MultiAgent Systems
Exploration versus Exploitation

Eric Pauwels (CWI & VU)

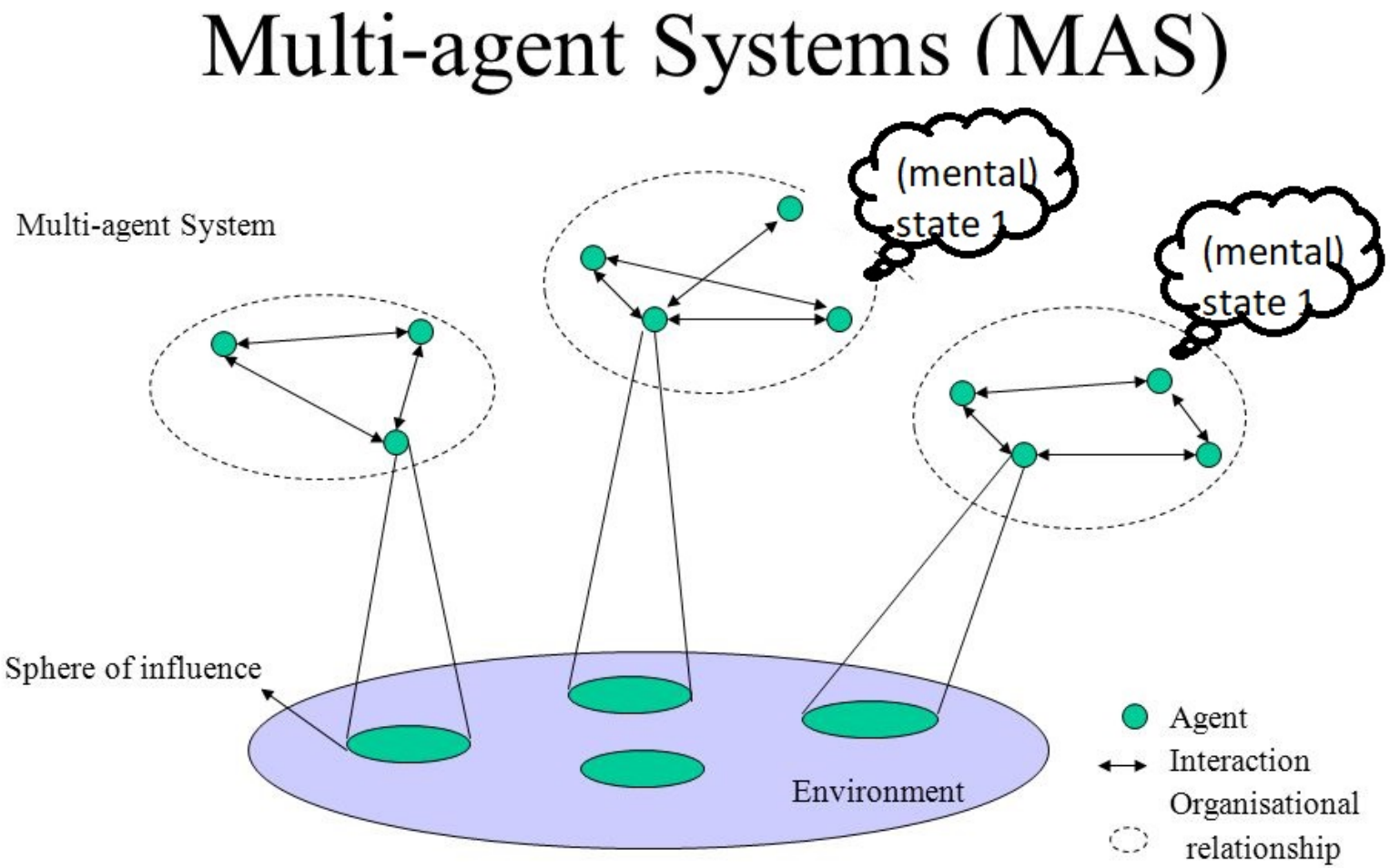
Version: November 22, 2021

Reading

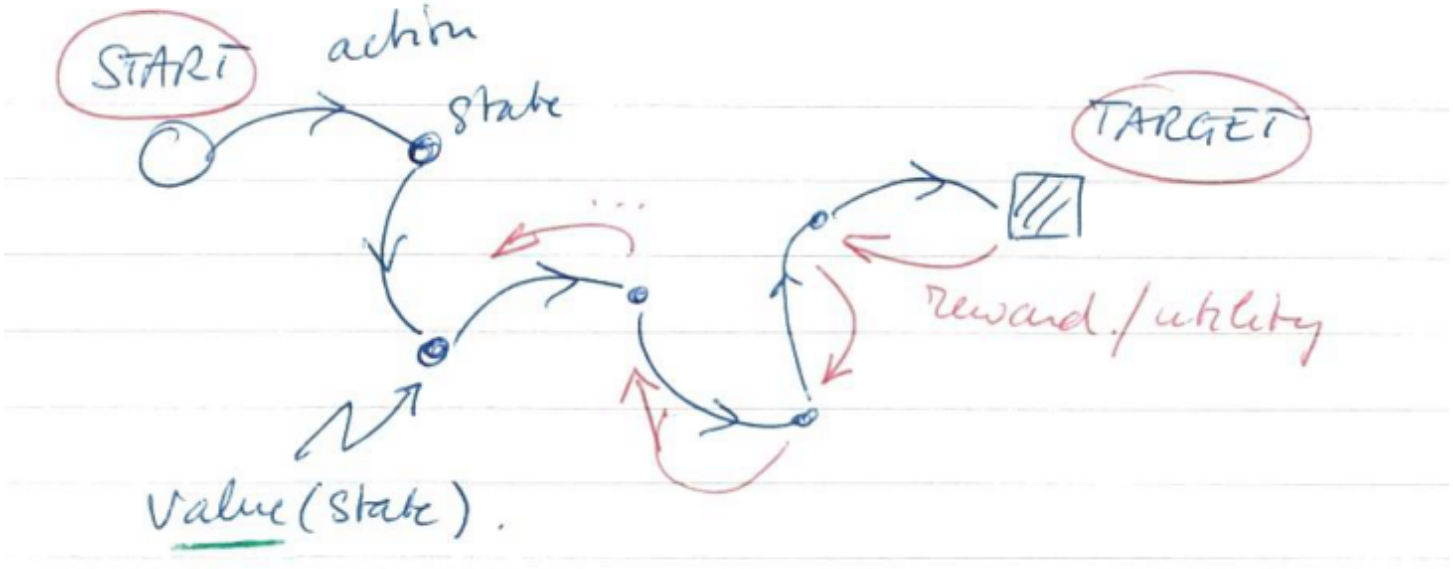
- Sutton & Barto (2nd ed.): chapters 1 & 2

Table of Contents

MultiAgent Systems: Overview



Recurring themes in (sequential) decision making



- States, actions, transitions, policy, value functions;
- Back-up, optimisation (planning and searching)

Sequential Decision Making

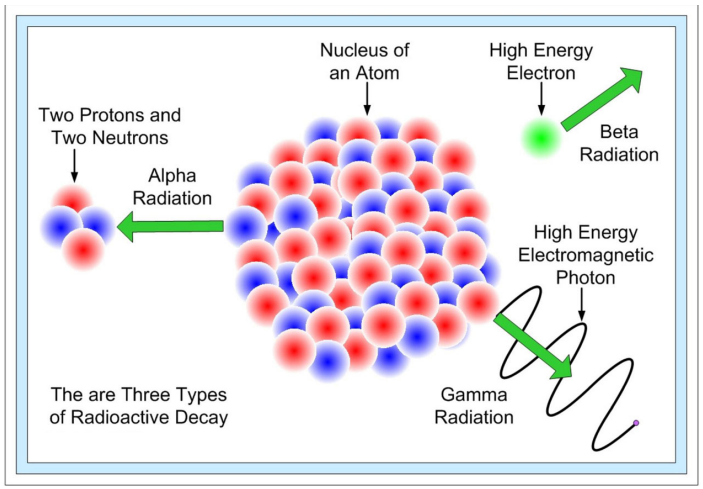
- In **sequential decision making** an agent tries to solve a sequential control problem by directly interacting with an unknown environment
- **Learning by trial and error** Agent tries out actions to learn about their consequences
- **Not supervised:** No examples of correct or incorrect behavior; instead only **rewards** for actions tried
- **Active learning:** agent has partial control over what data it will obtain for learning
- **On-line learning:** it must maximize performance during learning, not afterwards

Table of Contents

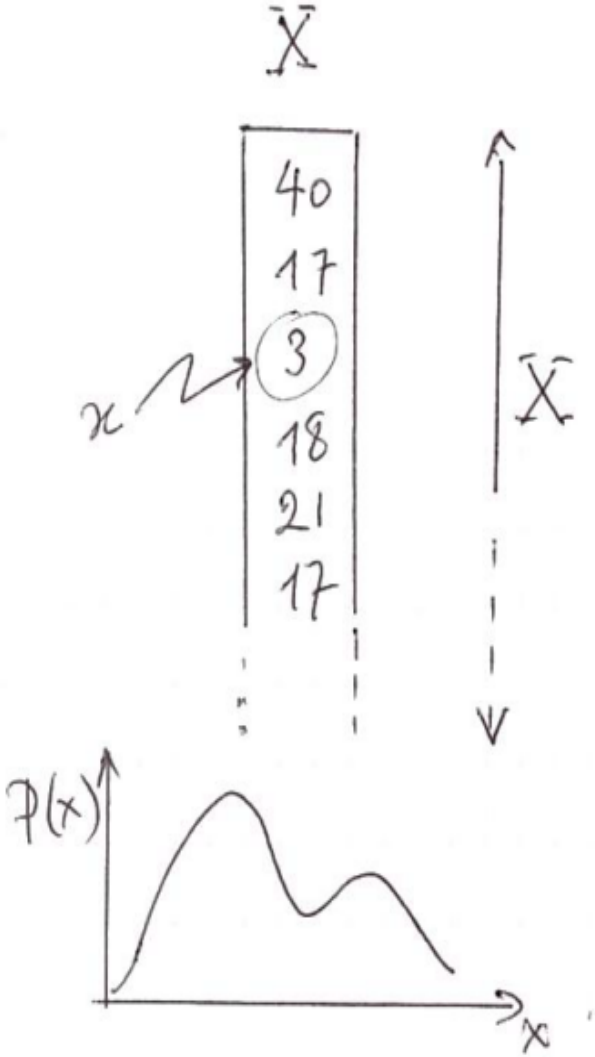
Stochastic Variables

- **Stochastic (or random) variables:** abstract model the idea of a randomly determined numerical outcome;
- Formally:

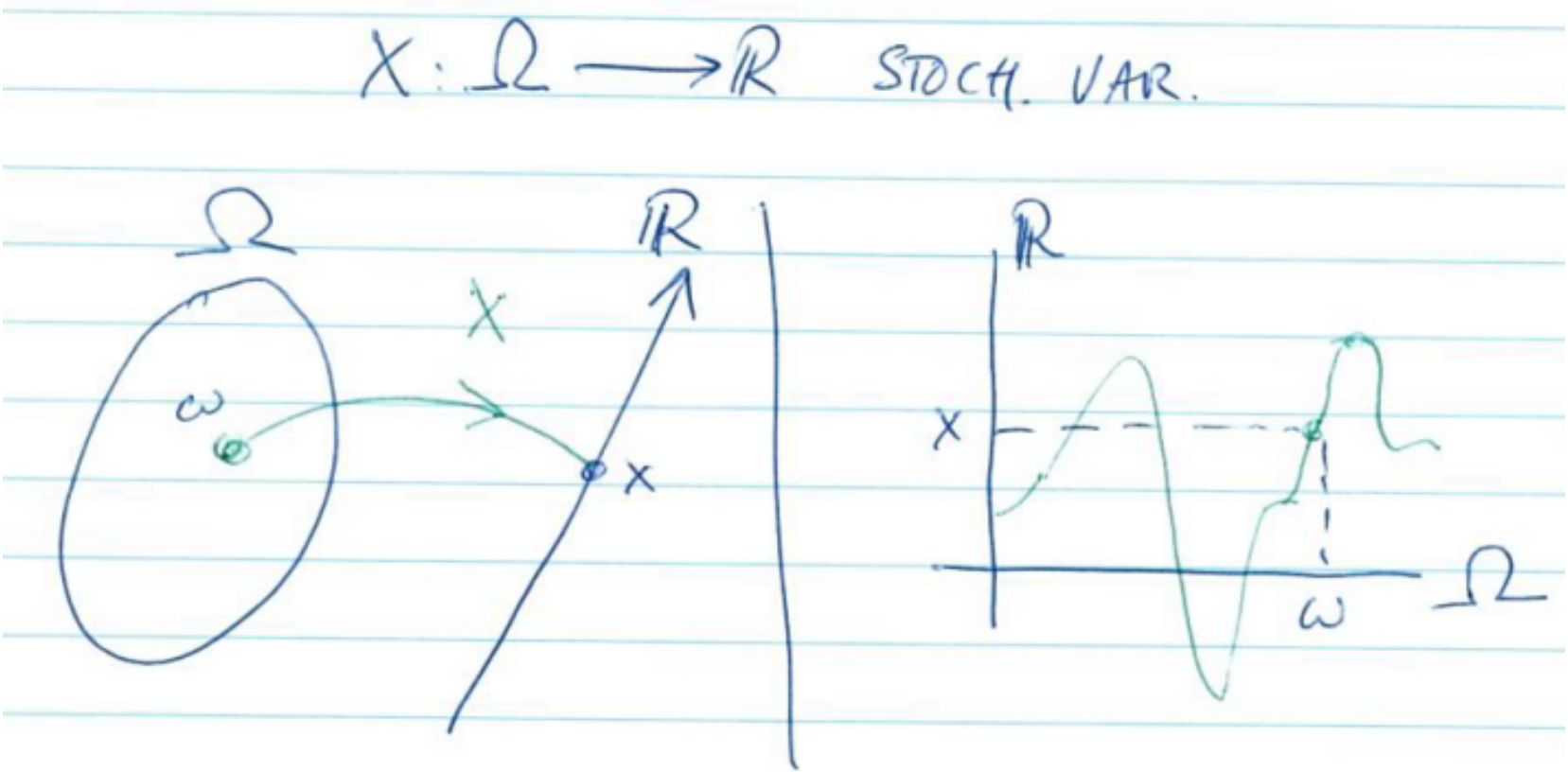
$$X : \Omega(\text{"outcomes"}) \longrightarrow \mathbb{R}$$



Stochastic variables



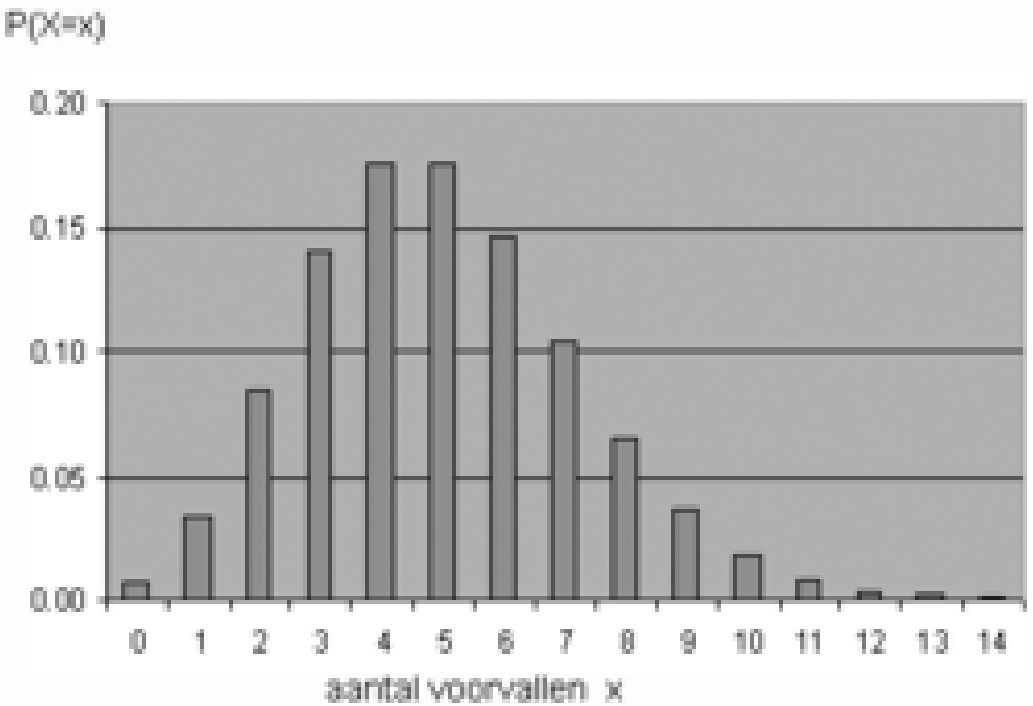
Stochastic Variable



Probability Distribution (1)

Discrete set of outcome values

- $X : \Omega \longrightarrow \{x_1, x_2, x_3 \ldots\}$
- $P(X = x_k) = p_k \qquad (p_k \geq 0)$
- $\forall k : \; p_k \geq 0 \qquad \text{and} \qquad \sum_k p_k = 1$



Probability Distribution (2)

Continuous outcome values $X : \Omega \longrightarrow \mathbb{R}$

- Characterized by density function $f(x)$ such that

$$P(a \leq X \leq b) = \int_a^b f(x)\, dx$$

- Informally: $P(x \leq X < x + dx) = f(x)\, dx$

- $\forall x : f(x) \geq 0$ and $\int_{-\infty}^{+\infty} f(x)\, dx = 1$

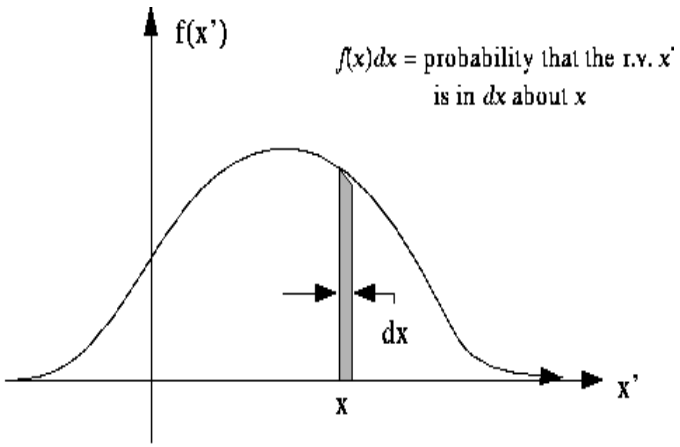
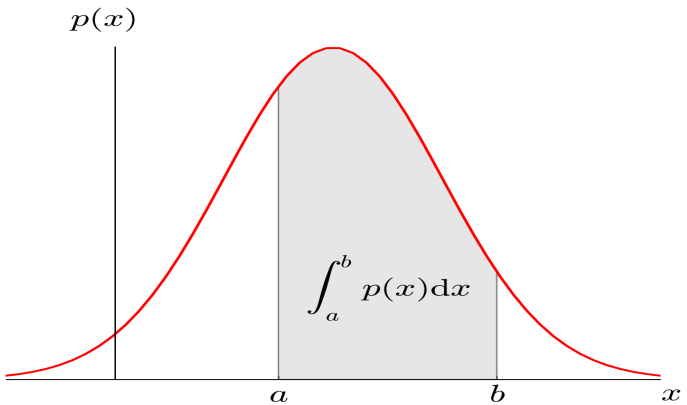


Figure 4. Typical Probability Distribution Function (*pdf*)



Expectation Operator and Expected Value

$$E(X) = \sum_k x_k P(X = x_k) = \sum_k x_k p_k \qquad (=: \mu)$$

$$E(X) = \int x\, f(x)\, dx \qquad (=: \mu)$$

$$\begin{aligned} Var(X) &= E\left((X - EX)^2\right) = E(X - \mu)^2 \\ &= \sum_k (x_k - \mu)^2 p_k \qquad \text{(discrete prob)} \\ &= \int (x - \mu)^2\, f(x)\, dx \qquad \text{(continuous prob)} \end{aligned}$$

Expectation and Variance of Linear Combintation

$$E(aX + bY) = aE(X) + bE(Y)$$

$$Var(aX + bY) = a^2 Var(X) + b^2 Var(Y) + 2ab\, Cov(X, Y)$$

In particular:

$$E(aX) = a\, E(X) \qquad Var(aX) = a^2 Var(X)$$

If X, Y independent: $Var(X \pm Y) = Var(X) + Var(Y)$

Independence

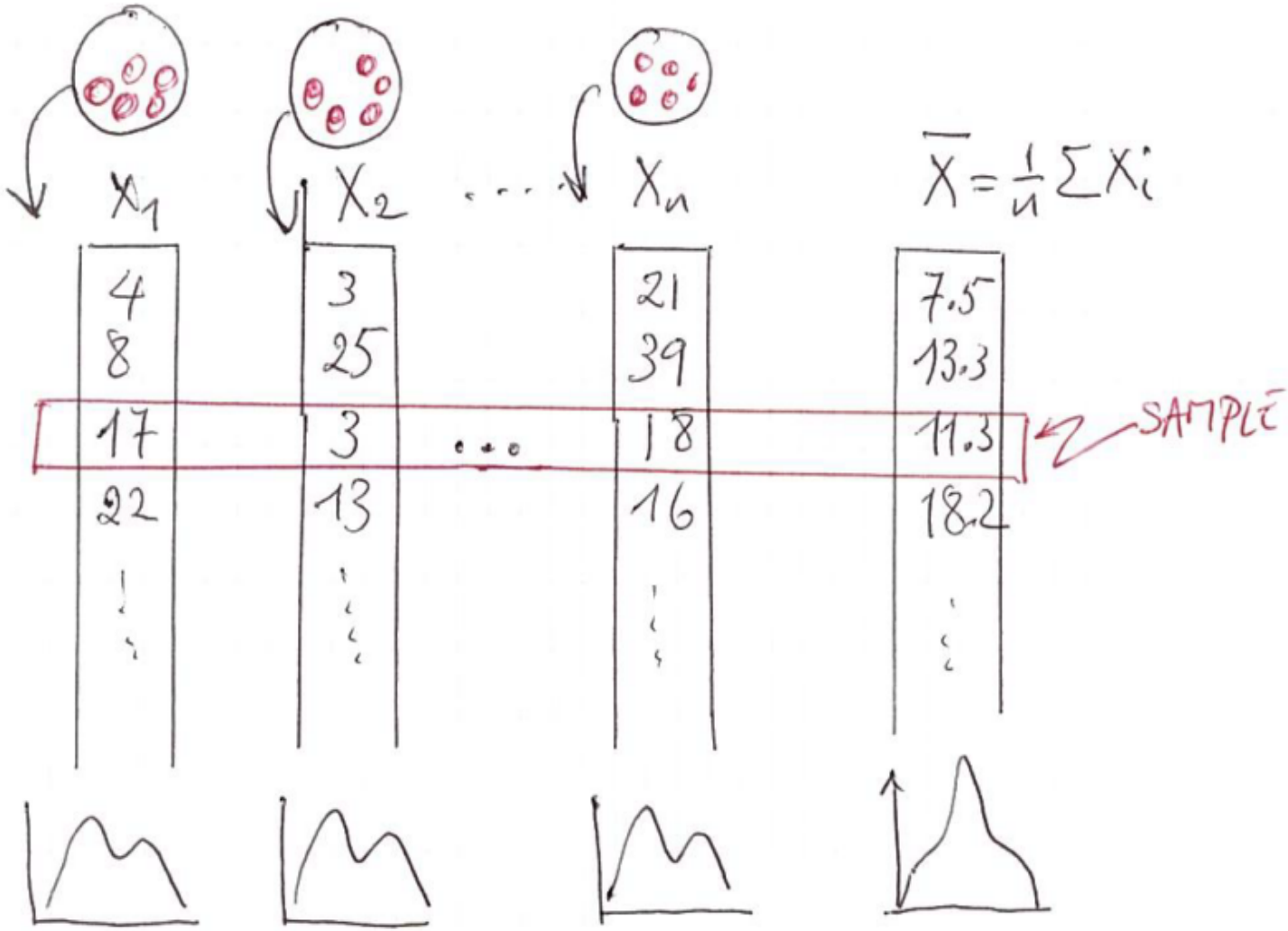
- Independent events:

$$P(A \ \& \ B) = P(A)P(B)$$

- Independent (discrete) stochastic variables:

$$P(X = a \ \& \ Y = b) = P(X = a)P(Y = b)$$

Sample of independent, identically distributed (i.i.d.) rvs



Expectation and Variance of Sample Mean

Let $X_1, X_2, \ldots, X_n$ be sample of size n from distribution (population) with mean μ and variance σ^2:

- Sample mean:

$$\overline{X}_n = \frac{1}{n} \sum_{i=1}^{n} X_i$$

- Mean and variance of sample mean:

$$E(\overline{X}_n) = \mu \qquad Var(\overline{X}_n) = \frac{\sigma^2}{n}$$

- (Asymptotic) distribution of sample mean (Central Limit Theorem, CLT):

$$\overline{X}_n \longrightarrow N(\mu, \sigma^2/n)$$

Conditional Probability and Independence

- **Conditional Probability**

$$P(A \mid B) := \frac{P(A \cap B)}{P(B)}$$

- **Independence**

$$P(A \cap B) = P(A)P(B \mid A) = P(B)P(A \mid B)$$

If A, B are independent: $\quad P(A \mid B) = P(A)$

- **Conditional independence:**

$$P(A \cap B \mid C) = P(A \mid C)P(B \mid C)$$

Bayes' Rule

Bayes' rule:

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}$$

Law of total probability:

Let $B_1, B_2, \ldots, B_n$ partition of outcome space:

$$P(A) = \sum_{i=1}^{n} P(A \cap B_i) = \sum_{i=1}^{n} P(A \mid B_i) P(B_i)$$

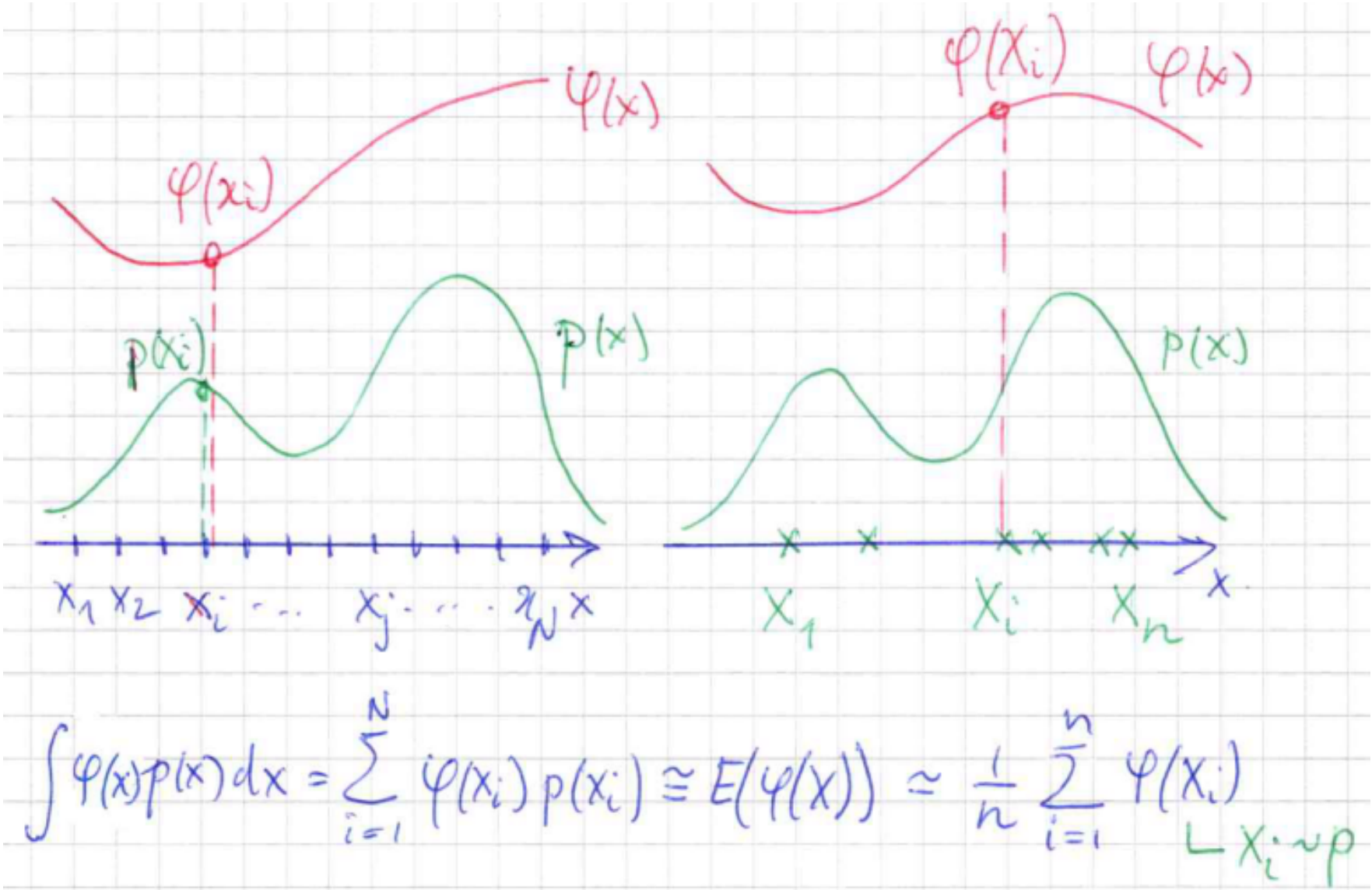
Monte Carlo: Sample-based computation

- Let $X_1, X_2, \ldots, X_n$ be a sample (i.i.d) from given (discrete/continuous) distribution;

$$EX = \left\{ \begin{array}{c} \sum_{k=0}^{\infty} x_k P(X = x_k) = \sum_{k=0}^{\infty} x_k p_k \\ \int x f(x) dx \end{array} \right\} \approx \frac{1}{n} \sum_{i=1}^{n} X_i$$

$$E\varphi(X) = \left\{ \begin{array}{c} \sum_{k=0}^{\infty} \varphi(x_k) p_k \\ \int \varphi(x) f(x) dx \end{array} \right\} \approx \frac{1}{n} \sum_{i=1}^{n} \varphi(X_i)$$

Monte Carlo approximation



$$\int \varphi(x)p(x)\,dx = \sum_{i=1}^{N} \varphi(x_i)\,p(x_i) \cong E(\varphi(x)) \simeq \frac{1}{n}\sum_{i=1}^{n} \varphi(X_i) \quad \llcorner X_i \sim p$$

Kullback-Leibler Divergence

- Let p, q be probability distributions (cont. or discrete)

KL 散度，这是一个用来衡量两个概率分布的相似性的一个度量指标

Kullback-Leibler divergence

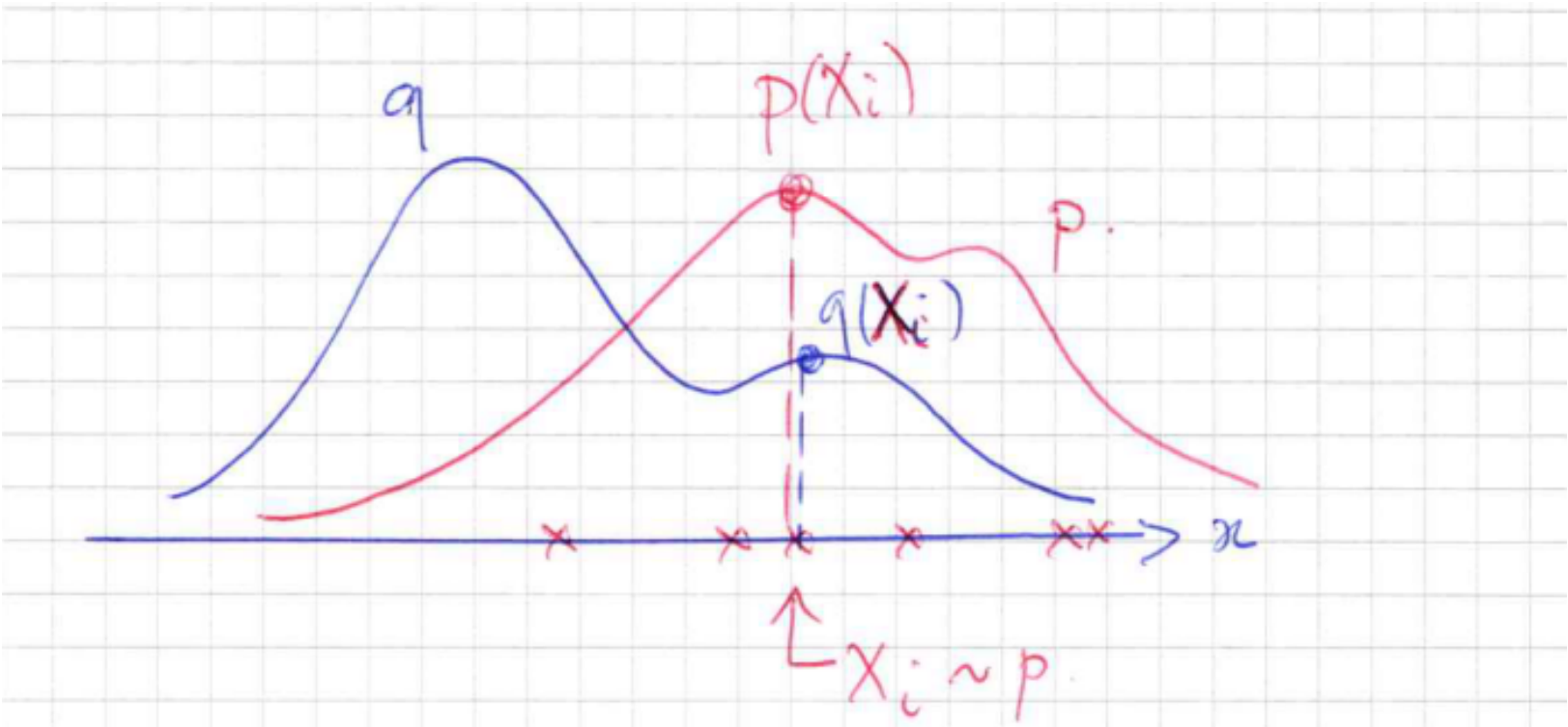
$$KL(p||q) := \int p(x) \log \frac{p(x)}{q(x)} \, dx \geq 0$$

$$KL(p||q) := \sum_k p_k \log \frac{p_k}{q_k} \geq 0$$

- Divergence is **not symmetric:** $KL(p||q) \neq KL(q||p)$
- Sample $X_1, X_2, \ldots, X_n \sim p$:

$$KL(p||q) \approx \frac{1}{n} \sum_{i=1}^{n} \log \frac{p(X_i)}{q(X_i)}$$

Kullback-Leibler divergence



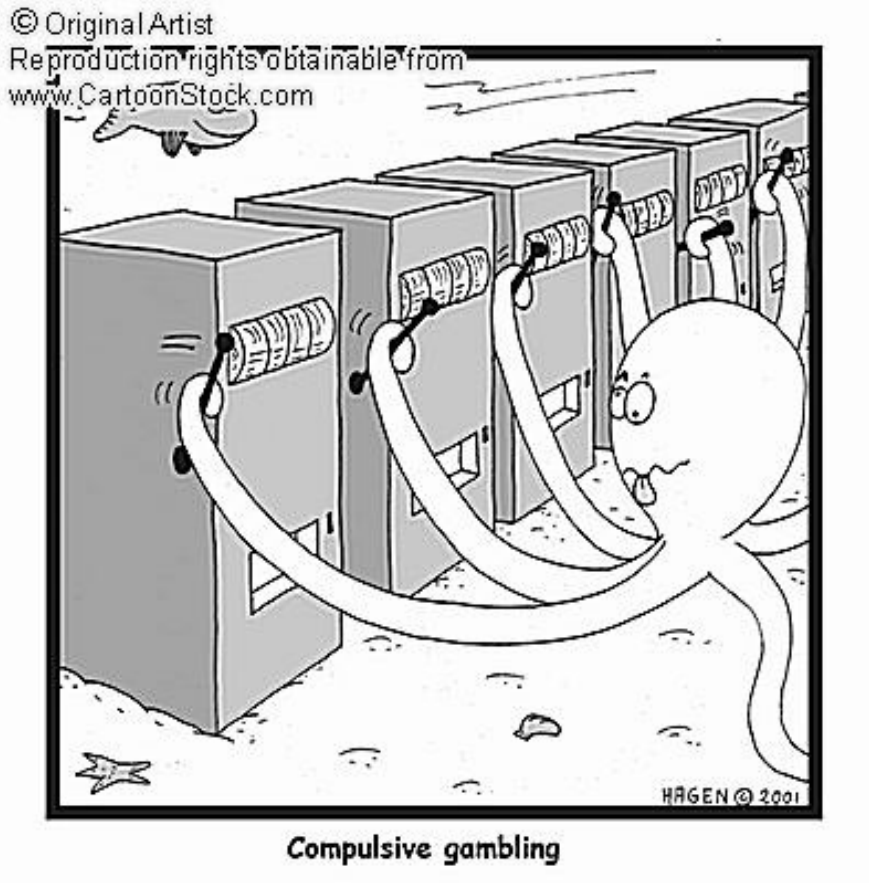
$$KL(p;q) = \int p(x) \log \frac{p(x)}{q(x)} dx \cong \frac{1}{n} \sum_{i=1}^{n} \log \frac{p(X_i)}{q(X_i)}$$

$X_i \sim p$

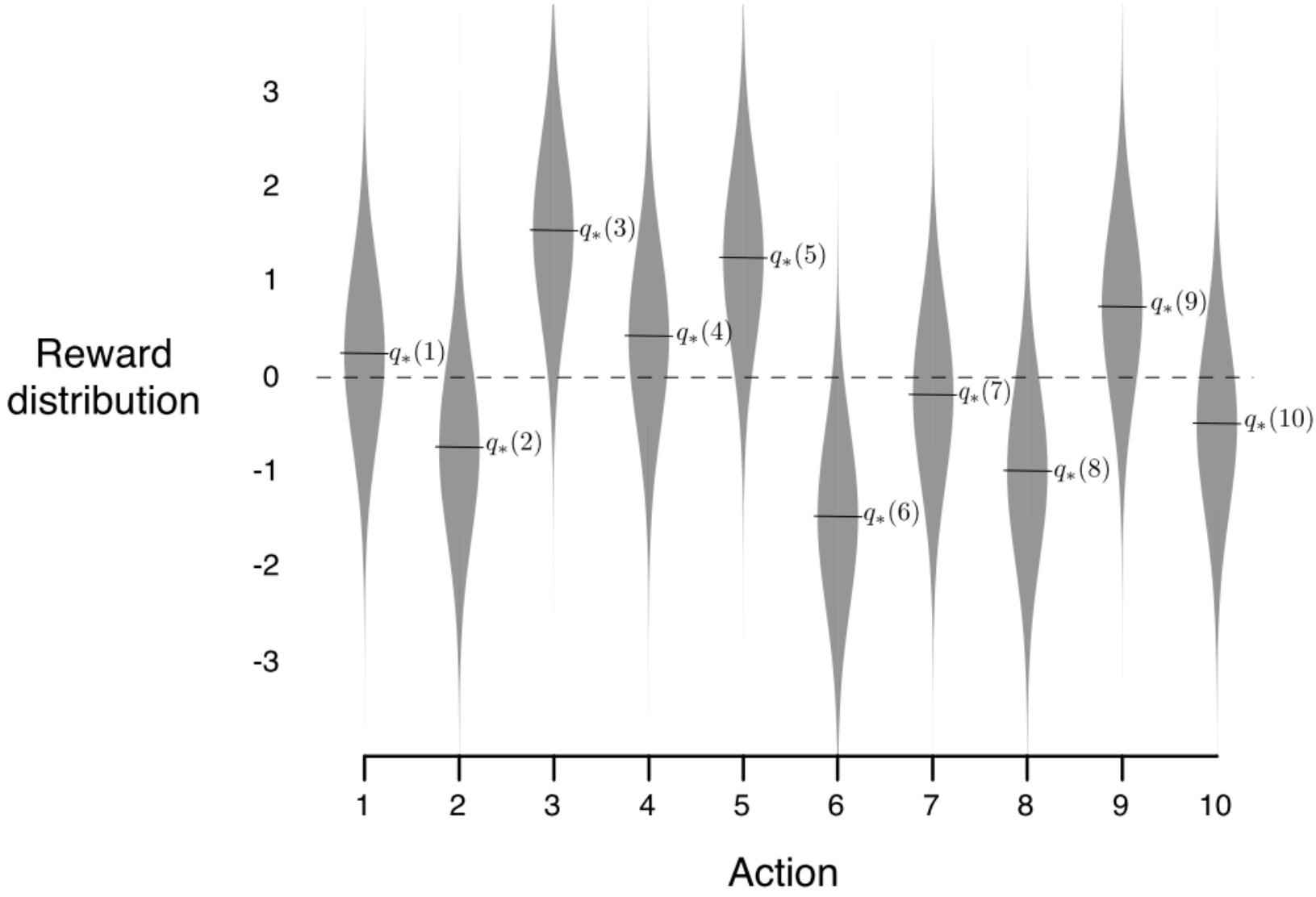
Table of Contents

Prototypical problem: K-armed bandit problem

- Sit before a slot machine (bandit) with many arms
- Each arm has an unknown stochastic payoff
- Goal is to maximize cumulative payoff over some period



K-armed bandit: example



Formalizing the k-armed bandit problem

- There are k **actions** available at each timestep;
- After action t, the agent receives (stochastic) reward $R_t \sim f_{a_t}$
- In a **finite-horizon** problem, the agent tries to maximize its **total reward** over T actions: $\sum_{t=1}^{T} R_t$
- In an **infinite-horizon** problem, the agent tries to maximize its **discounted total reward**: $\sum_{t=0}^{\infty} \gamma^t R_t$ where $0 < \gamma < 1$
- The **discount factor** γ can be interpreted as the probability of the game continuing after each step

Exploration and exploitation

The agent's ability to get reward in the future depends on what it knows about the arms.

- It must **explore** the arms in order to learn about them and improve its chances of getting future reward;
- It must use what it already knows in order to maximize its total reward; Thus it must **exploit** by pulling the arms it expects to give the largest rewards;

Balancing exploration and exploitation

- The main challenge in a k-armed bandit is how to **balance the competing needs of exploration and exploitation**
- If the horizon is finite, exploration should decrease as the horizon gets closer
- If the horizon is infinite but $\gamma < 1$, exploration should decrease as the agent's uncertainty about expected rewards goes down

Action-value methods

Formal Setup:

- Each arm (i.e. action $a = 1, \ldots, k$) generates stochastic reward R sampled from probability distribution f_a with unknown mean $q(a)$, hence: $q(a) := E_{f_a}(R)$
- **Action-value**: $q(a)$ can be thought of as the (average) value (quantity) generated by taking action a;
  - Compare to *state-action value* $q(s, a)$ in RL
- We use the sample average $Q_t(a)$ is an estimate of $q(a)$. Specifically, if action a has been chosen k_a times, yielding rewards $R_1, R_2, \ldots, R_{k_a}$, then:

$$Q_t(a) = \frac{1}{k_a} \sum_{i=1}^{k_a} R_i$$

Action-value methods

- Asymptotically: 渐进线

$$Q_t(a) \longrightarrow q(a) \qquad \text{as} \qquad t, k_a \longrightarrow \infty.$$

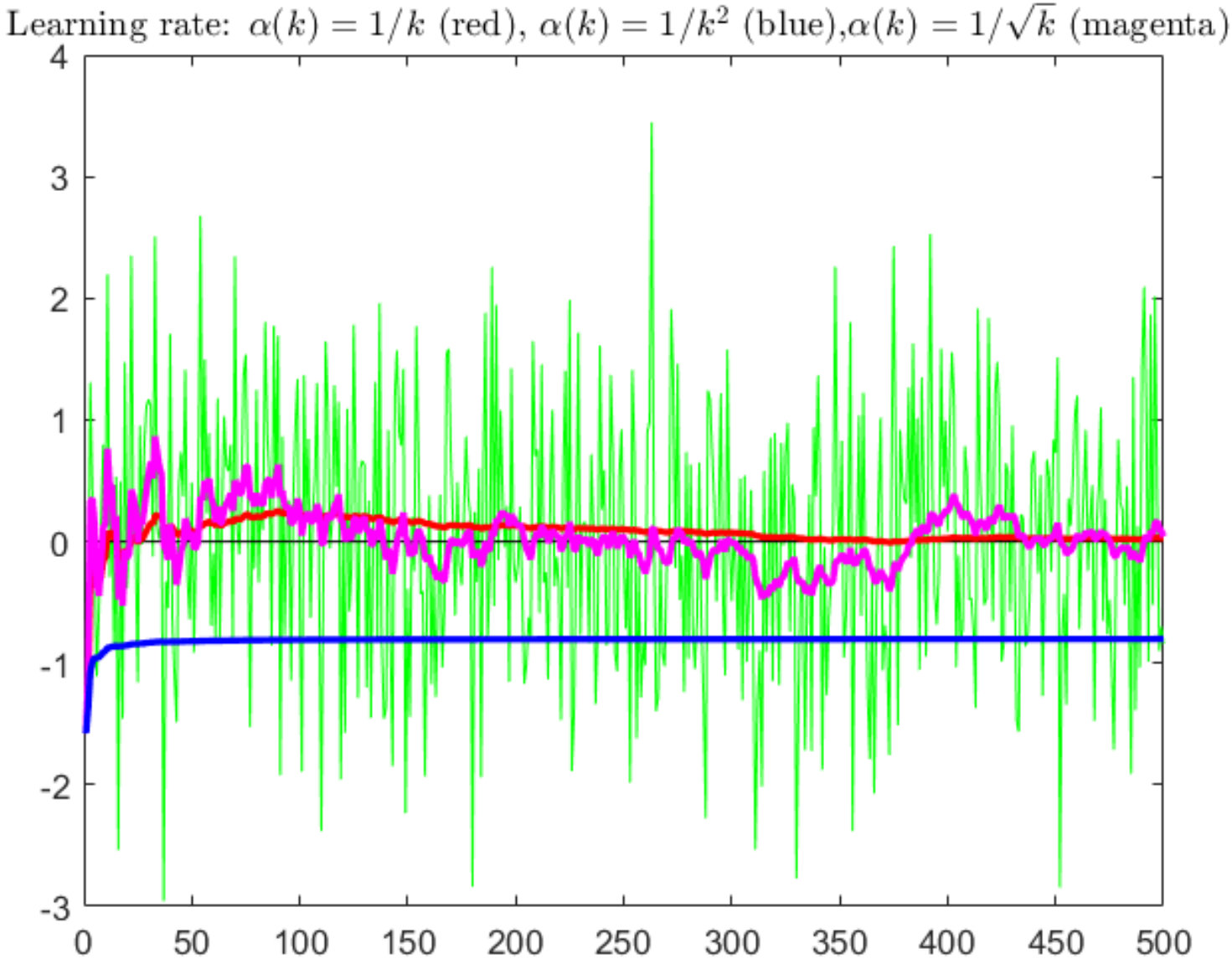
- Incremental implementation if a is selected at time $t+1$:

$$\begin{aligned} Q_{t+1}(a) &= \frac{k_a Q_t(a) + R_{t+1}}{k_a + 1} \\ &= \frac{k_a}{k_a + 1} Q_t(a) + \frac{1}{k_a + 1} R_{t+1} \\ &= Q_t(a) + \frac{1}{k_a + 1}\Big[R_{t+1} - Q_t(a)\Big] \end{aligned}$$

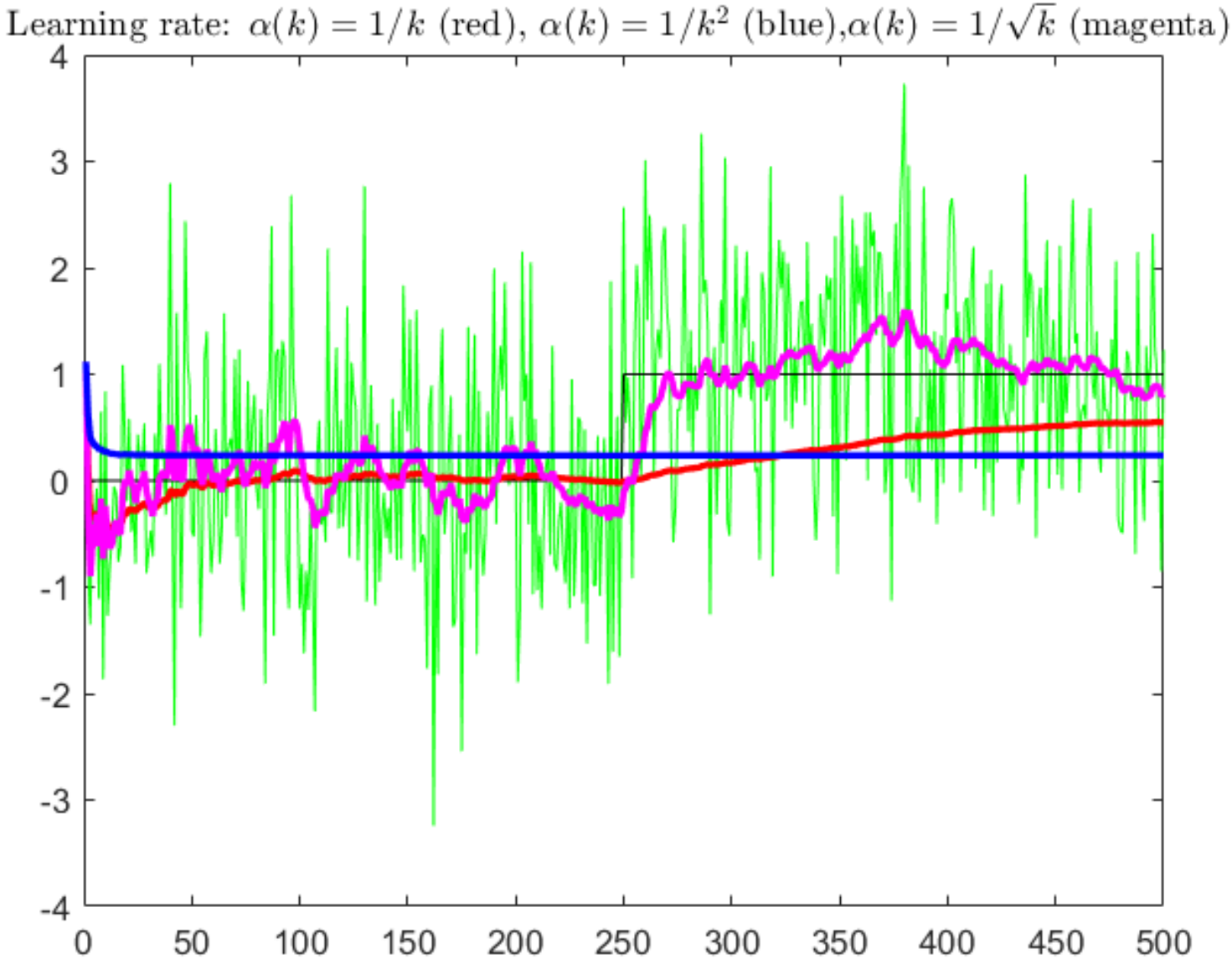
- Example of an update rule for estimates:

$$NewEst \leftarrow OldEst + LearningRate[NewData - OldEst]$$

Note on Learning Rate



Note on Learning Rate



Defining exploration vs. exploitation

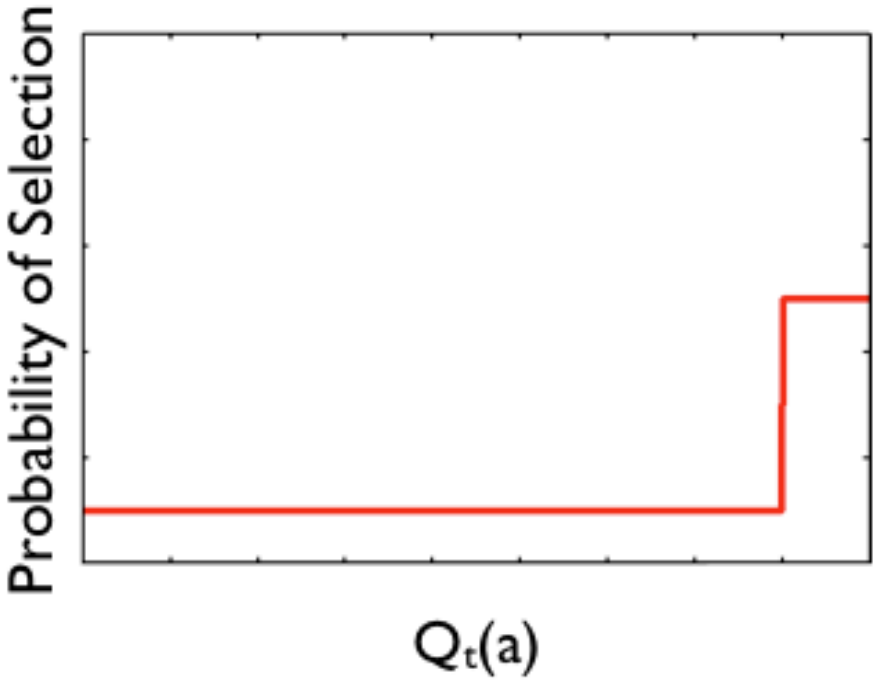
- When using action-value methods, exploration and exploitation are easy to define.
- **Exploiting** means taking the **greedy** action:

$$a^* = \arg\max_a Q_t(a)$$

- **Exploring** means taking any other action:
- Frequently used exploration strategies
  1. ϵ-greedy
  2. Soft-max

Epsilon-greedy exploration

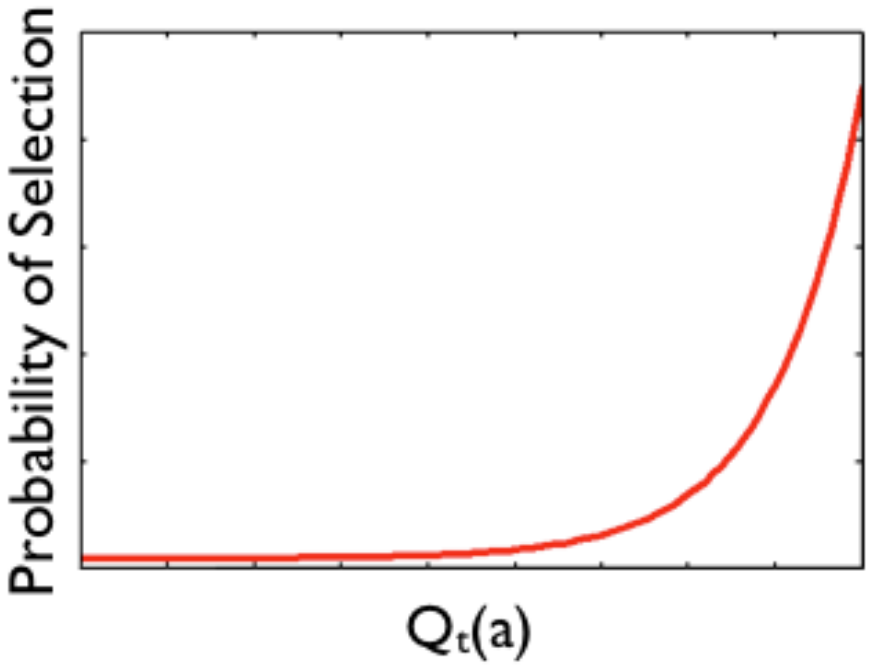
In ϵ-**greedy** exploration, the agent selects a random action with probability ϵ, and the greedy action otherwise



Softmax exploration

In **softmax** exploration, the agent chooses actions according to a **Boltzmann** distribution

$$p(a) = \frac{e^{Q(a)/\tau}}{\sum_{a'} e^{Q(a')/\tau}}$$

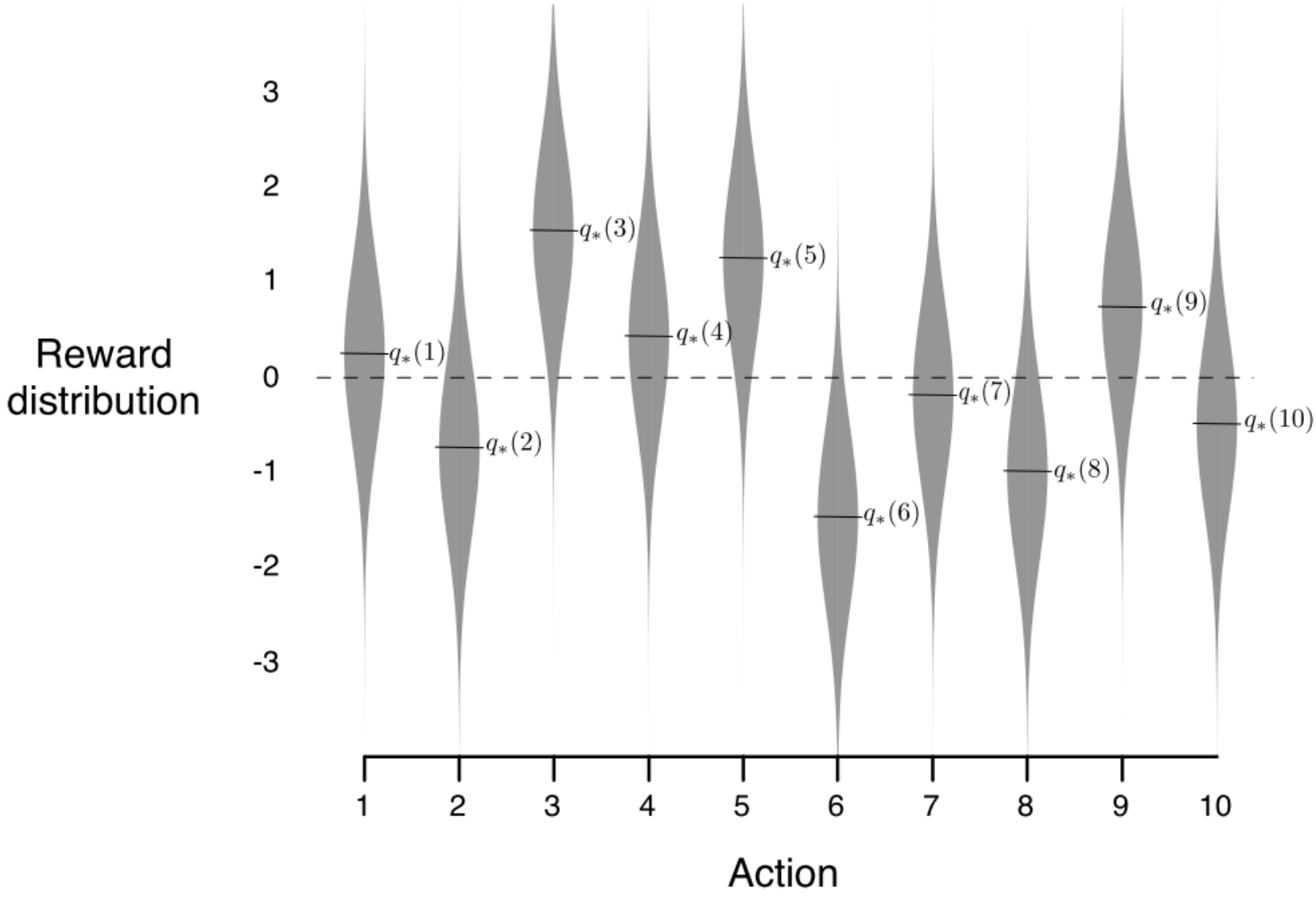


Optimistic vs. Realistic Initialization

How to ensure that every arm is sampled at least once (preferably: at beginning of exploration!)

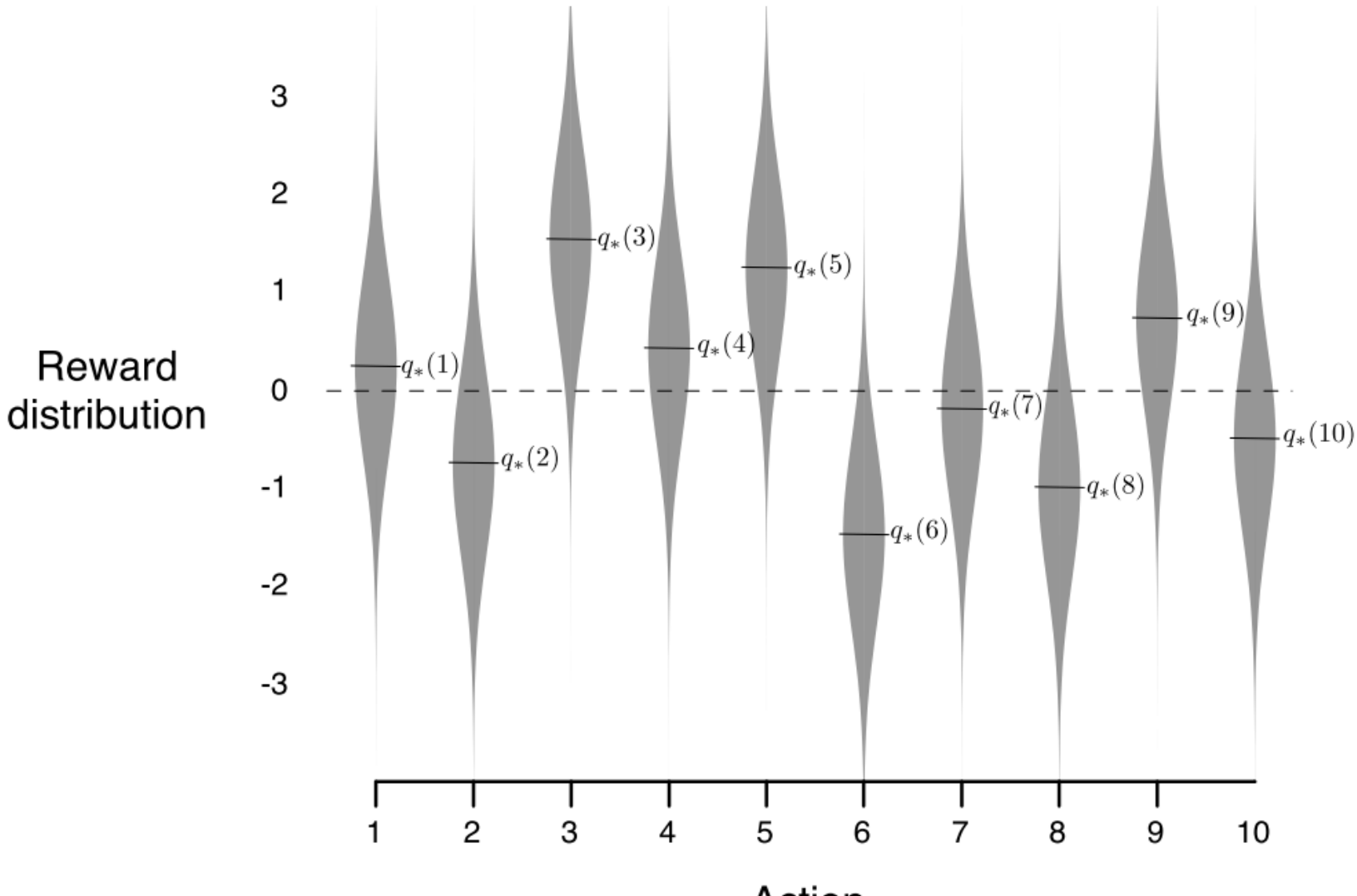
- In optimistic initialization, the agent initializes its action-value estimates higher than the largest possible reward
- The agent always selects the greedy action
- Rewards are always disappointing, directing the agent to the un-explored arms

Experiment: 10 bandits setup

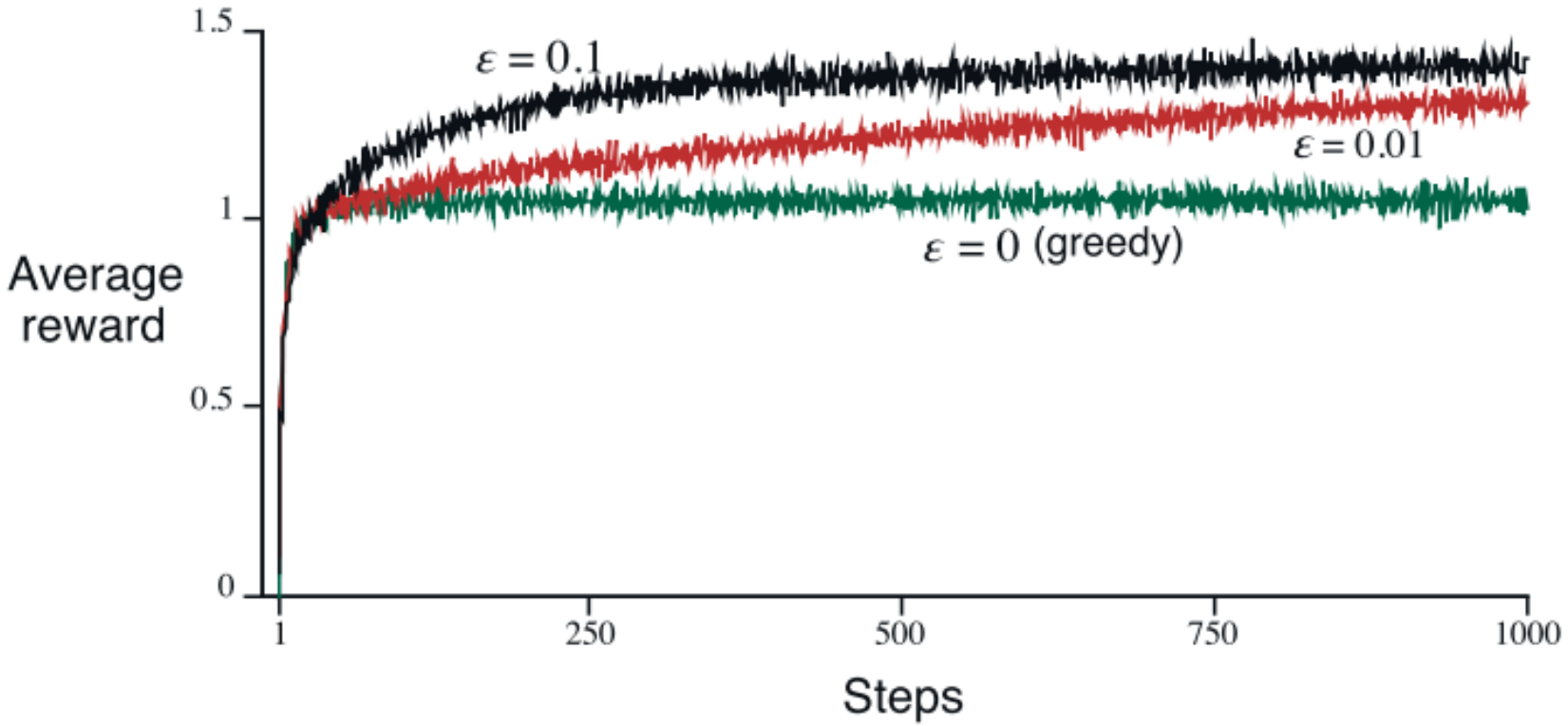


Experiment: 10 bandits setup: Typical result

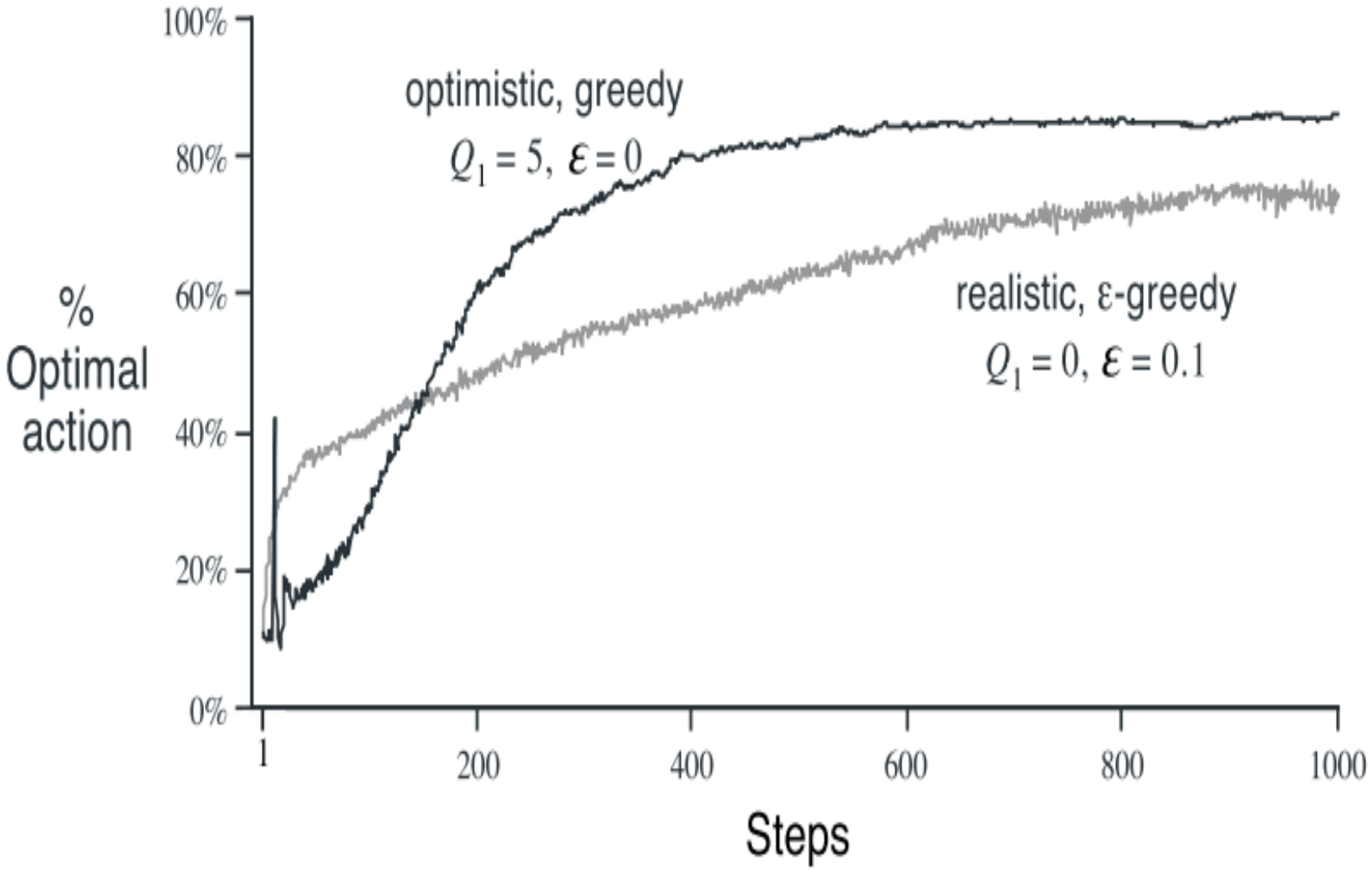
$k = 10, \epsilon = 0.1$, optimistic initialisation



Experiment: 10 bandits: greedy versus ϵ-greedy



Experiment: Realistic vs. Optimistic initialization



Can we do better?

- **Mean reward value** for each action: $q(a) := E(R \mid A = a)$
- **Optimal mean reward value** (unknown!):

$$q^* = q(a^*) = \max_a q(a)$$

- **Opportunity gap:** $\Delta_a := q^* - q(a)$
- **Regret at time t:** opportunity loss for one action at step t:

$$L_t := q^* - q(A_t)$$

- **Expected total regret** (up to time T):

$$\ell_T := E\left(\sum_{it=1}^{T} L_t\right) = E\left(\sum_{t=1}^{T}(q^* - q(A_i))\right)$$

Expected total regret

Expected total regret (up to time T):

$$\ell_T := E\left(\sum_{it=1}^{T} L_t\right) = E\left(\sum_{t=1}^{T}(q^* - q(A_i))\right)$$

- Opportunity gap: $\Delta_a := q^* - q(a)$;
- $N_t(a)$: number of selection of action a up till time t;
- Expected total regret:

$$\ell_t := E\left(\sum_{i=1}^{t} L_i\right) = \sum_a \Delta_a E N_t(a).$$

Expected total regret (proof)

$$\ell_T := E\left(\sum_{t=1}^{T} L_i\right) = \sum_a \Delta_a EN_T(a).$$

Proof

$$\begin{aligned}
\ell_T &= E\left(\sum_{t=1}^{T}(q^* - q(A_t))\right) \\
&= \sum_{t=1}^{T}\left(\sum_a (q^* - q(a))\, P(A_t = a)\right) \\
&= \sum_a \Delta_a \left\{\sum_{t=1}^{T} P(A_t = a)\right\} = \sum_a \Delta_a\, EN_T(a).
\end{aligned}$$

We are using: $E1_{(A=a)} = 1 \cdot P(A=a) + 0 \cdot P(A \neq a) = P(A=a)$

Expected total regret for ϵ-greedy

- Expected total regret:

$$\ell_t = \sum_a \Delta_a \, EN_t(a).$$

- For ϵ-greedy (when we have found optimal a^*):

$$EN_t(a) = \begin{cases} (1-\epsilon)t & \text{if } a = a^*, \Delta_a = 0 \\ \dfrac{\epsilon}{k-1} t & \text{if } a \neq a^*, , \Delta_a > 0 \end{cases}$$

Hence: $L_t = (\epsilon \overline{\Delta})\, t$ where $\overline{\Delta} = \dfrac{1}{k-1} \sum_{a \neq a^*} \Delta_a$

- For constant exploration (ϵ) total regret grows linear in t,

Can we do better? Lai-Robbins

Lai & Robbins (1985)

Asymptotically, total regret is at least logarithmic in number of steps:

$$\ell_t \geq A \log t \qquad \text{as } t \longrightarrow \infty$$

where

$$A = \sum_{a:\Delta_a>0} \frac{\Delta_a}{KL(f_a; f_a^*)}$$

- $KL(f; g)$ Kullback-Leibler divergence;
- Hard problems: similar looking distributions with different means;
- What exploration schemes would give rise to this performance?

UCB: Upper confidence bounds

- **UCB: Optimism in the face of uncertainty!**
- Neither ϵ-greedy nor softmax consider uncertainty in action-value estimates
- Goal of exploration: reduce uncertainty
- So focus exploration on most uncertain actions
- Compute confidence intervals for each action
- Always take action with highest upper bound

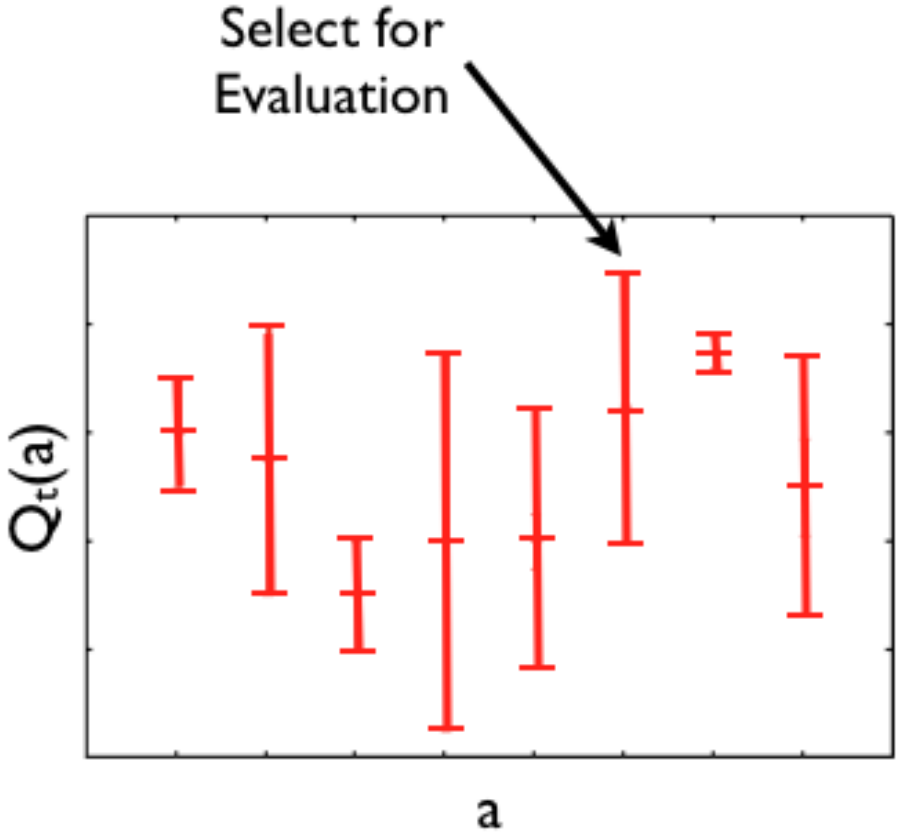
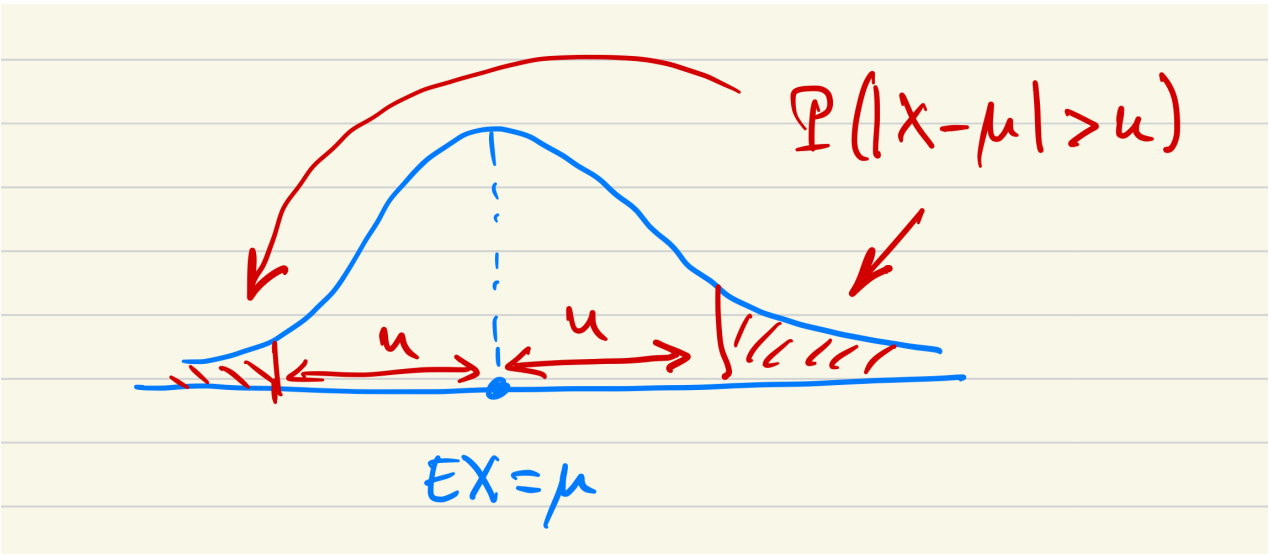


Table of Contents

Aside: Concentration Inequalities

- **Concentration inequalities** provide bounds on how much a random variable X can deviate from some value (typically, its expected value $E(X)$):

$$P(|X - E(X)| > u) = \quad ??$$

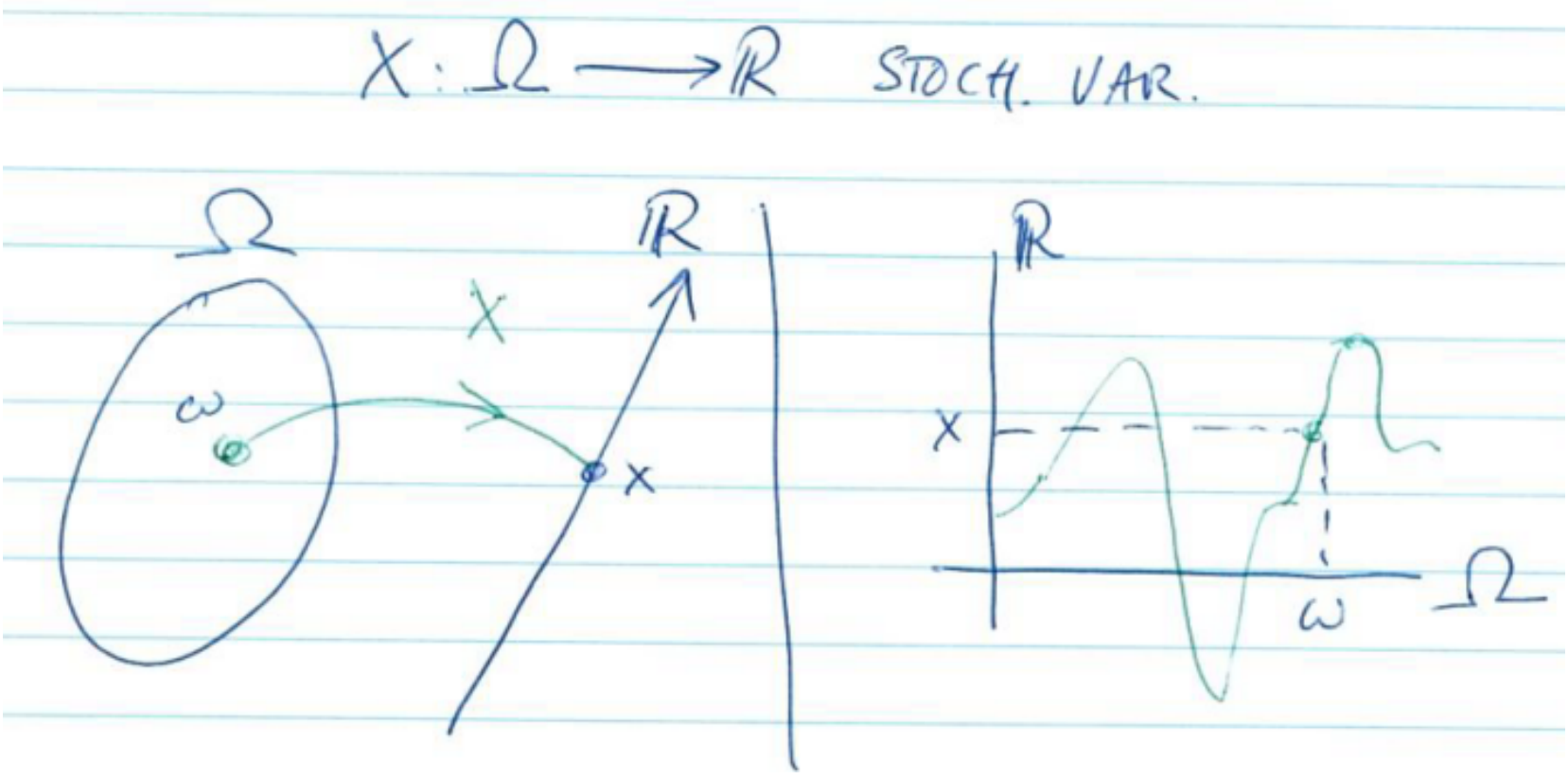


Concentration inequality 刻画了一组随机变量的和（或者样本平均数）与其期望值的偏离程度，在算法收敛性分析过程中是非常有用的一类不等式。

Aside: Concentration Inequalities

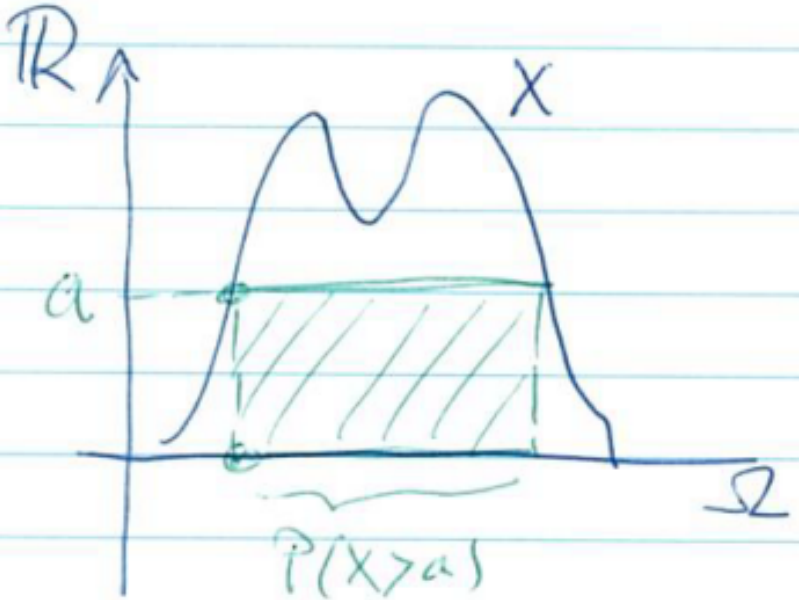
- **General but weak bounds:** Markov, Chebychev,
- For **sums (or means) of independent rv's;** stronger bounds!
  - Hoeffding, Azuma, ...

Markov – Chebychev (1)



Markov – Chebychev (2)

CHEBYSHEV inequality



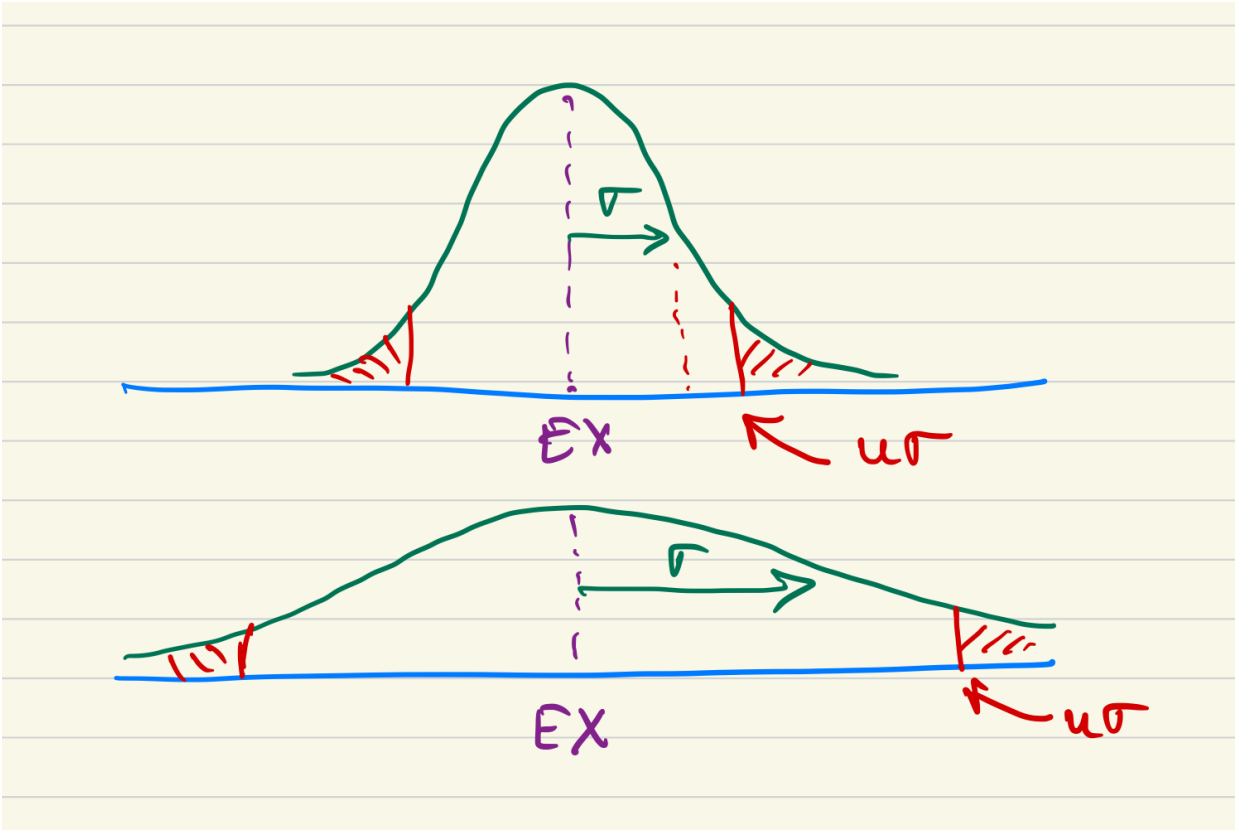
$X \geq 0$ Stoch. Var.

$$a \cdot P(X > a) \leq EX = \int X \, dP.$$

$$\boxed{P(X > a) \leq \frac{EX}{a}}$$

Markov – Chebychev (3)

- Standard deviation σ provides natural yard stick (unit-length)!
- So it is natural to look at standardized deviation $|X - \mu|/\sigma$, i.e. deviation $|X - \mu|$ relative to σ.



Markov – Chebychev (3)

- For $Y \geq 0$ we have:

$$P(Y > u) \leq \frac{EY}{u}$$

- Now take

$$Y = \frac{|X - \mu|}{\sigma} \quad \text{then} \quad EY^2 = 1$$

$$P\left(\frac{|X - \mu|}{\sigma} > u\right) = P(Y > u) = P(Y^2 > u^2) \quad \leq \frac{1}{u^2}$$

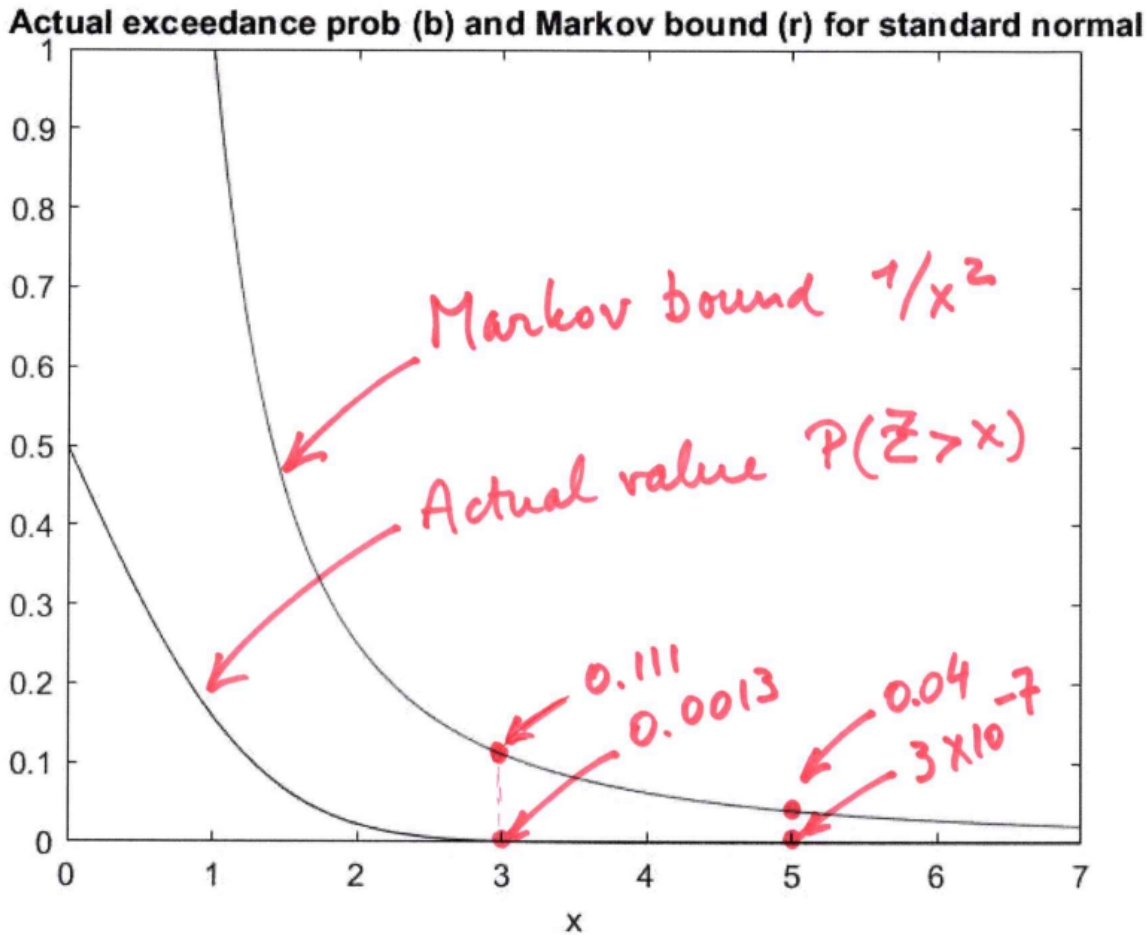
- Markov-Chebychev inequality:

$$P\left(\frac{|X - \mu|}{\sigma} > u\right) \leq \frac{1}{u^2}$$

Markov – Chebychev (4)

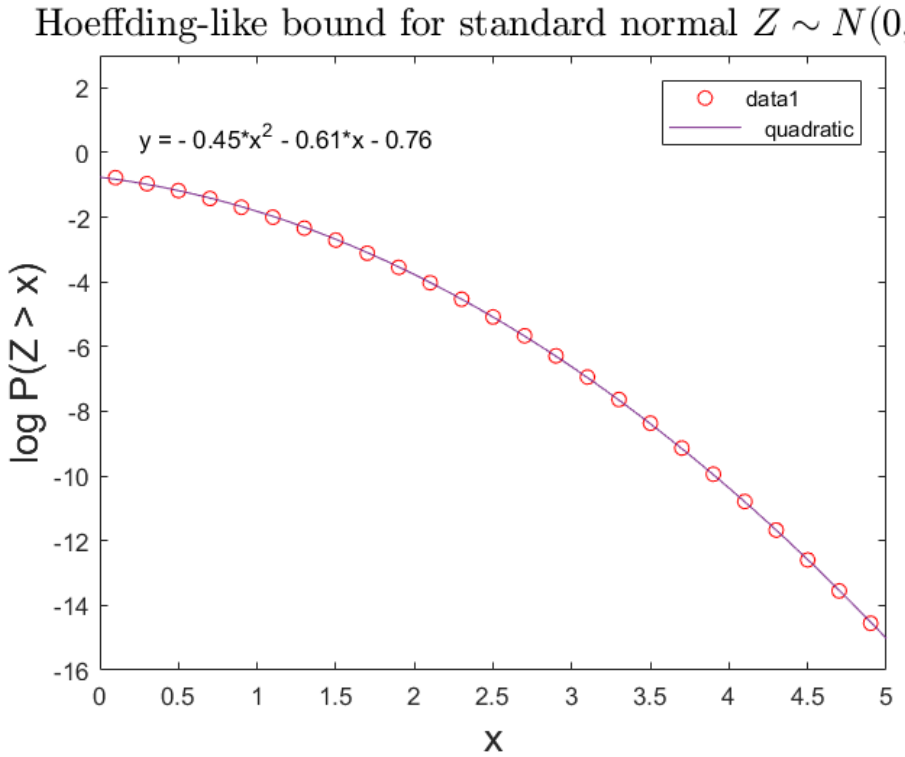
The Markov-Chebychev bound is not very tight!

$$\text{For } Z \sim N(0,1): \quad P(Z > x) \leq 1/x^2$$



Hoeffding inequality: Numerical experiment

From numerical experiments for $Z \sim N(0,1)$ we observe



$$\log P(Z > x) \approx -\alpha x^2$$

$$\log P(\sigma Z > \sigma x) \approx -\alpha x^2$$

$$\log P(\sigma Z > u) \approx -\alpha u^2/\sigma^2$$

$$P(\sigma Z > u) \approx e^{-\alpha u^2/\sigma^2}$$

$$P\left(\frac{\sigma}{\sqrt{n}} Z > u\right) \approx e^{-\alpha n u^2/\sigma^2}$$

Fron CLT: $\overline{X}_n \sim N(\mu, \sigma^2/n) \implies P(\overline{X}_n > \mu + u) \approx e^{-nu^2/2\sigma^2}$

Hoeffding inequality for sum of bounded rv's

Hoeffding's inequality

Let $X_1, X_2, \ldots, X_n$ be i.i.d. (bounded) random variables such that $c \leq X_i \leq d$ (where $L = d - c < \infty$) with mean $\mu := E(X_i)$.
Consider the sample mean:

$$\overline{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$$

Then the exceedance probability is exponentially bounded.

$$P(\overline{X} \geq \mu + u) = P(\mu \leq \overline{X} - u) \leq e^{-2nu^2/L^2}$$

and similarly:

$$P(\overline{X} \leq \mu - u) \equiv P(\mu \geq \overline{X} + u) \leq e^{-2nu^2/L^2},$$

UCB: Applying Hoeffding to Upper Confidence Bounds

Hoeffding's inequality

$$P(\overline{X} \geq \mu + u) = P(\mu \leq \overline{X} - u) \leq e^{-2nu^2/L^2}$$
$$P(\overline{X} \leq \mu - u) = P(\mu \geq \overline{X} + u) \leq e^{-2nu^2/L^2},$$

Apply to estimation of bandit reward

How does **unknown real mean** $q(a)$ be estimated by **observed sample mean** $Q_t(a)$?

$$p(t) := P\left(q(a) > Q_t(a) + U_t(a)\right) \leq e^{-2N_t(a)U_t(a)^2/L^2}$$

UCB: Applying Hoeffding to Upper Confidence Bounds

Hoeffding's upper bound:

$$p(t) := P\left(q(a) > Q_t(a) + U_t(a)\right) \leq e^{-2N_t(a)U_t(a)^2/L^2}$$

Given exceedance probability ($p(t)$), solve for upper limit $U_t(a)$:

$$U_t(a) = c\sqrt{\frac{-\log p(t)}{N_t(a)}}, \qquad \text{where} \quad c = L/\sqrt{2.}$$

Now, let exceedance prob. go to zero over time: e.g. p(t) $= t^{-1}$;

$$U_t(a) = c\sqrt{\frac{\log t}{N_t(a)}}.$$

UCB1-Algorithm: Upper confidence bounds Optimism in the face of uncertainty!

UCB1-Algorithm

$$a^*_{t+1} = \arg\max_a \left(Q_t(a) + c \sqrt{\frac{\log t}{N_t(a)}} \right)$$

- DON'T choose the arm that has performed best sofar, but...
- DO choose the one that could **reasonably** perform best in the future!
- Optimism in the face of uncertainty! :-)

UCB1-Algorithm

$$a^*_{t+1} = \arg\max_a \left(Q_t(a) + c \sqrt{\frac{\log t}{N_t(a)}} \right)$$

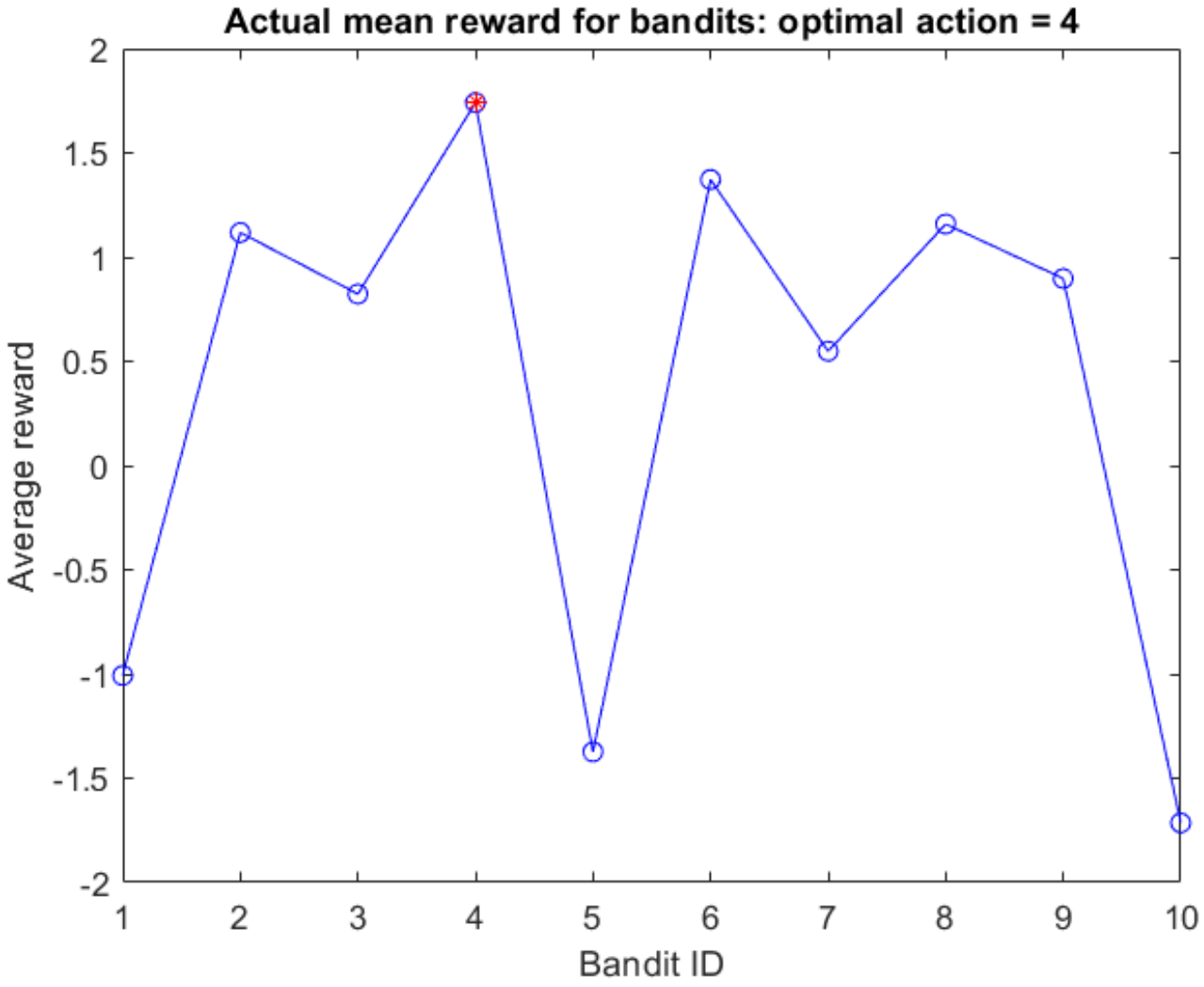
- The parameter c is some tunable width parameter;
- Every time action a is sampled, $N_t(a)$ increases, hence $U_t(a)$ decreases;
- Every time a is sampled, the UCB for the *other* actions increases (since t increases); Every action is sampled eventually!
- UCB1 algorithm **achieves logarithmic asymptotic total regret**!

UCB1-Algorithm

$$a^*_{t+1} = \arg\max_a \left(Q_t(a) + c \sqrt{\frac{\log t}{N_t(a)}} \right)$$

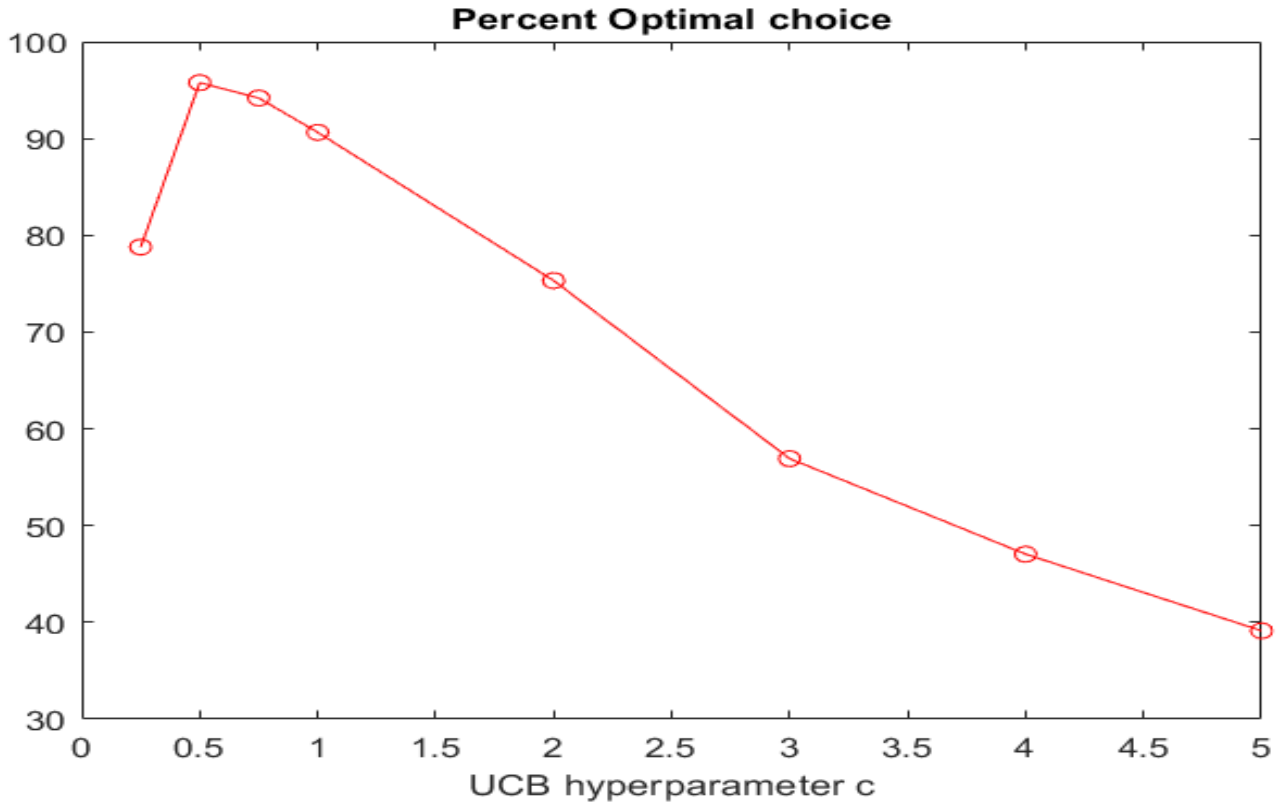
- UCB1 algorithm **achieves logarithmic asymptotic total regret**!
- **Take home message** (B. Christian & T. Griffiths):
  In the long run, optimism is the best prevention for regret!

UCB for k-bandit problem



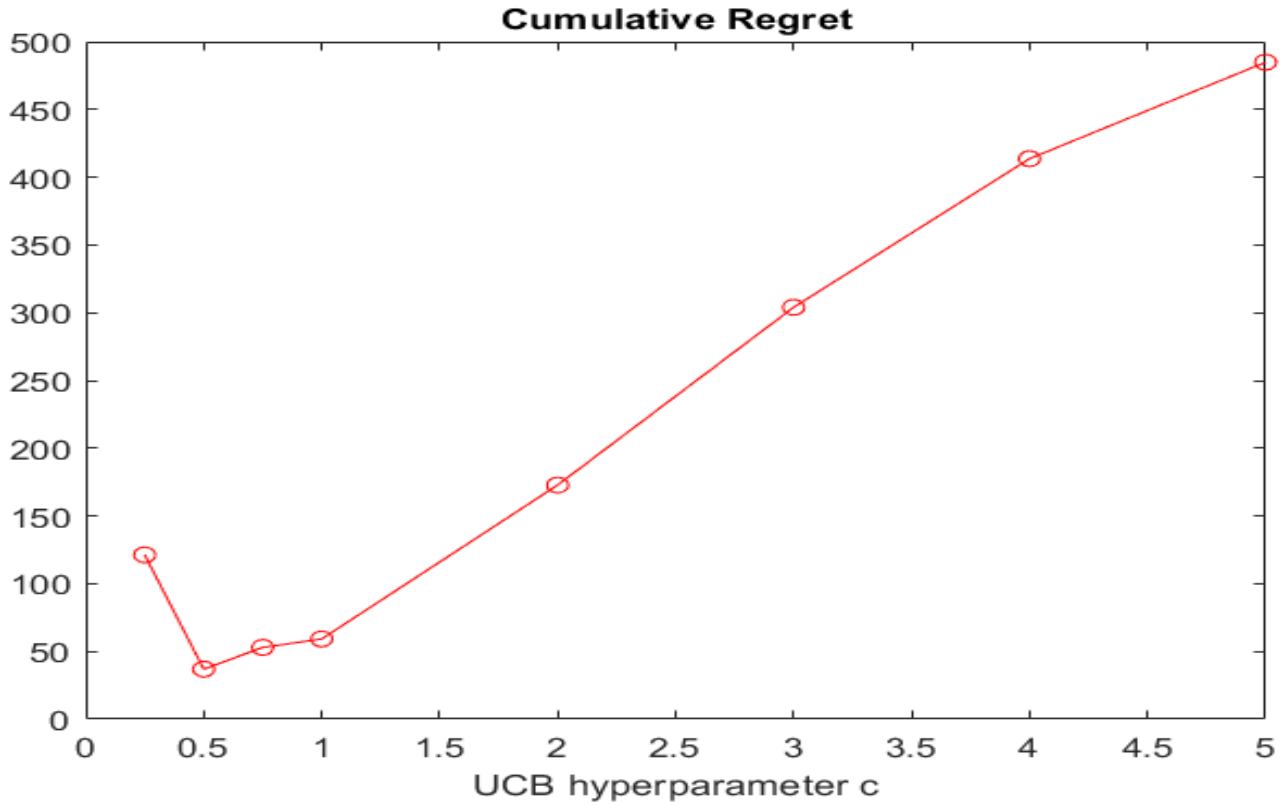
UCB for k-bandit problem

- Duration of game (number of arms pulled): $T = 1000$
- Averaged over 10 experiments (same k-bandit problem)

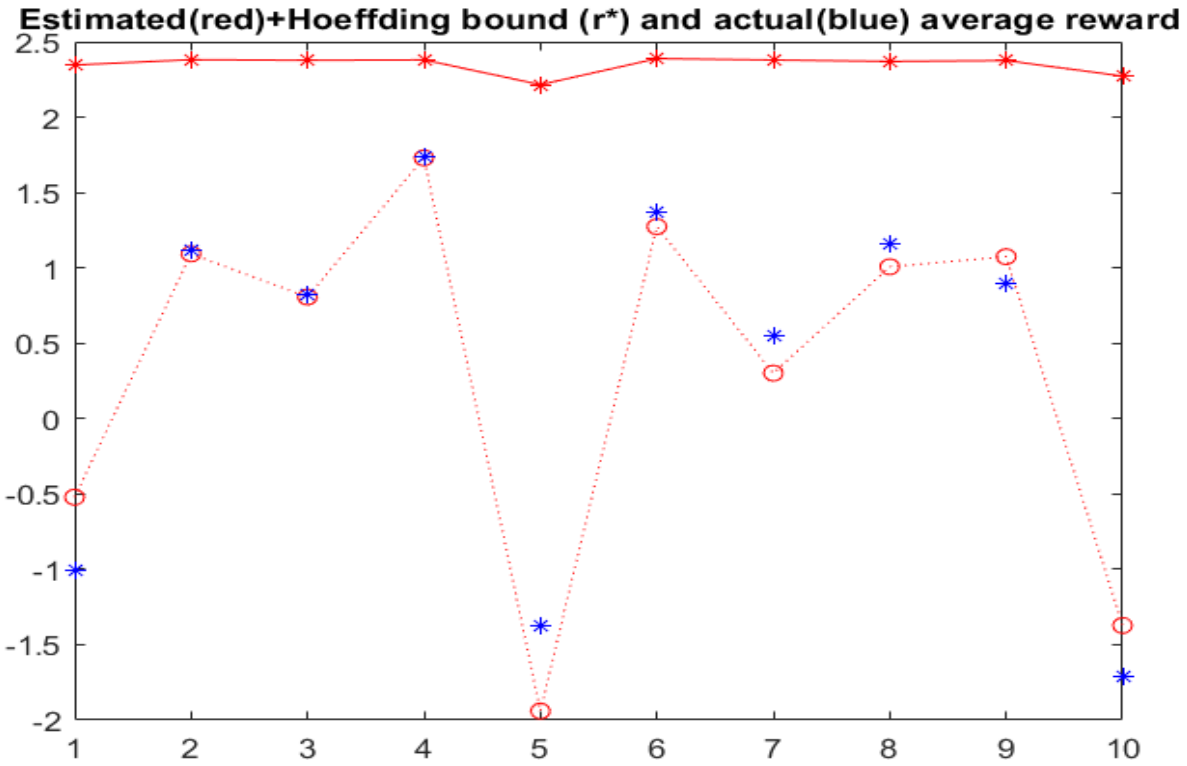


UCB for k-bandit problem

- Duration of game (number of arms pulled): $T = 1000$
- Averaged over 10 experiments (same k-bandit problem)

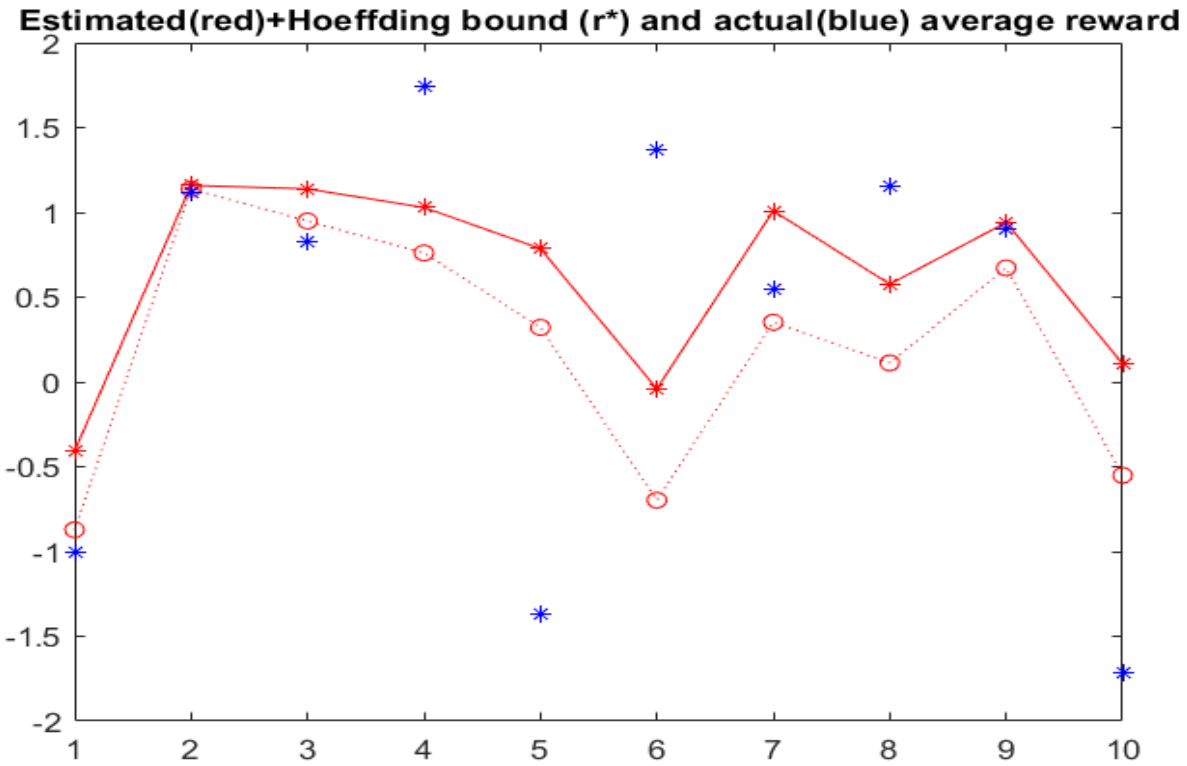


UCB: Influence of hyper-parameter $c = 5$ (large)



- c large $\Longrightarrow$ changes in UCB are inflated;
- Max UCB changes more frequently, hence more exploration.

UCB: Influence of hyper-parameter: $c = 0.5$ (small);



- c small $\Longrightarrow$ changes in UCB are tempered;
- Max UCB changes less frequently, hence less exploration.

UCB Experiment: Evolution of Total Regret

UCB algo applied to $k = 10, c = 1$, averaged over 100 experiments,

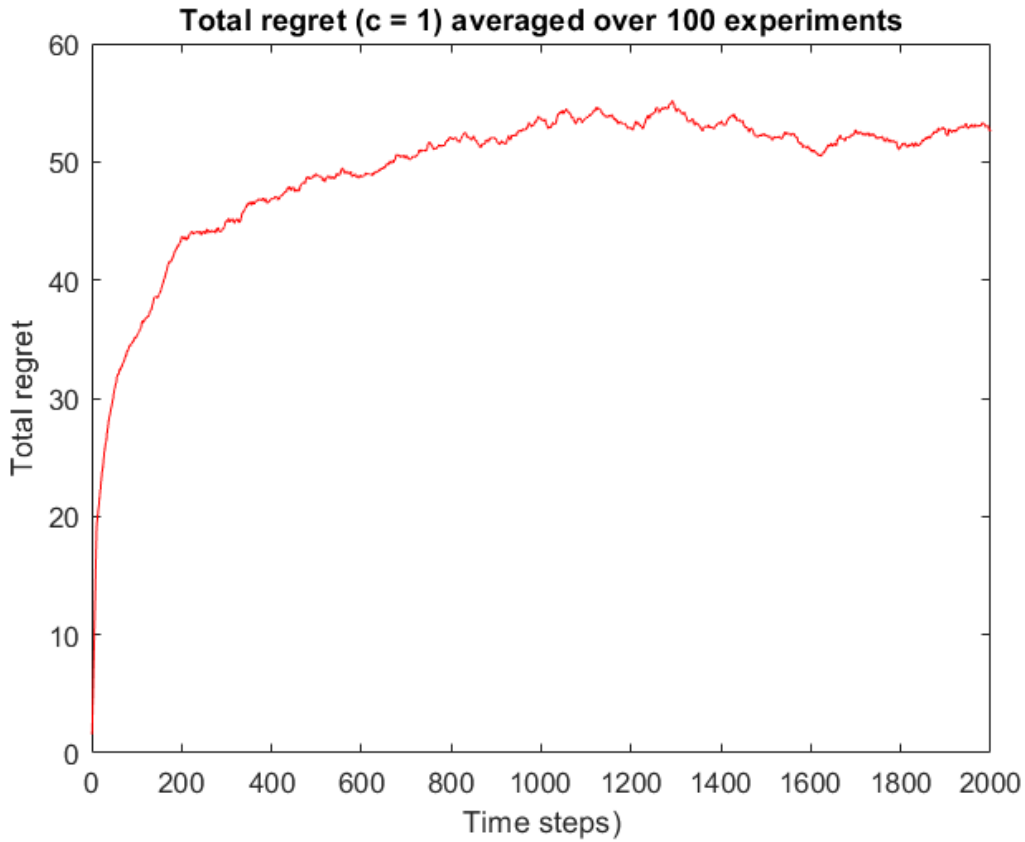


Figure: $\ell_t \sim \log(t)$ where $t =$ number of timesteps

UCB Experiment: Evolution of Total Regret

UCB algo applied to $k = 10, c = 1$, averaged over 100 experiments

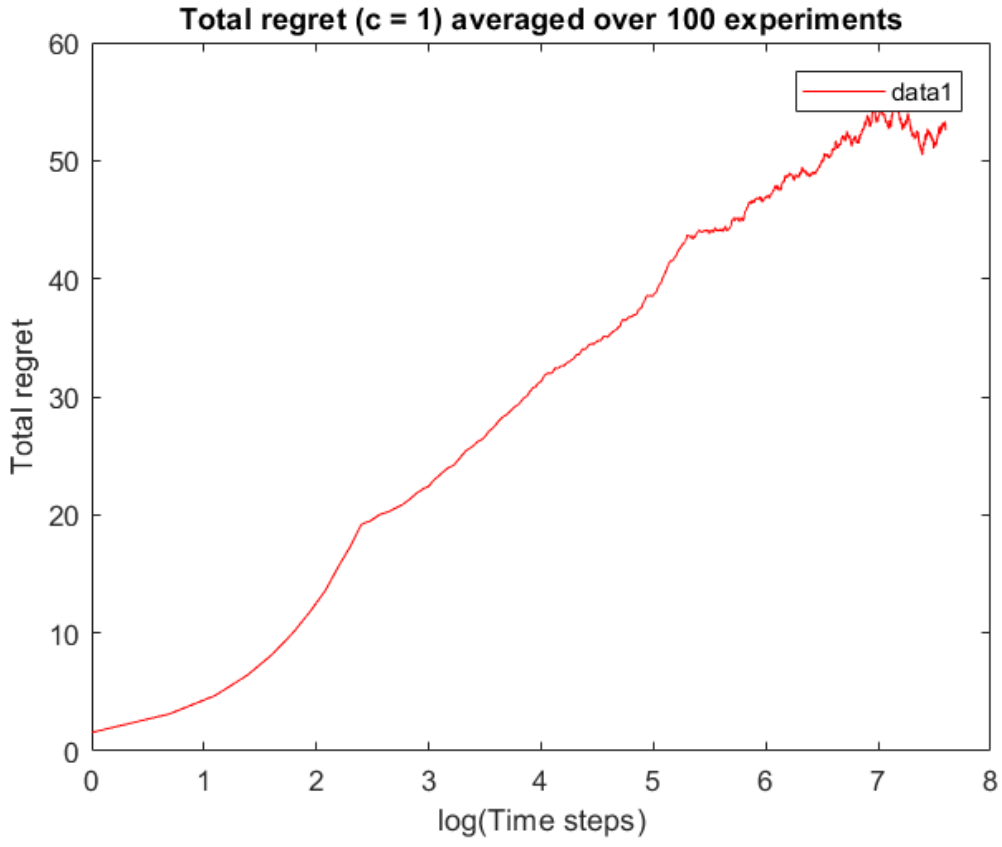
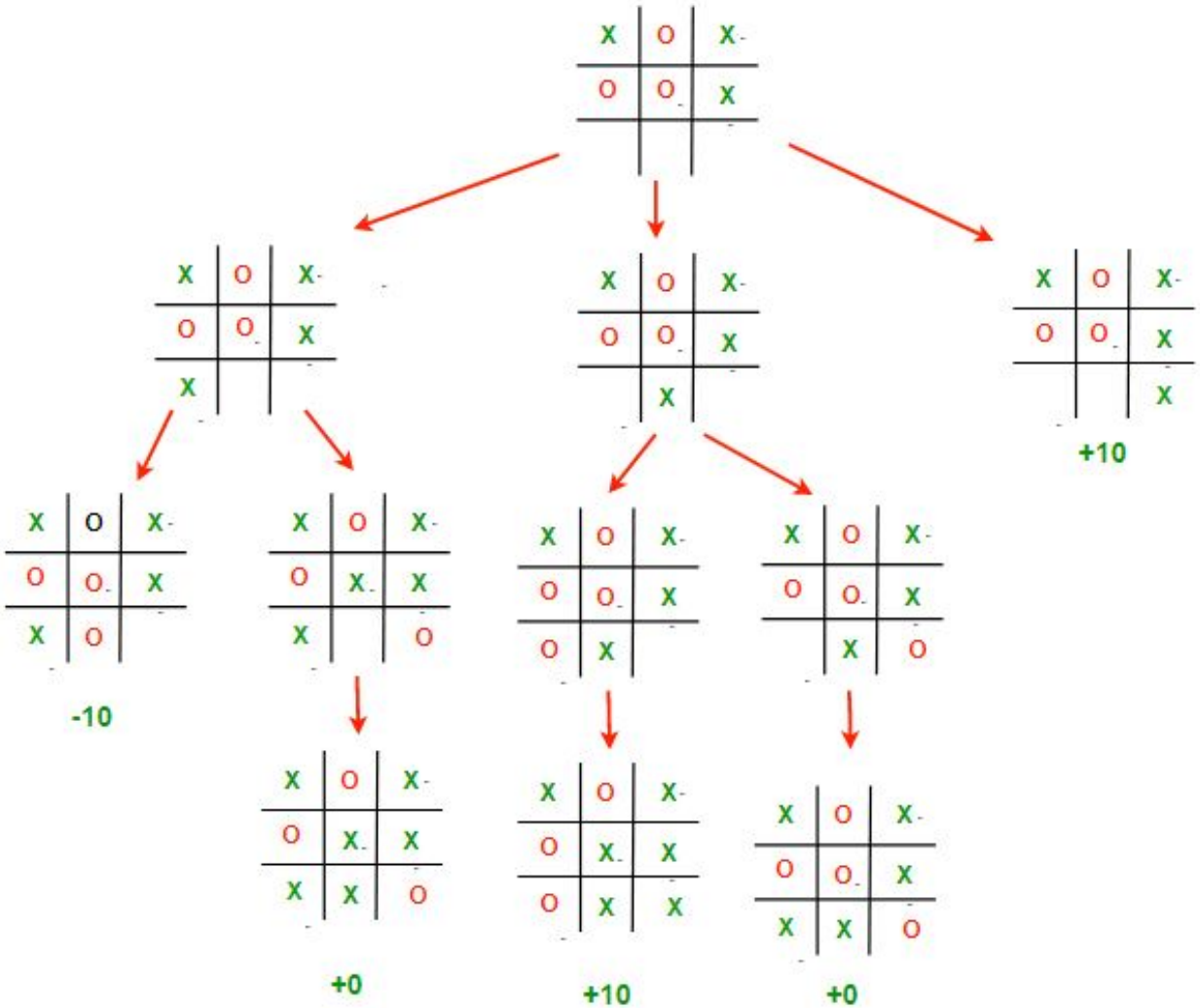


Figure: Recalibrating time-axis as $\log(t)$ yields roughly linear evolution for total regret ℓ_t (confirming $\ell_t \sim A \log(t)$).

Table of Contents

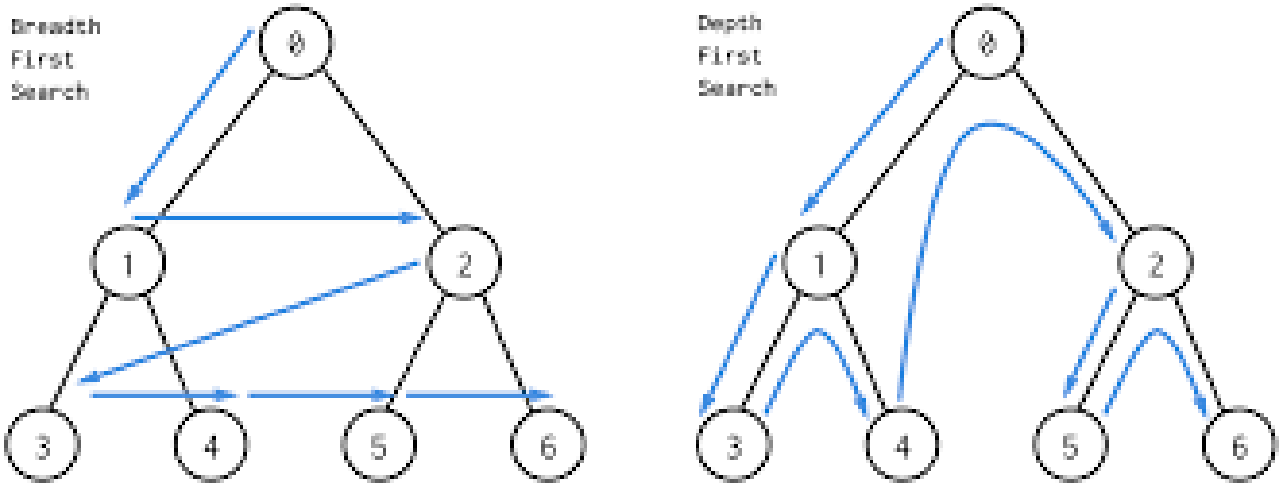
Tree Search

Tree search: generic problem underlying many AI tasks:



Tree Search Algo's

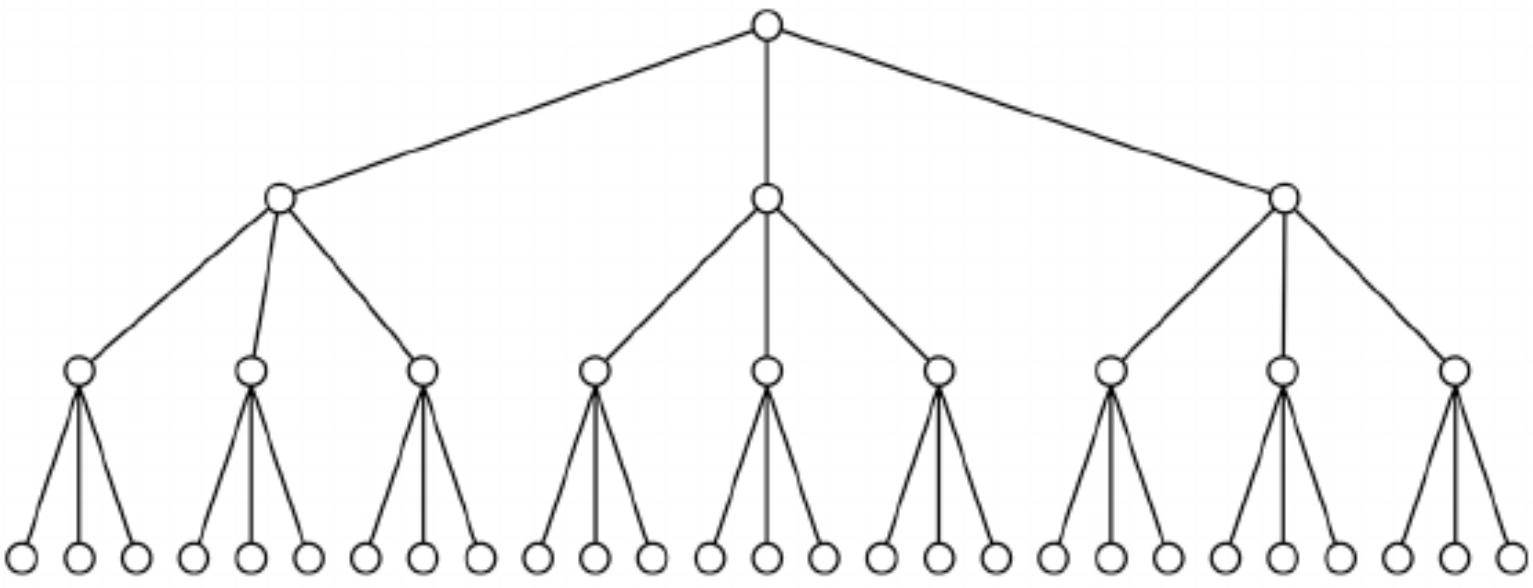
- Uninformed search: breadth or depth first search;



- A^* **search**: introduce heuristics that *inform search*;
- **Minimax search** (2 player adversarial) with $\alpha\beta$ pruning; Remove nodes outside optimal window;

Tree Search: Branching factor and depth

Difficulty: Exponentially growth of Size of search space grows exponentially: $S \sim b^d$



Monte Carlo Tree Search (MCTS)

MCTS: main idea

- Decision-time planning
- Prioritise computational budget:
  - No systematic scan of all nodes
  - Primarily focus on nodes that look promising
- Balance exploration and exploitation, e.g.
  - ϵ-greedy, or UCB1 (UCT: Upper Confidence bounds for Trees)
- MCTS is an important ingredient in many of the recent success stories in game AI (e.g. AlphaZero GO);

MCTS: Tree policy vs. roll-out

MCTS: Amplification of loop

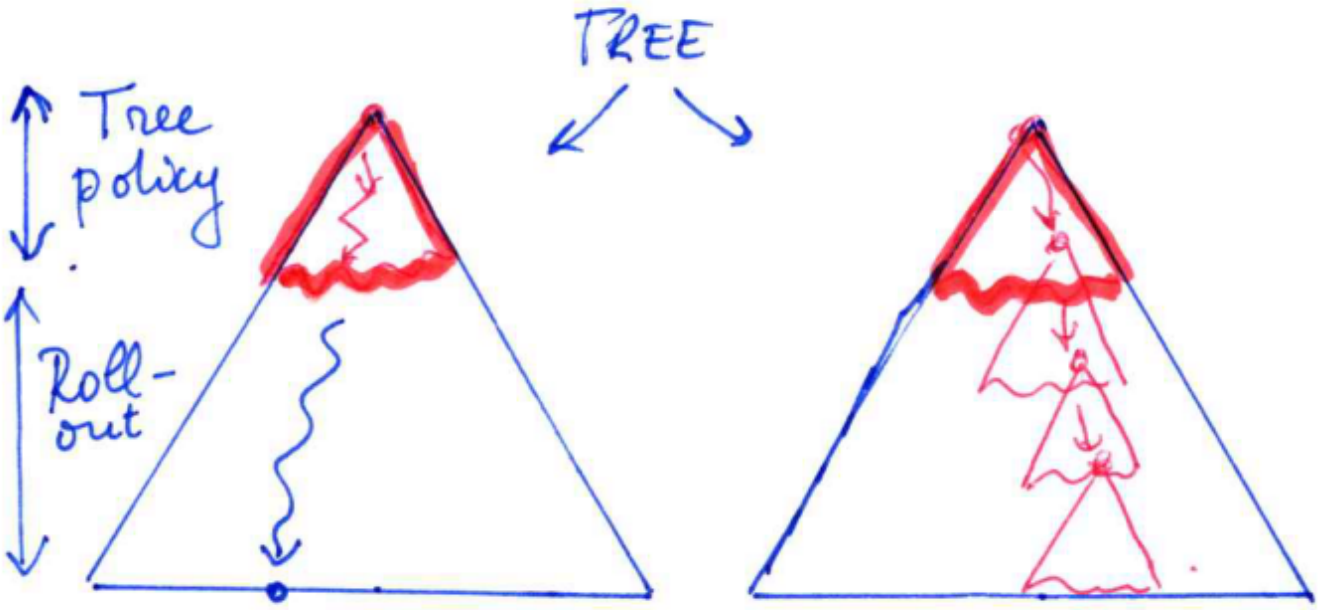
- **Tree traversal and node selection:**
  - use tree policy to construct path from root to (most) promising "tree policy (aka snowcap)" leaf node;
  - Tree policy: Always choose child node with best (but finite) UCB-value:

$$UCB(node_i) = \overline{x}_i + c\sqrt{\frac{\log N}{n_i}}$$

    $\overline{x}_i$: mean node value; n_i: #visits of node i; N #visits parent;
  - Proceed till tree policy leaf node has been reached: this node has children that haven't been explored. .
- **Expansion** of selected (tree policy) leaf node; i.e. Randomly pick unexplored child (action).
- **Simulation** based on roll-out policy
  - roll-out: *random* or *fast heuristic* (roll-out states not stored!);
- **Backup:** update values along tree traversal path ;

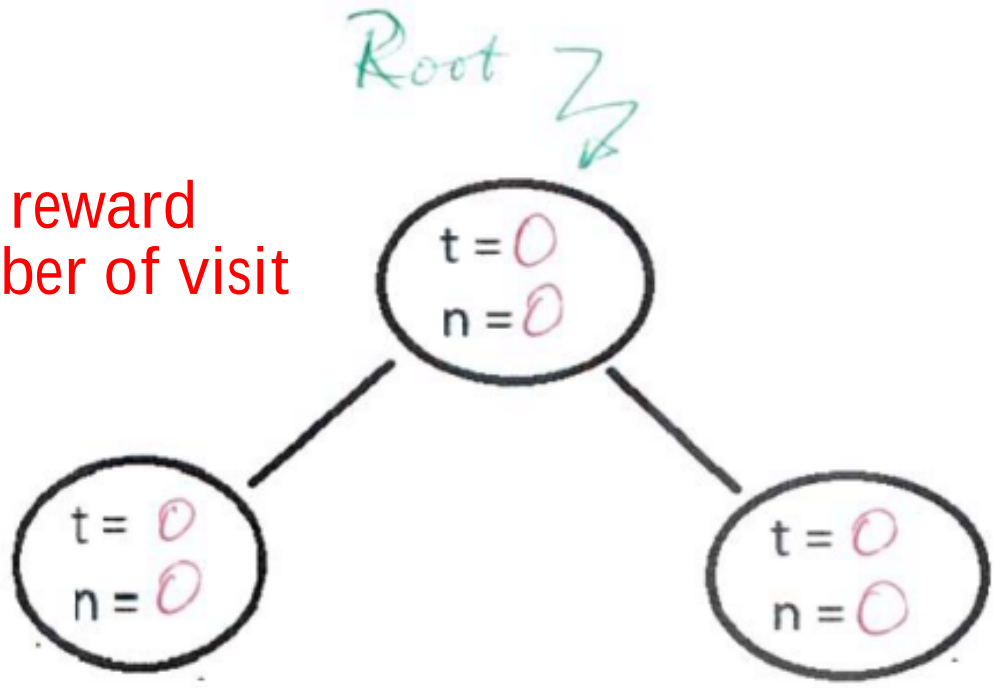
MCTS: Tree policy vs. roll-out

- Repeat loop (always starting in same root node) until predetermined computational budget has been exhausted.
- Then choose the best child-node as new root node and repeat.



MCTS: Worked Example

t: total reward
n: number of visit

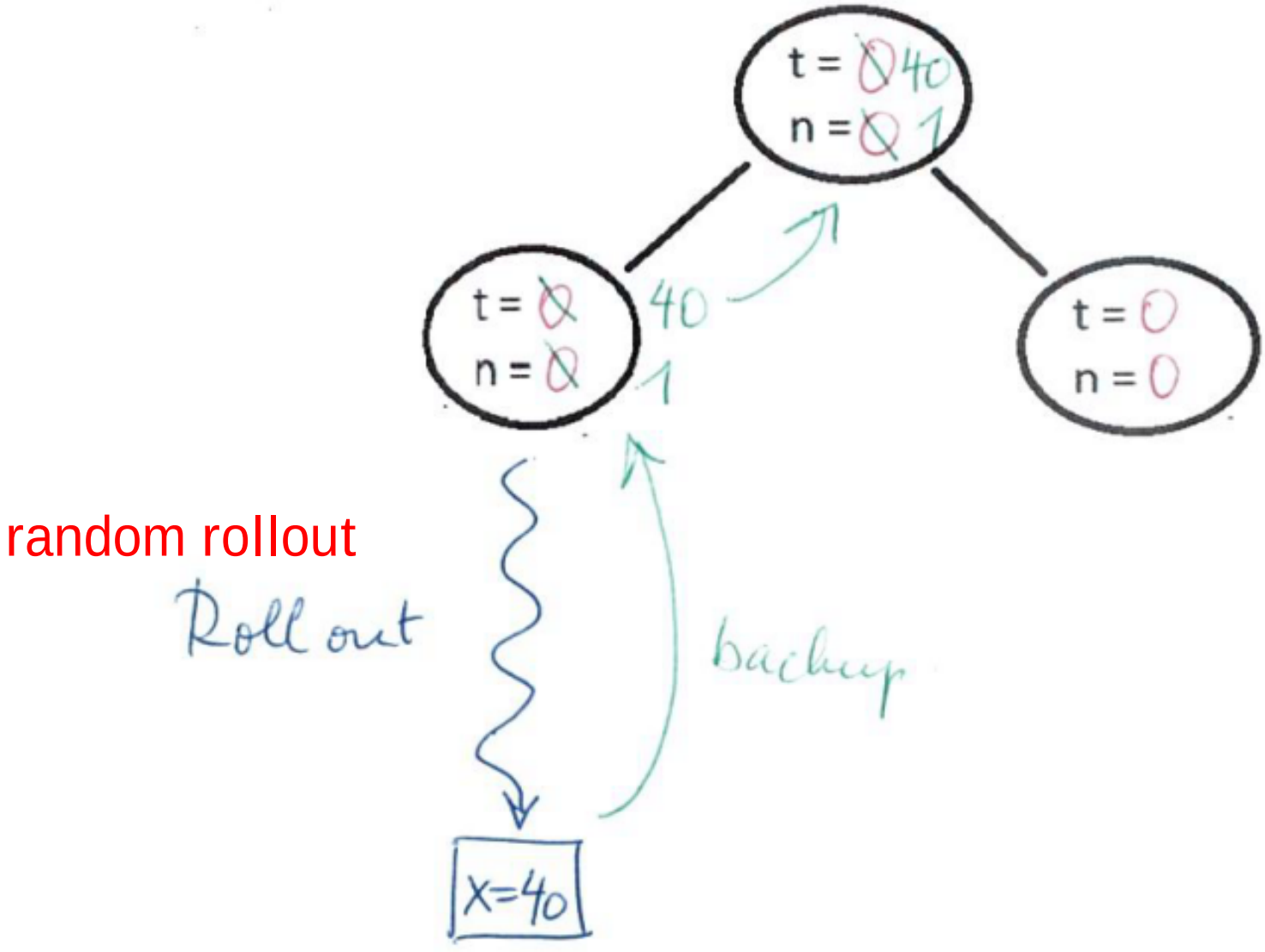


$$UCB_1(s_i) = \bar{x}_i + 2\sqrt{\frac{\log N}{n_i}}$$

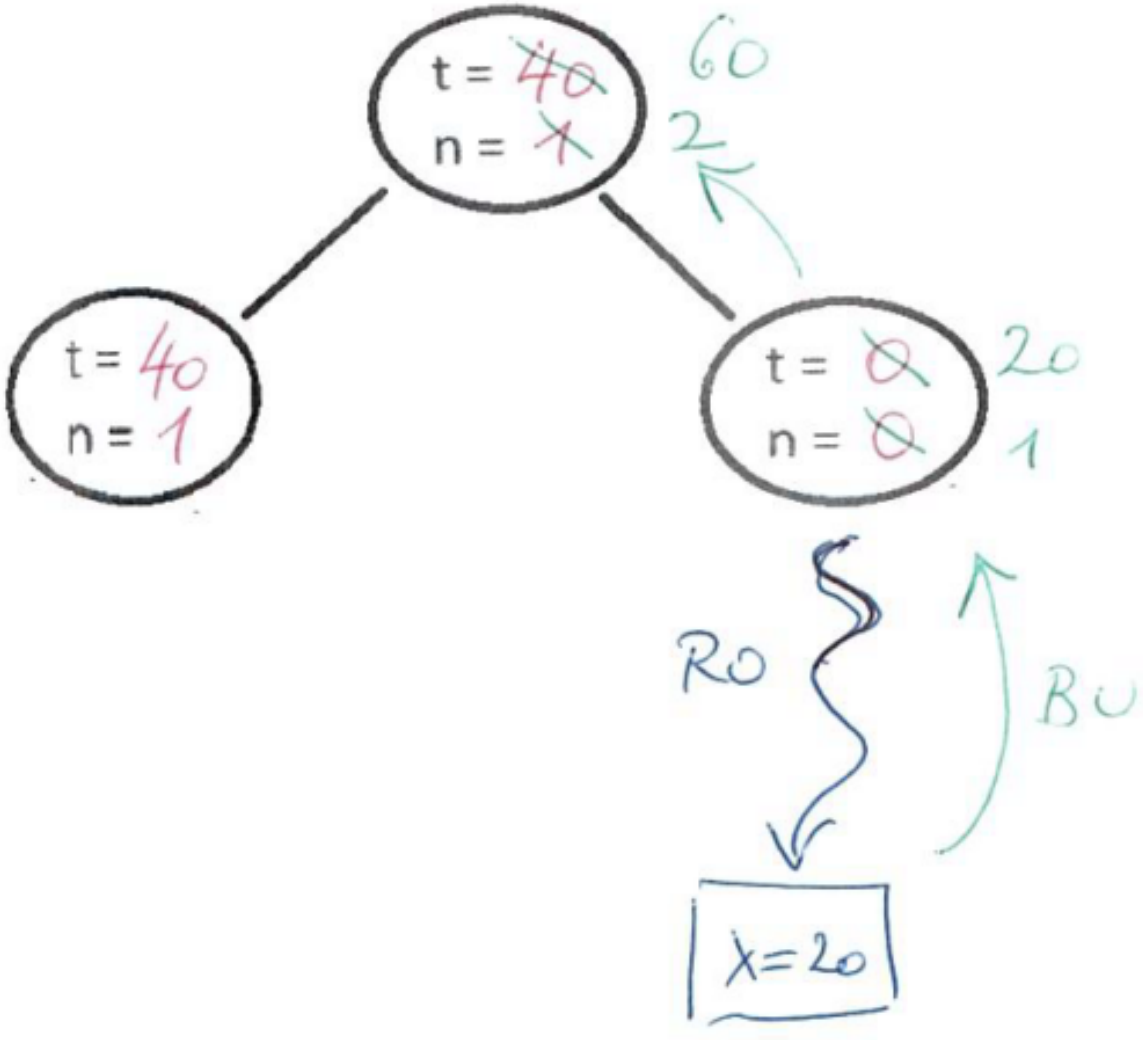
$\log N$ ← # visits parent
n_i ← # visits node.

What's the meaning of ni?
-> # of visit for child node

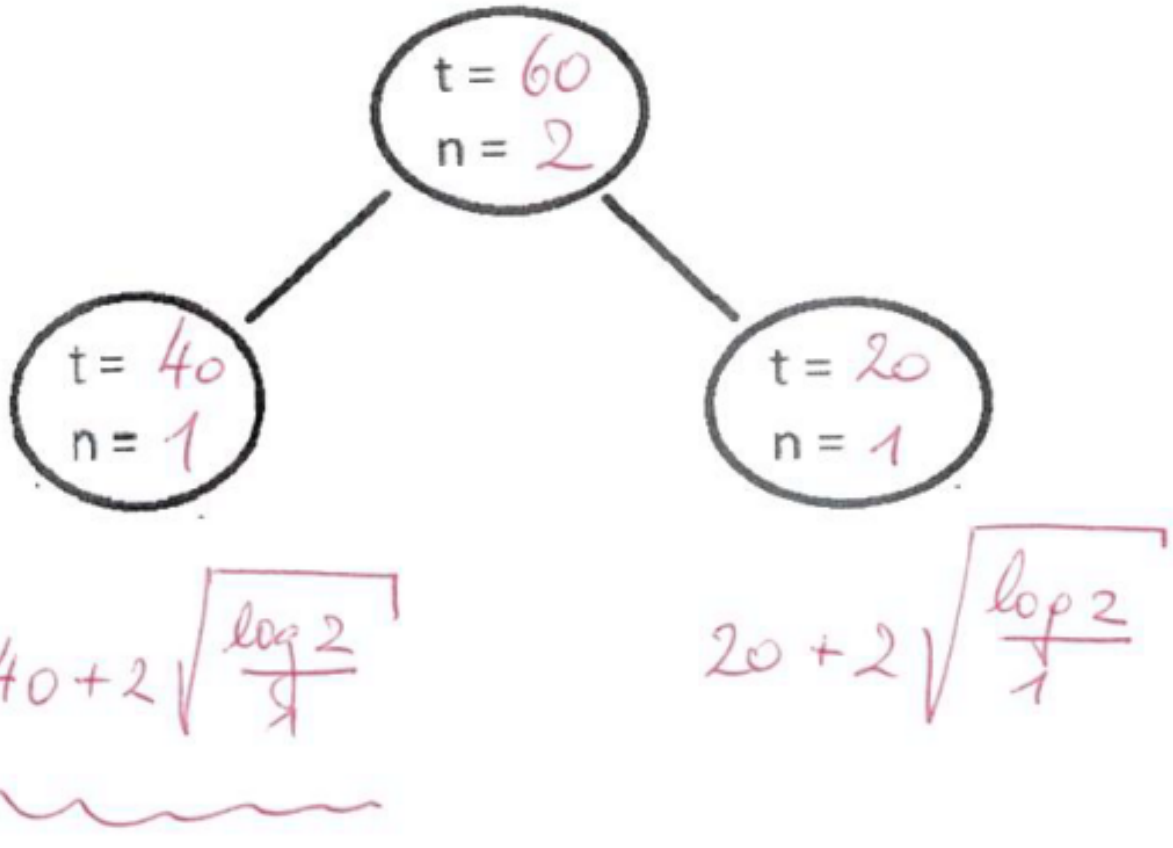
MCTS: Worked Example



MCTS: Worked Example

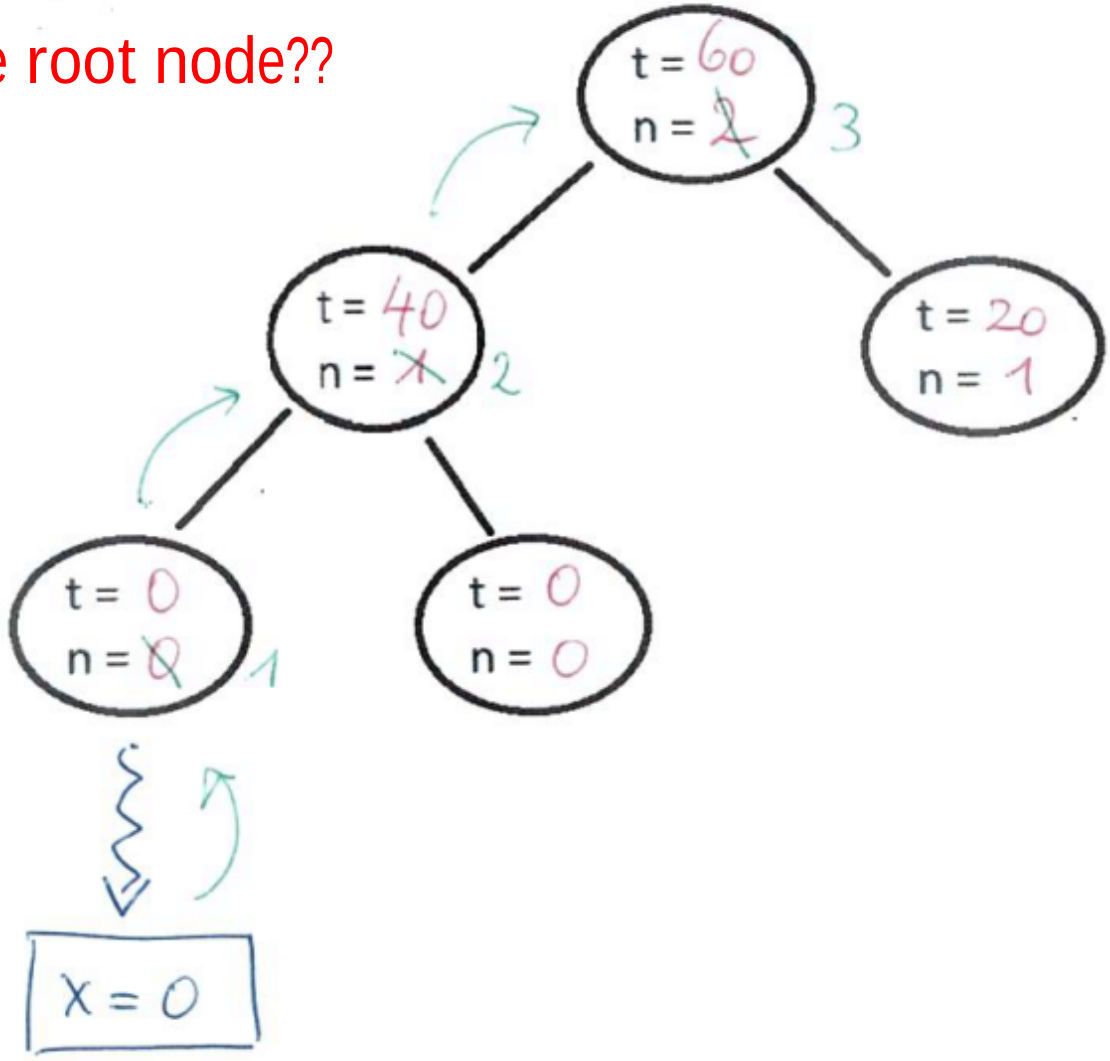


MCTS: Worked Example

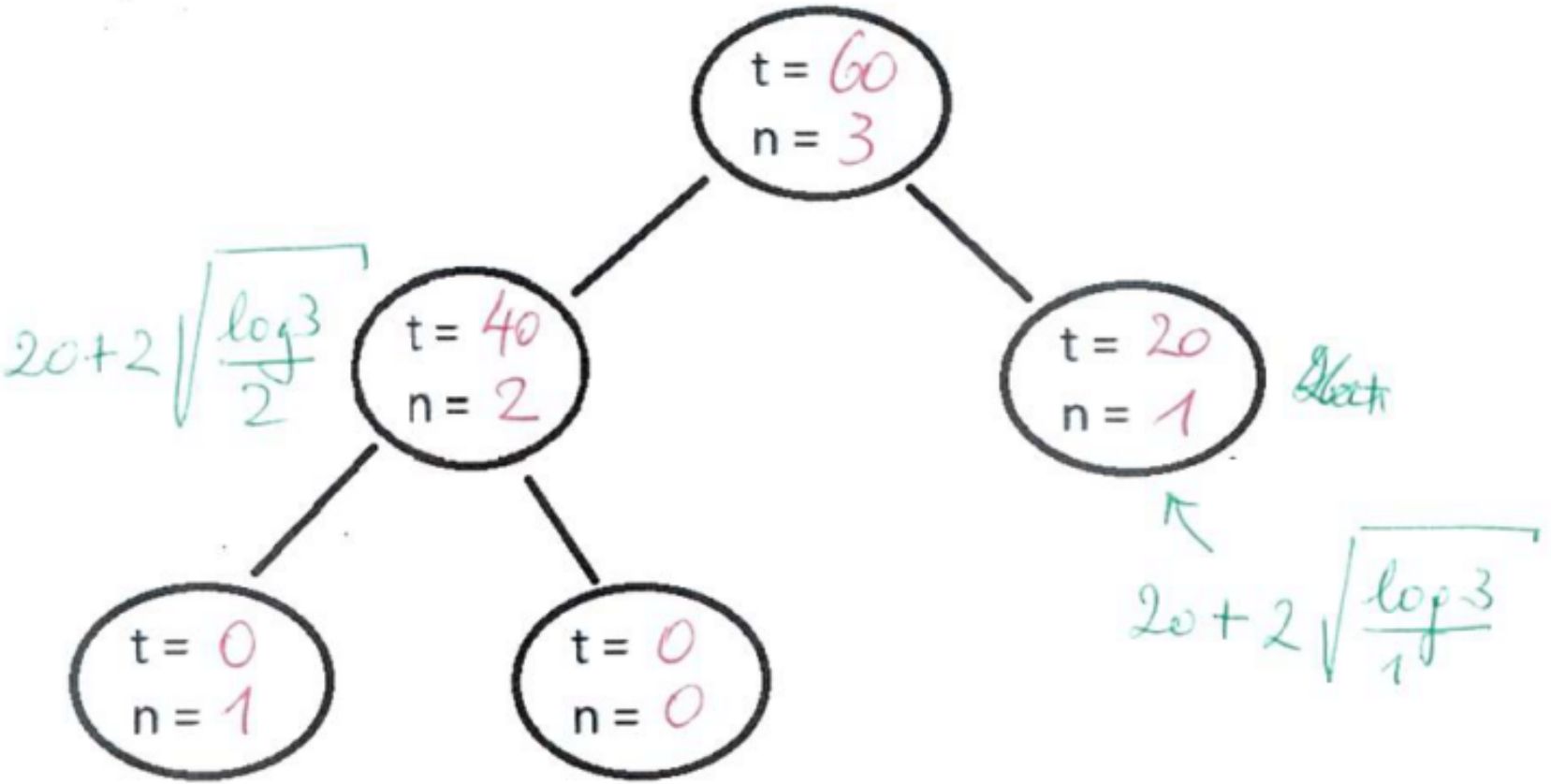


MCTS: Worked Example

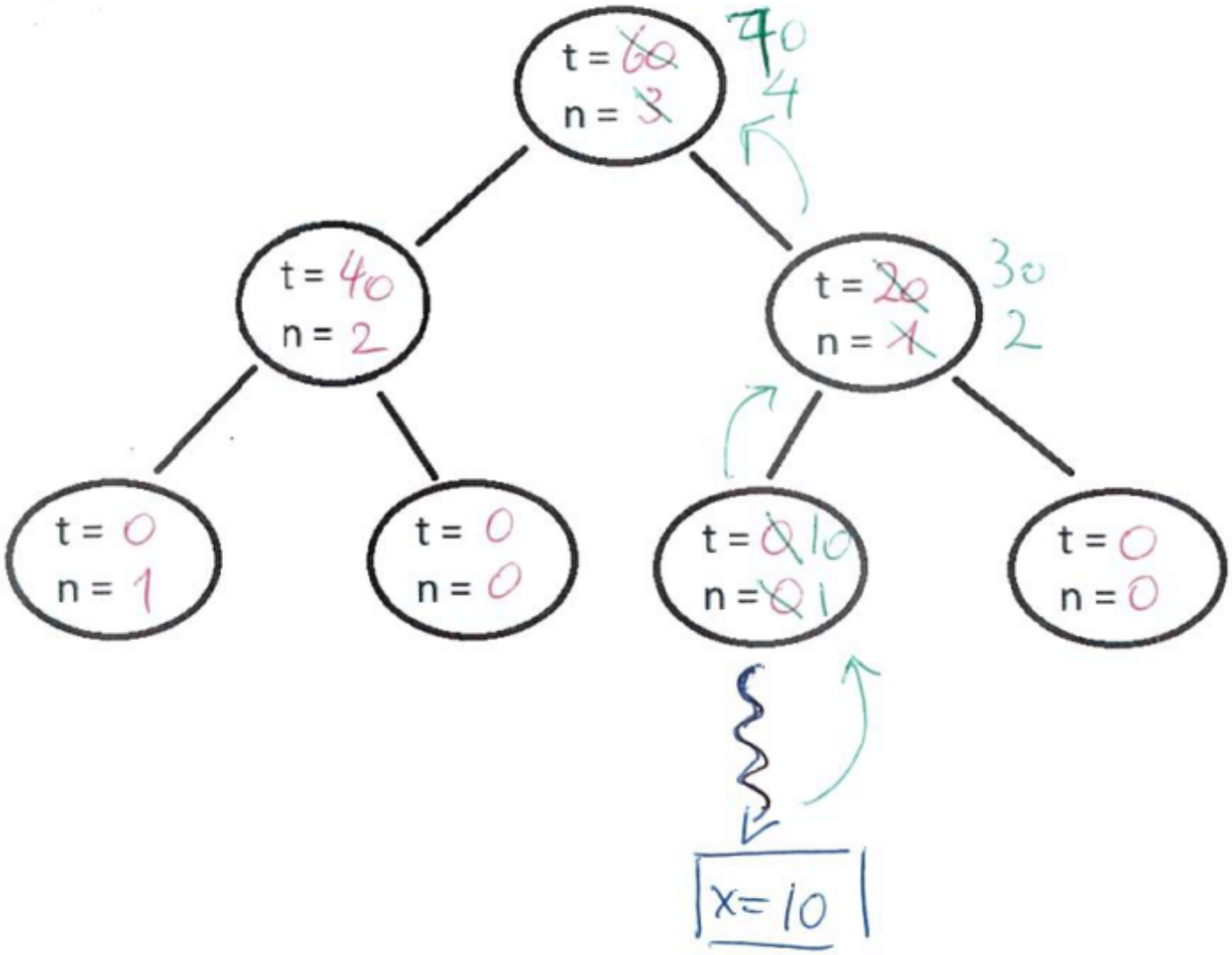
? update till the root node??



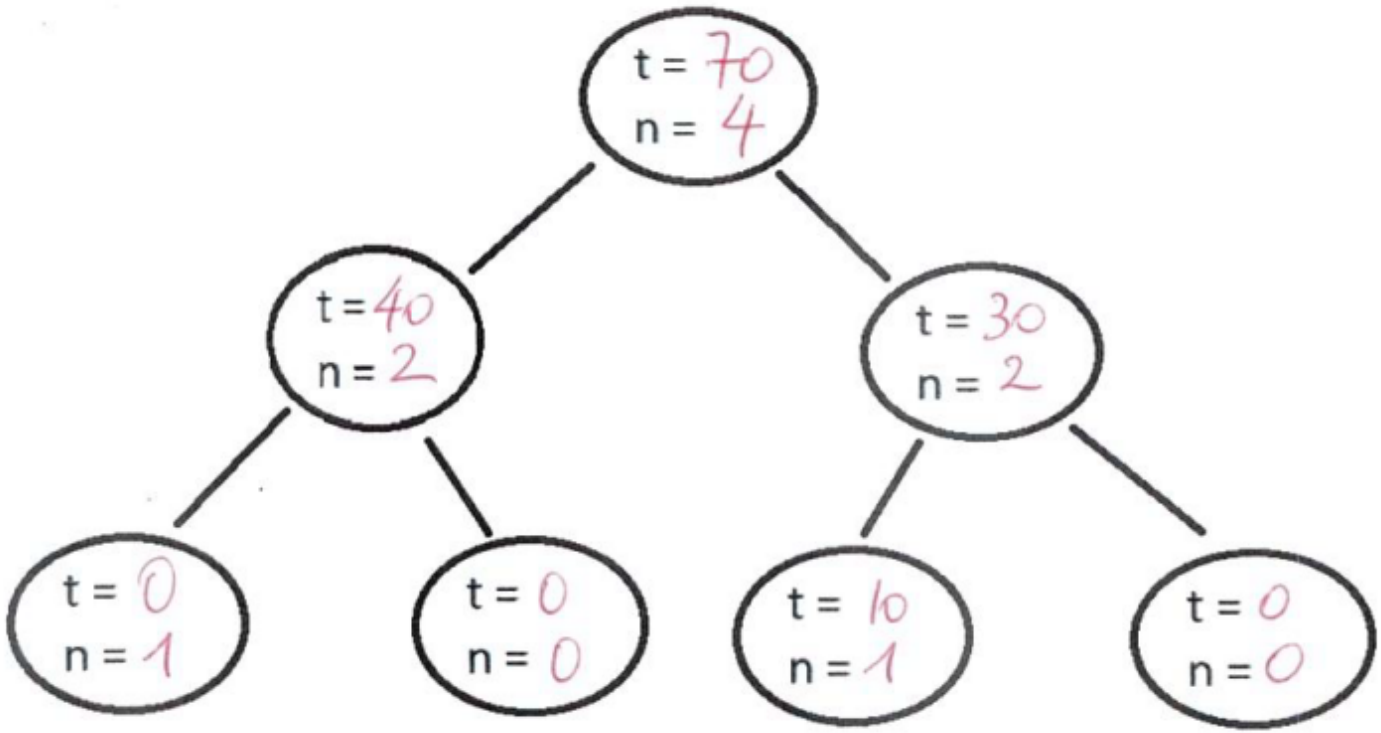
MCTS: Worked Example



MCTS: Worked Example



MCTS: Worked Example

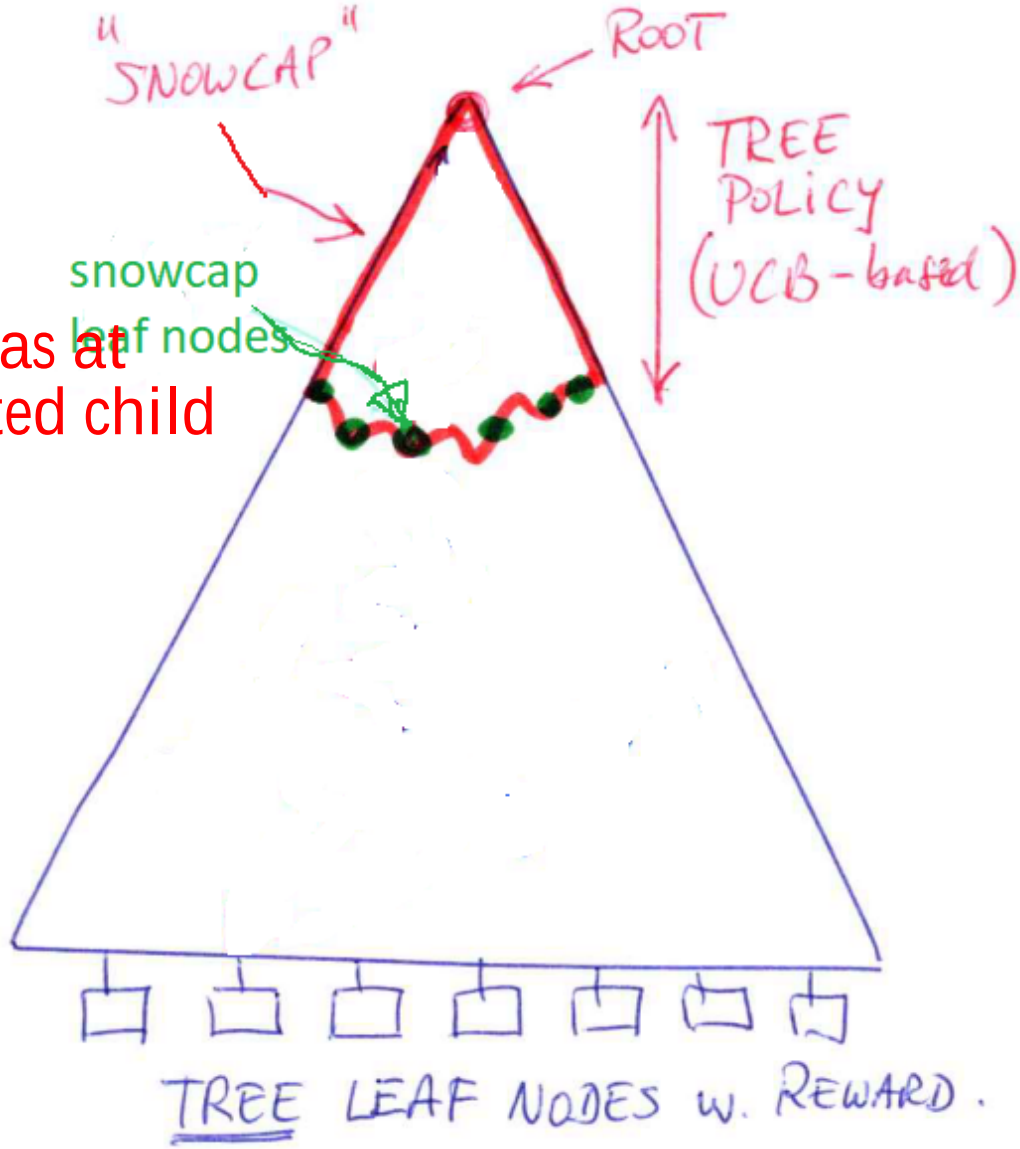


Assuming that we need to decide at this point:
go LEFT (expected value $= 40/20 = 20$ vs. $30/2 = 15$ for RIGHT).

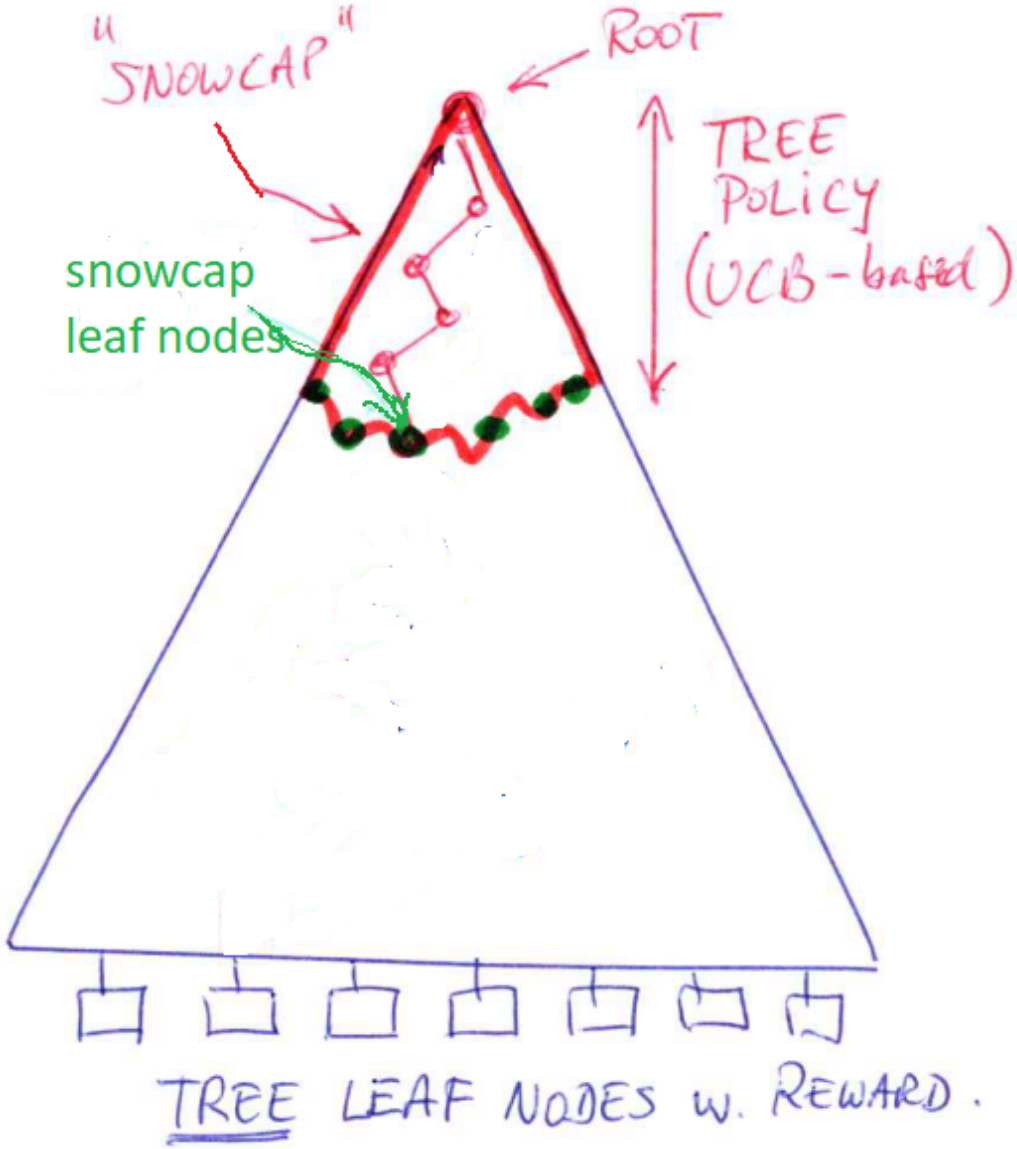
MCTS Summary: Tree and "snowcap" (i.e. subtree of explored nodes)

snowcap node:

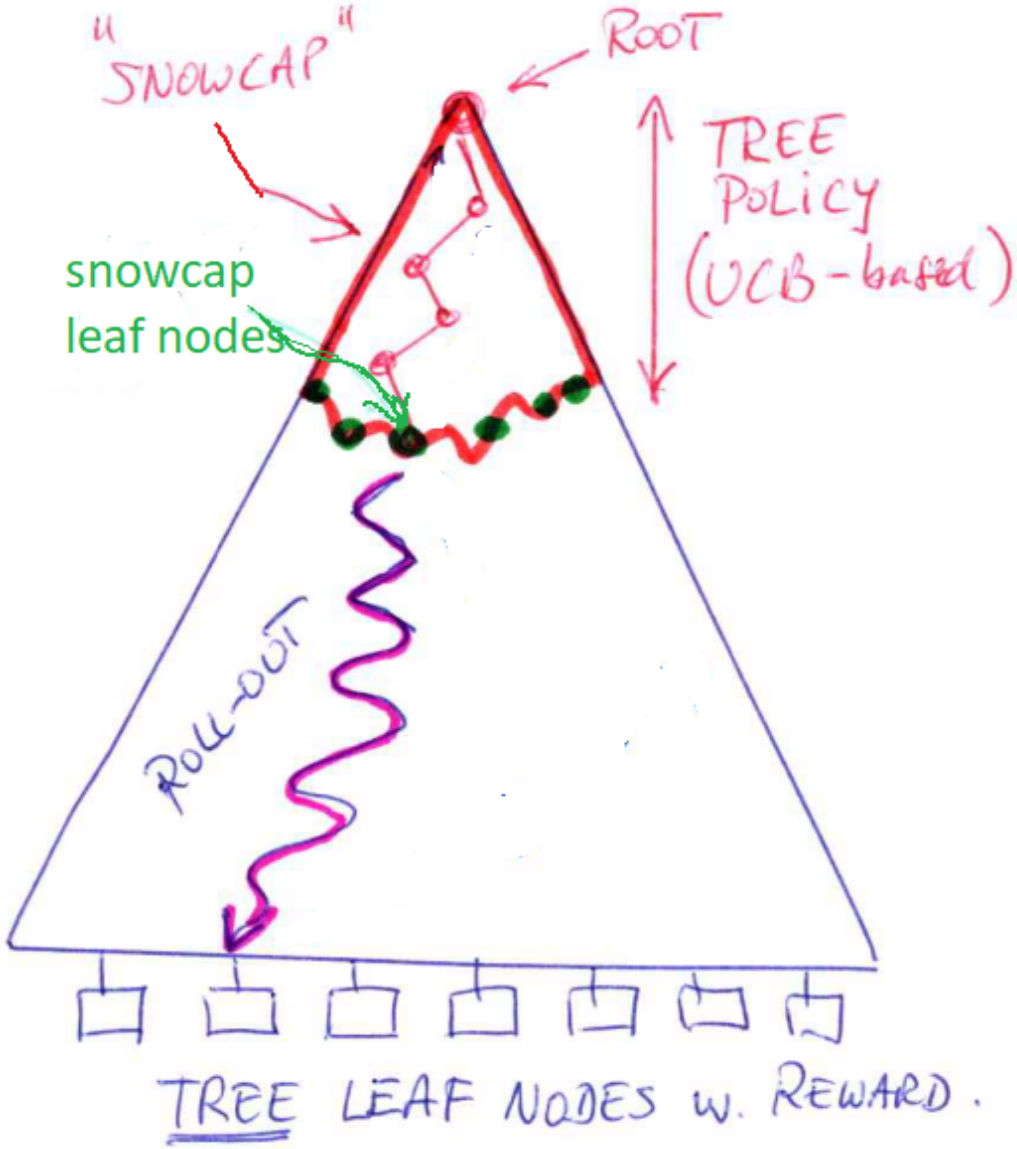
leaf node that has at least one unvisited child node



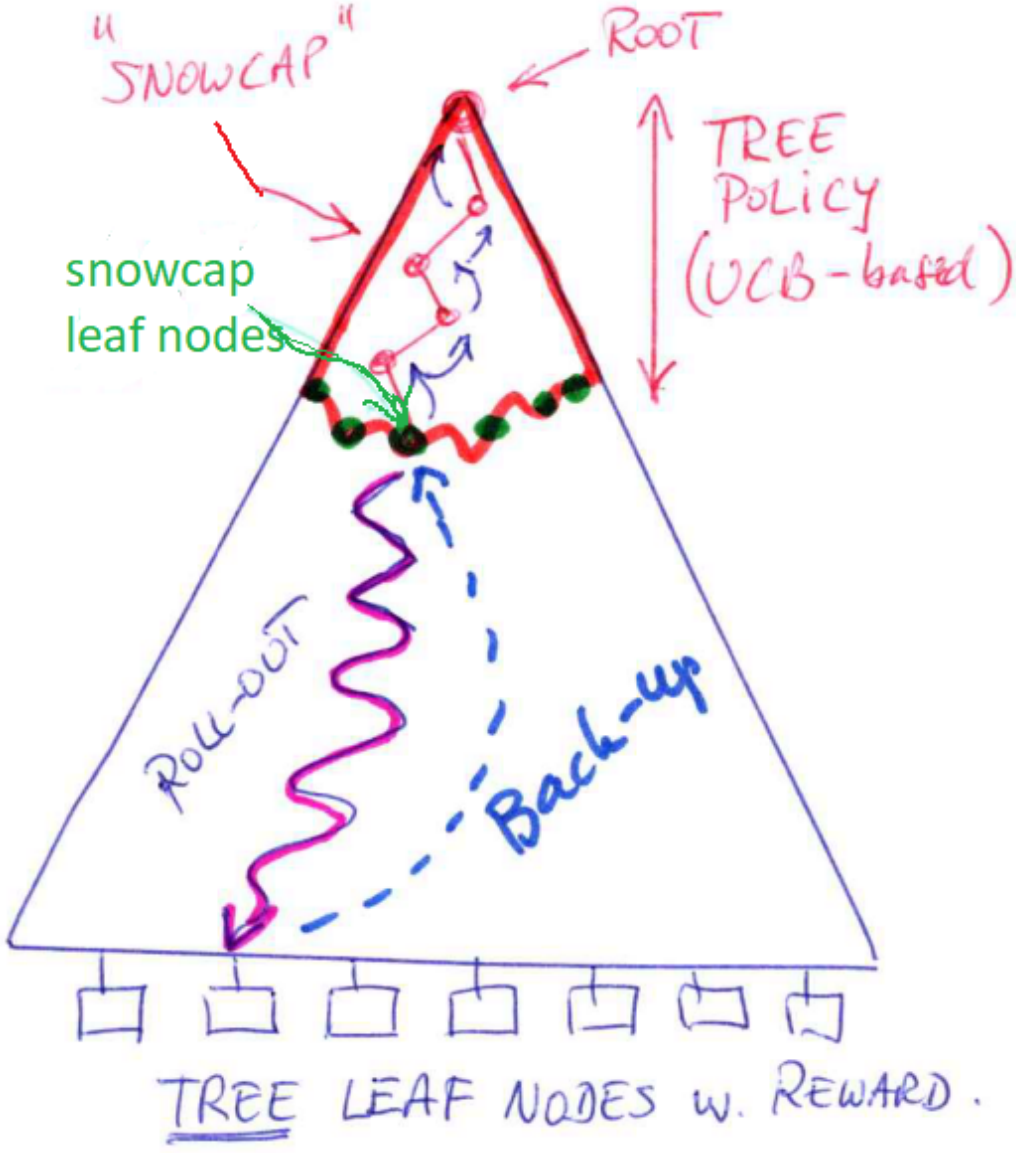
MCTS Summary: Use tree policy to construct path to leaf nodes in "snowcap edge"



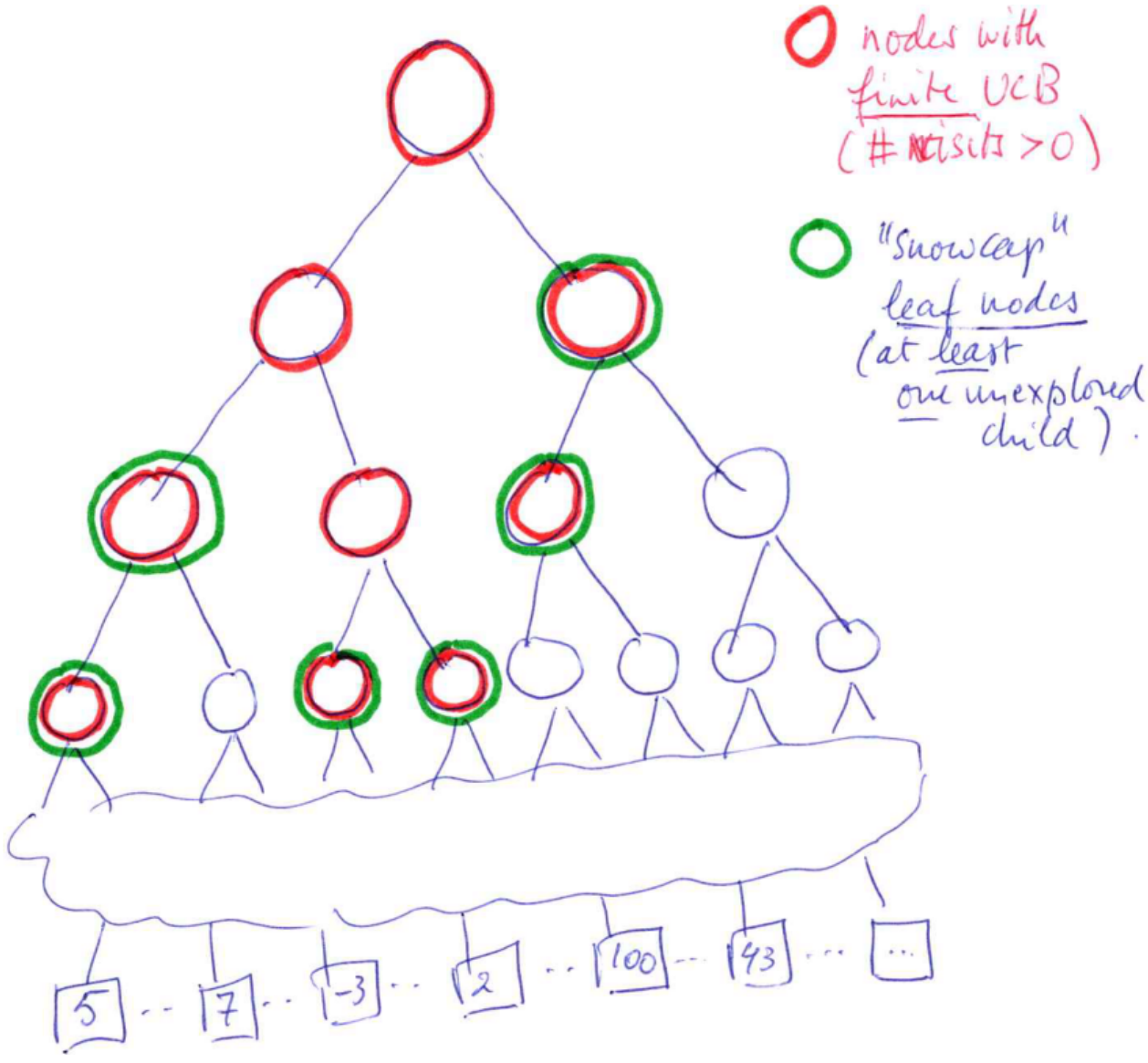
MCTS Summary: Roll-out from "snowcap edge' ' to leaf nodes (reward) in tree



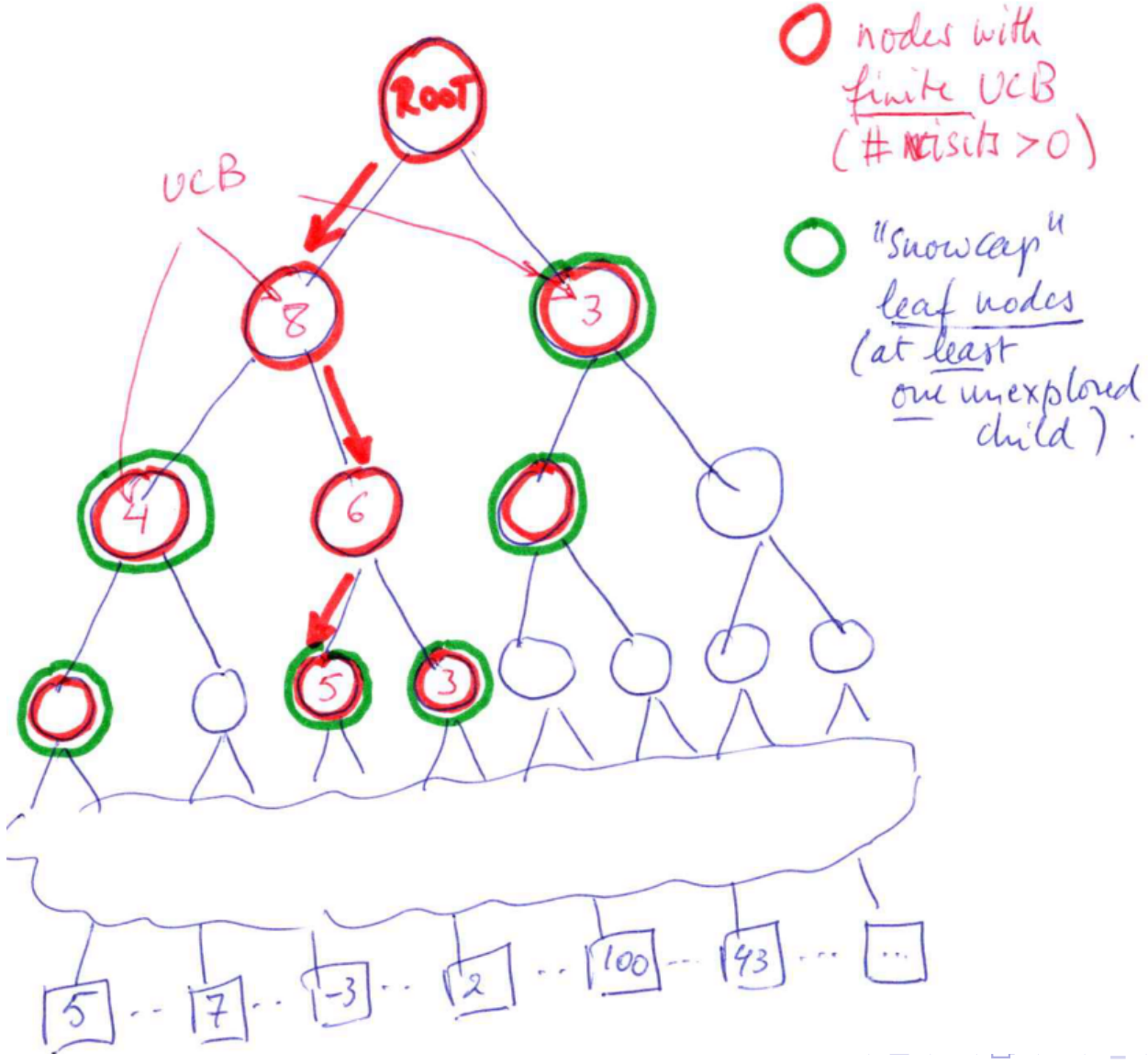
MCTS Summary: Backup values in TP path



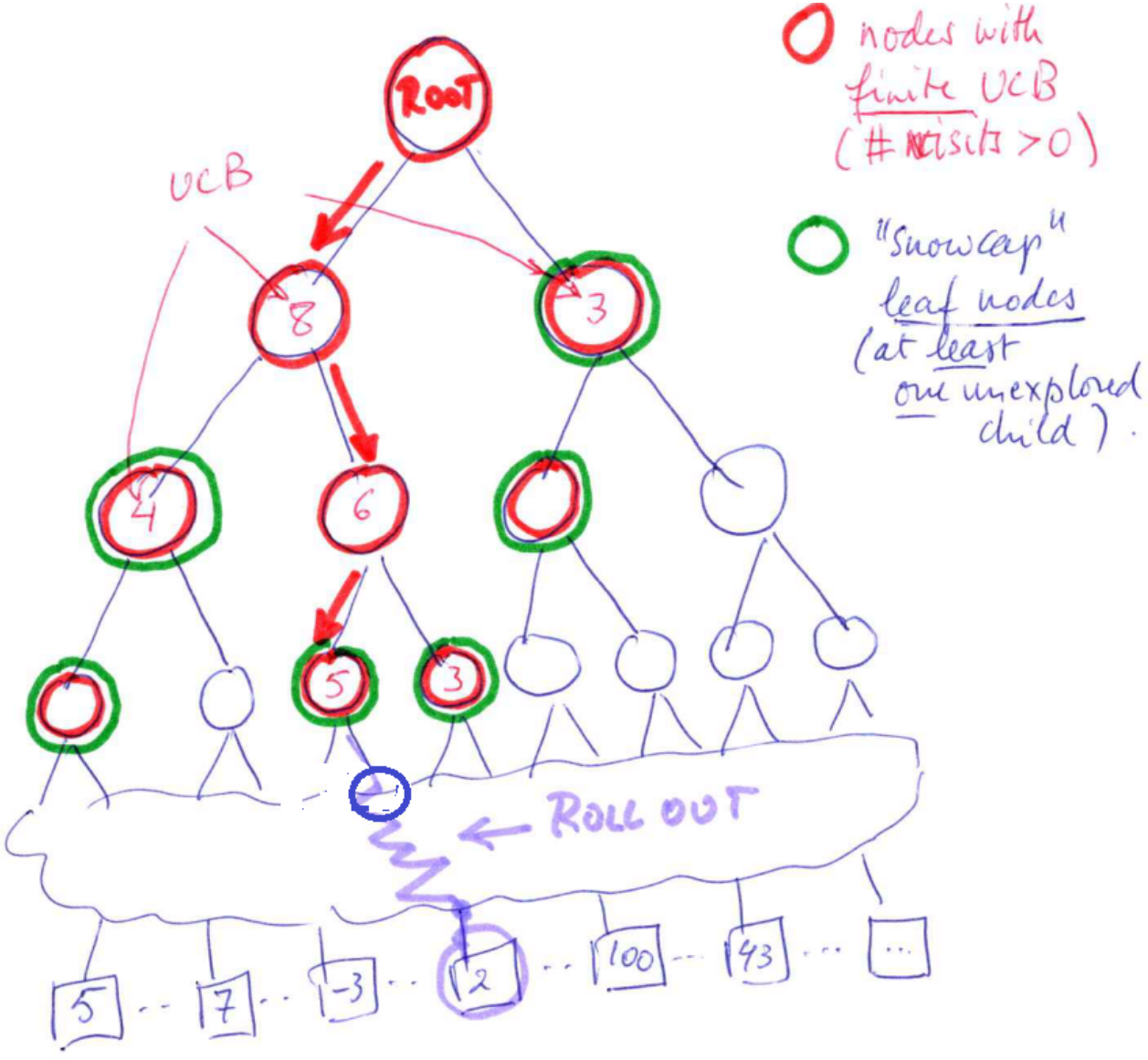
MCTS Summary: Close-up of tree top ("snowcap")



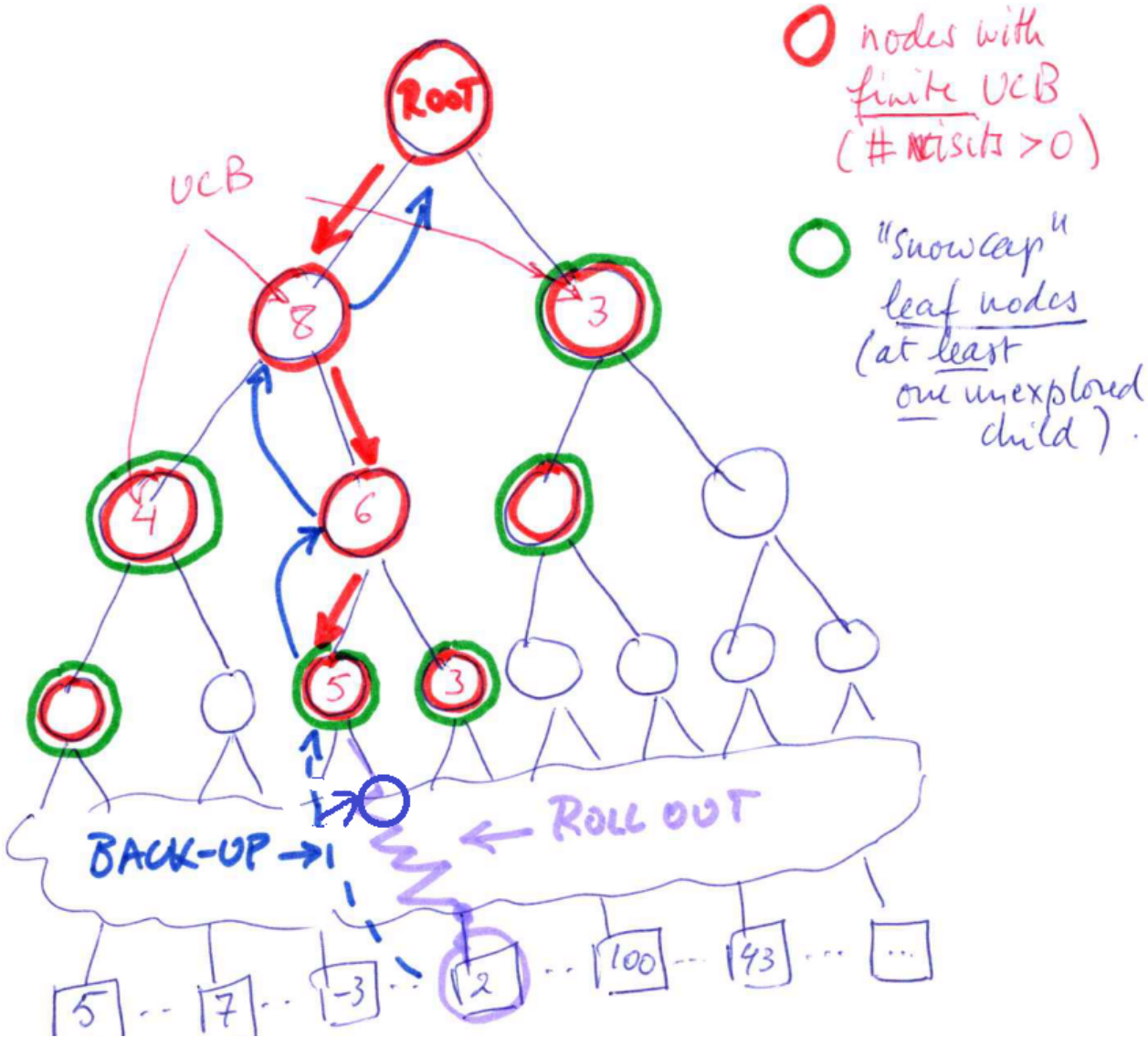
MCTS Summary: Applying tree policy (TP) to find path to "snowcap leaf nodes" (using UCB-values)



MCTS Summary: Roll-out from "snowcap edge"



MCTS Summary: Backup values in TP path



Summary

- **Exploitation:** Be greedy: Choose best action (according to current information)
- **Exploration:** Try something new (use randomisation)
- **Epsilon-greedy:** Simple but effective combination of exploration and exploitation
- **Upper confidence bound (UCB)**:
  - *Optimism in the face of uncertainty!*
  - Asymptotic total regret achieves logarithmic (optimal) bound.
- **Monte Carlo Tree Search (MCTS)**: Use UCB to focus search on most promising part of the tree.