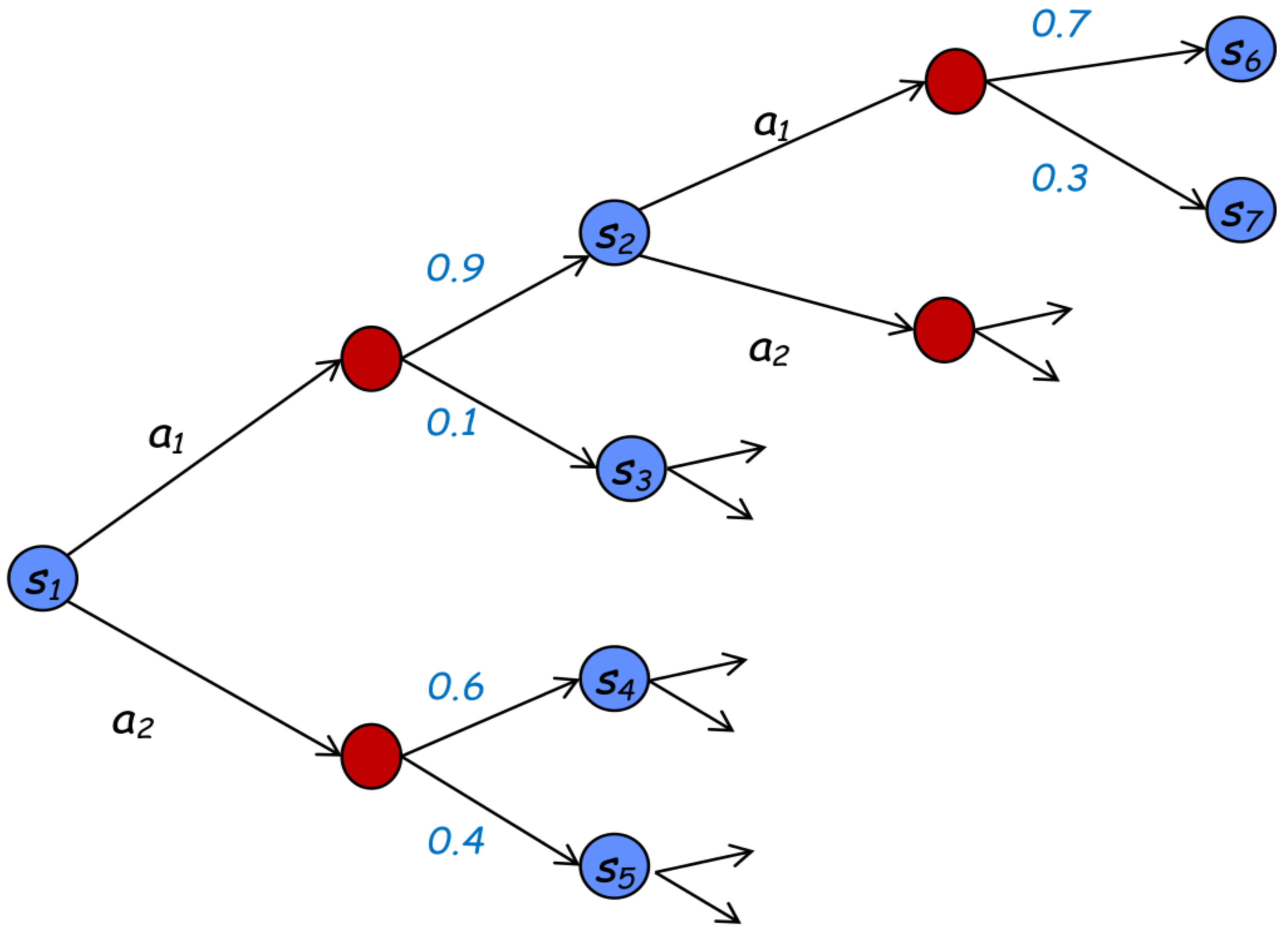
MCTS and UCT

Tutorial 12, PUI 2021

Jan Mrkos

MDP



Offline algorithms

- Policy iteration
- Value iteration
- …

Online algorithms

- Replanning
- MCTS

MDP model

Problem model/definition

(S, T, R, A)

Solution

Explicitly **use** distributions and reward functions

Probability calculations to find policy

- VI
- PI

Sample distributions and reward functions to get outcomes of actions

Approximate best actions based on averages

- MCTS
- Robust Replan

Solution evaluation

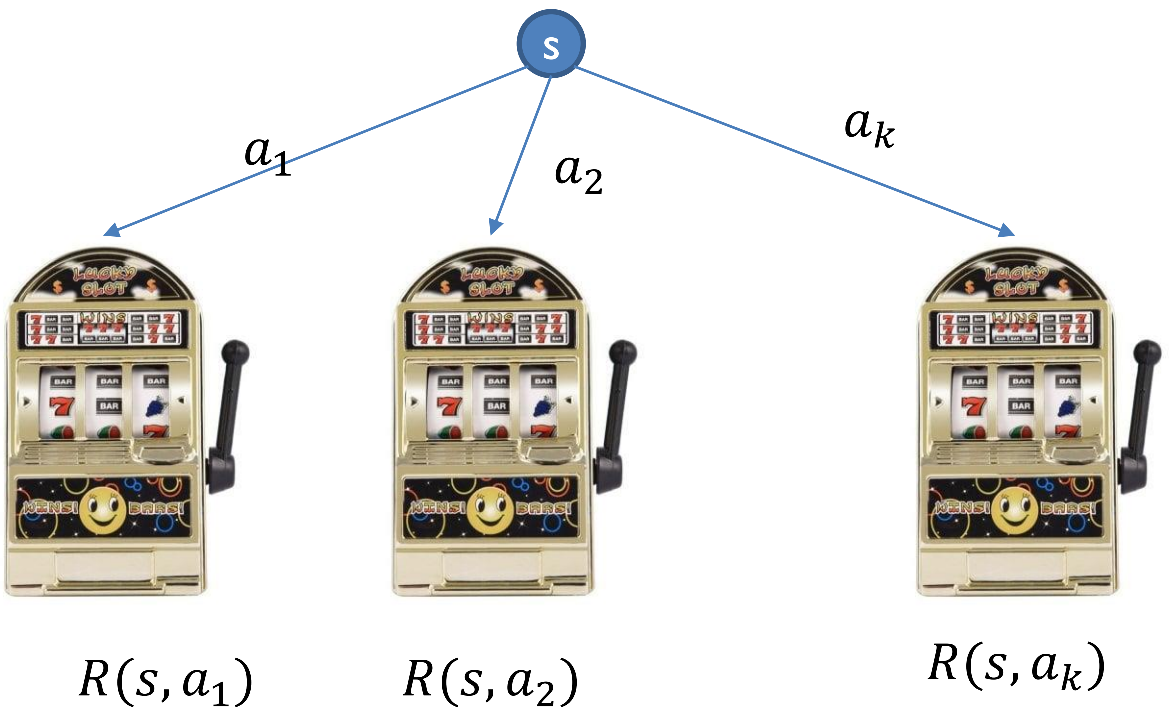
Evaluate in simulator
(that can sample distributions)

Monte-Carlo and MDPs

- Exact state space description not available in large state spaces, but there exist simulators:
  - Traffic simulations
  - Robotics simulators
  - Go
- Monte-Carlo in MDPs
  - Use simulator to evaluate stochastically selected actions
  - Finite (but large) state set S
  - Finite action set A
  - Stochastic, real-valued, bounded reward function R(s, a)=r
  - Stochastic transition function T(s,a)=s'

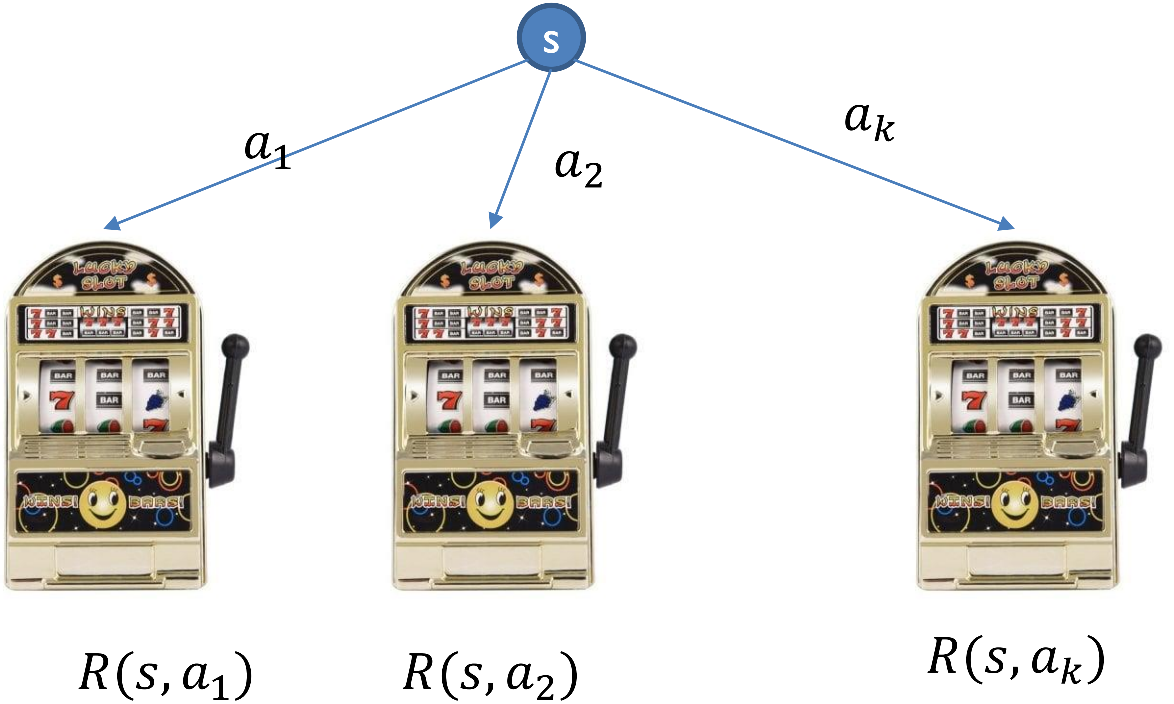
Planning in single state

- Multi-Armed Bandit Problem
  - Which action will yield best expected reward?
  - Simulator returns reward R(s, a)



UCB Adaptive Bandit Algortihm

- **Task:** find arm-pulling strategy such that the expected total reward at time n is close to the best possible.
  - Uniform Bandit – bad choice, wastes time with bad arms
  - Need to balance exploitation of good arms with exploration of poorly understood arms.



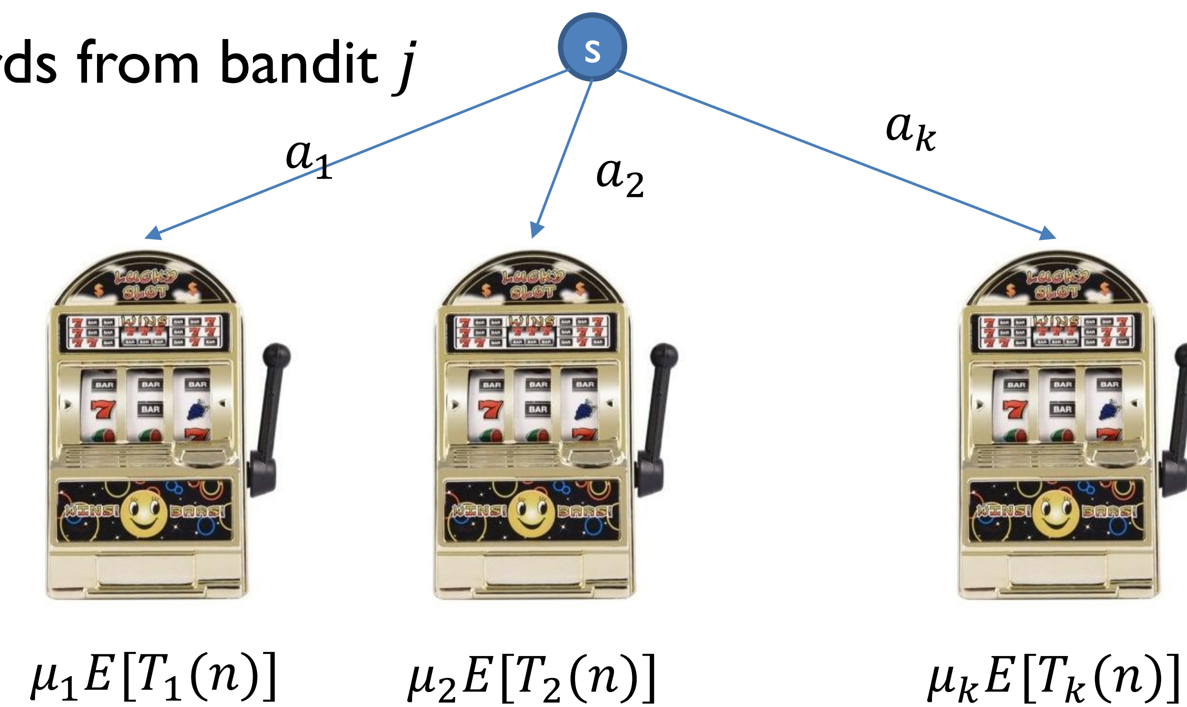
Regret

- Aiming at "reward as close as possible to the best reward" means we are minimizing **regret:**

$$R_n = \mu^* n - \sum_{j=1}^{k} \mu_j E[T_j(n)]$$

Where μ_j are the expected payoffs of arms, μ^* is the best expected payoff and $E[T_j(n)]$ is the expected number of pulls on arm j in total n pulls.

- $X_{j,1}, X_{j,2}$ … = i.i.d r.v. of rewards from bandit j
- μ_j= expected value of X_j



Minimizing regret - UCB

- Exploration may increase regret
- Upper Confidence Bounds [Auer er. al., 2002]: (for rewards in (0,1))

$$UCB = \bar{X}_j + \sqrt{\frac{2 \ln n}{n_j}}$$

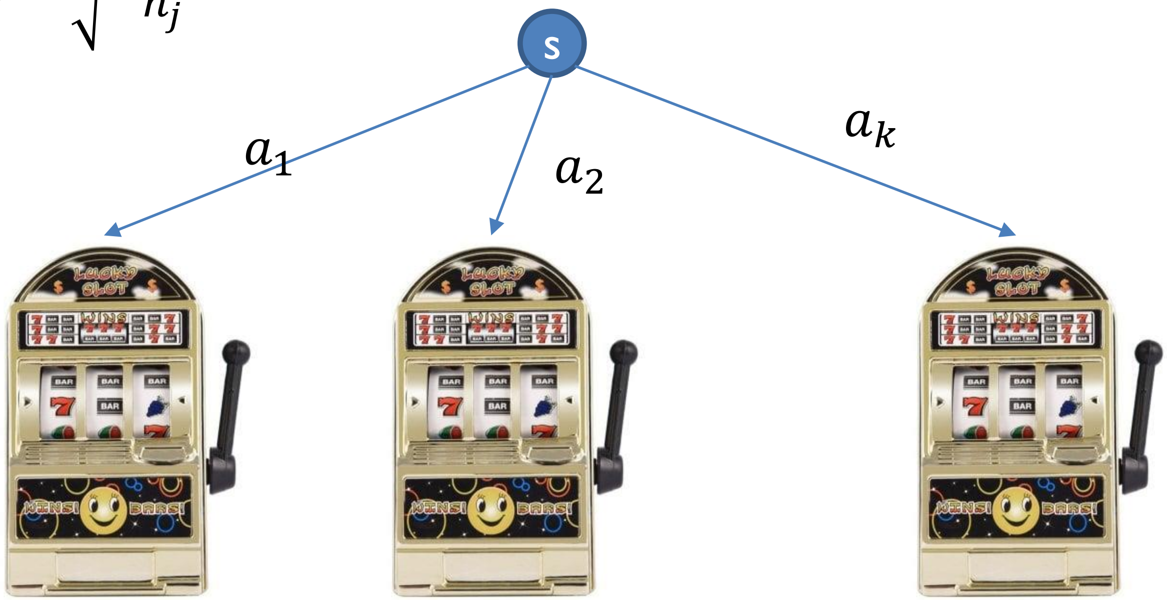
Exploitation

Exploration

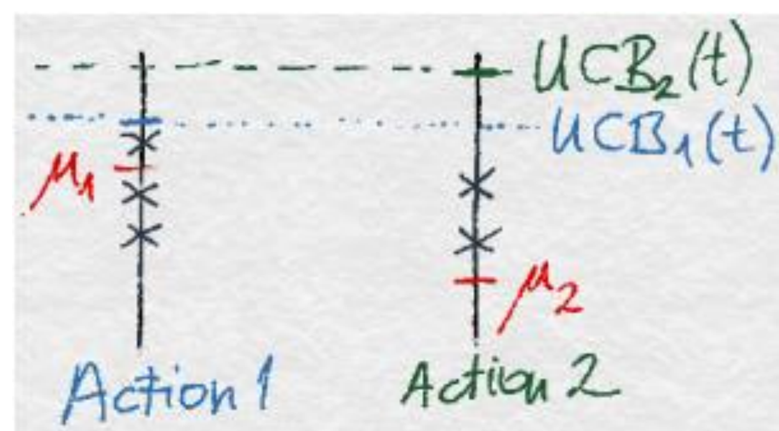
- When choosing arm, always select arm with highest UCB value
- $\bar{X}_j$ = mean of observed rewards, n = number of plays so far
- Selecting arms with UCB gives regret growing with O(ln n), SLOWEST POSSIBLE!

UCB - Example

$$\bar{X}_j + \sqrt{\frac{2 \ln n}{n_j}}$$



- Play all arms once initially
- Then based on the formula



UCB - Example

$$\bar{X}_j + \sqrt{\frac{2\ln n}{n_j}}$$

- $\sqrt{\frac{2\ln n}{n_j}}$ is based of bound of the form $P\left(\bar{X}_j - E[X] \geq f(\sigma, n)\right) \leq \sigma$ (Comes from PAC, probably approximately correct)
- And σ is chosen to be time dependent (by n), goes to zero.
- This form derived for rewards in [0,1] interval. In practice, the formula includes "exploration constant" that scales the exploration term to the range of rewards:

$$\bar{X}_j + C\sqrt{\frac{\ln n}{n_j}}$$

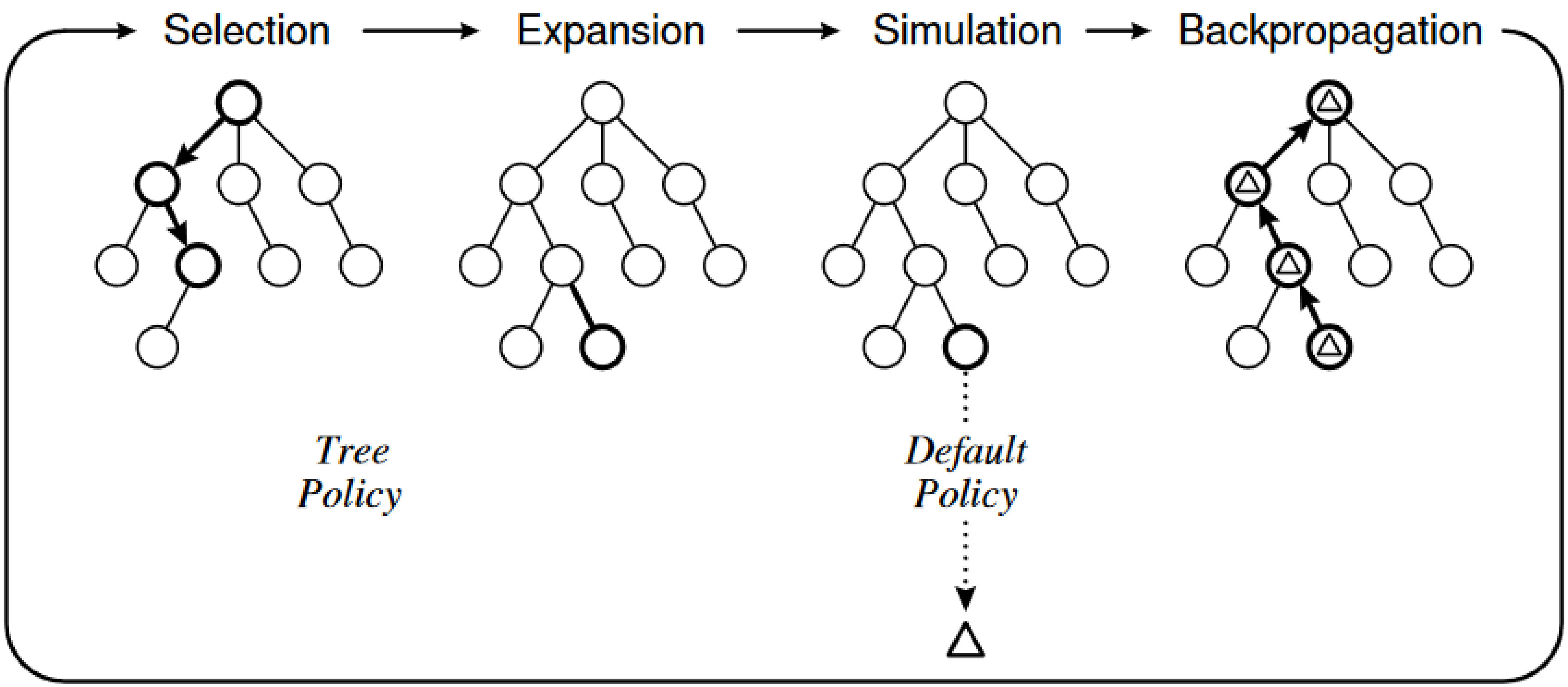
Excel example:

https://drive.google.com/open?id=1A9Kr-JDz_ZJIYOX3aFMrFaLUAPeAZV7Z

Google sheets:

https://docs.google.com/spreadsheets/d/17xxXMAGbXqjt6N1tah3VwKbusz5c44kGcAWQuhV93P0/edit?usp=sharing

UCB for Trees = UCT



- Tree node:
  - Associated state,
  - incoming action,
  - number of visits,
  - accumulated reward

- External slides by Michele Sebag: https://drive.google.com/open?id=1ytp9l33_6WNe62qLAzV326iS4WmYeFpY

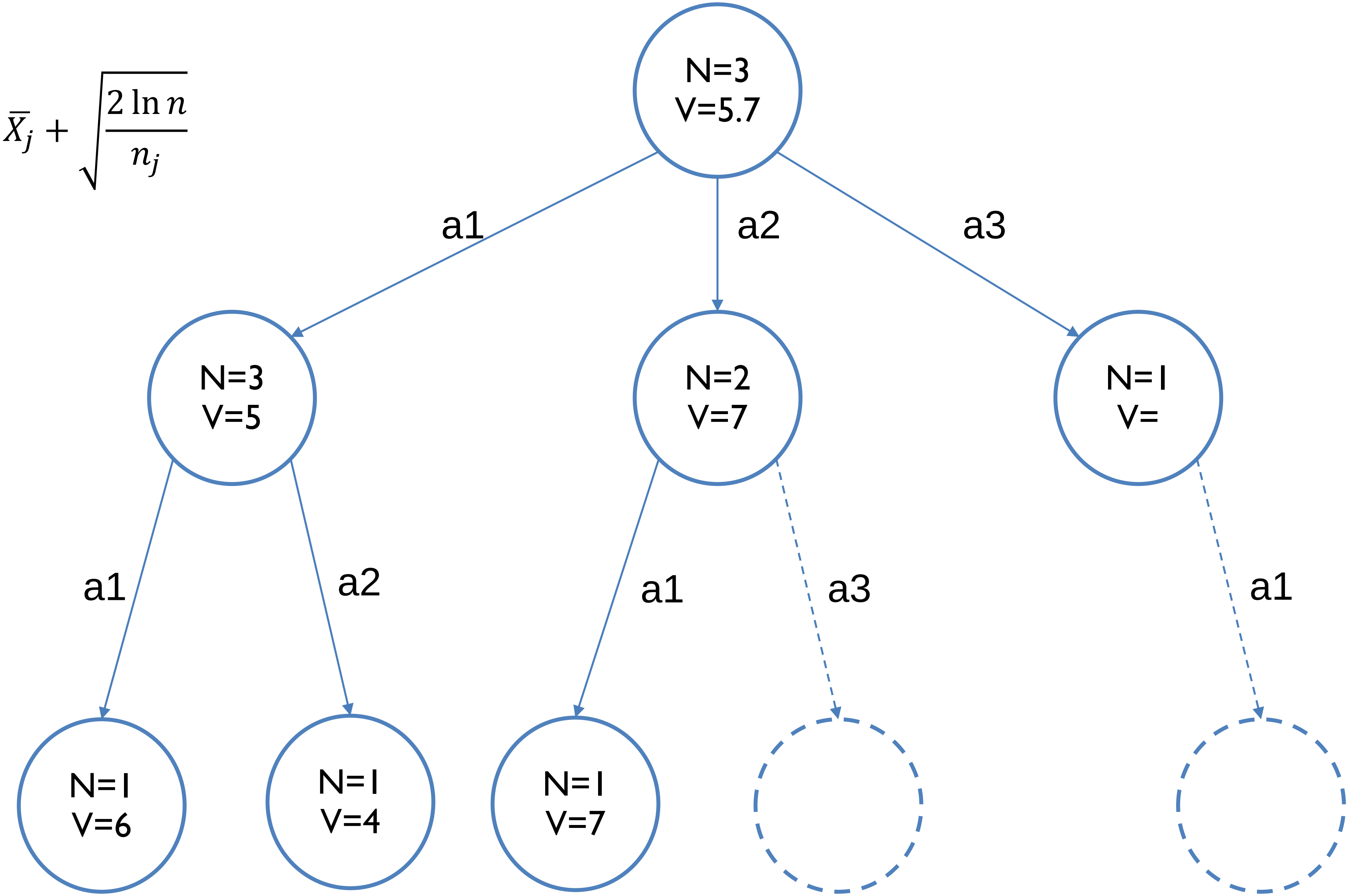
UCT algorithm

Algorithm 1 General MCTS approach.

function MCTSSEARCH(s_0)
create root node v_0 with state s_0
while within computational budget **do**
$v_l \leftarrow$ TREEPOLICY(v_0)
$\Delta \leftarrow$ DEFAULTPOLICY($s(v_l)$)
BACKUP(v_l, Δ)
return a(BESTCHILD(v_0))

MCTS example

$$\bar{X}_j + \sqrt{\frac{2 \ln n}{n_j}}$$



UCT in detail

Algorithm 2 The UCT algorithm.

function UCTSEARCH(s_0)
  create root node v_0 with state s_0
  while within computational budget **do**
    $v_l \leftarrow$ TREEPOLICY(v_0)
    $\Delta \leftarrow$ DEFAULTPOLICY($s(v_l)$)
    BACKUP(v_l, Δ)
  return a(BESTCHILD($v_0, 0$))

function TREEPOLICY(v)
  while v is nonterminal **do**
    if v not fully expanded **then**
      return EXPAND(v)
    else
      $v \leftarrow$ BESTCHILD(v, Cp)
  return v

function EXPAND(v)
  choose $a \in$ untried actions from $A(s(v))$
  add a new child v' to v
    with $s(v') = f(s(v), a)$
    and $a(v') = a$
  return v'

function BESTCHILD(v, c)

$$\textbf{return} \quad \underset{v' \in \text{children of } v}{\arg\max} \frac{Q(v')}{N(v')} + c\sqrt{\frac{2 \ln N(v)}{N(v')}}$$

function DEFAULTPOLICY(s)
  while s is non-terminal **do**
    choose $a \in A(s)$ uniformly at random
    $s \leftarrow f(s, a)$
  return reward for state s

function BACKUP(v, Δ)
  while v is not null **do**
    $N(v) \leftarrow N(v) + 1$
    $Q(v) \leftarrow Q(v) + \Delta(v, p)$
    $v \leftarrow$ parent of v

Algorithms from (PDF) A Survey of Monte Carlo Tree Search Methods (researchgate.net)

Picking the best action

When the time is up, how do we pick the best action?

- Highest UCB scoring action
- **Most used action**
- others

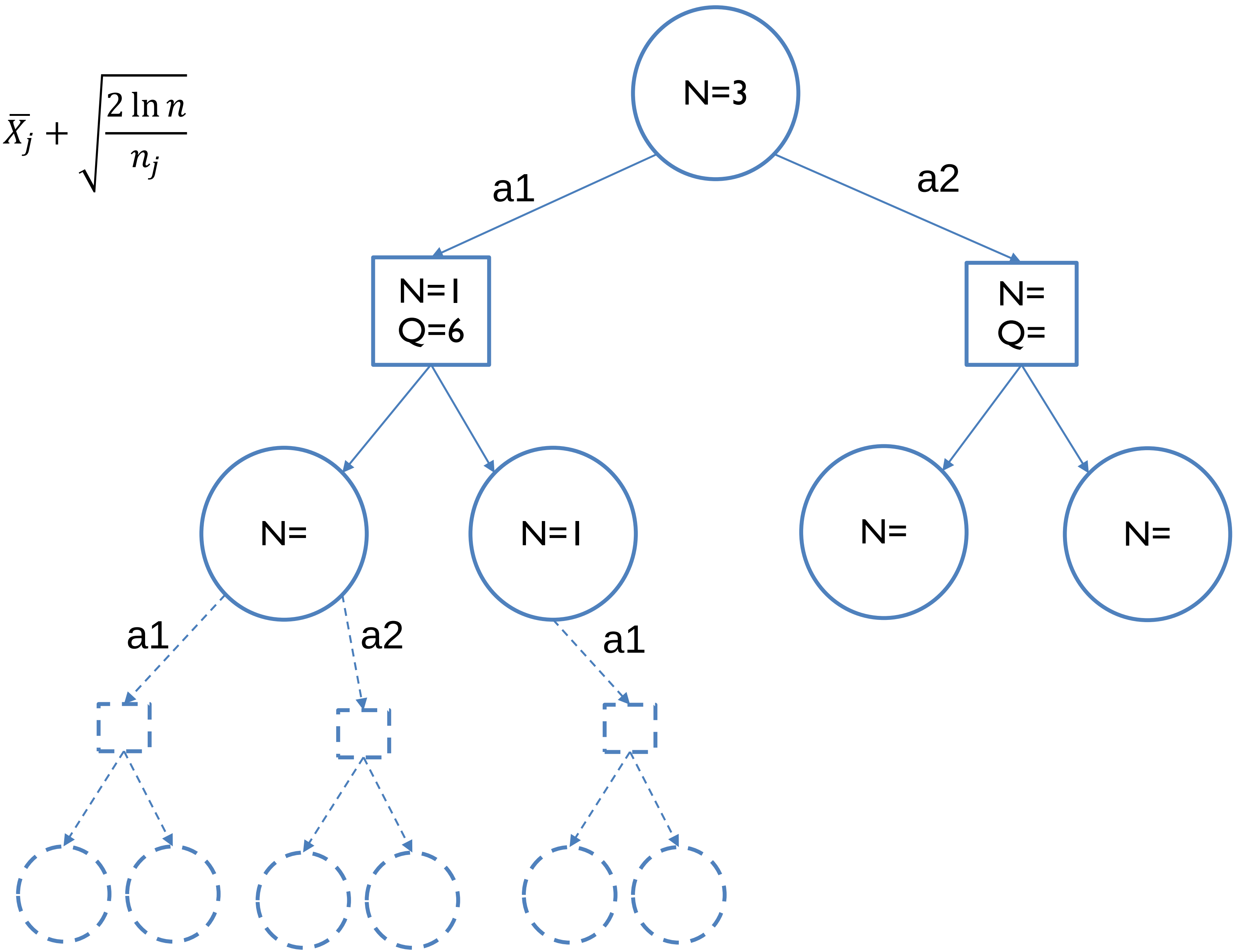
MCTS for MDP example

How do you apply MCTS to MDPs?

- What about stochastic outcomes?
- What about rewards from transitions?

MCTS for MDP example

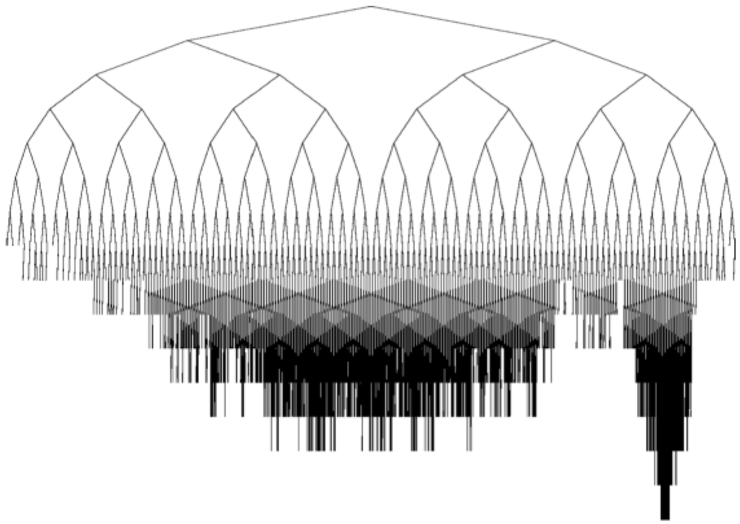
$$\bar{X}_j + \sqrt{\frac{2 \ln n}{n_j}}$$



MCTS notes

- Aheuristic
  - Does not require any domain specific knowledge
  - Domain specific knowledge can provide significant speedups
- Anytime
  - Can return currently best action when stopped at any time
- Asymmetric
  - Tree is not explored fully
- MCTS = UCT? No consistency

  in the naming



[Arnaud et al., 2007]