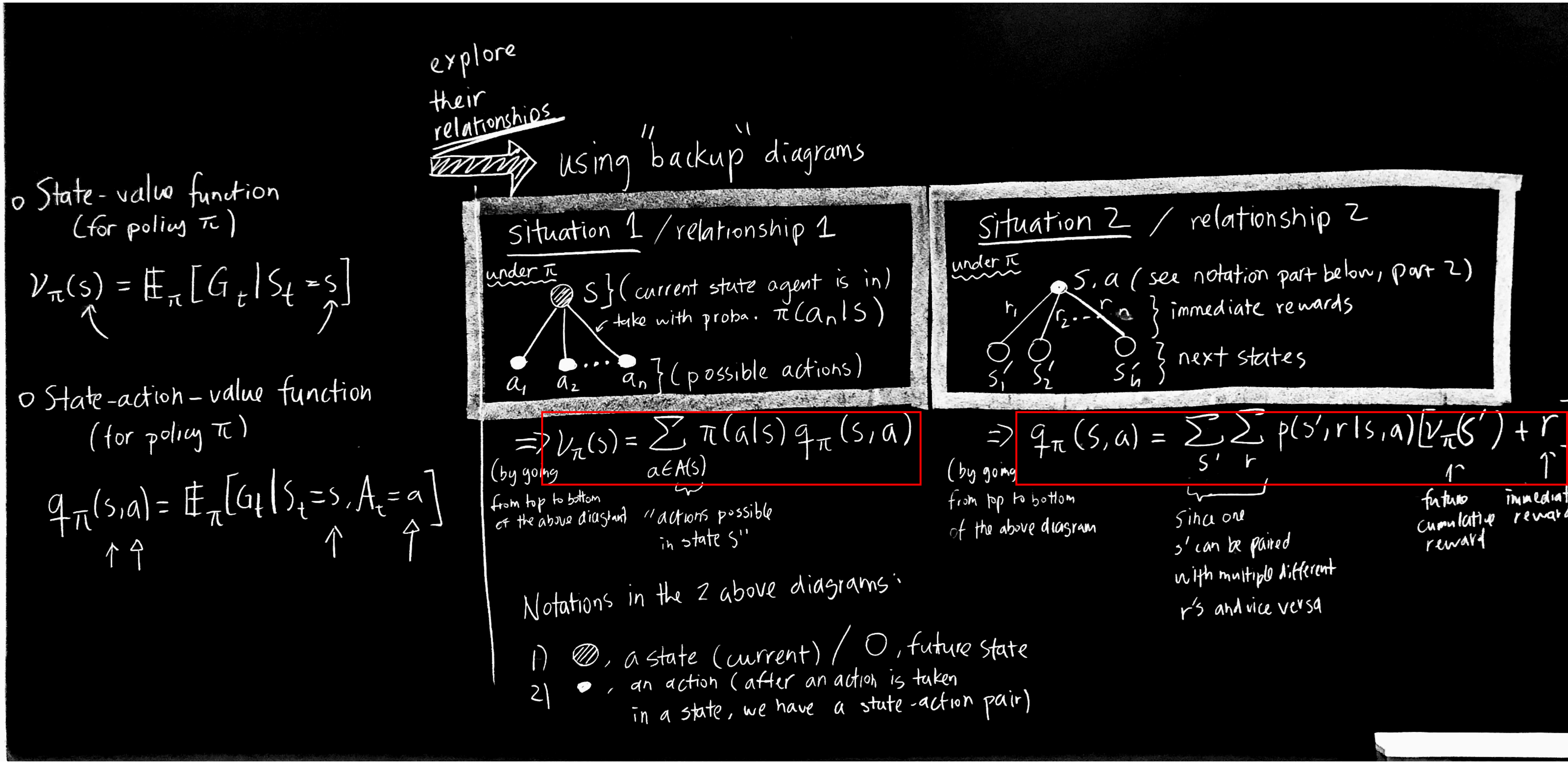
Motivation of Dynamic Programming Chapter

Everything in the chapter serves to answer this question:

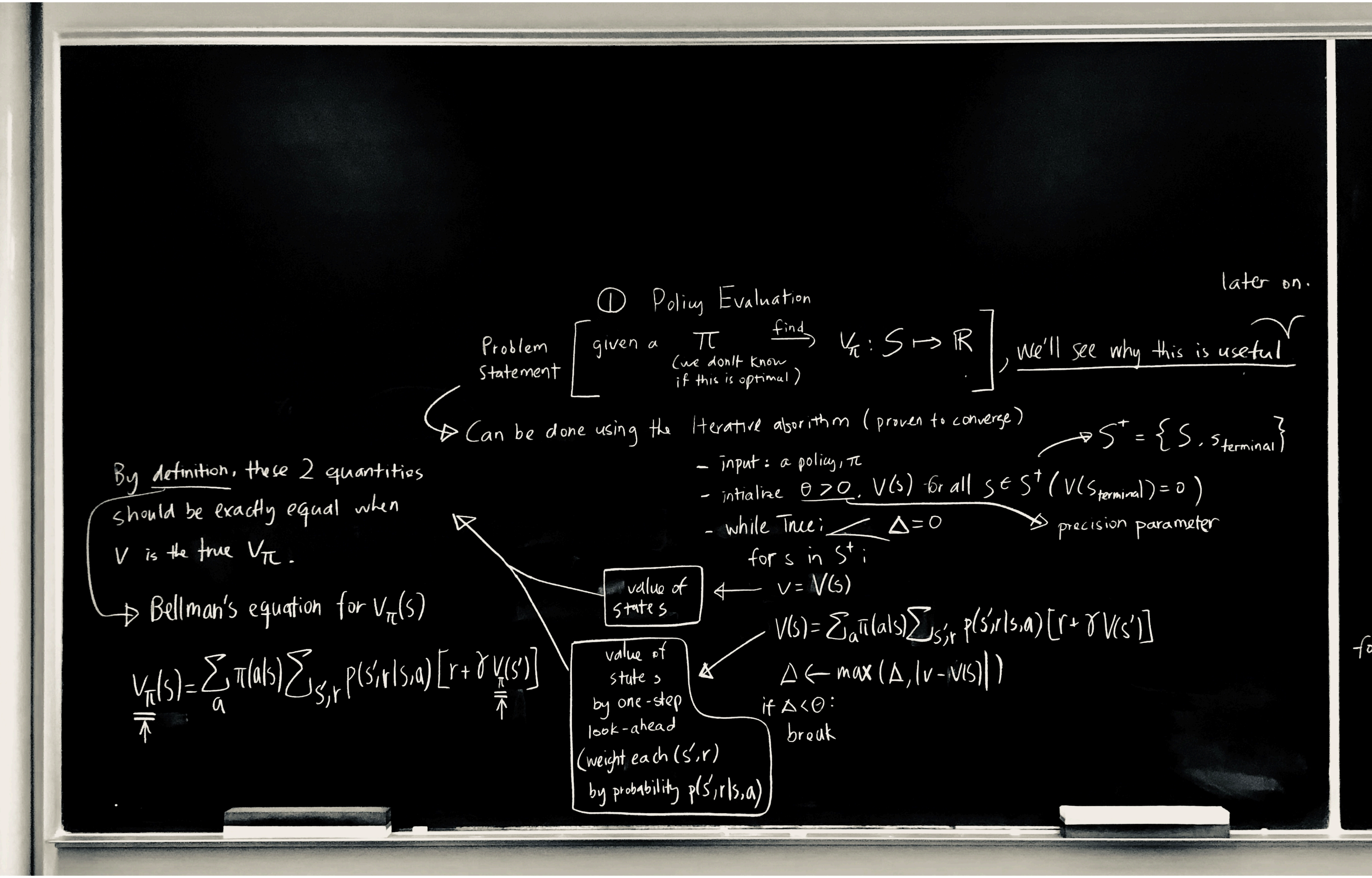
given the full knowledge of an MDP (the entire table of $p(s', r | s, a)$),
how to find the optimal policy π that maximizes the cumulative reward?

Understand the connection between state-value function v(s) and action-value function q(s, a)



We have found the true value of all states or all state-action pairs when the two equations in red boxes are true for all states / all state-action pairs. Both of these come from the definition of "value", the cumulative reward onwards. For example, the equation in the left red box describes that the cumulative reward from s onwards should equal the weighted sum of the values of possible actions (when the agent is in that state). This definition is justified by our daily experience as agents.

Policy Evaluation Algorithm For Full-MDP



Policy Improvement

Sutton 2018, Chapter 4, Dynamic Programming

Zhihan Yang

Policy improvement algorithm

The policy improvement algorithm happens after each iteration of policy iteration and can be summed up very concisely:

1. For each accessible state, sum up the immediate reward of arriving in that state and the value of that state (expected cumulative reward of being in that state and onwards).
2. Choose the action that take yourself to the accessible state with the highest sum. This is called **greedy action selection**.

However, the most important question is why this simple algorithm works; we'll answer this by proving the convergence of this algorithm (to the optimal value function and the optimal policy) when used in combination with policy evaluation (which we discussed in the previous session).

Proof

The second step of the algorithm updates the old policy such that now the new policy $\pi(a|s)$ gives the highest probability to the action a that maximizes $q_\pi(s,a)$ - the highest value of $\pi(a|s)$ and the highest value of $q_\pi(s,a)$ now occur for the same a. By understanding this alignment and inspecting equation 1 (expressing v_π in terms of q_π), it becomes clear why $V_{\text{new}}(s) \geq V_{\text{old}}(s)$.

(Recall that the value of a state v_π is related to the value of its accessible states $q_\pi(s,a)$ by the following relationship.)

$$v_\pi(s) = \sum_a \pi(a|s) q_\pi(s,a) \tag{1}$$

There are two cases that allow $V_{\text{new}}(s) \geq V_{\text{old}}(s)$:

1. $V_{\text{new}}(s) = V_{\text{old}}(s)$
   - This is the Bellman's optimality equation, which indicates that both V are already optimal.
2. $V_{\text{new}}(s) > V_{\text{old}}(s)$
   - This is what happens otherwise.

Therefore, we see that V improves unless it is already optimal.