Making Decisions

9

9 Making Decisions

9.1 Decision making agent

9.2 Preferences

9.3 Utilities

9.4 Decision networks

9.5 Sequential decision^{+}

9.6 Multiagent systems*

9.7 Game theory

Decision making agent

```
def DECISION-THEORETIC-AGENT(percept)
    persistent: belief-state, probabilistic beliefs about the current state of the world
                action, the agent's action

    Updated belief-state by decision-theoretic policy based on action and percept
    calculate outcome for actions
        given action descriptions and utility of current belief-state
    select action with highest expected utility
        given outcomes and utility information
    return action
```

Decision theories: an agent's choices

- Utility theory: worth or value
  utility function – preference ordering over a choice set
- Game theory: strategic interaction between rational decision-makers

Hint: AI $\rightarrow$ Economy $\rightarrow$ Computational economy

Making decisions under uncertainty

Suppose I believe the following:

$$
\begin{aligned}
P(A_{25} \text{ gets me there on time} | \ldots) &= 0.04 \\
P(A_{90} \text{ gets me there on time} | \ldots) &= 0.70 \\
P(A_{120} \text{ gets me there on time} | \ldots) &= 0.95 \\
P(A_{1440} \text{ gets me there on time} | \ldots) &= 0.9999
\end{aligned}
$$

Which action to choose?

Depends on my preferences for missing flight vs. airport cuisine, etc.

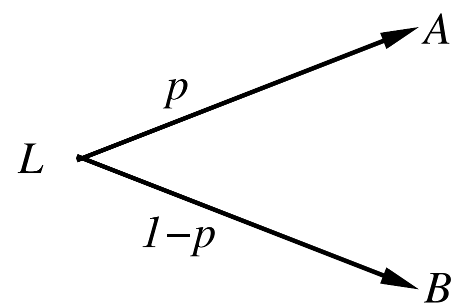
Utility theory is used to represent and infer preferences

Decision theory = probability theory + utility theory

Preferences

An agent chooses among prizes (A, B, etc.) and lotteries, i.e., situations with uncertain prizes

Lottery $L = [p, A;\ (1-p), B]$



In general, a lottery (state) L with possible outcomes $S_1, \cdots, S_n$ that occur with probabilities $p_1, \cdots, p_n$

$L = [p_1, S_1; \cdots;\ p_n, S_n]$

each outcome S_i of a lottery can be either an atomic state or another lottery

Preferences

Notation

$A \succ B$ $\quad$ A preferred to B
$A \sim B$ $\quad$ indifference between A and B
$A \succsim B$ $\quad$ B not preferred to A

Rational preferences

preferences of a rational agent must obey constraints
$\Rightarrow$ behavior describable as maximization of expected utility

Axioms of preferences

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

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

Substitutability

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

$$(A \succ B \Rightarrow [p, A;\; 1-p, C] \succ [p, B; 1-p, C])$$

Monotonicity

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

Decomposability

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

Rational preferences

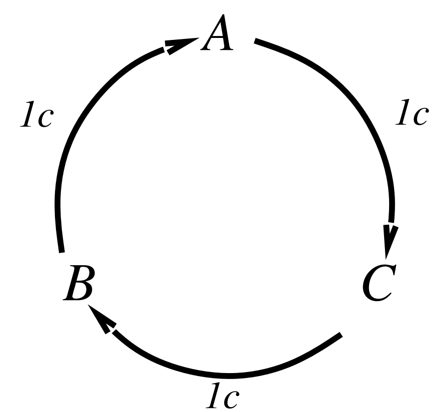
Violating the constraints leads to self-evident irrationality

E.g.: an agent with intransitive preferences can be induced to give away all its money

If $B \succ C$, then an agent who has C would pay (say) 1 cent to get B

If $A \succ B$, then an agent who has B would pay (say) 1 cent to get A

If $C \succ A$, then an agent who has A would pay (say) 1 cent to get C



Utilities

Preferences are captured by a utility function, $U(s)$
assigns a single number to express the desirability of a state

The expected utility of an action given the evidence, $EU(a|\mathbf{e})$
the average utility value of the outcomes, weighted by the probability that the outcome occurs
$U(a|\mathbf{e}) = \sum_{s'} P(\textsc{Result}(a) = s'|a, \mathbf{e})U(s')$

Theorem (Ramsey, 1931; von Neumann and Morgenstern, 1944): Given preferences satisfying the axioms, there exists a real-valued function U s.t.

$$U(A) \geq U(B) \quad \Leftrightarrow \quad A \succsim B$$
$$U([p_1, S_1; \ \ldots \ ; \ p_n, S_n]) = \sum_i \ p_i U(S_i)$$

Maximizing expected utility

MEU principle

Choose the action that maximizes expected utility

$a^* = argmax_a EU(a|\mathbf{e})$

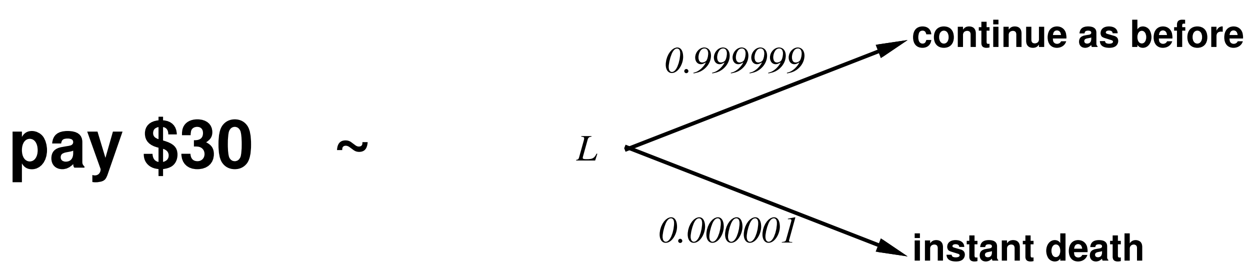
Note: an agent can be entirely rational (consistent with MEU) without ever representing or manipulating utilities and probabilities

E.g., a lookup table for perfect tic-tac-toe

Utility function

Utility map states (lotteries) to real numbers. Which numbers?

Standard approach to the assessment of human utilities
compare a given state A to a standard lottery L_p that has
"best possible prize" $u_\top$ with probability p
"worst possible catastrophe" $u_\perp$ with probability $(1-p)$
adjust lottery probability p until $A \sim L_p$



Say, pay a monetary value on life

Utility scales

Normalized utilities: $u_\top = 1.0$, $u_\perp = 0.0$

Micromorts (micro-mortality): one-millionth chance of death
useful for Russian roulette, paying to reduce product risks, etc.

QALYs: quality-adjusted life years
useful for medical decisions involving substantial risk

Note: behavior is **invariant** w.r.t. +ve linear transformation

$$U'(x) = k_1 U(x) + k_2 \quad \text{where } k_1 > 0$$

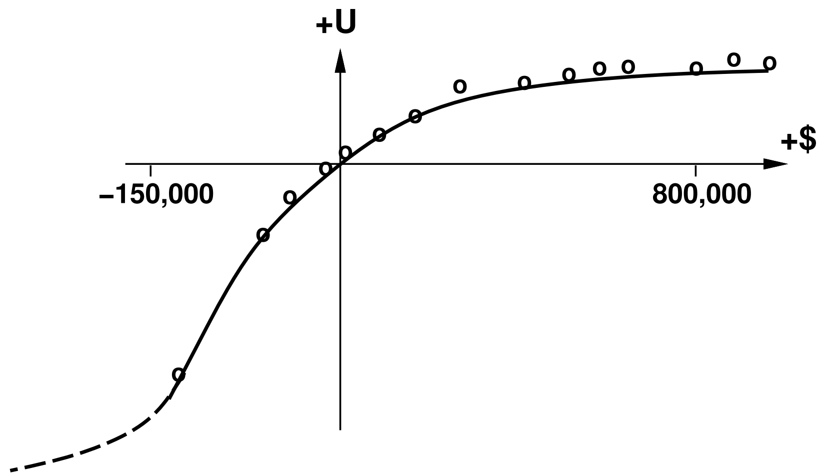
Money

Money does **not** behave as a utility function

Given a lottery L with expected monetary value $EMV(L)$,
usually $U(L) < U(EMV(L))$, i.e., people are risk-averse

Utility curve: for what probability p am I indifferent between a prize x and a lottery $[p, \$M;\ (1-p), \$0]$ for large M?

Typical empirical data, extrapolated with risk-prone behavior



Multiattribute utility

How can we handle utility functions of many variables $X_1 \ldots X_n$?
E.g., what is $U(Deaths, Noise, Cost)$?

How can complex utility functions be assessed from preference behaviour?

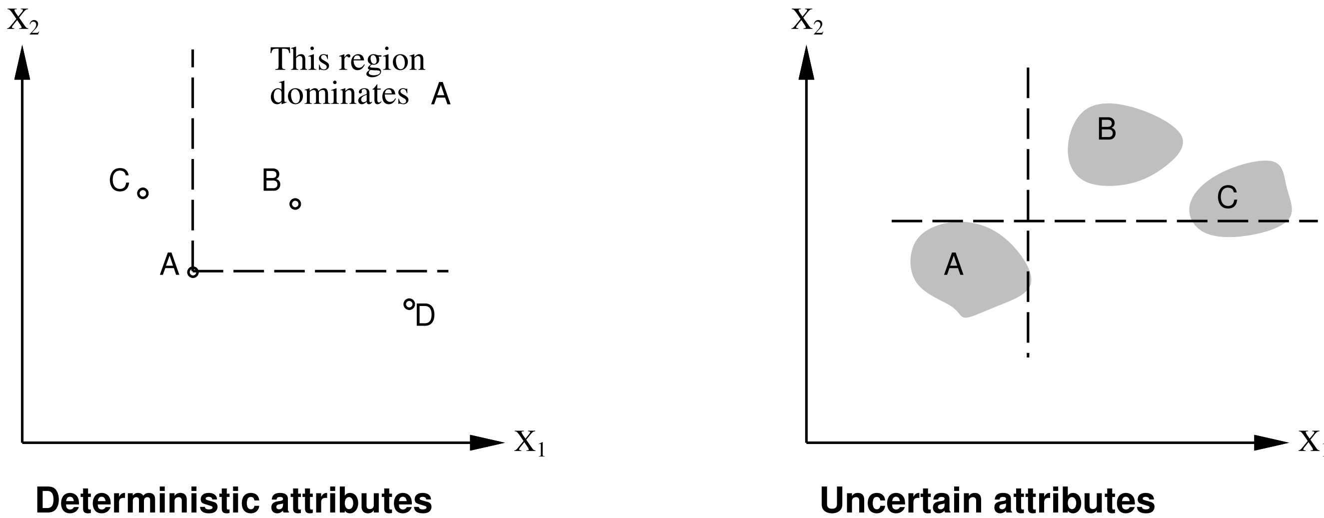
Idea 1: identify conditions under which decisions can be made without complete identification of $U(x_1, \ldots, x_n)$

Idea 2: identify various types of **independence** in preferences and derive consequent canonical forms for $U(x_1, \ldots, x_n)$

Strict dominance

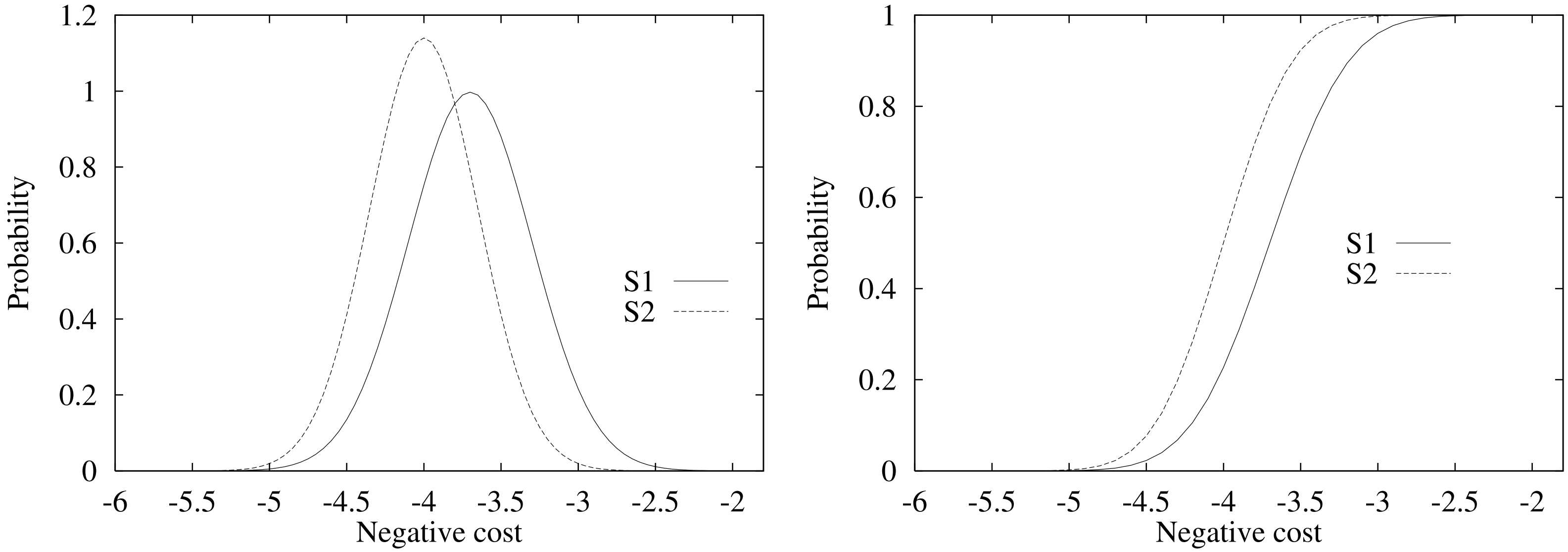
Typically define attributes such that U is monotonic in each

Strict dominance: choice B strictly dominates choice A iff
$\forall\, i \;\; X_i(B) \geq X_i(A)$ (and hence $U(B) \geq U(A)$)



Strict dominance seldom holds in practice

Stochastic dominance



Distribution p_1 **stochastically dominates** distribution p_2 iff

$$\forall t \quad \int_{-\infty}^{t} p_1(x)dx \leq \int_{-\infty}^{t} p_2(t)dt$$

If U is monotonic in x, then A_1 with outcome distribution p_1 stochastically dominates A_2 with outcome distribution p_2:

$$\int_{-\infty}^{\infty} p_1(x)U(x)dx \geq \int_{-\infty}^{\infty} p_2(x)U(x)dx$$

Multiattribute: stochastic dominance on all attributes $\Rightarrow$ optimal

Stochastic dominance

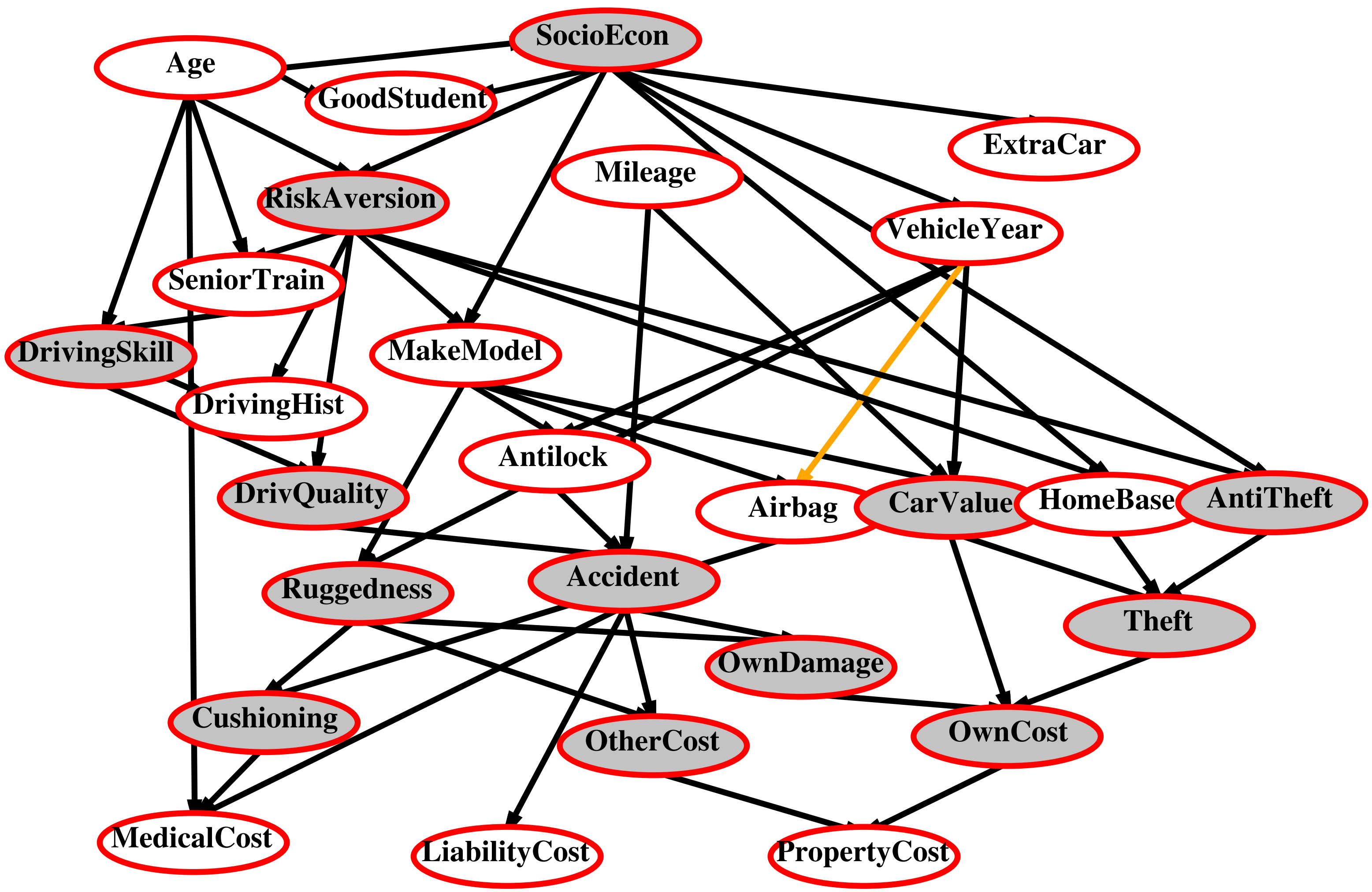
Stochastic dominance can often be determined without exact distributions using **qualitative** reasoning

E.g., construction cost increases with distance from city
S_1 is closer to the city than S_2
$\Rightarrow S_1$ stochastically dominates S_2 on cost

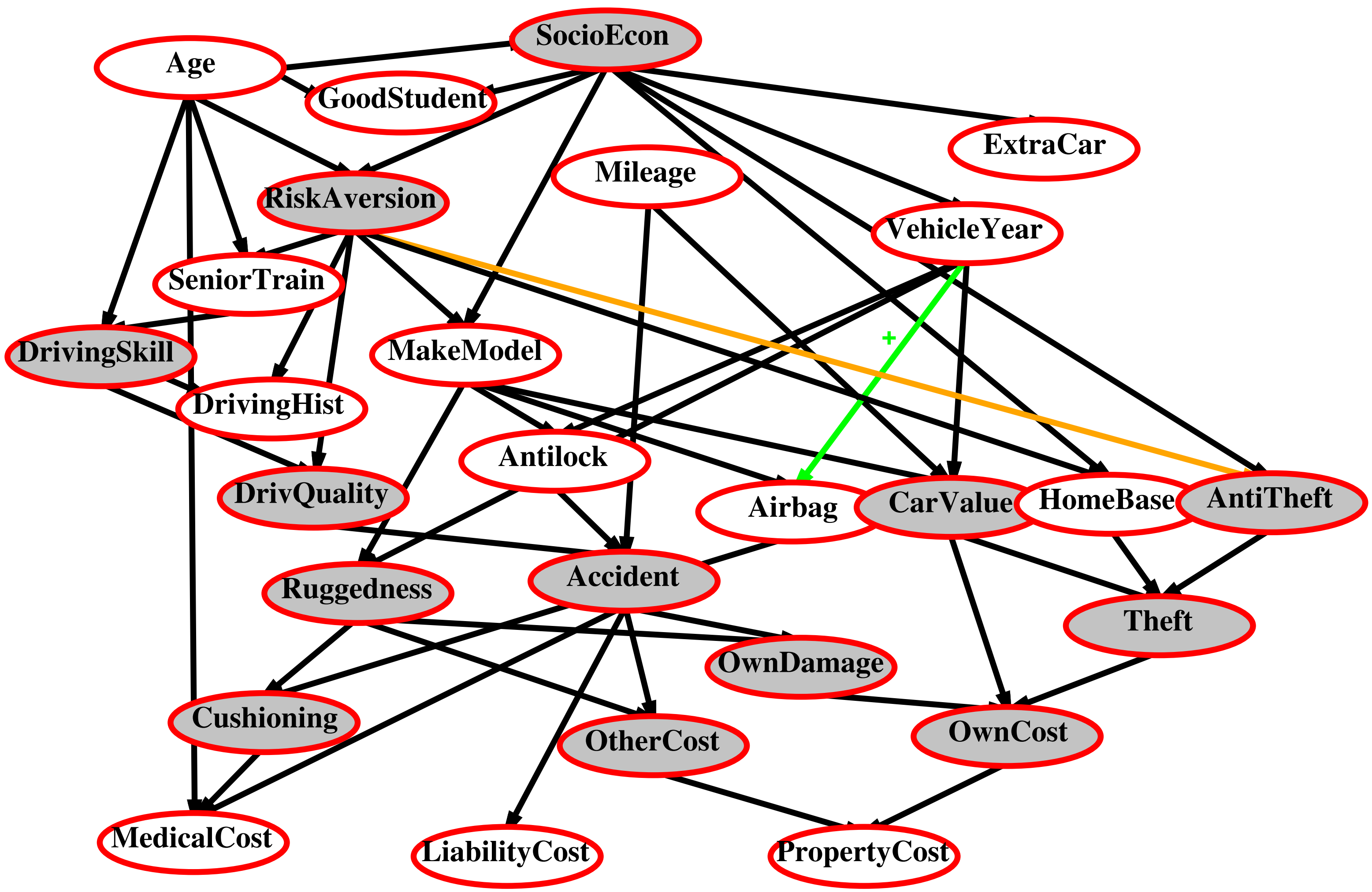
E.g., injury increases with collision speed

Can annotate belief networks with stochastic dominance information
$X \xrightarrow{+} Y$ (X positively influences Y) means that
For every value $\mathbf{z}$ of Y's other parents $\mathbf{Z}$
$\forall\, x_1, x_2 \quad x_1 \geq x_2 \Rightarrow \mathbf{P}(Y|x_1, \mathbf{z})$ stochastically dominates $\mathbf{P}(Y|x_2, \mathbf{z})$

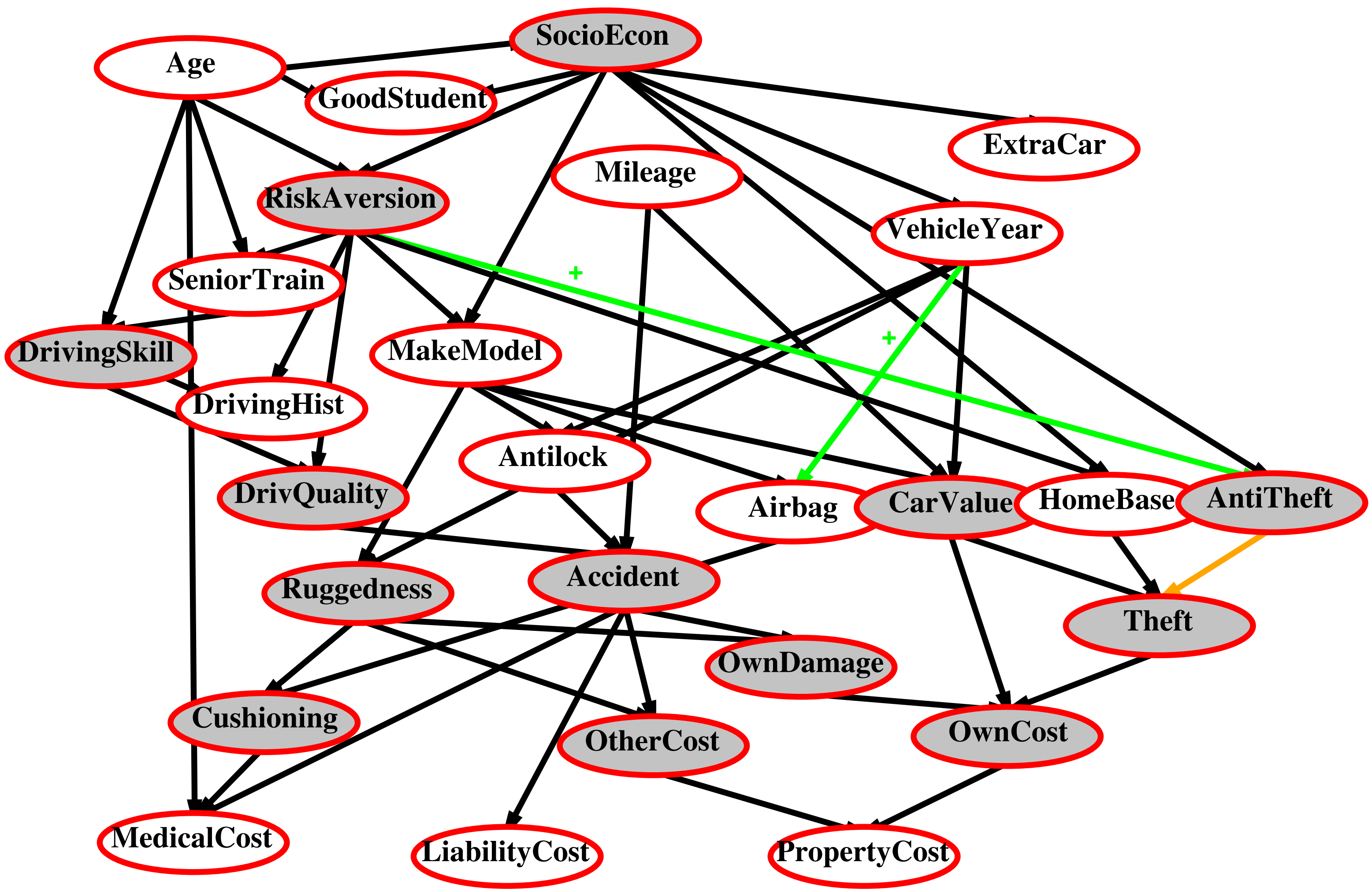
Label the arcs + or −



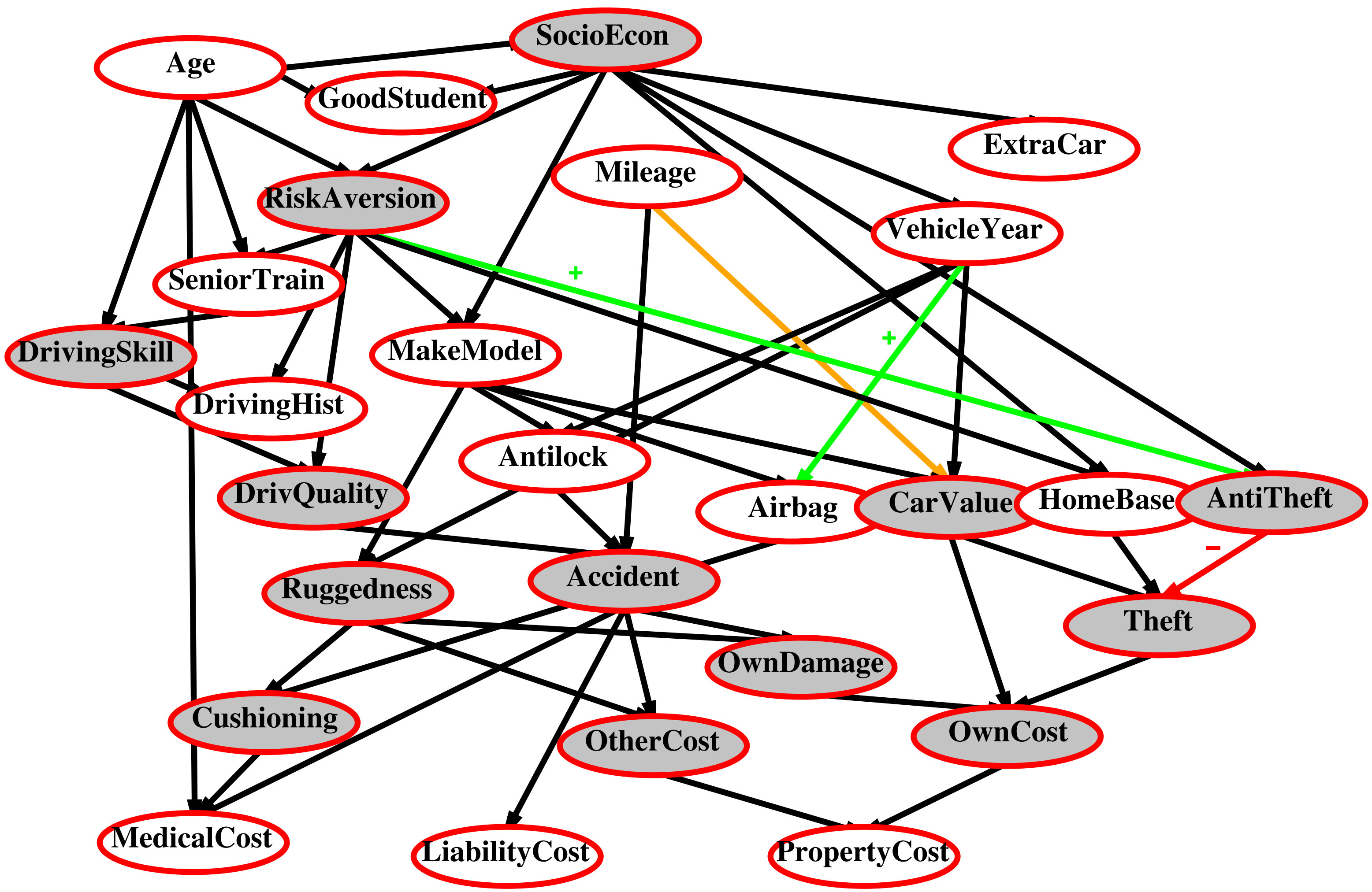
Label the arcs + or −



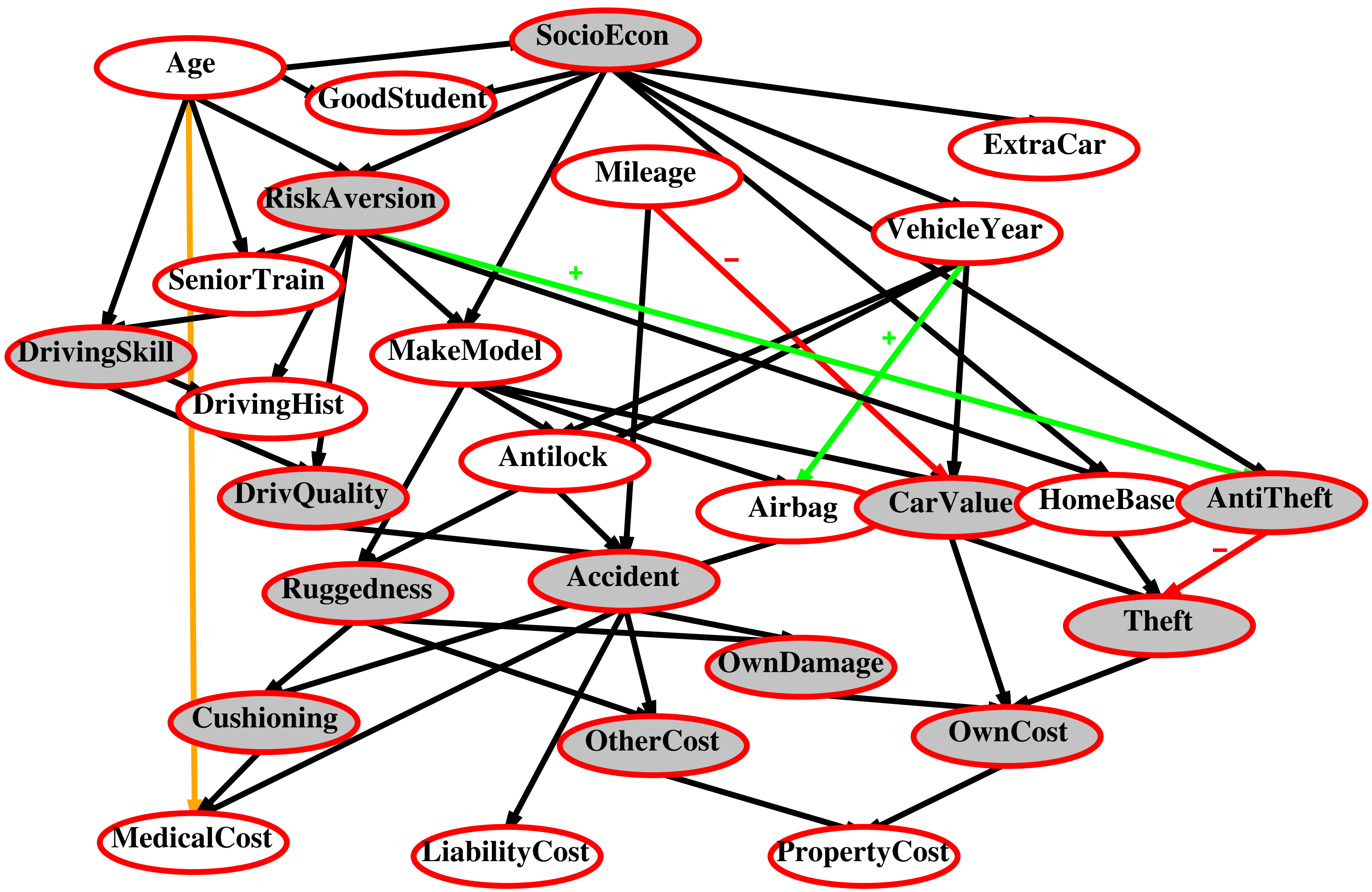
Label the arcs + or −



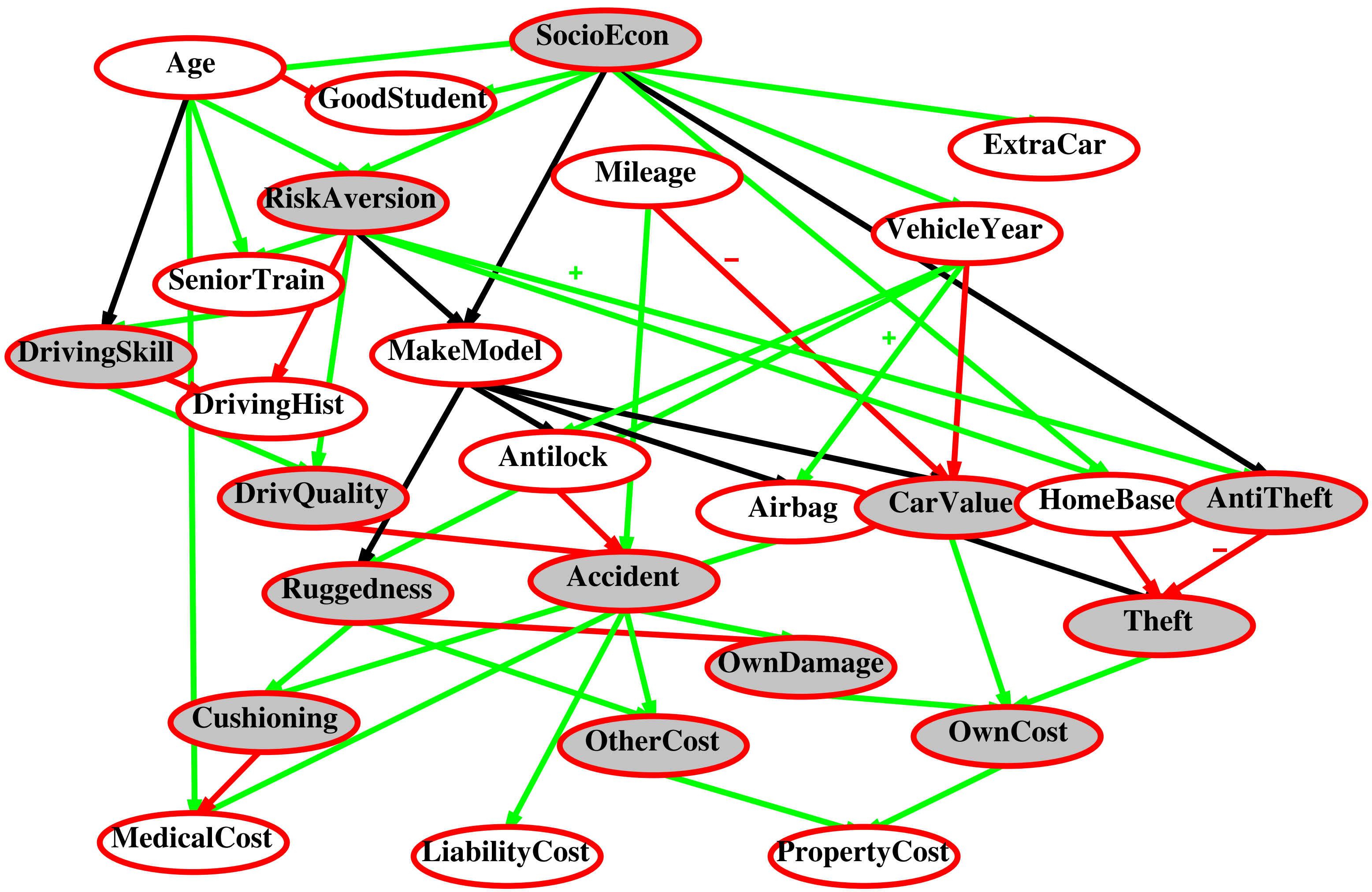
Label the arcs + or −



Label the arcs + or −



Label the arcs + or −



Preference structure: deterministic

X_1 and X_2 preferentially independent of X_3 iff
preference between $\langle x_1, x_2, x_3 \rangle$ and $\langle x'_1, x'_2, x_3 \rangle$
does not depend on x_3

E.g., $\langle Noise, Cost, Safety \rangle$:
$\langle$20,000 suffer, \$4.6 billion, 0.06 deaths/mpm$\rangle$ vs.
$\langle$70,000 suffer, \$4.2 billion, 0.06 deaths/mpm$\rangle$

Theorem (Leontief, 1947): if every pair of attributes is P.I. of its complement, then every subset of attributes is P.I of its complement: mutual P.I..

Theorem (Debreu, 1960): mutual P.I.
$\Rightarrow$ $\exists$ additive value function

$$V(S) = \Sigma_i V_i(X_i(S))$$

Hence assess n single-attribute functions; often a good approximation

Preference structure: stochastic

Need to consider preferences over lotteries

$\mathbf{X}$ is utility-independent of $\mathbf{Y}$ iff
preferences over lotteries in $\mathbf{X}$ do not depend on $\mathbf{y}$

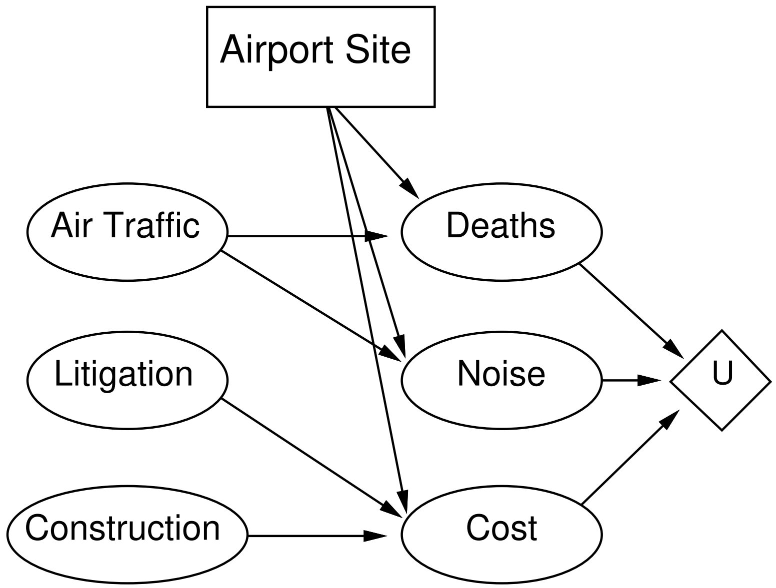
Mutual U.I.: each subset is U.I of its complement
$\Rightarrow \exists$ multiplicative utility function:

$$
\begin{aligned}
U = & k_1U_1 + k_2U_2 + k_3U_3 \\
& + k_1k_2U_1U_2 + k_2k_3U_2U_3 + k_3k_1U_3U_1 \\
& + k_1k_2k_3U_1U_2U_3
\end{aligned}
$$

Routine procedures and software packages for generating preference tests to identify various canonical families of utility functions

Decision networks

Add action nodes (rectangles) and utility nodes to belief networks
to enable rational decision making



Decision networks algorithm

1. Set the evidence variables for the current state

2. For each possible value of the decision node
    (a) Set the decision node to that value
    (b) Calculate the posterior probabilities for the parent nodes of the utility node
    using a standard probabilistic inference algorithm
    (c) Calculate the resulting utility for the action

3. Return MEU action

The value of information

Idea: compute the value of acquiring each possible piece of evidence
Can be done **directly from decision network**

Example: buying oil drilling rights
Two blocks A and B, exactly one has oil, worth k
Prior probabilities 0.5 each, mutually exclusive
Current price of each block is $k/2$
"Consultant" offers accurate survey of A. Fair price?
Solution: compute the expected value of information
= expected value of best action given the information
minus expected value of best action without information
Survey may say "oil in A" or "no oil in A", **prob. 0.5 each** (given!)
= [$0.5 \times$ value of "buy A" given "oil in A"
+ $0.5 \times$ value of "buy B" given "no oil in A"]
– 0

$$= (0.5 \times k/2) + (0.5 \times k/2) - 0 = k/2$$

General formula

Current evidence E, current best action α
Possible action outcomes S_i, potential new evidence E_j

$$EU(\alpha|E) = \max_a \Sigma_i \, U(S_i) \; P(S_i|E, a)$$

Suppose we knew $E_j = e_{jk}$, then we would choose $\alpha_{e_{jk}}$ s.t.

$$EU(\alpha_{e_{jk}}|E, E_j = e_{jk}) = \max_a \Sigma_i \, U(S_i) \; P(S_i|E, a, E_j = e_{jk})$$

E_j is a random variable whose value is *currently* unknown
$\Rightarrow$ must compute expected gain over all possible values:

$$VPI_E(E_j) = \left(\Sigma_k \, P(E_j = e_{jk}|E) EU(\alpha_{e_{jk}}|E, E_j = e_{jk})\right) - EU(\alpha|E)$$

(VPI = value of perfect information)

Properties of VPI

Nonnegative — in expectation, not post hoc

$$\forall\, j, E \;\; VPI_E(E_j) \geq 0$$

Nonadditive—consider, e.g., obtaining E_j twice

$$VPI_E(E_j, E_k) \neq VPI_E(E_j) + VPI_E(E_k)$$

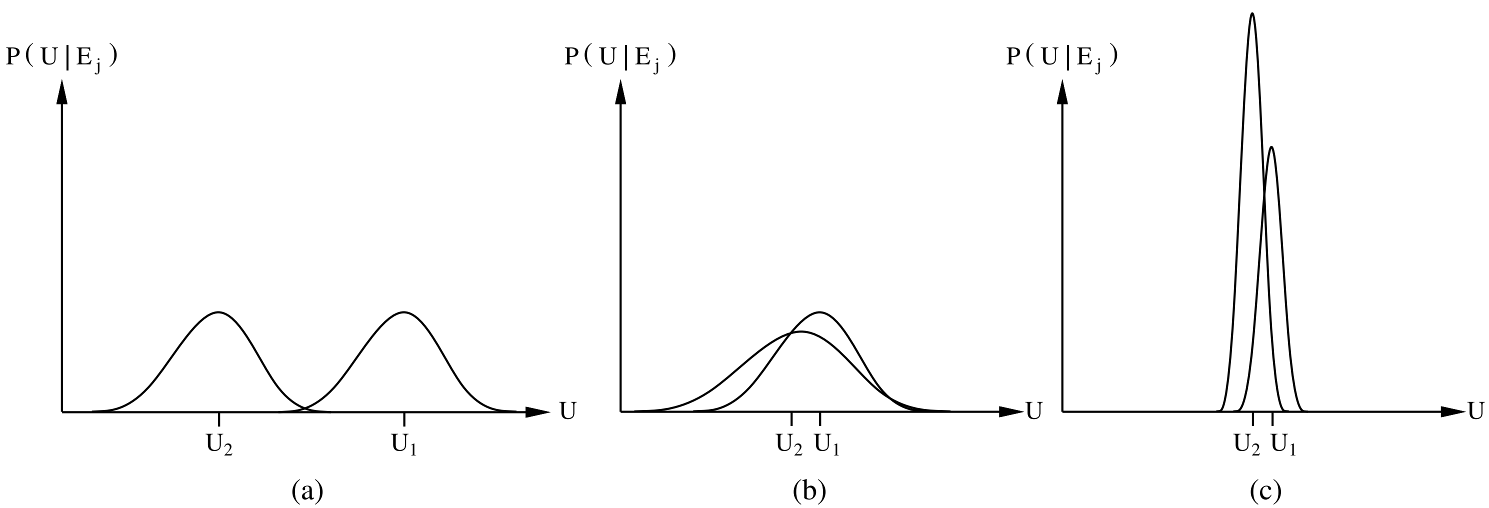
Order-independent

$$VPI_E(E_j, E_k) = VPI_E(E_j) + VPI_{E,E_j}(E_k) = VPI_E(E_k) + VPI_{E,E_k}(E_j)$$

Note: when more than one piece of evidence can be gathered, maximizing VPI for each to select one is not always optimal
$\Rightarrow$ evidence-gathering becomes a **sequential** decision problem

Qualitative behaviors

a) Choice is obvious, information worth little
b) Choice is nonobvious, information worth a lot
c) Choice is nonobvious, information worth little



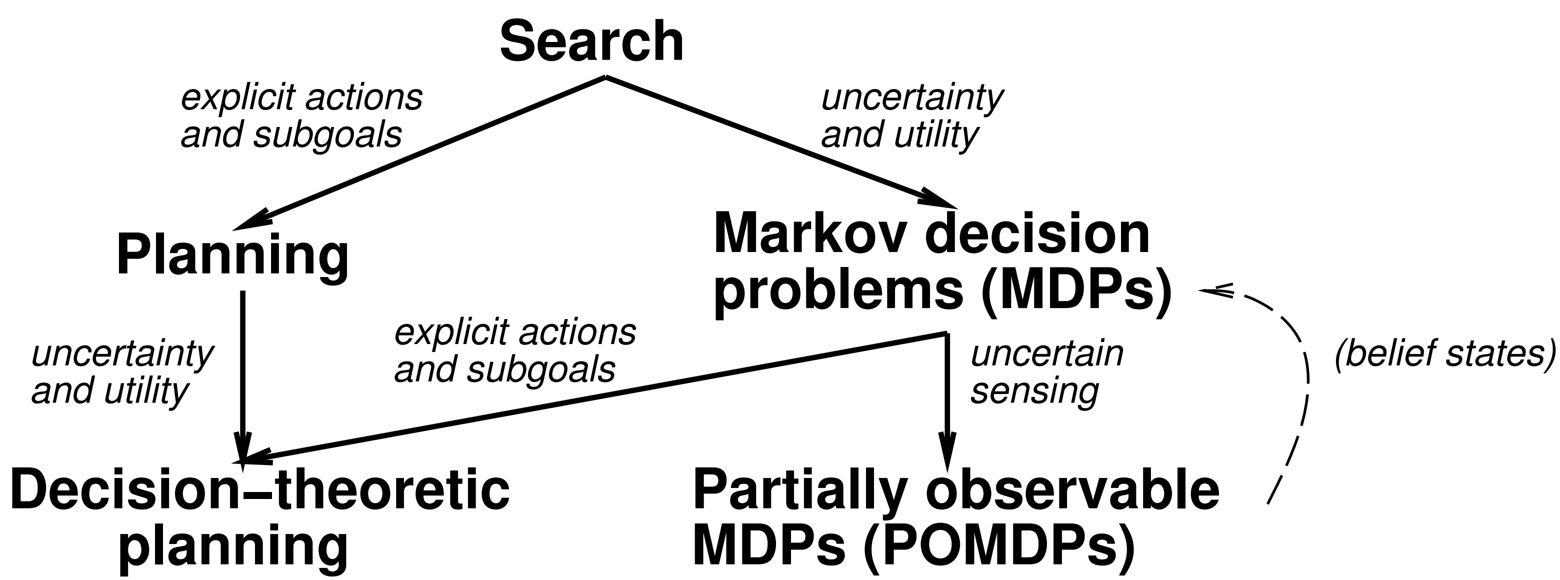
Information-gathering agent

def INFORMATION-GATHERING-AGENT(*percept*)
    persistent: D, a decision network

    integrate *percept* into D
    $j \leftarrow$ the value that maximizes $VPI(E_j)/Cost(E_j)$
    if $VPI(E_j) > Cost(E_j)$
        then return REQUEST(E_j)
    else return the best action from D

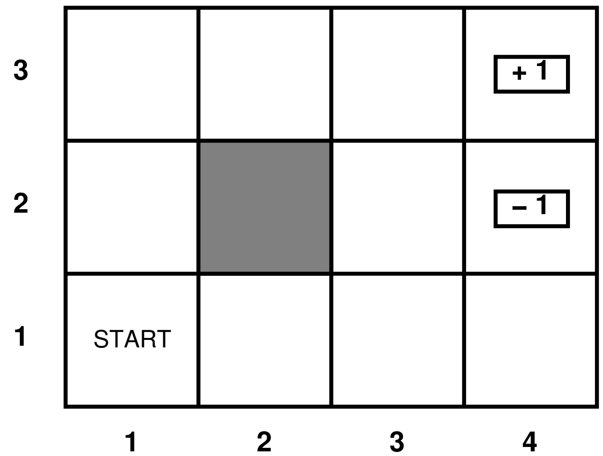
Sequential decision$^+$

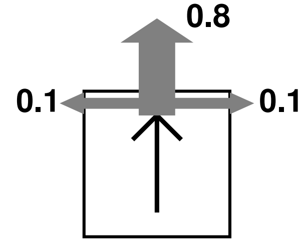
Sequential decision problems: utilities depend on a sequence of decisions; incorporating utilities, uncertainty and sensing; including search and planning as special cases



MDP (Markov decision process): observable, stochastic environment with a Markovian transition model and additive rewards

MDP





Say, $[Up, Up, Right, Right, Right]$ with probability $0.8^5 = 0.32768$

States $s \in S$, actions $a \in A$

Model $T(s, a, s') \equiv P(s'|s, a) =$ probability that a in s leads to s'

Reward function $R(s)$ (or $R(s, a)$, $R(s, a, s')$)

$$= \begin{cases} -0.04 & \text{(small penalty) for nonterminal states} \\ \pm 1 & \text{for terminal states} \end{cases}$$

Solving MDPs

In search problems, the solution is to find an optimal *sequence*

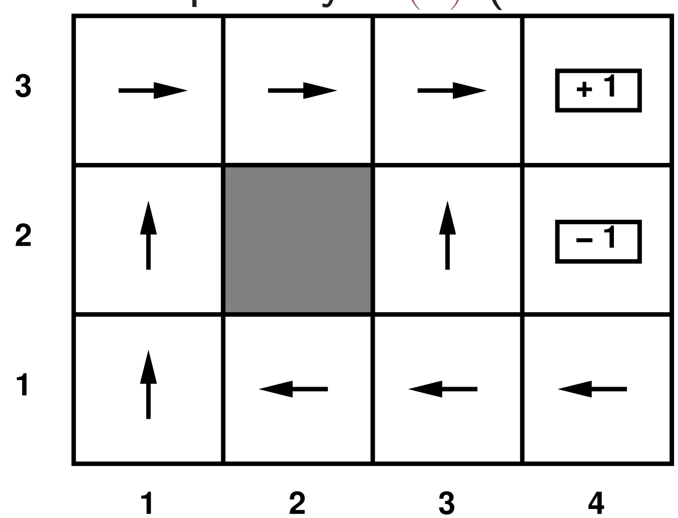
In MDPs, the solution is to find an optimal policy $\pi(s)$

i.e., the best action for every possible state s

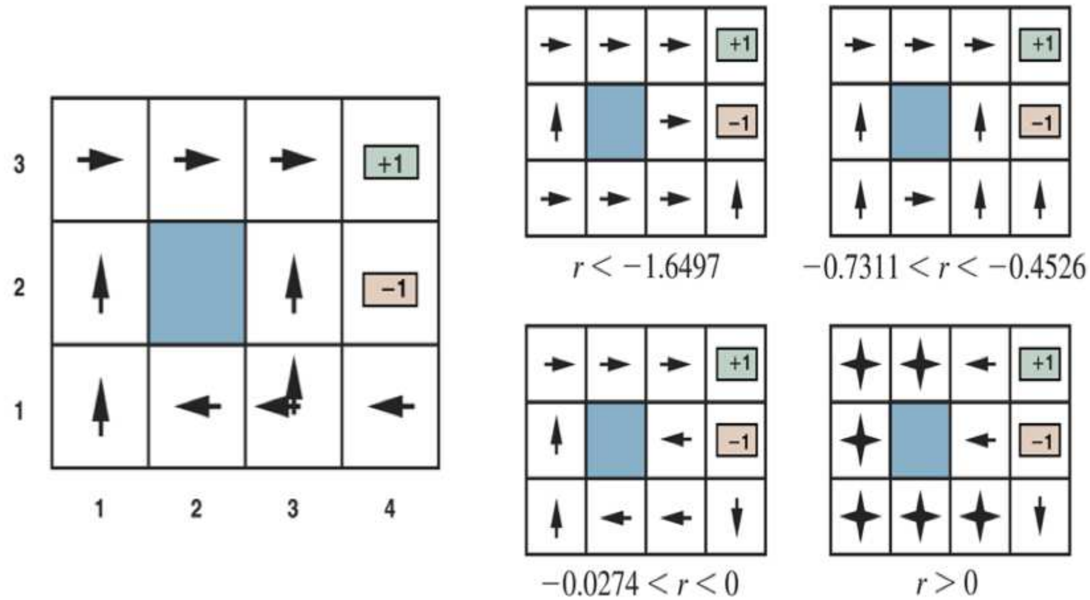
(because one can't predict where one will end up)

The optimal policy maximizes (say) the *expected sum of rewards*

Optimal policy when state penalty $R(s)$ (r in the picture) is -0.04:



Risk and reward



Two optimal policies in state $(3, 1)$, and policies for four different ranges of r

Utility of state sequences

Need to understand preferences between *sequences* of states

Typically consider stationary preferences on reward sequences

$$[r, r_0, r_1, r_2, \ldots] \succ [r, r'_0, r'_1, r'_2, \ldots] \;\Leftrightarrow\; [r_0, r_1, r_2, \ldots] \succ [r'_0, r'_1, r'_2, \ldots]$$

Theorem: there are only two ways to combine rewards over time

1) *Additive* utility function
$U([s_0, s_1, s_2, \ldots]) = R(s_0) + R(s_1) + R(s_2) + \cdots$

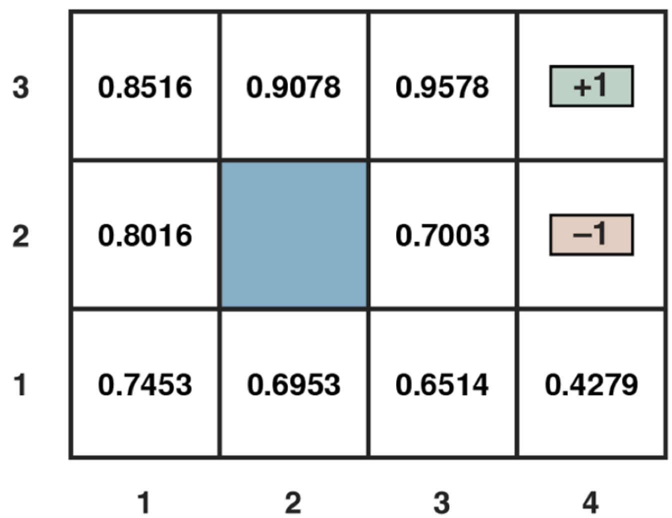
2) *Discounted* utility function
$U([s_0, s_1, s_2, \ldots]) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots$
where γ is the discount factor (describing the preference of an agent for current rewards over future rewards)

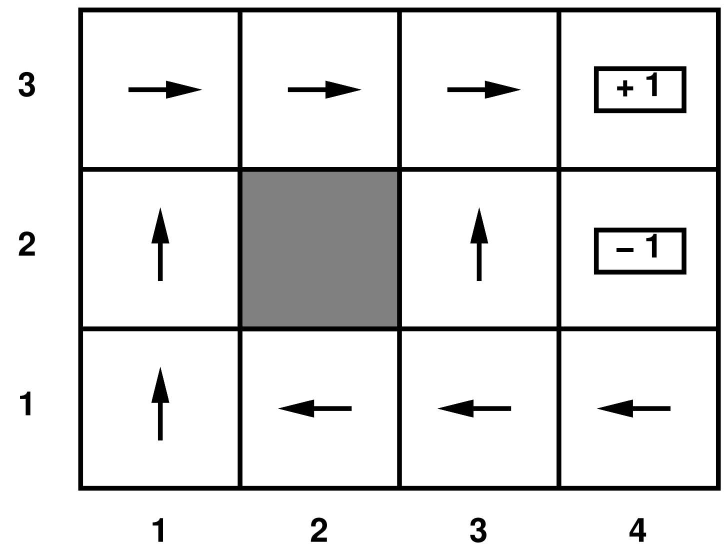
Utility of states

Utility of a *state* (aka. its *value*)

$U(s) =$ <u>expected (discounted) sum of rewards (until termination)</u>
<u>assuming optimal actions</u>

Given the utilities of the states, choosing the best action is just MEU
maximize the expected utility of the immediate successors





γ=1, higher for states closer to the exit $+1$

Utility of states#

Problem: infinite lifetimes $\rightarrow$ additive utilities are infinite

1) Finite horizon: termination at a *fixed time* T
$\rightarrow$ nonstationary policy: $\pi(s)$ depends on time left

2) Absorbing state(s): w/ prob. 1, agent eventually "dies" for any π
$\rightarrow$ expected utility of every state is finite

3) Discounting: assuming $\gamma < 1$, $R(s) \leq R_{\max}$,

$$U([s_0, \ldots s_\infty]) = \Sigma_{t=0}^{\infty}\gamma^t R(s_t) \leq R_{\max}/(1-\gamma)$$

Smaller $\gamma \Rightarrow$ shorter horizon

4) Maximize system gain = average reward per time step
Theorem: optimal policy has constant gain after initial transient
E.g., taxi driver's daily scheme cruising for passengers

Bellman equation

Definition of the utility of states leads to a simple relationship among utilities of neighboring states

expected sum of rewards
$= \text{current reward}$
$+ \gamma \times$ expected sum of rewards after taking best action

Bellman equation (1957)

$$U(s) = R(s) + \gamma \max_a \Sigma_{s'} U(s')T(s,a,s')$$

$$\begin{aligned} U(1,1) = & -0.04 \\ & + \gamma \max\{0.8U(1,2) + 0.1U(2,1) + 0.1U(1,1), && \textit{up} \\ & \quad 0.9U(1,1) + 0.1U(1,2) && \textit{left} \\ & \quad 0.9U(1,1) + 0.1U(2,1) && \textit{down} \\ & \quad 0.8U(2,1) + 0.1U(1,2) + 0.1U(1,1)\} && \textit{right} \end{aligned}$$

One equation per state $= n$ nonlinear equations in n unknowns

Value iteration algorithm

Idea: Start with arbitrary utility values
Update to make them locally consistent with Bellman equation
Everywhere locally consistent $\Rightarrow$ global optimality
Repeat for every s simultaneously until "no change" — Bellman update

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

Value iteration algorithm#

```
def VALUE-ITERATION(mdp,ε)
   inputs: mdp, an MDP with states S, actions A(s), transition
   model  P(s' | s, a), rewards R(s, a, s'),discount γ
             ε, the maximum error allowed in the utility of any state
   local variables: U, U', vectors of utilities for states in S, initially zero
                       δ, the maximum relative change in the utility of any state

   repeat
       U ← U';δ ← 0
       for each state s in S do
           U'[s] ← max_{a ∈ A(s)} BELLMAN-VALUE(mdp, s, a, U)
   until δ ≤ ε(1 − γ) / γ
   return U
```

Convergence*

Define the max-norm $||U|| = \max_s |U(s)|$
so $||U - V||$ = maximum difference between U and V

Let U^t and U^{t+1} be successive approximations to the true utility U

Theorem: For any two approximations U^t and V^t

$$||U^{t+1} - V^{t+1}|| \leq \gamma \, ||U^t - V^t||$$

I.e., any distinct approximations must get closer to each other
so, in particular, any approximation must get closer to the true U
and value iteration converges to a unique, stable, optimal solution

Theorem: if $||U^{t+1} - U^t|| < \epsilon$, then $||U^{t+1} - U|| < 2\epsilon\gamma/(1-\gamma)$
I.e., once the change in U^t becomes small, we are almost done.

MEU policy using U^t may be optimal long before convergence of values

Policy iteration

Howard (1960): search for an optimal policy and utility values simultaneously

Algorithm

$\pi \leftarrow$ an arbitrary initial policy
repeat until no change in π
compute utilities given π
update π as if utilities were correct (i.e., local MEU)

To compute utilities given a fixed π (value determination)

$$U(s) = R(s) + \gamma \sum_{s'} U(s') T(s, \pi(s), s'), \quad \text{for all } s$$

i.e., n simultaneous linear equations in n unknowns, solve in $O(n^3)$

Note: Reinforcement learning algorithms operate by performing such updates based on the observed transitions made in an initially unknown environment

Bandit problems$^{+\#}$

One-armed bandit (slot machine): a gambler can insert a coin, pull the lever, and collect the winnings (if any)
n-armed (multi-armed) bandit: behind each of n (independent) lever is a fixed but unknown probability distribution of winnings
Bernoulli bandit: each of n-armed produces a reward of 0 or 1 with a fixed but unknown probability $\Leftarrow$ formal model of sequential decision

E.g., deciding which of n possible new treatments to try to cure a disease

The tradeoff between <u>exploitation</u> (to get the machine the highest expected payoff) and <u>exploration</u> (to get more information about the expected payoffs of the other machines)

Markov reward process

Defined each arm M_i as a Markov reward process (MRP): a special MDP with only one possible action a_i

– states S_i

– transition model $P_i\left(s' \mid s, a_i\right)$

– reward $R_i\left(s, a_i, s'\right)$

The arm defines a distribution over sequences of rewards $R_{i,0}$, $R_{i,1}$, . . . (random variables)

Example: two-arms bandit

Consider two arms M, M_1, the discount factor $\gamma = 0.5$. Pulling arms yield the sequences of rewards
M: 0, 2, 0, 7.2, 0, 0, $\ldots$
M_1: 1, 1, 1, 1, 1, 1, $\ldots$

Utility (total discounted reward) for each arm
$U(M) = (1.0 \times 0) + (0.5 \times 2) + \left(0.5^2 \times 0\right) + \left(0.5^3 \times 7.2\right) = 1.9$
$U\left(M_1\right) = \Sigma_{t=0}^{\infty} 0.5^t = 2.0$

Seem like the best choice is to go with M_1. But starting with M and then switching to M_1 after the fourth reward gives the sequence
$S =$0, 2, 0, 7.2, 1, 1, 1, 1, 1, 1, $\ldots$

$U(S) = (1.0 \times 0) + (0.5 \times 2) + \left(0.5^2 \times 0\right) + \left(0.5^3 \times 7.2\right) + \Sigma_{t=4}^{\infty} 0.5^t = 2.025$

The strategy S is optimal: all other switching times give less reward

One-arm bandit

Consider two arms M, M_λ, the discount factor γ, yielding the sequences of rewards
M: R_0, R_1, R_2, . . .
M_λ: λ, λ, λ, . . .   (constant)

Equivalent to one arm M that produces rewards R_0, R_1, R_2, . . . and cost λ for each pull

– pulling arm M is equivalent to not pulling M_λ, so it gives up a reward of λ each time

Pull the first arm T times (0, 1, . . . , $T-1$, the stopping time is T)

An optimal strategy is to run arm M up to time T and then switches to M_λ for the rest of time

– if $T=0$ then choosing M_λ immediately
– if $T=\infty$ then never choosing M_λ

Gittins index

Consider λ s.t. an optimal strategy is exactly indifferent (tipping point) between

(a) running up to the best possible stopping time and then switching to M_λ forever, and

(b) choosing M_λ immediately

$$\max_{T>0} E\left[\left(\Sigma_{t=0}^{T-1}\gamma^t R_t\right) + \Sigma_{t=T}^{\infty}\gamma^t\lambda\right] = \Sigma_{t=0}^{\infty}\gamma^t\lambda \Longleftrightarrow$$

Theorem $\lambda = \max_{T>0}\frac{E\left(\Sigma_{t=0}^{T-1}\gamma^t R_t\right)}{E\left(\Sigma_{t=0}^{T-1}\gamma^t\right)}$ (Gittins index of M)

Optimal policy for any bandit problem: pull the arm that has the highest Gittins index, then update the Gittins indices

– computing the first Gittins index $O(n)$ time

(index of arm depends only on the properties of that arm)

– computing each decision after the first one $O(n)$

(Gittins indices of arms that are not selected remain unchanged)

Gittins index

T	1	2	3	4	5	6
R_t	0	2	0	7.2	0	0
$\sum \gamma^t R_t$	0.0	1.0	1.0	1.9	1.9	1.9
$\sum \gamma^t$	1.0	1.5	1.75	1.875	1.9375	1.9687
ratio	0.0	0.6667	0.5714	1.0133	0.9806	0.9651

For $0 < \lambda \leq 1.0133$, the optimal policy collects the first four rewards from M and then switches to M_λ
For $\lambda > 1.0133$, the optimal policy always chooses M_λ

Hint: Approximately optimal bandit policies are needed for more realistic problems

Gittins index

Consider each arm M of n-armed bandit at each state s, the agent has two choices

either continue with as before, or quit and receive an infinite sequence of λ-rewards

$\Rightarrow$ turn M into an MDP

optimal policy is just the optimal stopping rule for M

equal to the value of an infinite sequence of λ-rewards

i.e., $\lambda/(1-\gamma)$ — λ unknown

$\Leftarrow$ restart MDP

at the tipping point, the choice to get an infinite sequence of λ-rewards $=$ the choice to go back and restart from its initial state s, a new MDP M^s

solving M^s by any of the MDP algorithms, say value iteration, a value of 2.0266 for the start state

Have $\lambda = 2.0266 \cdot (1-\gamma) = 1.0133$ as before

POMDP*

POMDP (partially observable MDP has an observation model $O(s, e)$ defining the probability that the agent obtains evidence e when in state s

Agent does not know which state it is in
$\rightarrow$ makes no sense to talk about policy $\pi(s)$

Theorem (Astrom, 1965): the optimal policy in a POMDP is a function $\pi(b)$, where b is the belief state (probability distribution over states)

Can convert a POMDP into an MDP in belief-state space, where
$T(b, a, b')$ is the probability that the new belief state is b'
given that the current belief state is b and the agent does a
I.e., essentially a filtering update step

POMDP

Solutions automatically include information-gathering behavior

If there are n states, b is an n-dimensional real-valued vector
$\rightarrow$ solving POMDPs is very (actually, PSPACE-) hard

The real world is a POMDP (with initially unknown T and O)

Decision theoretic planning*

Planner designed in terms of probabilities and utilities in the decision networks

– support a computationally tractable inference about plans and partial plans

– numeric values to individual goals, but measures lack any precise meaning

Using decision-theoretic planning allows designers to judge the effectiveness of the planning system

– specify a utility function over the entire domain and rank the plan results by desirability

– modular representations that separately specify preference information so as to allow a dynamic combination of relevant factors

Multiagent systems*

Multiagent systems (MAS): the environment contains multiple *actors*

- Multiagent planning
- Multiagent decision making
  - Game theory

Cooperation and coordination

– Adopt a convention before engaging in joint activity

– – a convention is any constraint on the selection of joint plans

– Otherwise, agents can use *communication* to achieve common knowledge of a feasible joint plan

Multiagent planning

Issue: concurrency — the plans of each agent may be executed simultaneously

agents take into account the way in which their own actions interact with the actions of other agents

Interleaved execution: the order of actions in the respective plans will be preserved

True concurrency: not a full serialized ordering of the actions, leave them partially ordered

Synchronization: there is a global clock that each agent has access to

concurrent constraint stating which actions must (not) be executed concurrently

Multiagent decision making

Benevolent agent assumption: one decision maker plans for the other agents, and tells them what to do

– require actors to synchronize their actions

Multiple decision-makers: the other actors (*counterparts*) are also decision makers

– they each have preferences and choose and execute their own plan

– all pursuing a common goal

Coordination: they are all pulling in the same direction

– the decision makers each pursue to the best of their abilities

$\Leftarrow$ game theory

Game theory

Recall: Games as adversarial search
the solution is a strategy
specifying a move for every opponent reply, with limited resources

Game theory: decisions making with multiple agents in uncertain environments
the solution is a policy (strategy profile) in which each player adopts a rational strategy

	deterministic	**chance**
perfect information	**chess, checkers, go, othello**	**backgammon monopoly**
imperfect information		**bridge, poker, scrabble nuclear war**

A brief history of game theory#

- Competitive and cooperative human interactions (Huygens, Leibniz, 17BC)
- Equilibrium (Cournot 1838)
- Perfect play (Zermelo, 1913)
- Zero-sum game (Von Neumann, 1928)
- *Theory of Games and Economic Behavior* (Von Neumann 1944)
- Nash equilibrium (non-zero-sum games) (Nash 1950) (the 1994 Nobel Memorial Prize in Economics)
- Mechanism design theory (auctions) (Hurwicz 1973, along with Maskin, and Myerson) (the 2007 Nobel Memorial Prize in Economics)

Trading Agents Competition (TAC) (since 2001)

Prisoner's dilemma

Two burglars, Alice and Bob, are arrested and imprisoned. Each prisoner is in solitary confinement with no means of communicating with the other. A prosecutor lacks sufficient evidence to convict the pair on the principal charge, and offers each a deal: if you testify against your partner as the leader of a burglary ring, you'll go free for being the cooperative one, while your partner will serve 10 years in prison. However, if you both testify against each other, you'll both get 5 years. Alice and Bob also know that if both refuse to testify they will serve only 1 year each for the lesser charge of possessing stolen property

should they testify or refuse??

Prisoner's dilemma

Single move game

- players: A, B
- actions: $testify$, $refuse$
- payoff (function): utility to each player for each combination of actions by all the players

– for single-move games: payoff matrix (strategic form)

– A strategy profile is an assignment of a strategy to each player

– – pure strategy - deterministic

	Alice :testify	Alice :refuse
Bob :testify	A = –5, B = –5	A=–10,B= 0
Bob:refuse	A= 0,B=–10	A = –1, B = –1

should they testify or refuse??

Dominant strategy

A dominant strategy is a strategy that dominates all others

a strategy s for player p strongly dominates strategy s' if the outcome for s is better for p than the outcome for s', for every choice of strategies by the other player(s)

a strategy s weakly dominates s' if s is better than s' on at least one strategy profile and no worse on any other

Note: it is rational to play a dominated strategy, and irrational not to play a dominant strategy if one exists

– being rational, Alice chooses the dominant strategy

– being clever and rational, Alice knows: Bob's dominant strategy is also to testify

Equilibrium

An outcome is Pareto optimal if there is no other outcome that all players would prefer

An outcome is Pareto dominated by another outcome if all players would prefer the other outcome

e.g., $(testify, testify)$ is Pareto dominated by $(-1, -1)$ outcome of $(refuse, refuse)$

A strategy profile forms an equilibrium if no player can benefit by switching strategies, given that every other player sticks with the same strategy

– local optimum in the policy space

Dominant strategy equilibrium: the combination of those strategies, when each player has a dominant strategy

Nash equilibrium

Nash equilibrium (NE) **theorem**: every game has at least one equilibrium

E.g., a dominant strategy equilibrium is a Nash equilibrium (in special case, the converse does not hold — Why??)

NE is a necessary condition for a solution
– it is not always a sufficient condition

In the simple case, playing NE guarantees that the player will not win in expectation

In complex games, determining how to tie against an NE may be difficult

If the opponent ever chooses suboptimal actions, then playing NE will result in victory in expectation

Zero-sum games

Two-players general-sum game is represented by two payoff matrices $\mathbf{A}=[a_{ij}]$ and $\mathbf{A}=[b_{ij}]$

if $a_{ij} = -b_{ij}$, called zero-sum game

(games in which the sum of the payoffs is always zero)

Mixed strategy - a randomized policy that selects actions according to a probability distribution

Maximin algorithm: a method for finding the optimal mixed strategy for two-player, zero-sum games

– apply the standard minimax algorithm

Maximin equilibrium of the game, and it is an NE

von Neumann zero-sum theorem: every two-player zero-sum game has a maximin equilibrium when you allow mixed strategies

NE in a zero-sum game is maximin for both players

Algorithms for finding Nash Equilibria

1. Input: a support profile

2. Enumerate all possible subsets of actions that might form mixed strategies

3. For each strategy profile enumerated in (2), check to see if it is an equilibrium

– Solving a set of equations and inequalities. For two players these equations are linear (and can be solved with basic linear programming); for n-players they are nonlinear

4. Output: an NE

Note: polynomial-time algorithms exist for special classes, such as two-player zero-sum games (three or more players NP-hard). For two-player non-zero-sum games, no polynomial-time (approximate) algorithm is known for finding an NE (PPAD-complete)

Other games*

Repeated games: players face the same choice repeatedly, but each time with knowledge of the history of all players? previous choices

E.g., the repeated version of the prisoner's dilemma

Sequential games: games consist of a sequence of turns that need not be all the same
– can be represented by a game tree (extensive form)
– add a distinguished player, chance, to represent stochastic games, specified as a probability distribution over actions

Bayes-Nash equilibrium: an equilibrium w.r.t. a player's prior probability distribution over the other players' strategies
Consider the other players are less than fully rational

Now, the most complete representations: partially observable, multiagent, stochastic, sequential, dynamic environments

Regret minimization*

Regret measures as the value of the difference between a made decision and the optimal decision

the regret for a sequence of strategies s_a consisting of always choosing action a would be, where T is the number of iterations

$$r^T(s_a) = \sum_{t=1}^{T} \left(v^t(a) - v^t\right)$$

For an algorithm to be no-regret, it must choose strategies in a way that guarantees that the reget value grows sublinearly for the optimal strategies

Counterfactual regret minimization*

Counterfactual regret minimization (CFR) for sequential games independently minimizes regret in each information set (state space)

– Vanilla CFR requires full traversals of the game tree, Monte Carlo CFR (MCCFR) needs only a portion of the tree to be traversed

– Regret-based pruning (RBP) prunes negative-regret actions from the tree traversal to speed it up

Variants of CFR (such as discounted or linear CFR) are the leading equilibrium-finding algorithm for imperfect-information games

Regret matching*

Minimax regret (i.e., minimax applied to regret) is to minimize the worst-case regret

Regret matching (RM) is a no-regret learning algorithm — the prob of an action is proportional to the *positive* regret on that action

on each iteration $t+1$, action $a \in A$ is selected according to probabilities

$$\sigma^{t+1}(a) = \frac{r_+^t(a)}{\Sigma_{a' \in A} r_+^t(a')}$$

where $r_+^t(a) = \max\{0, r^t(a)\}$

Theorem: When both players in a two-player zero-sum game use a no-regret learning algorithm, the average of the strategies played over all iterations converges to an NE

Counterfactual regret minimization algorithm*

def CFR(h,i,t,π_1,π_2) // with chance sampling

  persistent: I, the information set containing history h

    A, the set of actions

    r,s,σ, the tables of regret,stratgy,profile

  $\forall\ I, r_I[a] \leftarrow 0$ // Initialize cumulative regret tables

  $\forall\ I, s_I[a] \leftarrow 0$ // Initialize cumulative strategy tables

  $\sigma^1(I, a) \leftarrow 1/|A(I)|$ // Initialize initial profile

  $v_\sigma \leftarrow 0$ // Regret values for the profile σ

  $v_{\sigma_{I\to a}}[a] \leftarrow 0$ // For all $a \in A(I)$

  if h is terminal **then return** $u_i(h)$

   else if h is a chance node **then**

    Sample a single outcome $a \sim \sigma_c(h, a)$

    return $\mathrm{CFR}(h.a, i, t, \pi_1, \pi_2)$

Counterfactual regret minimization algorithm*

for $a \in A(I)$ **do**
  if $P(h) = 1$ **then**
    $v_{\sigma_{I \rightarrow a}}[a] \leftarrow \mathrm{CFR}(h.a, i, t, \sigma^t(I,a) \cdot \pi_1, \pi_2)$
  else if $P(h) = 2$ **then**
    $v_{\sigma_{I \rightarrow a}}[a] \leftarrow \mathrm{CFR}(h.a, i, t, \pi_1, \sigma^t(I,a) \cdot \pi_2)$
  $v_\sigma \leftarrow v_\sigma + \sigma^t(I,a) \cdot v_{\sigma_{I \rightarrow a}}[a]$
if $P(h) = i$ **then**
  for $a \in A(I)$ **do**
    $r_I[a] \leftarrow r_I[a] + \pi_{-i} \cdot (v_{\sigma_{I \rightarrow a}}[a] - v_\sigma)$
    $s_I[a] \leftarrow s_I[a] + \pi_i \cdot \sigma^t(I,a)$
  $\sigma^{t+1}(I) \leftarrow$ regret-matching values computed by regret table r_I
return v_σ

Example: Libratus

Recall: Imperfect information games

Porker: surpass human experts in the game of heads-up no-limit Texas hold'em, which has over 10^{160} decision points

Libratus: two-ply heads-up no-limit Texas hold'em poker

- Depended on game theory
  - NE-finding algorithms based on CFR
  - Information and action abstraction for similarity of subgame
- Did not depend on deep learning
  - Except for Deep CFR (NN for computing CFR)
  - Outperform deep learning-based DeepStack
  - AlphaZero can not win Texas hold'em

ReBel (2020): achieved superhuman performance in heads-up no-limit in 2020, extended AlphaZero by Nash equilibrium (with knowledge)

Example: DeepNash*

Stratego: 10^{535} states, larger than Texas hold'em (10175 times larger than Go)

– actions with no obvious link between action and outcome

– can not be broken down into sub-problems as in poker

– impossible to use AlphaGo-like or Libratus-like algorithms

– challenge all existing search techniques as the search space becomes intractable

DeepNash used an evolutionary (equivalent to reinforcement learning) game theory, without search, via self-play

– got up to a human expert level

R-NaD (Regularised Nash Dynamics) algorithm: converge to an (approximate) ϵ-Nash equilibrium

implemented using a deep convolutional network

Auctions*

Auction: a mechanism design for selling some goods to members of a pool of bidders

– inverse game theory

Given that agents pick rational strategies, what game should we design? e.g., cheap airline tickets

Ascending-bid (English auction)

1. The center starts by asking for a minimum (or reserve) bid
2. If some bidder is willing to pay that amount, then asks for some increment, and continues up from there
3. End when nobody is willing to bid anymore, then the last bidder wins the item

Auction design (e.g., efficiency) and implementation (algorithm)

Inverse auction

Given that the center picks a rational strategy, what game should we design?