2.2: Agent Decision Making

AI6125 : Multi-Agent System

Assoc Prof Zhang Jie

Based on “An Introduction to MultiAgent Systems” by Michael Wooldridge, John Wiley & Sons, 2002/2009
“Artificial Intelligence: A Modern Approach” by S. Russell and P. Norvig.. Prentice-Hall, third edition, 2010

Overview

- Make Simple Decisions
- Make Complex Decisions
  - Sequential decision making
  - Agent’s utility depends on a sequence of decisions
  - Based on Chapters 16 & 17 in reference book: “Artificial Intelligence: A Modern Approach” by S. Russell and P. Norvig. Prentice-Hall, third edition, 2010

Making Simple Decisions

- Utility Theory
- Multi-Attribute Utility Functions
- Decision Networks
- The Value of Information

Beliefs and Uncertainty

Utility Function

Outcome Probabilities

Expected Utility

上个PPT的例题计算.

Maximum Expected Utility

- Expected Utility

$$EU(A \mid E) = \sum_i P(Result_i(A) \mid E, Do(A))U(Result_i(A))$$

- Principle of Maximum Expected Utility
  - Choose action A with highest $EU(A \mid E)$

Example

Robot

Turn Right → Hits wall (P = 0.1; U = 0)
→ Finds target (P = 0.9; U = 10)   9

Turn Left → Fall water (P = 0.3; U = 0)
→ Finds target (P = 0.7; U = 10)   7

Choose action "Turn Right"

Basis of Utility Theory

- Rational preference
  - Preference of rational agent $\Longrightarrow$ obey constraints
  - Behavior describable as maximization of expected utility
- Notation
  - Lottery(*L*): a complex decision making scenario
    - Different outcomes are determined by chance
  - $L = [p, A; 1 - p, B]$
  - $A \succ B$: *A* is preferred to *B*
  - $A \sim B$: indifference between *A* and *B*
  - $A \succeq B$: *B* is not preferred to *A*

Basis of Utility Theory

- Constraints
  - Orderability

$$(A \succ B) \vee (B \succ A) \vee (A \sim B)$$

  - Transitivity

$$(A \succ B) \wedge (B \succ C) \Rightarrow (A \succ C)$$

  - Continuity

$$A \succ B \succ C \Rightarrow \exists p [p, A; 1-p, C] \sim B$$

?

Basis of Utility Theory

- Constraints (cont.)
  - Substitutability

$$A \sim B \Rightarrow [p, A; 1-p, C] \sim [p, B; 1-p, C]$$

  - Monotonicity

$$A \succ B \Rightarrow (p \geq q \Leftrightarrow [p, A; 1-p, B] \succeq [q, A; 1-q, B]$$

  - Decomposability

$$[p, A; 1-p, [q, B; 1-q, C]] \\ \sim [p, A; (1-p)q, B; (1-p)(1-q), C]$$

Basis of Utility Theory

- Utility Principle

$$U(A) > U(B) \Leftrightarrow A \succ B$$

- Maximum Expected Utility principle

$$U([p_1, S_1; \ldots; p_n, S_n]) = \sum_i p_i U(S_i)$$

- Utility Function
  - Represents that the agent's actions are **trying to achieve**
  - Can be **constructed by observing agent's preferences**

Utility Functions

- Utility
  - Mapping state to real numbers
  - Approach
    - Compare *A* to standard lottery L_p
      - $u_\top$: best possible prize with prob. *p*
      - $u_\perp$: worst possible catastrophe with prob. *1-p*
    - Adjust *p* until $A \sim L_p$

$$\$30 \sim L \quad \begin{cases} \xrightarrow{0.9} \text{continue} \\ \xrightarrow{0.1} \text{death} \end{cases}$$

Utility Functions

- Utility Scales
  - Positive linear transform

    $$U'(x) = k_1 U(x) + k_2 \quad where \quad k_1 > 0$$
  - Normalized utility

    $u_\top = 1.0,\ u_\perp = 0.0$
  - Micromort
    - one-millionth chance of death
    - russian roulette, insurance
  - QALY

    quality-adjusted life years

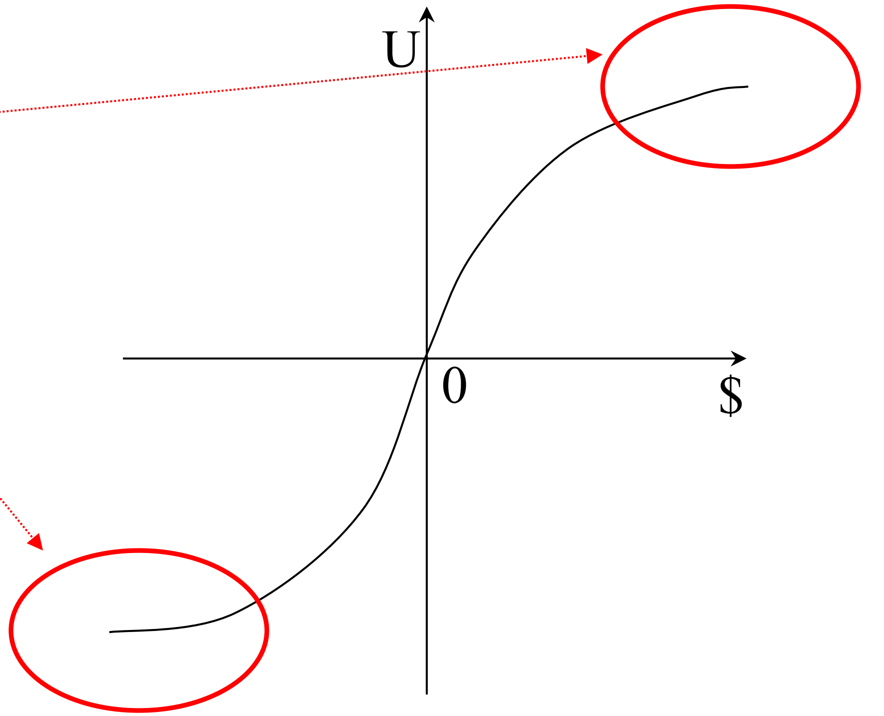
Utility Functions

- Money: does NOT behave as a utility function
  - Given a lottery ***L***
  - risk-averse

$$U(S_L) < U(S_{EMV(L)})$$

  - risk-seeking

$$U(S_L) > U(S_{EMV(L)})$$



Multi-attribute Utility Functions

- Multi-Attribute Utility Theory (MAUT)
  - Outcomes are characterized by 2 or more attributes.
  - Site a new airport
    - disruption by construction, cost of land, noise,….
- Approach
  - Identify regularities in the preference behavior

Multi-attribute Utility Functions

- Notation
  - Attributes

$$X_1, X_2, X_3, \ldots$$

  - Attribute value vector

$$X = < x_1, x_2, \ldots. >$$

  - Utility Fn. (function)

$$U(x_1, \ldots, x_n) = f[f_1(x_1), \ldots, f_n(x_n)]$$

Multi-attribute Utility Functions

- Dominance
  - Certain (strict dominance, Fig.1)
    - airport site ***S1*** cost less, less noise, safer than ***S2:* strict dominance of *S1* over *S2***
  - Uncertain(Fig. 2)

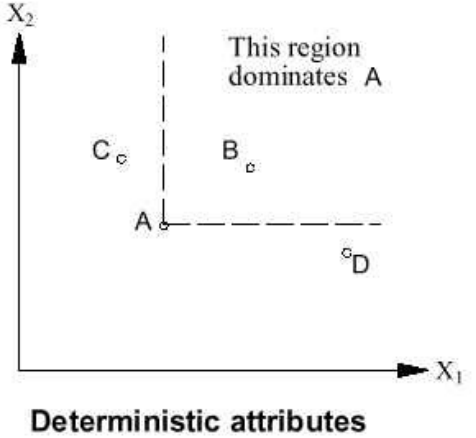


Fig. 1

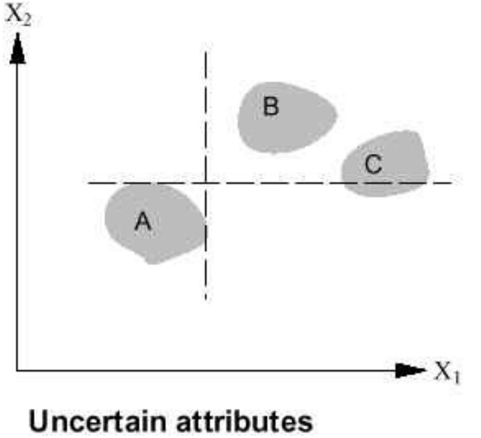


Fig.2

Multi-attribute Utility Functions

- Dominance(cont.)
  - Stochastic dominance
    - In real world problem
    - ***S1*** : avg \$3.7billion,

      standard deviation : \$0.4billion

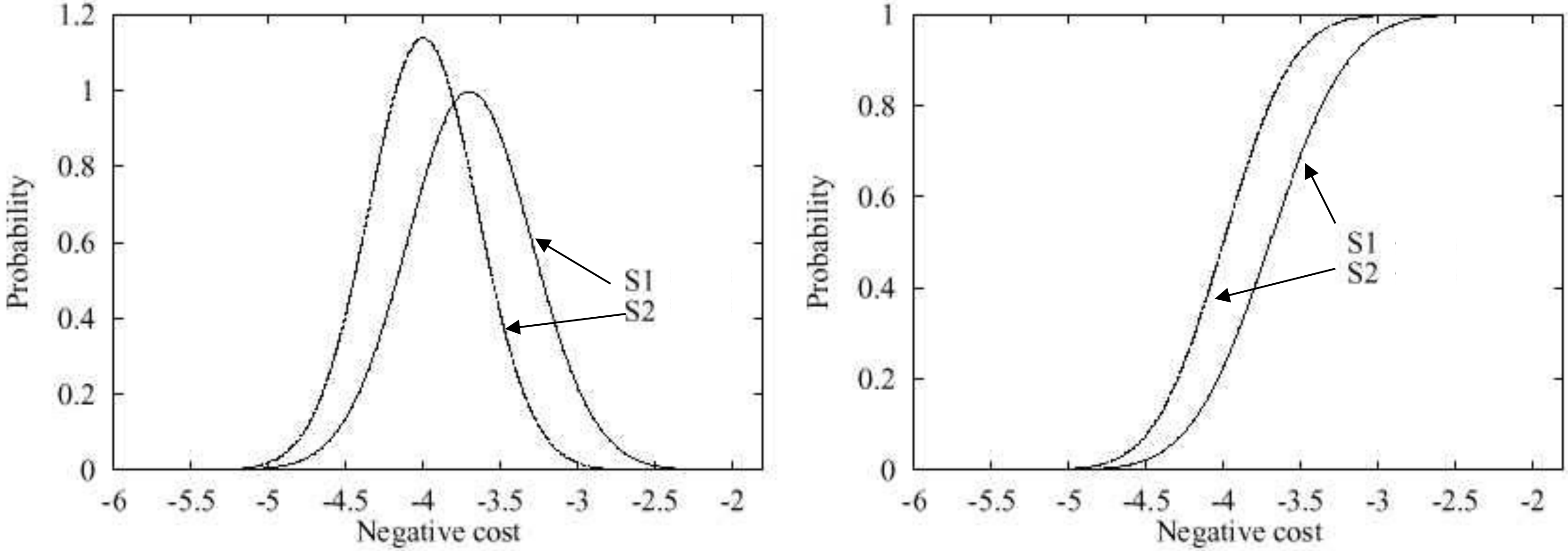
      S2 : avg \$4.0billion,

      standard deviation : \$0.35billion

      - ***S1* stochastically dominates *S2***

Multi-attribute Utility Functions

- Dominance(cont.)



Multi-attribute Utility Functions

- Preferences without Uncertainty
  - Preferences between concrete outcome values.
  - Preference structure
    - X1 & X2 ***preferentially independent*** of X3 iff

      Preference between $< x_1, x_2, x_3 > \& < x'_1, x'_2, x_3 >$

      Does not depend on x_3

    - Airport site : <Noise, Cost, Safety>

      <20,000 suffer, $4.6billion, 0.06deaths/mpm>

      vs. <70,000 suffer, $4.2billion, 0.06deaths/mpm>

Multi-attribute Utility Functions

- Preferences without Uncertainty (cont.)
  - Mutual preferential independence (MPI)
    - Every pair of attributes is P.I of its complements.
    - Airport site : <Noise, Cost, Safety>
      - Noise & Cost **P.I** Safety
      - Noise & Safety **P.I** Cost
      - Cost & Safety **P.I** Noise

      : <Noise,Cost,Safety> exhibits **MPI**
    - Agent's preference behavior

$$\max[V(S) = \sum_{I} V_i(X_i(S))]$$ ← 直接相加即可.

Multi-attribute Utility Functions

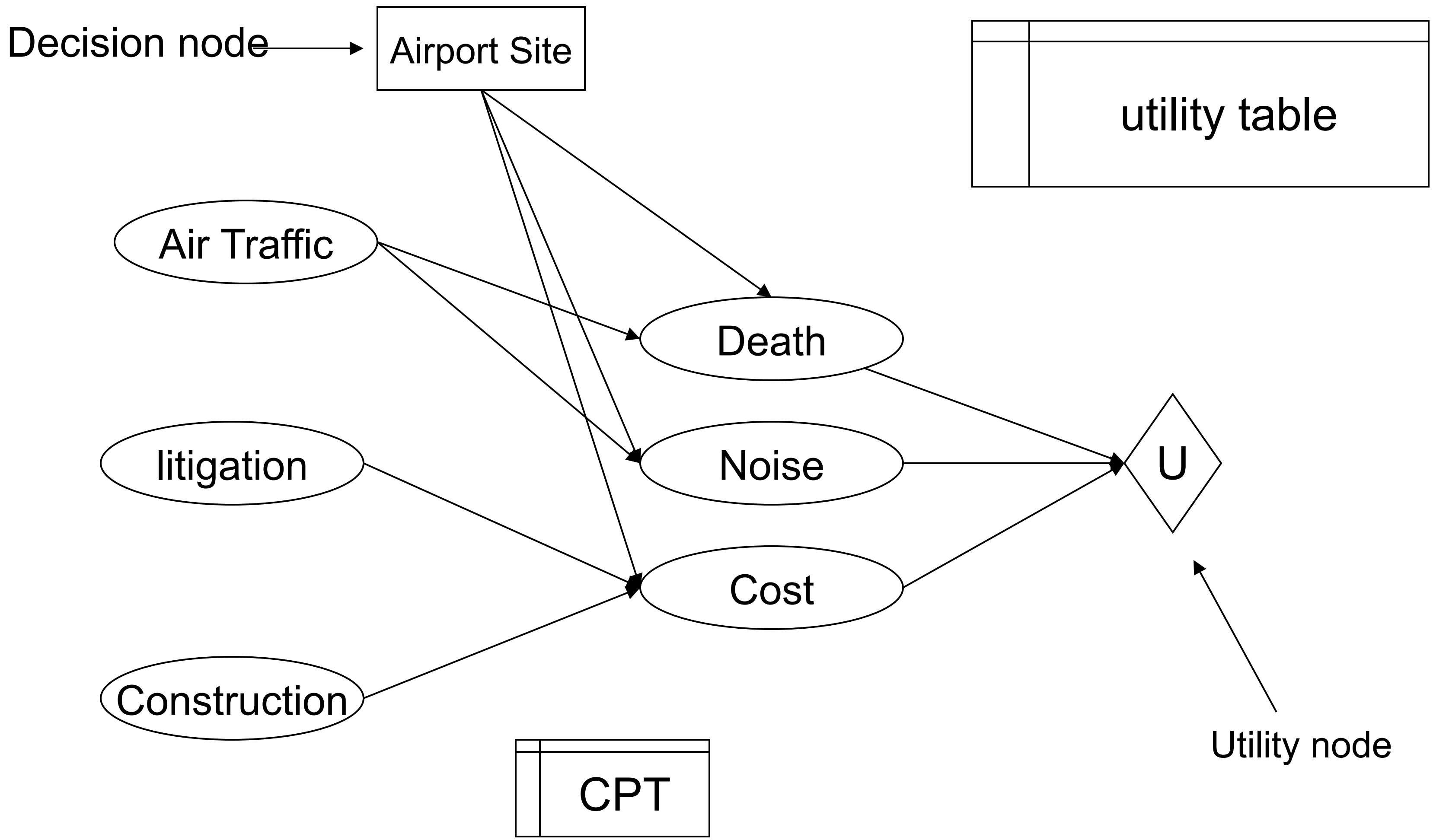
- Preferences with Uncertainty
  - Preferences btw. Lotteries' utility
  - Utility Independence (UI)
    - **X** is ***utility-independent*** of **Y** iff preferences over lotteries' attribute set **X** do not depend on particular values of a set of attribute **Y.**
  - Mutual U.I(MUI)
    - Each subset of attributes is U.I of the remaining attributes
    - agent's behavior (for 3 attributes): **multiplicative Utility Function**

$$U = k_1 U_1 + k_2 U_2 + k_3 U_3 + k_1 k_2 U_1 U_2 + k_2 k_3 U_2 U_3 + k_3 k_1 U_3 U_1 \\ + k_3 k_1 U_3 U_1$$

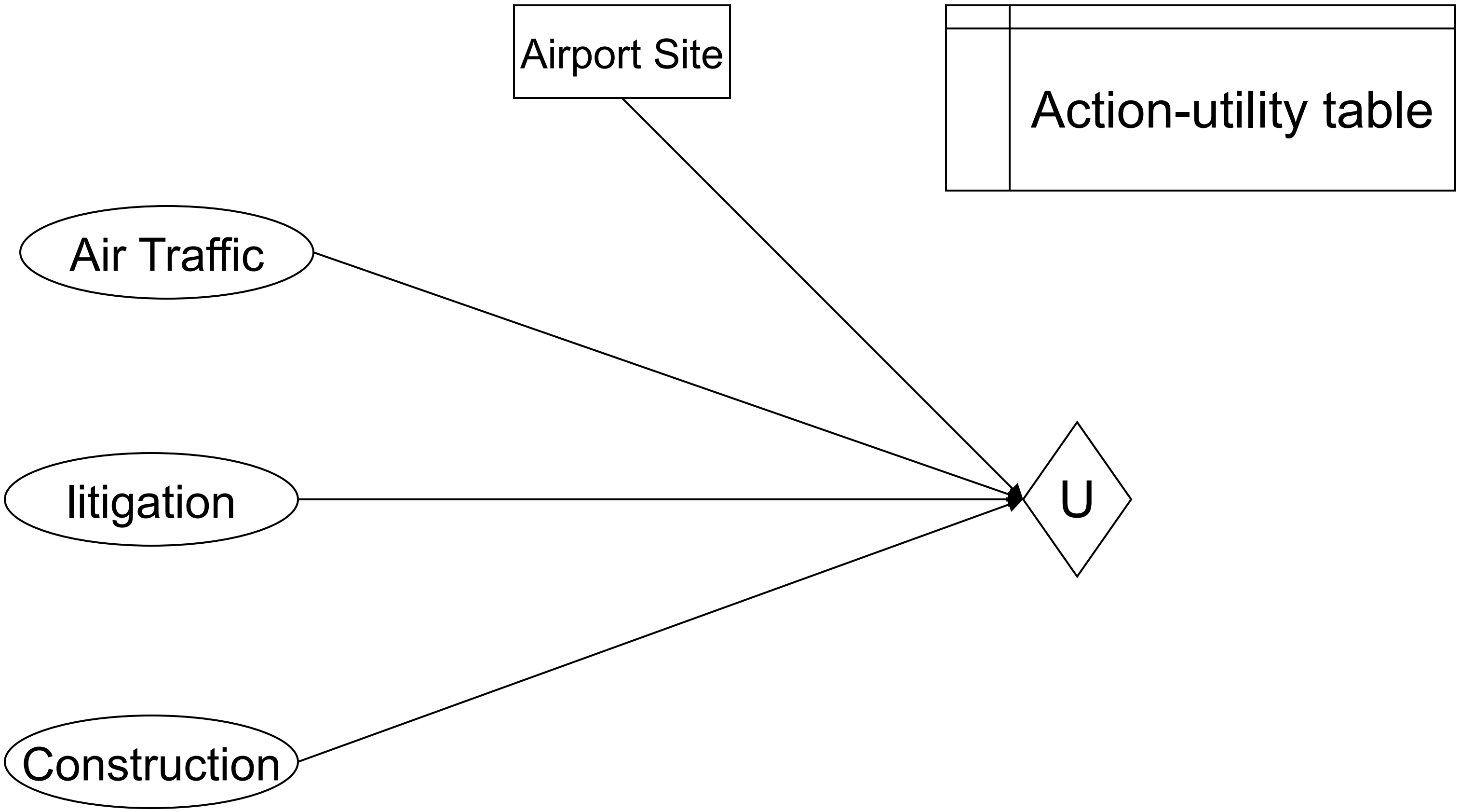
Decision Networks

- Simple formalism for expressing & solving decision problem
- Belief networks + decision & utility nodes
- Nodes
  - Chance nodes
  - Decision nodes
  - Utility nodes

A Simple Decision Network



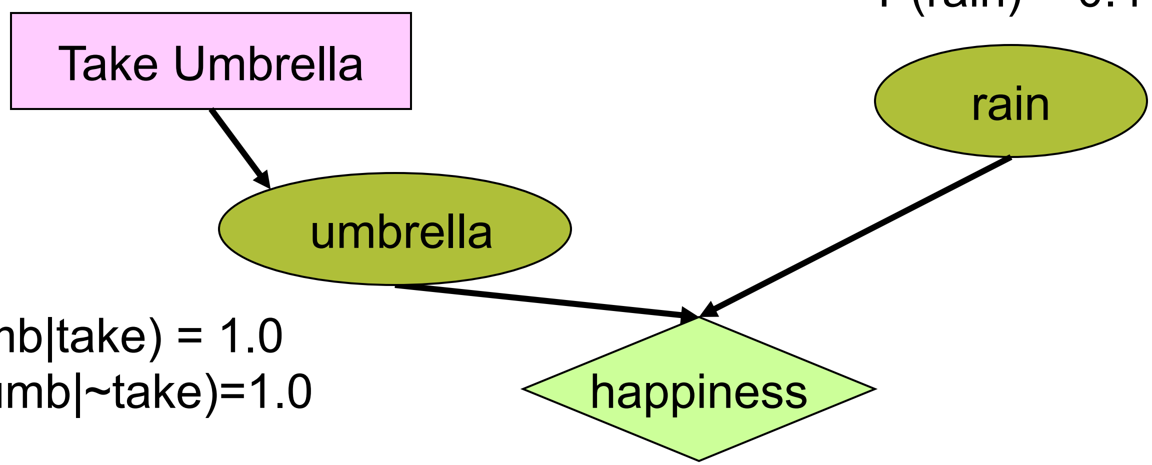
A Simplified Representation



Umbrella Network

take/don’t take

P(rain) = 0.4



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

U(~umb, ~rain) = 100
U(~umb, rain) = -100
U(umb,~rain) = 0
U(umb,rain) = -25

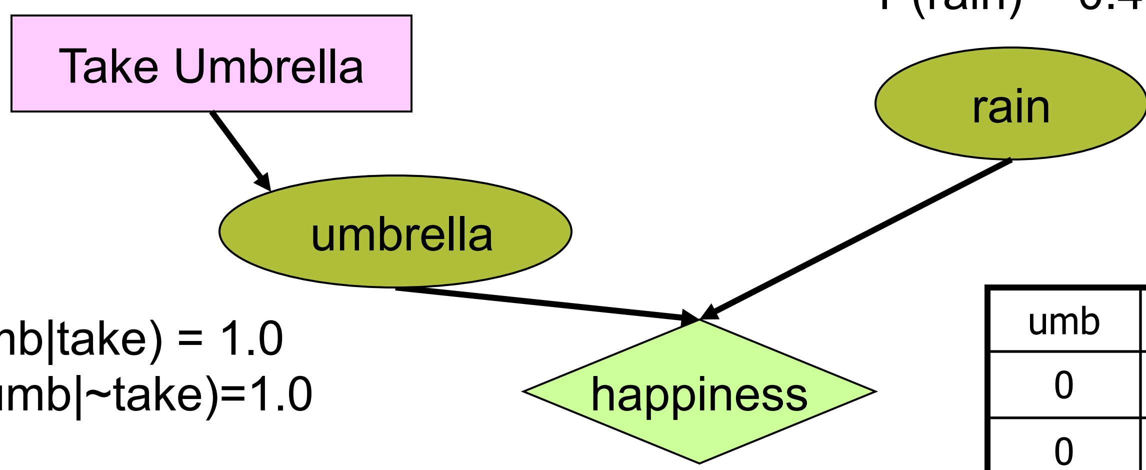
Evaluating Decision Networks

- Set the evidence variables for current state
- For each possible value of the decision node:
  - Set decision node to that value
  - Calculate the posterior probability of the parent nodes of the utility node, using BN inference
  - Calculate the resulting utility for action
- return the action with the highest utility

Umbrella Network

take/don't take

P(rain) = 0.4



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

U(~umb, ~rain) = 100
U(~umb, rain) = -100
U(umb,~rain) = 0
U(umb,rain) = -25

#1

umb	rain	P(umb,rain | take)
0	0	
0	1	
1	0	
1	1	

#2: EU(take)

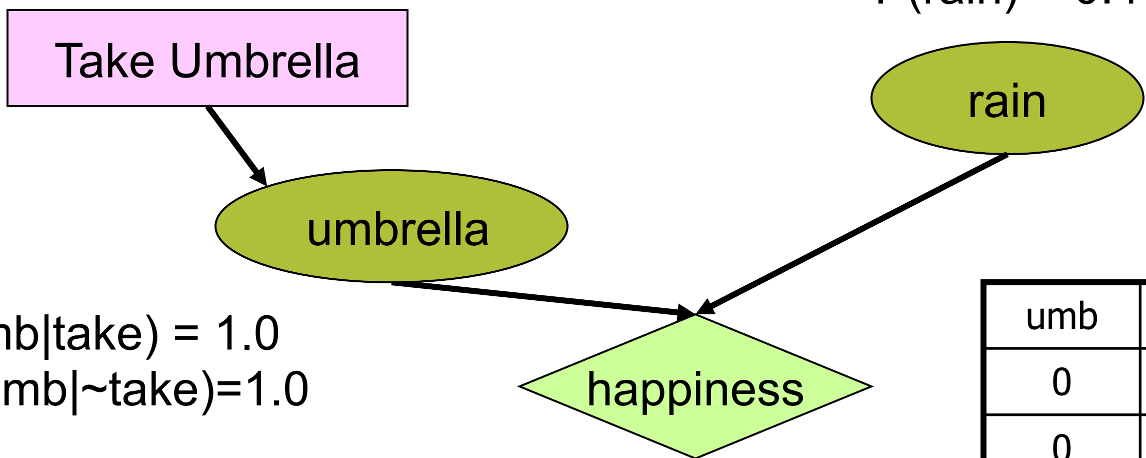
Umbrella Network

umb	rain	P(umb, rain | take)
0	0	~~0.4~~ ~~0.6~~ 0
0	1	~~0.4~~ 0
1	0	0.6
1	1	0.4

EU(take) = ~~60-40-10~~
=-10

take/don't take

P(rain) = 0.4



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

U(~umb, ~rain) = 100
U(~umb, rain) = -100
U(umb,~rain) = 0
U(umb,rain) = -25

#1

umb	rain	P(umb,rain | ~take)
0	0	0.6
0	1	0.4
1	0	0
1	1	0

#2: EU(~take) =20

EU(~take) > EU(take)
20 > -10

Value of Information (VOI)

- Suppose agent's current knowledge is E. The value of the current best action α is

$$EU(\alpha \mid E) = \max_A \sum_i U(Result_i(A)) P(Result_i(A) \mid E, Do(A))$$

- The value of the new best action (after new evidence E′ is obtained):

$$EU(\alpha' \mid E, E') = \max_A \sum_i U(Result_i(A)) P(Result_i(A) \mid E, E', Do(A))$$

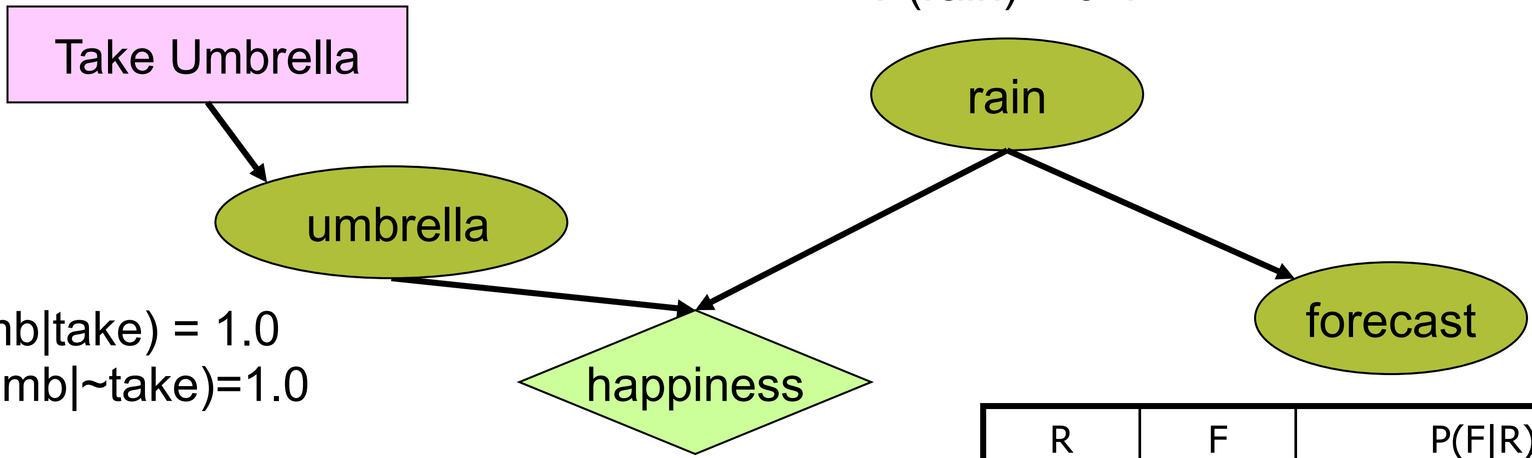
- the value of information for E′ is:

$$VOI(E') = \sum_k P(e_k \mid E) EU(\alpha_{ek} \mid e_k, E) - EU(\alpha \mid E)$$

Umbrella Network

take/don't take

P(rain) = 0.4



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

U(~umb, ~rain) = 100
U(~umb, rain) = -100
U(umb,~rain) = 0
U(umb,rain) = -25

R	F	P(F|R)
0	0	0.8
0	1	0.2
1	0	0.3
1	1	0.7

VOI

- VOI(forecast)= $P(rainy)EU(\alpha_{rainy}) + P(\sim rainy)EU(\alpha_{\sim rainy}) - EU(\alpha)$

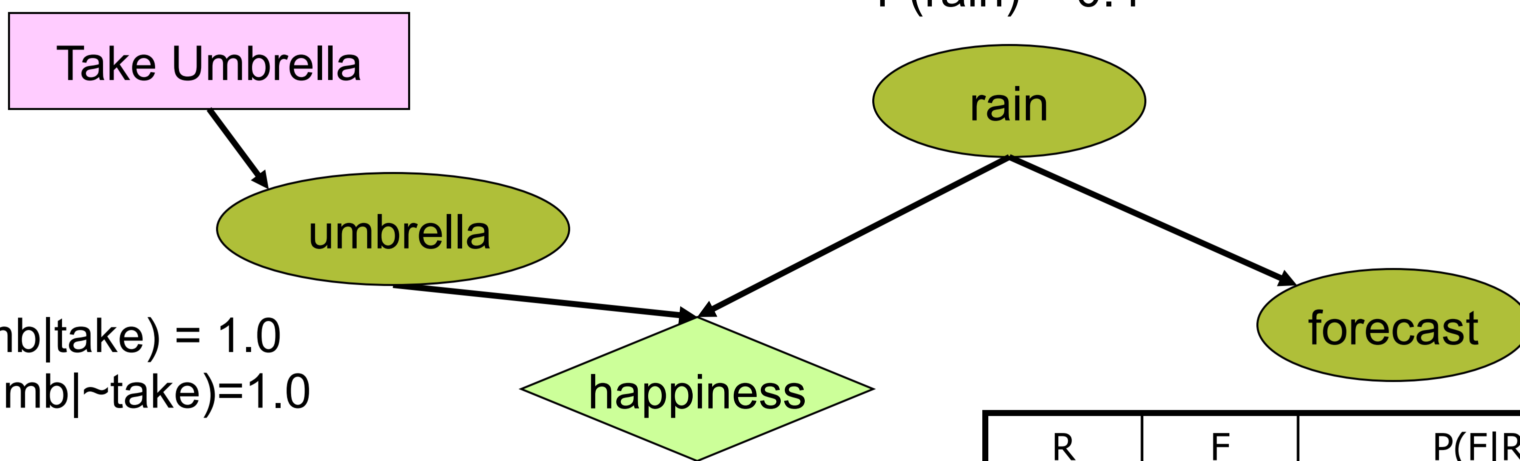
Umbrella Network

P(F=rainy) = 0.4

F	R	P(R|F)
0	0	0.8
0	1	0.2
1	0	0.3
1	1	0.7

take/don't take

P(rain) = 0.4



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

U(~umb, ~rain) = 100
U(~umb, rain) = -100
U(umb,~rain) = 0
U(umb,rain) = -25

R	F	P(F|R)
0	0	0.8
0	1	0.2
1	0	0.3
1	1	0.7

umb	rain	P(umb,rain | take, rainy)
0	0	
0	1	
1	0	
1	1	

#1: EU(take|rainy)

umb	rain	P(umb,rain | ~take, rainy)
0	0	
0	1	
1	0	
1	1	

#2: EU(~take|rainy)

umb	rain	P(umb,rain | take, ~rainy)
0	0	
0	1	
1	0	
1	1	

#3: EU(take|~rainy)

umb	rain	P(umb,rain |~take, ~rainy)
0	0	
0	1	
1	0	
1	1	

#4: EU(~take|~rainy)

Making Complex Decisions

simple : one decision.

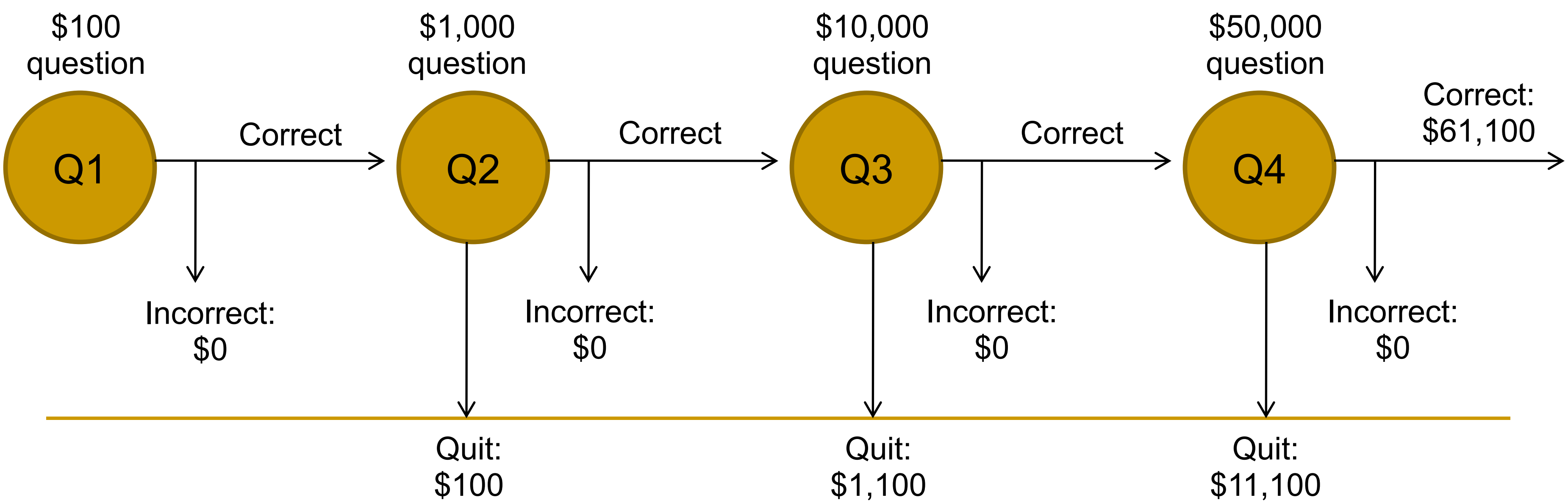
- Make a sequence of decisions
  - Agent's utility depends on a sequence of decisions
  - Sequential Decision Making
- Markov Property
  - Transition properties depend only on the current state, not on previous history (how that state was reached)
  - Markov Decision Processes

Markov Decision Processes

- Components:
  - **States** s, beginning with initial state s_0
  - **Actions** a
    - Each state s has actions A(s) available from it
  - **Transition model** P(s' | s, a)
    - *Markov assumption*: the probability of going to s' from s depends only on s and a and not on any other past actions or states
  - **Reward function** R(s)
- **Policy** π(s): the action that an agent takes in any given state
  - The "solution" to an MDP   optimal policy.

Game Show

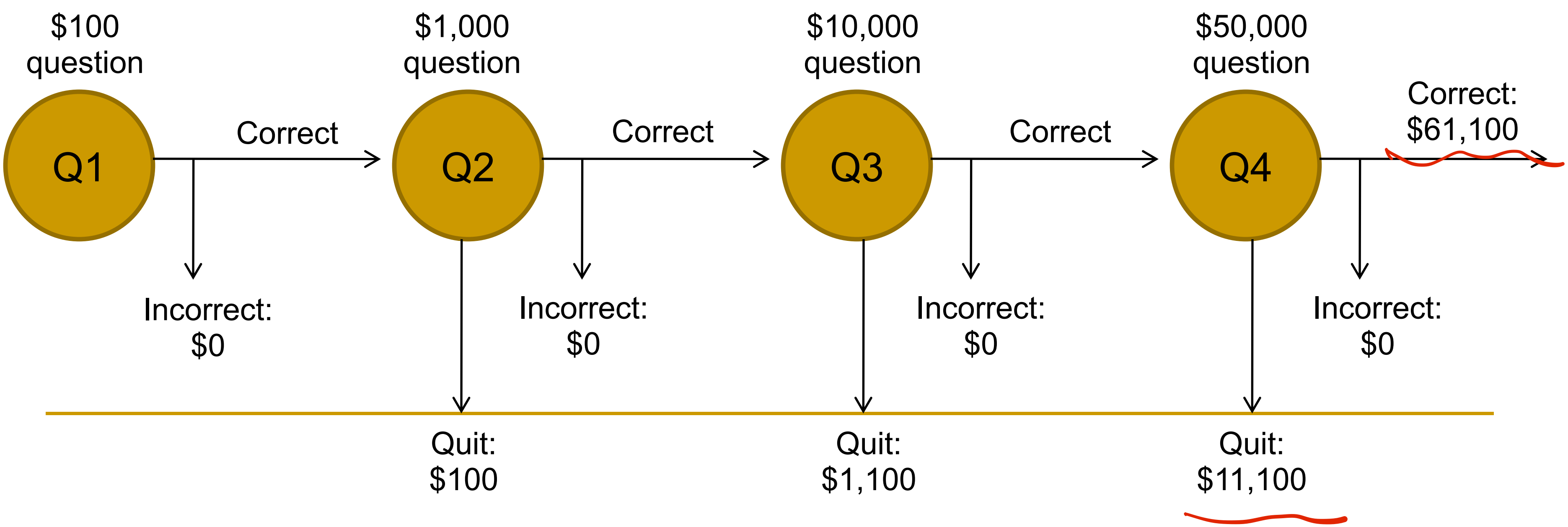
- A series of questions with increasing level of difficulty and increasing payoff
- Decision: at each step, take your earnings and quit, or go for the next question
  - If you answer wrong, you lose everything



Game Show

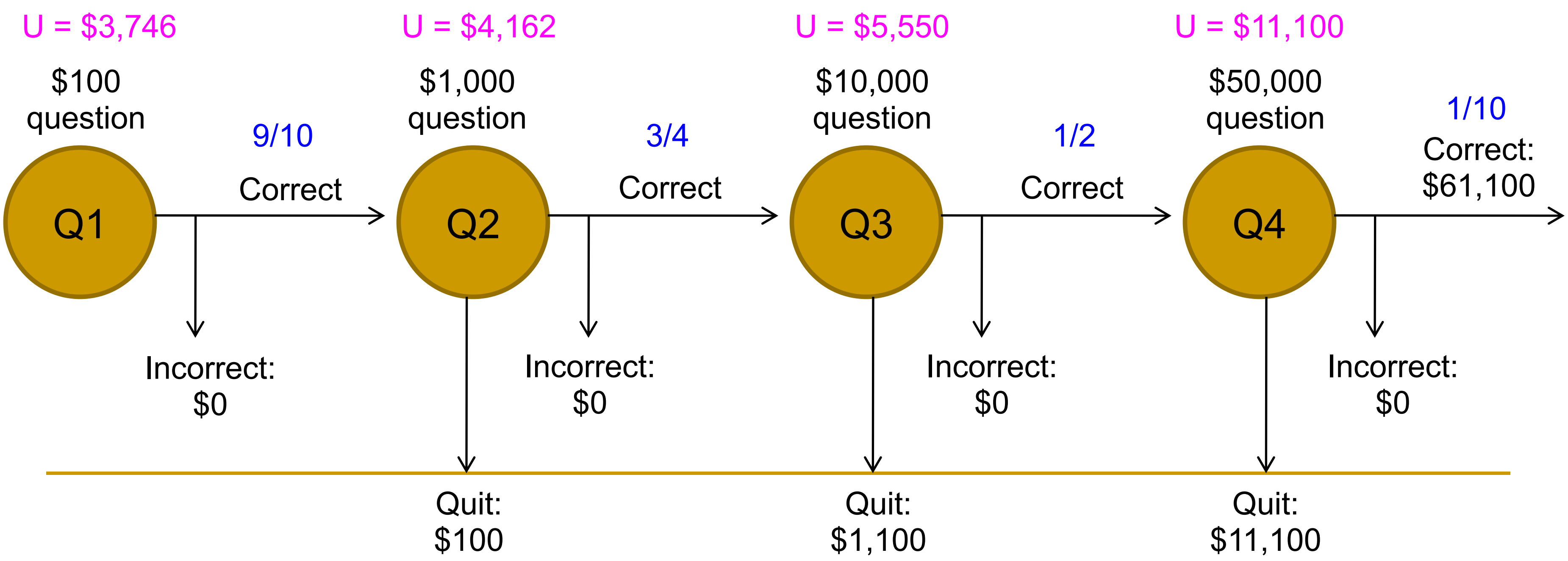
- Consider $50,000 question
  - Probability of guessing correctly: 1/10
  - Quit or go for the question?
- What is the expected payoff for continuing?

  0.1 * 61,100 + 0.9 * 0 = 6,110
- What is the optimal decision?

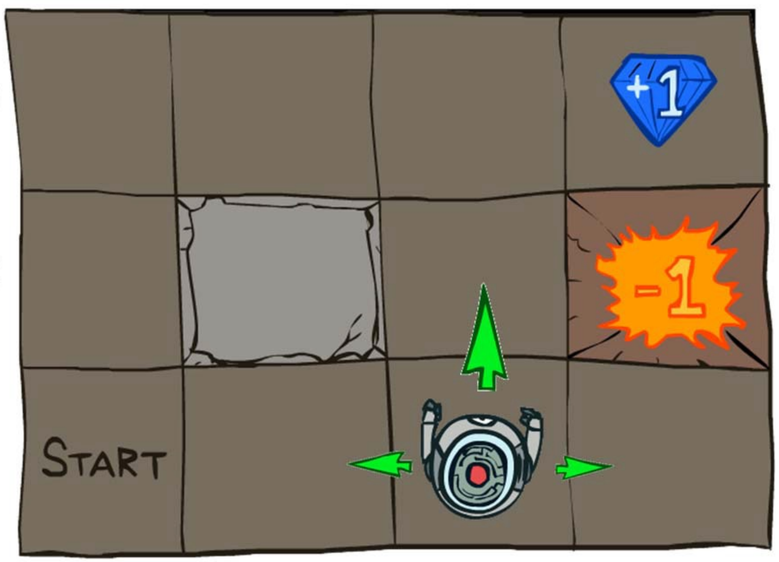


Game Show

- What should we do in Q3?
  - Payoff for quitting: $1,100
  - Payoff for continuing: 0.5 * $11,100 = $5,550
- What about Q2?
  - $100 for quitting vs. $4,162 for continuing
- What about Q1?

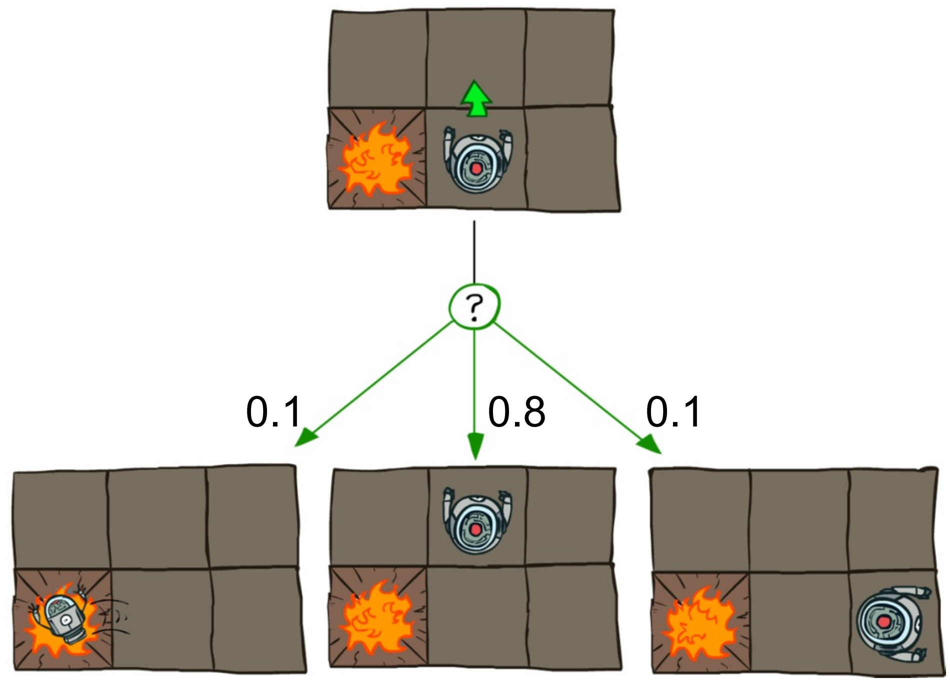


Grid World



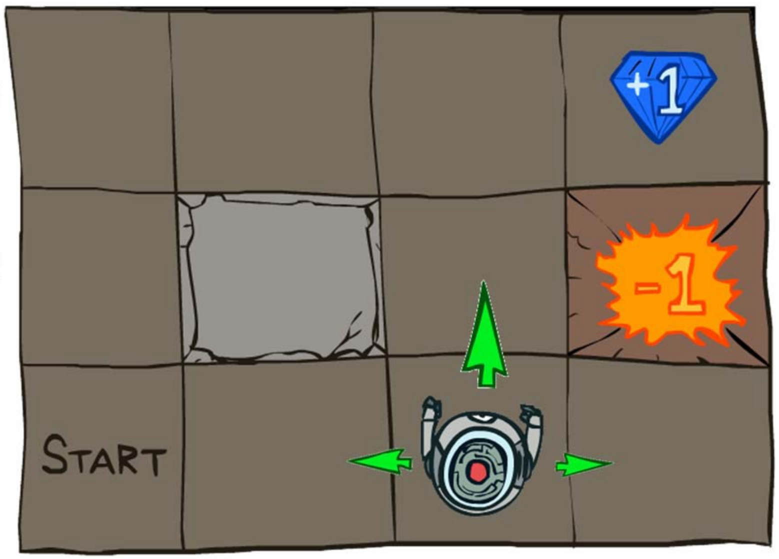
R(s) = -0.04 for every non-terminal state

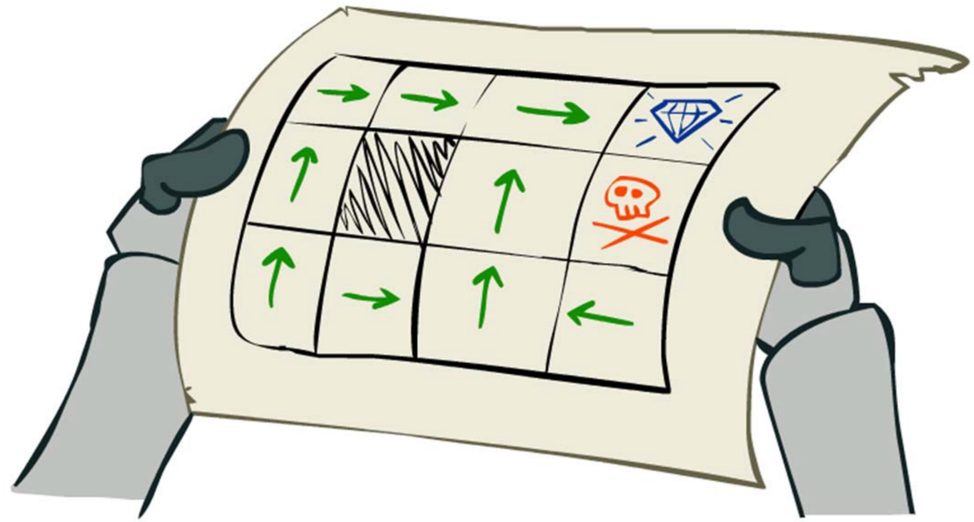
Transition model:



Source: P. Abbeel and D. Klein

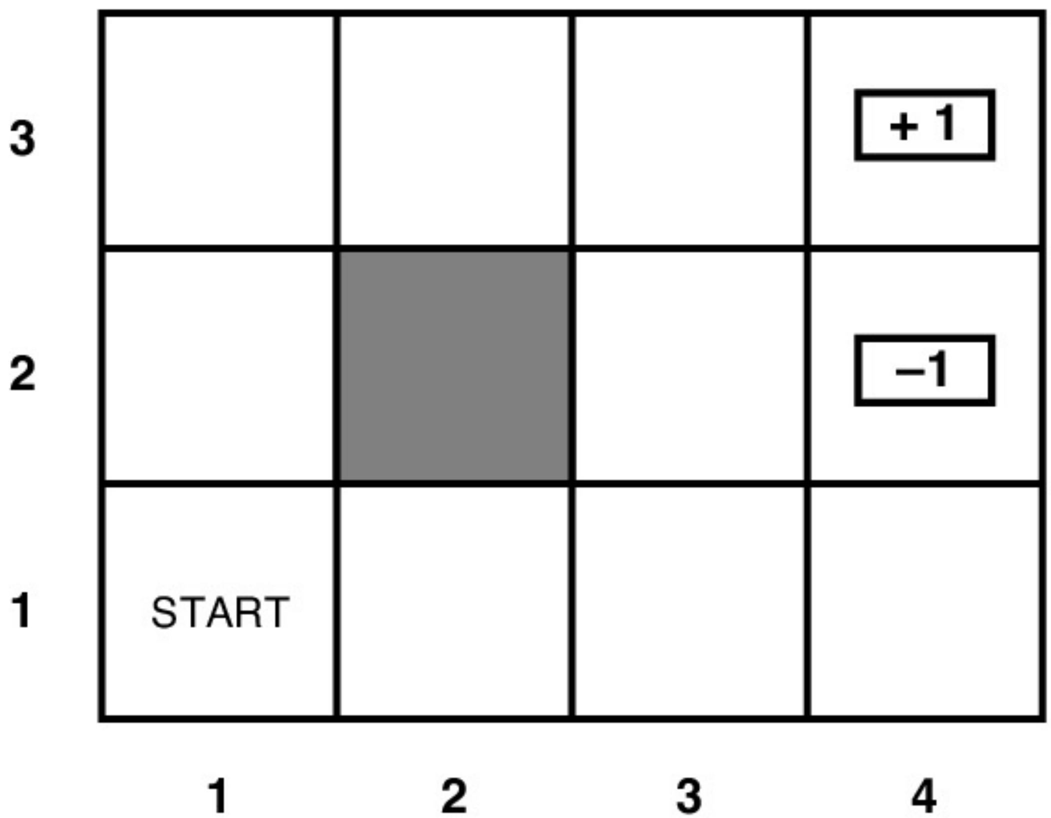
Goal: Policy



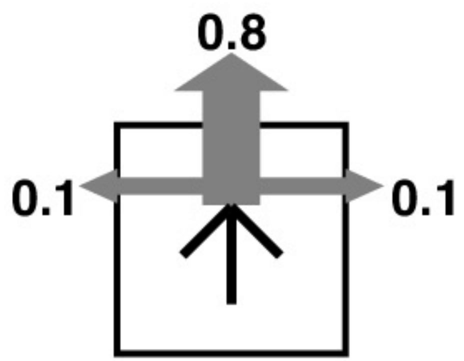


Source: P. Abbeel and D. Klein

Grid World

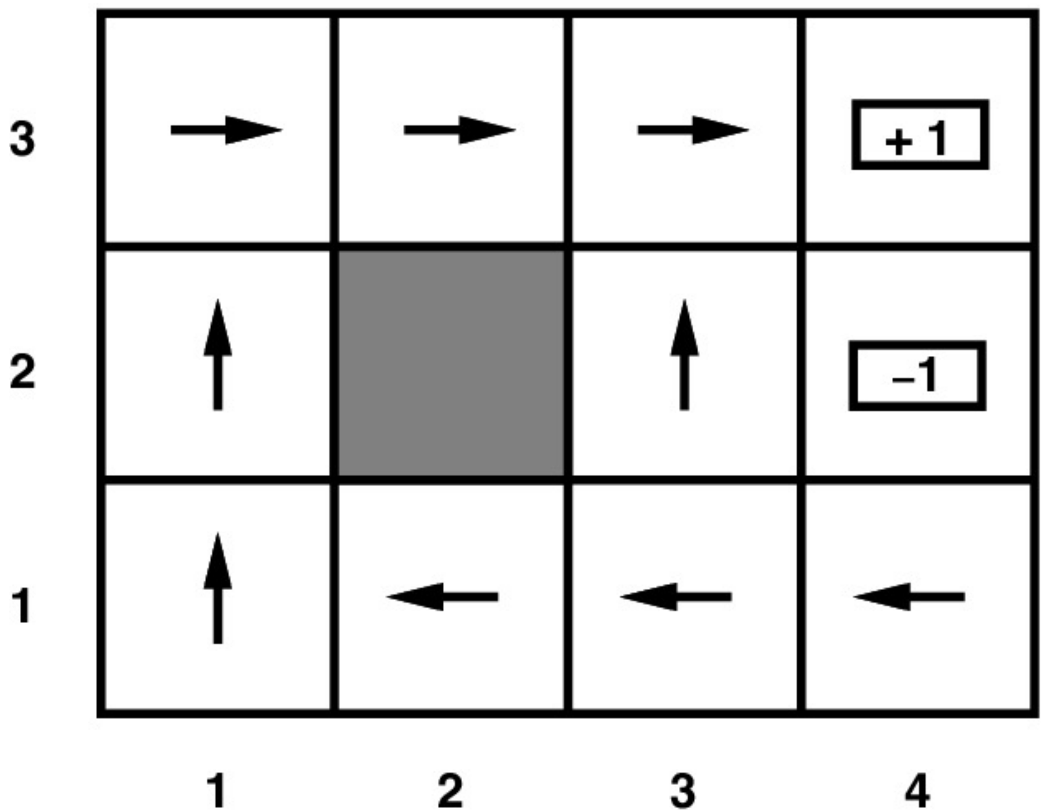


Transition model:



R(s) = -0.04 for every non-terminal state

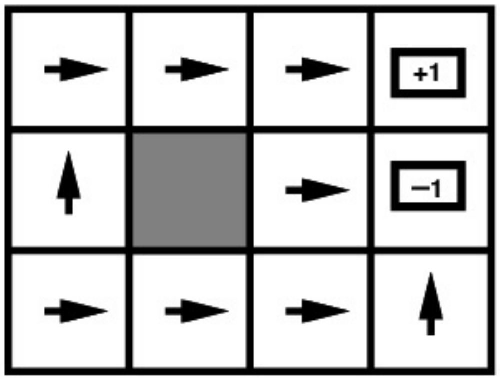
Grid World



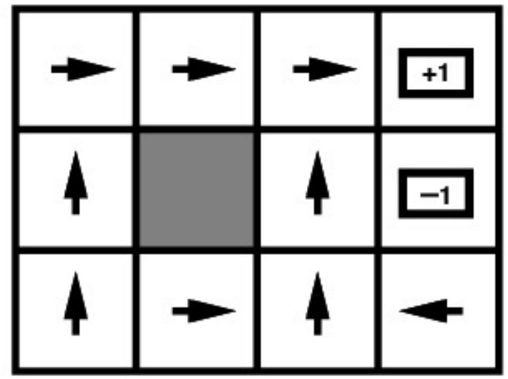
Optimal policy when R(s) = -0.04 for every non-terminal state

Grid World

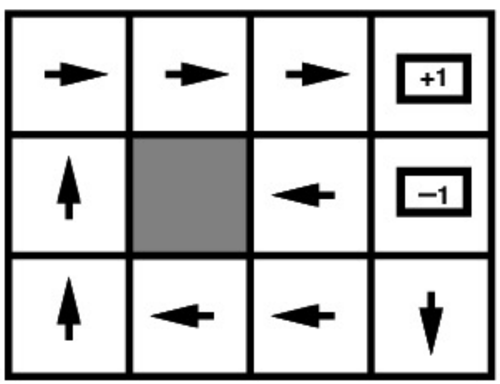
- Optimal policies for other values of R(s):



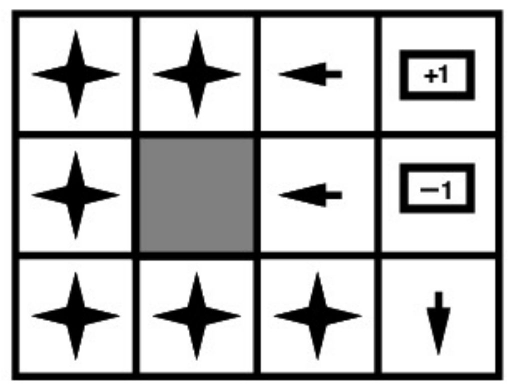
$R(s) < -1.6284$



$-0.4278 < R(s) < -0.0850$



$-0.0221 < R(s) < 0$



$R(s) > 0$

Solving MDPs

- MDP components:
  - **States** s
  - **Actions** a
  - **Transition model** $P(s' \mid s, a)$
  - **Reward function** $R(s)$
- The solution:
  - **Policy** $\pi(s)$: mapping from states to actions
  - How to find the optimal policy?

Maximizing Expected Utility

- The optimal policy should maximize the *expected utility* over all possible state sequences produced by following that policy:

probability.

$$\sum_{\substack{\text{state sequences} \\ \text{starting from } s_0}} P(\text{sequence}) \underbrace{U(\text{sequence})}$$

- How to define the utility of a state sequence?
  - Sum of rewards of individual states
  - Problem: infinite state sequences

Utilities of State Sequences

- Normally, we would define the utility of a state sequence as the sum of the rewards of the individual states
- **Problem:** infinite state sequences
- **Solution:** *discount* the individual state rewards by a factor γ between 0 and 1:

$$U([s_0, s_1, s_2, \ldots]) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \ldots$$

$$= \sum_{t=0}^{\infty} \gamma^t R(s_t) \leq \frac{R_{\max}}{1-\gamma} \qquad (0 < \gamma < 1)$$

  - Sooner rewards count more than later rewards
  - Makes sure the total utility stays bounded
  - Helps algorithms converge

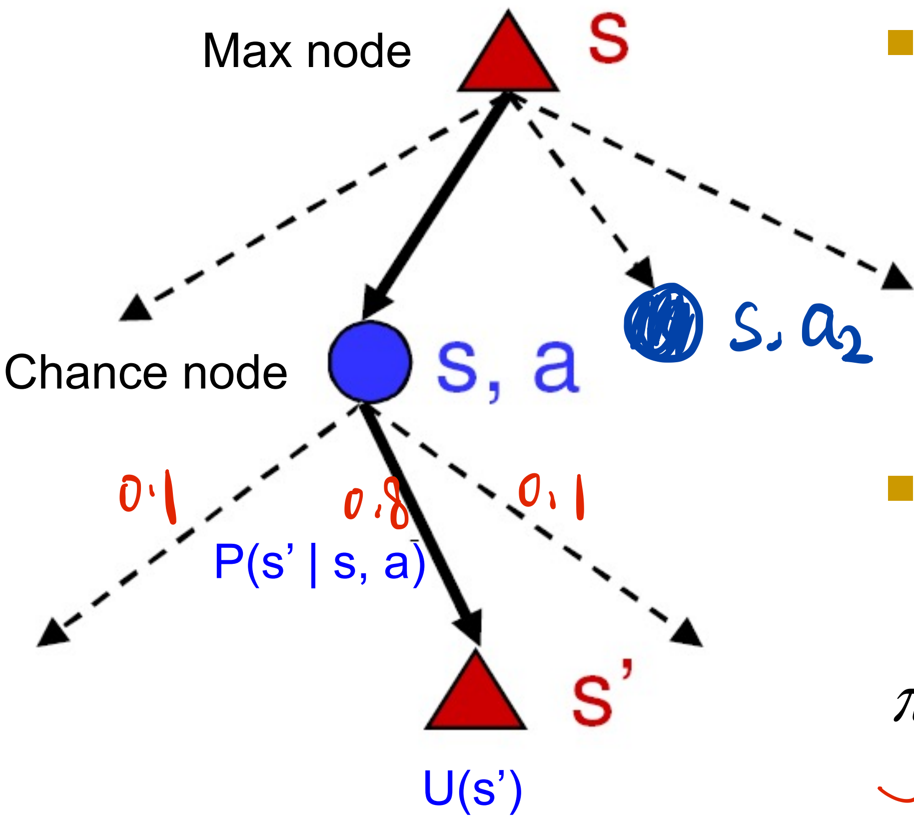
Utilities of States

- Expected utility obtained by policy π starting in state s:

$$U^{\pi}(s) = \sum_{\substack{\text{state sequences} \\ \text{starting from s}}} P(\text{sequence}) \underset{\substack{\uparrow \\ \text{discounted}}}{U}(\text{sequence})$$

- The "true" utility of a state, denoted U(s), is the expected sum of discounted rewards if the agent executes an *optimal* policy starting in state s
- Reminiscent of minimax values of states…

Finding the Utilities of States



- What is the expected utility of taking action **a** in state **s**?

3个 p×U 相加

$$\sum_{s'} P(s' \mid s,a)U(s')$$

- How do we choose the optimal action?

可能是 a/a₂

$$\pi^*(s) = \arg\max_{a \in A(s)} \sum_{s'} P(s' \mid s,a)U(s')$$

- What is the recursive expression for U(s) in terms of the utilities of its successor states?

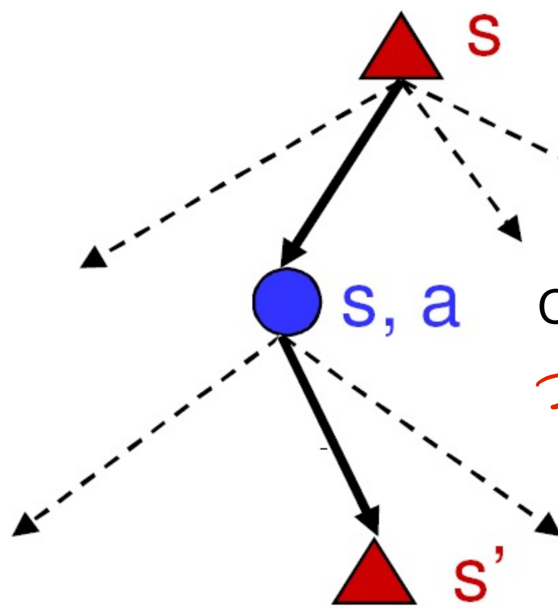
reward + discounted future rewards

$$U(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s,a)U(s')$$

The Bellman Equation

- Recursive relationship between the utilities of successive states:

$$\boxed{U(s) = R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U(s')}$$



Receive reward R(s)

Choose optimal action a
for best U(s')

End up here with P(s' | s, a)
Get utility U(s')
(discounted by γ)

The Bellman Equation

- Recursive relationship between the utilities of successive states:

$$\boxed{U(s) = R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U(s')}$$

- For *N* states, we get *N* equations in *N* unknowns
  - Solving them solves the MDP
  - We could try to solve them through expectimax search, but that would run into trouble with infinite sequences
  - Instead, we solve them algebraically
  - Two methods: **value iteration** and **policy iteration**

Method 1: Value Iteration

- Start out with every *U*(*s*) = 0
- Iterate until convergence
  - During the *i*th iteration, update the utility of each state according to this rule:

$$U_{i+1}(s) \leftarrow R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U_i(s')$$

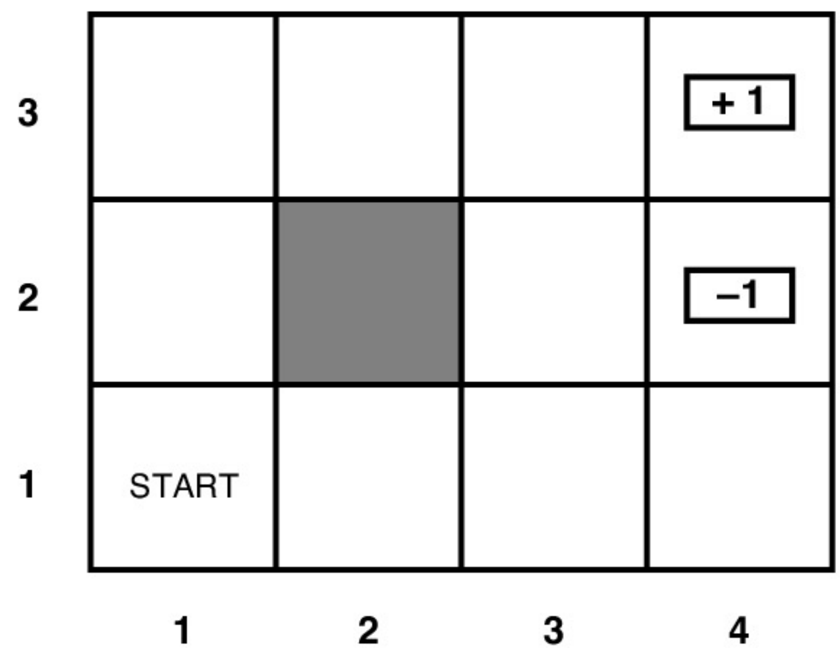
- In the limit of infinitely many iterations, guaranteed to find the correct utility values
  - In practice, don't need an infinite number of iterations…

Value Iteration

transition function.

- What effect does the update have?

$$U_{i+1}(s) \leftarrow R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' | s, a) U_i(s')$$



Method 2: Policy Iteration

- Start with some initial policy π_0 and alternate between the following steps:
  - **Policy evaluation:** calculate $U^{\pi_i}(s)$ for every state s
  - **Policy improvement:** calculate a new policy π_{i+1} based on the updated utilities

$$\pi^{i+1}(s) = \arg\max_{a \in A(s)} \sum_{s'} P(s' | s, a) U^{\pi_i}(s')$$

Policy Evaluation

- Given a fixed policy π, calculate $U^{\pi}(s)$ for every state *s*
- The Bellman equation for the optimal policy:

$$U(s) = R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' | s, a) U(s')$$

  - How does it need to change if our policy is fixed?

$$U^{\pi}(s) = R(s) + \gamma \sum_{s'} P(s' | s, \pi(s)) U^{\pi}(s')$$

  - Can solve a linear system to get all the utilities! ?
  - Alternatively, can apply the following update:

$$U_{i+1}(s) \leftarrow R(s) + \gamma \sum_{s'} P(s' | s, \pi_i(s)) U_i(s')$$

Summary

- Decision theory combines probability and utility theory
- A rational agent chooses the action with maximum expected utility
- Multi-attribute utility theory deals with utilities that depend on several attributes
- Decision networks extend BBN with additional nodes
- Making complex decisions – a sequence of decisions
- Markov decision processes assume Markov property
- Two methods for computing optimal policy
  - Value iteration
  - Policy iteration