CS480 – Introduction to Artificial Intelligence

Topic: Making Simple Decisions
Chapter: 16

Mustafa Bilgic

http://www.cs.iit.edu/~mbilgic

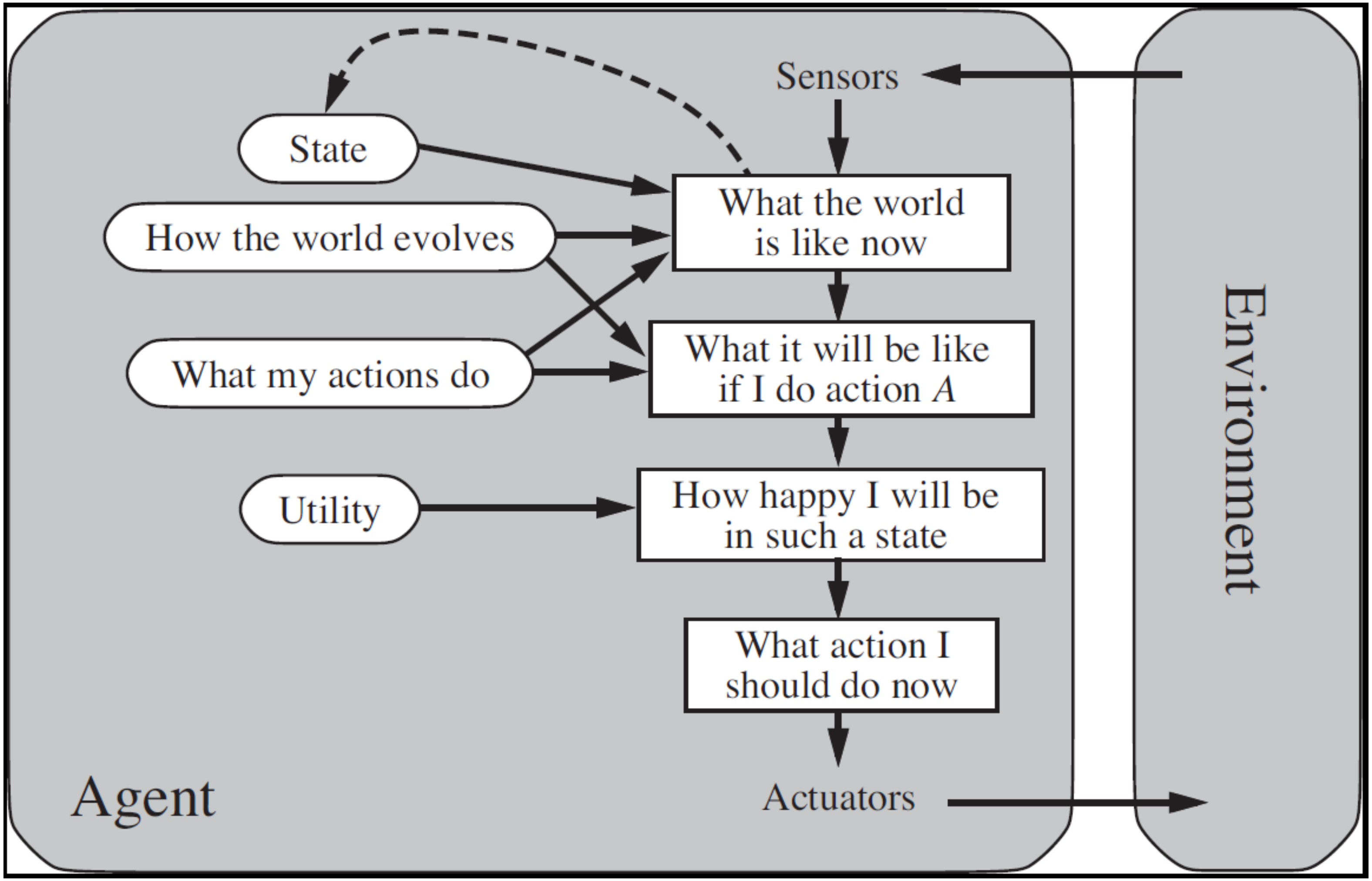
https://twitter.com/bilgicm

MOTIVATION

- Goal-based agent of Chapter 3
  - Fully observable & deterministic
- Now – Chapter 16
  - The world might be partially observable
  - The actions might be non-deterministic

We discuss "how an agent should make decisions so that it gets what it wants— on average, at least."

UTILITY-BASED AGENT



UTILITY

- $P(RESULT(a) = s' \mid a, \boldsymbol{e})$
  - The probability of ending up in state s' after taking action a, given that we have so far observed $\boldsymbol{e}$
- The agent's preferences are captured by a utility function U(s)
- Expected utility of an action a given evidence $\boldsymbol{e}$, $EU(a \mid \boldsymbol{e})$, is the average utility of the possible outcomes weighted by their probabilities

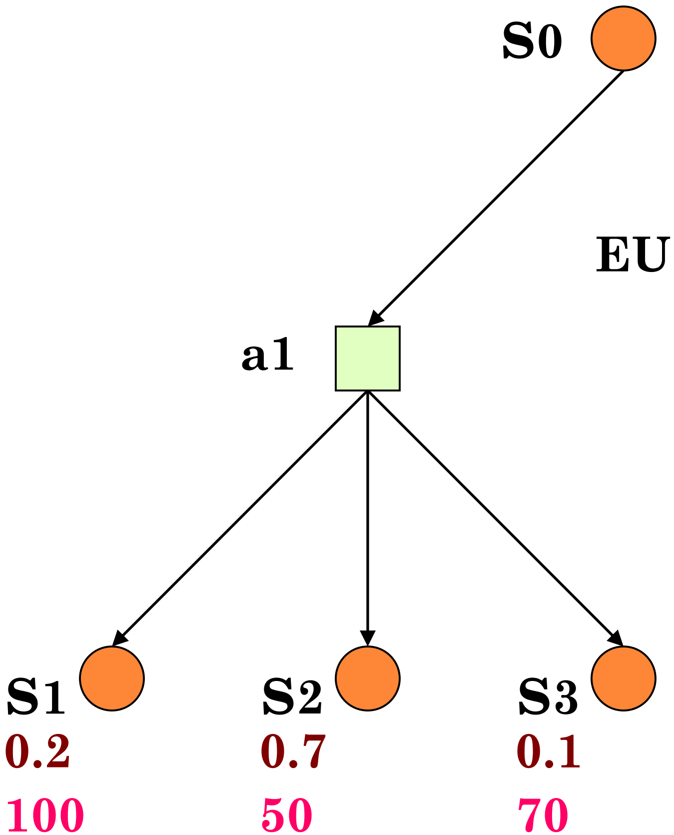
$$EU(a \mid \boldsymbol{e}) = \sum_{s'} P(RESULT(a) = s' \mid a, \boldsymbol{e}) \times U(s')$$

Maximum Expected Utility Principle (MEU)

- Choose action that maximizes the expected utility

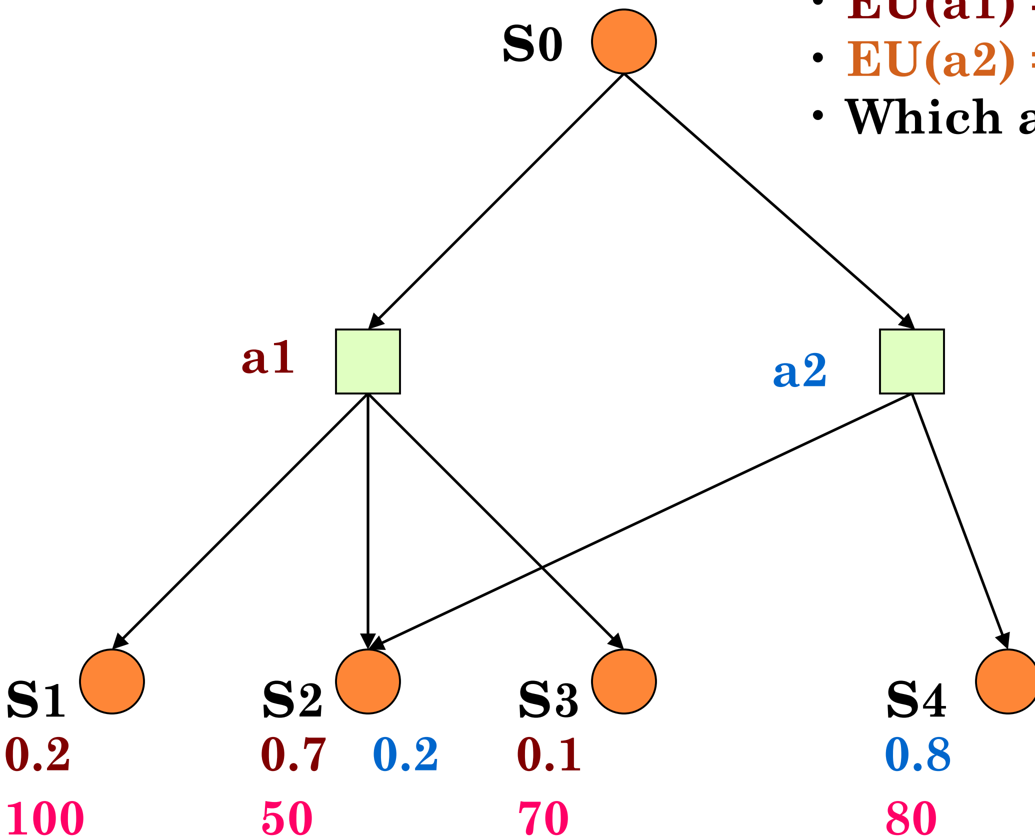
$$action = \arg\max_{a} EU(a \mid \boldsymbol{e})$$

One Action Example



EU(a1) = 100 x 0.2 + 50 x 0.7 + 70 x 0.1
= 20 + 35 + 7
= 62

Two Actions Example



- EU(a1) = 62
- EU(a2) = 0.2*50+0.8*80 = 74
- Which action to take?

Utility Theory – Rational Preferences

- Notation
  - A > B: the agent prefers A over B
  - A ~ B: the agent is indifferent between A and B
  - A ≥ B: the agent prefers A over B or is indifferent between them
- Lottery: n possible outcomes with probabilities
  - $[p_1, S_1; p_2, S_2; \ldots p_n, S_n]$
  - Each S_i can be an atomic state or another lottery

Axioms of utility theory

1. **Orderability**
   - $A>B$, $B>A$, or $A \sim B$
2. **Transitivity**
   - $A > B$ and $B > C \Rightarrow A > C$
3. **Continuity**
   - $A > B > C \Rightarrow \exists p\ [p, A; (1-p), C] \sim B$

Axioms of utility theory

4. **Substitutability**
   - $A \sim B \Rightarrow [p, A; (1\text{-}p), C] \sim [p, B; (1\text{-}p), C]$
   - $A > B \Rightarrow [p, A; (1\text{-}p), C] > [p, B; (1\text{-}p), C]$
5. **Monotonicity**
   - $A > B \Rightarrow (p > q \Leftrightarrow [p, A; (1\text{-}p), B] > [q, A; (1\text{-}q), B])$
6. **Decomposibility**
   - $[p, A; (1\text{-}p), [q, B; (1\text{-}q), C]] \sim [p, A; (1\text{-}p)q, B; (1\text{-}p)(1\text{-}q), C]$

PREFERENCES LEAD TO UTILITY

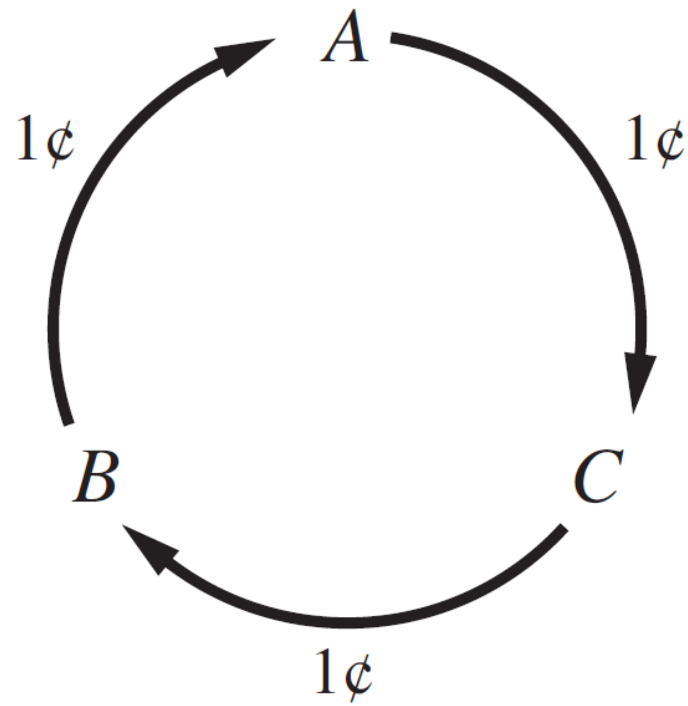
- **Existence of utility**
  - If an agent's preferences obey the axioms of utility, then there exists a function such that
    - $U(\text{A}) > U(\text{B}) \Leftrightarrow \text{A} > \text{B}$, and
    - $U(\text{A}) = U(\text{B}) \Leftrightarrow \text{A} \sim \text{B}$.
- **Expected utility of a lottery**
  - $U([p_1, \text{S}_1; p_2, \text{S}_2; \ldots\ p_n, \text{S}_n]) = p_1 U(\text{S}_1) + p_2 U(\text{S}_2) + \ldots + p_n U(\text{S}_n)$

Rationality

- If an agent's preferences do not obey the axioms of utility theory, then that agent can be made to behave irrationally
- For e.g., if an agent's preferences do not obey transitivity for three or more products, then the agent can be tricked to pay money in a cyclic manner indefinitely (or till the agent runs out of money)

EXAMPLE: VIOLATING TRANSITIVITY

- A > B
- B > C
- Transitivity requires A>C, but instead assume the agent prefers C over A, i.e, C>A
- Then the agent can be stripped of all of its money through cyclic transactions



Rationality

- An agent is rational if its preferences obey the axioms of utility theory, not matter how odd its preferences are
- An agent might have completely different preferences from another agent and both can still be rational, if and only if, their individual preferences obey the axioms of utility theory

UTILITY ≠ MONEY

- Most agents prefer more money to less money,
  - Thus it obeys the monotonicity constraint,
  - But this does not mean money behaves as a utility function
- For example, which lottery would you prefer
  - L_1: [1, $1 Million]
  - L_2: [0.5, $0; 0.5, $2.5 Million]
- If money served as a utility function, then you'd prefer L_2 no matter what, but the answer *often* depends on how much money you currently have
  - The utility of money depends on what you *prefer*
    - If you are short on cash, a little more *certain* money can help
    - If you are already billionaire, you might take the risk
    - Or if you are swimming in debt, you might like to gamble

Utility ≠ Money

- Let's say you currently have \$k and let S_k represent the state of having \$k
- $EU(L_1) = U(S_{k+1M})$
- $EU(L_2) = 0.5*U(S_k) + 0.5*U(S_{k+2.5M})$
- The rational choice depends on your preferences for S_k, S_{k+1M}, and $S_{k+2.5M}$
  - i.e, it depends on the values of $U(S_k)$, $U(S_{k+1M})$, and $U(S_{k+2.5M})$
- $U(S_i)$ does not have to be a linear function of i, and for people it often is not
  - However, U(.) has to obey the six axioms

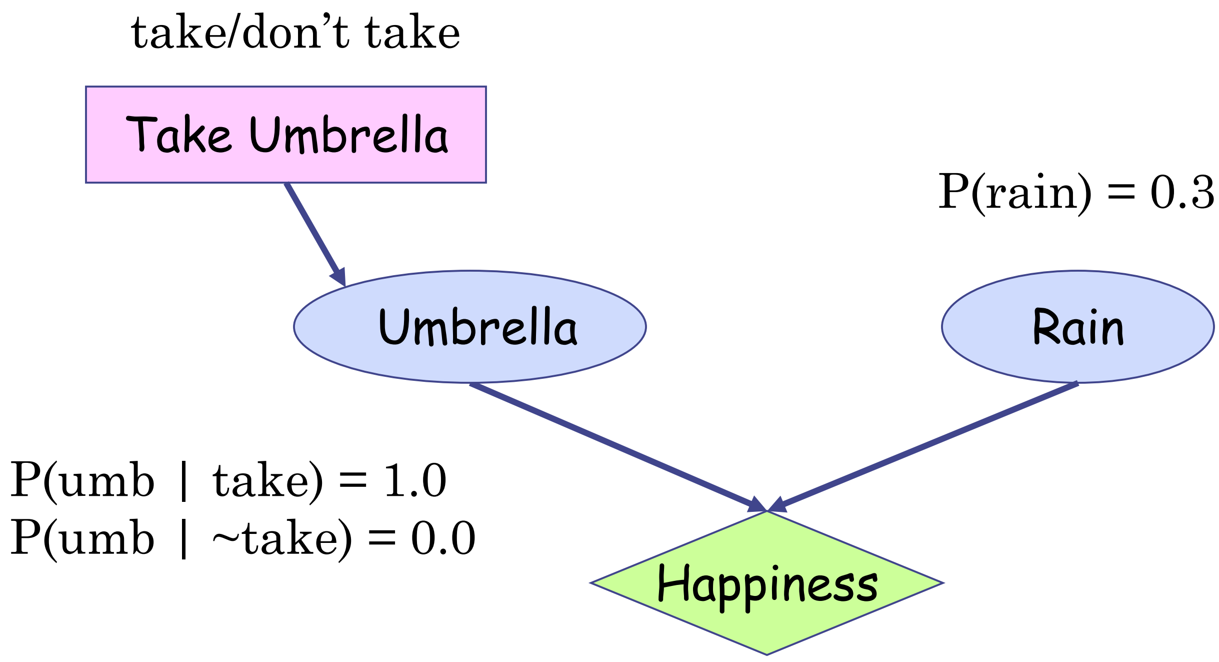
Human Judgment

- Read 16.3.2 – 16.3.4

Decision Networks

- Builds on Bayesian networks
- In addition to the chance nodes (ovals), decision networks have
  - Decision nodes – square
    - Represents actions
  - Utility nodes – diamond
    - Represents utilities for possible states and actions

Umbrella Example

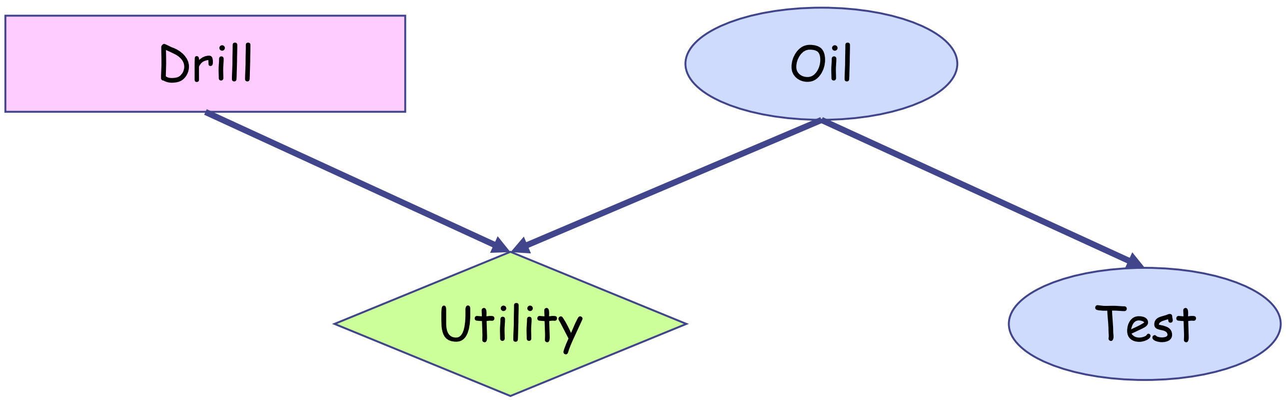


U(~umb, ~rain) = 100
U(~umb, rain) = 0
U(umb, ~rain) = 20
U(umb, rain) = 70

DRILL

drill/don't drill

P(oil) = 0.01



P(positive | oil) = 0.9
P(positive | ~oil) = 0.05

U(~drill, ~oil) = 100
U(~drill, oil) = -500
U(drill, ~oil) = -100
U(drill, oil) = 1000

Decision networks - applications

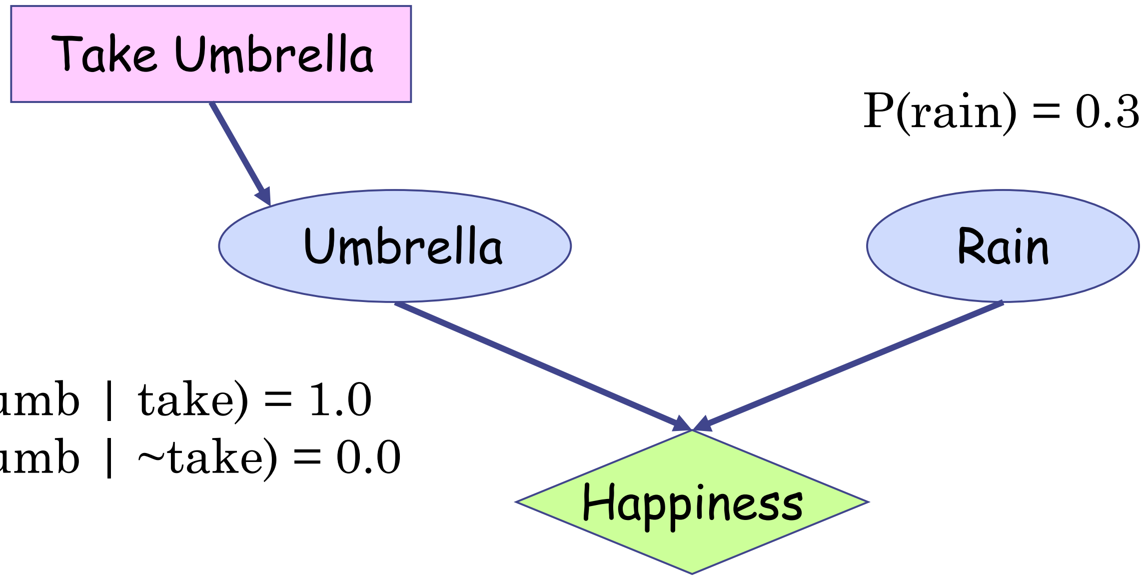
- Used for
  - What action to take
  - What information to gather
  - How much to pay for a piece of information
- For example:
  - Medical diagnosis: which test to perform, which treatment to prescribe, …
  - Marketing: which project to invest in, how much to spend on marketing, how much to spend on user surveys, …

EVALUATING DECISION NETWORKS

- Set evidence nodes **E** to their values **e**
- For each choice **a** of action **A**
  - Set **A=a**
  - Compute the posterior probability of the parent chance nodes of the utility node; i.e., compute P(Pa(Utility) | **e, a**)
  - Compute expected utility using the utility node and the probability distribution P(Pa(Utility) | **e, a**)
- Choose action **a** with the maximum expected utility

UMBRELLA EXAMPLE

take/don't take



P(rain) = 0.3

P(umb | take) = 1.0
P(umb | ~take) = 0.0

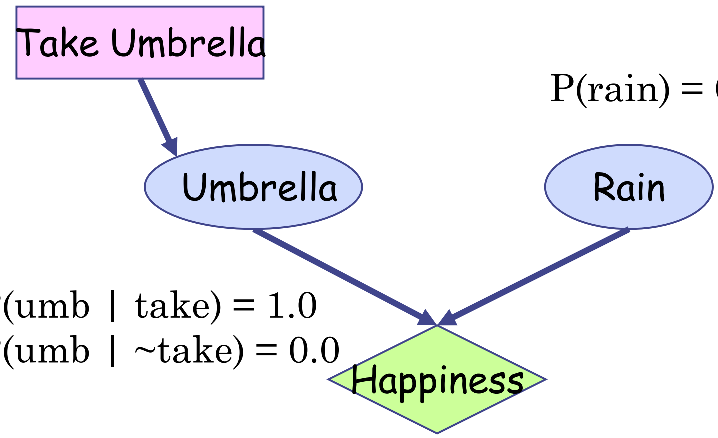
U(~umb, ~rain) = 100
U(~umb, rain) = 0
U(umb, ~rain) = 20
U(umb, rain) = 70

Umbrella Example

- Take Umbrella = take
  - Compute P(Umbrella, Rain | take)
  - Compute expected utility
- Take umbrella = ~take
  - Compute P(Umbrella, Rain | ~take)
  - Compute expected utility
- MEU principle: choose the action with the highest expected utility

Umbrella Example

take/don't take



P(rain) = 0.3

P(umb | take) = 1.0
P(umb | ~take) = 0.0

U(~umb, ~rain) = 100
U(~umb, rain) = 0
U(umb, ~rain) = 20
U(umb, rain) = 70

Take Umbrella = take

Umb	Rain	P(Umb, Rain | take)
~umb	~rain	0 x 0.7 = 0
~umb	rain	0 x 0.3 = 0
umb	~rain	1 x 0.7 = 0.7
umb	rain	1 x 0.3 = 0.3

Expected Utility = 0 x 100 + 0 x 0 + 0.7 x 20 + 0.3 x 70 = 35

Take Umbrella = ~take

Umb	Rain	P(Umb, Rain | ~take)
~umb	~rain	1 x 0.7 = 0.7
~umb	rain	1 x 0.3 = 0.3
umb	~rain	0 x 0.7 = 0
umb	rain	0 x 0.3 = 0

Expected Utility = 0.7 x 100 + 0.3 x 0 + 0.0 x 20 + 0.0 x 70 = 70

MEU Principle: Don't take it.

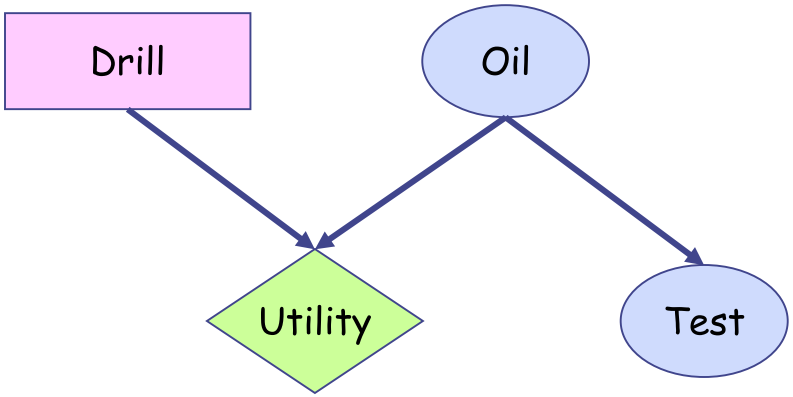
Value of Information

- If I am allowed to observe the value of a chance node, how much valuable is that information to me?
- Value of information
  - Expected utility after the information is acquired
    - Minus
  - Expected utility before the information is acquired
- There is one catch: we do not know the content of the information before we acquire it
  - Solution: take an expectation over the possible outcomes

How much is the Test worth?

DRILL

drill/don't drill   P(oil) = 0.01



U(~drill, ~oil) = 100
U(~drill, oil) = -500
U(drill, ~oil) = -100
U(drill, oil) = 1000

P(positive | oil) = 0.9
P(positive | ~oil) = 0.05

1. Compute MEU before Test
2. Compute MEU
   a. Assuming Test = positive
   b. Assuming Test = negative
3. VOI(Test) =

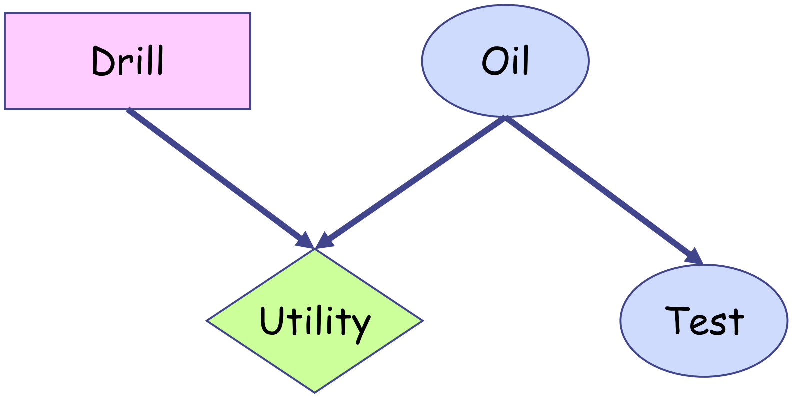
   P(Test = pos)*(MEU | Test = pos) +

   P(Test = neg)*(MEU | Test = neg) –

   MEU before Test

DRILL

drill/don't drill   P(oil) = 0.01



U(~drill, ~oil) = 100
U(~drill, oil) = -500
U(drill, ~oil) = -100
U(drill, oil) = 1000

P(positive | oil) = 0.9
P(positive | ~oil) = 0.05

MEU before Test

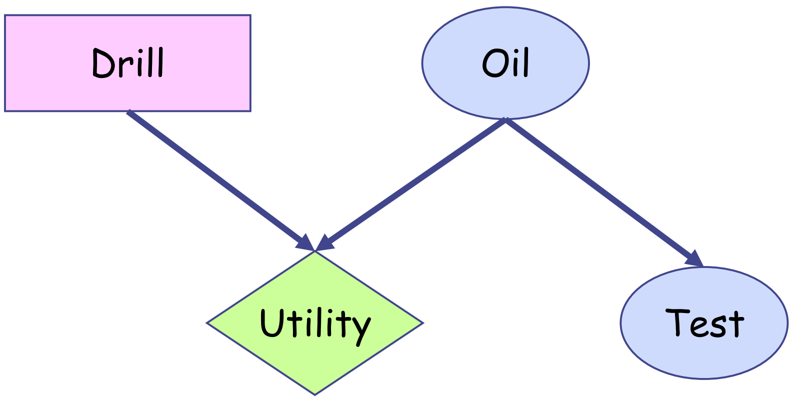
Drill = drill ⇒
EU = P(o | d) * U(d, o) + P(~o | d)*U(d, ~o)
= 0.01 * 1000 + 0.99 * -100
= 10 – 99
= -89

Drill = ~drill ⇒
EU = P(o | ~d) * U(~d, o) + P(~o | ~d)*U(~d, ~o)
= 0.01 * -500 + 0.99 * 100
= -5 + 99
= 94

MEU before Test = 94

DRILL

drill/don't drill P(oil) = 0.01



U(~drill, ~oil) = 100
U(~drill, oil) = -500
U(drill, ~oil) = -100
U(drill, oil) = 1000

P(positive | oil) = 0.9
P(positive | ~oil) = 0.05

MEU if Test = pos

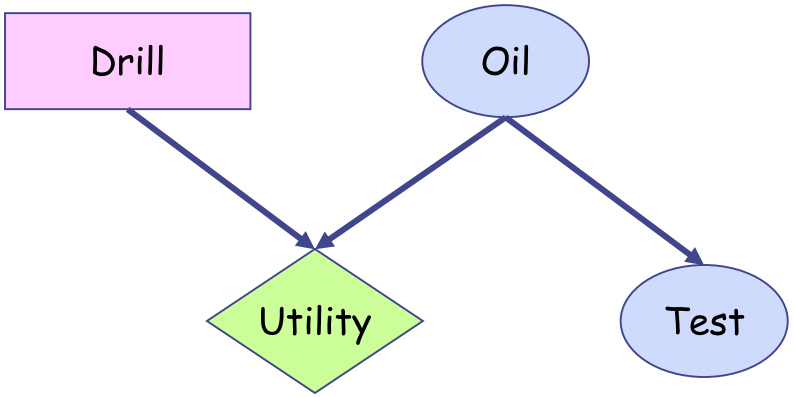
Drill = drill ⇒
EU = P(o | p, d) * U(d, o) + P(~o | p, d)*U(d, ~o)
= ? * 1000 + ? * -100
= ?
= ?

Drill = ~drill ⇒
EU = P(o | p, ~d) * U(~d, o) + P(~o | p, ~d)*U(~d, ~o)
= ? * -500 + ? * 100
= ?
= ?

MEU | pos = ?

DRILL

drill/don't drill   P(oil) = 0.01



U(~drill, ~oil) = 100
U(~drill, oil) = -500
U(drill, ~oil) = -100
U(drill, oil) = 1000

P(positive | oil) = 0.9
P(positive | ~oil) = 0.05

MEU if Test = neg

Drill = drill ⇒
EU = P(o | n, d) * U(d, o) + P(~o | n, d)*U(d, ~o)
= ? * 1000 + ? * -100
= ?
= ?

Drill = ~drill ⇒
EU = P(o | n, ~d) * U(~d, o) + P(~o | n, ~d)*U(~d, ~o)
= ? * -500 + ? * 100
= ?
= ?

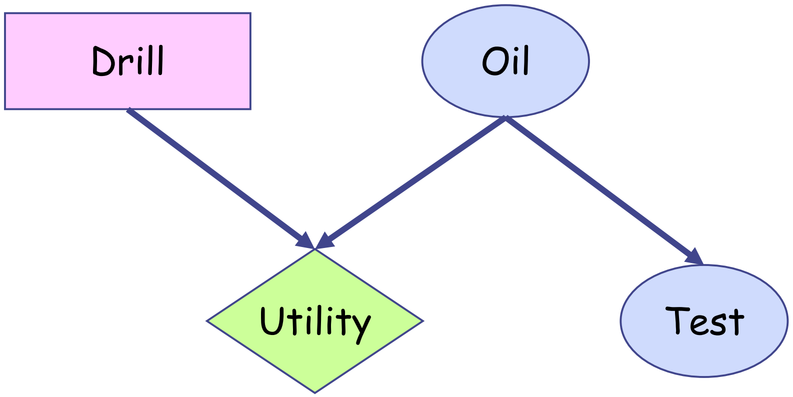
MEU | neg = ?

DRILL

VOI(Test)

VOI(Test) =
P(Test = pos) x (MEU | pos) +
P(Test = neg) x (MEU | neg) –
MEU before Test
= ? * ? + ? * ? – 94

drill/don't drill P(oil) = 0.01



U(~drill, ~oil) = 100
U(~drill, oil) = -500
U(drill, ~oil) = -100
U(drill, oil) = 1000

P(positive | oil) = 0.9
P(positive | ~oil) = 0.05

Making Simple Decisions - Summary

- MEU principle
- Decision networks
- Value of information