Swarmrobotics Workshop

MAXQ-Q Learning with MDLe
1. Introduction

What this lecture is about …

• Reinforcement learning
  – Hierarchical reinforcement learning
  – MAXQQ learning Algorithm
• MAXQ-Q learning with MDLe
  – Integration into MDLe
  – Needed Additions in MDLe
2. Reinforcement learning

Reinforcement Learning

- Agent learns a behaviour in a world
  - Unknown
  - Dynamic
  - Fully observable
- Learning through try and error
  - Looks at current state $s$
  - Performs action $a$
  - Looks at resulting state $s'$ and receives reward $r$
2. Reinforcement learning

Example of Reinforcement Learning

- **Actions**
  - Movement (N, E, S, W)
  - Pickup
  - Putdown

- **Rewards**
  - Action -1
  - Successful Putdown +20
  - Unsuccessful Putdown -10
  - Unsuccessful Pickup -10
2. Reinforcement learning

Reinforcement Learning

- **Exploration**
  - Tries action and gets Reward/Penalty
  - Better action selection in future
- **Exploitation**
  - Tries to maximize reward
- **Find an optimal policy**
  - Optimal value function
2. Reinforcement learning

Optimal value function

- Passenger at location Y (0,0) Destination B (3,0)
2. Reinforcement learning

Drawbacks of Reinforcement Learning

- Many training iterations are needed
  - >100000 for complicated tasks
  - Solutions:
    - Hierarchical Reinforcement Learning
    - Model Based Reinforcement Learning
- State space grows exponentially with state variables
  - Bad scaling
- Learned knowledge can’t be transferred to similar tasks
  - Hierarchical Reinforcement Learning
  - MAXQ value function decomposition
3. MAXQ

MAXQ learning

- Task decomposition  
  - Discover and exploit hierarchical structure  
  - Programmer defines hierarchy  
- Value Function Decomposition  
  - Value function of subtask + Completion function  
- State Abstraction  
  - Irrelevant variables  
  - Funnel abstractions  
  - Structural constraints
3. MAXQ

Task decomposition

- Root
  - Get
    - Pickup
      - North
  - Put
    - Navigate(t)
    - Putdown
      - East
      - West
MAXQ value function decomposition

\[ V(root, s) = V(west, s) + C(navigate(Y), s, west) + C(get, s, navigate(Y)) + C(root, s, get). \]
3. MAXQ

MAXQQ learning

Function MAXQQ(state s, subtask p) returns float

Let TotalReward = 0

while p is not terminated do

    Choose and execute action a

    if a is primitive
        Observe one-step reward r
    else
        r := MAXQQ(s,a), invokes subroutine a and returns total reward received during a

    TotalReward := TotalReward + r

    if a is a primitive
        \[ V(a, s) = (1 - \alpha)V(a, s) + \alpha r \]

    else
        a is a subroutine
        \[ C'(p, a, s) := (1 - \alpha)C(p, s, a) + \alpha \max_{a'}[V(a', s') + C(p, s', a')] \]

end // while

return TotalReward

end
3. MAXQ

MAXQ learning

\[
V(a, s) = (1 - \alpha)V(a, s) + \alpha r
\]

\[
C(p, a, s) := (1 - \alpha)C'(p, s, a) + \alpha \max_{a'}[V(a', s') + C'(p, s', a')]
\]
4. MAXQ with MDLe

Integration in MDLe

Root

Get

Put

Navigate(t)

Pickup

North

East

South

West

Putdown
4. MAXQ with MDLe

Implementation in MDLe

- RUNION
  - Goal State
  - Temperature (exploration vs exploitation)
  - $C(p,s,a)$
- ATOM
  - Reward
  - $V(s)$
  - $V(a,s)$
4. MAXQ with MDLe

Integration in MDLe

- Correct order of sequence
- Reward propagation
- State representation