AIMLC ZG512 -
Deep Reinforcement Learning

Session #14:
Model Based Algorithms

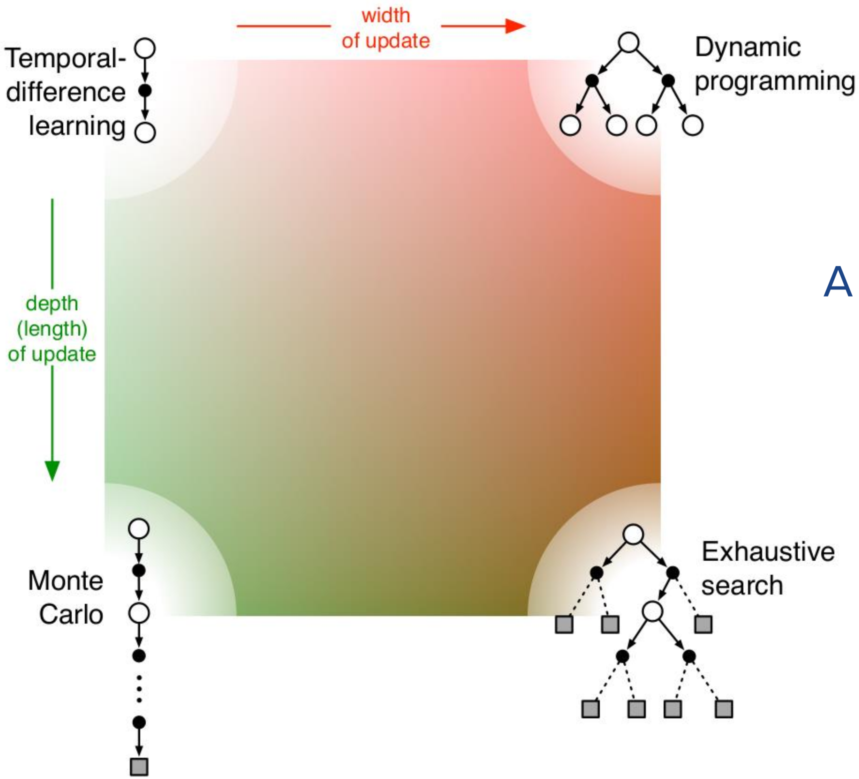
S. P. Vimal, Department of CSIS, WILP Division (vimalsp@wilp.bits-pilani.ac.in)

Agenda for the class

- ➔ Introduction
- ➔ Upper-Confidence-Bound [UCB] Action Selection
- ➔ Monte-Carlo Tree Search [MCTS]
- ➔ AlphaGo & AlphaGo Zero [Next Class]
- ➔ MuZero, PlaNet [Next Class]

S. P. Vimal, Department of CSIS, WILP Division (vimalsp@wilp.bits-pilani.ac.in)

Monte-Carlo Tree Search (MCTS)



A summary of pre-mid sem coverage !!!

Monte-Carlo Tree Search (MCTS)

Rollout Algorithms:

- Decision-time planning algorithms
- Produce Monte-Carlo estimates of action values only for each current state and for a given policy (Rollout policy)
- Simple, as there is no need to approximate a function over either the
  - entire state space (or)
  - state-action space

- How & Why?
  - Averaging the returns of the simulated trajectories produces estimates of $q\pi(s, a')$ for each action $a' \in A(s)$.
  - The policy selects an action in s that maximizes these estimates & then follows $\boldsymbol{\pi}$
- Aim of a rollout algorithm is to improve upon the rollout policy
  - Rollout policy could be completely random !!!

Monte-Carlo Tree Search (MCTS)

Rollout Algorithms:

- Decision-time planning algorithms
- Produce Monte-Carlo estimates of action values only for each current state and for a given policy (Rollout policy)
- Simple, as there is no need to approximate a function over either the
  - entire state space (or)
  - state-action space

- How & Why?
  - Averaging the returns of the simulated trajectories produces estimates of $q_\pi(s, a')$ for each action $a' \in A(s)$.
  - The policy selects an action in s that maximizes these estimates & then follows $\boldsymbol{\pi}$
- Aim of a rollout algorithm is to improve upon the rollout policy
  - Rollout policy could be completely random !!!

- MCTS is a recent and strikingly successful example of decision-time planning
- An enhanced rollout algorithm
  - Accumulates value estimates obtained from the simulations to successively direct simulations toward more highly-rewarding trajectories

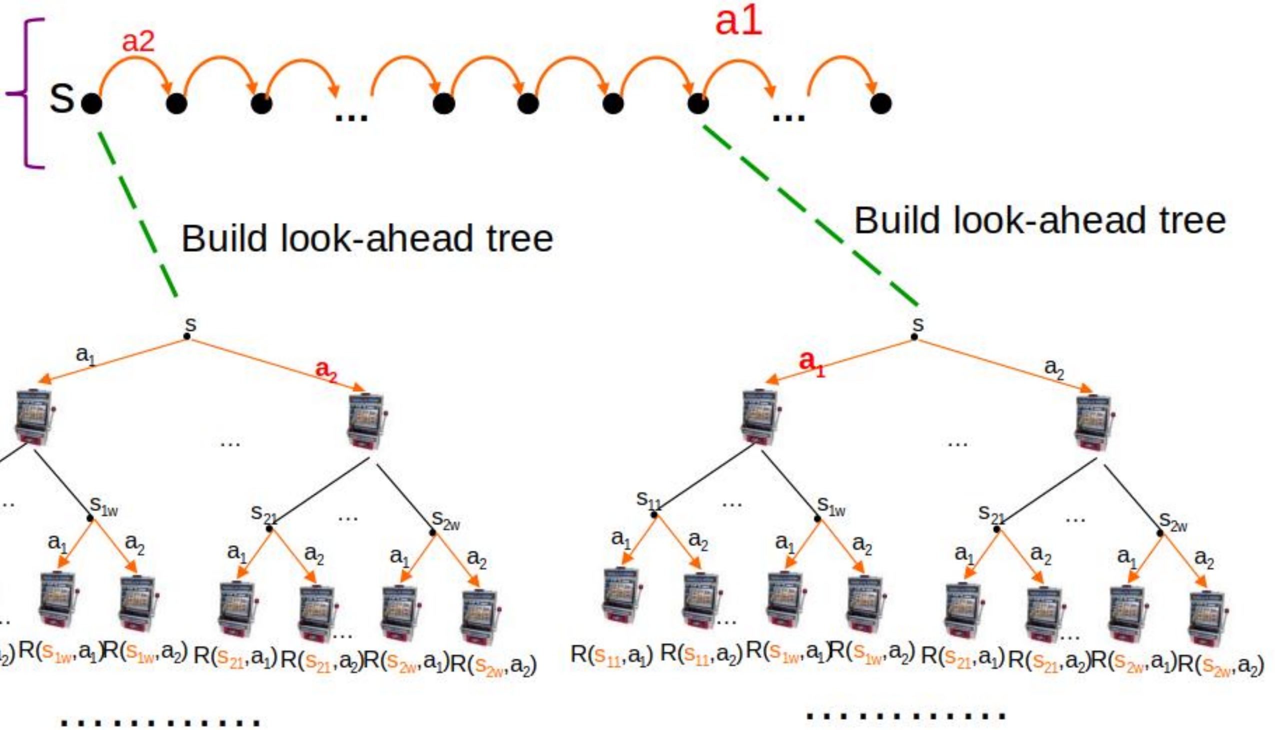
Monte-Carlo Tree Search (MCTS)

How MCTS works?

- MCTS is *executed* after encountering each new state (s)
    - [?] to select the agent's action for s
- *Each execution is an iterative process* that simulates many trajectories starting from s and
    - running to a terminal state (or)
    - until discounting makes any further reward negligible to the return
- Focus on multiple simulations starting at s by extending the initial portions of trajectories that have received high evaluations from earlier simulations.

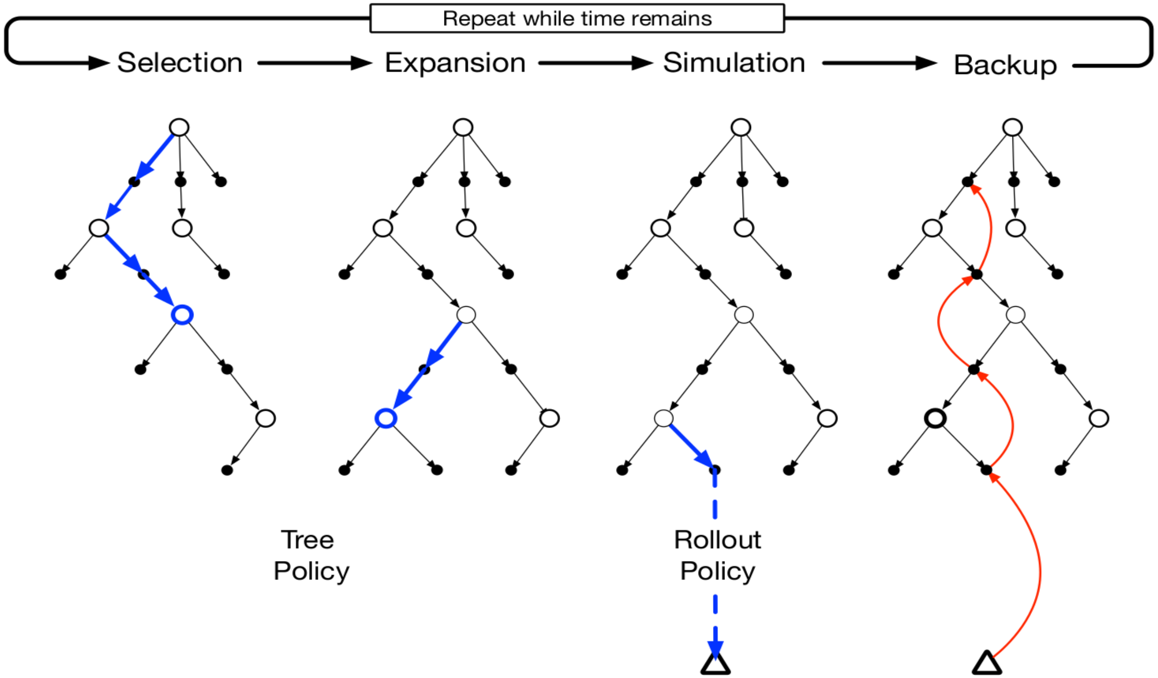
Monte-Carlo Tree Search (MCTS)

Real world state/action sequence



S. P. Vimal, Department of CSIS, WILP Division (vimalsp@wilp.bits-pilani.ac.in)

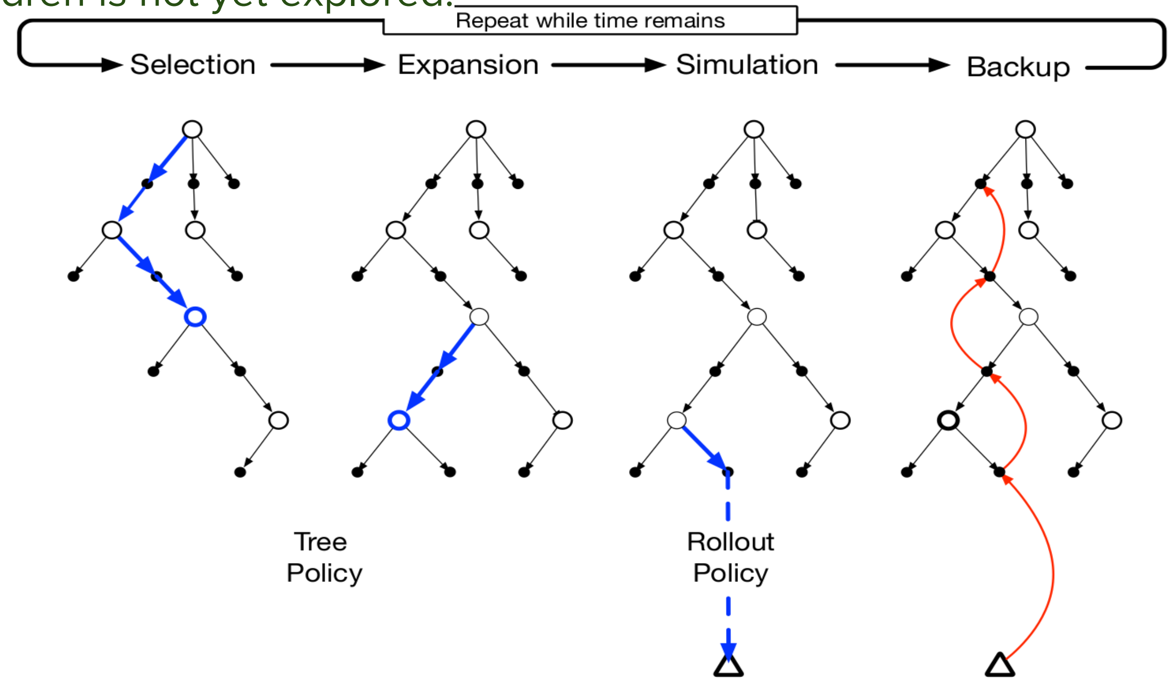
Monte-Carlo Tree Search (MCTS)



$$S_i = x_i + C\sqrt{\frac{\ln(t)}{n_i}}$$

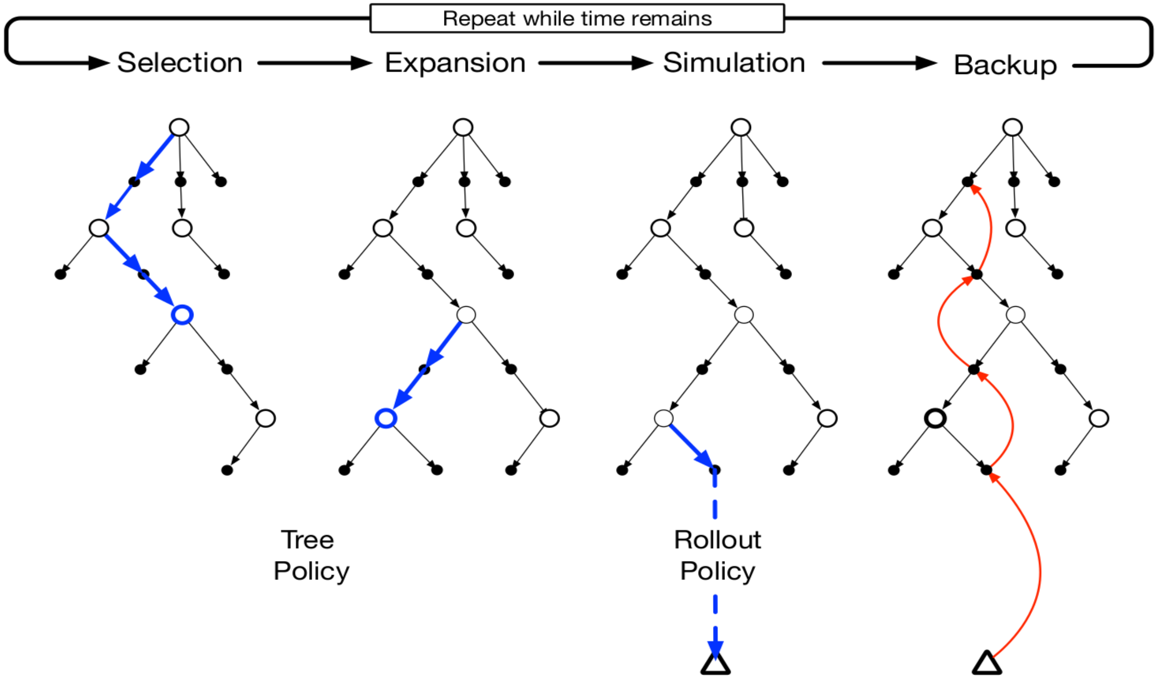
Monte-Carlo Tree Search (MCTS) -- Selection

Select: Select a single node in the tree that is *not fully expanded*. By this, we mean at least one of its children is not yet explored.



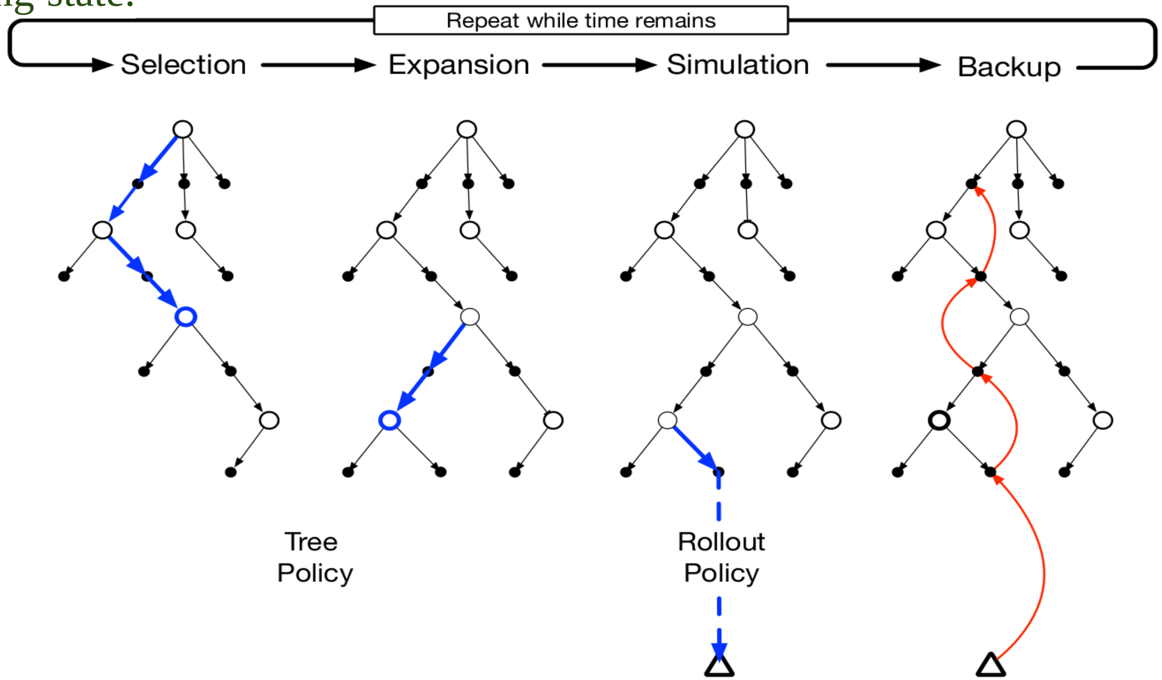
Monte-Carlo Tree Search (MCTS) -- Expansion

Expand: Expand this node by applying one available action (as defined by the MDP) from the node.



Monte-Carlo Tree Search (MCTS) -- Simulation

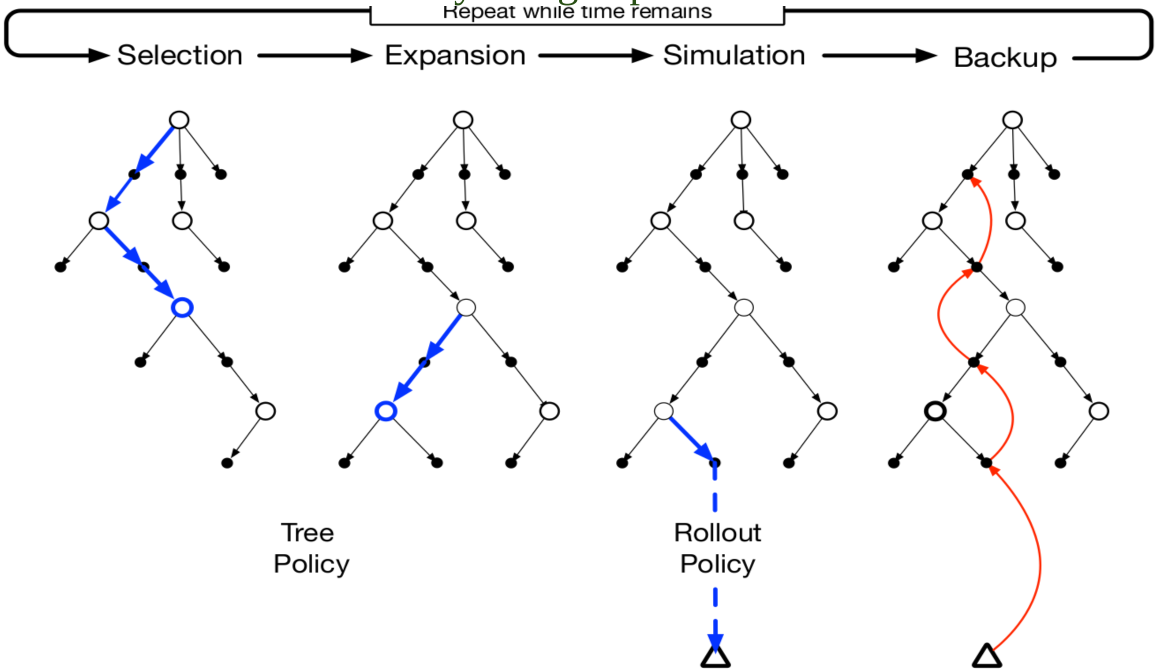
Simulation: From one of the outcomes of the expanded, perform a complete random simulation oto a terminating state.



S. P. Vimal, Department of CSIS, WILP Division (vimalsp@wilp.bits-pilani.ac.in)

Monte-Carlo Tree Search (MCTS) -- Backup

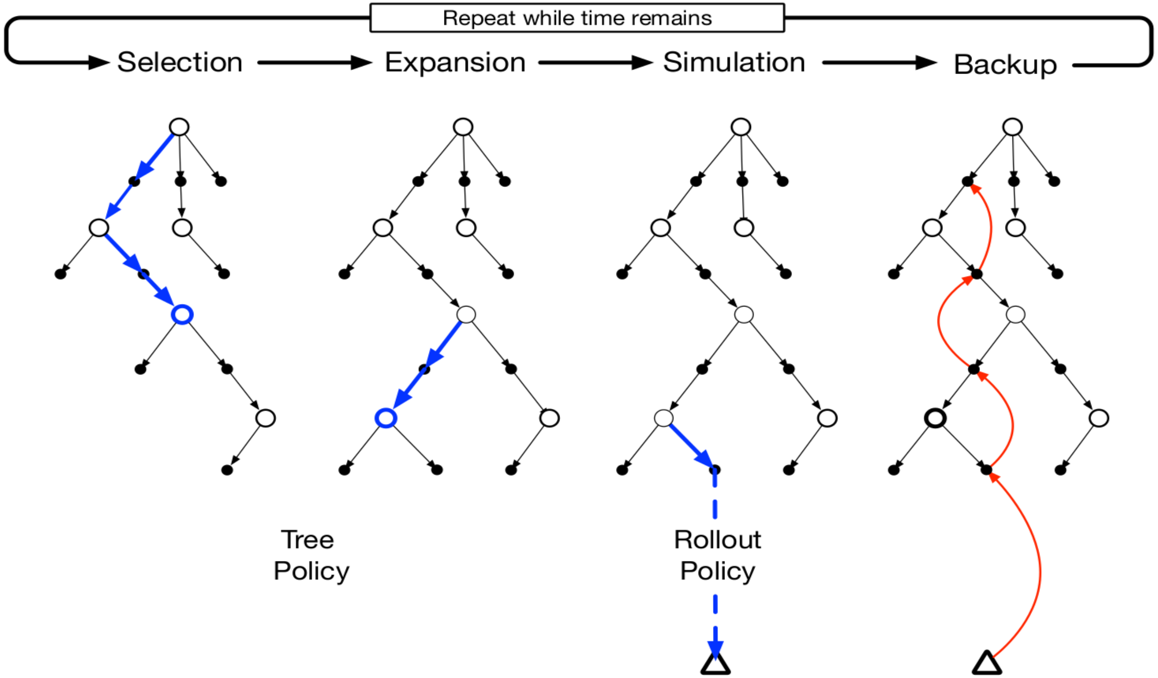
Backup/ Backpropagate: The value of the node is *back propagated* to the root node, updating the value of each ancestor node on the way using expected value



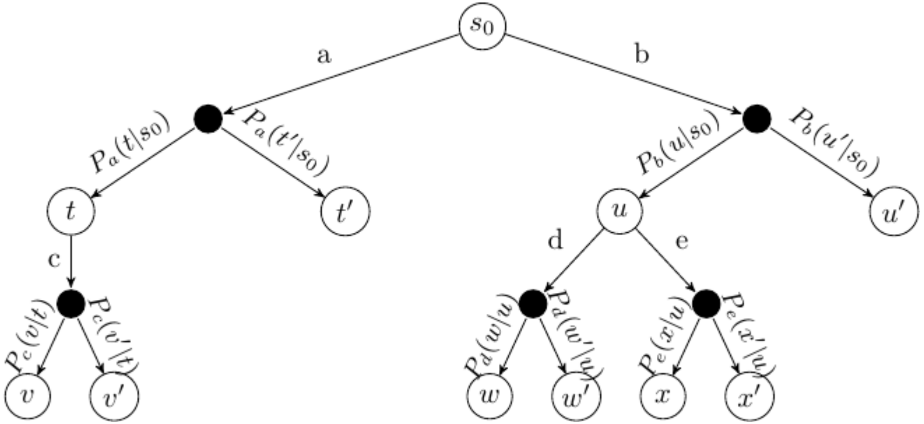
S. P. Vimal, Department of CSIS, WILP Division (vimalsp@wilp.bits-pilani.ac.in)

Monte-Carlo Tree Search (MCTS) -- Summarizing

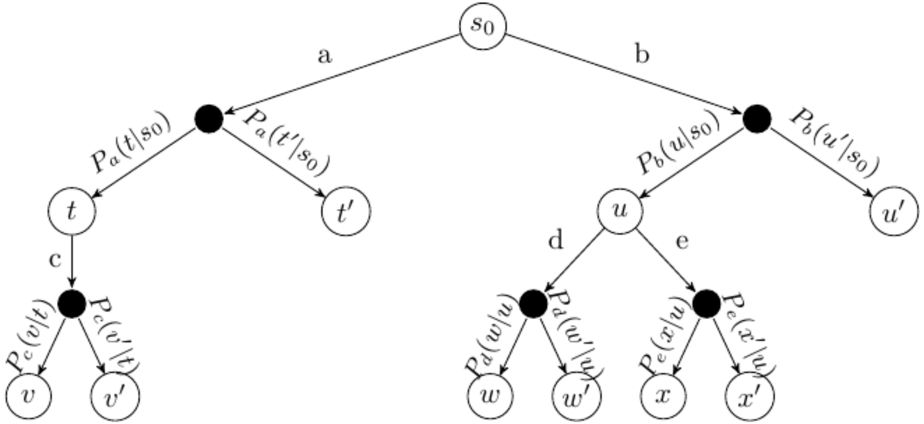
Comments on the overall approach,,,,

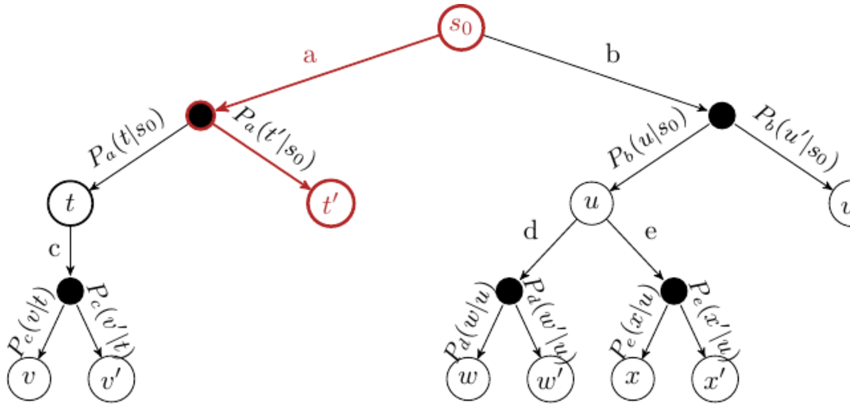


Monte-Carlo Tree Search (MCTS) -- Selection

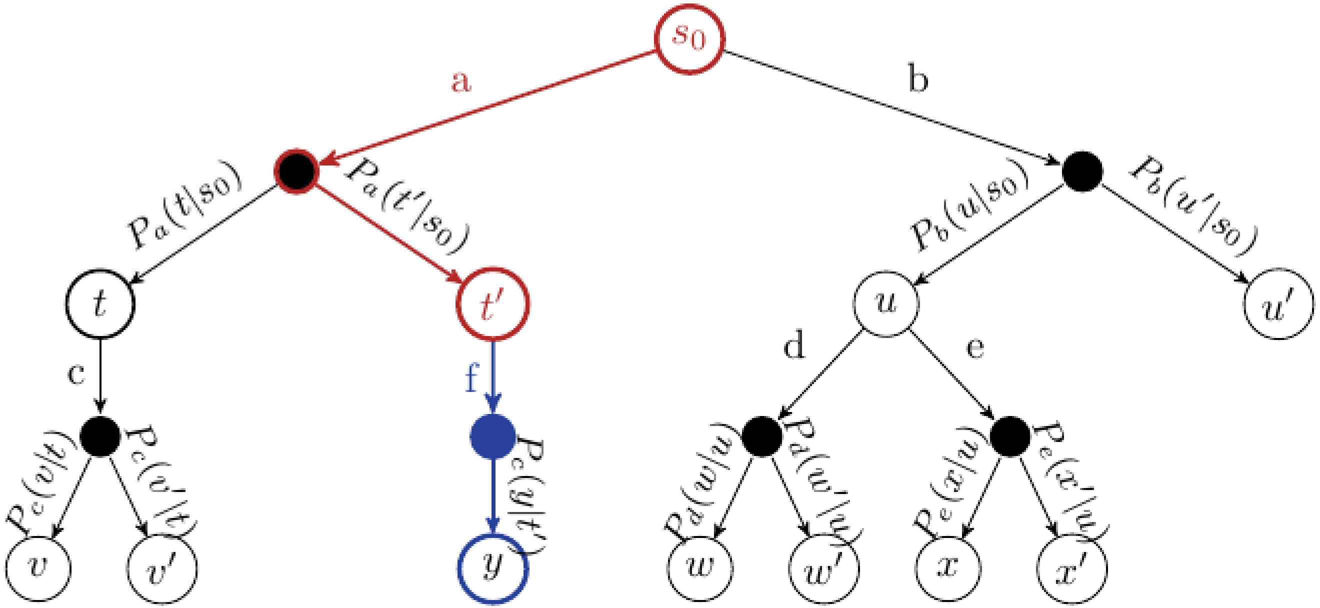


Monte-Carlo Tree Search (MCTS) -- Selection



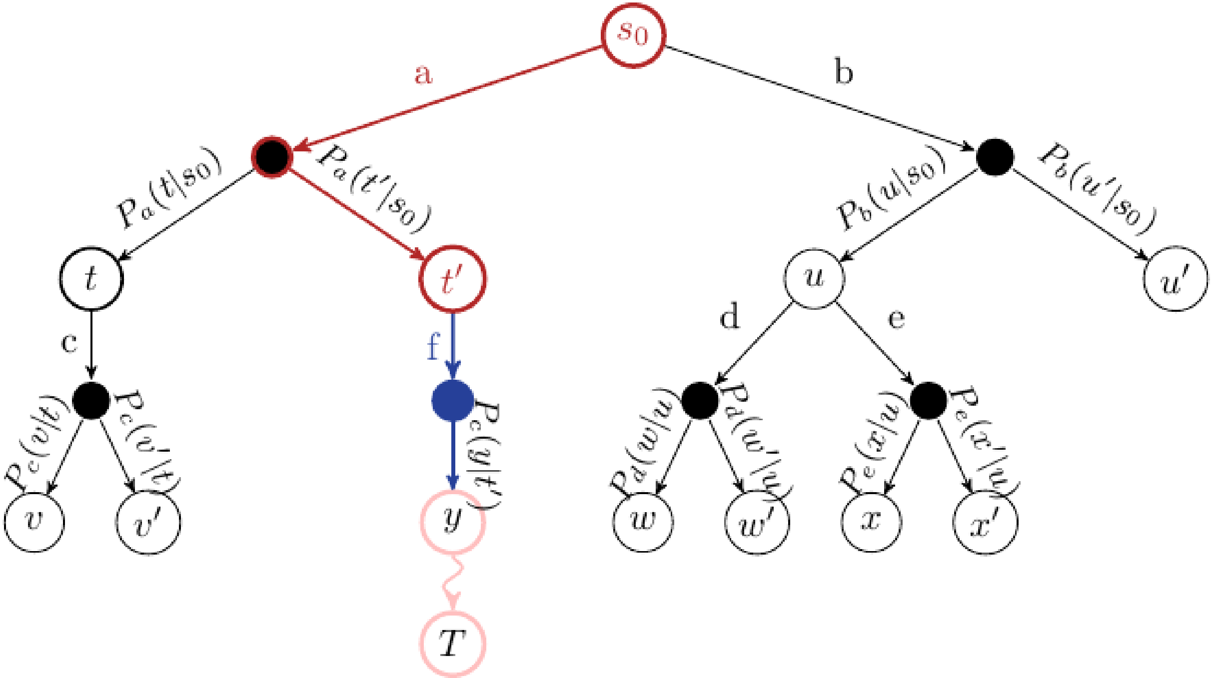


Monte-Carlo Tree Search (MCTS) -- Expansion

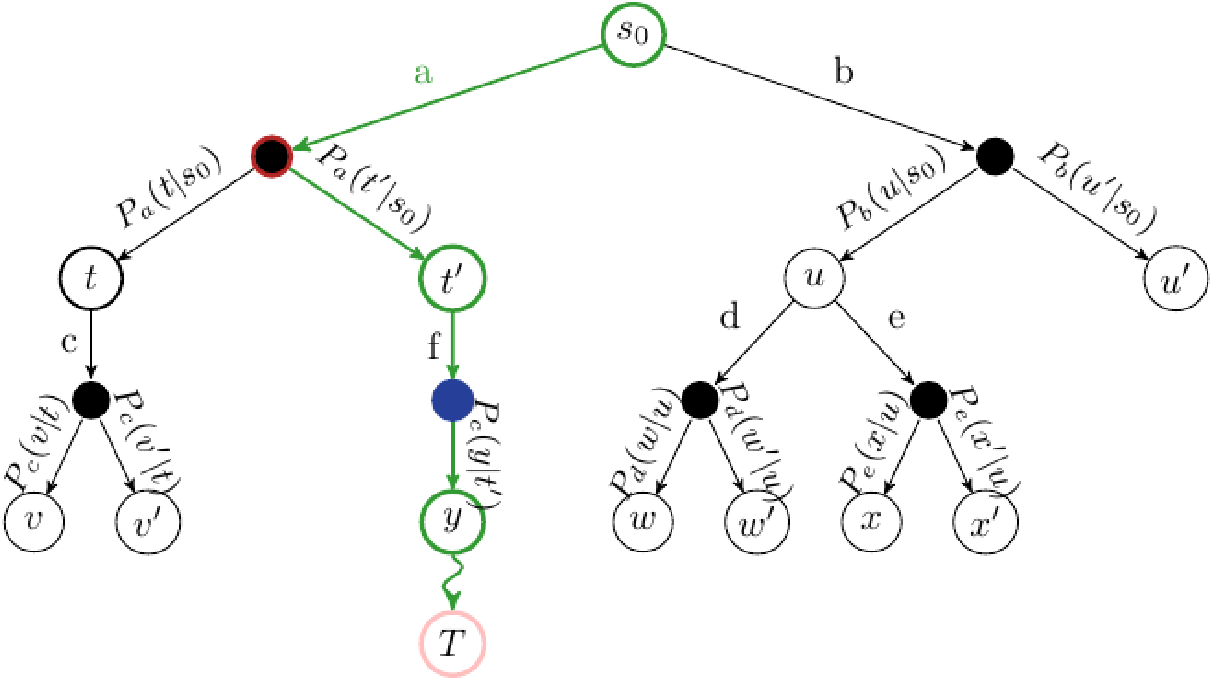


S. P. Vimal, Department of CSIS, WILP Division (vimalsp@wilp.bits-pilani.ac.in)

Monte-Carlo Tree Search (MCTS) -- Simulation



Monte-Carlo Tree Search (MCTS) -- Backup



Monte-Carlo Tree Search (MCTS)

Algorithm – Monte-Carlo Tree Search

Input: MDP $M = \langle S, s_0, A, P_a(s' \mid s), r(s, a, s')\rangle$, base value function Q, time limit T.

Output: updated Q-function Q

while $currentTime < T$

  $selected\_node \leftarrow \text{Select}(s_0)$

  $child \leftarrow \text{Expand}(selected\_node)$ – expand and choose a child to simulate

  $G \leftarrow \text{Simulate}(child)$ – simulate from $child$

  $\text{Backpropagate}(selected\_node, child, G)$

return Q

Monte-Carlo Tree Search (MCTS)

Function – $\mathrm{Select}(s : S)$

Input: state s

Output: unexpanded state

while s is fully expanded

    Select action a to apply in s using a multi-armed bandit algorithm

    Choose one outcome s' according to $P_a(s' \mid s)$

    $s \leftarrow s'$

return s

Monte-Carlo Tree Search (MCTS)

Function – $\text{Expand}(s : S)$

Input: state s

Output: expanded state s'

Select an action a from s to apply

Expand one outcome s' according to the distribution $P_a(s' \mid s)$ and observe reward r

return s'

S. P. Vimal, Department of CSIS, WILP Division (vimalsp@wilp.bits-pilani.ac.in)

Monte-Carlo Tree Search (MCTS)

Procedure – $\text{Backpropagation}(s : S; a : A)$

Input: state-action pair (s, a)

Output: none

do

$N(s,a) \leftarrow N(s,a) + 1$

$G \leftarrow r + \gamma G$

$Q(s,a) \leftarrow Q(s,a) + \frac{1}{N(s,a)}[G - Q(s,a)]$

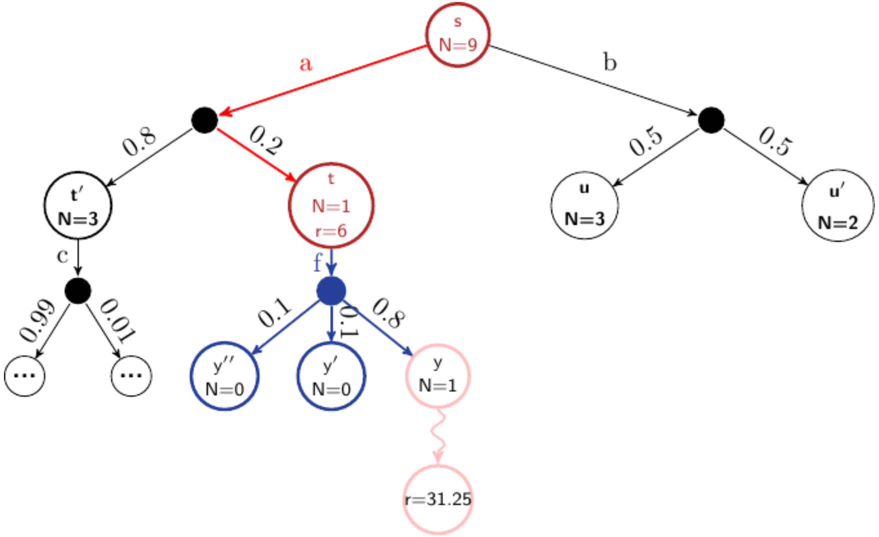
$s \leftarrow$ parent of s

$a \leftarrow$ parent action of s

while $s \neq s_0$

S. P. Vimal, Department of CSIS, WILP Division (vimalsp@wilp.bits-pilani.ac.in)

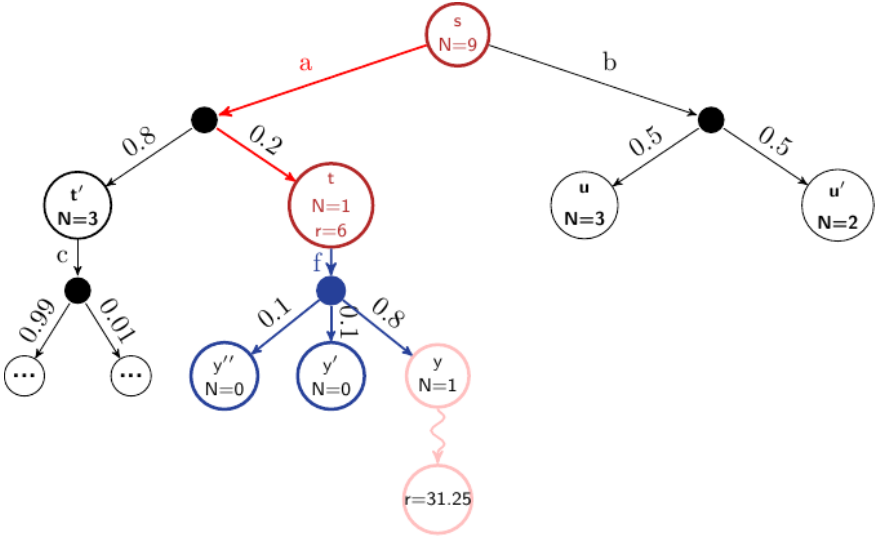
Monte-Carlo Tree Search (MCTS)



Before backpropagation

$$\begin{aligned} Q(s,a) &= 18 \\ Q(t,f) &= 0 \end{aligned}$$

Monte-Carlo Tree Search (MCTS)



The backpropagation step is then calculated for the nodes y, t, and s as follows:

$$
\begin{aligned}
Q(y,g) &= \gamma^2 \times 31.25 \ \ \text{(simulation is 3 steps long and receives reward of 31.25)}\\
&= 20\\[1em]
N(t,f) &\leftarrow N(t,f)+1 = N(y)+N(y')+N(y'')+1 = 2\\
Q(t,f) &= Q(t,f) + \frac{1}{N(t,f)}[r+\gamma G - Q(t,f)]\\
&= 0 + \frac{1}{2}[0 + 0.8 \cdot 20 - 0]\\
&= 8\\[1em]
N(s,a) &\leftarrow N(s,a)+1 = N(t)+N(t')+1 = 5\\
Q(s,a) &= Q(s,a) + \frac{1}{N(s,a)}[r+\gamma G - Q(s,a)]\\
&= 18 + \frac{1}{5}[6 + 0.8 \cdot (0.8 \cdot 20) - 18]\\
&= 18 + \frac{1}{5}[6 + 12.8 - 18]\\
&= 18.16
\end{aligned}
$$

Before backpropagation

$$
\begin{aligned}
Q(s,a) &= 18\\
Q(t,f) &= 0
\end{aligned}
$$

S. P. Vimal, Department of CSIS, WILP Division (vimalsp@wilp.bits-pilani.ac.in)

S. P. Vimal, Department of CSIS, WILP Division (vimalsp@wilp.bits-pilani.ac.in)

Upper-Confidence-Bound Action Selection

- **ε**-greedy action selection forces the non-greedy actions to be tried,
  - Indiscriminately, with no preference for those that are nearly greedy or particularly uncertain
- It would be better to select among the non-greedy actions according to their potential for actually being optimal
  - Take into account both how close their estimates are to being maximal and the uncertainties in those estimates.

$$A_t \doteq \underset{a}{\arg\max}\left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}}\right]$$

S. P. Vimal, Department of CSIS, WILP Division (vimalsp@wilp.bits-pilani.ac.in)

Upper-Confidence-Bound Action Selection

- Each time a is selected the uncertainty is presumably reduced
- Each time an action other than a is selected, t increases but $N_t(a)$ does not; because t appears in the numerator, the uncertainty estimate increases.
- Actions with lower value estimates, or that have already been selected frequently, will be selected with decreasing frequency over time

Action Value at time t for a

Confidence Level

Measure of Uncertainty

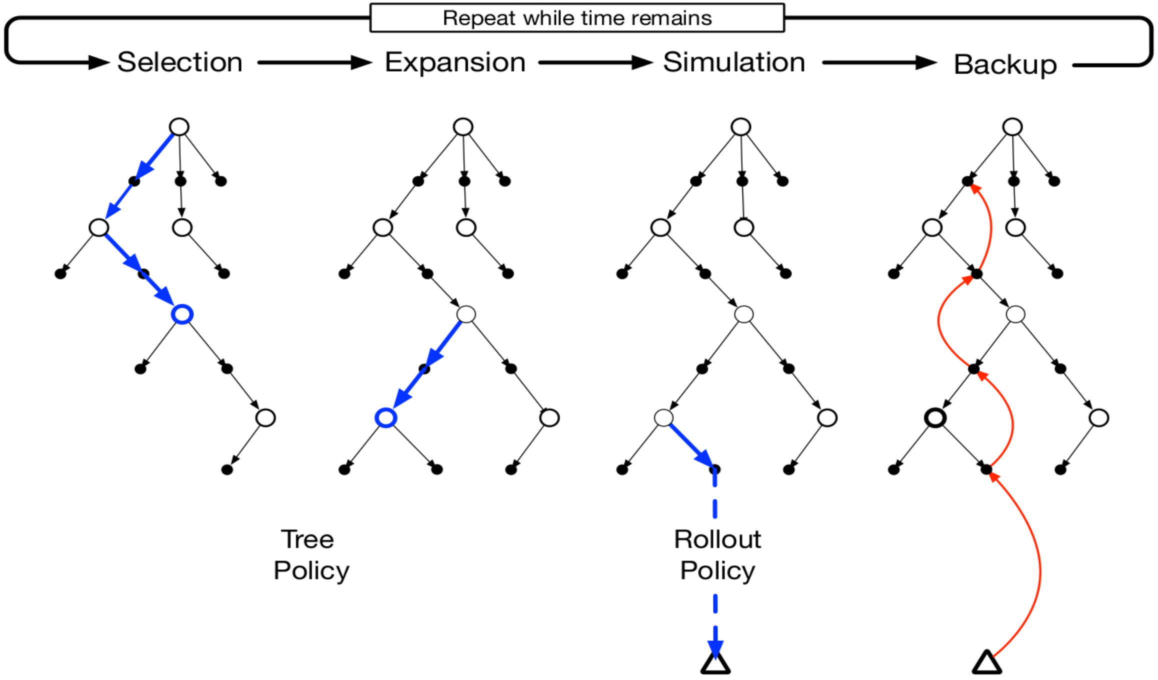
$$A_t \doteq \underset{a}{\arg\max} \left[Q_t(a) + c\boxed{\sqrt{\frac{\ln t}{N_t(a)}}} \right]$$

S. P. Vimal, Department of CSIS, WILP Division (vimalsp@wilp.bits-pilani.ac.in)

Monte-Carlo Tree Search (MCTS)

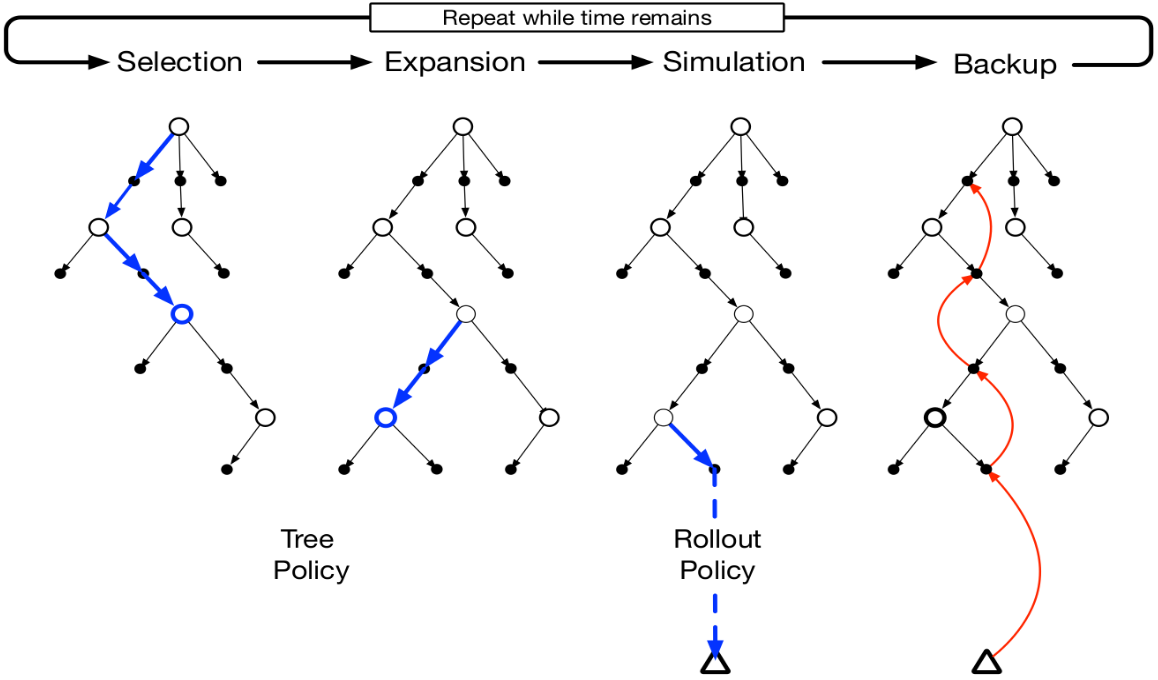
Can the selection of action in Tree policy use UCB?

$$S_i = x_i + C\sqrt{\frac{\ln(t)}{n_i}}$$



Monte-Carlo Tree Search (MCTS)

Can the selection of action in Tree policy use UCB?



<u>Upper Confidence Trees (UCT):</u>
MCTS with UCB for Tree policy

Required Readings and references

1. https://rl-lab.com/#play
2. https://www.aionlinecourse.com/tutorial/machine-learning/upper-confidence-bound-%28ucb%29
3. https://towardsdatascience.com/monte-carlo-tree-search-in-reinforcement-learning-b97d3e743d0f
4. https://gibberblot.github.io/rl-notes/single-agent/mcts.html
5. https://towardsdatascience.com/alphazero-chess-how-it-works-what-sets-it-apart-and-what-it-can-tell-us-4ab3d2d08867
6. https://medium.com/geekculture/muzero-explained-a04cb1bad4d4
7. https://towardsdatascience.com/everything-you-need-to-know-about-googles-new-planet-reinforcement-learning-network-144c2ca3f284
8. https://blog.research.google/2019/02/introducing-planet-deep-planning.html?m=1

Thank you