

## Scope

- Designed Markov Decision Process (MDP) framework using Dynamic Programming (DP) in satellite network control domain.
- For the simplicity of the problem formulation, we have supposed single ground station to single ground station only.
- Both utilized value iteration method and policy iteration method.

## I. MDP Formulation

For creating MDP framework environment, we need 5 necessary variable spaces  $M = (S, A, P, R, \gamma)$  and policy  $\pi$ . We are simulating the network structure, transmitting one data packet from Ground Station m to Ground station n by using satellite network (SAT1 - SAT48)

1.  $S$  : Finite **Set** of States - Location of the data packet. Total 78 States (30 GS + 48 SATS)
2.  $A$  : Finite **Set** of Actions - Ordering the agents (GS + SAT) to mover the data packet from one place to another (GS  $\rightarrow$  SAT, SAT  $\rightarrow$  SAT, SAT  $\rightarrow$  GS)
3.  $P$  : State Trasmission Probability **Matrix**- Probability when the data is successfully transmitted by given action. (e.g. 0.8 indicates that failure of data transmission in 20% occation)
4.  $R$  : Reward **Function** - Reward when the data is moved to certain agent. GS  $\rightarrow$  SAT / SAT  $\rightarrow$  SAT: minus reward. Only positive reward is when the data is transmitted to final destination SAT which can directly transmit to GS.
5.  $\gamma$  : Discount **Factor** - Discount factor of the reward in order to sum of series of rewards are converged.
6.  $\pi(a | s)$  : Policy probability - Wheter deterministic  $\pi = 1$  for optimal and 0 for all other case or stocastic  $\pi \sim (0, 1)$

Caution: The Satellites and Ground Stations themselves are not the states. Instead, they are agents and formulate the grid area. The phenomenons that the data is located at each agent are states.

## II. Review of basic MDP Theory

We have to define  $M = (S, A, P, R, \gamma)$  from given dataset. In order to do that, we have to understand the basic framework of the process of MDP simulation, include knowing know to define each variable.

## II.1 State Transition Probability Matrix $P$ using $S, A$

$$P_{ss'}^a = p(S_{t+1} = s' | S_t = s, A_t = a)$$

- Currently, each states may be connected to other state by network graph. The created network graph is composed by 0,1 - 0 indicates no connection and 1 indicates connection.
- If the probability is 0.8, the 0.2 probability should be returning to current state.
- The actions should be deciding the probability to transmute to other states (then where?) or staying in current state.
- There are different scale of action for different states because the number of satellites or ground stations connected to each agent should be different.

## II.2 Policies $\pi$ : Mapping from State to Actions

$$\pi(a|s) = p(A_t = a | S_t = s)$$

- Initially, we assume there is same possibility for all available actions, then converge over the **policy iteration** process.
- In value iteration process, we don't have to consider policy before all the state value functions are converged.
- Also, we have to choose between deterministic of stochastic policy

## II.3 Value Function

**State-value function**  $v_\pi(s)$ : Expected return starting from state  $s$ , in the case following policy  $\pi$  after that.

$$v_\pi(s) = E_\pi[G_t | S_t = s] = E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s]$$

- Another Expression [Sutton]

$$\begin{aligned} v_\pi(s) &= E_\pi[G_t | S_t = s] = E_\pi \left[ \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s \right] = E_\pi \left[ R_{t+1} + \sum_{k=0}^{\infty} \gamma^k R_{t+k+2} | S_t = s \right] \\ &= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r | s, a) \left[ r + \gamma E_\pi \left[ \sum_{k=0}^{\infty} \gamma^k R_{t+k+2} | S_{t+1} = s' \right] \right] \\ &= \sum_a \pi(a|s) \sum_{s', r} p(s', r | s, a) [r + \gamma v_\pi(s')] \rightarrow \text{Bellman Equation for } v_\pi \end{aligned}$$

- It is really a sum over all values of the three variables,  $a, s'$ , and  $r$ .
- For each triple, we compute its probability,  $\pi(a|s) p(s', r | s, a)$ , weight the quantity in brackets by that probability in brackets by that probability, then sum over all possibilities to get an expected value.

**Action-value function**  $q_\pi(s, a)$ : Expected return starting from state  $s$  and action  $a$ , in the case following policy  $\pi$  after that.

- $q_{\pi}(s, a) = E_{\pi}[G_t | S_t = s, A_t = a] = E_{\pi}[R_{t+1} + \gamma q_{\pi}(S_{t+1}) | S_t = s, A_t = a]$

## II.4 Bellman Expectation Equation

**Relationship between  $v_{\pi}(s)$  and  $q_{\pi}(s, a)$  and Reward function  $R(s, a)$**

- $$v_{\pi}(s) = \sum_{a \in A} \pi(a|s) q_{\pi}(s, a)$$
- $$q_{\pi}(s, a) = R(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) v_{\pi}(s')$$
- $R_s^a = E[R_{t+1} | S_t = s, A_t = a]$

**Bellman expectation equation derivation for  $v_{\pi}(s)$  and  $q_{\pi}(s, a)$**

- For deterministic optimal policy,
- $$v_{\pi}(s) = \sum_{a \in A} \pi(a|s) q_{\pi}(s, a) = \sum_{a \in A} \pi(a|s) \left[ R(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) v_{\pi}(s') \right]$$
- $$q_{\pi}(s, a) = R(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) v_{\pi}(s') = R(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) \left[ \sum_{a' \in A} \pi(a'|s') q_{\pi}(s', a') \right]$$

## II.5 Bellman Optimality Equation

**Optimal state-value function  $v_*(s)$ :** Maximum value function for all policies

- $$\begin{aligned} v_*(s) &= \max_{\pi} v_{\pi}(s) = \max_a q_*(s, a) \\ &= \max_a \left( R(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) v_*(s') \right) \\ &= R(s, a) + \gamma \max_a \sum_{s' \in S} p(s'|s, a) v_*(s') \end{aligned}$$

- Another Expression[Sutton]: Last two equations are two forms of the Bellman optimality equation for  $v_*$

- $$\begin{aligned}
v_*(s) &= \max_{a \in A(s)} q_{\pi_*}(s, a) = \max_a E_{\pi_*}[G_t \mid S_t = s, A_t = a] \\
&= \max_a E_{\pi_*} \left[ \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a \right] \\
&= \max_a E_{\pi_*} \left[ R_{t+1} + \sum_{k=0}^{\infty} \gamma^k R_{t+k+2} \mid S_t = s, A_t = a \right] \\
&= \max_a E[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a] \\
&= \max_{a \in A(s)} \sum_{s', r} p(s', r \mid s, a) [r + \gamma v_*(s')]
\end{aligned}$$

**Optimal action-value equation**  $q_*(s, a)$ : Maximum action value function for all policies

- $$\begin{aligned}
q_*(s, a) &= \max_{\pi} q_{\pi}(s, a) \\
&= \max_a \left( R(s, a) + \gamma \sum_{s' \in S} p(s' \mid s, a) v_{\pi}(s') \right) \\
&= R(s, a) + \gamma \sum_{s' \in S} p(s' \mid s, a) \max_{\pi} v_{\pi}(s') \\
&= R(s, a) + \gamma \sum_{s' \in S} p(s' \mid s, a) v_*(s') \\
&= R(s, a) + \gamma \sum_{s' \in S} p(s' \mid s, a) \max_{a'} q_*(s', a')
\end{aligned}$$

- Another Expression[Sutton]

- $$\begin{aligned}
q_*(s, a) &= E \left[ R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \mid S_t = s, A_t = a \right] \\
&= \sum_{s', r} p(s', r \mid s, a) \left[ r + \gamma \max_{a'} q_*(s', a') \right]
\end{aligned}$$

## II.6 Solving the Bellman Optimality Equation

We can directly define the optimal value function using Bellman optimality equation.

- $$v_*(s) = \max_a \left( R(s) + \gamma \sum_{s' \in S} p(s'|s, a) v_\pi(s') \right)$$

$$= R(s) + \gamma \max_a \sum_{s' \in S} p(s'|s, a) v_*(s')$$

$$q_*(s, a) = R(s) + \gamma \sum_{s' \in S} p(s'|s, a) v_*(s')$$

And optimal policy is simply the action that attains this max

- $$\pi_*(s) = \arg \max_a \sum_{s' \in S} p(s'|s, a) v_*(s')$$
- For deterministic optimal policy,  $\pi_*(a|s) = \begin{cases} 1 & \text{if } a = \arg \max_{a \in A} q_*(s, a) \\ 0 & \text{otherwise} \end{cases}$

Bellman optimality equation is non-linear, and there is no closed form in general. So there are many iterative solution methods including Value Iteration, Policy Iteration, Q-learning and SARSA.

## II.7 Value Iteration

### Algorithm

1. Initialize an estimate for the value function arbitrarily (or zeros):  $v(s) \leftarrow 0 \quad \forall s \in S$
2. Repeat, update  $v(s)$ :  $v(s) \leftarrow R(s) + \gamma \max_a \sum_{s' \in S} p(s'|s, a) v_*(s'), \forall s \in S$
3. Output a deterministic policy,  $\pi$  such that  $\pi(s) = \arg \max_a \sum_{s', r} p(s', r | s, a) [r + \gamma v(s')]$

## II.8 Policy Iteration

- Given a policy  $\pi$ , then evaluate the policy  $\pi$
- Improve the policy by acting greedy with respect to  $v_\pi$
- Policy iteration requires fewer iterations than value iteration, but each iteration requires solving a linear system instead of just applying bellman operator.
- In practice, policy iteration is often faster, especially if the transition probabilities are structured (e.g. sparse) to make solution of linear system different.

### Algorithm

1. Initialize policy  $\hat{\pi}$  (e.g. randomly)
2. Compute a value function of policy,  $v_{\pi}$  (e.g. via solving linear system or Bellman expectation equation iteratively).
3. Update  $\pi$  to be greedy policy with respect to  $v_{\pi}$ :  $\pi(s) \leftarrow \arg \max_a \sum_{s' \in S} p(s'|s, a) v_{\pi}(s')$
4. If policy  $\pi$  changed in last iteration, return to step 2.

### [Another Approach from Sutton]

#### Policy Evaluation

First we consider how to compute the state-value function  $v_{\pi}$  for an arbitrary policy  $\pi$ . This is called policy evaluation in the DP literature.

$$\begin{aligned}
 \bullet \quad v_{\pi}(s) &= E_{\pi}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots | S_t = s] \\
 &= E_{\pi}[R_{t+1} + \gamma v_{\pi}(S_{t+1}) | S_t = s] \\
 &= \sum_a \pi(a|s) \sum_{s', r} p(s', r | s, a) [r + \gamma v_{\pi}(s')]
 \end{aligned}$$

If the environment's dynamics are completely known then given equation is a system of  $|S|$  simultaneous linear equations in  $|S|$  unknowns (the  $v_{\pi}(s), s \in S$ ). Consider a sequence of approximate value functions  $v_0, v_1, v_2, \dots$  each mapping  $S \rightarrow R$ . Initial approximation  $v_0$  is chosen arbitrarily and each successive approximation is obtained by using the Bellman equation for  $v_{\pi}$  as an update rule.

$$\begin{aligned}
 \bullet \quad v_{k+1}(s) &= E_{\pi}[R_{t+1} + \gamma v_k(S_{t+1}) | S_t = s] \\
 &= \sum_a \pi(a|s) \sum_{s', r} p(s', r | s, a) [r + \gamma v_k(s')]
 \end{aligned}$$

#### Policy Improvement

We know how good it is to follow the current policy from  $s$  - that is  $v_{\pi}(s)$  - but would it be better or worse to change to the new policy? One way to answer this question is to consider selecting  $a$  in  $s$  and thereafter following the existing policy  $\pi$ .

The value of this way of behaving is:

$$\begin{aligned}
 \bullet \quad q_{\pi}(s, a) &= E_{\pi}[R_{t+1} + \gamma v_{\pi}(S_{t+1}) | S_t = s, A_t = a] \\
 &= \sum_{s', r} p(s', r | s, a) [r + \gamma v_{\pi}(s')]
 \end{aligned}$$

The key criterion is whether this is greater than or less than  $v_{\pi}(s)$ . If it is greater-that is, if it is better to select  $a$  one in  $s$  and thereafter follow  $\pi$  than it would be to follow  $\pi$  all the time-then one would expect it to be better still to select  $a$  every time  $s$  is encountered, and that the new policy would in fact be a better one overall.

that this is true is a special case of a general result called the policy improvement theorem. Let  $\pi$  and  $\pi'$  be any pair of deterministic policies such that, for all  $s \in S$ .

- $q_\pi(s, \pi'(s)) \geq v_\pi(s)$

Then the policy  $\pi'$  must be good as good as, or better than,  $\pi$ . That is, it must obtain greater or equal return from all states  $s \in S$ .

- $v_{\pi'}(s) \geq v_\pi(s)$

So far we have seen how, given a policy and its value function, we can easily evaluate a change in the policy at a single state to a particular action. It is a natural extension to consider changes at all states and to all possible actions, selecting at each state the action that appears best according to  $q_\pi(s, a)$ .

In other words, to consider the new greedy policy  $\pi'$  given by:

- $$\begin{aligned} \pi'(s) &= \arg \max_a q_\pi(s, a) \\ &= \arg \max_a E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = a] \\ &= \arg \max_a \sum_{s', r} p(s', r \mid s, a) [r + \gamma v_\pi(s')] \end{aligned}$$

where  $\arg \max_a$  denotes the value of  $a$  at which the expression that follows is maximized.

Suppose the new greedy policy,  $\pi'$ , is as good as, but not better than the old policy  $\pi$ . Then  $v_\pi = v_{\pi'}$ , and it follows that for all  $s \in S$ :

- $$\begin{aligned} v_{\pi'}(s) &= \max_a E[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = a] \\ &= \max_a \sum_{s', r} p(s', r \mid s, a) [r + \gamma v_\pi(s')] \end{aligned}$$

But this is the same as the Bellman optimality equation, and therefore,  $v_{\pi'}$  must be  $v_*$ , and both  $\pi$  and  $\pi'$  must be optimal policies. Policy improvement thus give us a strictly better policy except when the original policy is already optimal.

In case of stochastic policy  $\pi$ , the policy improvement theorem carries through as stated for the stochastic case, using the natural definition

- $q_\pi(s, \pi'(s)) = \sum_a \pi'(a \mid s) q_\pi(s, a)$

## Policy Iteration

Once a policy  $\pi$  has been improved using  $v_\pi$  to yield a better policy  $\pi'$ , we can then compute  $v_{\pi'}$  and improve it again to yield an even better  $\pi''$ . We can thus obtain a sequence of monotonically improving policies and value functions.

$\pi_0 \rightarrow \text{evaluation} \rightarrow v_{\pi_0} \rightarrow \text{improvement} \rightarrow \pi_1 \rightarrow \text{evaluation} \rightarrow v_{\pi_1} \rightarrow \text{improvement} \rightarrow \dots \rightarrow \pi_* \rightarrow \text{evaluation} \rightarrow v_*$

### III. MDP application on Satellite Network System