A futuristic scene with a boat on water and light trails in the sky. The background features a dark sky with several bright, diagonal light trails in shades of orange and red, suggesting a high-speed or space-themed environment. In the foreground, a small boat with three figures is silhouetted against the water. The overall mood is mysterious and technological.

“To succeed, planning alone is insufficient.  
One must improvise as well.”

Isaac Asimov, *Foundation*

# **CMPUT 365**

## **Introduction to RL**

# Reminder I

You **should be enrolled in the private session** we created in Coursera for CMPUT 365.

I **cannot** use marks from the public repository for your course marks. You **need to check, every time**, if you are in the private session and if you are submitting quizzes and assignments to the private section.

At the end of the term, I **will not port grades** from the public session in Coursera.

If you have any questions or concerns, **talk with the TAs** or email us `cmput365@ualberta.ca`.

# Reminder II

- What I plan to do today:
  - Monte-Carlo Tree Search. Wrap up Part I of Textbook.
  - An exercise.
- Midterm 2 is Friday.
  - Same thing as before (no calculators, closed-book, no cheat sheet).
  - Bring your OneCard.

**Please, interrupt me at any time!**



## Last Class: When the Model Is Wrong

- A model can be wrong for all sorts of reasons (e.g., stochastic environment, function approximation, non-stationarity in the environment).
- An incorrect model often leads to suboptimal policies.
- One needs to constantly explore to refine the learned model.
  - Exploration: take actions that improve the model.
  - Exploitation: behaving in the optimal way given the current model.
- Dyna-Q+: Provides “bonus rewards” for long-untried actions. Specifically, consider the reward  $r + \kappa\sqrt{\tau}$  in the **planning step**, where  $\tau$  is the number of time steps since that transition was tried for the last time.



# Rollout Algorithms

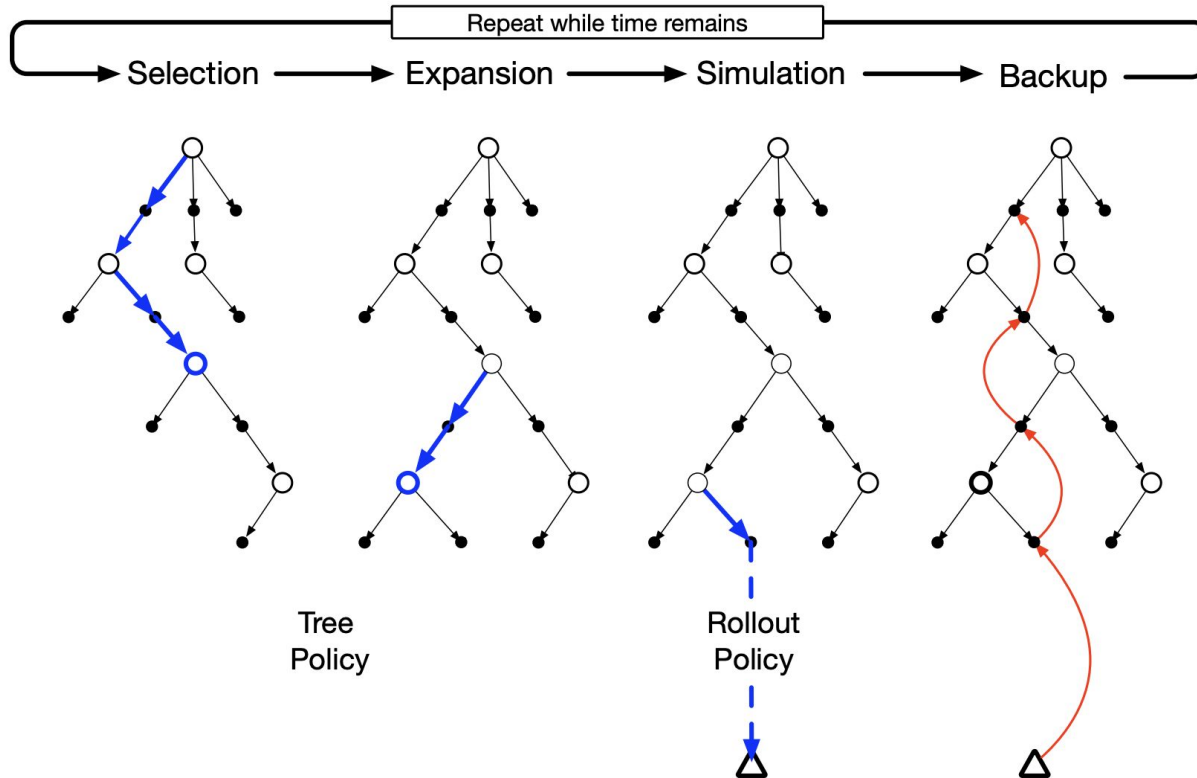
- Rollout algorithms are decision-time planning algorithms based on MC control applied to simulated trajectories that all begin at the current environment state.
  - They estimate action values for a given policy by averaging the returns of many simulated trajectories that start with each possible action and then follow the given policy.
- Unlike the Monte Carlo control algorithms previously described, the goal of a rollout algorithm is not to estimate a complete optimal action-value function,  $q_*$ , or a complete action-value function,  $q_\pi$ , for a given policy  $\pi$ .
  - They produce Monte Carlo estimates of action values only for each current state and for a given policy usually called the rollout policy.
- They are not learning algorithms *per se*, but they do leverage the RL toolkit.

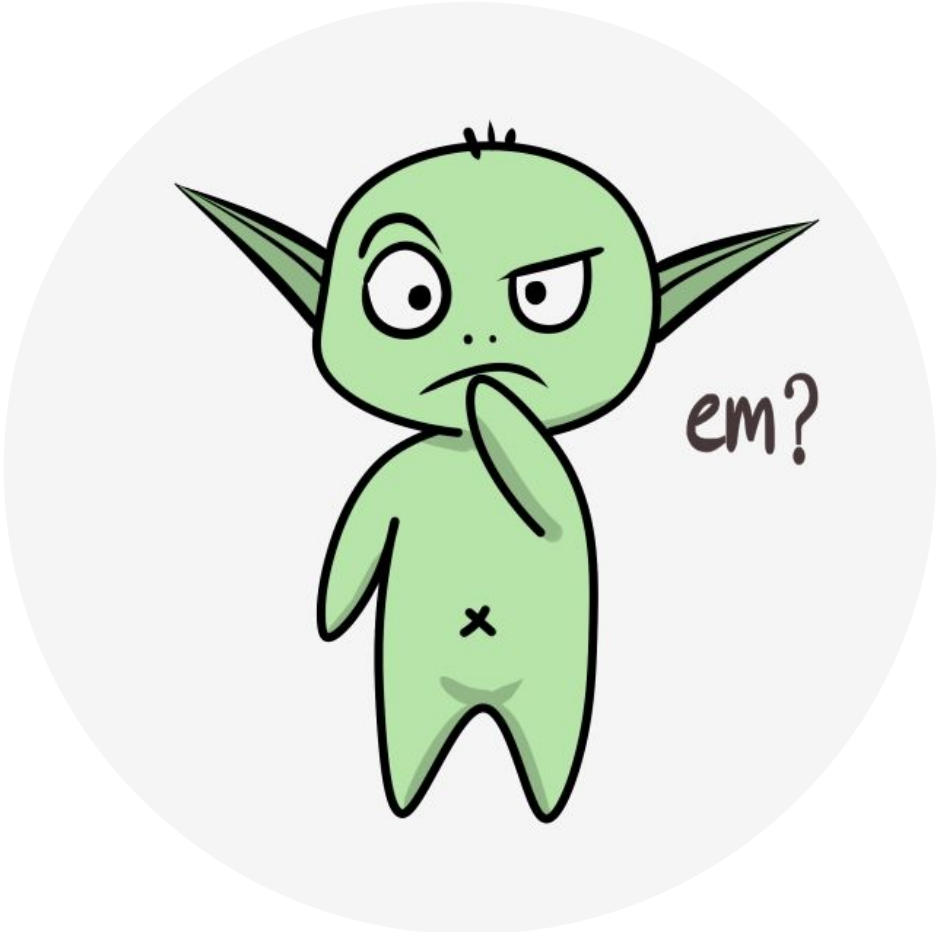
# Monte Carlo Tree Search (MCTS)

- MCTS is a great example of a rollout, decision-time planning algorithm.
  - But enhanced by the addition of a means for accumulating value estimates obtained from the MC simulations in order to successively direct simulations toward more highly-rewarding trajectories.
- The core idea of MCTS is to successively focus multiple simulations starting at the current state by extending the initial portions of trajectories that have received high evaluations from earlier simulations.
  - Monte Carlo value estimates are maintained only for the subset of state–action pairs that are most likely to be reached in a few steps, which form a tree rooted at the current state.

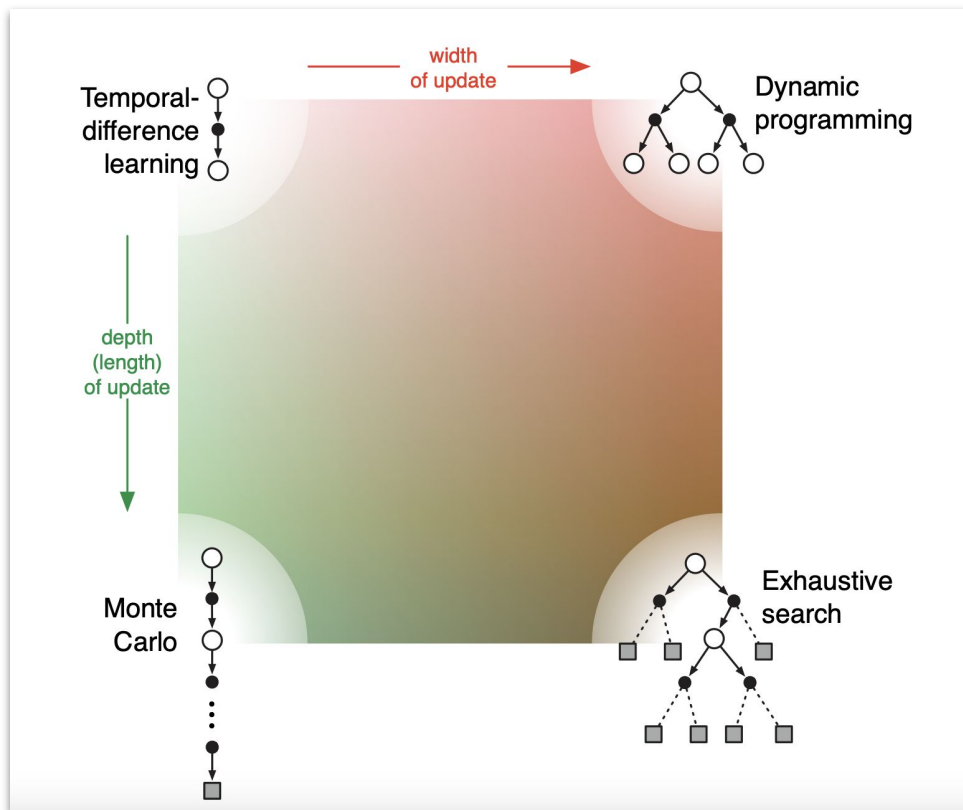


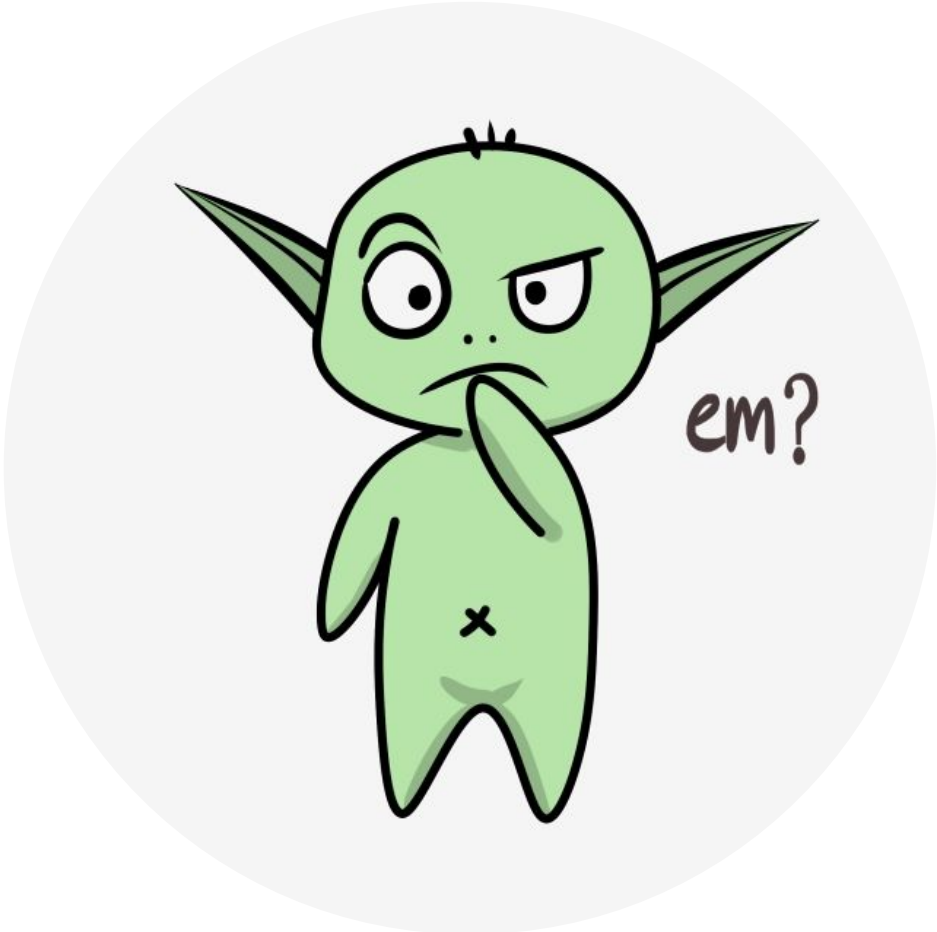
# Monte Carlo Tree Search (MCTS)





# Wrapping Up





## Exercise

Consider an MDP with three states  $\mathcal{S} = \{1, 2, 3\}$ , where each state has two possible actions  $\mathcal{A} = \{1, 2\}$  and a discount rate  $\gamma = 0.5$ . Suppose estimates of  $Q(S, A)$  are initialized to 0 and you observed the following episode according to an unknown behaviour policy where  $S_3$  is the terminal state.

$$S_0 = 1, A_0 = 1, R_1 = -7, S_1 = 3, A_1 = 2, R_2 = 5, S_2 = 1, A_2 = 1, R_3 = 10$$

- Suppose you used Q-learning with the above trajectory to estimate  $Q(S, A)$ , what are your new estimates for  $Q(S = 1, A = 1)$  using  $\alpha = 0.1$ ?
- What is one possible model for this environment? Is the model stochastic or deterministic?
- Suppose in the planning loop, after search control, we would like to update  $Q(S = 1, A = 1)$  with Q-planning. What are the possible outputs of  $\text{Model}(S = 1, A = 1)$ ?
- If your model outputs  $R = R_3$  and  $S' = S_3$ , what is  $Q(S = 1, A = 1)$  after one Q-planning update? Use the estimates of  $Q(S, A)$  from before.

