Artificial Intelligence Methods for Social Good

M4-3 [Sequential Decision Making]: Partially Observable Markov Decision Processes (POMDPs)

08-537 (9-unit) and 08-737 (12-unit)

Instructor: Fei Fang

feifang@cmu.edu

Wean Hall 4126

Outline

- Partially Observable Markov Decision Process (POMDP)
- Monte Carlo Tree Search (MCTS)
- Partially Observable Monte Carlo Planning (POMCP)
  - MCTS for POMDP

Learning Objective

- Understand the concept of
  - Partially Observable Markov Decision Process (POMDP)
  - Belief state
- Compute belief state distribution
- Construct belief-state MDP
- Describe
  - Monte Carlo Tree Search (MCTS)
  - Particle filtering

Recall: Markov Chain

- Markov Chain definition
  - **S:** set of states, $s_t \in S$
  - **T**ransition function (Markov property): $P(s_{t+1}|s_t)$

Recall: Markov Decision Process

- MDP definition
  - **S:** set of states, $s_t \in S$
  - **A:** set of actions, $a_t \in A$
  - **T**ransition function (Markov property): $P(s_{t+1}|s_t, a_t)$
  - **R**eward function $r_t = R(s_t, a_t), \gamma \in [0, 1]$

Recall: Hidden Markov Model

- HMM definition
  - **S:** set of states, $s_t \in S$
  - **T**ransition function (Markov property): $P(s_{t+1}|s_t)$
  - $\boldsymbol{b_0}$: Initial state distribution, i.e., $P(s_0)$
  - **O**: Observation likelihoods / Emission probabilities: $P(o_t|s_t)$ with $o_t \in O$

Partially Observable Markov Decision Process

- POMDP definition
  - **S:** set of states, $s_t \in S$
  - **A:** set of actions, $a_t \in A$
  - **T**ransition function (Markov property): $P(s_{t+1}|s_t, a_t)$
  - **R**eward function $r_t = R(s_t, a_t), \gamma \in [0, 1]$
  - $\boldsymbol{b_0}$: Initial state distribution, i.e., $P(s_0)$
  - **O**: Observation likelihoods / Emission probabilities: $P(o_t|s_t)$ with $o_t \in O$

Belief Update

- $b_t(s)$: probability of $s_t = s$
- Updated using Bayesian Rule given action a_t and observation o_{t+1} in the next time step
- $b_{t+1}(s') \propto p(o_{t+1}|s) \sum_s p(s'|s, a_t) b_t(s)$

- Exp 1

Quiz 1

- Transition graph same as Exp 1
  - $P(s^2|s^1, a^1) = 1, P(s^1|s^1, a^2) = 1$
  - $P(s^1|s^2, a^1) = 1, P(s^2|s^2, a^2) = 1$
- Emission probability
  - $P(o^1|s^1) = 1, P(o^2|s^2) = 1$
- $b_0 = [0.5,0.5]$
- What is b_1 given $a_0 = a^1$ and $o_1 = o^1$?
  - [1,0]
  - [0,1]
  - [0.5,0.5]
  - [0.25,0.75]

History and Policy

- History
  - $h_t = \{a_1, o_1, \ldots, a_t, o_t\}$
  - Sequence of actions and observations

- POMDP Policy
  - Option 1: define on belief state: $\mathrm{a} = \pi(b)$
    - Given b_0 and π, we can execute a POMDP for many steps, getting reward for every step
  - Option 2: define on history: $\mathrm{a} = \pi(h)$

POMDP as belief MDP

- POMDP can be converted into an MDP with belief state
- POMDP:
  - **S:** set of states, $s_t \in S$
  - **A:** set of actions, $a_t \in A$
  - **T**ransition function (Markov property): $P(s_{t+1}|s_t, a_t)$
  - **R**eward function $r_t = R(s_t, a_t), \gamma \in [0, 1]$
  - $\boldsymbol{b_0}$: Initial state distribution, i.e., $P(s_0)$
  - **O**: Observation likelihoods / Emission probabilities: $P(o_t|s_t)$ with $o_t \in O$
- Corresponding belief state MDP
  - State: Belief state b, set of states $\mathcal{B} \subset \mathbb{R}^{|S|}$
  - Action: $\mathrm{a_t} \in A$
  - Transition function: $P(b_{t+1}|b_t, a_t)$
  - Reward function: $r_t = \sum_{s_t} b(s_t)R\,(s_t, a_t)$
- Exp 1

Simple Solution to POMDP

- Simple solution
  - Construct belief state MDP
  - Discretize belief state space of the constructed MDP
  - Solve the MDP using value iteration, policy iteration or other MDP solving techniques
  - Map the solution back to POMDP
- Limitations
  - Curse of dimensionality: When $|S|$ is large, even a coarse discretization leads to a huge number of states!
  - Exp 1

Other Solutions to POMDP

- Exact solution approaches
  - Value iteration
  - Policy iteration
  - Intractable

- Online Planning approach
  - Point-Based Value Iteration
  - Branch and bound

Monte Carlo Tree Search

- General framework to make online decision in sequential decision making problems
  - E.g., online planning in MDPs, to determine game plays in Go, chess, video games etc

Monte Carlo Tree Search

- MCTS for single player setting: online planning in an unknown environment
- You are now in some state, need to choose an action, but you know nothing about the environment
- Helper: a simulator tells you your available actions, and reward after you take the action

Green player controlled by you
Yellow player controlled by some algorithm
Actions={up, down, nothing}

Monte Carlo Tree Search

- Build a search tree node by node
- Select → Expand → Simulate → Back propagate → Select → …

- Simplest MCTS
  - In each iteration
    - Select: Choose the branch with the highest value
    - Expand: Add one node by randomly selecting an action
    - Simulate: Uniform random rollout
    - Back propagate: update mean return (average accumulated reward) along the path
  - Output: action correspond to branch with highest value at the root node after K iterations

Monte Carlo Tree Search

- More advanced MCTS
  - Upper Confidence Bounds for Trees (UCT)
    - For each node, keep track of estimated action value and visit count: $Q(s,a)$ and $N(s,a)$
    - Select: Balance exploration vs exploitation:
      - If some actions never been chosen, randomly choose among them
      - Choose branch with highest augmented action value (also referred to as Upper Confidence Bounds (UCB)):

$$Q^{\oplus}(s,a) = Q(s,a) + c\sqrt{\frac{\ln N(s)}{N(s,a)}}$$

  - Other advanced options:
    - Simulate: Terminate after T_0 steps and estimate the reward
    - Expand: Add more nodes to he tree
    - Output: Optimal action at root node, as well as Q and N in the subtree corresponds to the optimal action
    - Initialize search tree with domain knowledge

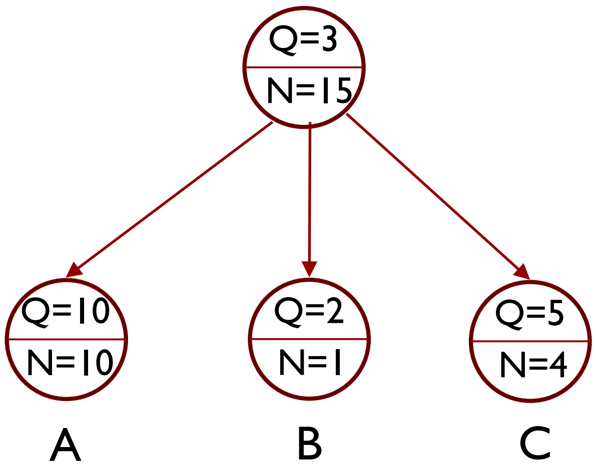
Monte Carlo Tree Search

- MCTS for multi-player setting: Tic-Tac-Toe

- Select
- Expand
- Simulate
- Back propagate

Quiz 2

- For the following tree, which leaf node will be expanded in UCT with $c = 1000$?



Partially Observable Monte Carlo Planning (POMCP)

- Apply MCTS to solve POMDP

- Partially Observable-UCT (PO-UCT)
  - Node in the search tree represent a history h
  - For each node, keep track of estimated history value $V(h)$ and visit count $N(h)$
  - Given belief state b, run one simulation
    - Select: sample initial state s, choose branch with highest

$$V^{\oplus}(h,a) = V(h,a) + c\sqrt{\frac{\ln N(h)}{N(h,a)}}$$

    - Expand: add a node
    - Simulate: Uniform random rollout
    - Back propagate: update $V(h)$
  - Output: Optimal action at root node, as well as V and N in the subtree corresponds to the optimal action

Online planning with PO-UCT

- In each time step:
  - Run PO-UCT, get optimal action a and V and N in the subtree correspond to a
  - Take optimal action a
  - Observe a real observation o
  - Update belief b
  - Initialize search tree for next time step with V and N in the subtree correspond to a

Monte Carlo Belief Update (Particle Filtering)

- Task: given b_t, a_t, o_{t+1}, (approximately) compute b_{t+1}
- Sample K states (particles) from initial state distribution b_t
- Set $B_{t+1}(s) = 0, \forall s$
- In each iteration
  - Randomly choose one particle to be s_t
  - Run simulation to get a sample successor state s' and sample observation o'
  - If $o' = o_{t+1}$, then add particle s' to the new state particles, i.e., $B_{t+1}(s') = B_{t+1}(s') + 1$
- Repeat until K particles are added to B_{t+1}
- Estimate b_{t+1} from B_{t+1} as $b_{t+1}(s) = \frac{B_{t+1}(s)}{\sum_{s'} B_{t+1}(s')}$

Partially Observable Monte Carlo Planning (POMCP)

- POMCP=PO-UCT + MC Belief Update with shared simulations
- For each node, keep track of estimated history value $V(h)$ and visit count $N(h)$, and also particles $B(h)$
  - Note that h encodes a and o
- During back propagation, update $B(h)$
- After the optimal action a is chosen, and the observation o is observed, search tree for next time step with belief state derived from $B(h)$ of the new root

Additional Resources

- *Planning and acting in partially observable stochastic domains*
- Leslie Pack Kaelbling, Michael L. Littman, Anthony R. Cassandra
- *Monte-Carlo Planning in Large POMDPs*
- David Silver, Joel Veness
- *Bandit based Monte-Carlo Planning*
- Levente Kocsis and Csaba Szepesvari