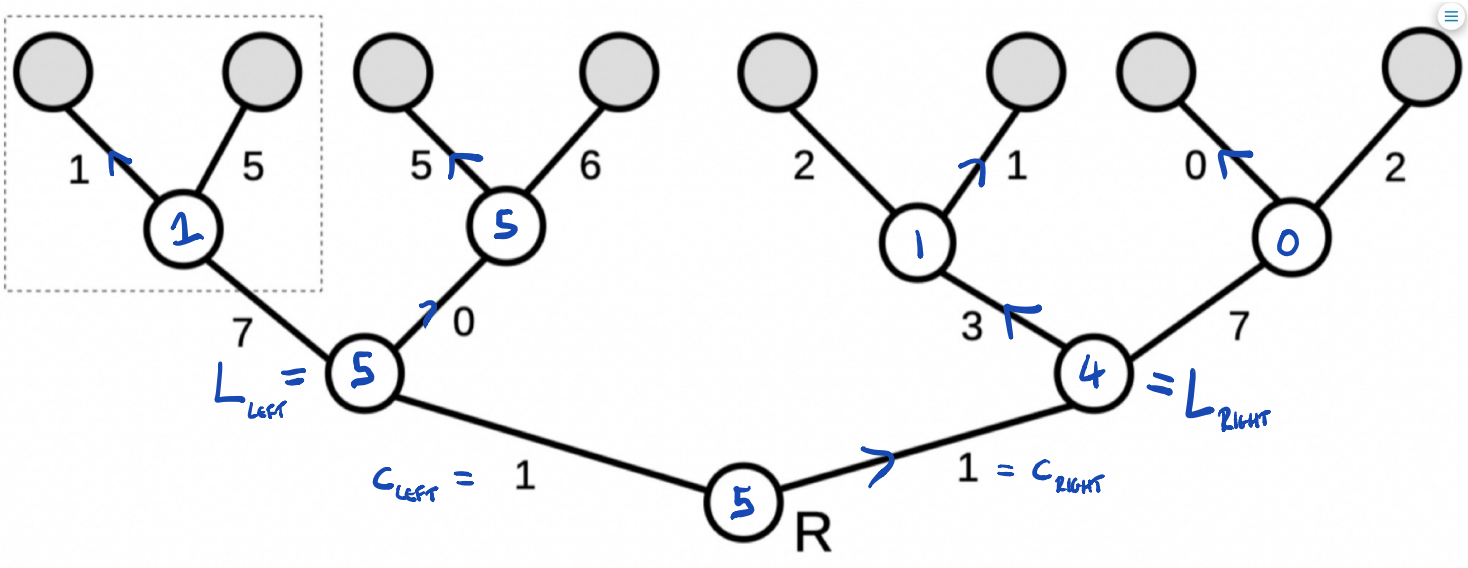
1. DYNAMIC PROGAMMING.

FIND THE SHORTEST PATH FROM THE ROOT

LOWEST COST



- SOLVED BACK-TO-FRONT
- PROBLEM REPEATS ITSELF i.e. SOLVE

GENERAL SOLUTION TO SHORTEST PATHS IS BELLMAN-FORD

$$L = \min_{a \in \{LEFT, RIGHT\}} \{ c_a + L_a \}$$

← BELLMAN EQN

ABSTRACT DEFINITION:

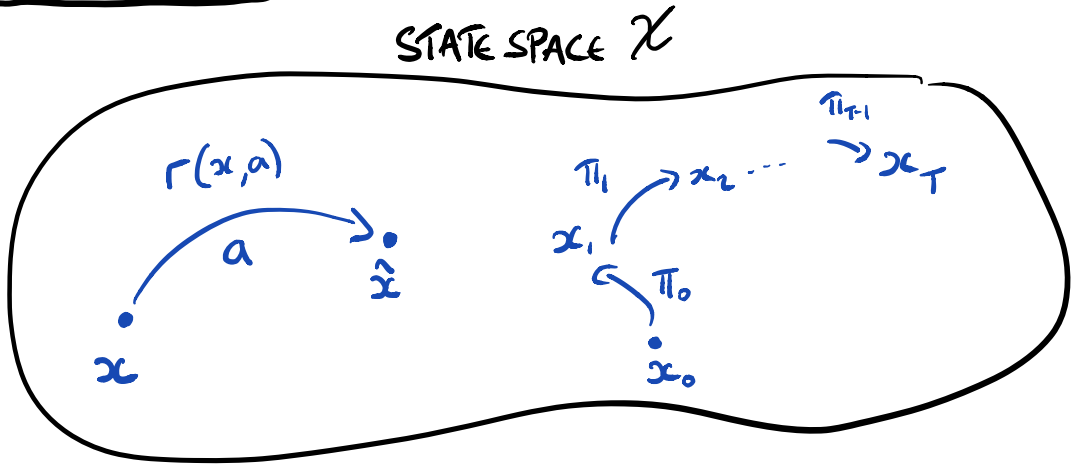
STATE $x \in \mathcal{X}$

ACTION $a \in \mathcal{A}$

REWARD $r(x,a)$

NEXT STATE

$$\hat{x} = f(x,a)$$



POLICY $\pi(x)$ OR π_t .

OBJECTIVE: [MAXIMIZE SUM OF REWARDS]

$$V_T(x_0) := \max \sum_{t=0}^{T-1} r(x_t, \pi_t) + r(x_T) \quad \text{OVER } \pi_0, \dots, \pi_{T-1} \in \mathcal{A}.$$

$V_T(x_0)$: VALUE FUNCTION

$\sum_{t=0}^{T-1} r(x_t, \pi_t) + r(x_T)$: $R_T(x, \pi)$

THE BELLMAN EQUATION

THEOREM: $V_t(x) = \underset{a \in A}{MAX} \left\{ r(x,a) + V_{t-1}(\hat{x}) \right\}$, $V_0(x) = r(x)$

PROOF:

$$V_t(x) = \underset{a_0 \in A, \dots, a_{t-1} \in A}{MAX} \; r(x,a_0) + r(x_1,a_1) + \dots + r(x_{t-1},a_{t-1}) + r(x_t)$$

$$\underset{a_0 \in A}{MAX} \; \underset{a_1, \dots, a_{t-1} \in A}{MAX} \; r(x,a_0) + r(x_1,a_1) + \dots + r(x_{t-1},a_{t-1}) + r(x_t)$$

$$= \underset{a_0 \in A}{MAX} \; r(x,a_0) + \underbrace{\underset{a_1, \dots, a_{t-1} \in A}{MAX} \; r(x_1,a_1) + \dots + r(x_{t-1},a_{t-1}) + r(x_t)}_{V_{t-1}(\hat{x})}$$

$$= \underset{a_0 \in A}{MAX} \left\{ r(x,a_0) + V_{t-1}(\hat{x}) \right\}$$

□

Some code:

```
def DP(time,state,f,r,A):
  '''
  Solves a dynamic program
  '''
  if time > 0 :
    Q = [ r[state][action] + DP(time-1, f[state][action]) for
    action in A ]
    V = max(Q)
  else :
    Q = r[state]
    V = max(Q)
  return V
```

THE PRINCIPLE OF OPTIMALITY:

(aka. WHEN SHOULD DYNAMIC PROGRAMMING WORK!)

"Principle of Optimality: An optimal policy has the property that whatever the initial state and initial decisions are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decisions." – Richard Bellman [2]

EXAMPLE (SHORTEST & LONGEST PATHS)

[SHORTEST PATH GOES THROUGH B FROM WHICH MUST TAKE SHORTEST PATH TO D]

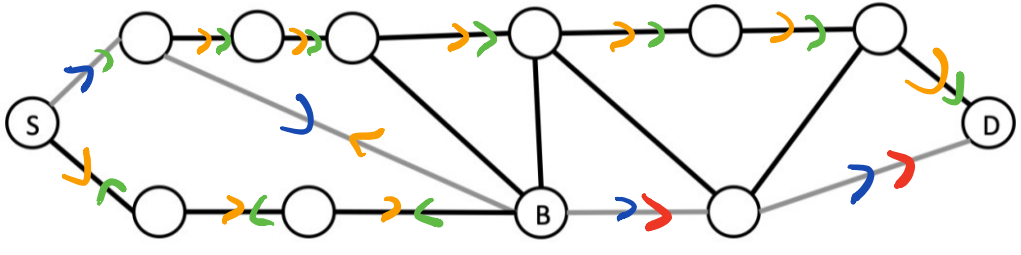
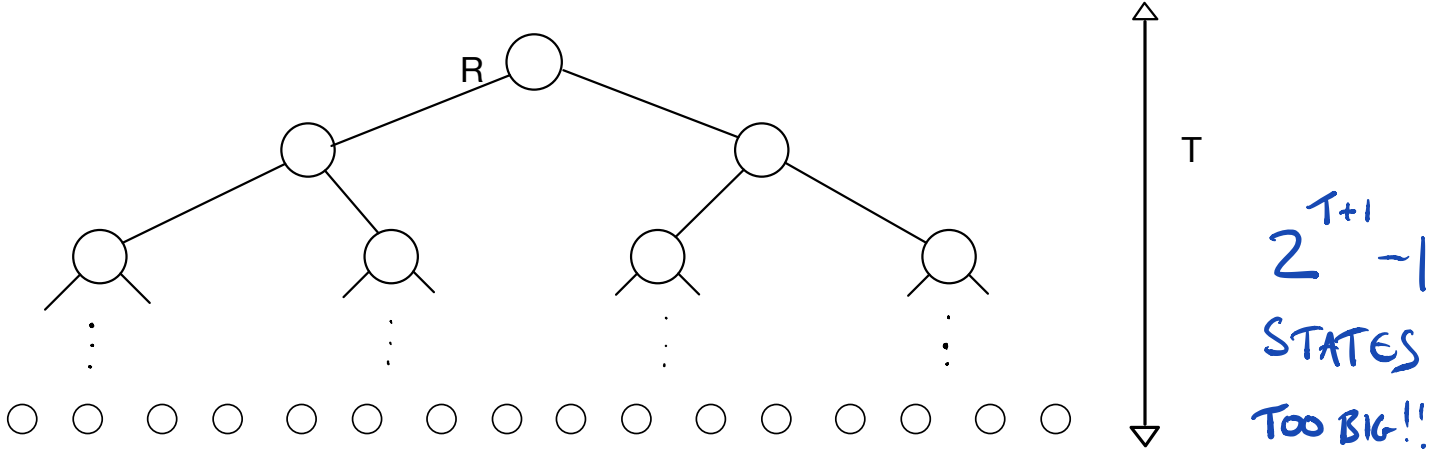


Figure 1.2: Shortest path from S to D.

THE CURSE OF DIMENSIONALITY:

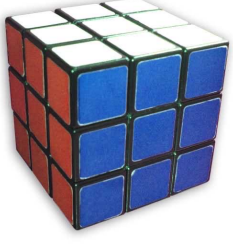
TREE EXAMPLE (AGAIN):



$2^{T+1} - 1$ STATES

TOO BIG!!

RUBIK'S CUBE:



- SOURCE: WIKIPEDIA!

4.3×10^{19}

A SHORTEST PATH PROBLEM

BUT

OPTIMAL SOLUTION UNKNOWN...

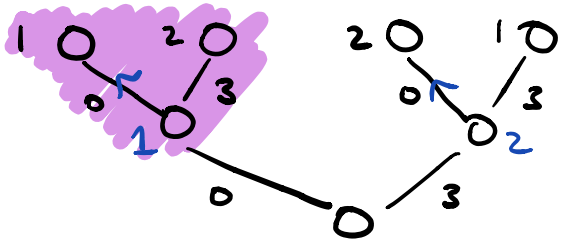
[i.e. WE KNOW HOW TO FIND THE OPTIMAL SOLUTION BUT COMPUTERS AREN'T FAST ENOUGH]

OTHER OBSERVATIONS

MINIMIZE: $L_T(x_0) := \text{MIN} \sum_{t=0}^{T-1} c(x_t, a_t) + c(x_T)$ OVER $a_1, \ldots, a_{T-1} \in A$

BELLMAN EQN: $L_t(x_t) = \underset{a \in A}{\text{MIN}} \left\{ c(x_t, a) + L_{t-1}(\hat{x}) \right\}$

MEMOIZE: SOME TIMES PROBLEM REPEATS eg.



SO STORE VALUES & ACTIONS

HELPS OVER COME "CURSE OF DIMENSIONALITY".

GENERALIZE: D.P. STILL WORKS FOR

$$\text{MAX} \sum_{t=0}^{T-1} r_t(x_t, a_t, x_{t+1}) + r_T(x_T)$$

s.t. $a_0 \in A_0, \ldots, a_{T-1} \in A_{T-1}$