How to measure exploration

Bad idea: by the sound of their name
Good idea: with $$$ it brought/lost you

Regret of policy π(a|s):

Consider an optimal policy, π*(a|s)

Regret per tick = optimal – yours

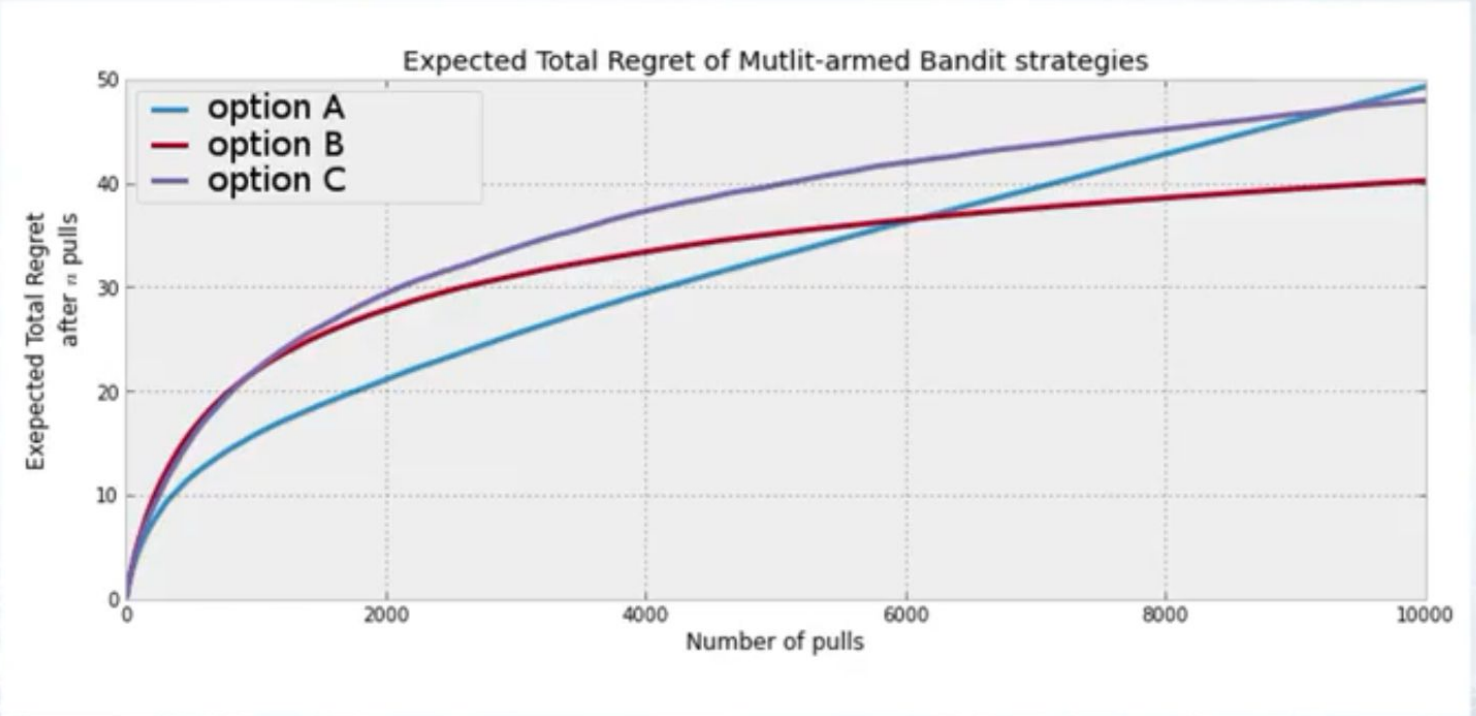
$$\eta = \sum_t \underset{s,a\sim\pi^*}{E} r(s,a) - \underset{s,a\sim\pi_t}{E} r(s,a)$$

Finite horizon: t < max_t Infinite horizon: t → inf

How to measure exploration

Bad idea: by the sound of their name
Good idea: with $$$ it brought/lost you

Regret of policy π(a|s):
Consider an optimal policy, π*(a|s)



What we already know

Strategies:

- **ε-greedy**
  - With probability ε take a uniformly random action;
  - Otherwise take optimal action.
- **Boltzman**
  - Pick action proportionally to transformed Qvalues

$$\pi(a|s) = softmax\left(Q(s,a)/\tau\right)$$

- Policy based: add entropy

Quiz time

Your agent uses ε-greedy strategy with **ε=0.1**

What about his regret?

$$\eta = \sum_t \mathop{E}_{s,a\sim\pi^*} r(s,a) - \mathop{E}_{s,a\sim\pi_t} r(s,a)$$

Stacking up diff

It grows linearly with time!

We never reach optimal policy cuz of **ε**

Q: How do we fix it?

Greedy in the limit

If exploration is asymptotically zero, $\lim_{t \to inf} \epsilon = 0$

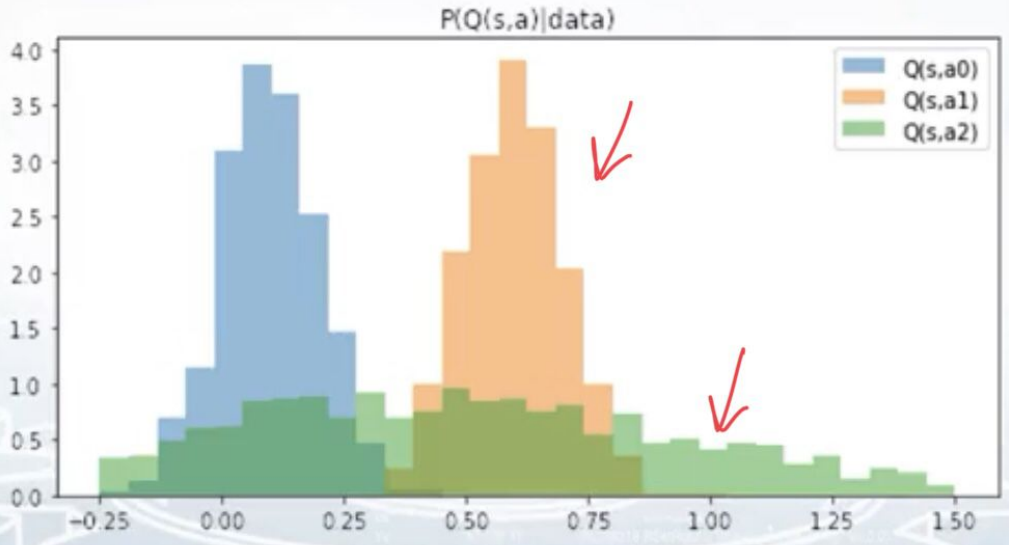
we will reach π* in the limit

Practical: multiply **ε** by 0.99 every K steps :)

Uncertainty in rewards

We want to try actions if we believe there's a chance they turn out optimal.

Idea: let's model how certain we are that Q(s,a) is what we predicted



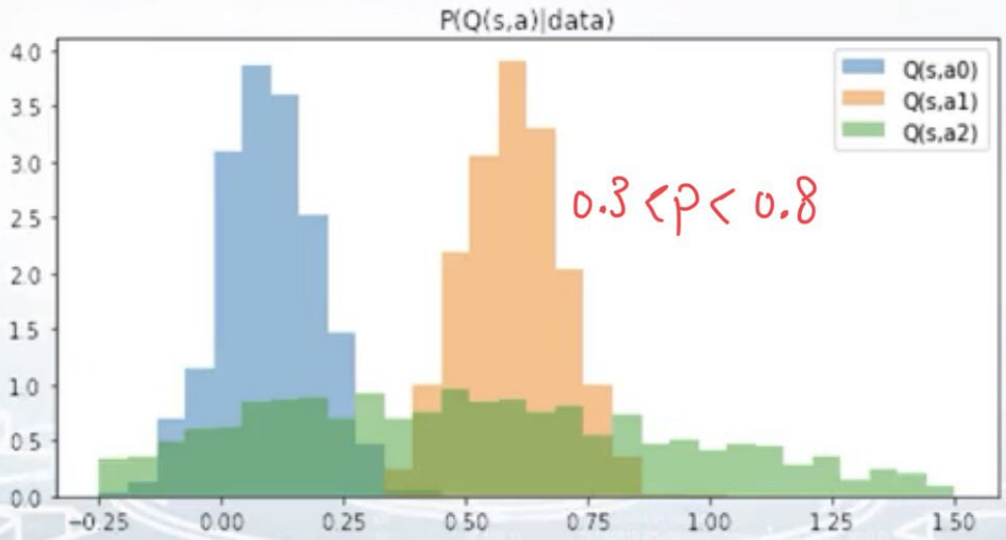
no point of choosing blue!

Our beliefs, not randomness in action

Thompson Sampling

- Policy:
  - sample **once** from each Q distribution
  - take argmax over samples
  - which actions will be taken?

Takes a1 with p ~ 0.65, a2 with p ~ 0.35, a0 ~ never



$P_{blue} = 0$

there will always exist some orange sample with higher prob

Optimism in face of uncertainty

faster exploration

Idea:

Prioritize actions with uncertain outcomes!

More uncertain = better.

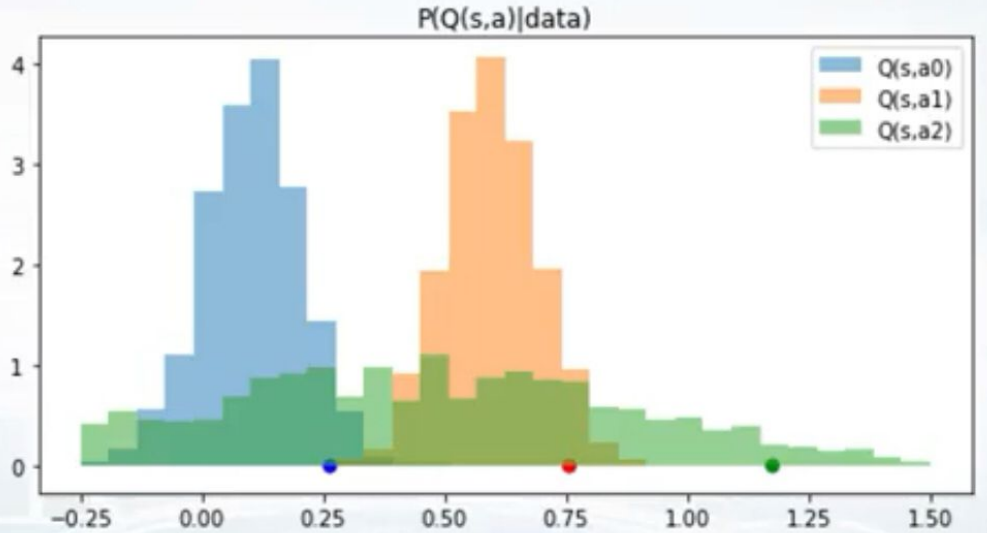
Greater expected value = better

Math: try until upper confidence bound is small enough.

Optimism in face of uncertainty

- Policy:
  - Compute 95% upper confidence bound *for each a*
  - Take action with highest confidence bound
  - What can we tune here to explore more/less?

to be adjusted



Frequentist approach

no need to know full distr,

only percentile would be enough

There's a number of inequalities that bound

P(x>t) < something

- E.g. Hoeffding inequality (arbitrary x in [0,1])

$$P\left(x - Ex \geq t\right) = e^{-2nt^2}$$

t: our interest "threshold"

n: # samples; q values

- Remember any others?

(Chernoff, Chebyshev, over9000)

UCB-1

One problem is that it might overestimate the value of an action

UCB-1 for bandits

Take actions in in proportion to

$$\tilde{v}_a = v_a + \sqrt{\frac{2 \log N}{n_a}}$$

Rule of large no applies

- N number of time-steps so far
- n_a times action **a** is taken

$v_a \rightarrow \tilde{v}_a$

Bayesian UCB

The usual way:

- Start from prior P(Q)
- Learn posterior P(Q|data)
- Take q-th percentile

Approach 1: learn parametric P(Q), *e.g. normal*

Approach 2: use bayesian neural networks

more powerful

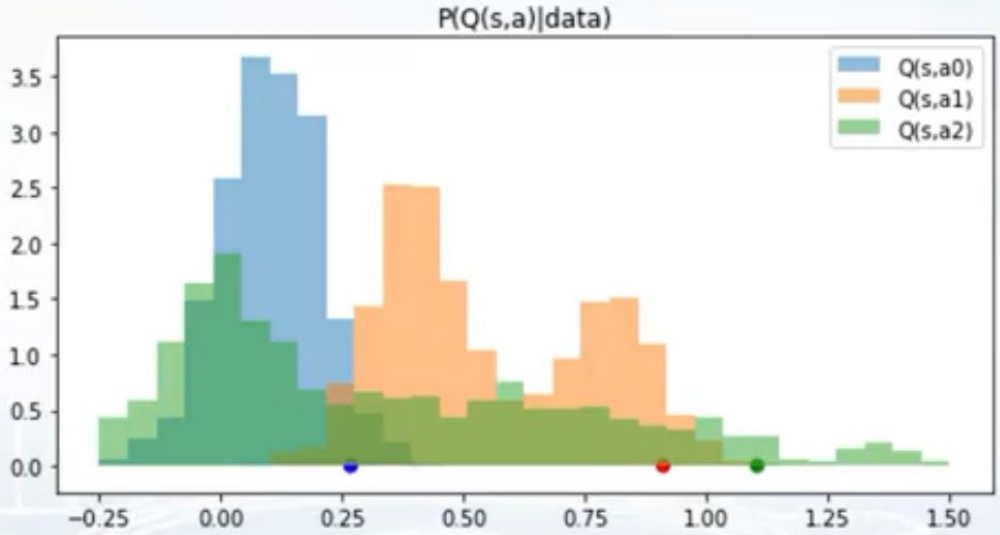
distributions instead of single values

i.e no restrictions on distr choice

Bayesian UCB

no more inequalities!

- UCB vs Bayesian UCB
  - Choose any distribution you want
  - Learn actual distribution, not upper bound
  - Only as good as your prior



Again, 3 points on horizontal axis are percentiles

Comparing regret

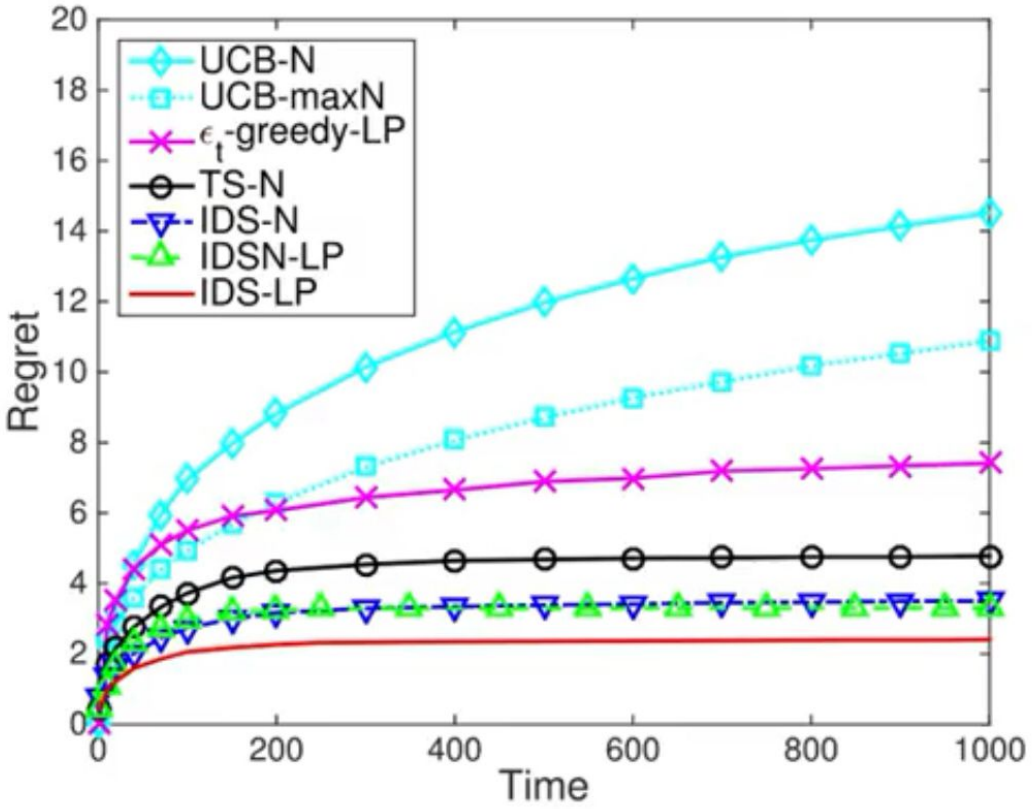


Figure: groundai.com, http://bit.ly/2FUmM5G

TL;DR this block

- Regret: “money you could have saved”
  - if you knew optimal actions from the get go
  - Less regret = better exploration

- ε-greedy and boltzmann never behave optimally
  - Make them greedy in the limit

- Using uncertainty
  - Thompson Sampling, UCB-1, Bayesian UCB

- Next: using exploration in model-based RL

Exploration reduces uncertainty

Model-free setting

- Unknown environment dynamics $p(s', r|s, a)$

Model-based setting

- Known model of the world
  - Distribution model: $p(s', r|s, a)$ explicit probs
  - Sample model: $(s', r) \backsim G(s, a)$

Do we need exploration for model-based learning?

Still don't know the global values of any action!

only immediate rewards are known,

which is still not enough

We don't know global value estimates

Given the model of the world, what is the best action?

1. Immediate rewards are **insufficient**
2. **Time constraints** on recovering sum of future rewards

Planning

Transform

- known immediate rewards & dynamics

into

- value / action-value estimates / explicit policy

Types of planning

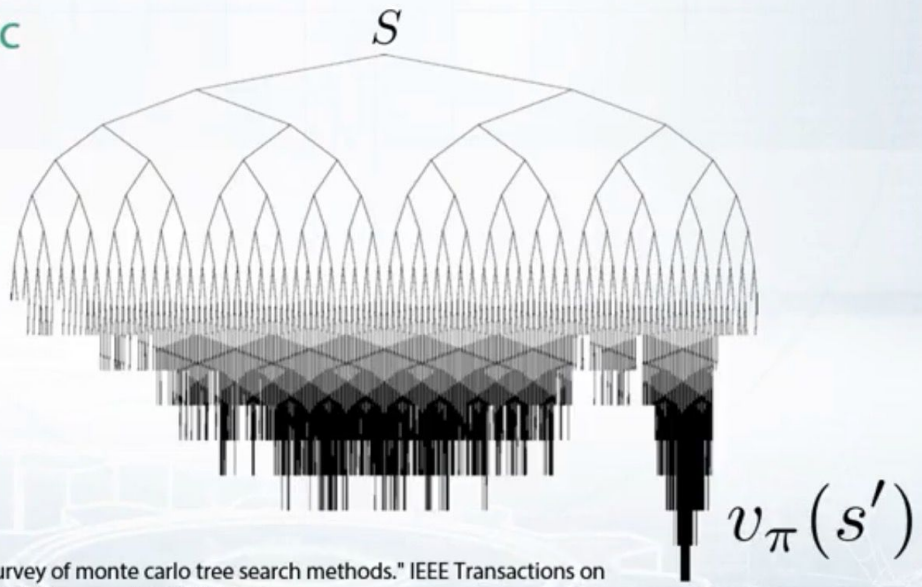
Model-free – **learn** the policy by trial and error

Model-based – **plan** to obtain the policy

- Background planning (DP)
  - learning from simulated experience
  - improves estimates in arbitrary states
- Decision-time planning (heuristic search, MCTS)
  - focuses on current state
  - plans action selection for current state only
  - commit an action only after planning is finished

Guided planning – heuristic search

1. Exhaustive
2. Full-width with fixed depth
3. Full-width with fixed depth, approximate at leaves
4. Heuristic



Browne, Cameron B., et al. "A survey of monte carlo tree search methods." IEEE Transactions on Computational Intelligence and AI in games 4.1 (2012): 1-43.

Guided planning – heuristic search

Heuristic search – an umbrella term for many algorithms

- lookahead planning guided by heuristic function
  - select only "valuable" nodes to expand

Heuristic search for RL

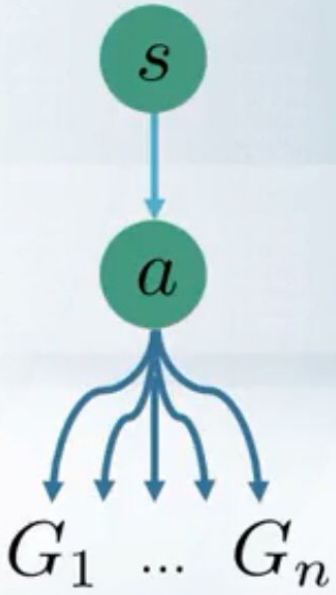
Heuristic is value fn

+ resources are focused only on valuable paths
+ nearest states contribute the most to the return
- bad planning results when the model is unreliable
- dependence on the quality of heuristic

* Only perform planning, when the model of the world is more precise than current value fn estimates

Obtain value estimates with rollouts

1. While within computational budget:
   a. From state s make action a
   b. Follow rollout policy until episode terminates
   c. Obtain a return G_i
2. Output $\hat{q}(s,a) = \frac{1}{n}\sum_{i=1}^{n} G_i$



+ No function approximation
+ Trials are independent → parallelization
– Dependence on # of returns averaged

time vs accuracy

Using rollouts for action selection

Greedy policy – policy improvement as in Value Iteration

$$\pi(s) \leftarrow \operatorname{argmax}_a \hat{q}(s, a)$$

Idea

- No estimation of q_*
- No further policy improvement

} Good rollout policy is needed!

It shall be only one policy improvement step behind the optimal policy

Using rollouts for action selection

Greedy policy – policy improvement as in Value Iteration

$$\pi(s) \leftarrow \operatorname{argmax}_a \hat{q}(s, a)$$

- No estimation of q_*
- No further policy improvement

} Good rollout policy is needed!

- Good rollout policy may require much of valuable time
  - May result in less reliable estimates of $\hat{q}(s, a)$

if it's slow, we run out of time

  - Dependence on quality of rollout policy
  + Quite good results even for **random** rollout policy
  - No estimates preserved between successive states
  - Does not respect uncertainty in action-value estimates

Recap: discussed so far

- Planning: model in, policy out
- Crucial constraint in planning: time
- Accurate planning: guided by heuristic
- Variant of heuristic: MC estimate
- Simplest policy: greedy w.r.t. to heuristic

Not yet covered

- Preserving estimates between successive time steps
- Dealing explicitly with uncertainty in estimates
- Guiding search with heuristic

Monte Carlo Tree Search at a glance

- Amazingly successful decision-time algorithms
- Distant relatives of heuristic search
- Usually require much of computation time
- Estimate action-values with MC rollouts
- Preserve value estimates between transitions
- Maintain two policies
  - tree policy – respects uncertainty, guides the search
  - rollout policy – estimates the return

Scheme of Monte Carlo Tree Search

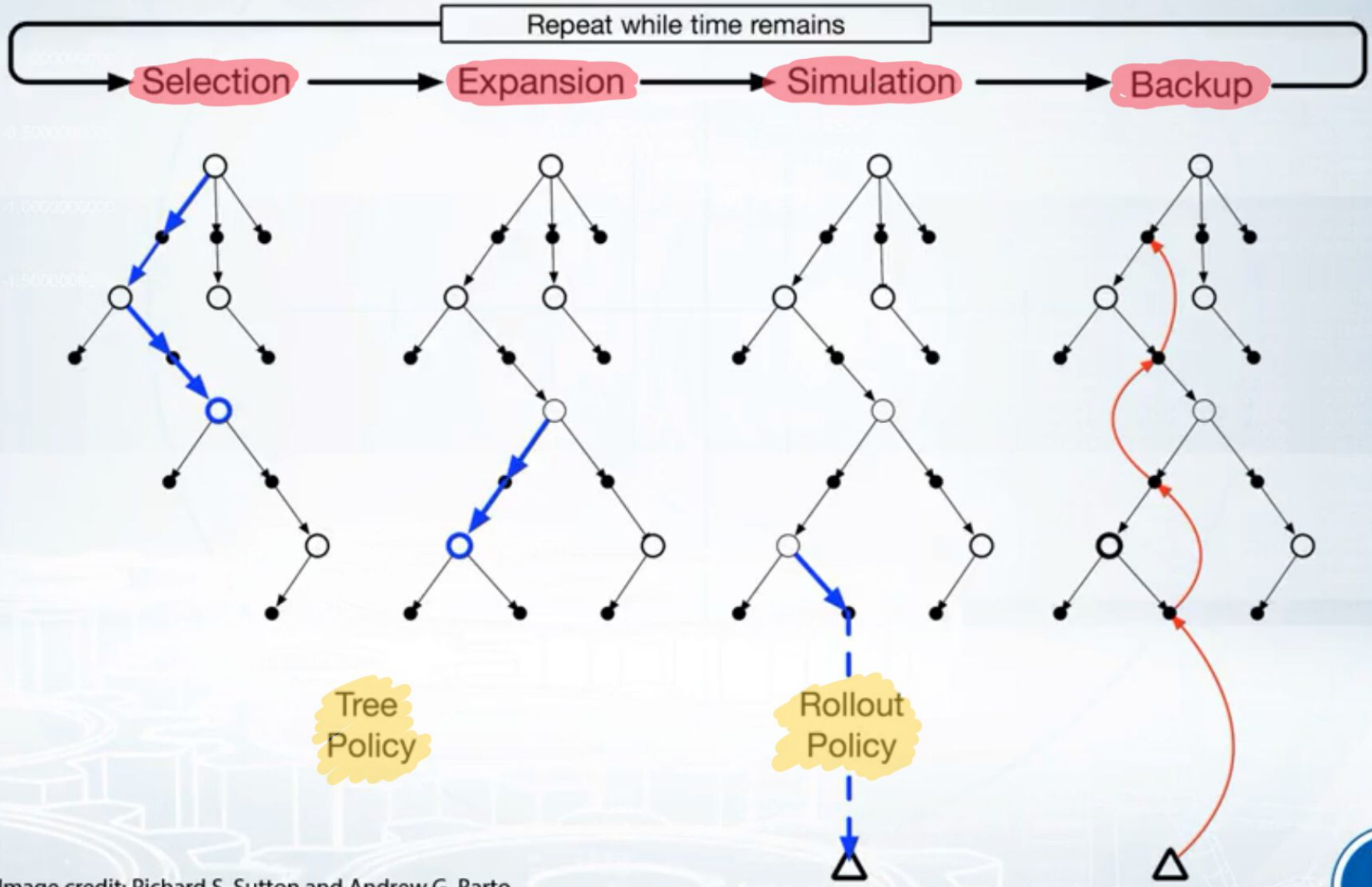


Image credit: Richard S. Sutton and Andrew G. Barto
"Reinforcement Learning: An Introduction", Draft of second edition (2017)

Tree policy: Upper Confidence Bounds for Trees

- Exploitation – actions with existing good estimates
- Exploration – rarely tested actions

* Treating action selection as multiarmed bandit (UCB1)

$$\pi_{tree}(s) = \text{argmax}_a \; \hat{q}(s,a) + 2C\sqrt{\frac{2\ln N(s)}{N(s,a)}}$$

Sim prev visited s

Sim made a in s

Tree policy: Upper Confidence Bounds for Trees

- Exploitation – actions with existing good estimates
- Exploration – rarely tested actions

Treating action selection as multiarmed bandit (UCB1)

Exploitation

$$\pi_{tree}(s) = \text{argmax}_a \; \hat{q}(s,a) + 2C\sqrt{\frac{2\ln N(s)}{N(s,a)}}$$

Exploration

- Good theoretical properties
- Constant C tunes the amount of exploration
- If $N(s,a) = 0$, then action is always selected

ensuring each action will be chosen

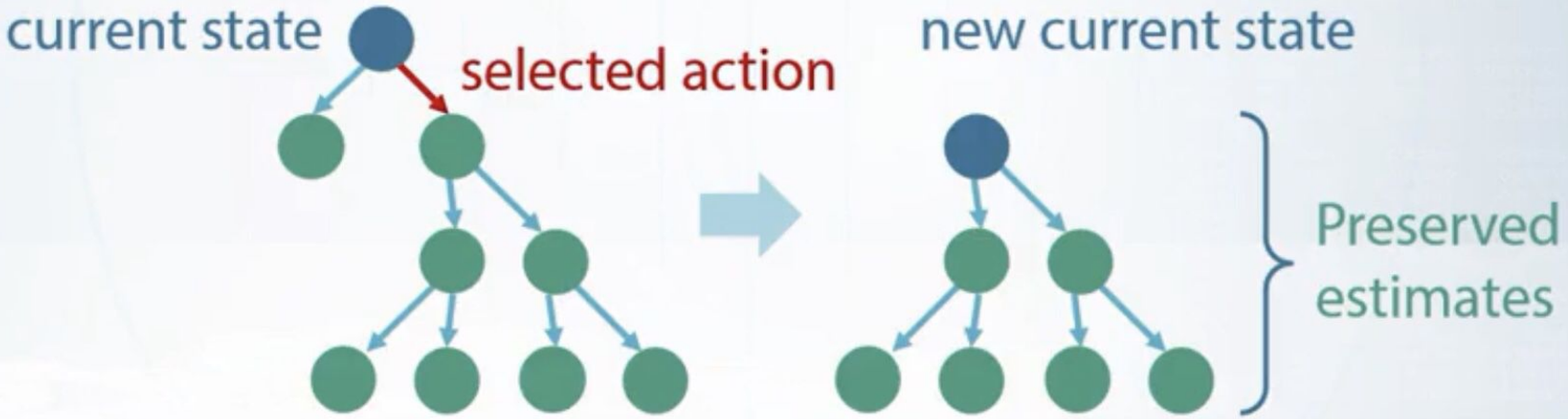
Action selection

1. Max – most valuable root child (action with highest $\hat{q}(s, a)$)
   - Simple and effective
2. Robust – most visited root child (action with highest $N(s, a)$)
   - Effective against outliers, may be suboptimal
3. Max-robust – both highest visit count & value
   - May require more planning time
4. Secure child – maximizes a lower confidence bound
   - Paranoid mode, real life applicable

Sacrificing some rewards ; to obtain safe trajectories

MCTS: benefits and drawbacks

+ Context independent – no hand designed heuristic
+ Asymmetric search – more promising directions first
+ Anytime – has the answer if stopped at any time
+ Saves a lot of computations



+ Simple in implementation
- Dependence on quality of rollout policy
- Computationally expensive