page content required...
- T = users accessing your website

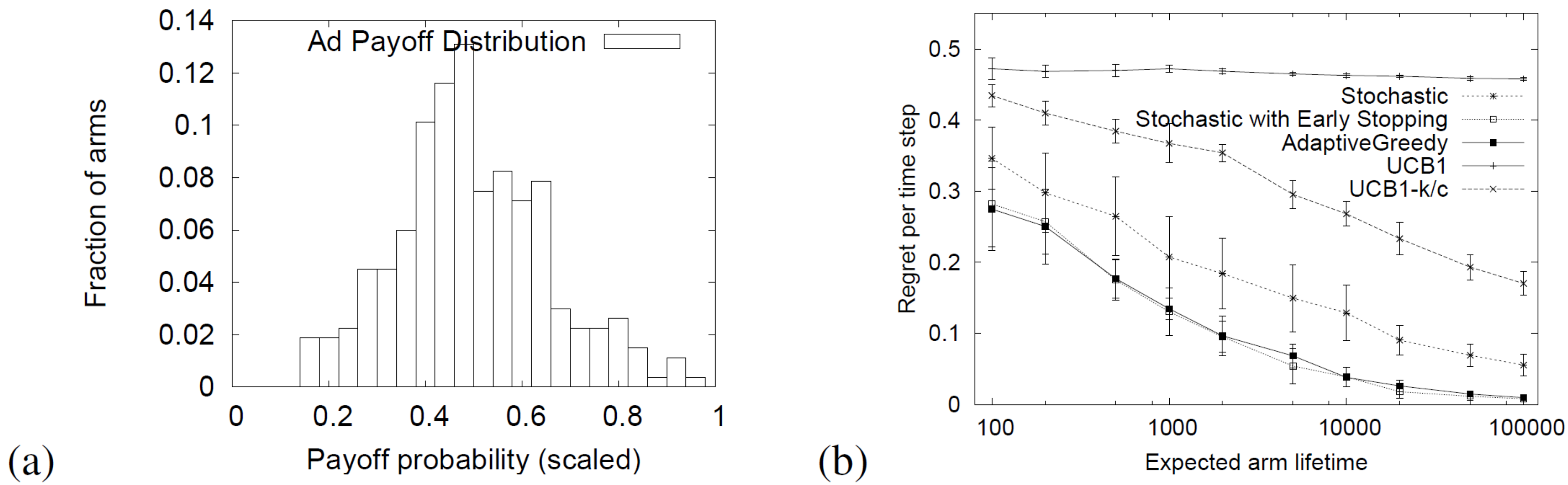


Internet advertising

- Where it fails: each of these displayed ads should be in the context of a search or other webpage
- Solution proposed: *contextual bandits*
- Context: user's query
- E.g. if a user input "flowers", choose only between flower ads
- Combination of supervised learning and reinforcement learning

[Lu et al., "Contextual multi-armed bandits",
13th International Conference on Artificial Intelligence and Statistics (AISTATS), 2010]

Internet advertising



[Lu et al., "Contextual multi-armed bandits",
13th International Conference on Artificial Intelligence and Statistics (AISTATS), 2010]

Network server selection

- A job has to be processed to one of several servers
- Servers have different processing speed (due to geographic location, load, …)
- Each server can be viewed as an arm
- Over time, you want to learn which is the best arm to play
- Used in routing, DNS server selection, cloud computing, …

Take home message

- Bandit problem: starting point for many application and context-specific tasks
- Widely studied in the literature, both from the methodological and the applicative perspective
- Still lots of open problems:
  - Exploration/exploitation dilemma
  - Theoretical proofs for many algorithms
  - Optimization in finite-time domain

Bibliography

1. [P. Auer, N. Cesa-Bianchi, P. Fischer, "Finite-time analysis of the multiarmed bandit problem", *Machine Learning,* 2002]
2. [R. Sutton, A. Barto, "Reinforcement Learning, an introduction. ', MIT Press, 1998']
3. [R. Agrawal, "Sample mean based index policies with O(log n) regret for the multi-armed bandit problem", *Advances in applied probability*, 1995]
4. [V. Kuleschov et al., "Algorithms for the multi-armed bandit problem", *Journal of Machine Learning Research,* 2000]
5. [D. Chakrabarti et al., "Mortal multi-armed bandits", *NIPS*, 2008]
6. Lu et al., "Contextual multi-armed bandits", *13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010]