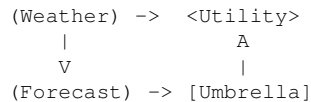


## CS 486 — Lecture 17: Decision Networks

### 1 The Weather DN

- Suppose we want to ask — do we take our umbrella or not?
- Depending on the weather, we may bring our umbrella.
- But we don't actually see the weather directly, we see the *forecast*.
- We can draw it like this:



- A policy specifies what the agent should do under all contingencies.
- For each decision variable, a policy specifies a value for the decision variable for each assignment of values to its parents.
- So, for our weather decision network, how many policies are there?
- Forecast has 3 possible values, and for each value of forecast, there are 2 possible decisions. This gives  $2^3 = 8$  possible policies.
- We have two approaches to solving the network problem:
  - Brute force: solve every policy's expected utility, and choose the best one.
  - VEA.
- Obviously we prefer VEA since brute force can result in having to calculate a very large number of policies compared to the number of nodes!
- Consider the policy  $\pi_1$ :
  - Take the umbrella if the forecast is cloudy
  - Leave the umbrella at home otherwise

What is the expected utility of this policy?

- In this case, using a brute force approach:

$$\begin{aligned} EU(\pi_1) &= P(\text{norain}) * P(\text{sunny}|\text{norain}) * u(\text{norain}, \text{leaveit}) \\ &\quad + P(\text{norain}) * P(\text{cloudy}|\text{norain}) * u(\text{norain}, \text{takeit}) \\ &\quad + P(\text{norain}) * P(\text{rainy}|\text{norain}) * u(\text{norain}, \text{leaveit}) \\ &\quad + P(\text{rain}) * P(\text{sunny}|\text{rain}) * u(\text{rain}, \text{leaveit}) \\ &\quad + P(\text{rain}) * P(\text{cloudy}|\text{rain}) * u(\text{rain}, \text{takeit}) \\ &\quad + P(\text{rain}) * P(\text{rainy}|\text{rain}) * u(\text{rain}, \text{leaveit}) \\ &= 0.7 * 0.7 * 100 + \dots + 0.13 * 0.6 * 0 \\ &= 64.05 \end{aligned}$$

- We see that this could take too long. We'll now investigate VEA.

## 1.1 VEA

- The general algorithm is as follows:
  1. Remove all variables that are not ancestors of the utility node.
  2. Create factors.
  3. While there are decision nodes remaining:
    - (a) Sum out each RV that is not a parent of a decision node
    - (b) Find the optimal policy for the last decision
  4. Return the optimal policies.
  5. Sum out all the remaining RVs to get the expected utility of the optimal policy.
- Let's apply this to our weather example:
  1. Nothing to be done.
  2. Define three factors,  $f_1(\text{Weather})$ ,  $f_2(\text{Forecast}, \text{Weather})$ ,  $f_3(\text{Weather}, \text{Umbrella})$
  3. Now:
    - (a) Weather is not a parent of any decision node. Sum out Weather. For example,  $f_1 * f_2 * f_3 = f_4(\text{Weather}, \text{Forecast}, \text{Umbrella})$ , then sum out  $f_4$  to get  $f_5(\text{Forecast}, \text{Umbrella})$ .
    - (b) Now let's find the optimal policy for, say, Umbrella.