Module 10
Online Optimization

CS 886 Sequential Decision Making and Reinforcement Learning

University of Waterloo

Policy Optimization

- Offline optimization:
  - Value iteration, policy iteration, linear programming
  - If initial state known, focus on reachable states

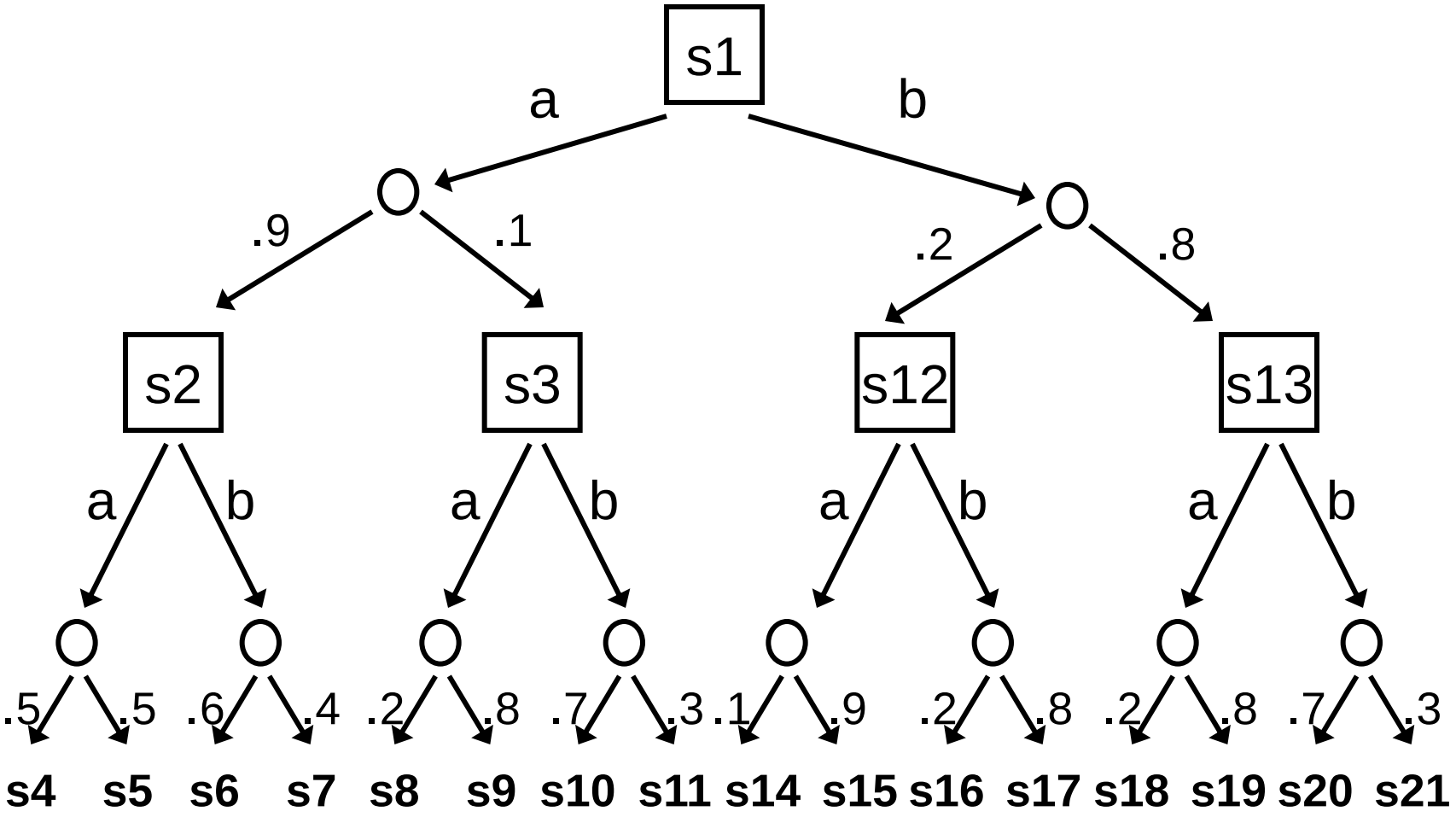
- Instead of pre-computing a full policy, could we optimize a policy as we execute it?
  - Yes: online optimization
  - At each step
    - Optimize only the next action
    - Use current state to focus computation on reachable states

Online Policy Optimization

- Idea: alternate between
  - Optimization of next action
    - Forward search from current state
  - Execution of action

- Forward Search
  - Branch and bound
  - Monte Carlo simulation

Expectimax Search Tree

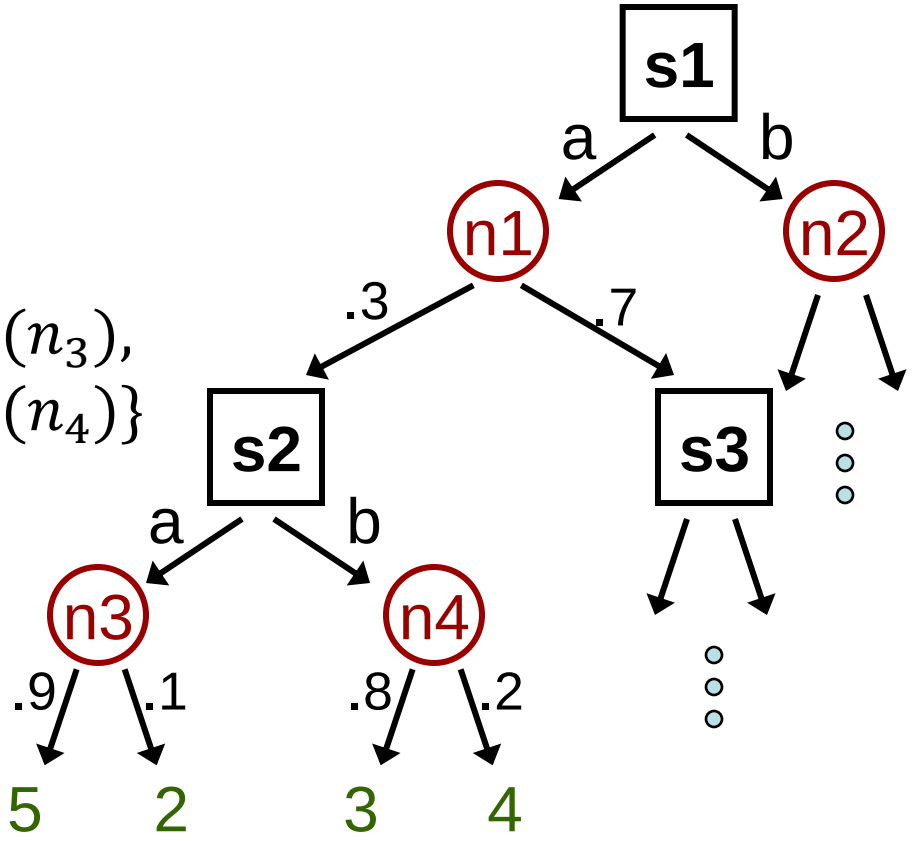


Expectimax Computation

- $V(n_3) = 0.9 \times 5 + 0.1 \times 2$
- $V(n_4) = 0.8 \times 3 + 0.2 \times 4$

- $V(s_2) = \max\{R(a, s_2) + \gamma V(n_3), R(b, s_2) + \gamma V(n_4)\}$

- $V(n_1) = .3V(s_2) + .7V(s_3)$



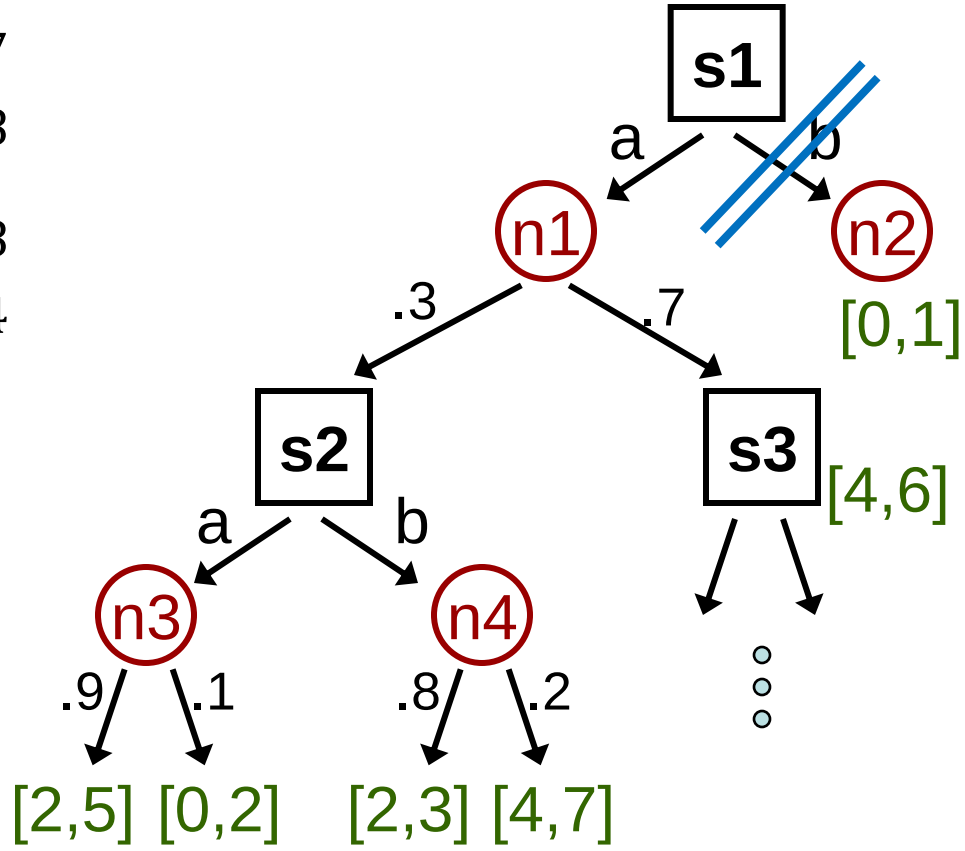
- $V(s_1) = \max\{R(a, s_1) + \gamma V(n_1), R(b, s_1) + \gamma V(n_2)\}$

Branch and Bound

- Use lower and upper bounds to prune branches
  - If $LB(s, a_1) > UB(s, a_2)$, then prune subtree under a_2
- Lower bounds:
  - Default bound: $\min_{s,a} R(s,a)/(1-\gamma) \leq V^*(s)\ \forall s$
  - Value of any policy π: $V^{\pi}(s) \leq V^*(s)\ \forall s$
- Upper bounds
  - Default bound: $\max_{s,a} R(s,a)/(1-\gamma) \geq V^*(s)\ \forall s$
  - Heuristic function: $h(s) \geq V^*(s)\ \forall s$
  - Value found by approximate linear programming

Branch and Bound

- $\bar{V}(n_3) = 0.9 \times 5 + 0.1 \times 2 = 4.7$
- $\underline{V}(n_3) = 0.9 \times 2 + 0.1 \times 0 = 1.8$
- $\bar{V}(n_4) = 0.8 \times 3 + 0.2 \times 7 = 3.8$
- $\underline{V}(n_4) = 0.8 \times 2 + 0.2 \times 4 = 2.4$
- $\bar{V}(s_2) = \max\{4.7, 3.8\} = 4.7$
- $\underline{V}(s_2) = \max\{1.8, 2.4\} = 2.4$
- $\bar{V}(n_1) = .3(4.7) + .7(6) = 5.61$
- $\underline{V}(n_1) = .3(2.4) + .7(4) = 3.52$



- Since $\underline{V}(n_1) > \bar{V}(n_2)$ prune subtree at n_2

Branch and Bound

Branch&Bound(s,n)

If $n = 0$

return $[lb(s), ub(s)]$

Else

$\underline{V}(s) \leftarrow lb(s), \quad \bar{V}(s) \leftarrow ub(s)$

For each $a \in A$ do

If $ub(s,a) \geq \underline{V}(s)$

For each s' such that $\Pr(s'|s,a) > 0$ do

$\left[\underline{V}(s'), \bar{V}(s')\right] \leftarrow Branch\&Bound(s', n-1)$

$\underline{V}(s) \leftarrow \max\{\underline{V}(s),\ R(s,a) + \gamma \Pr(s'|s,a)\, \underline{V}(s')\}$

$\bar{V}(s) \leftarrow \max\{\bar{V}(s),\ R(s,a) + \gamma \Pr(s'|s,a)\, \bar{V}(s')\}$

return $[\underline{V}(s), \bar{V}(s)]$

Branching Factor

- The action branching factor can be reduced by branch and bound.

- What about the observation branching factor?
- Can we reduce it?
  - Yes: sparse sampling

Sparse Sampling

- Idea: approximate expectations by k samples

$$\sum_{s'} \Pr(s'|s,a)\, V(s') \approx \frac{1}{k} \sum_{i=0}^{k} V(s_i')$$

where $s_i' \sim \Pr(s'|s,a)$

Sparse Sampling

SparseSampling(s,n)

If $n = 0$

return $\tilde{V}(s)$ ($\tilde{V}$ is an initial estimate)

Else

$V(s) \leftarrow -\infty$

For each $a \in A$ do

$Q(s,a) \leftarrow R(s,a)$

For $i = 0$ to k do

${s_i}' \sim \Pr(s'|s,a)$

$Q(s,a) \leftarrow Q(s,a) + \gamma SparseSampling({s_i}', n-1)/k$

$V(s) \leftarrow \max\{V(s), Q(s,a)\}$

return $V(s)$

Analysis

- Sparse sampling:
  - Find optimal action with long enough planning horizon and large enough sample size
  - Complexity exponential in planning horizon
  - Complexity independent of $|\mathcal{S}|$
  - Can tackle continuous state MDPs

- To mitigate the number of actions and states, combine branch&bound with sparse sampling.