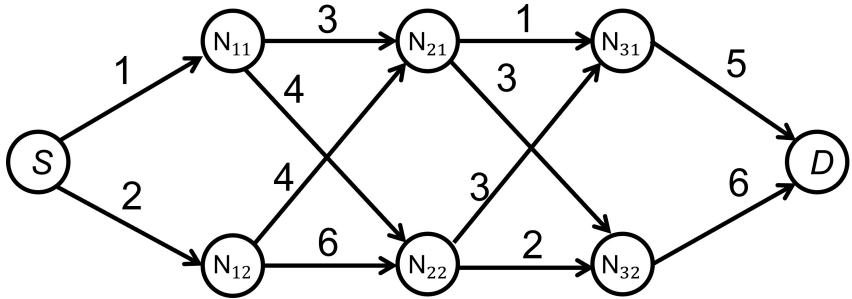
Stochastic Dynamic Programming (DP)[1]

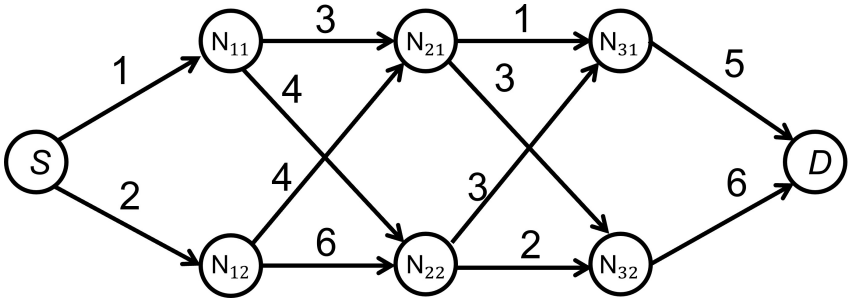
[1] Sections 1.4-1.6: Yang&Ying

Finite-Horizon Stochastic DP



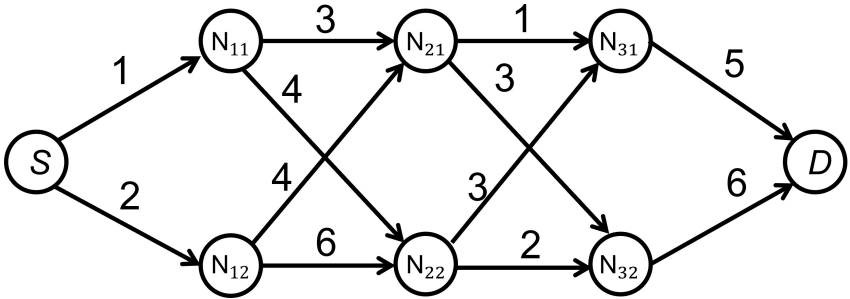
$u_k = \text{up} \rightarrow \begin{cases} P(x_{k+1} = \text{top}) = 0.6 \\ P(x_{k+1} = \text{bottom}) = 0.4 \end{cases}$ and $u_k = \text{down} \rightarrow \begin{cases} P(x_{k+1} = \text{top}) = 0.4 \\ P(x_{k+1} = \text{bottom}) = 0.6 \end{cases}$

Stochastic DP example



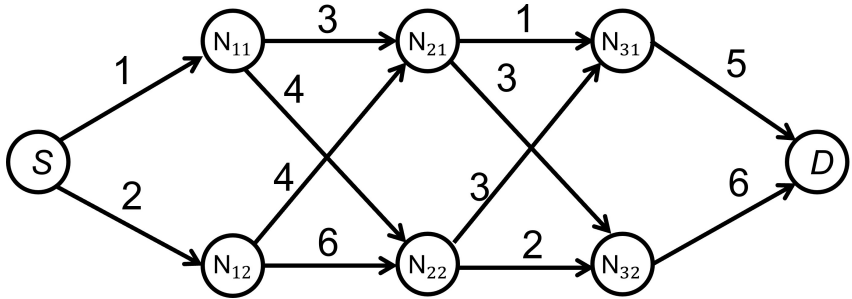
Optimal policy: A fixed sequence $(u_0, u_1, \ldots, u_{N-1})$?

Stochastic DP example



Optimal policy: A fixed sequence $(u_0, u_1, \ldots, u_{N-1})$?
NO.

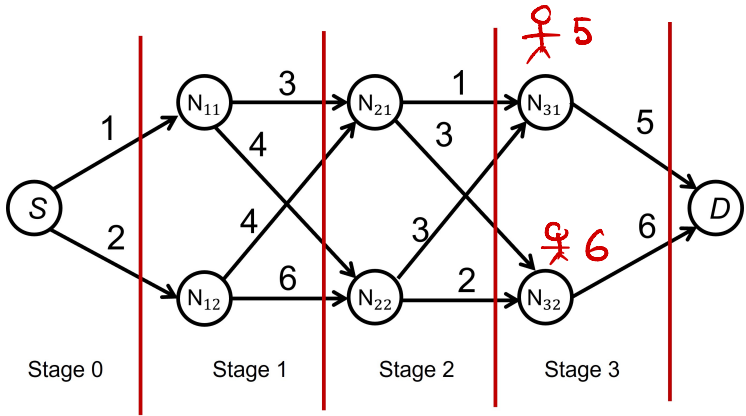
Stochastic DP example



Optimal policy: A fixed sequence $(u_0, u_1, \ldots, u_{N-1})$?

State dependent actions (a policy): $\mu_k^*(x_k)$, $\forall x_k$

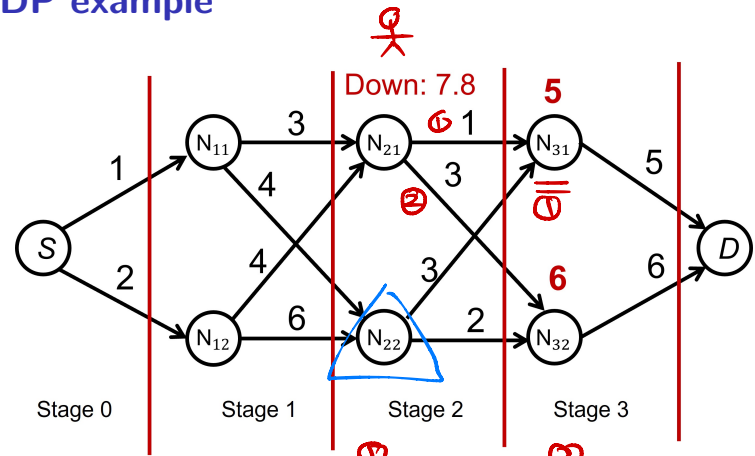
Stochastic DP example



A node at stage 3 has only one possible path to reach destination D.

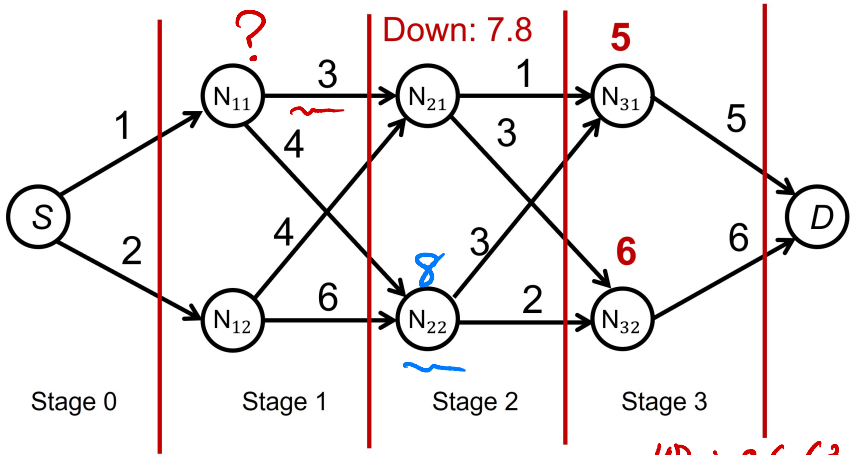
- Reward-to-go from the top node: 5
- Reward-to-go from the bottom node: 6

Stochastic DP example



- Reward-to-go with action Up: $0.6 \times (1+5) + 0.4 \times (3+6) = 3.6 + 3.6 = 7.2$
- Reward-to-go with action Down: $0.4 \times (1+5) + 0.6 \times (3+6) = 7.8$
- $u_2^*(N_{21}) =$ **down**

Stochastic DP example



up : 0.6 (3+7.8)
+ 0.4 (4+8)

- Reward-to-go (Up): $0.6 \times (3 + 5) + 0.4 \times (2 + 6) = 8$
- Reward-to-go (Down): $0.4 \times 8 + 0.6 \times 8 = 8$
- $u_2^*(N_{22})$ can be either up or down

Stochastic DP (finite-horizon)

- Random transitions $P(x_{k+1}|x_k, u_k)$
- Value function:

$$E\left[r_T(x_T) + \sum_{k=0}^{T-1} r_k(x_k, u_k)\right]$$

terminal reward

- Given initial state x_0, find a policy

$$\pi = \{\mu_0, \mu_1(x_1), \ldots, \mu_{T-1}(x_{T-1})\}$$

such that

$$V(x_0) = \max_{\pi} E\left[r_T(x_T) + \sum_{k=0}^{T-1} r_k(x_k, \mu_k(x_k))\right]$$

Stochastic DP

The Bellman Equation:

$$V_k^*(x_k) = \max_{u_k} E\left[r_k(x_k, u_k) + V_{k+1}^*(x_{k+1})\right]$$

$$= \max_{u_k} \sum_r rP(r_k = r|x_k, u_k) + \sum_{x \in X} V_{k+1}^*(x) P(x_{k+1} = x|x_k, u_k)$$

Given state x_k and $V_{k+1}^*(x_{k+1})$,

$$u_k^* \in \arg\max_{u_k} E[r_k(x_k, u_k) + V_{k+1}^*(x_{k+1})]$$

Backward-forward algorithm: First compute V_k^* backward and then find u_k^* forward.

Example: A Betting Game

You have \$2 and have to play a game 3 times. For each game, the chance of winning is 0.4 and the money you bet is doubled, and chance of losing is 0.6 and you lose the money you bet. You can only bet an integer dollar amount.

Goal: Find a policy that maximizes your chance of ending up with \$4 or more.

stage ? state ? action ? reward ?

Example: A Betting Game

You have \$2 and have to play a game 3 times. For each game, the chance of winning is 0.4 and the money you bet is doubled, and chance of losing is 0.6 and you lose the money you bet. You can only bet an integer dollar amount.

Goal: Find a policy that maximizes your chance of ending up with \$4 or more.

DP formulation

- stage i: the ith game (for $i = 0, 1, 2, 3$)
- state at stage i: $x_i =$ money available at the beginning of game i
- action at stage i: $u_i =$ how much to bet (must be an integer)
- reward at stage i: when $i = 3$, $r_3(x_3) = \begin{cases} 1, & x_3 \geq 4 \\ 0, & \text{otherwise} \end{cases}$

  when $i = 0, 1, 2$, $r_i(x_i) = 0$
- Objective: $V_0^*(x_0 = 2) = \max_{\mu_k} E\left[\sum_{k=0}^{3} r_k(x_k) \middle| x_0 = 2\right]$.

Example: A Betting Game

The Bellman Equation:

$$
\begin{aligned}
V_k^*(x_k) &= \max_u E[V_{k+1}^*(x_{k+1})] \\
&= \max_u \left(V_{k+1}^*(x_k + u) \times 0.4 + V_{k+1}^*(x_k - u) \times 0.6 \right).
\end{aligned}
$$

Example: A Betting Game

The Bellman Equation:

$$\begin{aligned} V_k^*(x_k) &= \max_u E[V_{k+1}^*(x_{k+1})] \\ &= \max_u \left(V_{k+1}^*(x_k + u) \times 0.4 + V_{k+1}^*(x_k - u) \times 0.6 \right). \end{aligned}$$

Stage 3

$$V_3^*(x_3) = \begin{cases} 1, & x_3 = 4, 5, \ldots, 16 \\ 0, & x_3 = 0, 1, 2, 3 \end{cases}$$

Example: A Betting Game

Stage 2

$$V_2^*(x_2) = \max_u \left(V_3^*(x_2 + u) \times 0.4 + V_3^*(x_2 - u) \times 0.6 \right)$$

Note that $u_k^* = 0$ when $x_k \geq 4$, so we only need to consider $x_k \in \{1, 2, 3\}$.

- $V_2^*(x_2 = 1) = \max\{\underset{u=0}{0}, \underset{u=1}{0}\} = 0$
- $V_2^*(x_2 = 2) = \max\{\underset{u=0}{0}, \underset{u=1}{0}, \underset{u=2}{0.4}\} = 0.4$
- $V_2^*(x_2 = 3) = \max\{0, \underset{u=1}{0.4}, \underset{u=2}{0.4}, \underset{u=3}{0.4}\} = 0.4$
- $V_2^*(x_2 = k) = 1, \; k \geq 4$

Example: A Betting Game

Stage 1

- $V_1^*(x_1 = 0) = 0$
- $V_1^*(x_1 = 1) = \max\{ V_2^*(x_2 = 1) = 0 \ (u = 0),$
  $0.4 V_2^*(x_2 = 2) + 0.6 V_2^*(x_2 = 0) = 0.16 \ (u = 1)\}$
  $= 0.16$
- $V_1^*(x_1 = 2) = \max\{ V_2^*(x_2 = 2) = 0.4 \ (u = 0),$
  $0.4 V_2^*(x_2 = 3) + 0.6 V_2^*(x_2 = 1) = 0.16 \ (u = 1),$
  $0.4 V_2^*(x_2 = 4) + 0.6 V_2^*(x_2 = 0) = 0.4 \ (u = 2)\}$
  $= 0.4$

Example

Stage 1

- $V_1^*(x_1 = 3) = \max\{V_2^*(x_2 = 3) = 0.4 \ (u = 0),$
  $0.4V_2^*(x_2 = 4) + 0.6V_2^*(x_2 = 2) = 0.64 \ (u = 1),$
  $0.4V_2^*(x_2 = 5) + 0.6V_2^*(x_2 = 1) = 0.4 \ (u = 2),$
  $0.4V_2^*(x_2 = 6) + 0.6V_2^*(x_2 = 0) = 0.4 \ (u = 3)\}$
  $= 0.64$
- $V_1^*(x_1 \geq 4) = 1$

Example

Stage 0

- $V_0^*(x_0 = 2) = \max\{V_1^*(x_1 = 2) = 0.4 \ (u = 0),$
  $0.4V_1^*(x_1 = 3) + 0.6V_1^*(x_1 = 1) = 0.352 \ (u = 1),$
  $0.4V_1^*(x_1 = 4) + 0.6V_1^*(x_1 = 0) = 0.4 \ (u = 2)\}$
  $= 0.4$

Optimal strategy: bet once with \$2

Infinite Horizon Discounted RL

Time-homogeneous stochastic systems (environments)

- Transition kernel $P(x_k = j | x_{k-1} = i, u_{k-1} = a) = P(x' = j | x = i, u = a)$ does not depend on k
- Reward function $r_k(x_k = x, u_k = u) = r(x, u)$ does not depend on k
- Discount factor $0 < \alpha < 1$ and initial state x_0, find a policy μ such that

$$V_0(x_0) = \max_{\mu} E\left[\sum_{k=0}^{\infty} \alpha^k r(x_k, \mu_k(x_k))\right]$$

The Bellman Equation

$$V_k^*(x_k) = \max_{\mu_k} E\left[r(x_k, \mu_k(x_k)) + \alpha V_{k+1}^*(x_{k+1})\right]$$

$$\Longrightarrow V^*(x) = \max_{\mu} E\left[r(x, \mu(x)) + \alpha V^*(x')\right]$$

The Bellman Equation in RL

Episodic RL

$$V_k^*(x_k) = \max_{\mu_k} E\left[r_k(x_k, \mu_k(x_k)) + V_{k+1}^*(x_{k+1})\right]$$

Infinite Horizon Discounted RL

$$V^*(x) = \max_{\mu} E\left[r(x, \mu(x)) + \alpha V^*(x')\right]$$

The goal of reinforcement learning: Learn the optimal policy μ^* by solving the Bellman equation.

Reference

- Chapters 1.1 and 1.2 of Dimitri P. Bertsekas, *Reinforcement Learning and Optimal Control*, Athena Scientific, 2019. Slides and lectures available at `https://web.mit.edu/dimitrib/www/RLbook.html`