Decision Making for Robots and Autonomous Systems

Fall 2015

George Konidaris
gdk@cs.duke.edu

presentation follows Sutton and Barto some excerpts from there
Multi-Arm Bandits

Direct model of:
- Actions (but no “consequences”)
- Stochastic outcomes
- Unknown utility
n-Arm Bandit

An $n$-armed bandit is defined as follows:

- Set of $n$ actions: $A = \{A_1, \ldots, A_n\}$
- Reward/utility distribution for each action: $P(r | A_i)$
- An agent lifetime, $k$.

At each time $t$:

- The agent selects action $A_t$
- The agent receives reward $r_t \sim P(r | A_t)$ (unknown)
- The agent wishes to maximize reward over its lifetime:

$$\sum_{t=1}^{k} r_t$$

(in expectation)
Example

Action 1: \( \mu = 0.5, \sigma = 0.1 \)
Action 2: \( \mu = 0.3, \sigma = 0.2 \)

0.39 0.46 0.09 0.66 0.48 0.39 0.46 0.56 0.1 0.26 0.32 0.31

A1 A2 A2 A1 A1 A2 A1 A1 A2 A2 A1 A2
Application Scenario 1
Application Scenario II

Google.com

amazon.com
Solution Form

What does a solution look like?

A policy specifies a distribution over actions to choose.

$$\pi : P(A_i)$$

What should the policy be a function of?
How should it choose actions?
Solution Form

Let’s define:

\[ Q(A_i) = \mathbb{E}[r_i] \]

If we knew \( Q \) exactly then, at every step \( t \), pick:

\[ \pi(A_i) = \begin{cases} 1 & i = \arg \max_i Q(A_i) \\ 0 & \text{otherwise} \end{cases} \]

This is a **stationary policy** (does not depend on \( t \)).
But

The agent does not know $Q$.

So?

- Take some actions to gather data.
- Estimate $Q$.

Core issue: balancing

- *exploitation* (taking actions based on maximal $Q$)
- *exploration* (getting data about other actions)
Estimating Q

First though, ignoring how data is obtained, how might we estimate Q given data?

Direct approach:
For each action $A_i$
  - Get $m$ samples $r_i^1, \ldots, r_i^m$
  - Estimate $Q$ as average:

$$Q(A_i) = \frac{\sum_n r_i^n}{m}$$

Incremental update:

$$Q(A_i)^j = Q(A_i)^{j-1} + \frac{1}{j} (r_i^j - Q(A_i)^{j-1})$$
What if Q changes?

This is known as a non-stationary problem.

Assume that Q changes relatively slowly. Update using learning rate $0 < \alpha \leq 1$:

$$Q(A_i)^j = Q(A_i)^{j-1} + \alpha (r^j_i - Q(A_i)^{j-1})$$

Converges provided $\sum_{k=1}^{\infty} \alpha_k(a) = \infty$ and $\sum_{k=1}^{\infty} \alpha_k^2(a) < \infty$. 
Now: Policies

How to get the data? How should the agent act?

Remember: data does not come for free. You have to take actions to get it. These actions could be suboptimal!

How to balance exploration vs. exploitation?
Epsilon-Greedy

Simplest possible way:

Parameter $0 < \epsilon \leq 1$

- Maintain $Q$ estimates.
- Occasionally (with probability $\epsilon$) pick action at random.
- Otherwise be greedy with respect to $Q$. 
Epsilon-Greedy

Exploration is *mandatory*. 

---

**Average reward**

$$\epsilon = 0.1$$

$$\epsilon = 0.01$$

**% Optimal action**

$$\epsilon = 0.1$$

$$\epsilon = 0.01$$

$$\epsilon = 0 \text{ (greedy)}$$
Downsides of Epsilon-Greedy

E-greedy exploration is uncontrolled:

- Fixed fraction of the time.
- Policy should be GLIE - greedy in limit of exploration.
- Treats all suboptimal actions equally.

Alternative: softmax.

$$\pi(A_i) = \frac{e^{Q(A_i)/\tau}}{\sum_j e^{Q(A_j)/\tau}}$$

$\tau$ parameter: temperature.

- High: lots of exploration.
- Tending to zero: greedy.

*High Q values more likely to be selected.*
Optimistic Initialization

Another alternative:

- Initialize the Q value optimistically.
- Do no exploration! Just MaxQ.

This works surprisingly well, but can do too much exploration.

Common approach: optimism in the face of uncertainty.
Optimistic Initialization

Why does this work?
Recent and very cool algorithm due to Auer, Cesa-Bianchi and Fischer (2002).

Let:

- $n$ be the number of actions taken so far
- $n_i$ be the number of times action $A_i$ taken

... then choose the action that maximizes:

$$Q(A_i) + \sqrt{\frac{2 \ln n}{n_i}}$$
UCB I

Why?!

Regret: (after $n$ steps)

\[
\text{regret}(n) = n \times Q(A^*) - \sum_{t=1}^{n} Q(A(t))
\]

what you did get

best you could’ve gotten

want to minimize regret
Way to minimize regret:

- Estimate a confidence interval for each $Q(A_i)$
- Pick the action the highest upper confidence bound.
- Either you are right (score) or you learn (useful)
Chernoff-Hoeffding Bound

So where do we get the bounds?

Hoeffding bound:

\[ P(\hat{Q}(A_i) - Q(A_i) \geq \epsilon) \leq e^{-2n_i \epsilon^2} \]

This bound is loose but general.

This gives us an expression for the confidence interval.

Plug it in!

Regret grows at rate \( \ln n \).
Coding Homework

Should be very quick! (not for credit)

5-arm bandit.
  • Assume normal distribution.
  • Assign each arm a mean and a variance.
  • Implement:
    • Epsilon greedy
    • Softmax
    • Optimistic
    • UCB1

... with the incremental averaging update. Plot *learning curve*.