Monte Carlo Tree Search (MCTS)

Consider the following real-life situation:

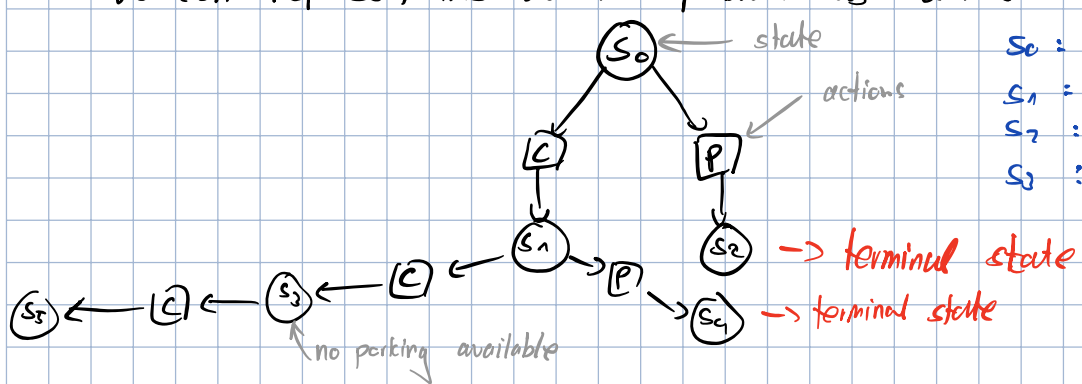
It is a Sunday afternoon and you're with your friends out for lunch, looking for a parking spot on University Ave. Your friend finds a parking spot, but it's still four blocks away from the restaurant. Do you keep going and hope for a spot closer to the restaurant or do you estimate that the stomachs of too many other fellow Palo Altonians is empty and they're getting food too?

→ State: position on University Ave and still driving or parked

→ Action: Continue or park? (c/p)

→ Reward: "−distance to restaurant" → closer to restaurant = less negative reward

→ We can represent this decision problem as a tree:



s_0 : at start position and driving
s_1 : 2nd block and still driving
s_2 : 1st block and parked
s_3 : No parking available

→ How does Monte Carlo Tree Search solve that problem?

Recipe – MCTS:

→ 1. **Selection**: Starting from the root node: choose for each decision the action that has the highest value of $\underbrace{Q(s,a) + \underbrace{c \cdot \sqrt{\frac{\log(N(s))}{N(s,a)}}}_{\text{exploration bonus}}}_{UCB1}$ until a leaf node has been reached

→ $Q(s,a)$ is action value for a given state

→ $N(s,a)$ how often a certain state-action combination has been tried → $N(s) = \sum_a N(s,a)$

→ c is exploration parameter → we set $c = 1$

2. Expansion : If the leaf node does not terminate the MDP, create leaf nodes for all allowable actions

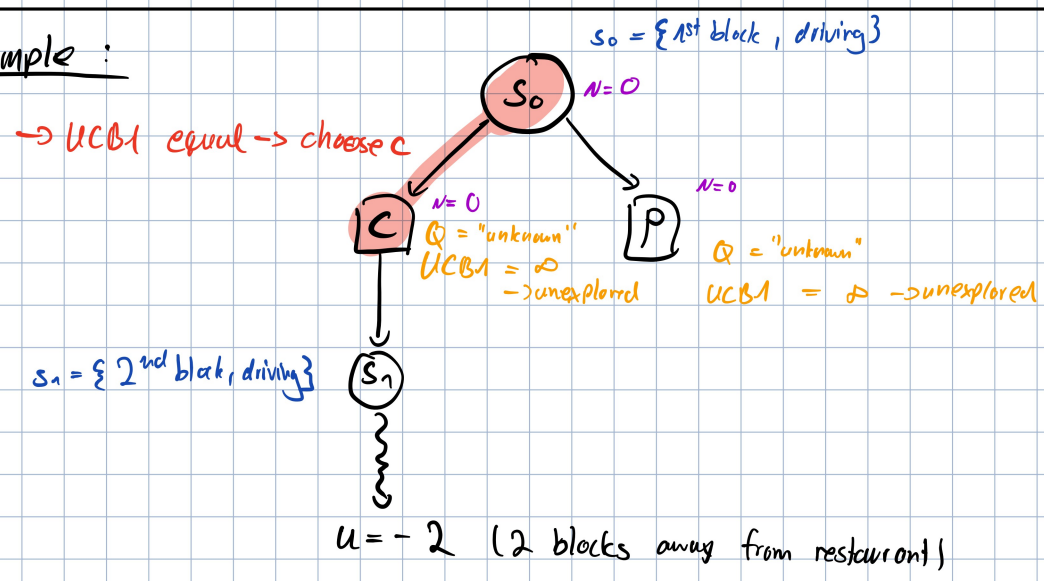
Choose action with highest UCB1 and sample state

3. Simulation : Simulate using a random policy (i.e. take random actions until a terminating state is reached)

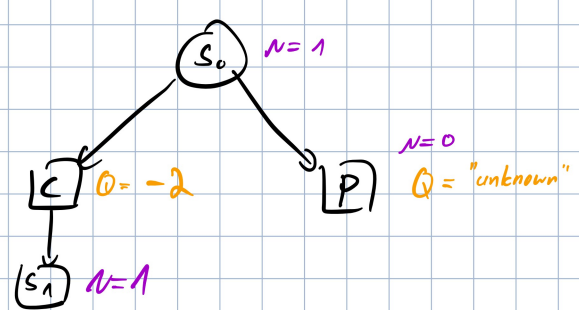
Report value for state

4. Update : Update $Q(s,a)$ for all parental nodes

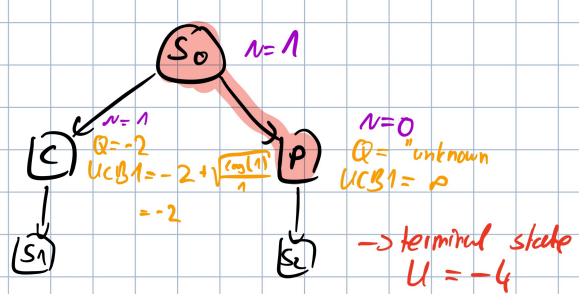
Example :



1st loop :
- selection
- expansion
- simulation

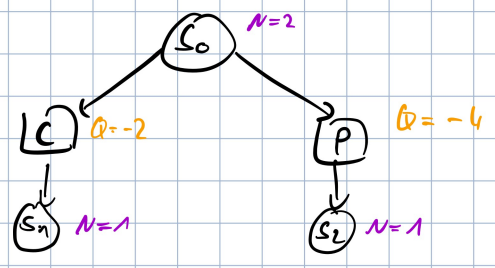


1st loop :
- update

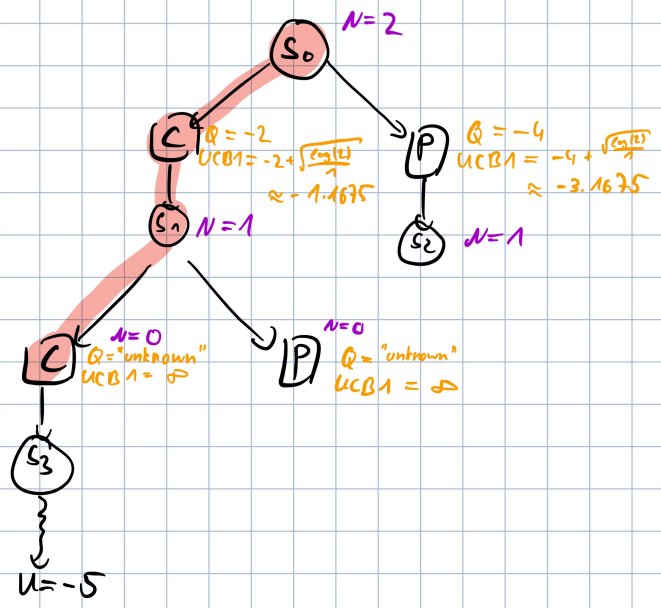
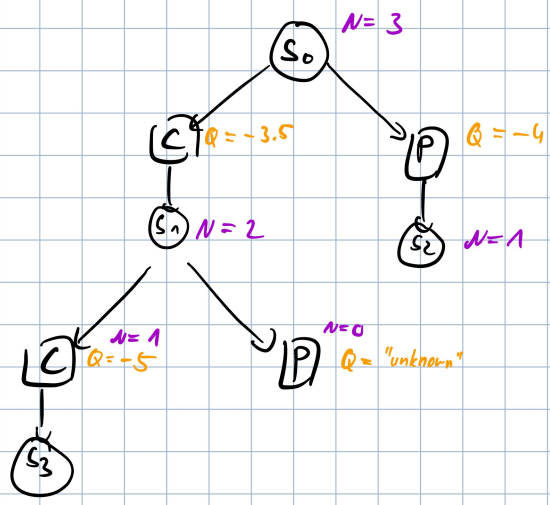


2nd loop :
- selection
- (expansion)
- simulation

2nd loop:
- update



3rd loop:
- selection
- expansion
- simulation



8rd loop:
- update