CS 581

Advanced Artificial Intelligence

March 04, 2024

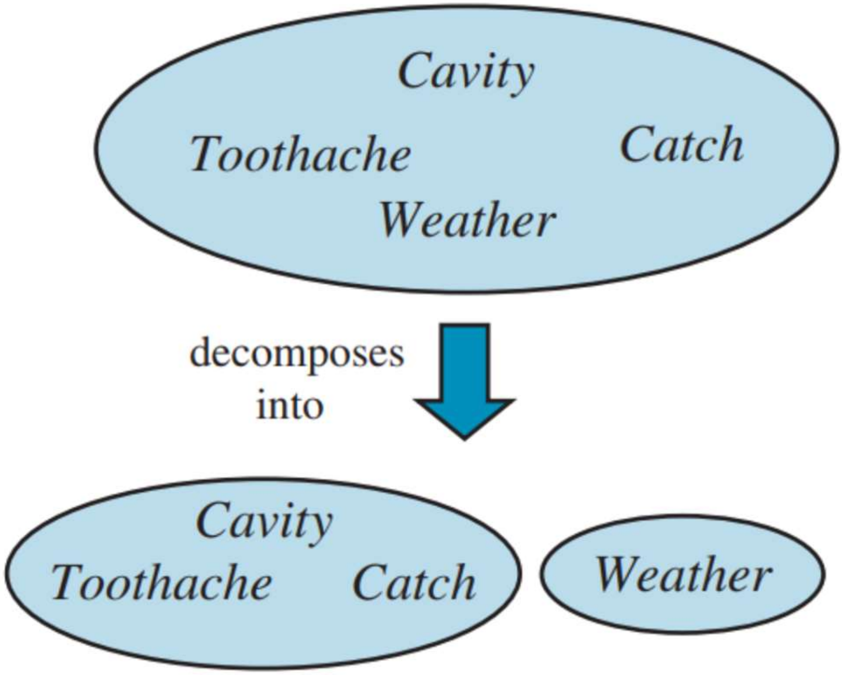
Announcements / Reminders

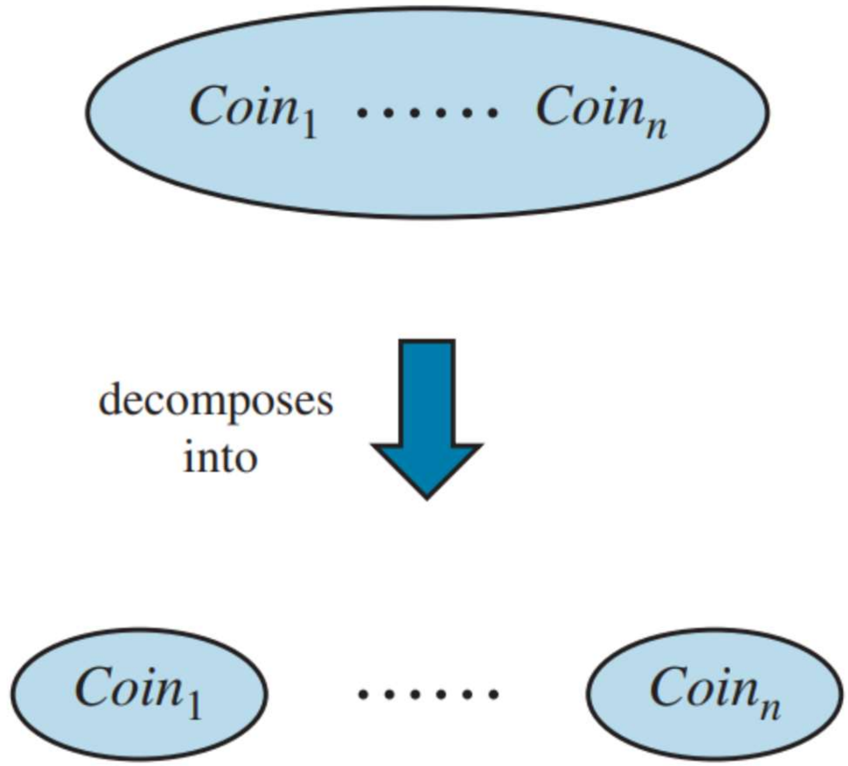
- **Please follow the Week 08 To Do List instructions (if you haven't already)**
- **Next week: Spring Break! No office hours.**
- **Programming Assignment #01: due on ~~Sunday 03/03~~ Tuesday 03/05 at 11:59 PM CST**

Plan for Today

- **Decision Networks**
- **Inference in Probabilistic Networks**

Factoring / Decomposition





Bayes' Rule

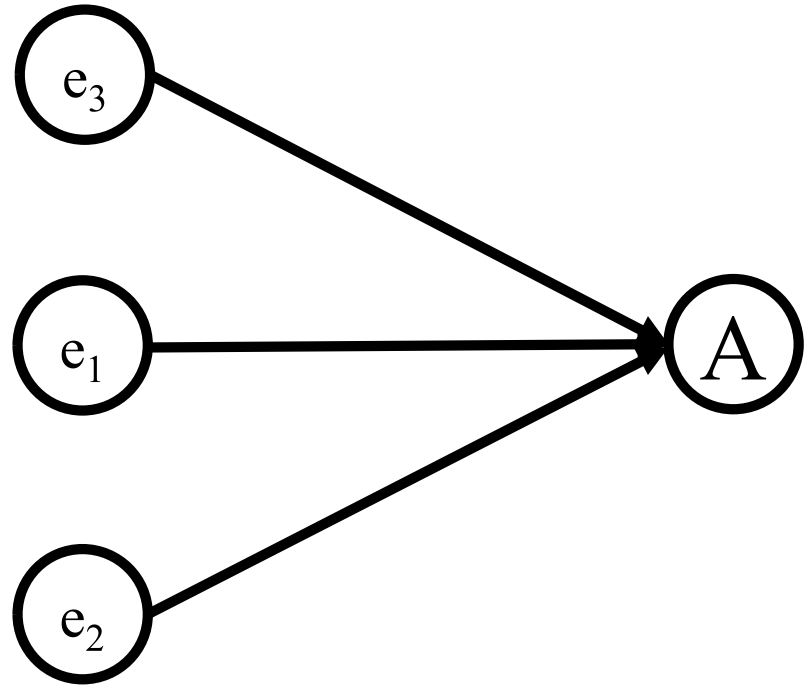
$$P(A \mid B) = \frac{P(B \mid A) * P(A)}{P(B)}$$

Prior vs. Posterior Probabilities

Prior Probability

$$P(A)$$

Posterior Probability



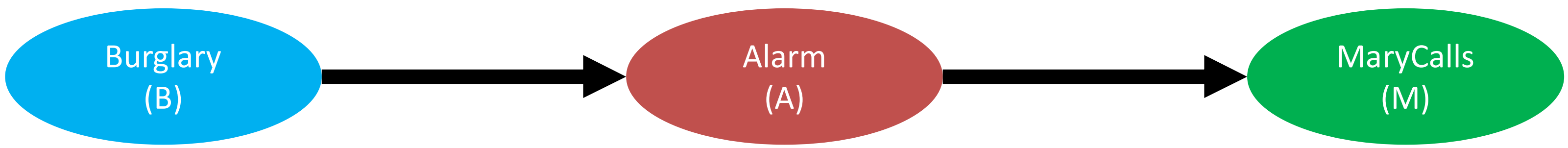
$$P(A \mid \text{parents}(A))$$

Use Chain Rule To Decompose

N Random Variables						Joint Probability
P_1	P_2	P_3	...	P_{N-1}	P_N	
true	true	true	...	true	true	false
true	true	true	...	true	false	true
true	true	false	...	false	true	false
...	...	...	...	...	...	...
false	false	true	...	true	false	true
false	false	true	...	false	true	true
false	false	false	...	false	false	false

Conditional Independence

Causal Chain:



by Conditional Probability formula

Joint Probabilities

by Chain Rule:

$P(f_1 \wedge f_2 \wedge \ldots \wedge f_n) = \prod_{i=1}^{n} P(f_i \mid parents(f_i))$

$$\boxed{P(M \mid A, B) = \frac{P(B, A, M)}{P(B, A)}} = \frac{P(B) * P(A \mid B) * P(M \mid A)}{P(B) * P(A \mid B)} = P(M \mid A)$$

Burglary and MaryCalls are CONDITIONALLY independent <u>given</u> Alarm.

If Alarm is <u>given</u>, what “happened before” Alarm does not directly influence MaryCalls.

Chain Rule

Conditional probabilities can be used to decompose conjunctions using the chain rule. For any propositions (random variables) $f_1, f_2, \ldots, f_n$ **:**

$$P(f_1 \wedge f_2 \wedge \ldots \wedge f_n) = \prod_{i=1}^{n} P(f_i \mid f_1 \wedge f_2 \wedge \ldots \wedge f_{i-1})$$

However, it can be rewritten as:

$$P(f_1 \wedge f_2 \wedge \ldots \wedge f_n) = \prod_{i=1}^{n} P(f_i \mid parents(f_i))$$

because with conditional independence(s) considered:

$$\prod_{i=1}^{n} P(f_i \mid f_1 \wedge f_2 \wedge \ldots \wedge f_{i-1}) = \prod_{i=1}^{n} P(f_i \mid parents(f_i))$$

Chain Rule

Conditional probabilities can be used to decompose conjunctions using the chain rule. For any propositions (random variables) $f_1, f_2, \ldots, f_n$ **:**

$P(f_1 \wedge f_2 \wedge \ldots \wedge f_n) =$

$P(f_1)\ *$

$P(f_2 \mid f_1)\ *$

$P(f_3 \mid f_1 \wedge f_2)\ *$

$\ldots$

$P(f_n \mid f_1 \wedge f_2 \wedge \ldots \wedge f_{n-1}) =$

$= \prod_{i=1}^{n} P(f_i \mid Parents(f_i))$ ← Enabled by conditional independence

Parents of Random Variable f_i

Parents of random variable f_i ($parents(f_i)$) is a minimal set of predecessors of f_i in the total ordering such that the other predecessors of f_i are conditionally independent of f_i given $parents(f_i)$.

A set of all predecessors of f_i : $A = \{f_1, f_2, \ldots, f_{i-1}\}$

A set of all parents of f_i : B

A set of all non-parents (predecessors NOT in B) of f_i : C

$$A = \{f_1, f_2, \ldots, f_{i-1}\} = B \cup C \; where \; B \cap C = \varnothing$$

when $parents(f_i)$ are given (all their values are known).

Parents of Random Variable f_i

Parents of random variable f_i ($parents(f_i)$) is a minimal set of predecessors of f_i in the total ordering such that the other predecessors of f_i are conditionally independent of f_i given $parents(f_i)$.

So: when $parents(f_i)$ are given, f_i probabilistically depends on each of its parents ($parents(f_i)$), but is independent of its other predecessors. That is

$$parents(f_i) \subseteq \{f_1, f_2, \ldots, f_{i-1}\}$$

such that:

$$P(f_i \mid f_1 \wedge f_2 \wedge \ldots \wedge f_{i-1}) = P(f_i \mid parents(f_i))$$

Bayes Network: Factorization

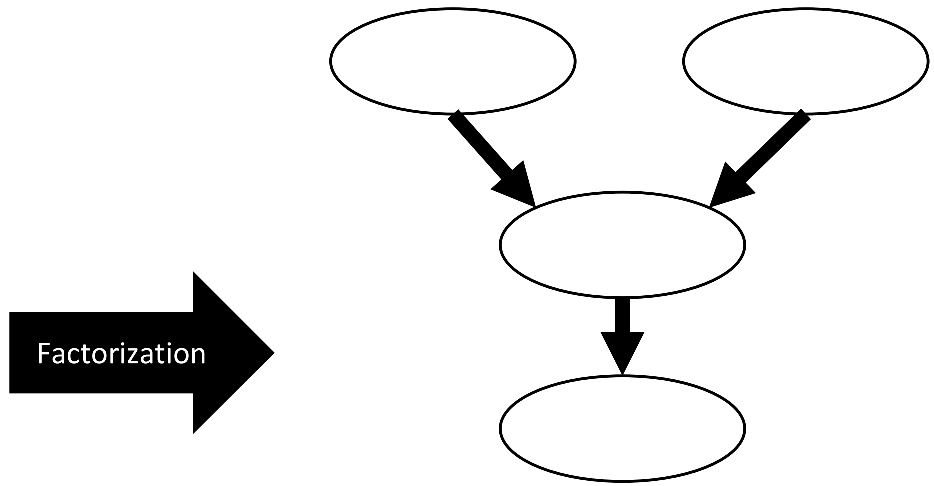
Chain rule AND definition of $parents(f_i)$ gives us:

$$P(f_1 \wedge f_2 \wedge \ldots \wedge f_n) = \prod_{i=1}^{n} P(f_i \mid parents(f_i))$$

Joint probability distribution **=** **Product of conditional probabilities after factorization of joint probability distribution**

N Random Variables						Joint Probability
P_1	P_2	P_3	...	P_{N-1}	P_N	
true	true	true	...	true	true	0.0011
true	true	true	...	true	false	0.0451
true	true	false	...	false	true	0.1011
...	...	...	...	...	...	...
false	false	true	...	true	false	0.0909
false	false	true	...	false	true	0.0651
false	false	false	...	false	false	0.2021

Joint probability distribution



Bayes Network: graph representation of joint probability distribution factorization

Bayesian (Belief) Network

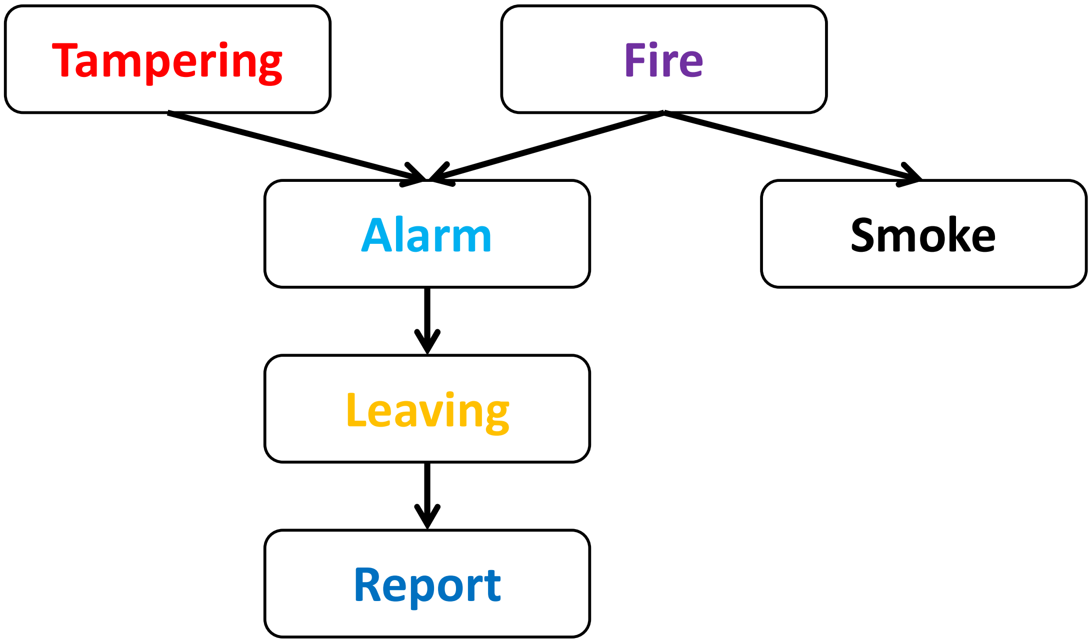
A Bayesian belief network describes the joint probability distribution for a set of variables.

A Bayesian network is an acyclic, directed graph (DAG), where the nodes are random variables (propositions). There is an edge (arc) from each elements of $\mathrm{parents}(X_i)$ into X_i. Associated with the Bayesian network is a set of conditional probability distributions - the conditional probability of each variable given its parents (which includes the prior probabilities of those variables with no parents).

Consists of:

- **a graph (DAG) with nodes corresponding to random variables**
- **a domain for each random variable**
- **a set of conditional probability distributions $P(X_i \mid \mathrm{parents}(X_i))$**

Bayesian (Belief) Network: Example



Random Variables (Propositions):

- **Tampering: true if the alarm is tampered with**
- **Fire: true if there is a fire**
- **Alarm: true if the alarm sounds**
- **Smoke: true if there is smoke**
- **Leaving: true if people leaving the building at once**
- **Report: true if someone who left the building reports fire**

Domain for all variables: $\{\text{true, false}\}$

NOTE: RVs don't have to be Boolean

Building Bayesian (Belief) Network

1. Order Random Variables (ordering matters!)
2. Create network nodes for each Random Variable
3. Add edges between parent nodes and children nodes
   - For every node node X_i:
     - choose a minimal set S of parents for X_i
     - for each parent node Y in S add an edge from Y to X_i

4. Add Conditional Probability Tables

Make it compact / sparse: choose your Random Variable ordering wisely.

Inference

Query (let's change it a bit for simplicity):

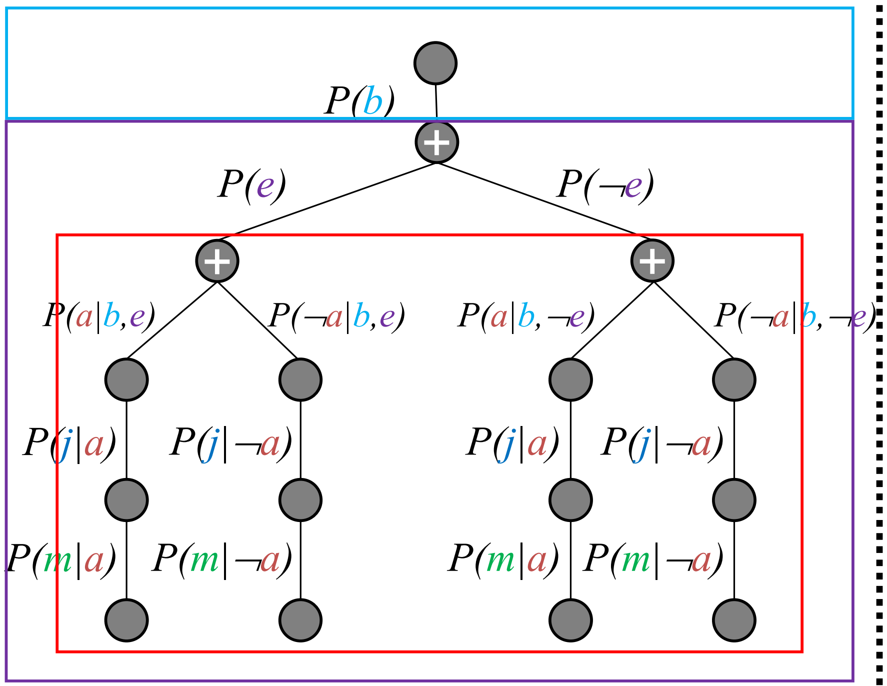
$P(Burglary = true \mid JohnCalls = true \wedge MaryCalls = true)$

Query rewritten:

$P(b \mid j, m)$

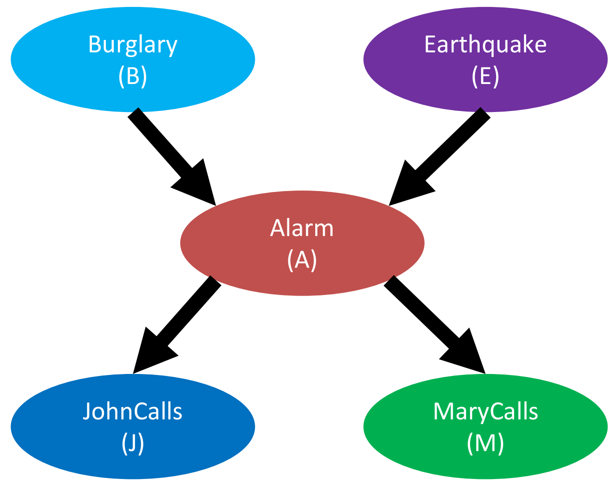
$= \alpha * \sum_e \sum_a P(b) * P(e) * P(a|b,e) * P(j|a) * P(m|a)$

$= \alpha * P(b) * \sum_e P(e) * \sum_a P(a|b,e) * P(j|a) * P(m|a)$



P(B)	P(¬B)
0.001	0.999

P(E)	P(¬E)
0.002	0.998



B	E	P(A|B,E)
t	t	0.95
t	f	0.94
f	t	0.29
f	f	0.001

A	P(J|A)
t	0.90
f	0.05

A	P(M|A)
t	0.70
f	0.01

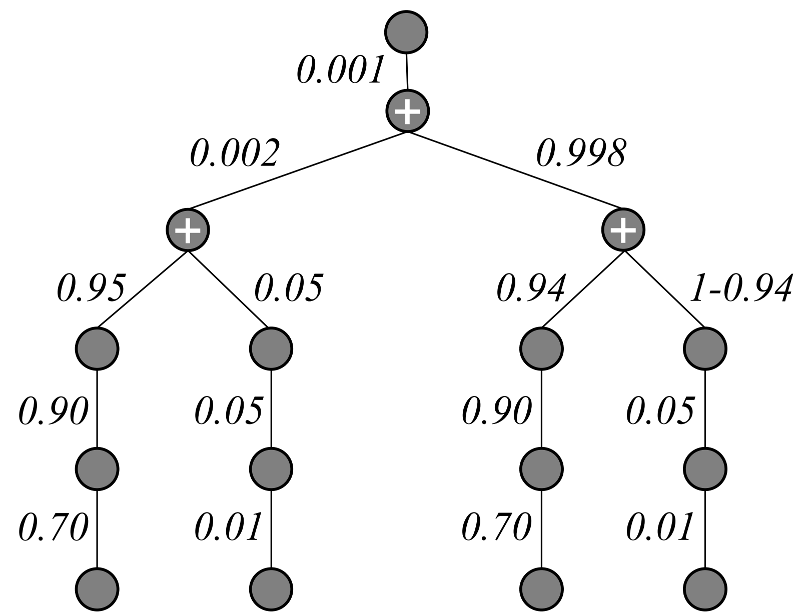
Inference

Query (let's change it a bit for simplicity):

$P(Burglary = true \mid JohnCalls = true \wedge MaryCalls = true)$

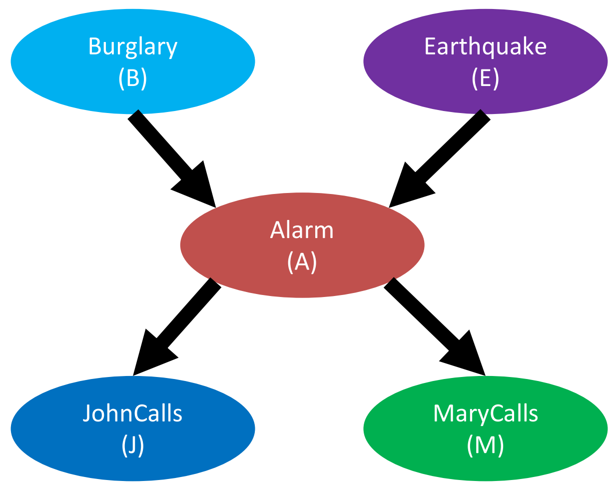
Query rewritten:

$$P(b \mid j, m)$$
$$= \alpha * \sum_{e}\sum_{a} P(b) * P(e) * P(a|b,e) * P(j|a) * P(m|a)$$
$$= \alpha * P(b) * \sum_{e} P(e) * \sum_{a} P(a|b,e) * P(j|a) * P(m|a)$$



P(B)	P(¬B)
0.001	0.999

P(E)	P(¬E)
0.002	0.998



B	E	P(A|B,E)
t	t	0.95
t	f	0.94
f	t	0.29
f	f	0.001

A	P(J|A)
t	0.90
f	0.05

A	P(M|A)
t	0.70
f	0.01

Inference

Query (let's change it a bit for simplicity):

$P(Burglary = true \mid JohnCalls = true \wedge MaryCalls = true)$

We can now calculate:

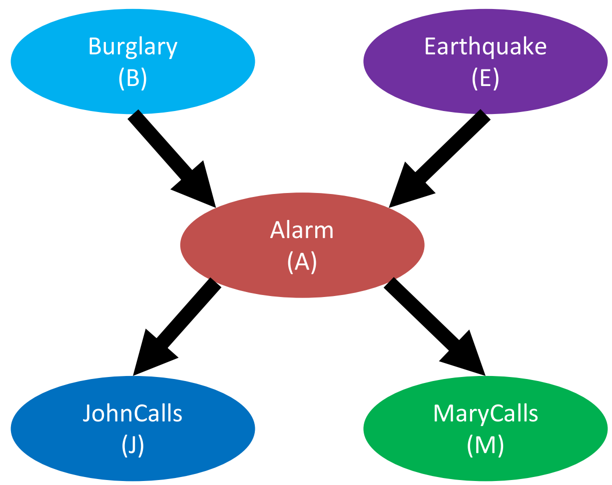
$$P(b \mid j, m) = \alpha * 0.00059224$$

And through similar process (not shown):

$$P(\neg b \mid j, m) = \alpha * 0.0014919$$

P(B)	P(¬B)
0.001	0.999

P(E)	P(¬E)
0.002	0.998



B	E	P(A|B,E)
t	t	0.95
t	f	0.94
f	t	0.29
f	f	0.001

A	P(J|A)
t	0.90
f	0.05

A	P(M|A)
t	0.70
f	0.01

Inference

Query (now we can get joint distribution):

$$P(Burglary \mid JohnCalls = true \wedge MaryCalls = true)$$

We can now calculate:

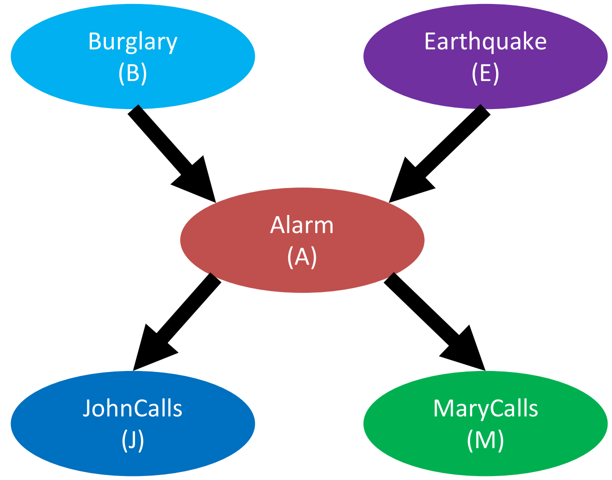
$$P(B \mid j, m) = \alpha * < 0.00059224, 0.0014919 >$$

And after normalization:

$$P(B \mid j, m) \approx < 0.284, 0.716 >$$

P(B)	P(¬B)
0.001	0.999

P(E)	P(¬E)
0.002	0.998

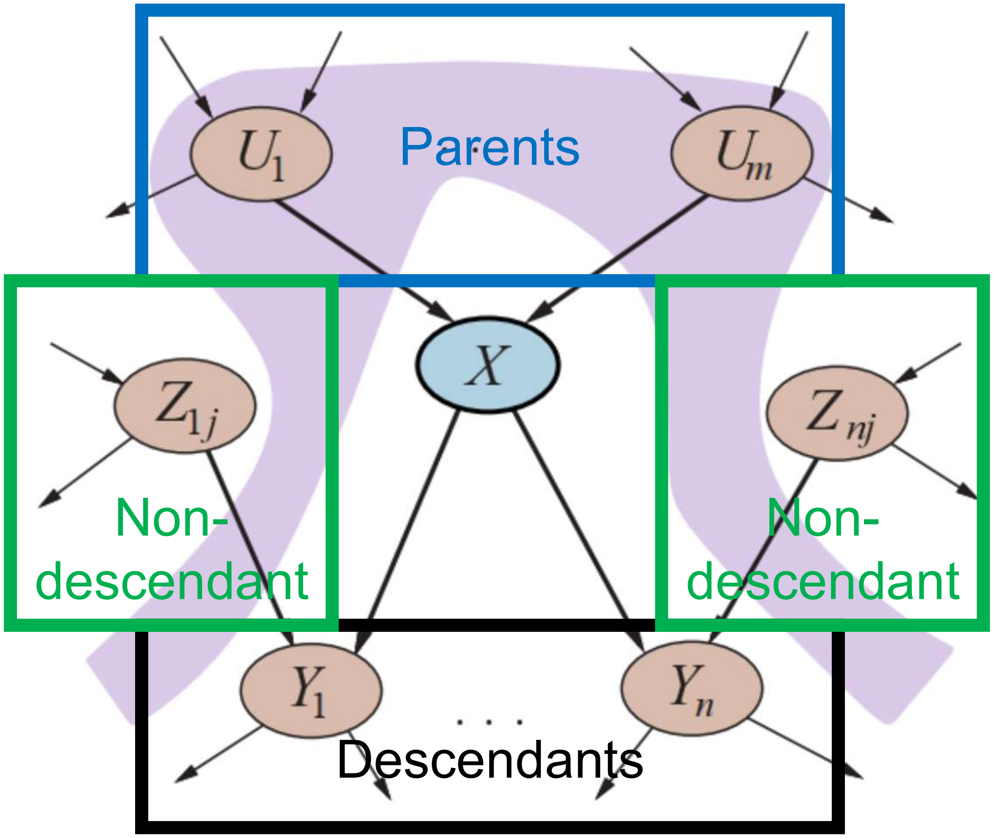


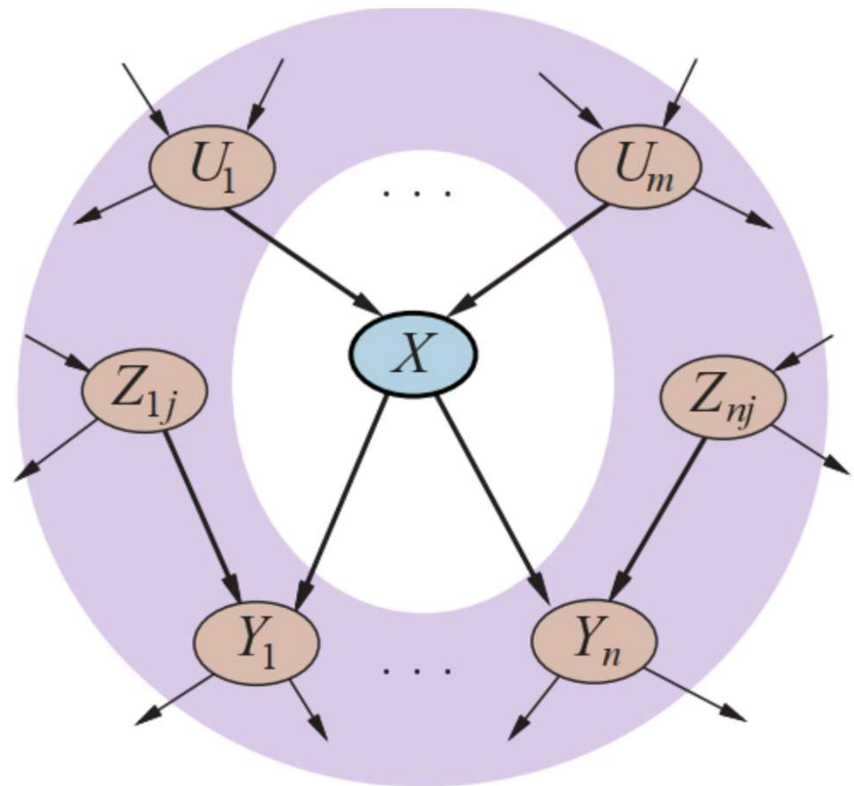
B	E	P(A|B,E)
t	t	0.95
t	f	0.94
f	t	0.29
f	f	0.001

A	P(J|A)
t	0.90
f	0.05

A	P(M|A)
t	0.70
f	0.01

More On Conditional Independence





Node X is conditionally independent of its non-descendants given its parents.

Node X is conditionally independent of ALL other nodes in the network its given its Markov blanket.

Why do we care?

An unconstrained joint probability distribution with N **binary** variables involves 2^{N} probabilities.
Bayesian network with at most k parents per each node (N) involves $\mathrm{N} * 2^{\mathrm{k}}$ probabilities ($\mathrm{k} < \mathrm{N}$).

Decision Networks

Decision Theory

- Decisions: every plan (actions) leads to an outcome (state)
- Agents have preferences (preferred outcomes)
- Preferences $\rightarrow$ outcome utilities
- Agents have degrees of belief (probabilities) for actions

Decision theory = probability theory + utility theory

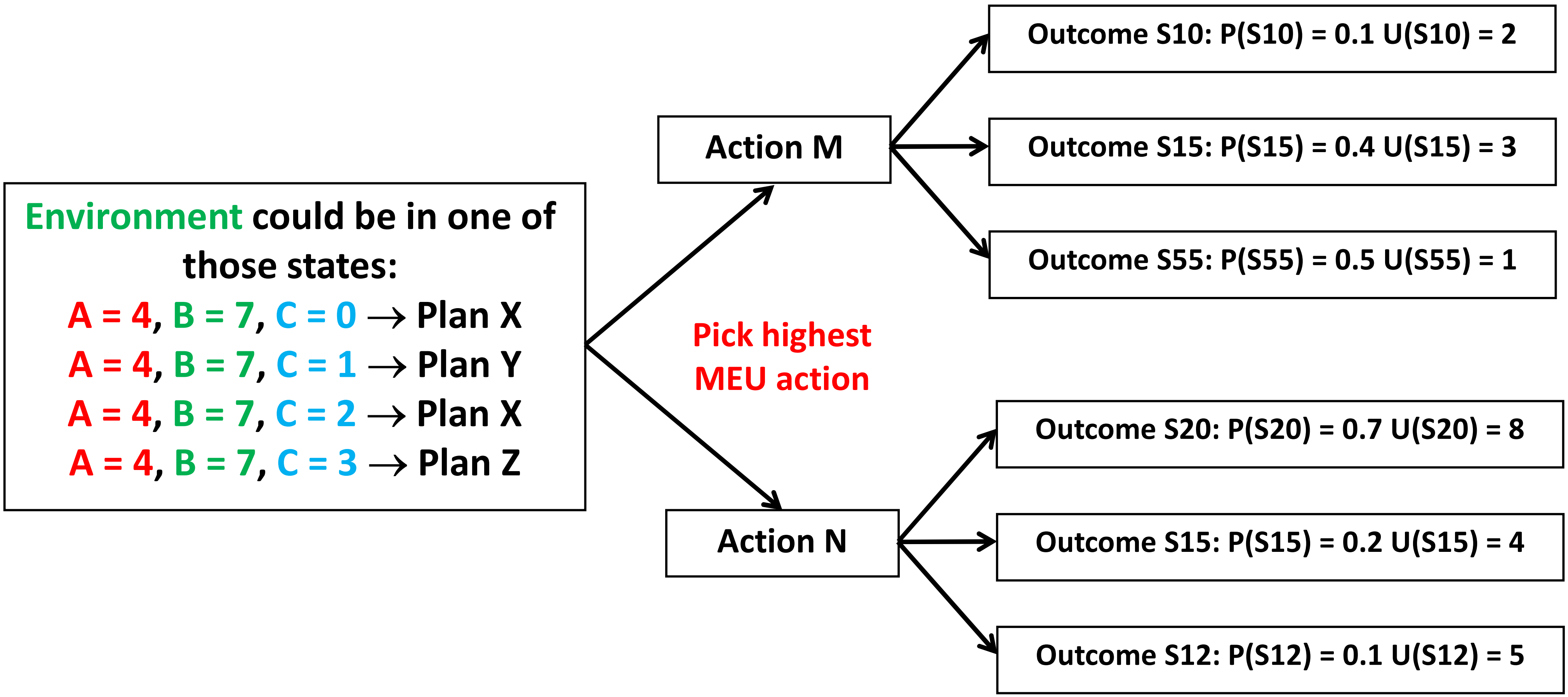
Decision Theory

- **Decisions**: every plan (**actions**) leads to an outcome (state)
- **Agents have preferences (preferred outcomes)**
- **Preferences $\rightarrow$ outcome utilities**
- **Agents have degrees of belief (probabilities) for actions**

Decision theory = $\underbrace{\text{probability theory}}_{\text{BELIEFS}}$ + $\underbrace{\text{utility theory}}_{\text{DESIRES}}$

Maximum Expected (Average) Utility

$$MEU(M) = P(S10) * U(S10) + P(S15) * U(S15) + P(S55) * U(S55)$$



$$MEU(N) = P(S20) * U(S20) + P(S15) * U(S15) + P(S12) * U(S12)$$

Agents Decisions

Recall that agent ACTIONS change the state:

- **if we are in state** s
- **action** a **is expected to**
- **lead to another state** s' **(outcome)**

Given uncertainty about the current state s **and action outcome** s' **we need to define the following:**

- **probability (belief) of being in state** s**:** $P(s)$
- **probability (belief) of action** a **leading to outcome** s'**:** $P(s' \mid s, a)$

Now:

$$P(s' \mid s, a) = P(\textsc{Result}(a) = s') = \sum_s P(s) * P(s' \mid s, a)$$

Expected Action Utility

The expected utility of an action a given the evidence is the average utility value of all possible outcomes s' of action a, weighted by their probability (belief) of occurence:

$$EU(a) = \sum_{s'} \sum_{s} P(s) * P(s' \mid s, a) * U(s') = \sum_{s'} P(Result(a) = s') * U(s')$$

Rational agent should choose an action that maximizes the expected utility:

$$\text{chosen action} = \underset{a}{\text{argmax}}\ EU(a)$$

State Utility Function

Agent's preferences (desires) are captured by the Utility function $U(s)$.

Utility function assigns a value to each state s to express how desirable this state is to the agent.

How Did We Get Here?

Let's start with relationships (and related notation) between agent's preferences:

- **agent prefers A over B:**

$$\mathrm{A} \succ \mathrm{B}$$

- **agent is indifferent between A and B:**

$$\mathrm{A} \sim \mathrm{B}$$

- **agent prefers A over B or is indifferent between A and B (weak preference):**

$$\mathrm{A} \succcurlyeq \mathrm{B}$$

The Concept of Lottery

Let's assume the following:

- **an action a is a lottery <u>ticket</u>**
- **the set of outcomes (resulting states) is a lottery**

A lottery L with possible outcomes $S_1, \ldots, S_n$ that occur with probabilities $p_1, \ldots, p_n$ is written as:

$$L = [p_1, S_1; p_2, S_2; \ldots ; p_n, S_n]$$

Lottery outcome S_i: atomic state or another lottery.

Lottery Constraints: Orderability

Given two lotteries A and B, a rational agent must either prefer one or else rate them as equally preferable:

Exactly one of $(\mathrm{A} \succ \mathrm{B})$, $(\mathrm{B} \succ \mathrm{A})$, or $(\mathrm{A} \sim \mathrm{B})$ holds

Lottery Constraints: Transitivity

Given three lotteries A, B, and C, if an agent prefers A to B AND prefers B to C, then the agent must prefer A to C:

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

Lottery Constraints: Continuity

If some lottery B is between A and C in preference, then there is some probability p for which the rational agent will be indifferent between getting B for sure or some other lottery that yields A with probability p and C with probability $1 - p$:

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

Lottery Constraints: Substitutability

If an agent is indifferent between two lotteries A and B, then the agent is indifferent between two more complex lotteries that are the same, except that B is subsituted for A in one of them:

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

Lottery Constraints: Monotonicity

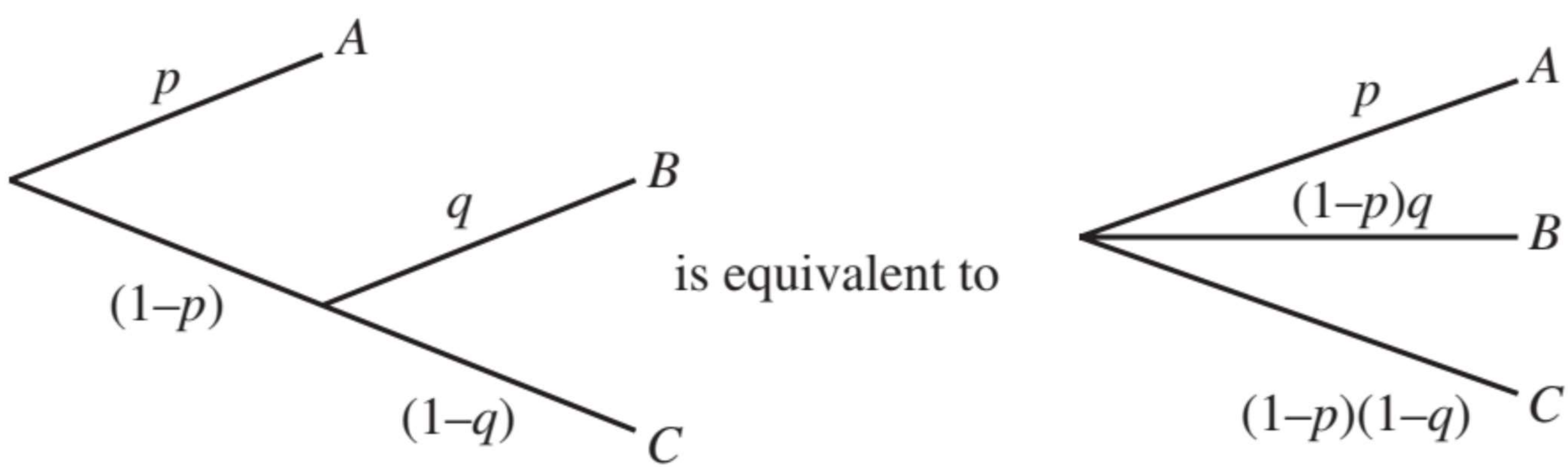
Suppose two lotteries have the same two possible outcomes, A and B. If an agent prefers A to B, then the agent must prefer the lottery that has a higher probability for A:

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

Lottery Constraints: Decomposability

Compound lotteries can be reduced to smaller ones using the laws of probability:

[p, A; 1- p, [q, B; 1- q, C]] ~ [p, A; (1- p)*q, B; (1- p)*(1- q), C]



Preferences and Utility Function

An agent whose preferences between lotteries follow the set of axioms (of utility theory) below:

- **Orderability**
- **Transitivity**
- **Continuity**
- **Subsitutability**
- **Monotonicity**
- **Decomposability**

can be described as possesing a utility function and maximize it.

Preferences and Utility Function

If an agent's preferences obey the axioms of utility theory, then there exist a function U such that:

$$U(A) = U(B) \textbf{ if and only if } (A \sim B)$$

and

$$U(A) > U(B) \textbf{ if and only if } (A \succ B)$$

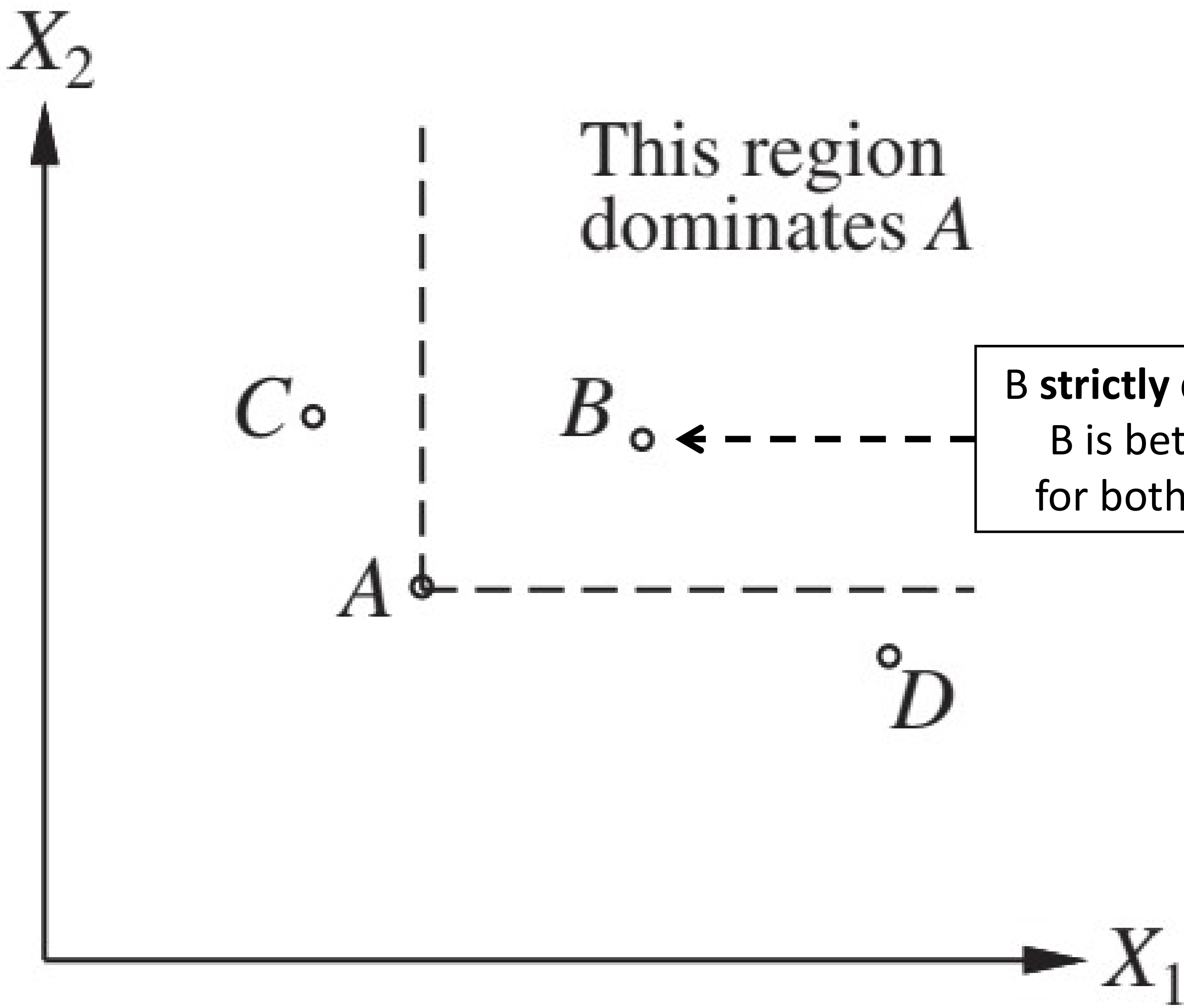
Multiattribute Outcomes

Outcomes can be characterized by more than one attribute. Decisions in such cases are handled by Multiattribute Utility Theory.

Attributes: $\mathbf{X} = X_1, \ldots, X_n$

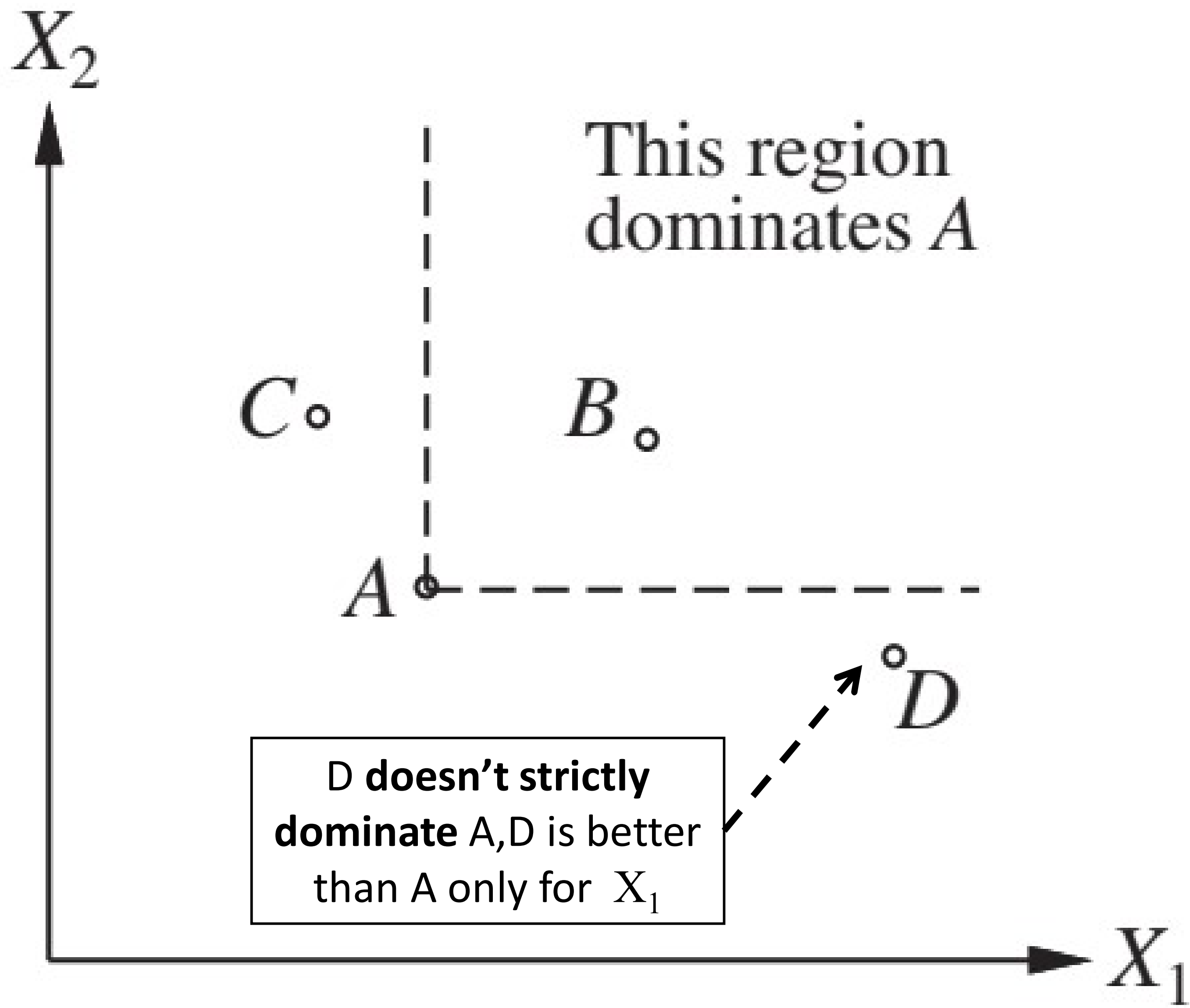
Assigned values: $\mathbf{x} = \langle x_1, \ldots, x_n \rangle$

Strict Dominance: Deterministic

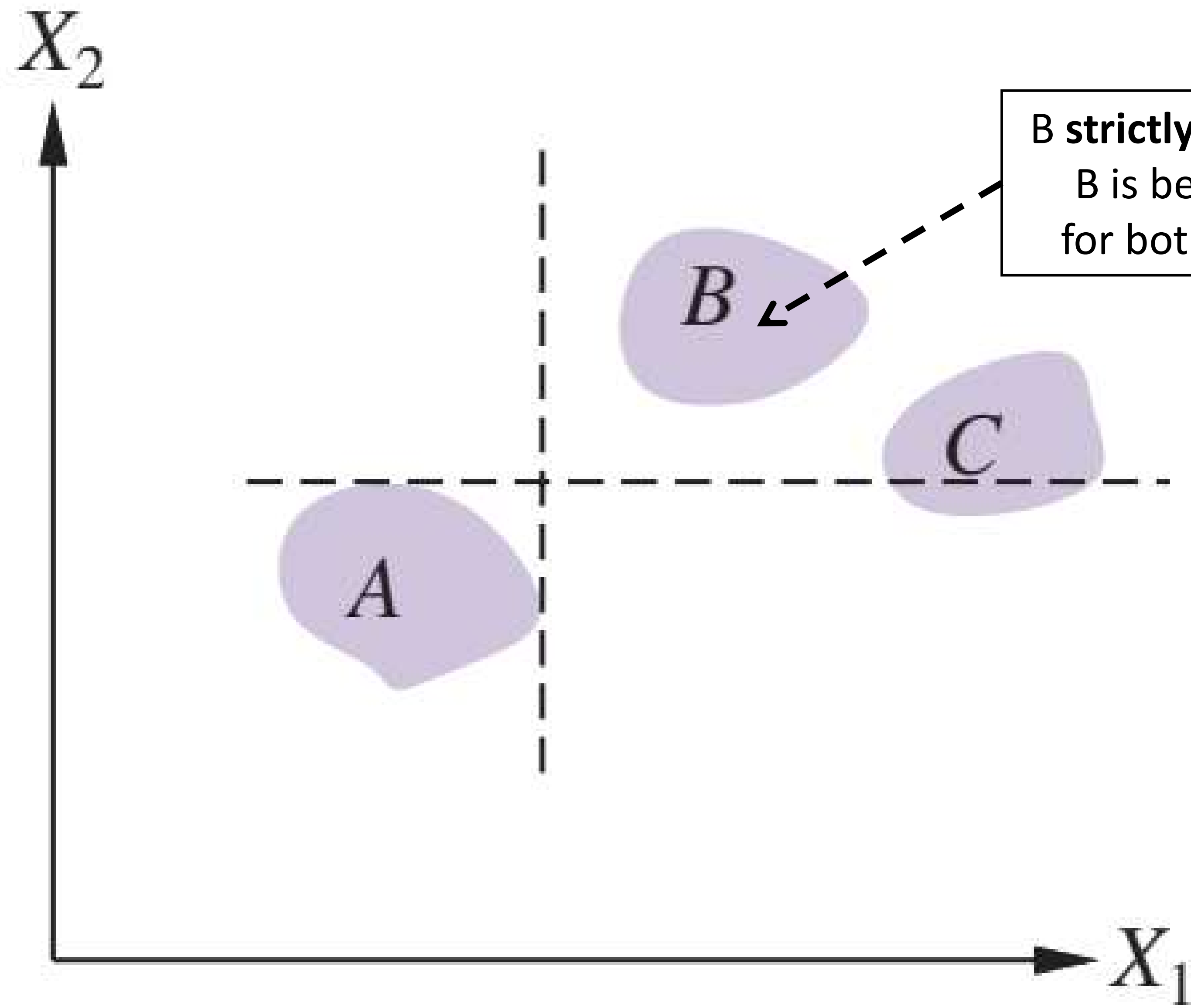


B **strictly dominates** A
B is better than A
for both X_1 and X_2

Strict Dominance: Deterministic



Strict Dominance: Uncertain



B **strictly dominates** A
B is better than A
for both X_1 and X_2

Decision Network (Influence Diagram)

Decision networks (also called influence diagrams) are structures / mechanisms for making rational decisions.

Decision networks are based on Bayesian networks, but include additional nodes that represent actions and utilities.

Decision Networks

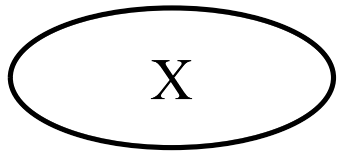
The most basic decision network needs to include:

- **information about current state** s
- **possible actions**
- **resulting state** s' **(after applying chosen action** a**)**
- **utility of the resulting state** $U(s')$

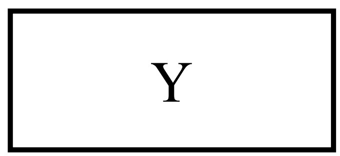
Decision Network Nodes

Decision networks are built using the following nodes:

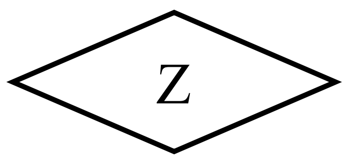
- **chance nodes:**



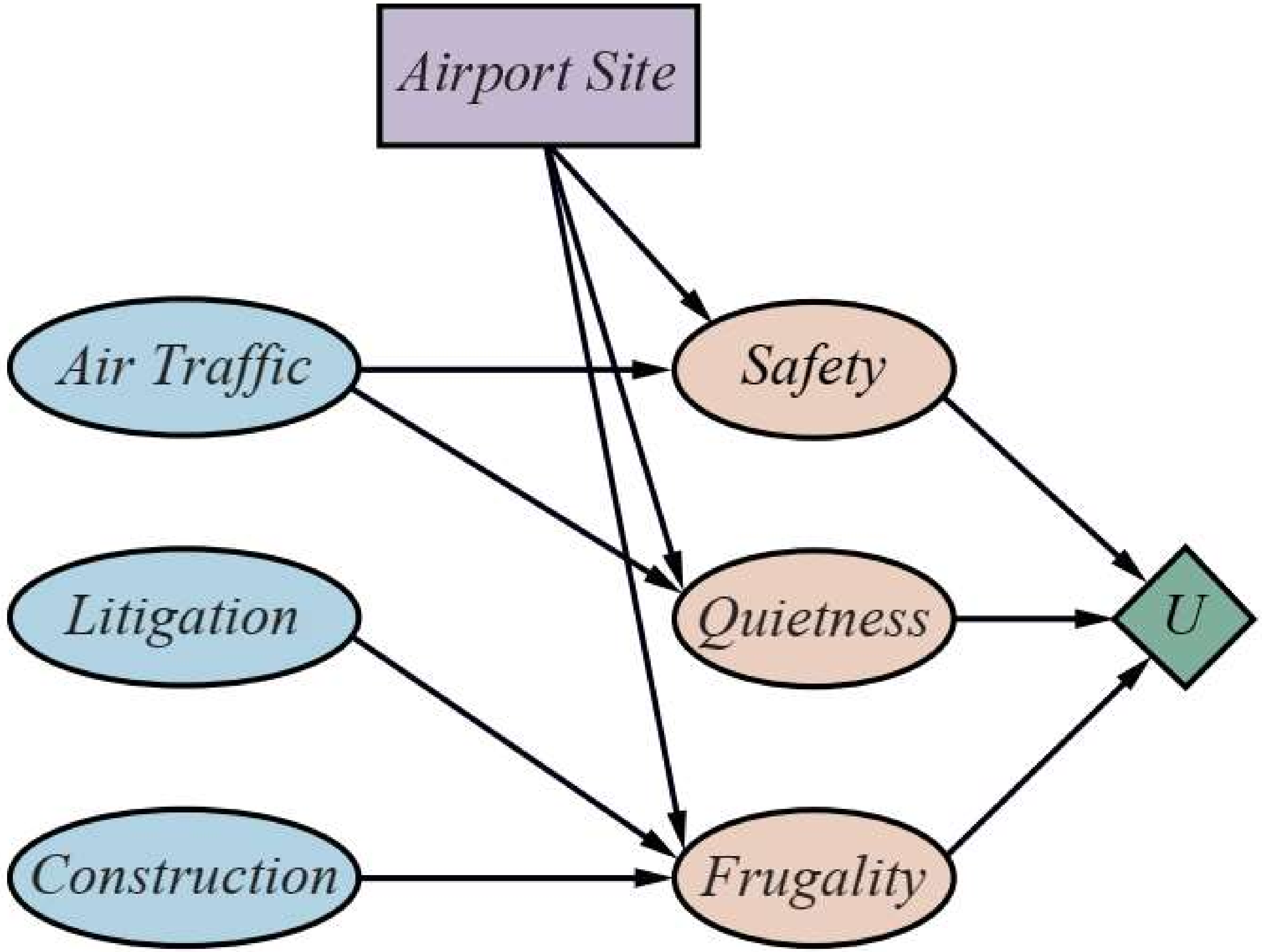
- **decision nodes:**



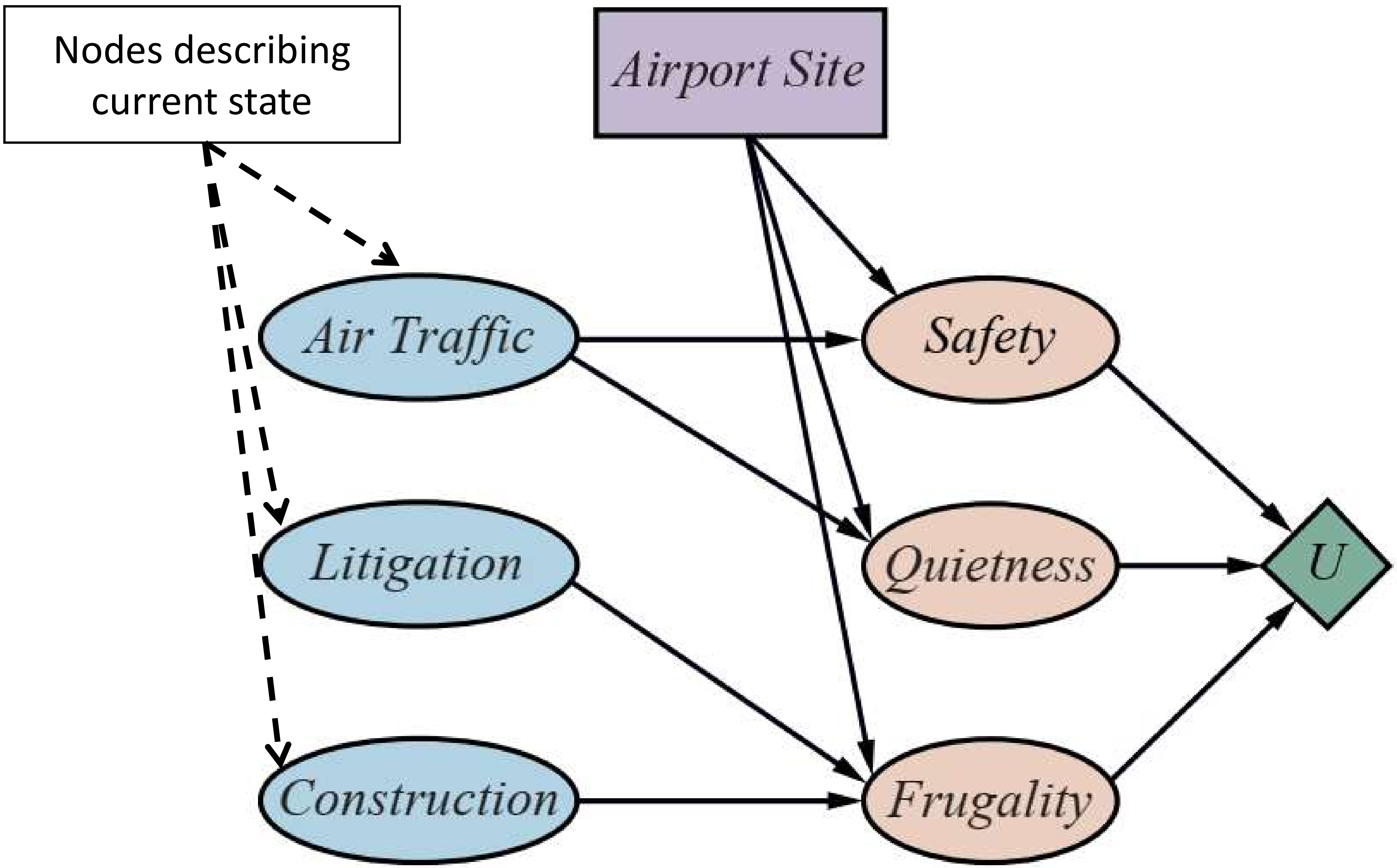
- **utility (or value) nodes**



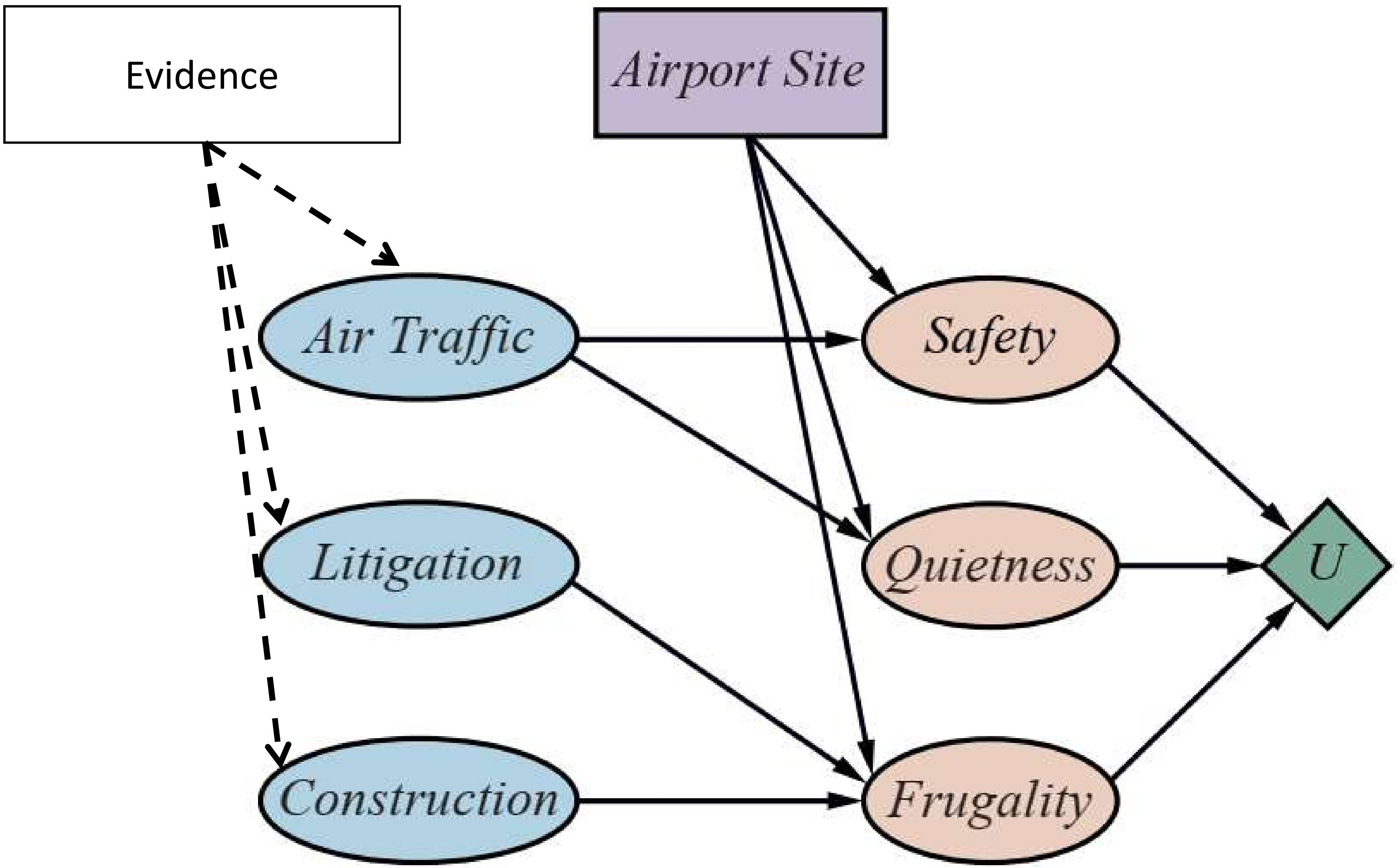
Decision Network: Example



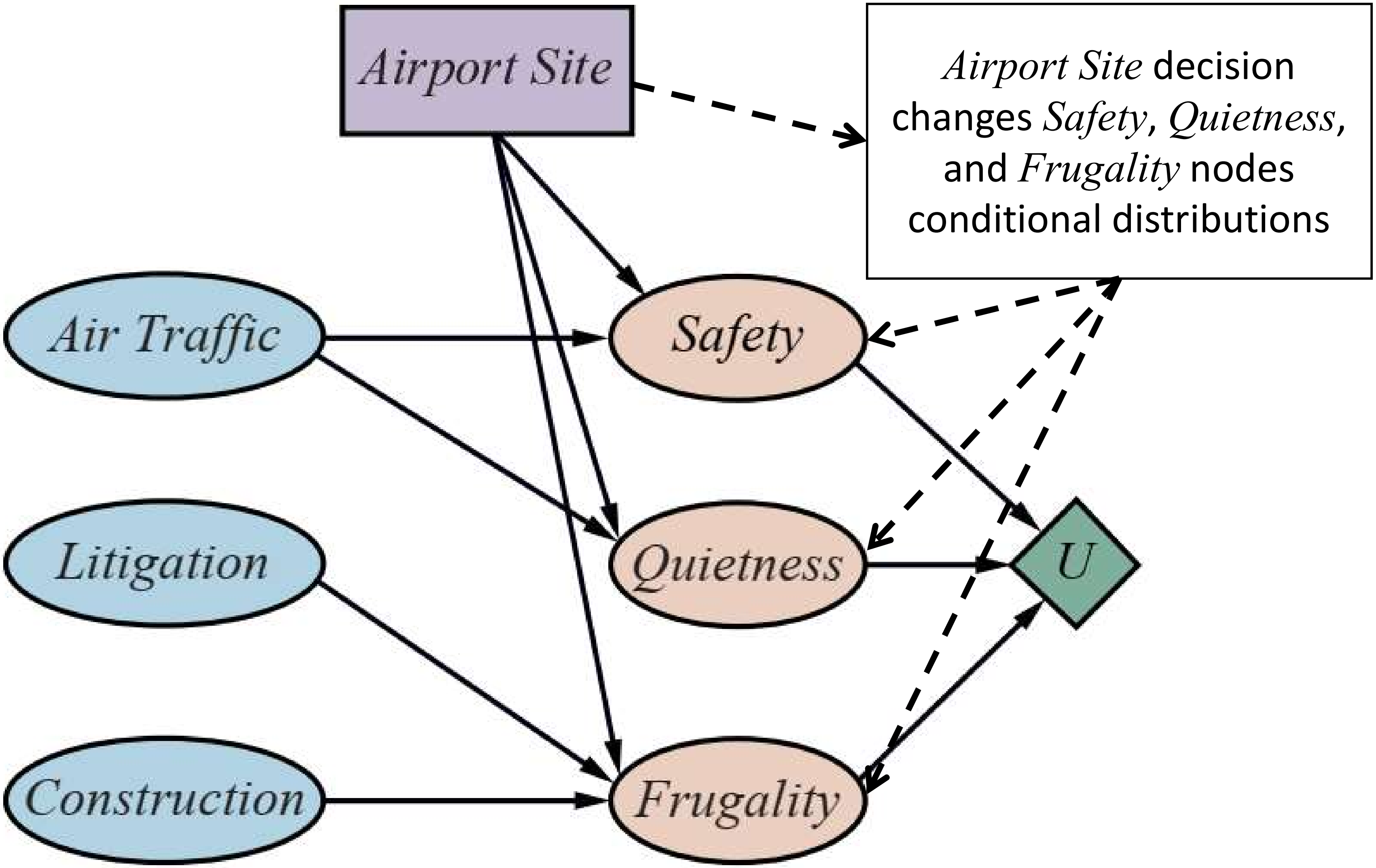
Decision Network: Example



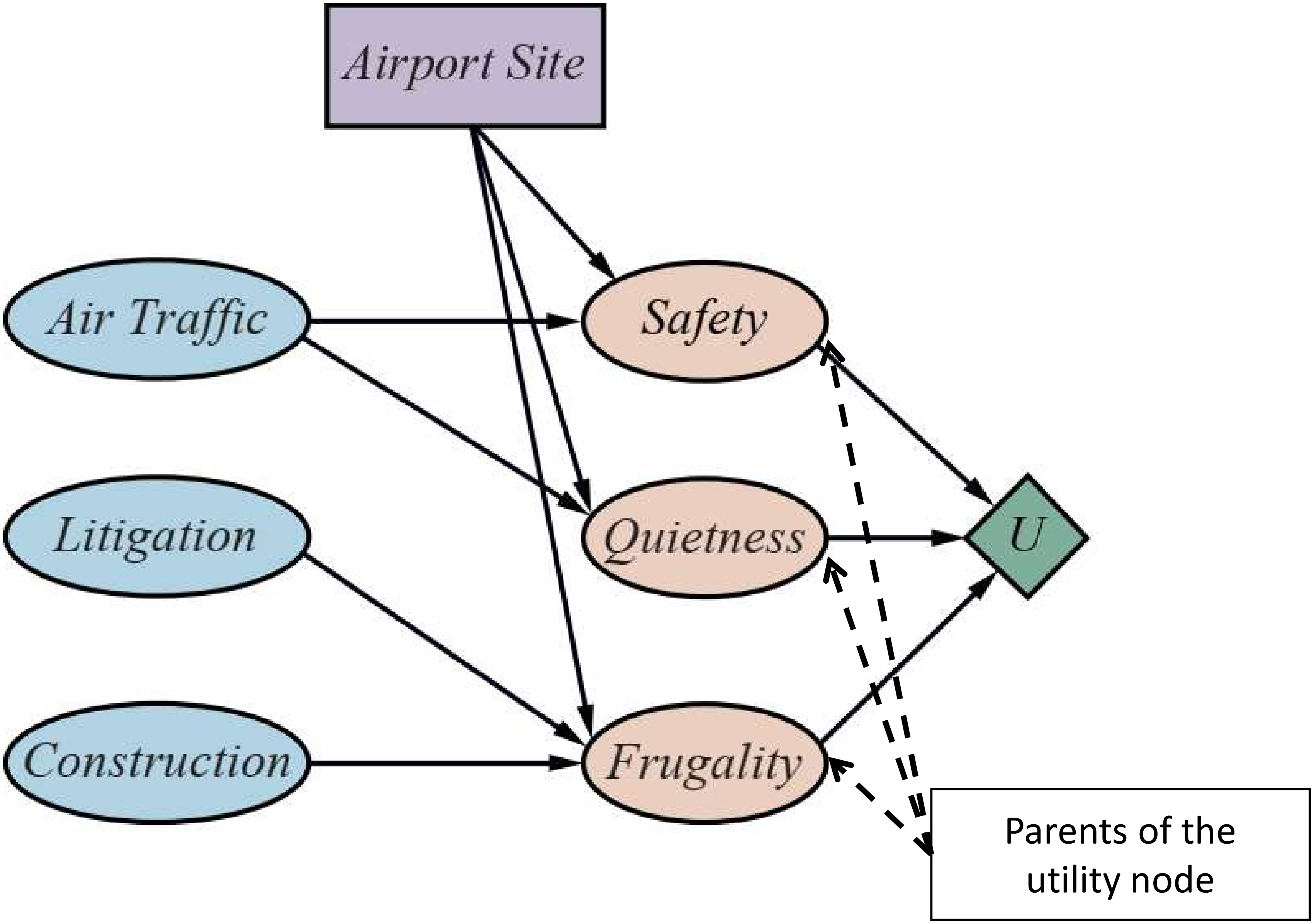
Decision Network: Example



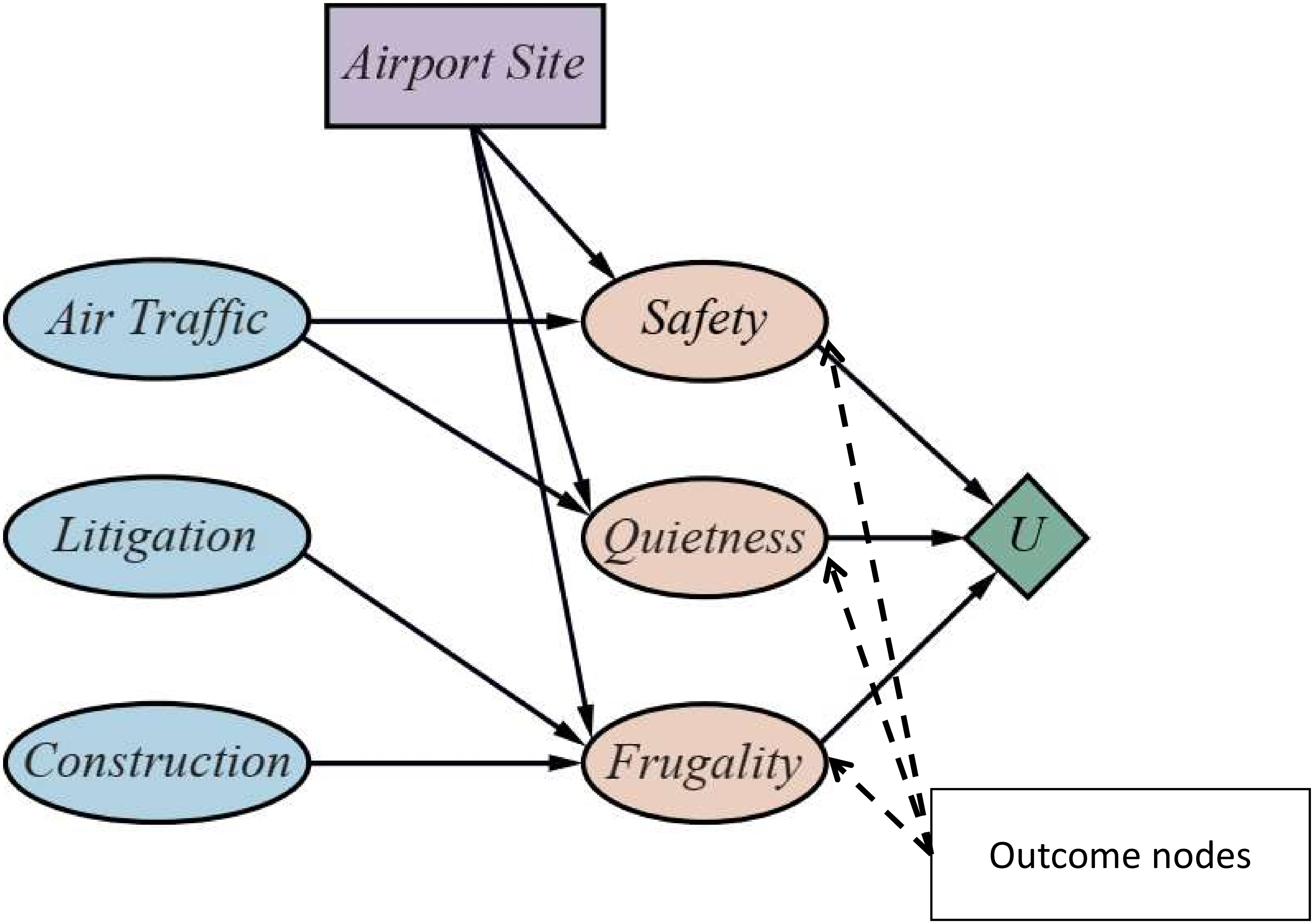
Decision Network: Example



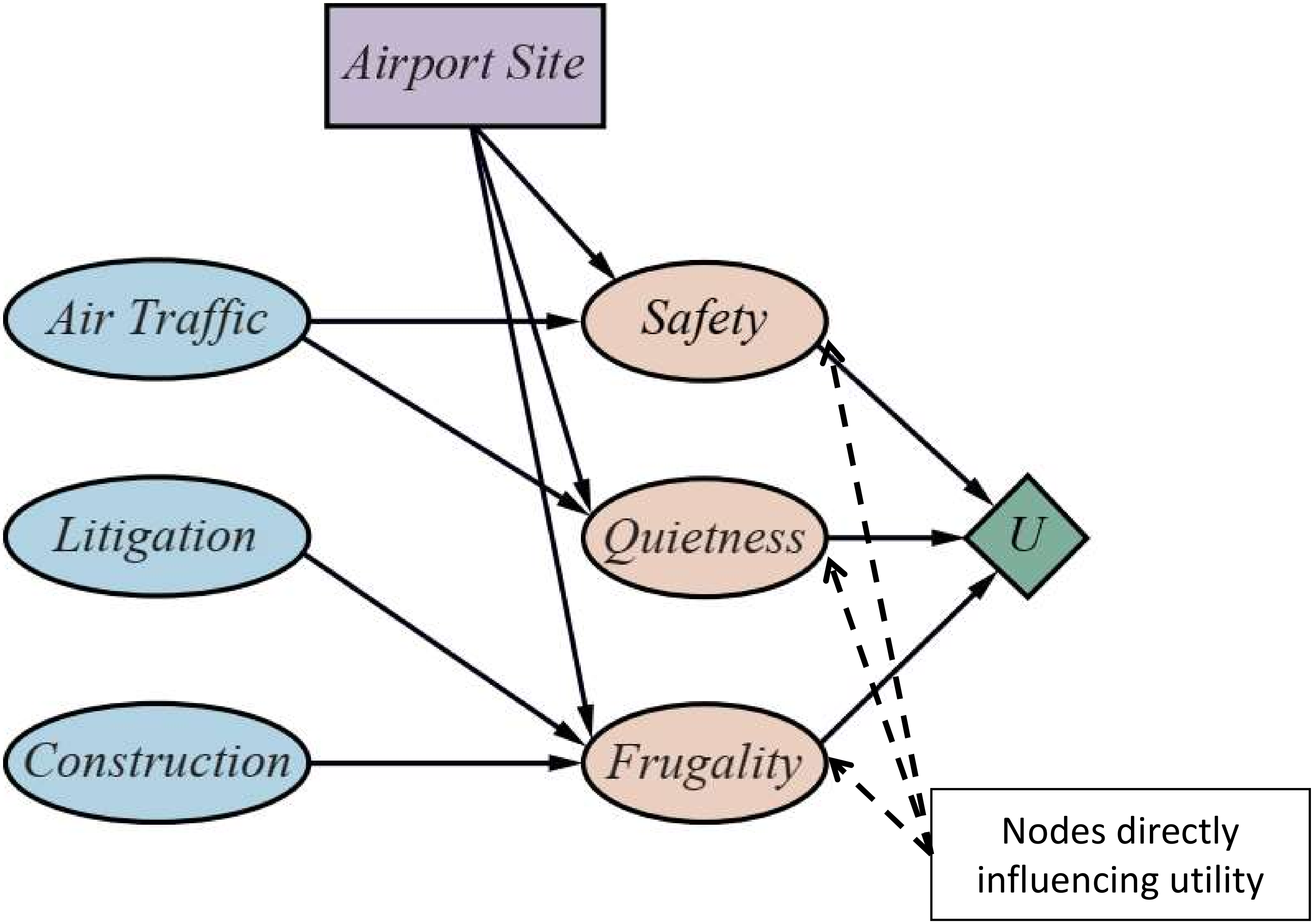
Decision Network: Example



Decision Network: Example



Decision Network: Example

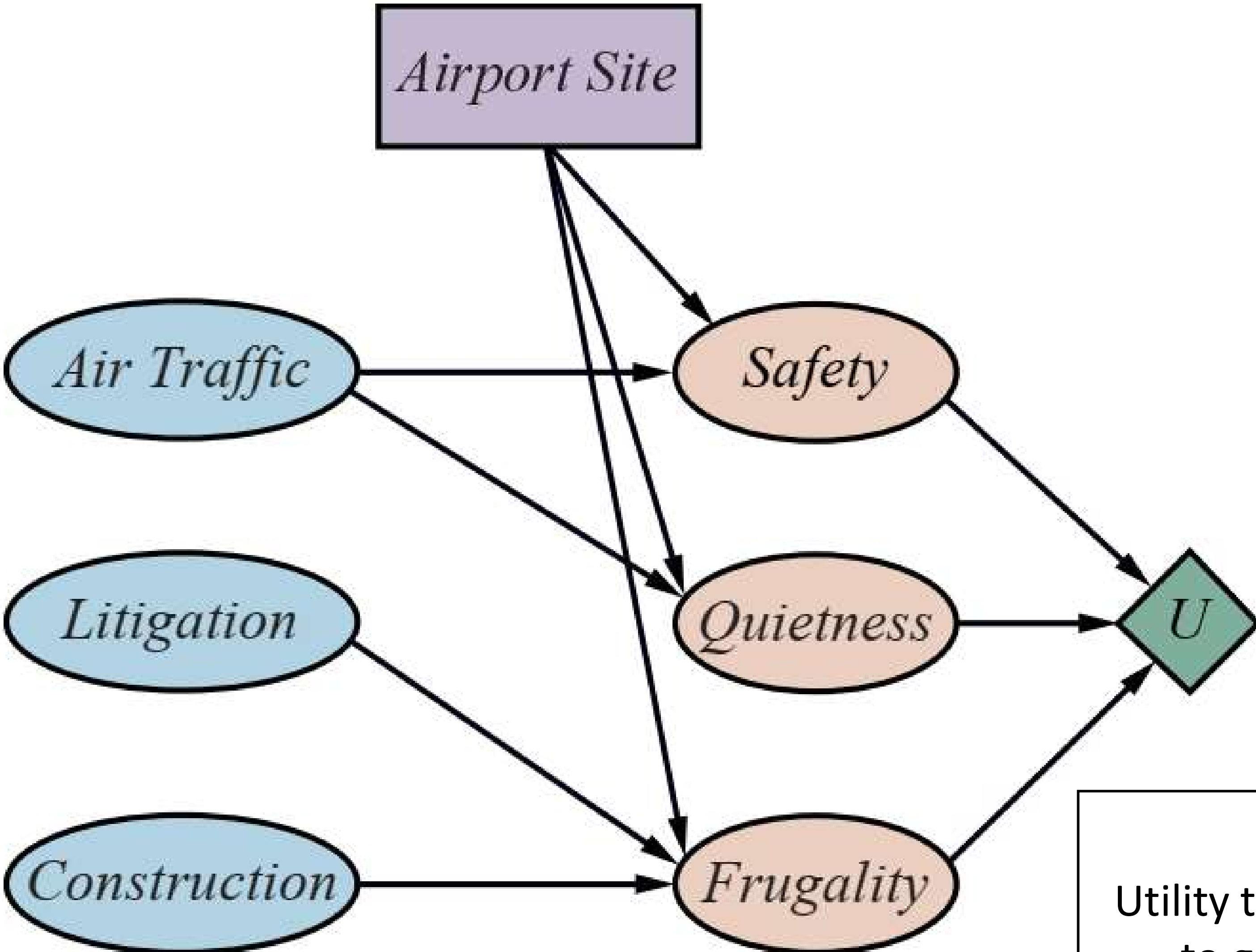


Decision Network: Evaluation

The algorithm for decision network evaluation is as follows:

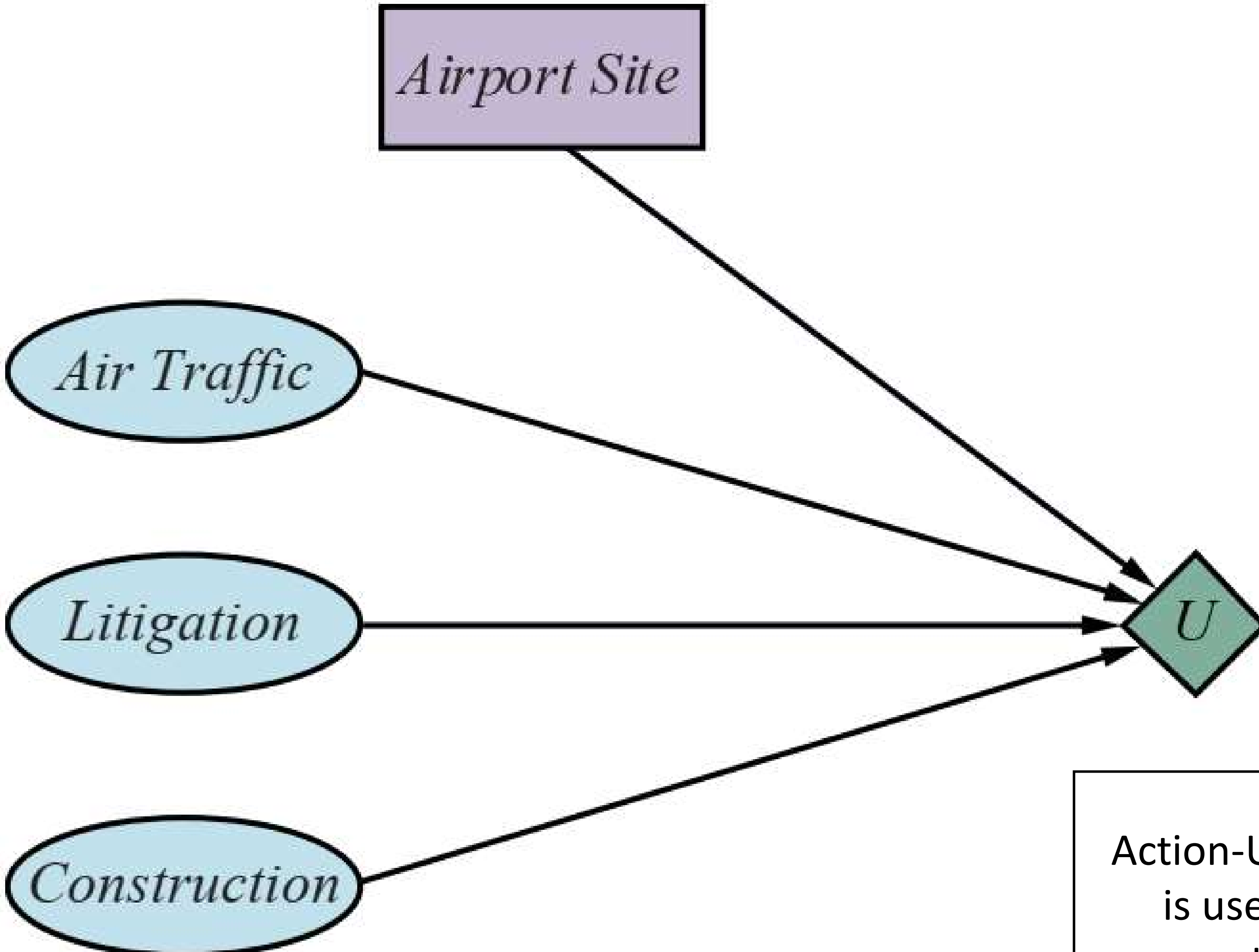
1. Set the evidence variables for the current state
2. For each possible value a of decision node:
   a. Set the decision node to that value
   b. Calculate the posterior probabilities for the parent nodes of the utility node
   c. Calculate the utility for the action / value a
3. Return the action with highest utility

Decision Network: Example



Utility table is used to get utility

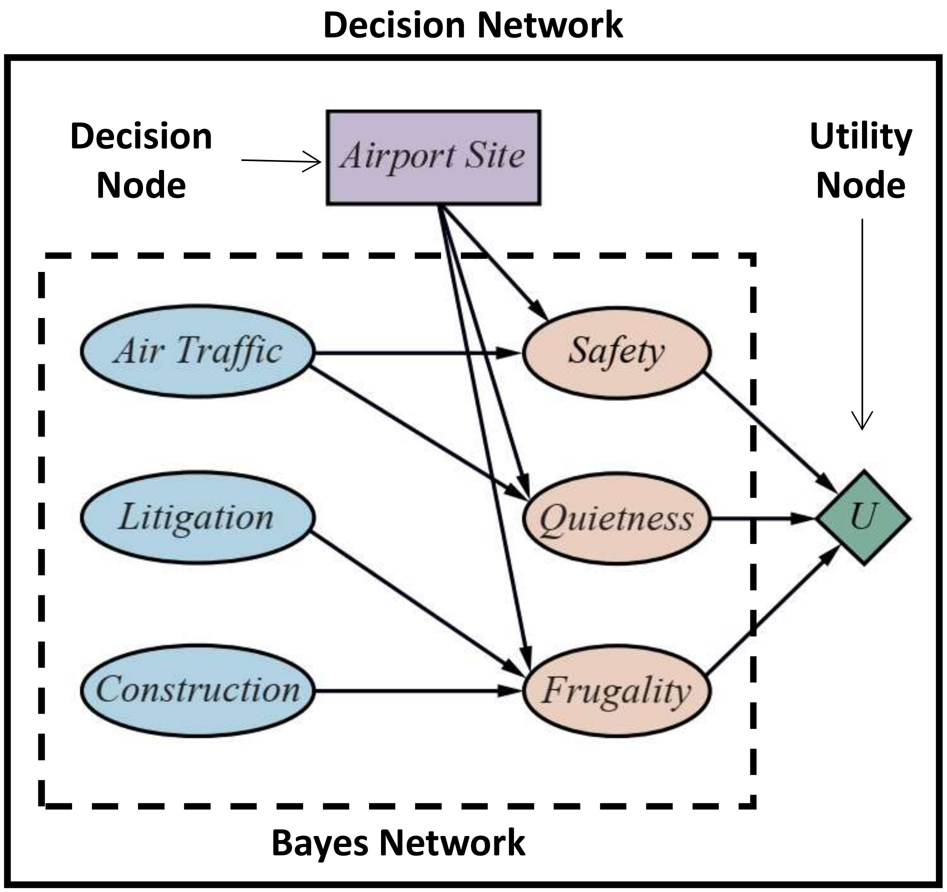
Decision Network: Simplified Form



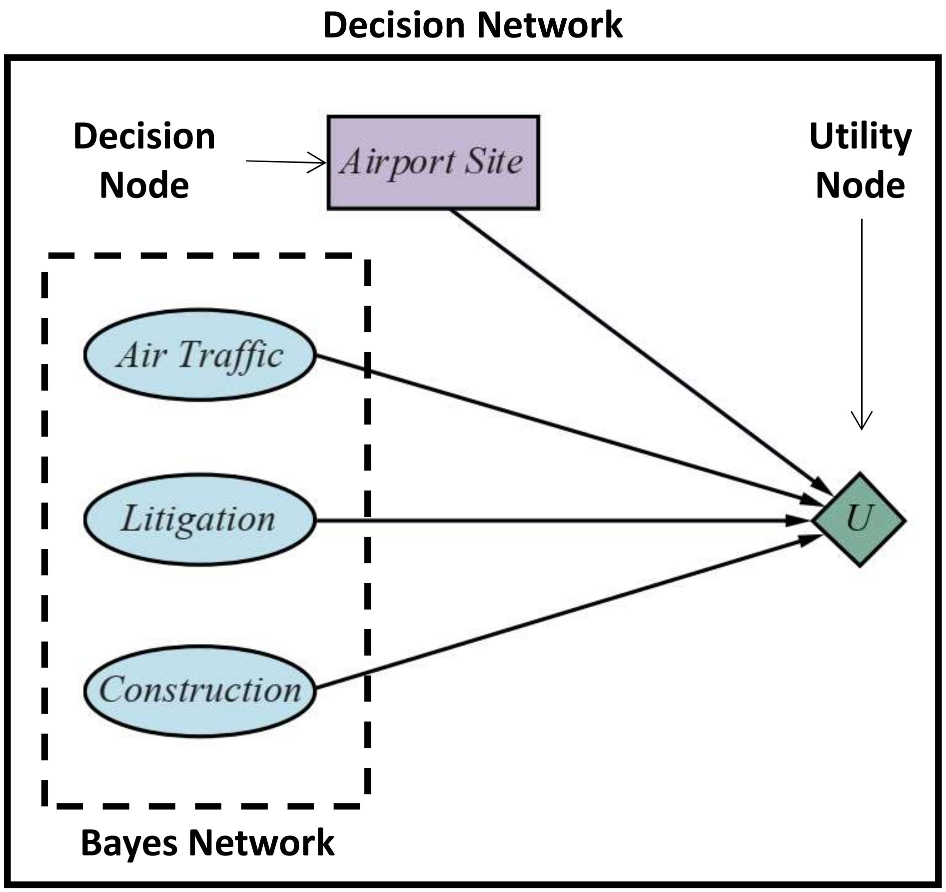
Action-Utility table is used to get expected utility

(Single-Stage) Decision Networks

General Structure

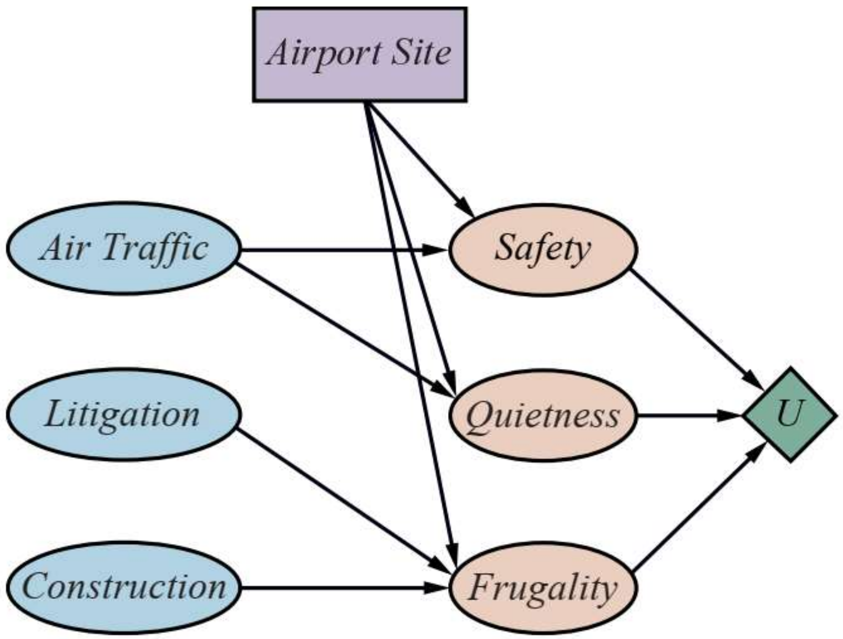


Simplified Structure



(Single-Stage) Decision Networks

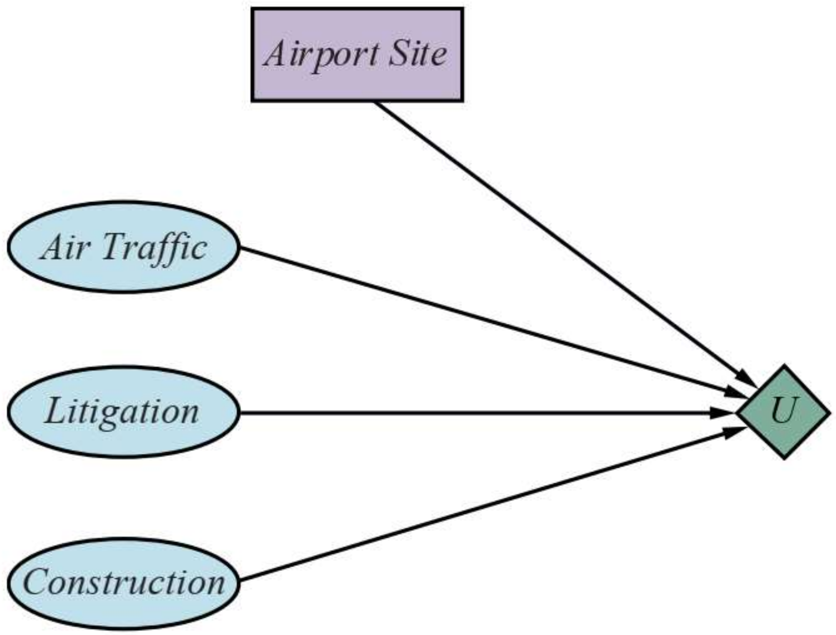
General Structure



Utility Table

S	low	low	low	low	high	high	high	high
Q	low	low	high	high	low	low	high	high
F	low	high	low	high	low	high	low	high
U	10	20	5	50	70	150	100	200

Simplified Structure



Action-Utility Table (not all columns shown)

AT	low	low	low	---	---	high	high	high
L	low	low	high	---	---	low	high	high
C	low	high	low	---	---	high	low	high
AS	A	A	A	---	---	B	B	B
U	10	20	5	---	---	150	100	200

Decision Network: Evaluation

The algorithm for decision network evaluation is as follows:

1. Set the evidence variables for the current state
2. For each possible value a of decision node:
   a. Set the decision node to that value
   b. Calculate the posterior probabilities for the parent nodes of the utility node
   c. Calculate the utility for the action / value a
3. Return the action with highest utility

Agent's Decisions

Recall that agent ACTIONS change the state:

- **if we are in state** s
- **action** a **is expected to**
- **lead to another state** s' **(outcome)**

Given uncertainty about the current state s **and action outcome** s' **we need to define the following:**

- **probability (belief) of being in state** s**:** $P(s)$
- **probability (belief) of action** a **leading to outcome** s'**:** $P(s' \mid s, a)$

Now:

$$P(s' \mid s, a) = P(\textsc{Result}(a) = s') = \sum_s P(s) * P(s' \mid s, a)$$

Expected Action Utility

The expected utility of an action a given the evidence is the average utility value of all possible outcomes s' of action a, weighted by their probability (belief) of occurence:

$$EU(a) = \sum_{s'} \sum_{s} P(s) * P(s' \mid s, a) * U(s') = \sum_{s'} P(Result(a) = s') * U(s')$$

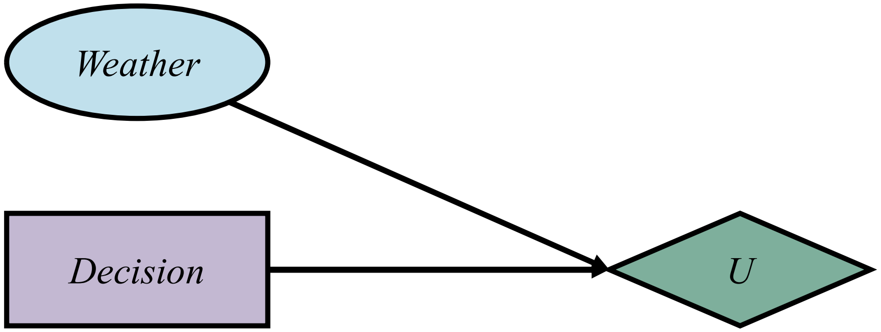
Rational agent should choose an action that maximizes the expected utility:

$$\text{chosen action} = \underset{a}{\text{argmax}}\ EU(a)$$

Decision Networks: Example

Decision: take umbrella

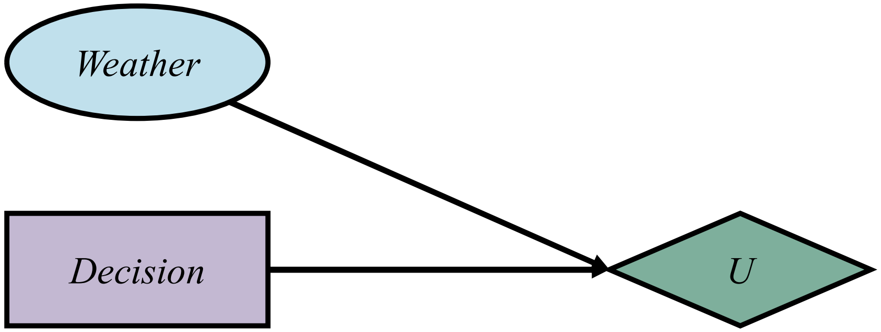
P(W=rain)	P(W=sun)
0.30	0.70



D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

Decision: leave umbrella

P(W=rain)	P(W=sun)
0.30	0.70



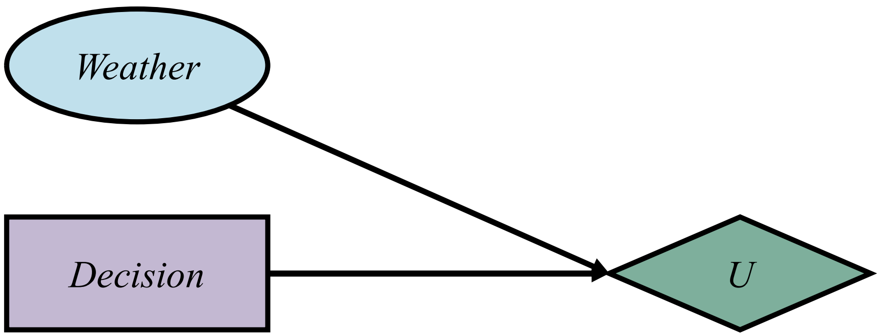
D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

Decision Networks: Example

Decision: take umbrella

$$EU(a) = \sum_{s'} P(Result(a) = s') * U(s')$$

P(W=rain)	P(W=sun)
0.30	0.70



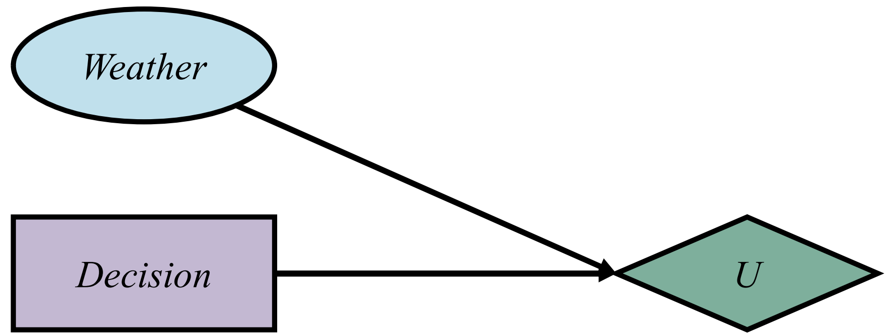
D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

$$EU(take) = ???$$

Decision: leave umbrella

$$EU(a) = \sum_{s'} P(Result(a) = s') * U(s')$$

P(W=rain)	P(W=sun)
0.30	0.70



D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

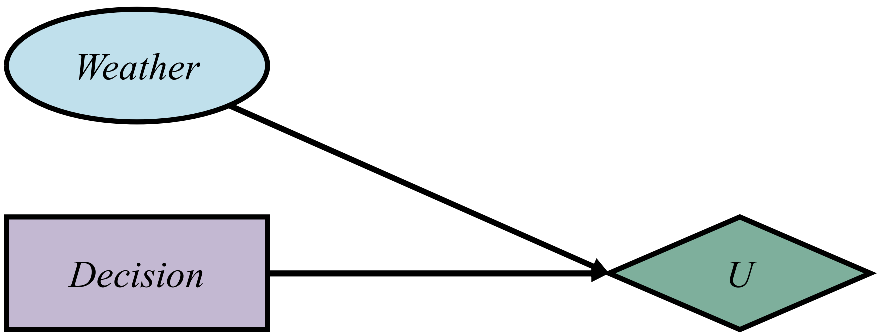
$$EU(leave) = ???$$

Decision Networks: Example

Decision: take umbrella

$EU(a) = \sum_{s'} P(Result(a) = s') * U(s')$

P(W=rain)	P(W=sun)
0.30	0.70



S_1': D = take, W = sun
S_2': D = take, W = rain
EU(take) =
P(Result(take) = S_1')*U(S_1') +
P(Result(take) = S_2')*U(S_2') =
0.70 * 20 + 0.30 * 70 = 35

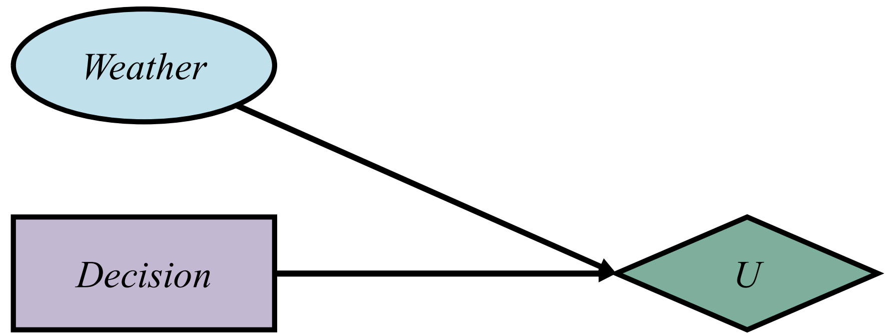
D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

$EU(take) = 35$

Decision: leave umbrella

$EU(a) = \sum_{s'} P(Result(a) = s') * U(s')$

P(W=rain)	P(W=sun)
0.30	0.70



S_3': D = leave, W = sun
S_4': D = leave, W = rain
EU(leave) =
P(Result(leave) = S_3')*U(S_3') +
P(Result(leave) = S_4')*U(S_4') =
0.70 * 100 + 0.30 * 0 = 70

D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

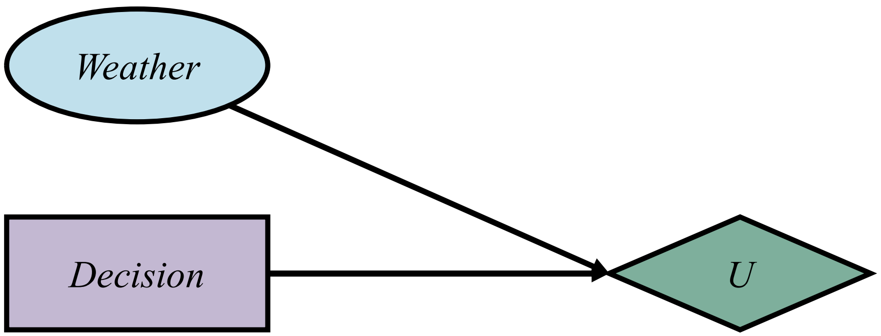
$EU(leave) = 70$

Decision Networks: Example

Which action to choose: take or leave Umbrella?

$$EU(a) = \sum_{s'} P(Result(a) = s') * U(s')$$

P(W=rain)	P(W=sun)
0.30	0.70

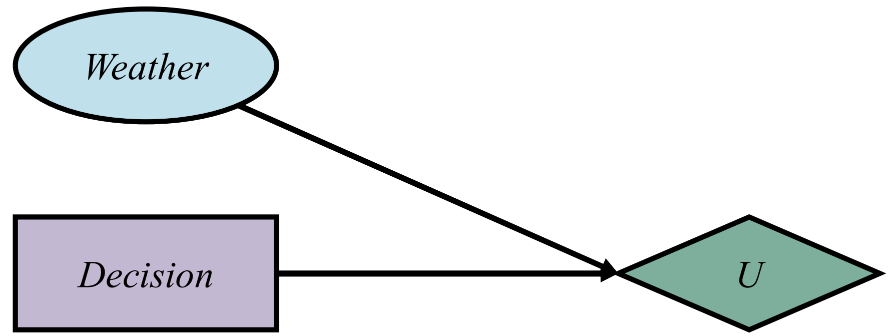


S_1': D = take, W = sun
S_2': D = take, W = rain
EU(take) =
P(Result(take) = S_1')*U(S_1') +
P(Result(take) = S_2')*U(S_2') =
0.70 * 20 + 0.30 * 70 = 35

D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

$$EU(a) = \sum_{s'} P(Result(a) = s') * U(s')$$

P(W=rain)	P(W=sun)
0.30	0.70



S_3': D = leave, W = sun
S_4': D = leave, W = rain
EU(leave) =
P(Result(leave) = S_3')*U(S_3') +
P(Result(leave) = S_4') U(S_4') =
0.70 * 100 + 0.30 * 0 = 70

D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

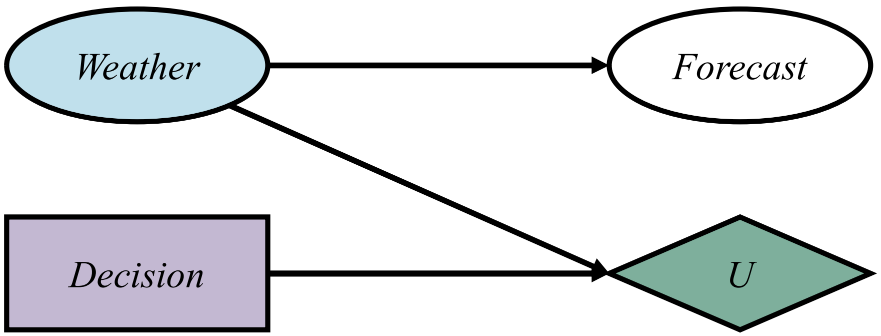
$$\text{action} = \underset{a}{\text{argmax}}\ EU(a) \mid \max(EU(take), \underline{EU(leave)}) = \max(35, \mathbf{70}) \rightarrow \text{leave}$$

Decision Networks: Example

Decision: take umbrella

P(W=rain)	P(W=sun)
0.30	0.70

P(F=sun)	P(F=rain)
0.59	0.41

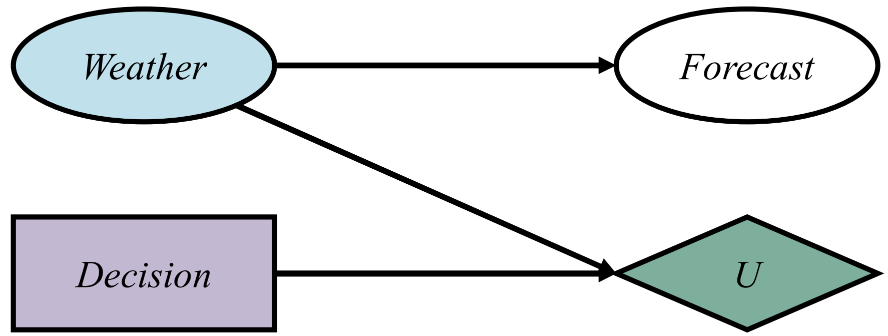


D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

Decision: leave umbrella

P(W=rain)	P(W=sun)
0.30	0.70

P(F=sun)	P(F=rain)
0.59	0.41



D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

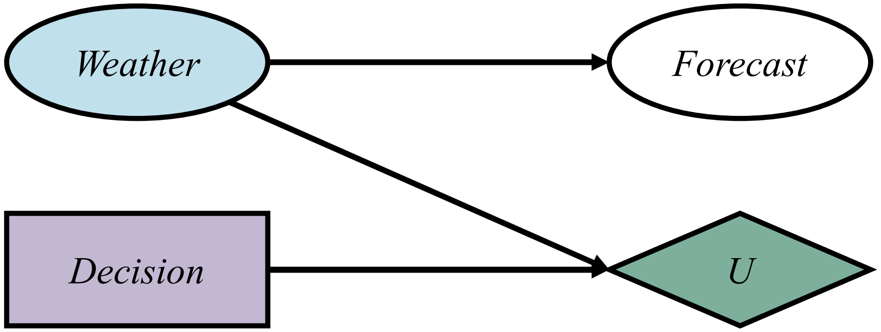
Decision Networks: Example

Decision: take umbrella given e

$\mathrm{EU}(a \mid e) = \sum_{s'} P(Result(a) = s' \mid e) * U(s')$

P(rain | F)	P(sun | F)
???	???

P(F=sun)	P(F=rain)
0.59	0.41



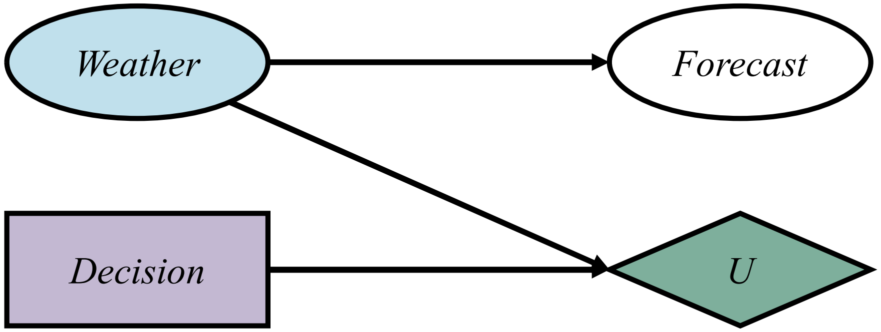
D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

Decision: leave umbrella given e

$\mathrm{EU}(a \mid e) = \sum_{s'} P(Result(a) = s' \mid e) * U(s')$

P(rain | F)	P(sun | F)
???	???

P(F=sun)	P(F=rain)
0.59	0.41



D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

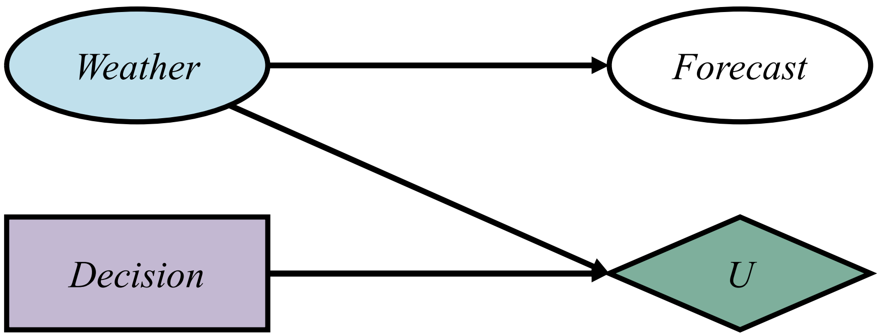
Decision Networks: Example

Decision: take umbrella given e

$$EU(a \mid e) = \sum_{s'} P(Result(a) = s' \mid e) * U(s')$$

P(W=rain)	P(W=sun)
0.30	0.70

P(F=sun)	P(F=rain)
0.59	0.41



D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

Conditional probabilities

Assume that we are given:

F	W	P(F|W)
sun	sun	0.80
rain	sun	0.20
sun	rain	0.10
rain	rain	0.90

By Bayes' Theorem:

$$P(W = sun \mid F = sun) = \frac{P(F = sun \mid W = sun) * P(W = sun)}{P(F = sun)} = \frac{0.80 * 0.70}{0.59} = 0.95$$

$$P(W = sun \mid F = rain) = \frac{P(F = rain \mid W = sun) * P(W = sun)}{P(F = rain)} = \frac{0.20 * 0.70}{0.41} = 0.34$$

$$P(W = rain \mid F = sun) = \frac{P(F = sun \mid W = rain) * P(W = rain)}{P(F = sun)} = \frac{0.10 * 0.30}{0.59} = 0.05$$

$$P(W = rain \mid F = rain) = \frac{P(F = rain \mid W = rain) * P(W = rain)}{P(F = rain)} = \frac{0.90 * 0.30}{0.41} = 0.66$$

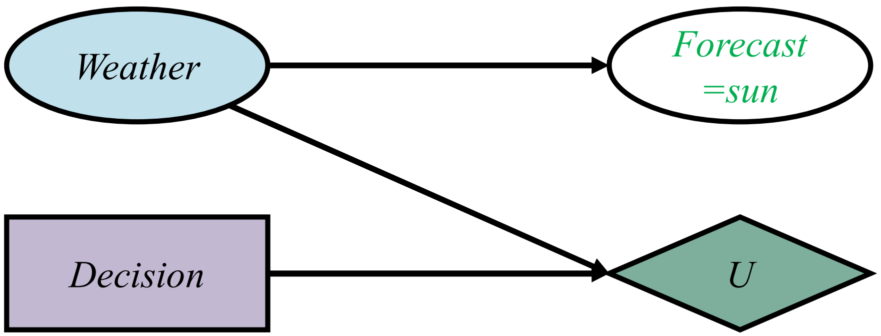
Decision Networks: Example

Decision: take umbrella given sun

$$EU(a \mid e) = \sum_{s'} P(Result(a) = s' \mid e) * U(s')$$

P(rain | F)	P(sun | F)
0.05	0.95

P(F=sun)	P(F=rain)
0.59	0.41



D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

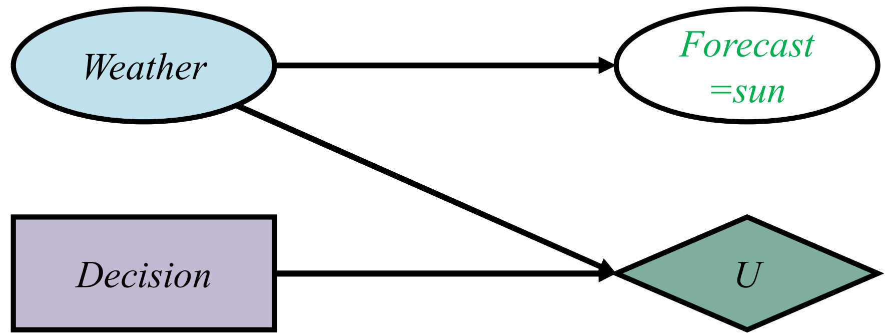
EU(take given sun forecast) = ? ? ?

Decision: leave umbrella given sun

$$EU(a \mid e) = \sum_{s'} P(Result(a) = s' \mid e) * U(s')$$

P(rain | F)	P(sun | F)
0.05	0.95

P(F=sun)	P(F=rain)
0.59	0.41



D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

EU(leave given sun forecast) = ? ? ?

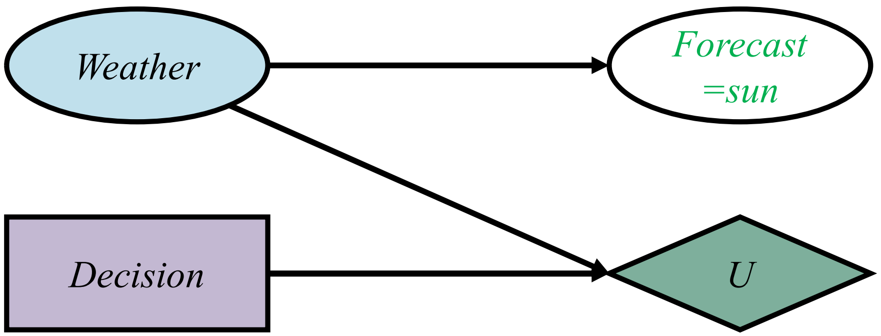
Decision Networks: Example

Decision: take umbrella given sun

$$\mathrm{EU}(a \mid e) = \sum_{s'} \mathrm{P}(Result(a) = s' \mid e) * \mathrm{U}(s')$$

P(rain | F)	P(sun | F)
0.05	0.95

P(F=sun)	P(F=rain)
0.59	0.41



S_1': D = take, W = sun
S_2': D = take, W = rain
EU(take) =
P(Result(take)=S_1'|e)*U(S_1') +
P(Result(take)=S_2'|e)*U(S_2') =
0.95 * 20 + 0.05 * 70 = 22.5

D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

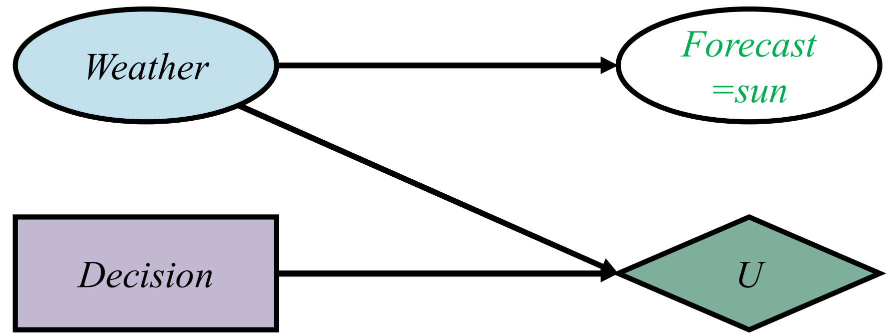
EU(take given sun forecast) = 22.5

Decision: leave umbrella given sun

$$\mathrm{EU}(a \mid e) = \sum_{s'} \mathrm{P}(Result(a) = s' \mid e) * \mathrm{U}(s')$$

P(rain | F)	P(sun | F)
0.05	0.95

P(F=sun)	P(F=rain)
0.59	0.41



S_3': D = leave, W = sun
S_4': D = leave, W = rain
EU(leave) =
P(Result(leave)=S_3'|e)*U(S_3') +
P(Result(leave)=S_4'|e)*U(S_4') =
0.95 * 100 + 0.05 * 0 = 95

D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

EU(leave given sun forecast) = 95

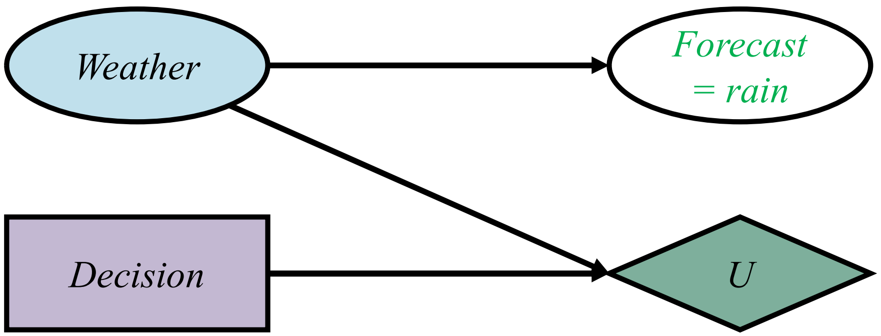
Decision Networks: Example

Decision: take umbrella given rain

$$EU(a \mid e) = \sum_{s'} P(Result(a) = s' \mid e) * U(s')$$

P(rain | F)	P(sun | F)
0.66	0.34

P(F=sun)	P(F=rain)
0.59	**0.41**



D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

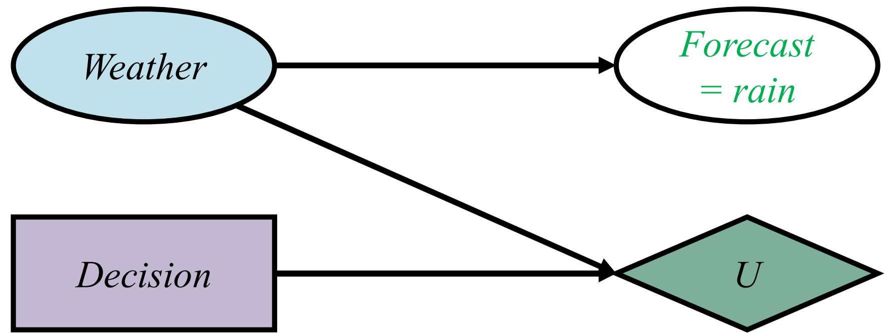
EU(take given rain forecast) = ???

Decision: leave umbrella given rain

$$EU(a \mid e) = \sum_{s'} P(Result(a) = s' \mid e) * U(s')$$

P(rain | F)	P(sun | F)
0.66	0.34

P(F=sun)	P(F=rain)
0.59	**0.41**



D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

EU(leave given rain forecast) = ???

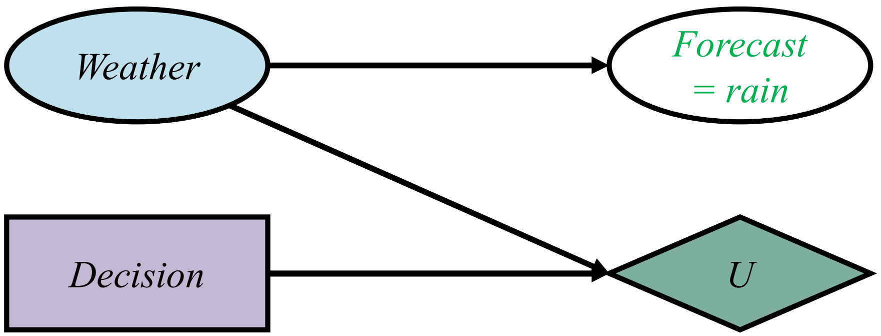
Decision Networks: Example

Decision: take umbrella given rain

$$EU(a \mid e) = \sum_{s'} P(Result(a) = s' \mid e) * U(s')$$

P(rain | F)	P(sun | F)
0.66	0.34

P(F=sun)	P(F=rain)
0.59	**0.41**



S_1': D = take, W = sun
S_2': D = take, W = rain
EU(take) =
P(Result(take)=S_1'|e)*U(S_1') +
P(Result(take)=S_2'|e)*U(S_2') =
0.34 * 20 + 0.66 * 70 = 53

D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

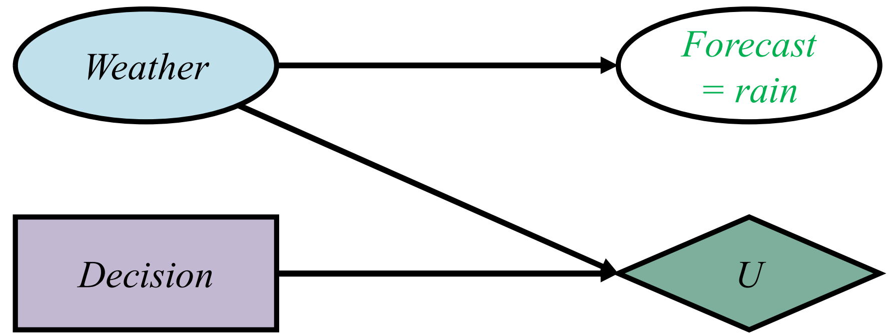
EU(take given rain forecast) = 53

Decision: leave umbrella given rain

$$EU(a \mid e) = \sum_{s'} P(Result(a) = s' \mid e) * U(s')$$

P(rain | F)	P(sun | F)
0.66	0.34

P(F=sun)	P(F=rain)
0.59	**0.41**



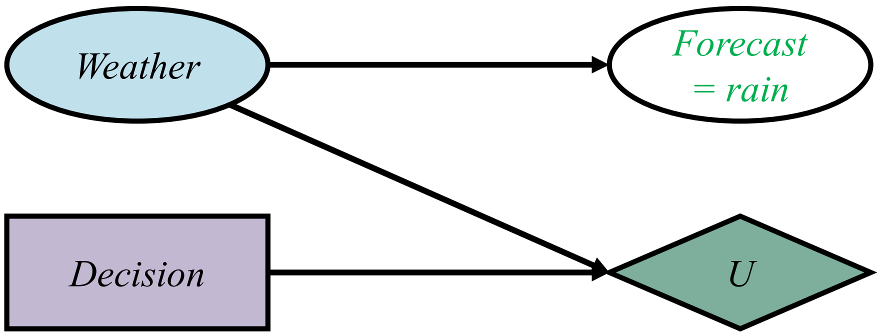
S_3': D = leave, W = sun
S_4': D = leave, W = rain
EU(leave) =
P(Result(leave)=S_3'|e)*U(S_3') +
P(Result(leave)=S_4'|e)*U(S_4') =
0.34 * 100 + 0.66 * 0 = 34

D	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

EU(leave given rain forecast) = 34

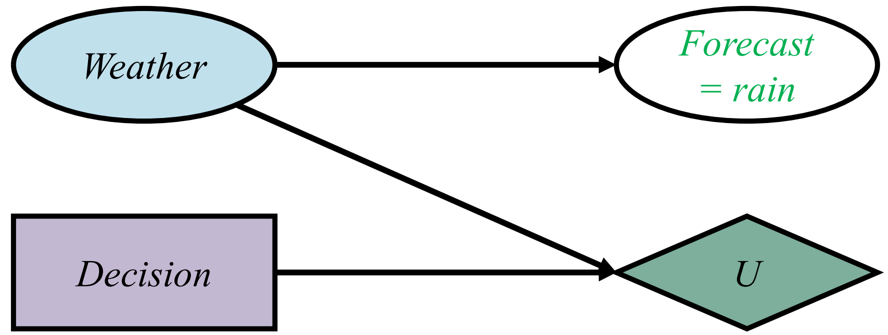
Decision Networks: Example

Decision: take umbrella given rain



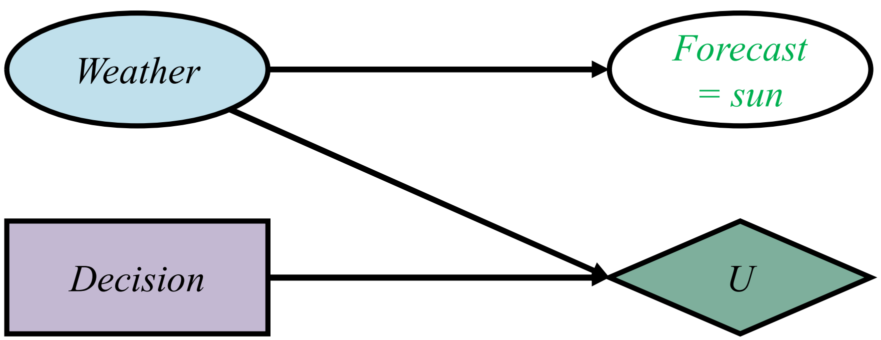
EU(take given rain forecast) = 53

Decision: leave umbrella given rain



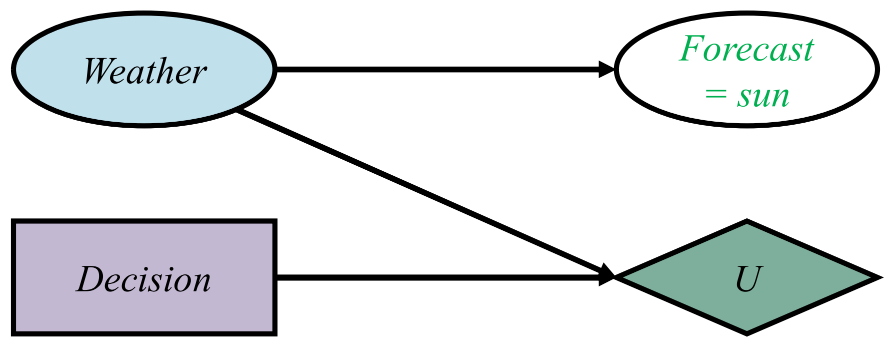
EU(leave given rain forecast) = 34

Decision: take umbrella given sun



EU(take given sun forecast) = 22.5

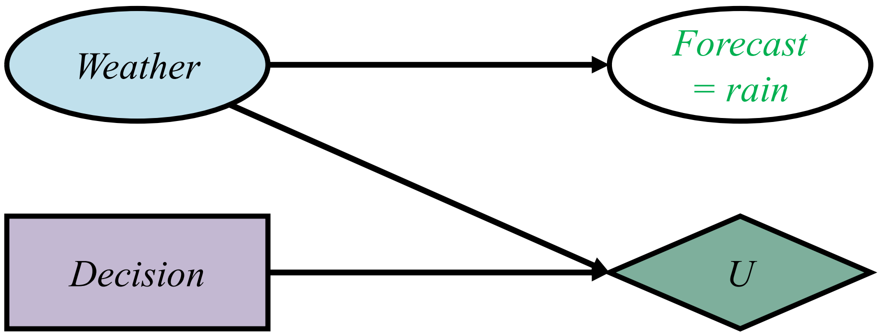
Decision: leave umbrella given sun



EU(leave given sun forecast) = 95

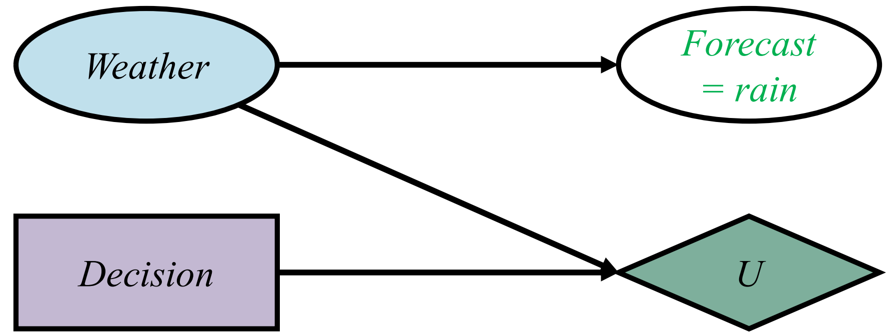
Decision Networks: Example

Decision:take umbrella given rain



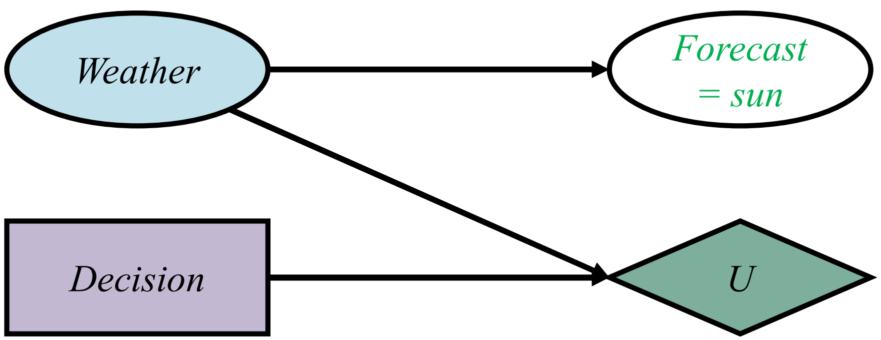
EU(take given rain forecast) = 53

Decision:leave umbrella given rain



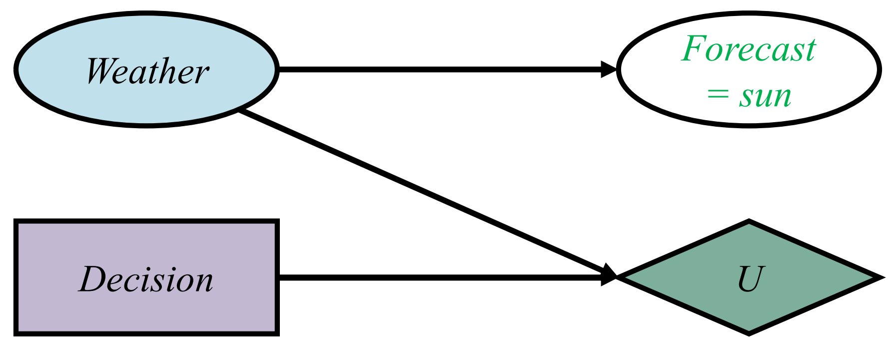
EU(leave given rain forecast) = 34

Decision:take umbrella given sun



EU(take given sun forecast) = 22.5

Decision:leave umbrella given sun



EU(leave given sun forecast) = 95

Value of Perfect Information

The value/utility of best action α without additional evidence (information) is :

$$\mathrm{MEU}(\alpha) = \max_{a} \textstyle\sum_{s'} \mathrm{P}(Result(a) = s') * \mathrm{U}(s')$$

If we include new evidence/information ($\mathrm{E_j} = \mathrm{e_j}$) given by some variable Ej, value/utility of best action α becomes:

$$\mathrm{MEU}(a_{e_j} \mid e_j) = \max_{a} \textstyle\sum_{s'} \mathrm{P}(Result(a) = s' \mid e_j) * \mathrm{U}(s')$$

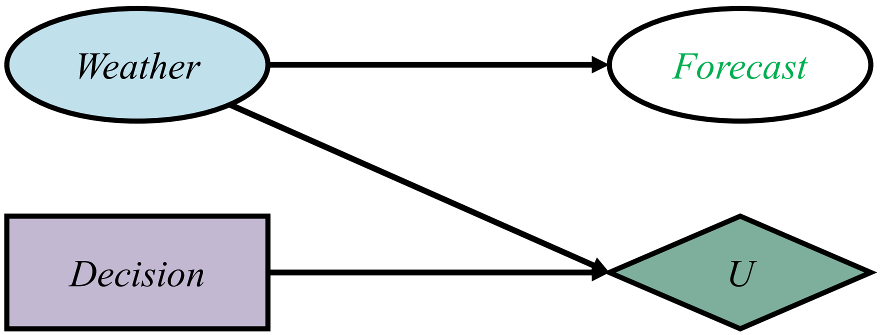
The value of additional evidence/information from Ej is:

$$\mathrm{VPI}(E_j) = \left(\textstyle\sum_{e_j} \mathrm{P}(E_j = e_j) * \mathrm{MEU}(a_{e_j} \mid E_j = e_j)\right) - MEU(a)$$

using our current beliefs about the world.

Decision Network: Example

Decision network



The value of best action α without additional evidence

$$\mathrm{MEU}(\alpha) = \mathrm{MEU}(\text{leave}) = 70$$

With evidence information ($E_j = e_j$) given by Forecast:

$$\mathrm{MEU}(a_{e_1} \mid e_1) = \mathrm{MEU}(\text{take} \mid F = rain) = 53$$

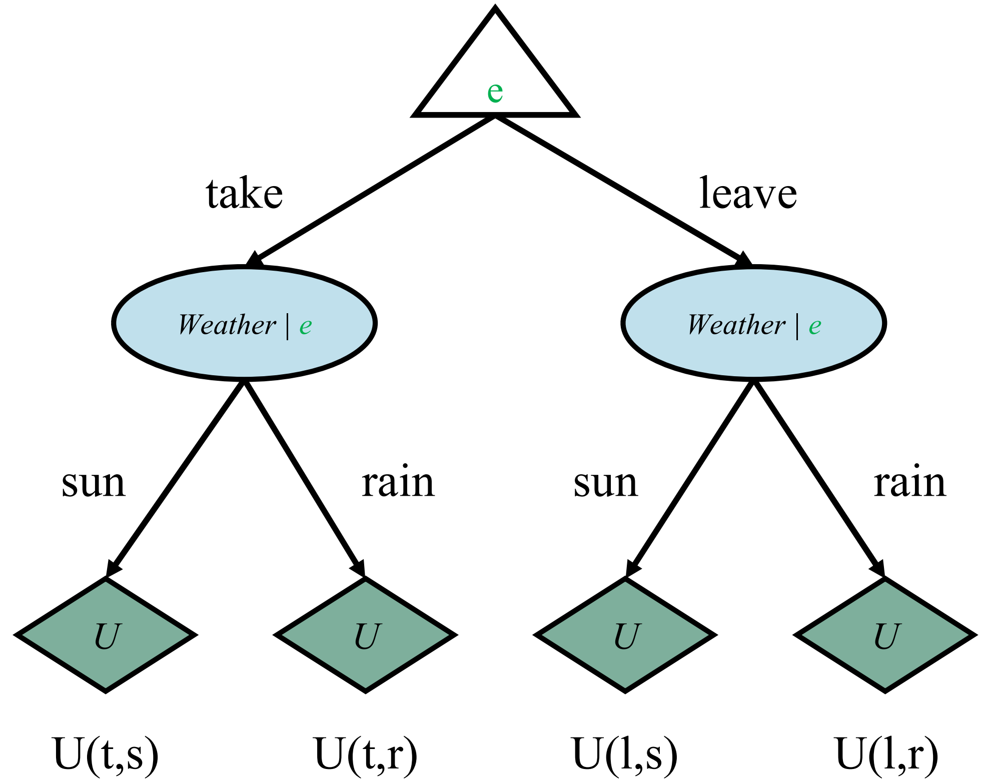
$$\mathrm{MEU}(a_{e_2} \mid e_2) = \mathrm{MEU}(\text{leave} \mid F = sun) = 95$$

The value of additional evidence / information from F is:

$$\mathrm{VPI}(E_j) = \left(\sum_{e_j} \mathrm{P}(E_j = e_j) * \mathrm{MEU}(a_{e_j} \mid E_j = e_j)\right) - MEU(a)$$

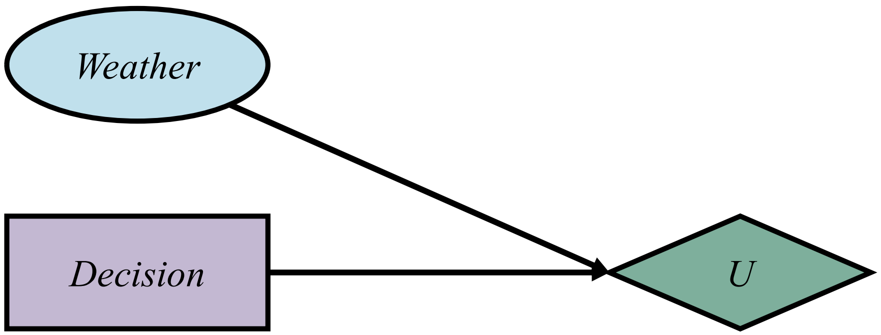
$$\mathrm{VPI}(F) = (\mathrm{P}(F = rain) * \mathrm{MEU}(take \mid F = rain) + \mathrm{P}(F = sun) * \mathrm{MEU}(\text{leave} \mid F = sun)) - MEU(leave) = (0.41 * 53 + 0.59 * 95) - 70 = 7.78$$

Outcome tree



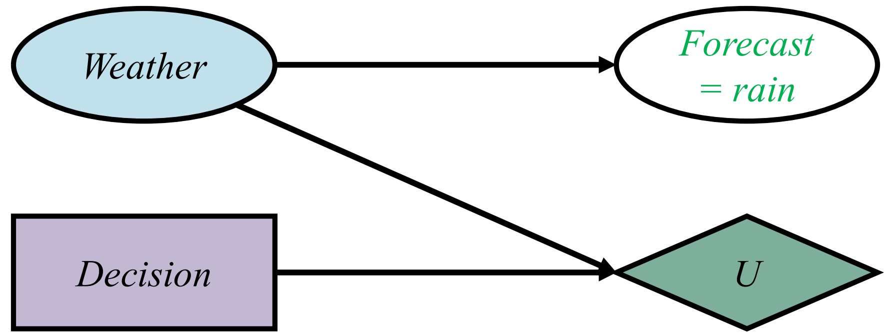
Decision Networks: Example

Decision: leave umbrella



$\text{EU(leave)} = 70$

Decision: take umbrella given rain



$\text{EU(take given rain forecast)} = 53$

The value of best action α without additional evidence

$$\text{MEU}(\alpha) = \text{MEU(leave)} = 70$$

With evidence information ($\text{E}_\text{j} = \text{e}_\text{j}$) given by Forecast:

$$\text{MEU}(a_{e_1} \mid e_1) = \text{MEU(take} \mid F = rain) = 53$$

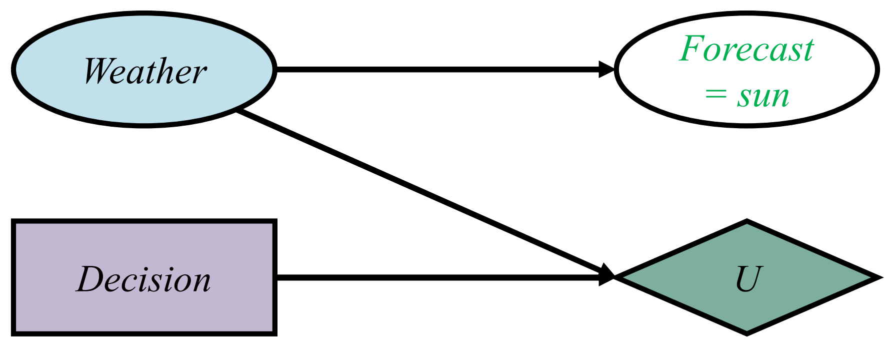
$$\text{MEU}(a_{e_2} \mid e_2) = \text{MEU(leave} \mid F = sun) = 95$$

The value of additional evidence / information from F is:

$$\text{VPI}(E_j) = \left(\sum_{e_j} \text{P}(E_j = e_j) * \text{MEU}(a_{e_j} \mid E_j = e_j)\right) - MEU(a)$$

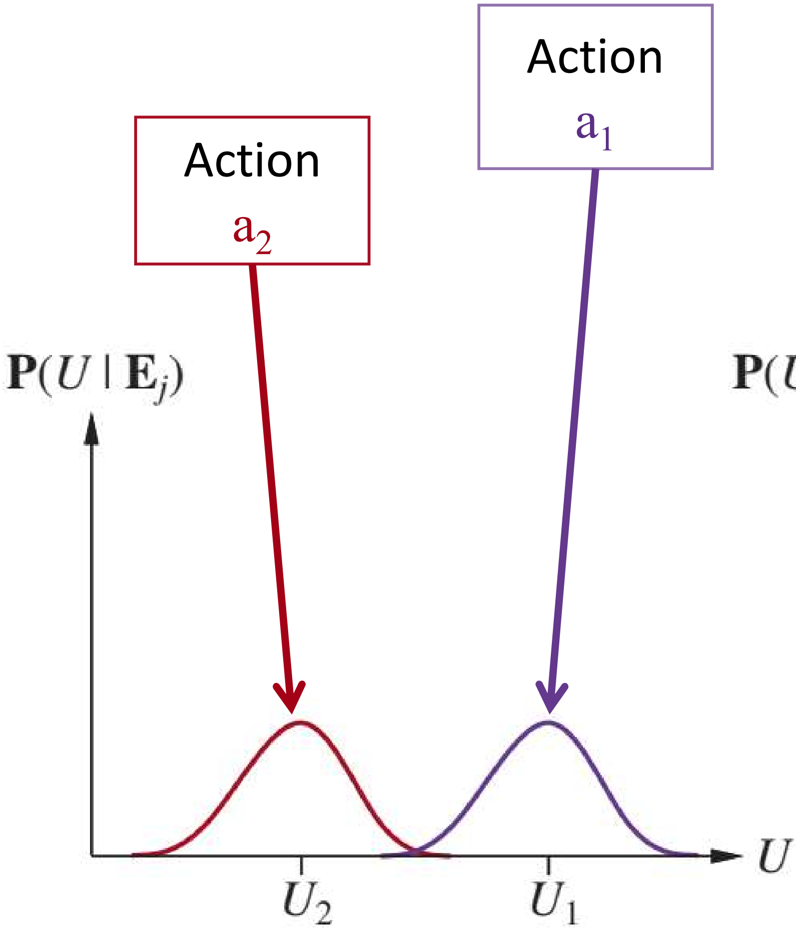
$$\text{VPI}(F) = (\text{P}(F = rain) * \text{MEU}(take \mid F = rain) + \text{P}(F = sun) * \text{MEU(leave} \mid F = sun)) - MEU(leave) = (0.41 * 53 + 0.59 * 95) - 70 = 7.78$$

Decision: leave umbrella given sun

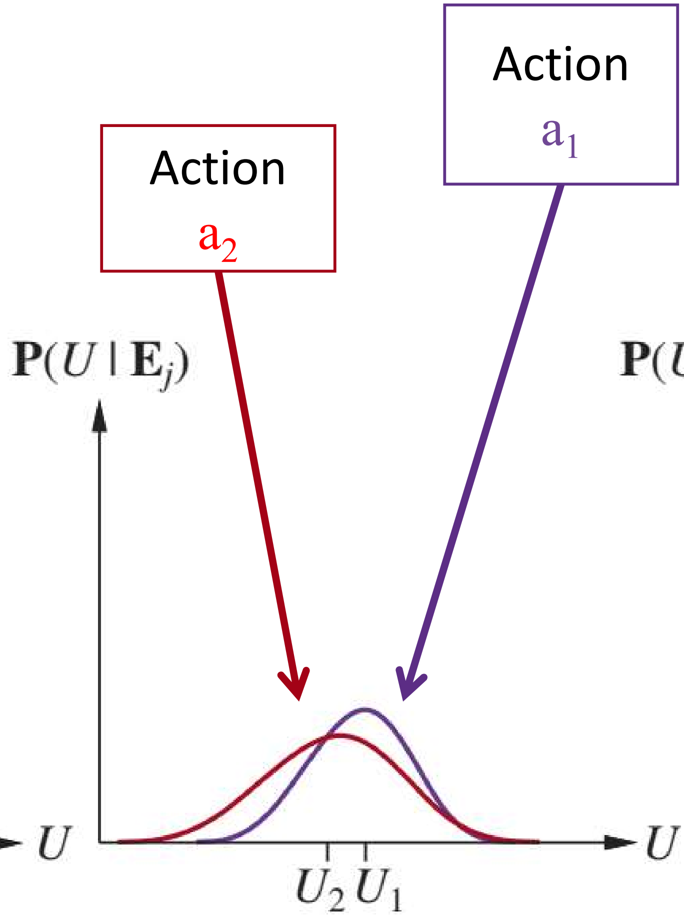


$\text{EU(leave given sun forecast)} = 95$

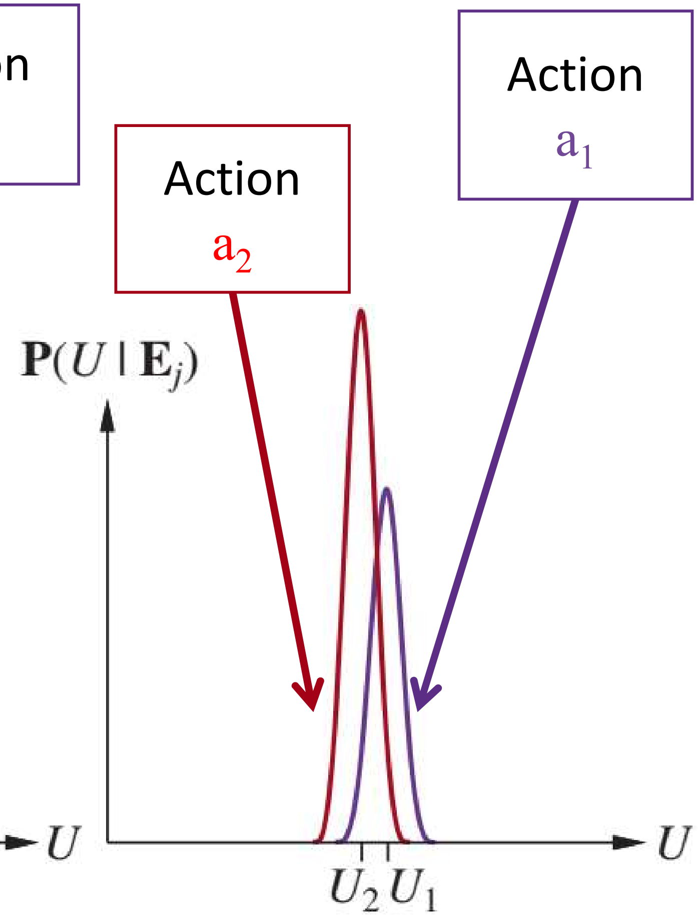
Utility & Value of Perfect Information



New information will not help here.



New information may help a lot here.



New information may help a bit here.

VPI Properties

Given a decision network with possible observations $\mathrm{E_j}$ (sources of new information / evidence):

- **The expected value of information is nonnegative:**

$$\forall_{\mathrm{j}}\ \mathrm{VPI}(E_j) \geq 0$$

- **VPI is not additive:**

$$\mathrm{VPI}(E_j, E_k) \neq \mathrm{VPI}(E_j) + \mathrm{VPI}(E_k)$$

- **VPI is order-independent:**

$$\mathrm{VPI}(E_j,\ E_k) = VPI(E_j) + VPI(E_k \mid E_j) = VPI(E_k) + VPI(E_j \mid E_k) = VPI(E_k, E_j)$$

Information Gathering Agent

function INFORMATION-GATHERING-AGENT(*percept*) **returns** an *action*
  persistent: D, a decision network

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