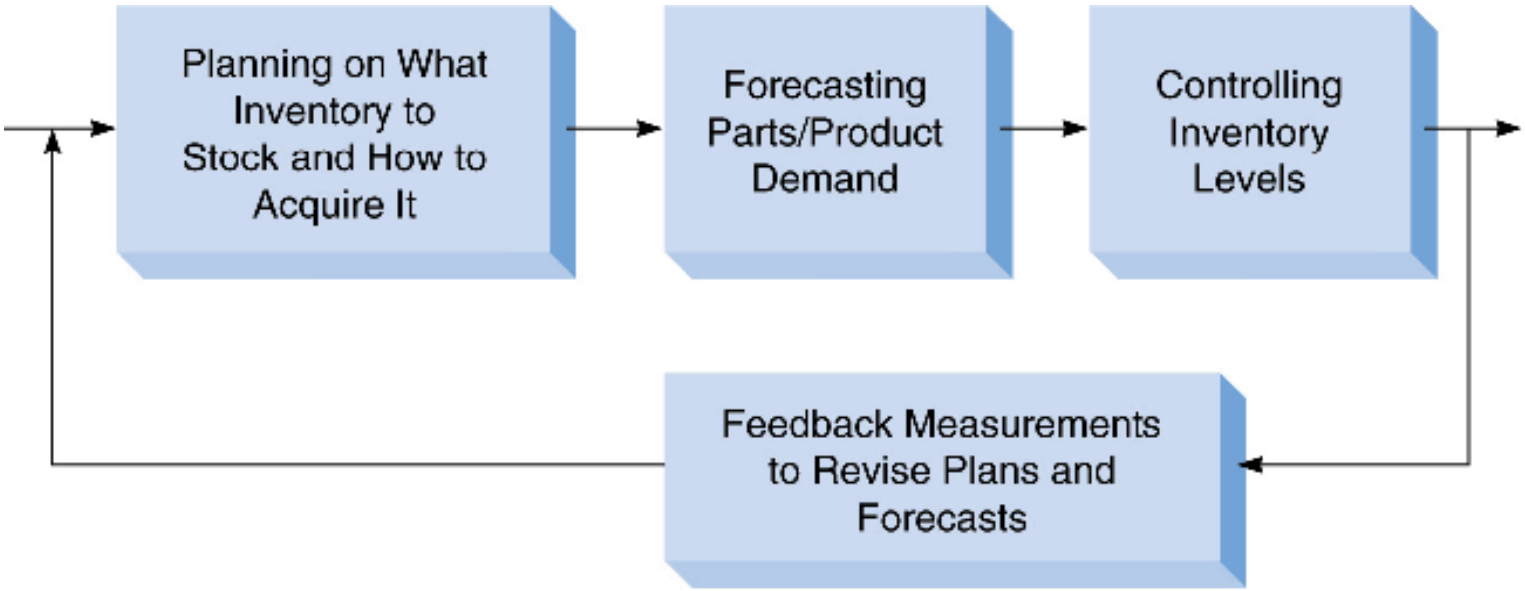
RL 2: Multi-Armed Bandits (MAB)

Michael Herrmann

University of Edinburgh, School of Informatics

15/01/2016

Example: Inventory Control



- Objective: Minimise total inventory **cost**
- Decisions:
  - How much to order?
  - When to order?

Components of Total Cost

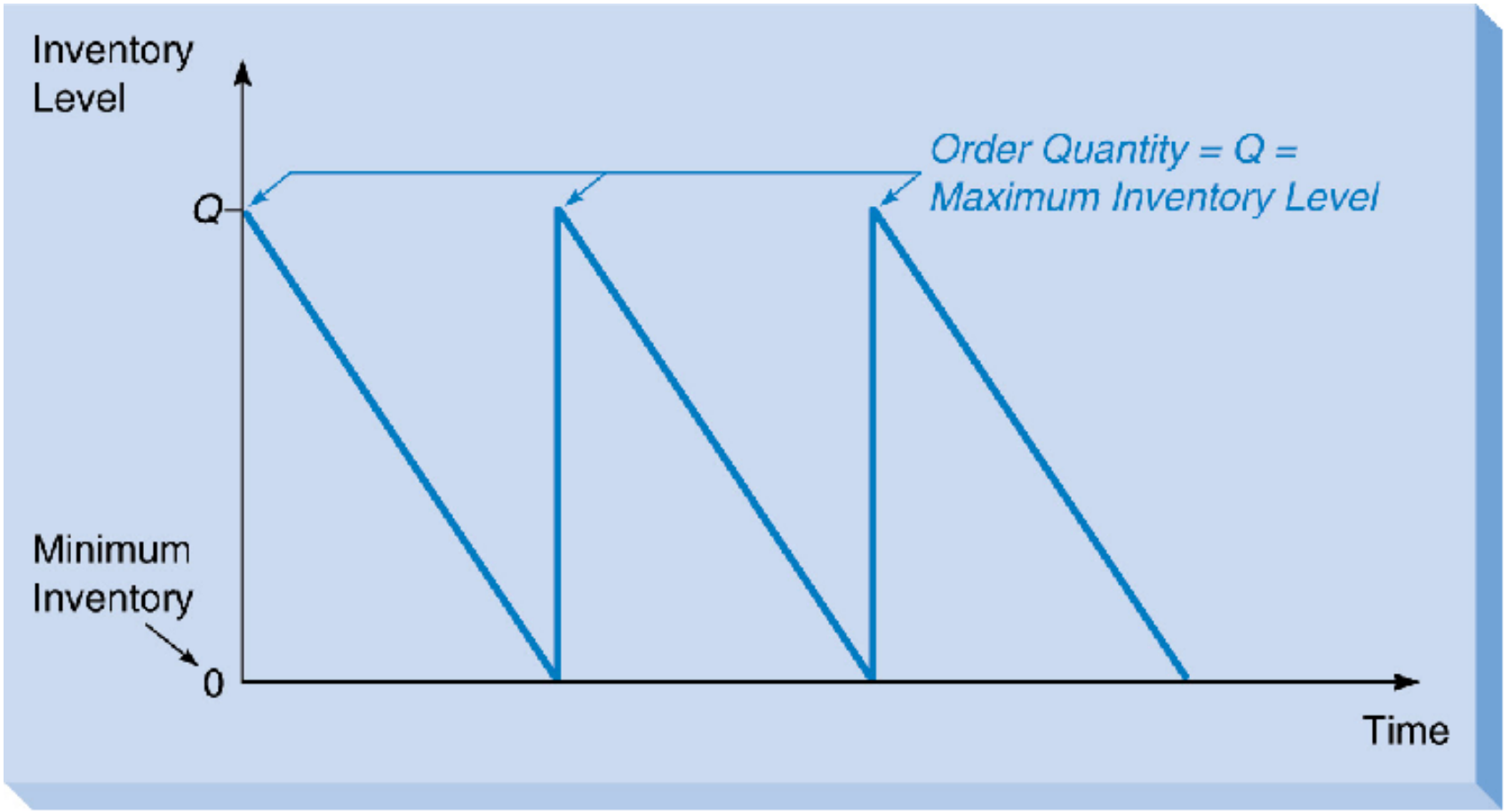
1. Cost of items
2. Cost of ordering
3. Cost of carrying or holding inventory
4. Cost of stockouts
5. Cost of safety stock (extra inventory held to help avoid stockouts)

The Economic Order Quantity Model - How Much to Order?

Assumptions

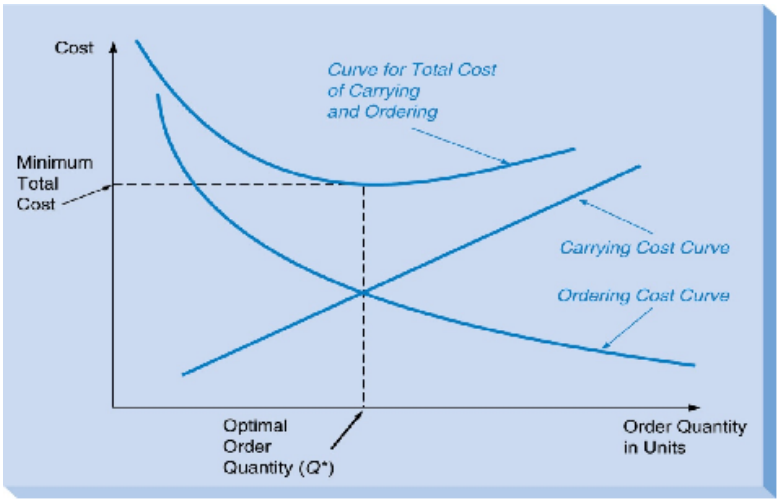
1. Demand is known and constant
2. Lead time is known and constant
3. Receipt of inventory is instantaneous
4. Quantity discounts are not available
5. Variable costs are limited to: ordering cost and carrying (or holding) cost
6. If orders are placed at the right time, stockouts can be avoided

Inventory Level Over Time Based on EOQ Assumptions



Economic order quantity, Ford W. Harris, 1913

EOQ Model Total Cost



$$C(Q) = \frac{\text{price}}{\text{item}} D + C_o \frac{D}{Q} + C_c \frac{Q}{2}$$

$$\frac{dC(Q)}{dQ} = -C_o \frac{D}{Q^2} + \frac{C_c}{2} \stackrel{!}{=} 0$$

$$Q^* = \sqrt{\frac{2DC_o}{C_c}}$$

D: demand, C_o: cost per order, C_c: cost per item, Q: quantity

Optimal order quantity (Q*): (Carrying cost)' $=-$ (Ordering cost)'

Realistically, how much to order
If these assumptions didn't hold?

- Demand is ~~known~~ and ~~constant~~
- Lead time (latency) is ~~known~~ and ~~constant~~
- Receipt of inventory is ~~instantaneous~~
- Quantity discounts are ~~not~~ available
- ~~Variable costs are limited to~~: ordering cost and carrying (or holding) cost
- If orders are placed at right time, stockouts ~~can be avoided~~

The result may require a more detailed stochastic optimisation.

Properties of RL learning tasks

- Does not assume that you know the model of the environment
- Associativity: Value of an action depends on state
- Active learning: Environment's response affects our subsequent actions and thus future responses
- Delayed reward: We find out the effects of our actions later
- Credit assignment problem: Upon receiving rewards, which actions were responsible for the rewards?

Relation to Metaheuristic Optimisation (MO)

- "Natural computing" including algorithms such as GA, ES, DE, PSO, or ACO
- For example, genetic algorithms (GA) maximise a fitness function by selecting, mutating and reproducing certain individuals in a population
- The individuals are described as strings over an alphabet, e.g. {G,A,T,C}
- The gradient of the fitness function is not known
- The population is a sample of points in the space of all strings

Difference between MO and RL

	RL	MO
representation	usually a single agent that must be restarted	population
learning steps	rewards is received after each action	fitness is evaluated in each generation
dynamics	Markovian state transitions	population moves in configuration space
global search	trial and error guided by earlier experiences	trial and error, may be guided by correlations
local search	policy gradient	hill climbing

Both are stochastic optimisation algorithms. Both can find their resp. global optimum under certain (quite artificial) conditions.

Note that many combinations are possible, e.g. collective RL or MO of the hardware of a robot that learns by RL

Chapter 2: Multi-Armed Bandits (MAB)

- A very simple model of reinforcement learning
- Focus only on action selection (one action at a time)
- Playground for study of the exploration-exploitation dilemma
- 1000s of publications and numerous applications

Multi-Armed Bandits (Herbert Robbins, 1952)

- N possible actions (one per machine = arm)
- Selecting action a_t leads to an immediate reward, $a_t \in \{1, \ldots, N\}$, $t = 1, \ldots, T$
- Reward depends only on present action and is characterised by (unknown) distribution which is fixed for each action, $r_t \sim P_{a_t}(r)$

- You can play for some period of time and you want to maximise average reward (utility)
- Simplifying assumption: No context. Later, we want to solve *contextual* problems.

Which is the best action/arm/machine?

What sequence of actions to take to find out?

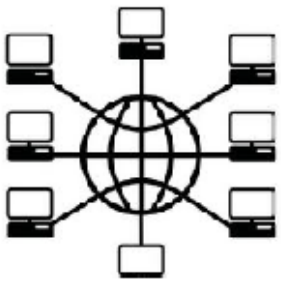
Numerous Applications!

Computer Go

Brain computer interface

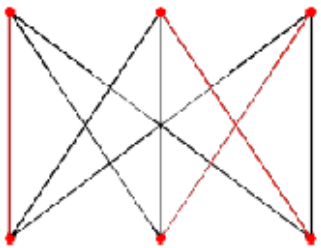
Medical trials

Packets routing



Ads placement

Dynamic allocation



Applications in Real Life: Decision Making

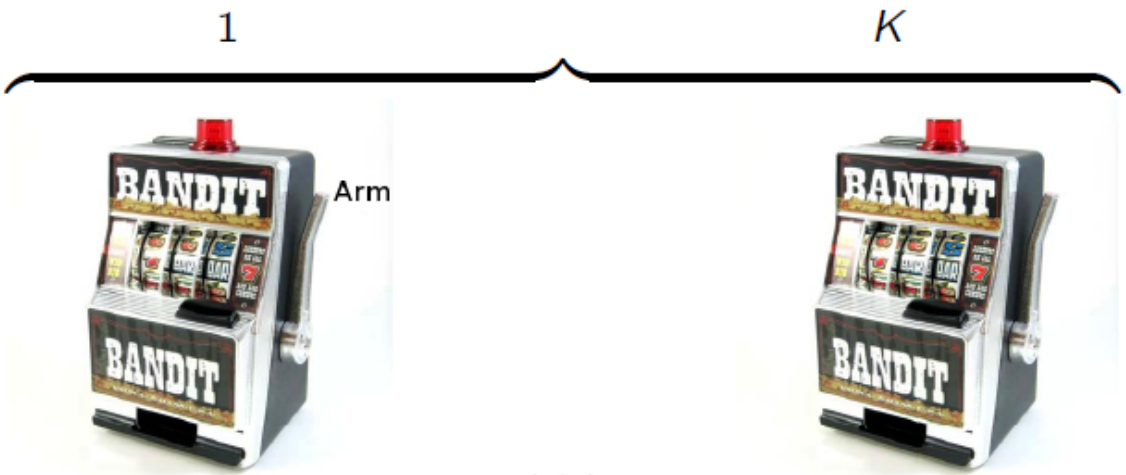
application	action	reward
"bandits"	arm	tokens
game	move	win
clinical trials	administered treatment	fraction of patients cured
advertisement	select ad	click on ad
routing	network link	transmission delay
...	...	...

Multi-armed bandit. Adapted from Wikipedia

Multi-Armed Bandits

Assume mean and variance of the arm's reward probabilities are

mean value:	0.5	0.2	0.8	0.4	0.2
std. dev.:	0.3	0.1	0.2	0.1	0.2



			...		
t=1	0.3	0.2	0.8	0.4	0.0
t=2	0.7	0.1	0.9	0.5	0.1
t=3	0.5	0.3	0.7	0.3	0.3
			...		

What action to choose next given rewards for actions 1, 3 and 2?

N-Armed Bandit Problem

- Choose repeatedly one of N actions; each is called a play
- After playing a_t, you get a reward r_t
- Try to estimate the *action value, i.e.* the *expected reward* conditioned on the chosen action

$$E\{r|a_t\} = Q(a_t)$$

- Objective is to maximise the (expected) reward, e.g.
  - in the long term, asymptotically,
  - over a given number of plays,
  - for non-stationary rewards
- To solve the N-armed bandit problem, you must explore a variety of actions (in order to estimate their values) and exploit the best of them (in order to maximise reward)

Exploration-Exploitation Dilemma

- Suppose, at time t you have arrived at reasonable estimates $\hat{Q}_t(a)$ of the true action values $Q(a)$, $a \in \mathcal{A}$, $|\mathcal{A}| = N$, i.e.

$$\hat{Q}_t(a) \approx Q(a)$$

- The greedy action at time t is a_t^*

$$a_t^* = \arg\max_{a \in \mathcal{A}} \hat{Q}_t(a)$$

- "Greedy" means exploitation only, i.e.

$$\begin{aligned} a_t &= a_t^* \Longrightarrow \text{exploitation} \\ a_t &\neq a_t^* \Longrightarrow \text{exploration} \end{aligned}$$

Dilemma:

- You can't exploit all the time; you can't explore all the time
- You should never stop exploring; but you may reduce exploring

The problem involves "a sequence of decisions, each of which is based on more information than its predecessors" (Gittins)

Action-Value Methods

Methods that adapt action-value estimates and nothing else, e.g.: suppose by the t-th play, action a had been chosen k_a times, producing rewards r_1, r_2 , . . . , r_{k_a}, then

$$\hat{Q}_t(a) = \frac{r_1 + r_2 + \cdots + r_{k_a}}{k_a}$$

The sample average is an estimator which approaches the reward expectation

$$\lim_{k_a \to \infty} \hat{Q}_{k_a}(a) = Q(a)$$

$$\sum_a k_a = t$$

Requires to choose each action infinitely often.

The greedy action selection strategy

Assume you have sampled all actions equally over t plays

Greedy action selection : $a_t^* = \arg\max_a \hat{Q}_t(a)$

Why might this be inefficient?

- Either t is large: you have spend a lot on exploration
- or t is small: estimation errors are large

Any compromises, that can be used for online estimation of the reward distribution for the good actions from a few samples?

ε-Greedy Action Selection

- Given the greedy action selection:

$$a_t^* = \arg\max_a \hat{Q}_t(a)$$

- we define ε-greedy:

$$a_t = \begin{cases} a_t^* & \text{with probability } 1-\varepsilon \\ \text{random action} & \text{with probability } \varepsilon \end{cases}$$

  . . . a simple way to balance exploration and exploitation

- Greedy[1] is ε-greedy for $\varepsilon = 0$

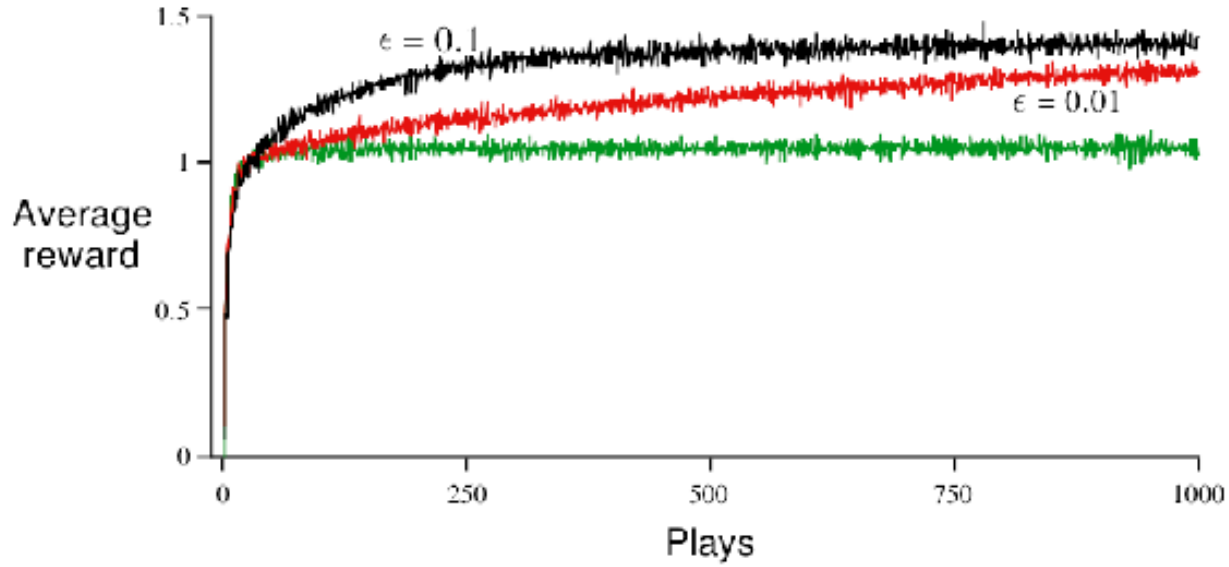
[1] Here is an initialisation problem involved. It may be advisable to try out each action a few times before continuing greedy or ε-greedy.

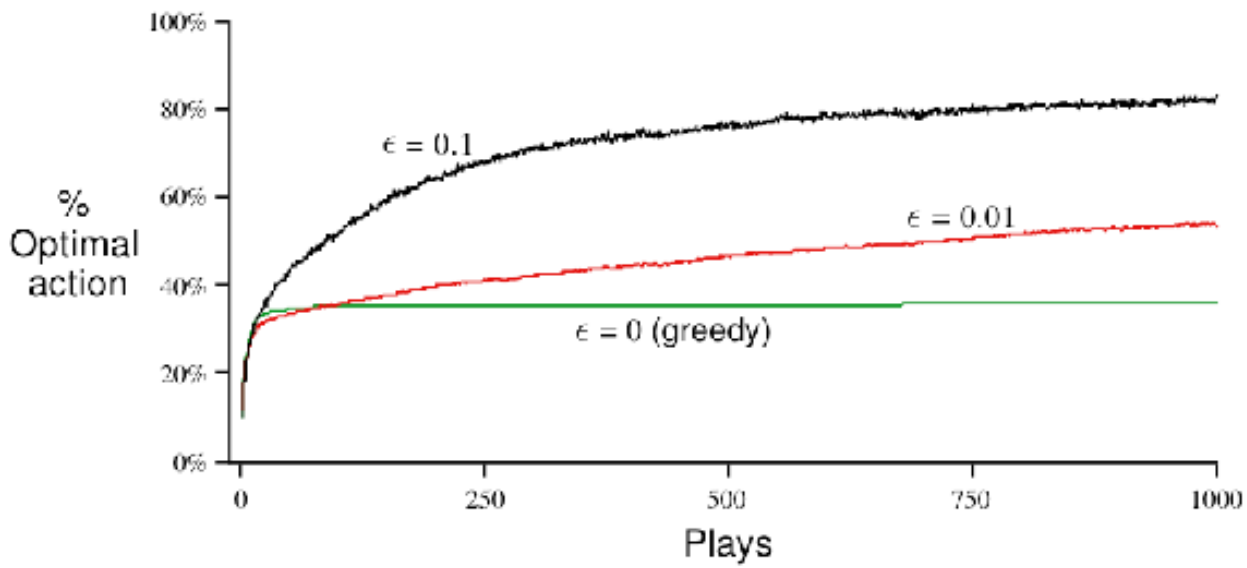
Worked Example: 10-Armed Testbed

- $N = 10$ possible actions
- $Q(a)$ are chosen randomly from a normal distribution $\mathcal{N}(0, 1)$
- Rewards r_t are also normal $\mathcal{N}(Q(a_t), 1)$
- 1000 plays with fixed $Q(a)$
- Average the results over 2000 trials, i.e. average over different random choices of $Q(a)$

see Sutton and Barto (1998)

ε-Greedy Methods on the 10-Armed Testbed





ε-Greedy Methods on the 10-Armed Testbed

- What value of ε is optimal?
- What would happen if we ran the experiment over $\geq$10,000 plays?
- Can we do better than ε-greedy (even for the best possible ε)?

Soft-max Action Selection

- Bias exploration towards promising actions
- Soft-max action selection methods grade action probabilities by estimated values
- The most common soft-max uses a Gibbs (or Boltzmann) distribution, i.e. chooses action a on play t with probability

$$\frac{e^{\hat{Q}_t(a)/\tau}}{\sum_{b=1}^{N} e^{\hat{Q}_t(b)/\tau}}$$

where τ is a "computational temperature":

  - $\tau \to \infty$: random, $\forall a \; P(a) = \frac{1}{N}$, pure exploration
  - $\tau \to 0$: greedy, pure exploitation

Incremental Implementation

- Sample average estimation method
- The average of the first k rewards is (separately for each action):

$$\hat{Q}_k = \frac{r_1 + r_2 + \cdots + r_k}{k}$$

- How to do this incrementally (without storing all the rewards)?
- We could keep a running sum and count, or, equivalently:

$$\hat{Q}_{k+1} = \hat{Q}_k + \frac{1}{k+1}\left(r_{k+1} - \hat{Q}_k\right)$$

- In words:

*NewEstimate = OldEstimate + StepSize * [NewData – OldEstimate]*

*= (1-StepSize) * OldEstimate + StepSize * NewData*

Tracking a Nonstationary Problem

- Choosing Q_k to be a sample average is appropriate in a stationary problem, i.e., when none of the $Q^*(a)$ change over time,
- But not in a nonstationary problem
- Better in the nonstationary case is to choose a constant $\alpha \in (0, 1]$

$$\begin{aligned} Q_{k+1} &= Q_k + \alpha (r_{k+1} - Q_k) \\ &= (1 - \alpha) Q_k + \alpha r_{k+1} \\ &= (1 - \alpha)^k Q_o + \sum_{i=1}^{k} \alpha (1 - \alpha)^{k-i} r_i \end{aligned}$$

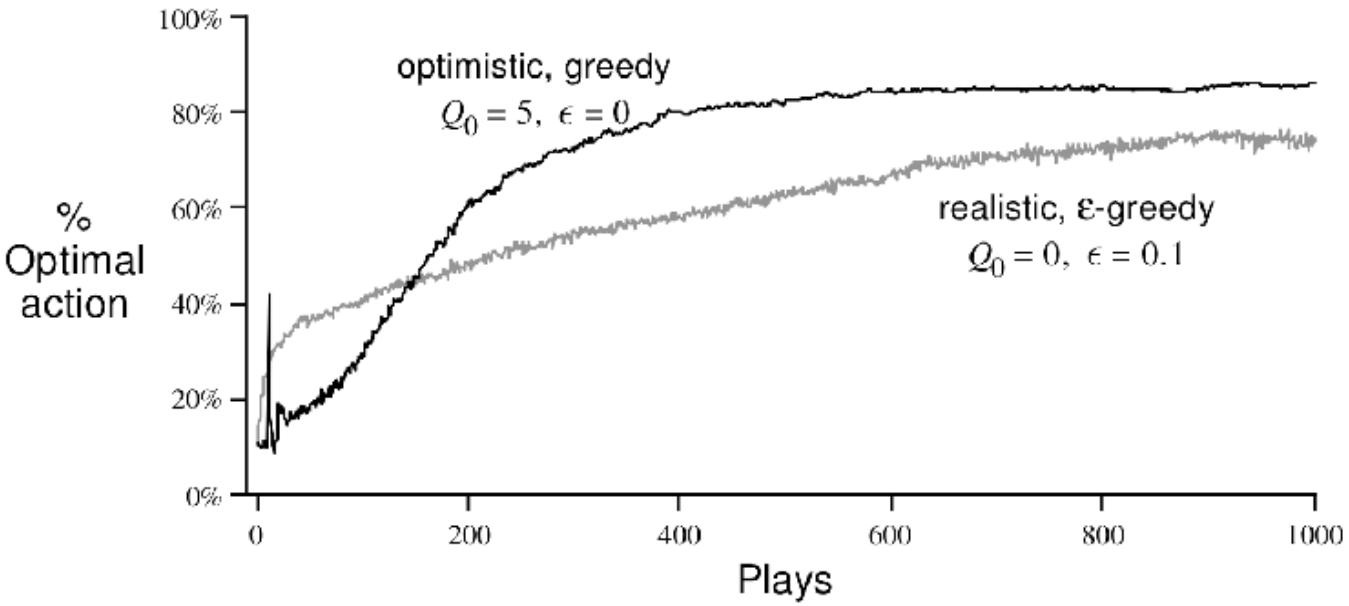
- This is an *exponential, recency-weighted average*

Optimistic Initial Values

All methods so far depend on $Q_0(a)$, i.e., they are biased

Encourage exploration: initialise the action values optimistically,

i.e., on the 10-armed testbed, use $Q_0(a) = 5 \; \forall a$



Does optimistic initialisation solve the exploration-exploitation dilemma?

How to evaluate exploitation of an algorithm: Regret

- Obtained reward depend on the reward distributions, so just consider the difference from the maximally expected reward: "regret"
- This does not work in practice, but helps to evaluate strategies.

[Reward sum of optimal strategy] – [Sum of actual collected rewards]

$$\begin{aligned} \rho &= T\mu^* - \sum_{t=1}^{T} \hat{r}_t = T\mu^* - \sum_{t=1}^{T} E\left[r_{i_t}(t)\right] \\ \mu^* &= \max_a \mu_a \end{aligned}$$

- If the average regret per round goes to zero with probability 1, asymptotically, we say the strategy has no-regret property
  $\sim$ guaranteed to converge to an optimal strategy
- ε-greedy is sub-optimal (so has some regret). Why?

How to evaluate exploration: Confidence bounds

- Estimate upper confidence bound $\hat{U}_t(k)$ for all action values
- Estimate should obey $Q(k) \leq \hat{Q}_t(k) + \hat{U}_t(k)$ with high prob.
- Choose action by comparing $\hat{Q}_t(k) + \hat{U}_t(k)$ rather than $\hat{Q}_t(k)$
- Try more often
  - rarely used action
  - actions with high-variance rewards tried more often
  - action with high estimates of average reward
- Select action maximising Upper Confidence Bound (UCB)

$$k_t = \arg\max_{k \in \mathcal{A}} \hat{Q}_t(k) + \hat{U}_t(k)$$

- In the course of time better estimates for rarely used actions become available, confidence bounds become narrower, estimates become better

Interval Estimation Procedure

- Associate to each arm k a (1-α) reward mean upper band
- Assume, e.g., rewards are normally distributed
- Arm k is observed N_k times to yield standard deviation (in addition to the empirical mean)
- α-upper bound:

$$\begin{aligned} \hat{U}_{\alpha,N_k}(k) &= \frac{\hat{\sigma}}{\sqrt{N_k}} c^{-1}(1-\alpha) \\ c(z) &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} \exp\left(-\frac{x^2}{2}\right) dx \end{aligned}$$

- c is a Cumulative Distribution Function, c^{-1} is the inverse function
- If α is carefully controlled, could be made zero-regret strategy
- In general, we don't know

Interval Estimation (simple variant)

- Attribute to each arm an "optimistic initial estimate" within a certain confidence interval
- Greedily choose arm with highest optimistic mean (upper bound on confidence interval)
- Infrequently observed arm will have over-valued reward mean, leading to exploration
- Frequent usage pushes optimistic estimate to true values

UCB Strategy (another variant)

- Again, based on notion of an upper confidence bound but more generally applicable
- Algorithm:
  - Play each arm k once ($k = 1, \ldots, K$)
  - At time $t > K$, play the arm for which

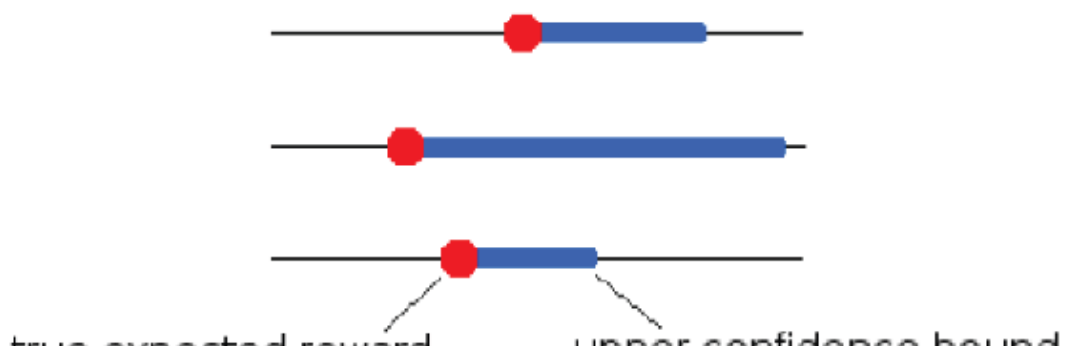
$$\bar{r}_j(t) + \sqrt{\frac{2 \ln t}{T_{j,t}}}$$

is maximal, where $T_{j,t}$ is the number of times the arm j has been played before time t.

UCB Strategy (based on Chernoff-Hoeffding Bound)

Intuition:

The second term $\sqrt{2 \ln t / T_{j,t}}$ is the the size of the one-sided $(1 - 1/t)$-confidence interval for the average reward (using Chernoff-Hoeffding bounds).



Let $X_1, X_2, \ldots, X_n$ be independent random variables in the range $[0, 1]$ with $\mathbb{E}[X_i] = \mu$. Then for $a > 0$,

$$P\left(\frac{1}{n}\sum_{i=1}^{n} X_i \geq \mu + a\right) \leq e^{-2a^2 n}$$

UCB Strategy – Behaviour

We will not try to prove the following result but I quote the final result to tell you why UCB may be a desirable strategy – regret is bounded.

Theorem

(Auer, Cesa-Bianchi, Fisher) At time T, the regret of the UCB policy is at most

$$\frac{8K}{\Delta^*} \ln T + 5K,$$

where $\Delta^* = \mu^* - \max_{i:\mu_i<\mu^*} \mu_i$ (the gap between the best expected reward and the expected reward of the runner up).

Variation on SoftMax

It is possible to drive regret down by annealing τ

Exp3: Exponential weight algorithm for exploration and exploitation

Probability of choosing arm k at time t is

$$P_k(t) = (1-\gamma)\frac{w_k(t)}{\sum_{j=1}^{k} w_j(t)} + \frac{\gamma}{K}$$

$$w_j(t+1) = \begin{cases} w_j(t)\exp\left(-\gamma\frac{r_j(t)}{P_j(t)K}\right) & \text{if arm } j \text{ is pulled at } t \\ w_j(t) & \text{otherwise} \end{cases}$$

$$\text{regret} = O\left(\sqrt{T\,K\,\log(K)}\right)$$

Asymptotically regret-free, simpler but tends to be worse than UCB

Auer et al. 1998

Towards an optimal solution: The Gittins Index

- Each arm delivers reward with a probability
- This probability may change through time but only when arm is pulled
- Goal is to maximise discounted rewards – future is discounted by an exponential discount factor $\gamma < 1$.
- The structure of the problem is such that, all you need to do is compute an "index" for each arm and play the one with the highest index
- Index is of the form ($k \in \mathcal{A}$):

$$\nu_k = \sup_{T>0} \frac{\left\langle \sum_{t=0}^{T} \gamma^t R^k(t) \right\rangle}{\left\langle \sum_{t=0}^{T} \gamma^t \right\rangle}$$

Gittins Index – Intuition

- Proving optimality is not within our scope
- Stopping time: the point where you should 'terminate' bandit
- Nice Property: Gittins index for any given bandit is independent of expected outcome of all other bandits
  - Once you have a good arm, keep playing until there is a better one
  - If you add/remove machines, computation does not really change
- BUT:
  - hard to compute, even when you know distributions
  - Exploration issues: Arms are not updated unless used[2].

[2] In the *restless bandit problem* bandits can change even when not played.

Extending the MAB Model

- In this lecture, we are in a single casino and the only decision is to pull from a set of N arms
  (except in the very last slides, not more than a single state!)

Next,[3]

- What if there is more than one state?
- So, in this state space, what is the effect of the distribution of payout changing based on how you pull arms?
- What happens if you only obtain a net reward corresponding to a long sequence of arm pulls (at the end)?

[3] Many slides are adapted from web resources associated with Sutton and Barto's Reinforcement Learning book. The original design of the slides has been prepared by Dr. Subramanian Ramamoorthy for his RL course.

Literature

- http://en.wikipedia.org/wiki/Multi-armed_bandit[4]
- H. Robbins (1952) Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society* **58**(5): 527–535.
- Cowan, Robin (July 1991) Tortoises and Hares: Choice among technologies of unknown merit 101 (407). pp. 801–814.
- P. Auer, N. Cesa-Bianchi, and P. Fischer (2002) Finite-time analysis of the multiarmed bandit problem. *Machine learning* **47**(2-3): 235-256.
- B. Si, K. Pawelzik, and J. M. Herrmann (2004) Robot exploration by subjectively maximizing objective information gain. *IEEE International Conference on Robotics and Biomimetics, ROBIO 2004*.

[4] Colour code: red – required reading; blue – recommended reading; black – good to know.