CS 480

Introduction to Artificial Intelligence

November 2nd, 2021

Announcements / Reminders

- **Programming Assignment #01:**
  - **due: ~~October 17th~~ ~~October 22th~~ ~~October 24th~~ November 3rd, 11:00 PM CST**
- **Programming Assignment #02:**
  - **TBA**
- **Written Assignment #03:**
  - **TBA**

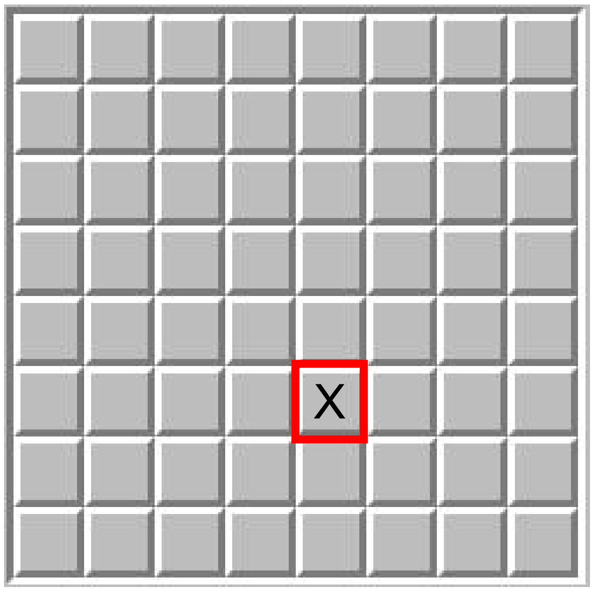
Plan for Today

- **Conditional Independence revisited**
  - **Markov models [BONUS MATERIAL]**
- **Fuzzy logic [BONUS MATERIAL]**
- **Making simple decisions**

Playing Minesweeper with Bayes' Rule

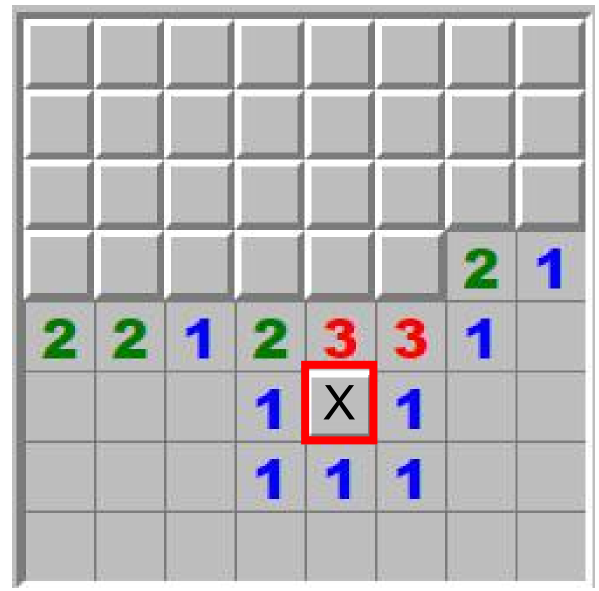
Prior probability / belief:

$$P(X = mine) = 0.5$$

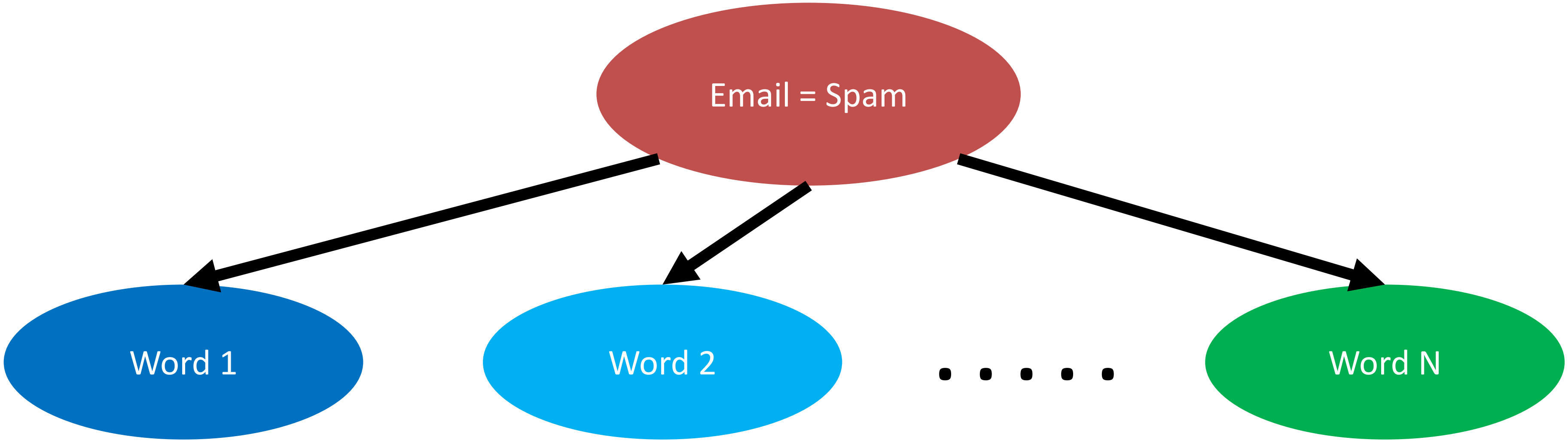


Posterior probability / belief:

$$P(X = mine \mid \boldsymbol{evidence}) = 1.0$$



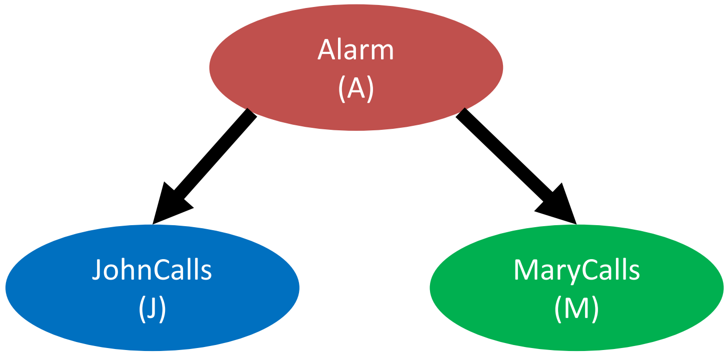
Naive Bayes Spam Filter



P(Email = spam | Word1) = 0.09
P(Email = spam | Word2) = 0.01
...
P(Email = spam | WordN) = 0.03

Conditional Independence

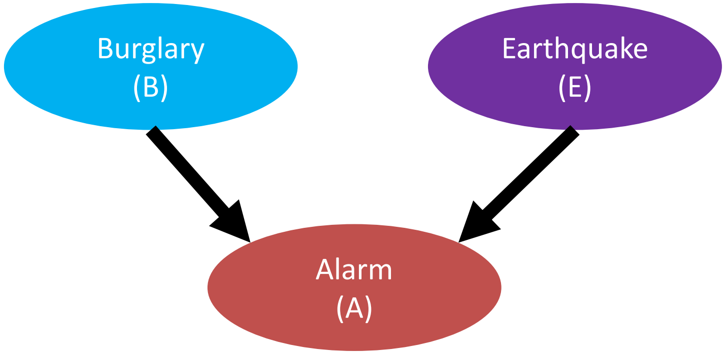
Common Cause:



JohnCalls and MaryCalls
are NOT independent

JohnCalls and MaryCalls are CONDITIONALLY independent given Alarm

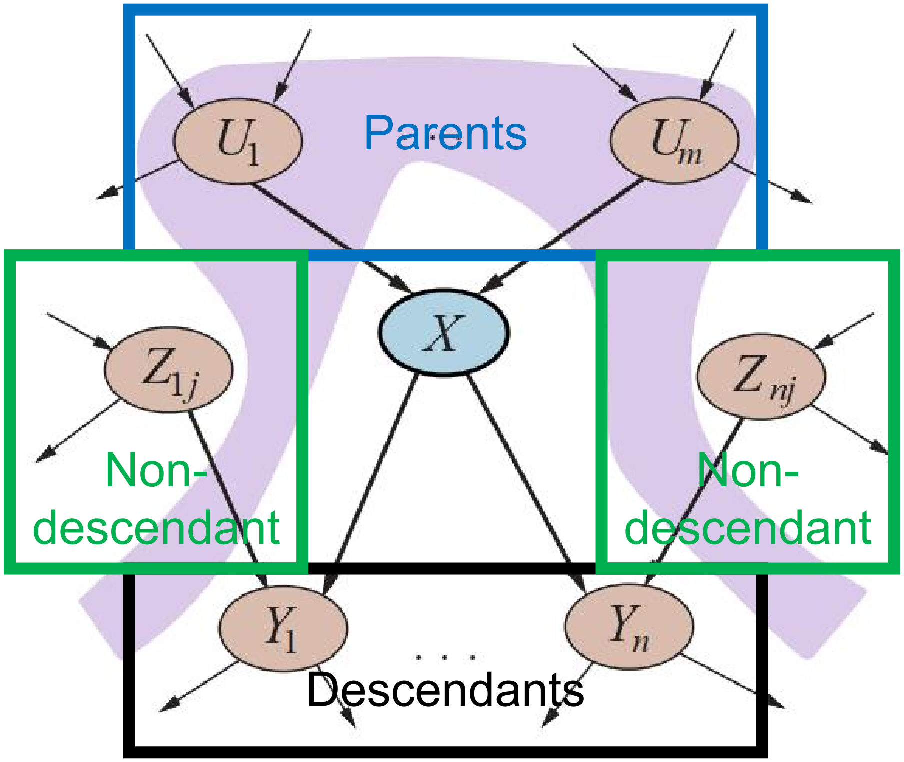
Common Effect:



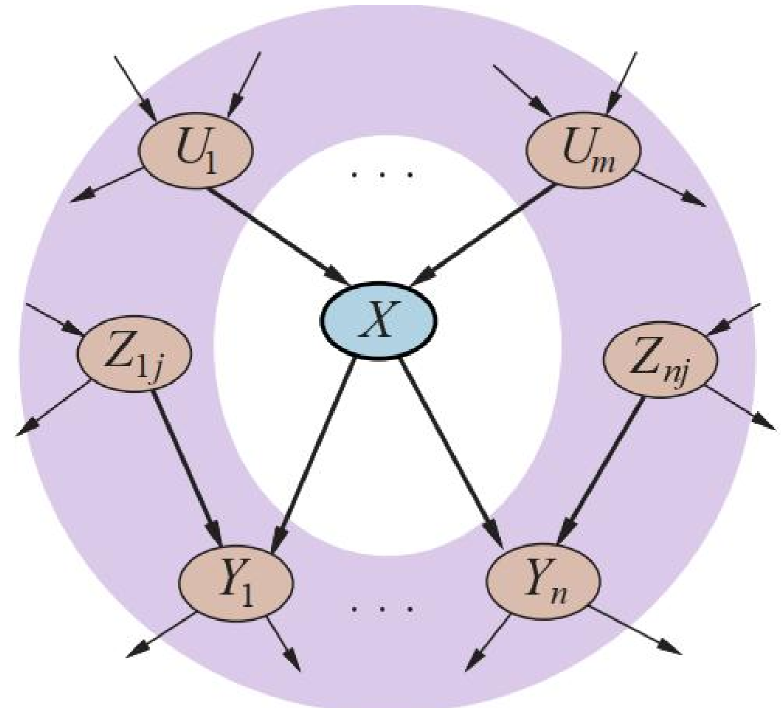
Burglary and Earthquake
are independent

Burglary and Earthquake are NOT CONDITIONALLY independent given Alarm

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 ($\mathrm{k} < \mathrm{N}$).

Chain Rule

Conditional probabilities can be used to decompose conjunctions using the chain rule. For any propositions $\boldsymbol{f_1, f_2, \ldots, f_n}$:

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

$$P(f_1) \; *$$

$$P(f_2 \mid f_1) \; *$$

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

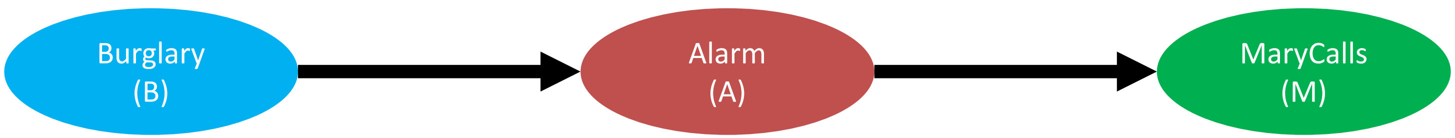
$$\ldots$$

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

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

Conditional Independence

Causal Chain:



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

Burglary and MaryCalls are CONDITIONALLY independent given Alarm.

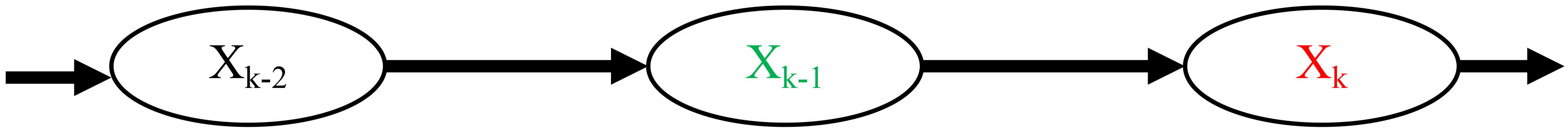
If Alarm is given, what “happened before” does not directly influence MaryCalls.

BONUS MATERIAL

Markov Chains / Markov Property

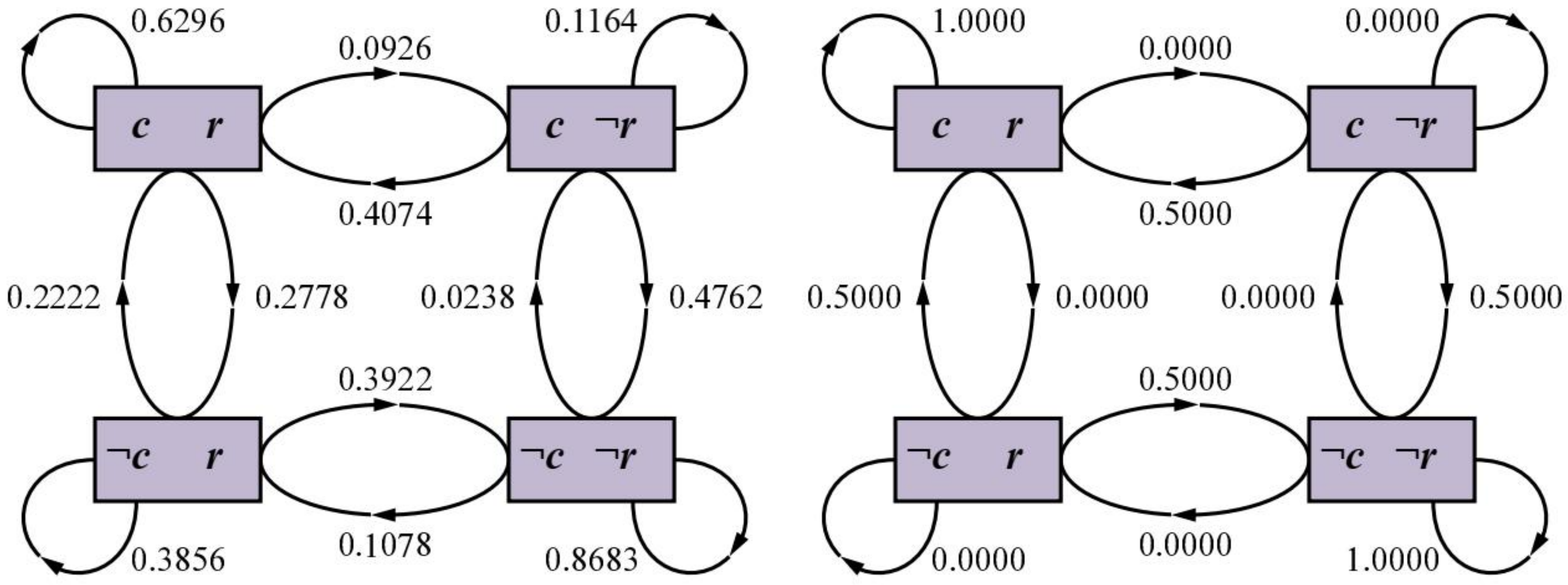
A sequence of random variables $\{X_i\}$ is called a Markov chain if it has the Markov property (<u>memoryless</u> property):

$$P(X_k = a \mid X_{k-1} = b, X_{k-2} = c, \ldots, X_1 = z) = P(X_k = a \mid X_{k-1} = b)$$



Markov Model

A Markov model is a stochastic model used to model (pseudo-) randomly changing systems.
Its key future is the assumpton that future states depend only on the current state, not on the events that occurred before it (it assumes the Markov property).



Check out this demo: https://setosa.io/ev/markov-chains/

Unknown Joint Probability Distribution

What if you don't know joint probability and direct sampling is difficult?

N Random Variables						
X_1	X_2	X_3	...	X_{N-1}	X_N	$P(X_1 \wedge X_2 \wedge X_3 \wedge \ldots \wedge X_N)$
true	true	true	...	true	true	???
true	true	true	...	true	false	???
true	true	false	...	false	true	???
...	...	...	...	...	...	???
false	false	true	...	true	false	???
false	false	true	...	false	true	???
false	false	false	...	false	false	???

Unknown Joint Probability Distribution

If you know the conditionals you can simulate a sequence of observation by sampling from conditional probability distribution (using Markov Chain Monte Carlo methods, such as Gibbs algorithm).

N Random Variables						
X_1	X_2	X_3	...	X_{N-1}	X_N	$P(X_1 \wedge X_2 \wedge X_3 \wedge \ldots \wedge X_N)$
true	true	true	...	true	true	???
true	true	true	...	true	false	???
true	true	false	...	false	true	???
...	...	...	...	...	...	???
false	false	true	...	true	false	???
false	false	true	...	false	true	???
false	false	false	...	false	false	???

Fuzzy Logic: the Idea

- **Boolean ("crisp") logic**

true | false

- **Fuzzy (many valued) logic**

true | false

Fuzzy Logic: the Idea

- **Boolean ("crisp") logic**

cold | hot

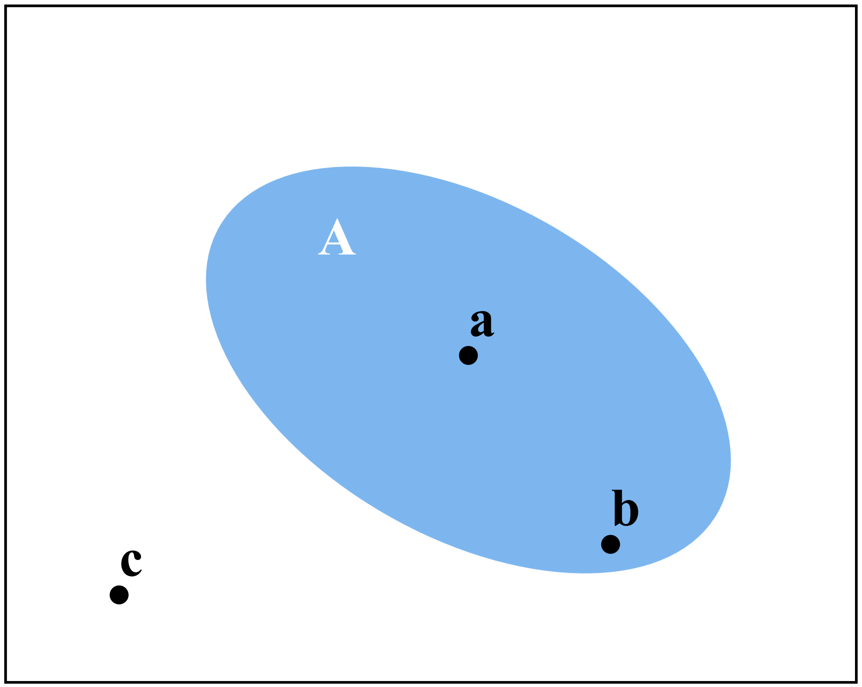
- **Fuzzy (many valued) logic**

cold | warm | hot

Fuzzy Logic: Fuzzy Sets

"Crisp" Set A

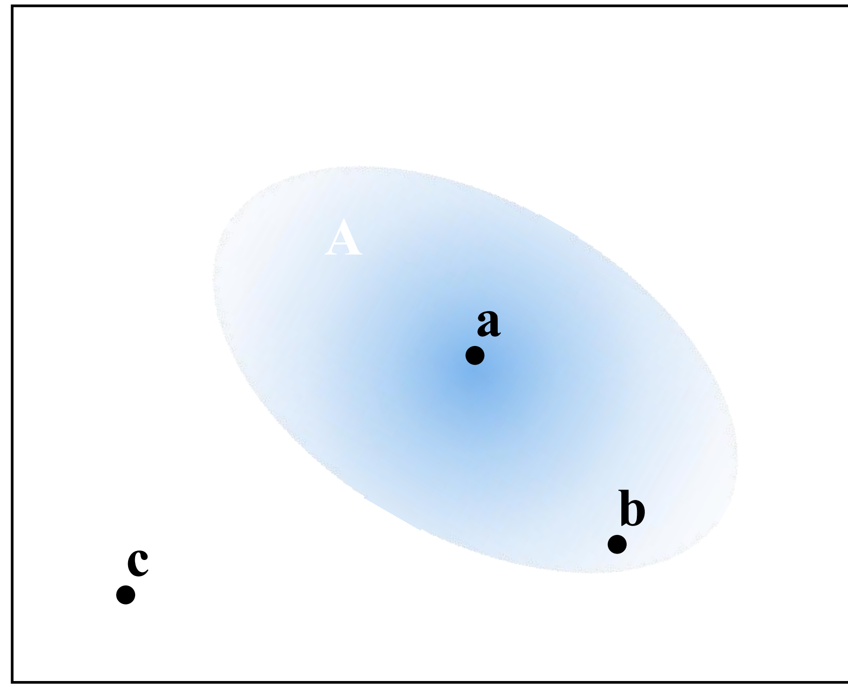
an element is a set member or not



$a \in A$
$b \in A$
$c \notin A$

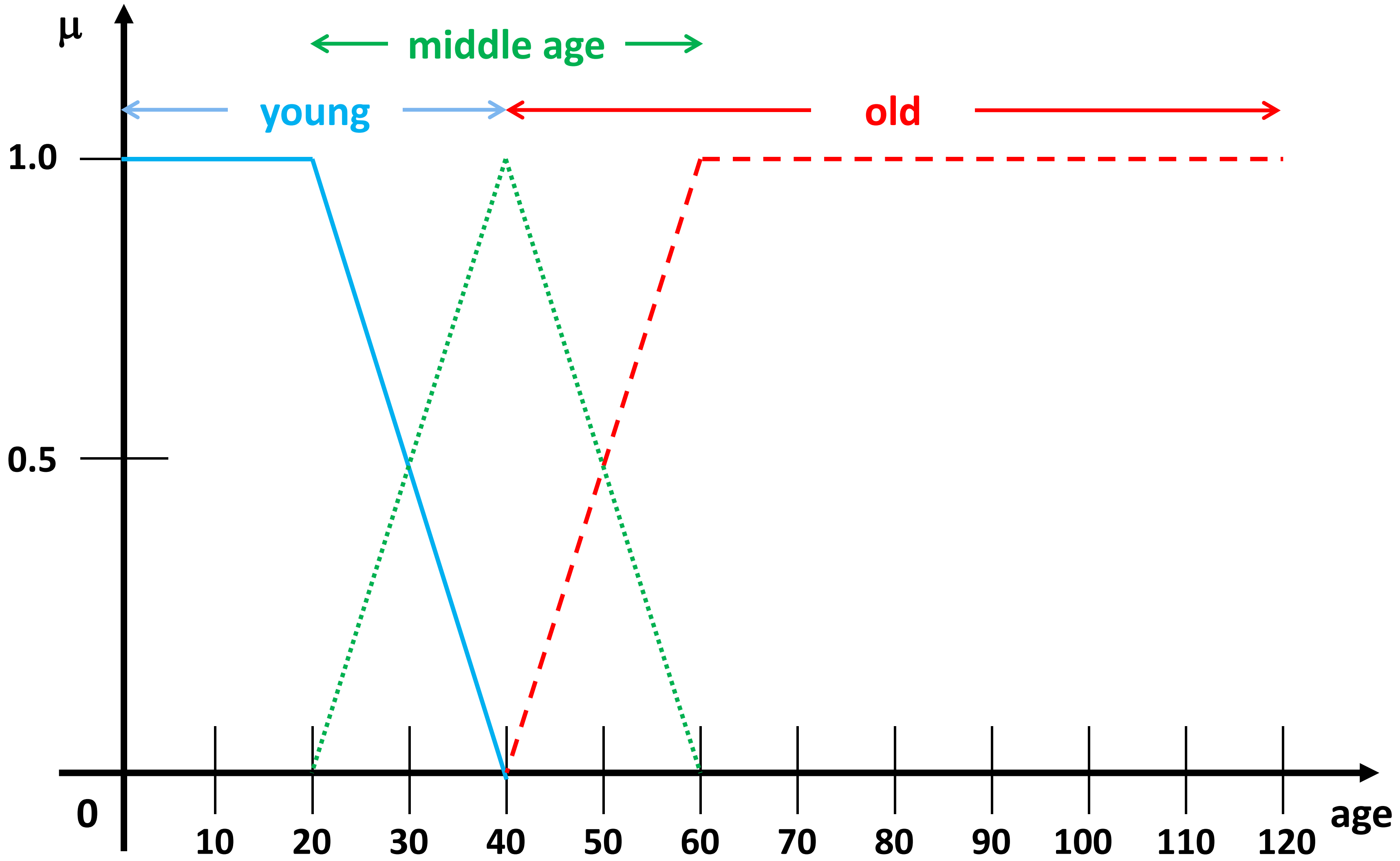
Fuzzy Set A:

an element is a set member
with some membership degree μ

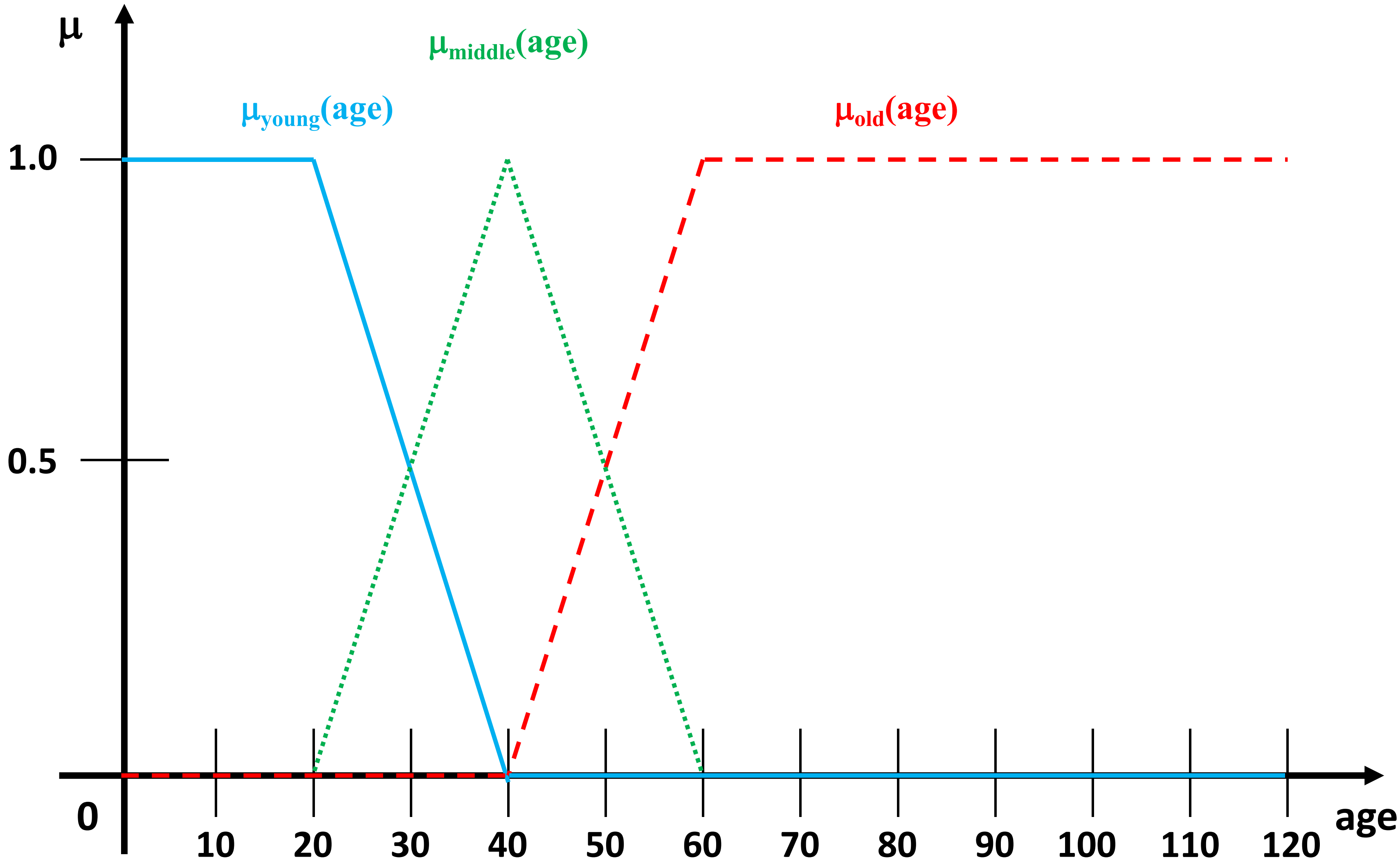


$\mu(a) = 1.0$
$\mu(b) = 0.1$
$\mu(c) = 0.0$

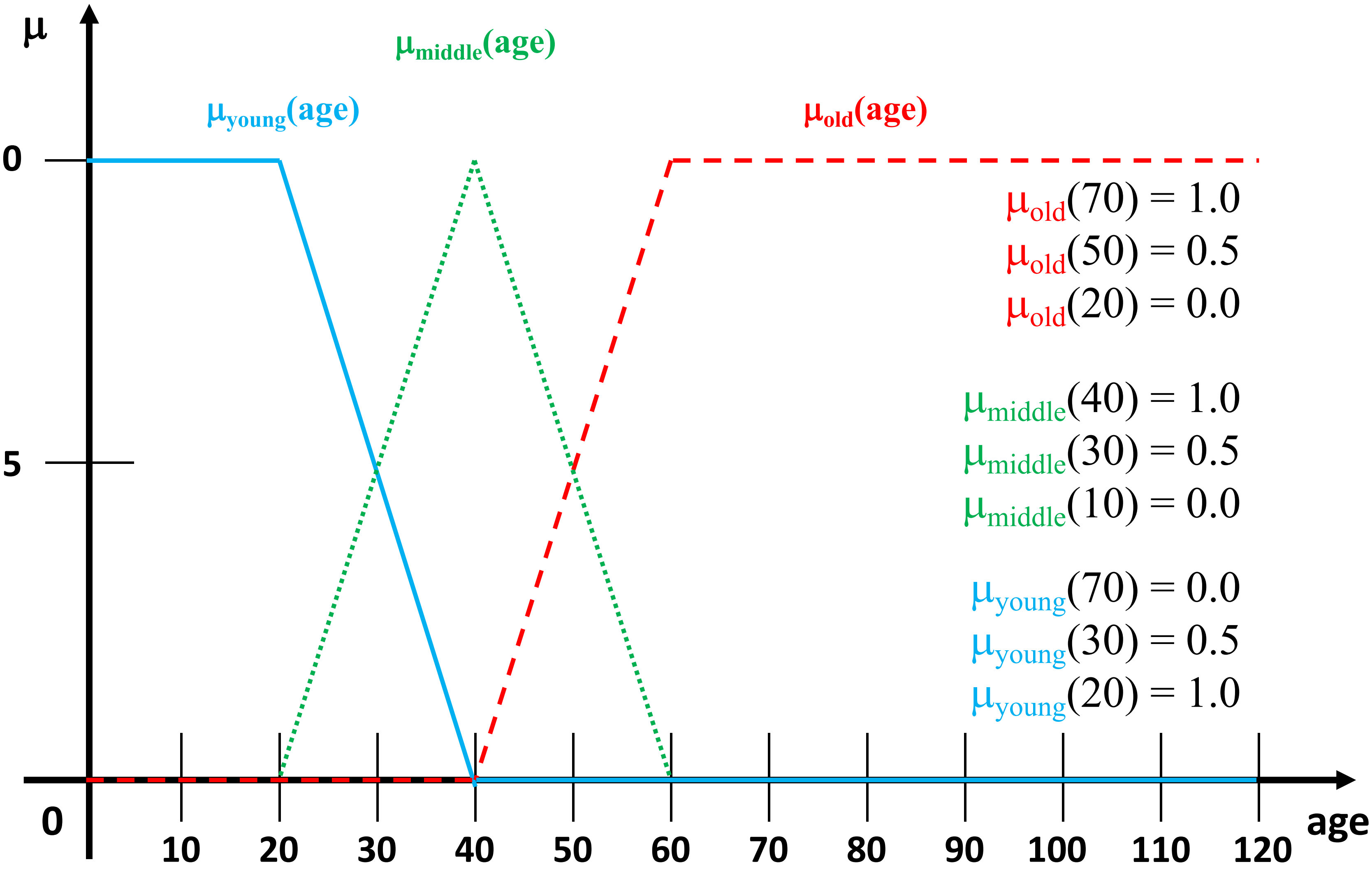
Fuzzy Logic: Fuzzy Sets



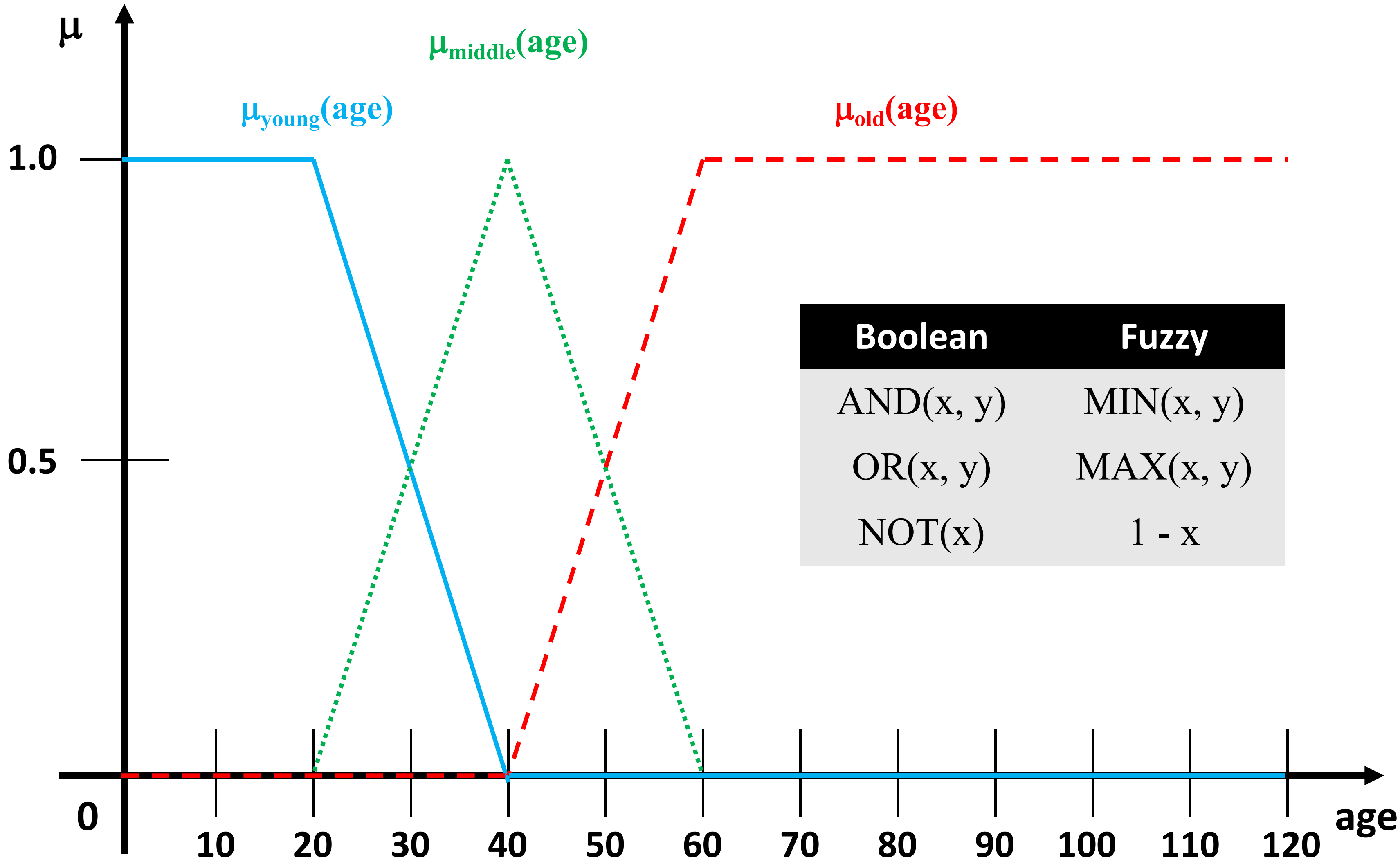
Fuzzy Logic: Membership Functions



Fuzzy Logic: Membership Functions

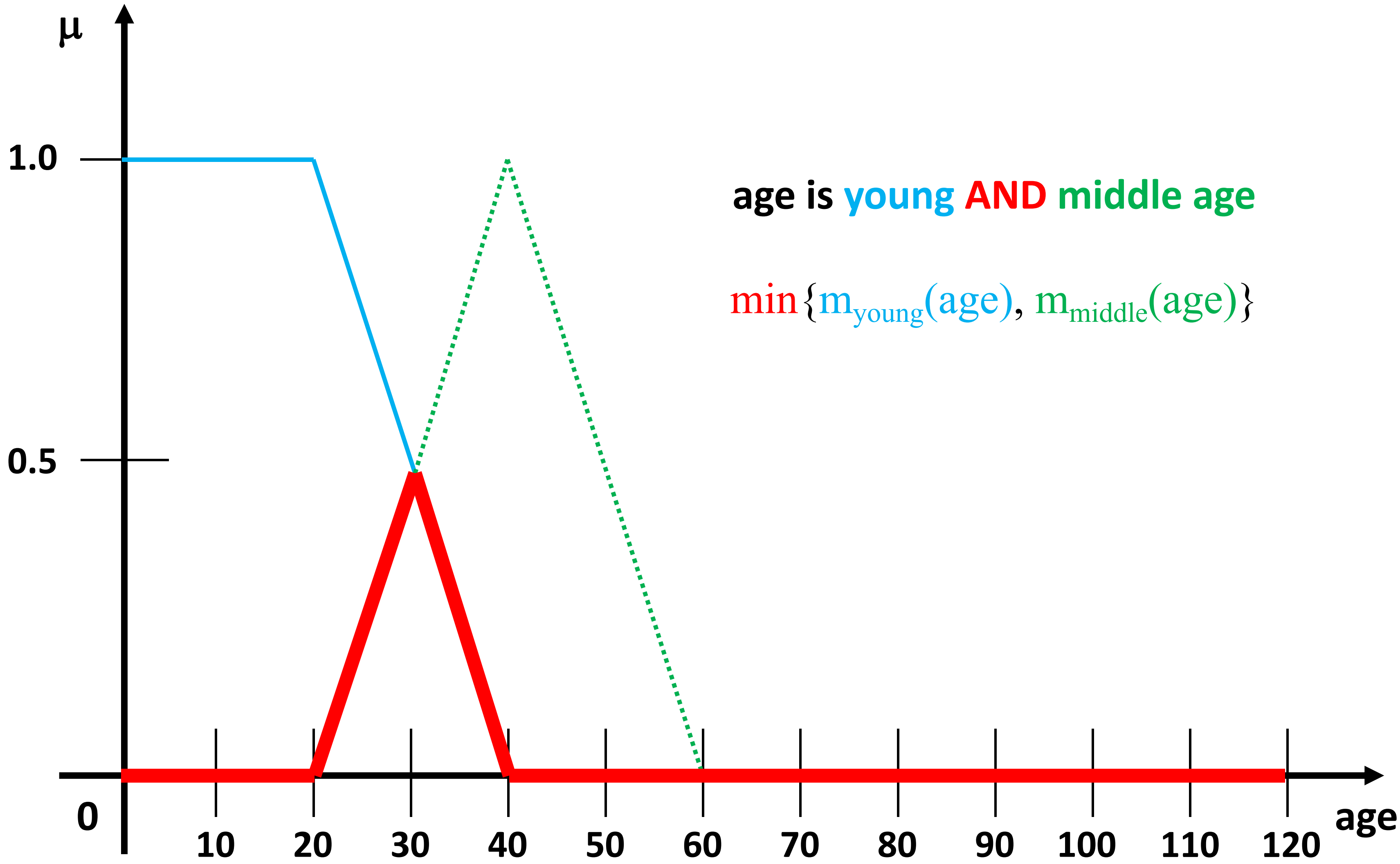


Fuzzy Logic: Logic Operators

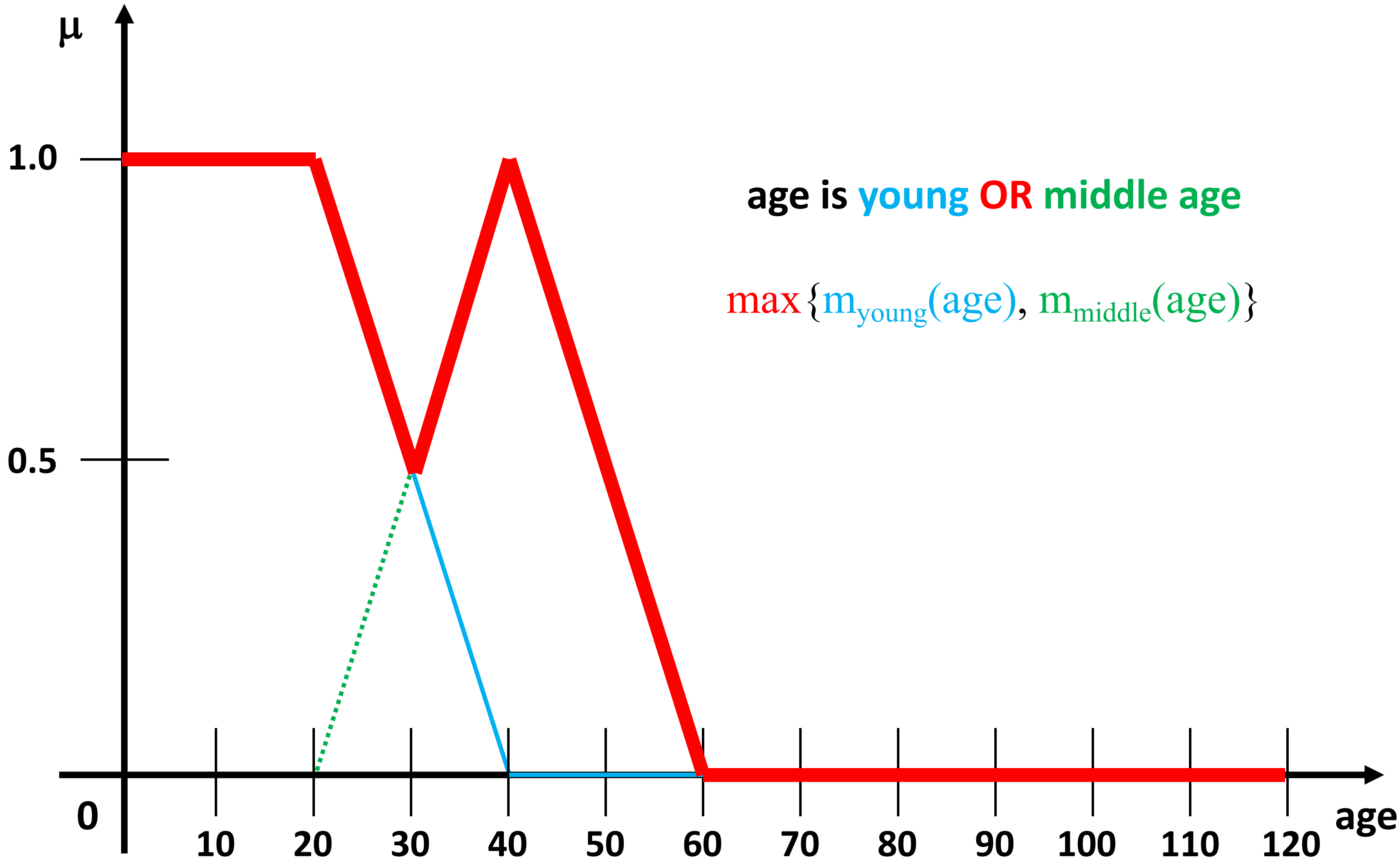


Boolean	Fuzzy
AND(x, y)	MIN(x, y)
OR(x, y)	MAX(x, y)
NOT(x)	1 - x

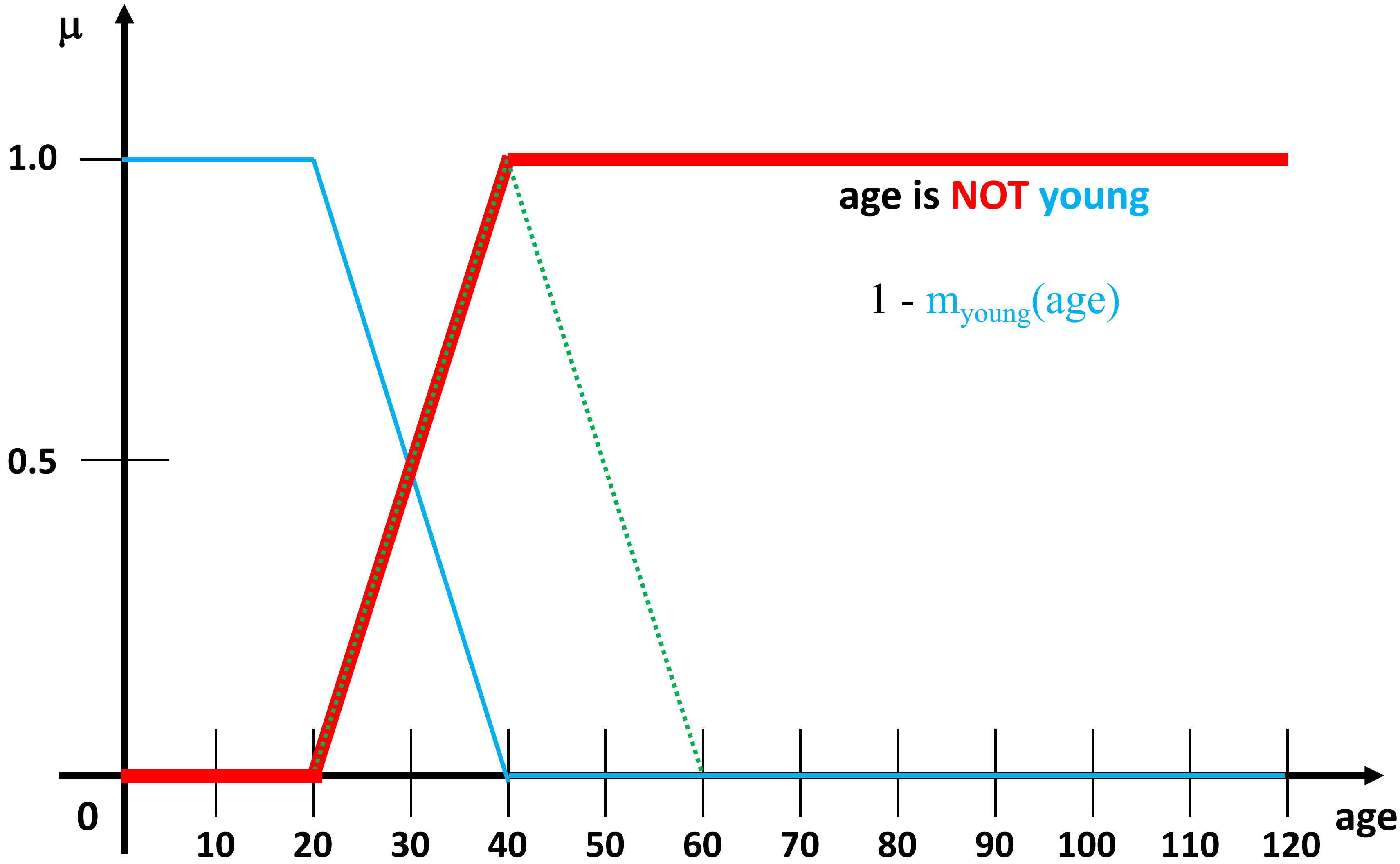
Fuzzy Logic: the AND Operator



Fuzzy Logic: the OR Operator

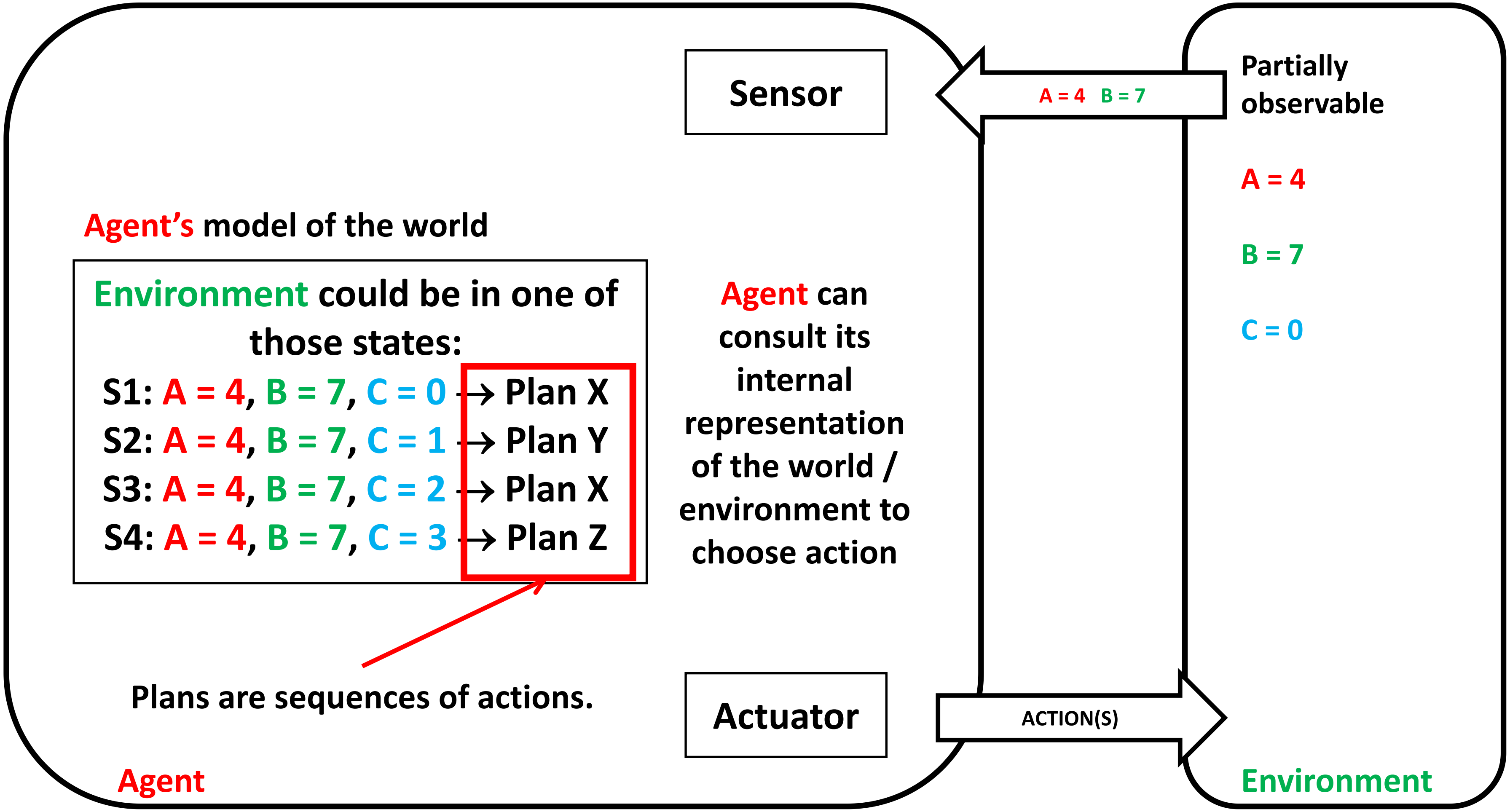


Fuzzy Logic: the NOT Operator



END OF BONUS MATERIAL

Agents and Belief State



Assume: $D_C = \{0,1,2,3\}$

Decision Theory

- Decisions: every plan (actions) leads to an outcome (state)
- Agents have preferences (preferred outcomes)
- Preferences → 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

$$MEU(M) = \frac{P(S10) * U(S10) + P(S15) * U(S15) + P(S55) * U(S55)}{3}$$

Environment could be in one of those states:

A = 4, B = 7, C = 0 → Plan X

A = 4, B = 7, C = 1 → Plan Y

A = 4, B = 7, C = 2 → Plan X

A = 4, B = 7, C = 3 → Plan Z

Action M

- Outcome S10: P(S10) = 0.1 U(S10) = 2
- Outcome S15: P(S15) = 0.2 U(S15) = 3
- Outcome S55: P(S55) = 0.5 U(S55) = 1

Pick highest MEU action

Action N

- Outcome S20: P(S20) = 0.7 U(S20) = 8
- Outcome S15: P(S15) = 0.3 U(S15) = 4
- Outcome S12: P(S12) = 0.6 U(S12) = 5

$$MEU(N) = \frac{P(S20) * U(S20) + P(S15) * U(S15) + P(S12) * U(S12)}{3}$$

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:

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

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.

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:

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

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

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

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 ticket**
- **the set of outcomes (resulting states) is a lottery**

A lottery L with possible outcomes $S_1, ..., S_n$ that occur with probabilities $p_1, ..., p_n$ 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**:**

$$(\mathrm{A} \succ \mathrm{B}) \wedge (\mathrm{B} \succ \mathrm{C}) \Rightarrow (\mathrm{A} \succ \mathrm{C})$$

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

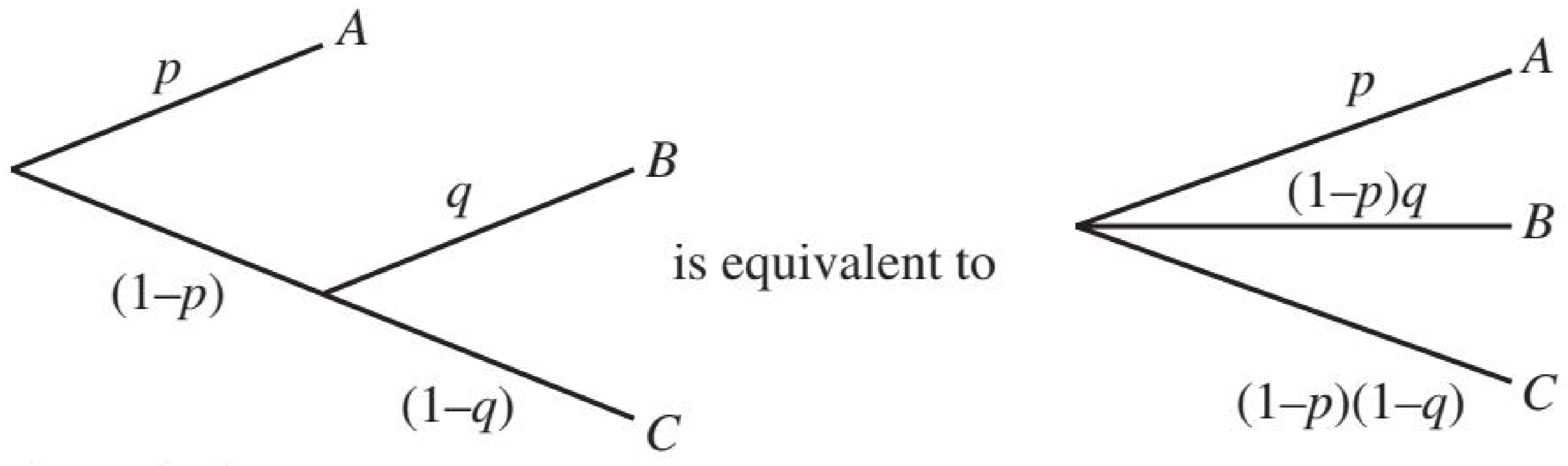
Suppose two lotteries have the same two possible outcomes, A and B. If an agent prefers A to B, then the agent must prefer the lottery that has a higher probability for A:

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

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:

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

and

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

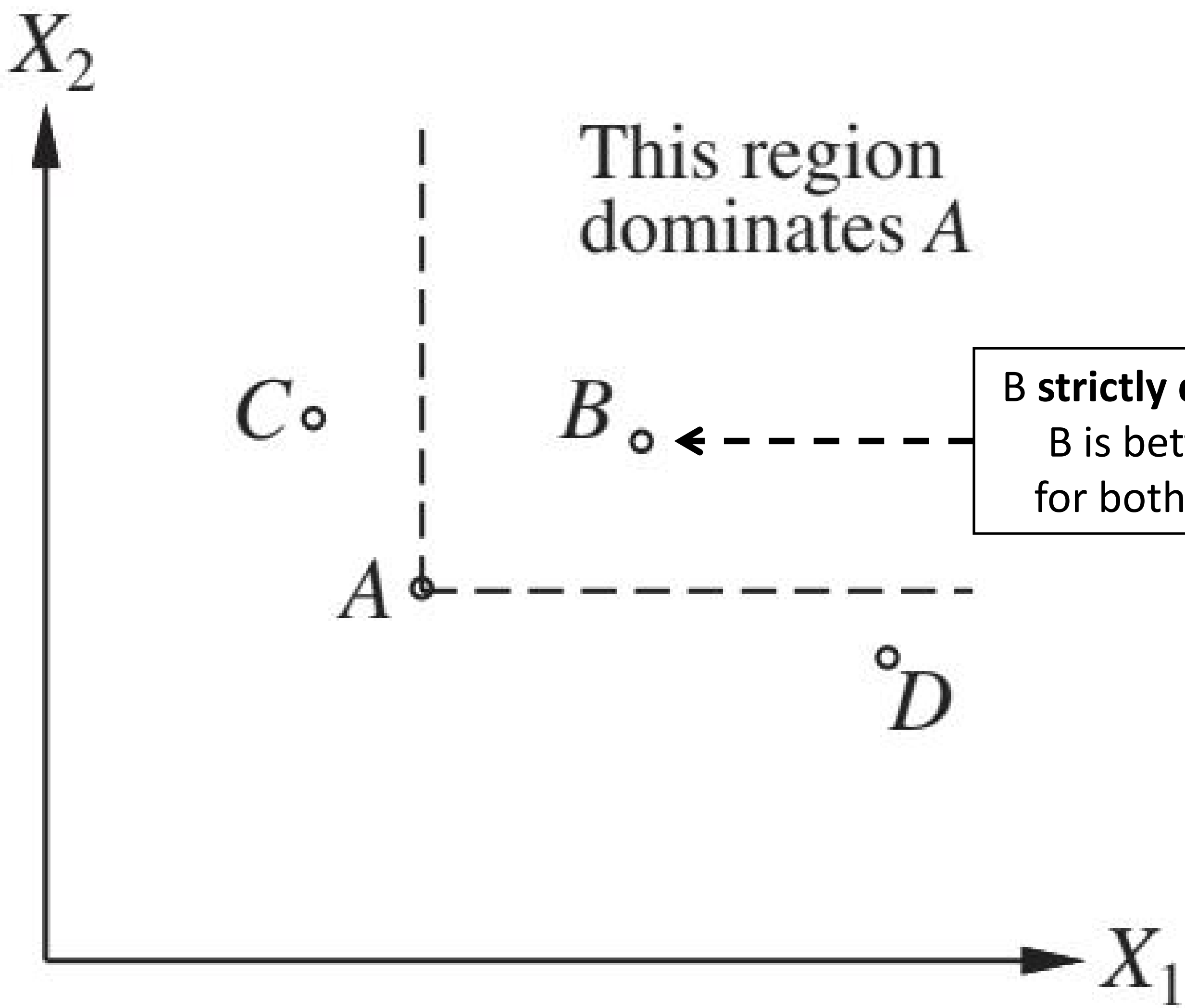
Multiattribute Outcomes

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

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

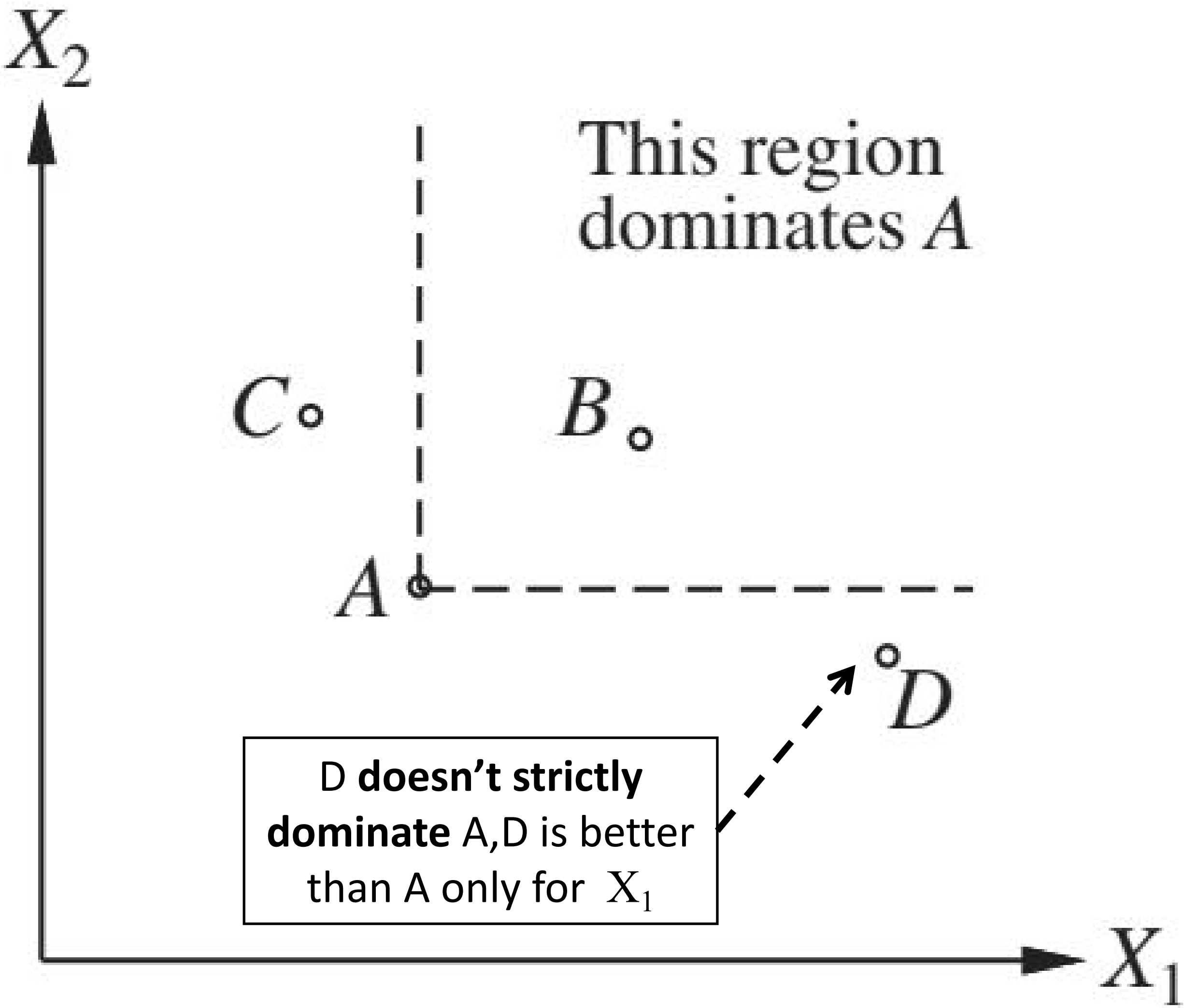
Assigned values: $\mathbf{x} = <x_1, ..., x_n>$

Strict Dominance: Deterministic



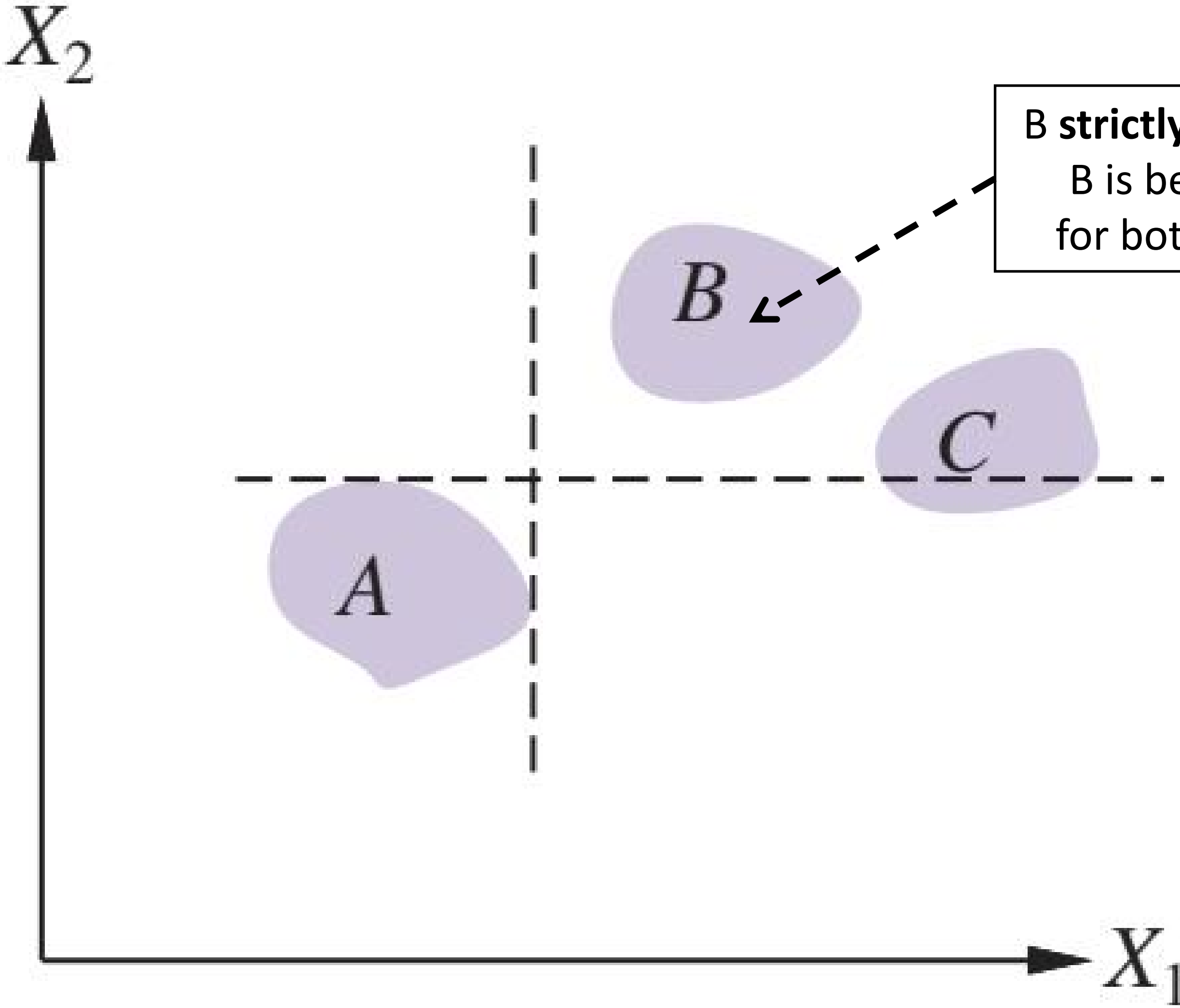
B **strictly dominates** A
B is better than A
for both X_1 and X_2

Strict Dominance: Deterministic



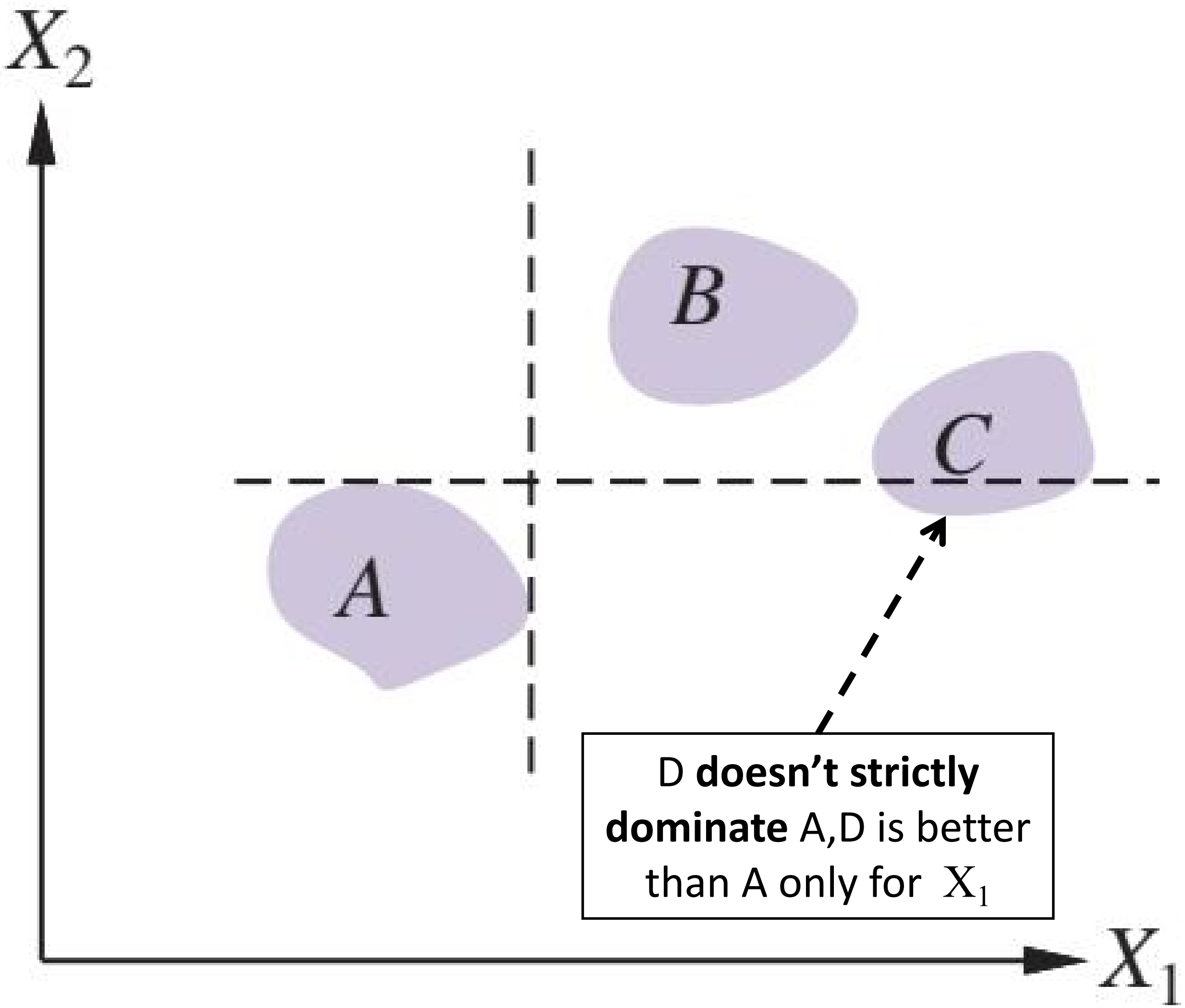
D **doesn't strictly dominate** A,D is better than A only for X_1

Strict Dominance: Uncertain



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

Strict Dominance: Uncertain



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 <u>additional nodes</u> that represent actions and utilities.

Decision Networks

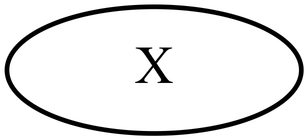
The most basic decision network needs to include:

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

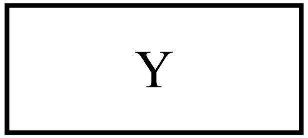
Decision Network Nodes

Decision networks are built using the following nodes:

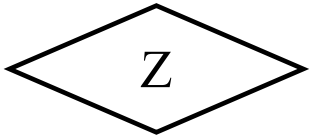
- **chance nodes:**



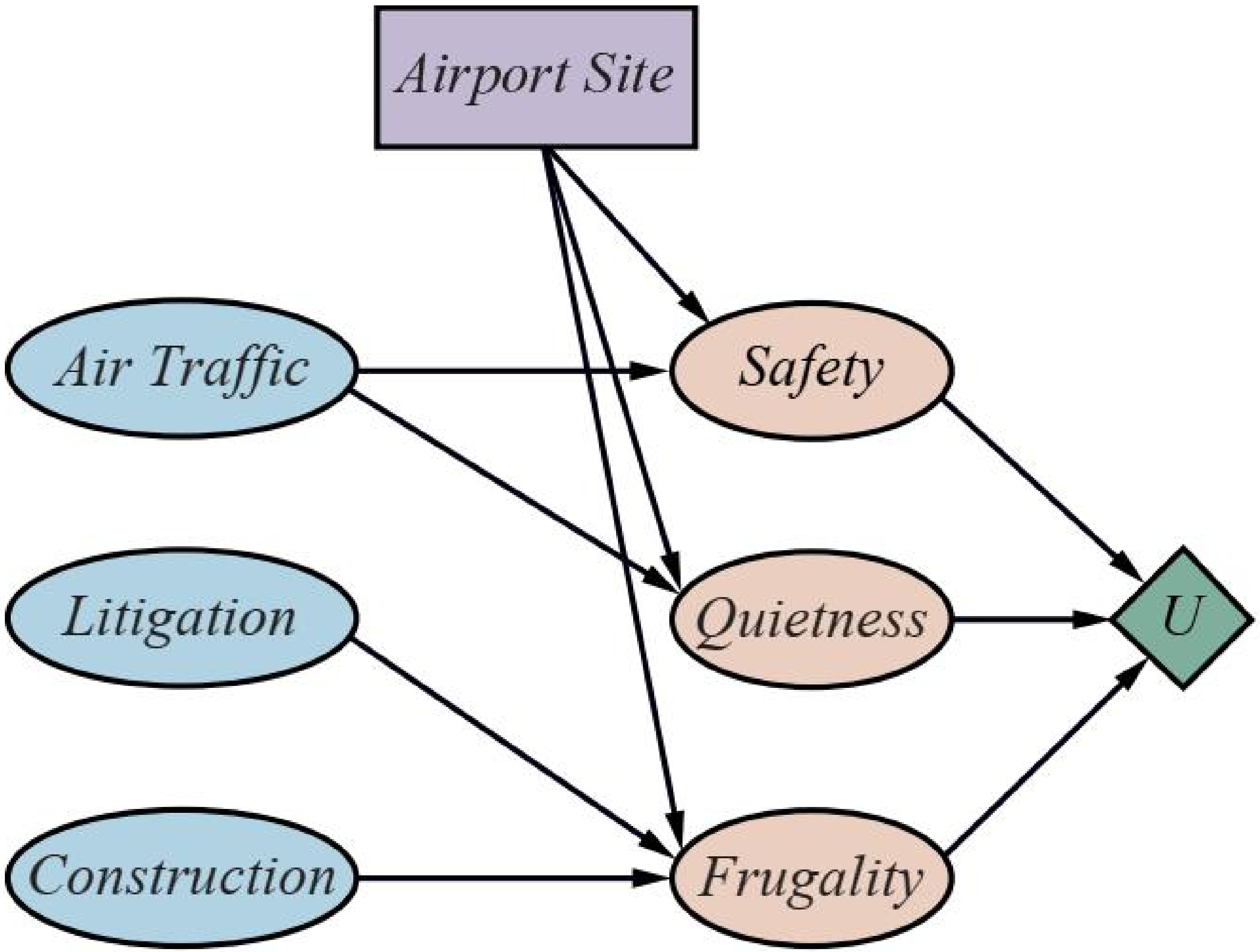
- **decision nodes:**



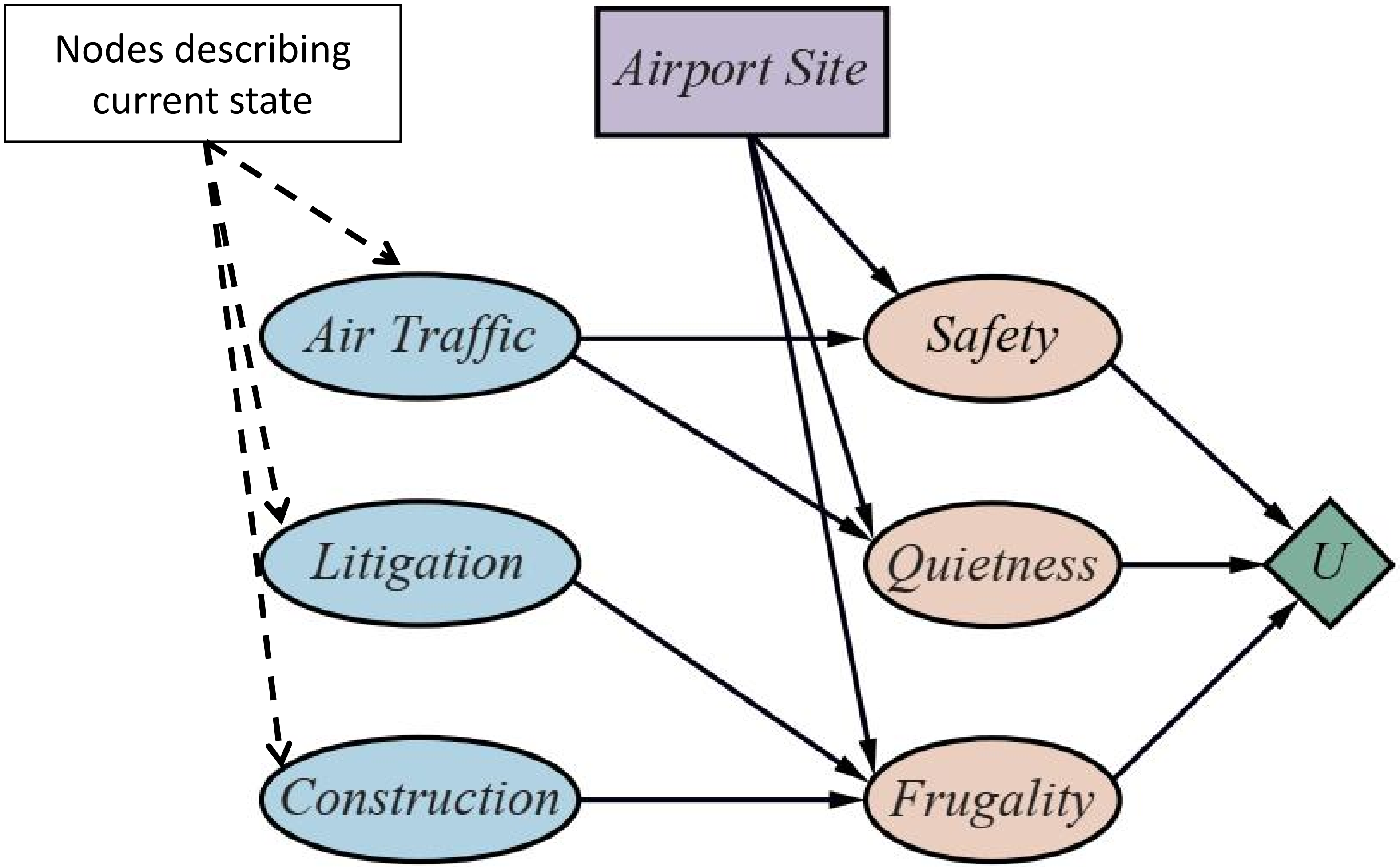
- **utility (or value) nodes**



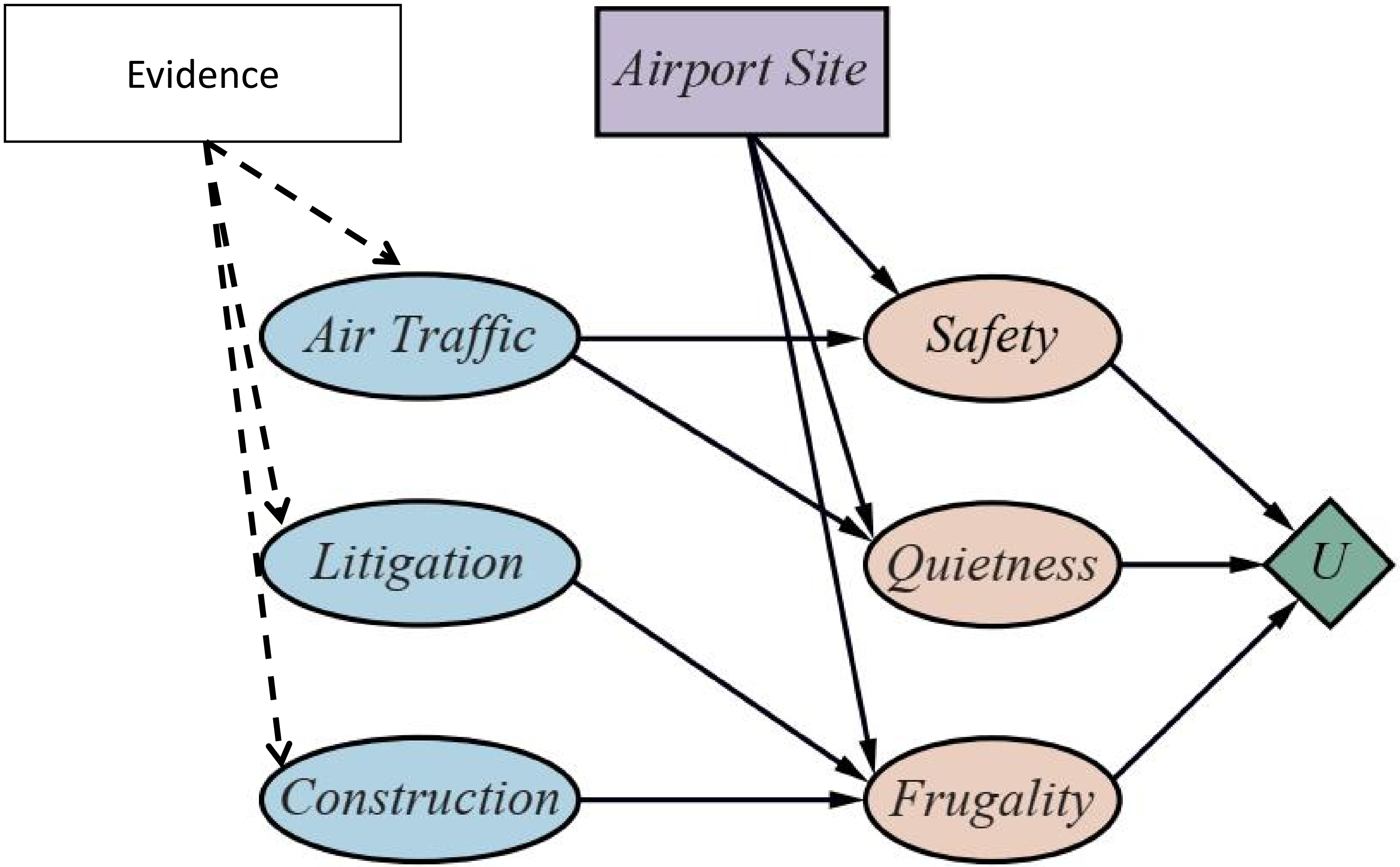
Decision Network: Example



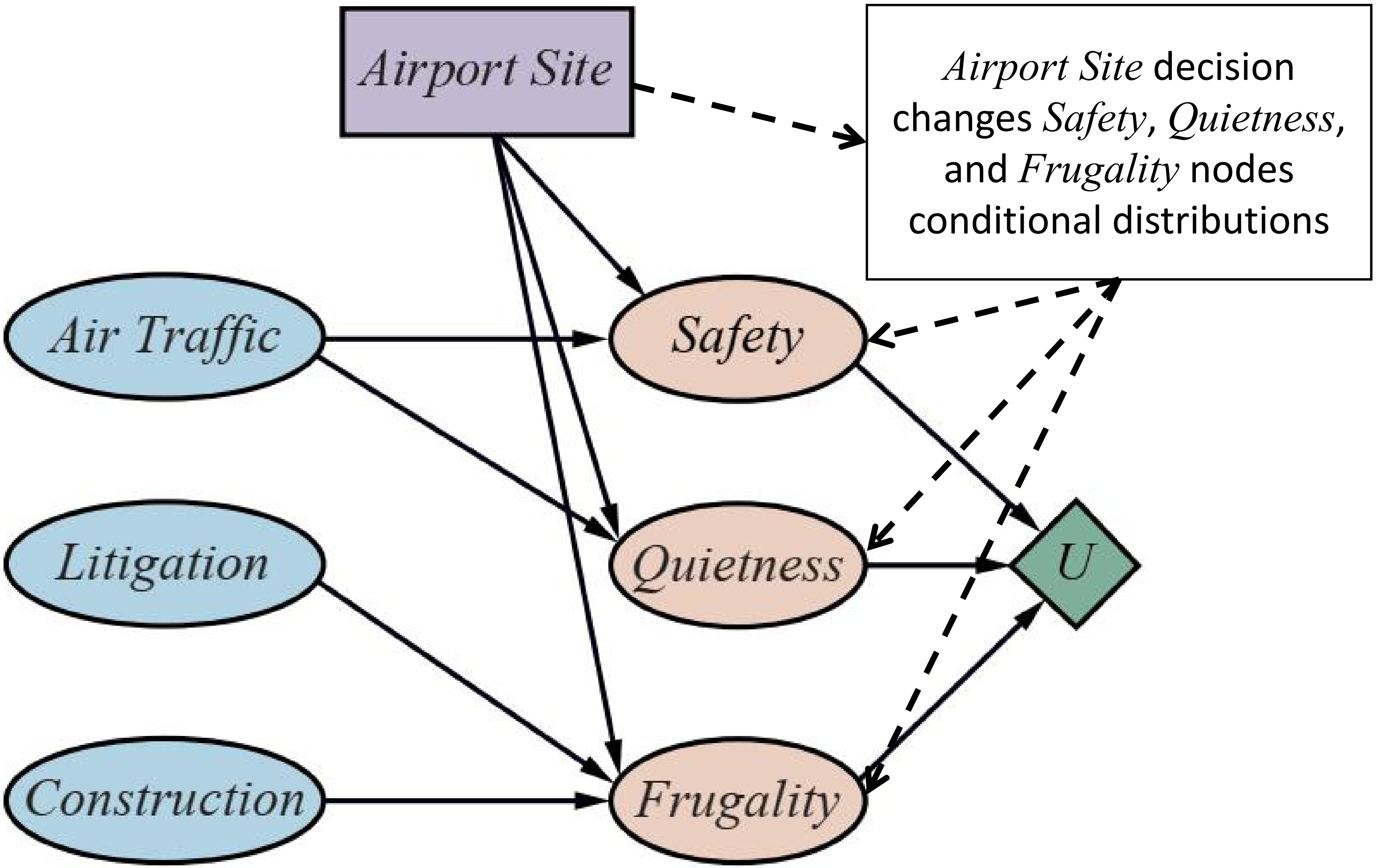
Decision Network: Example



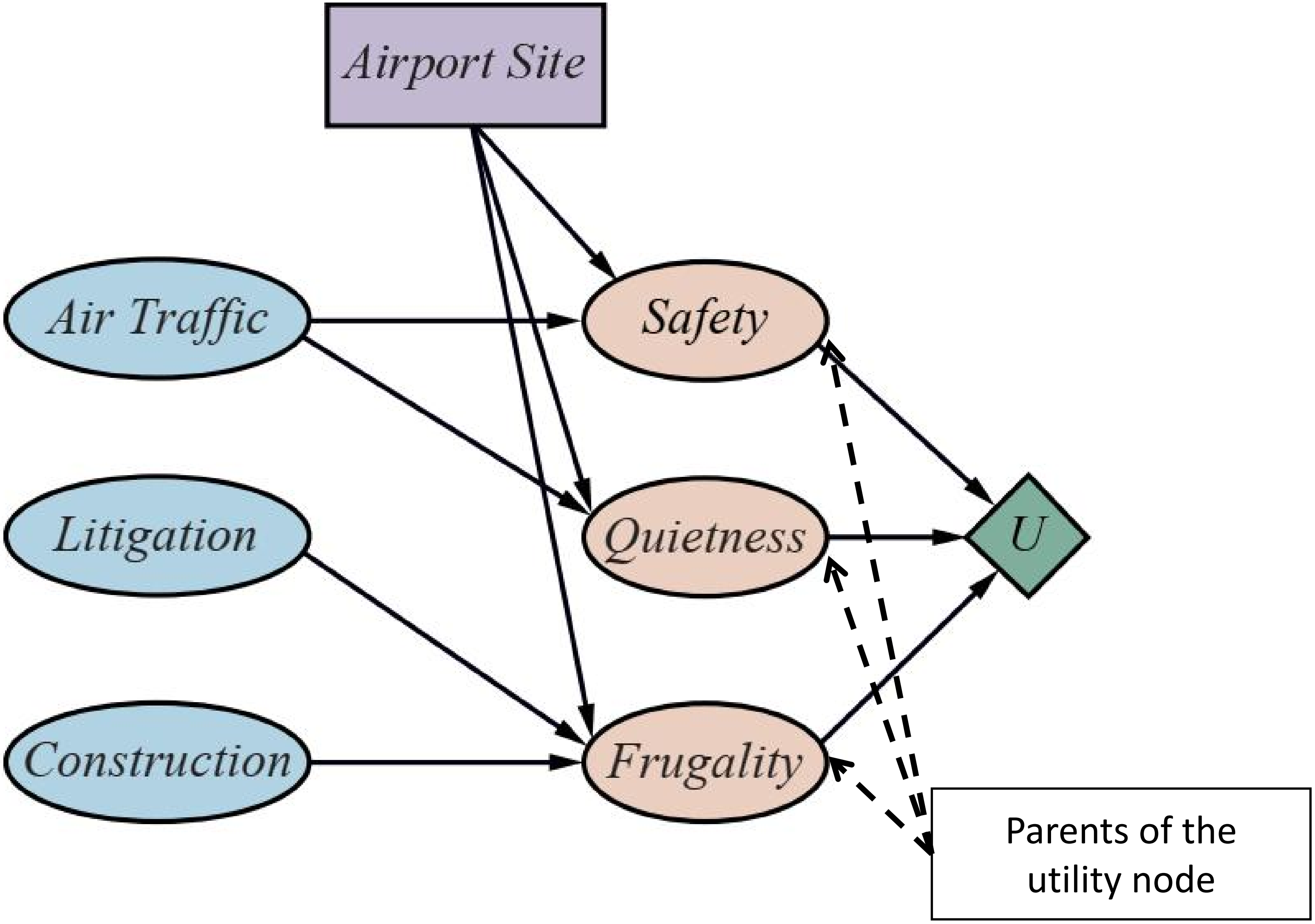
Decision Network: Example



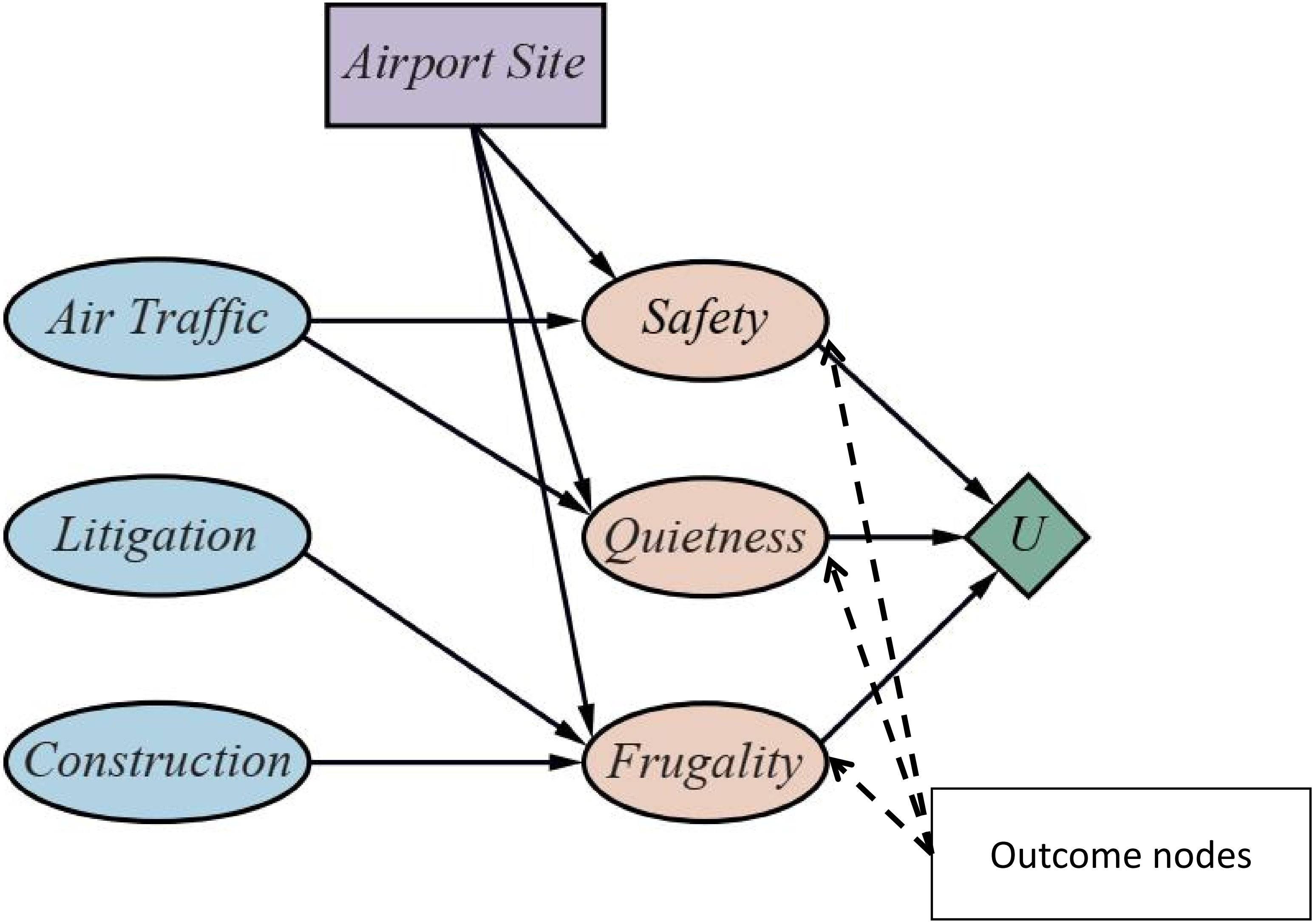
Decision Network: Example



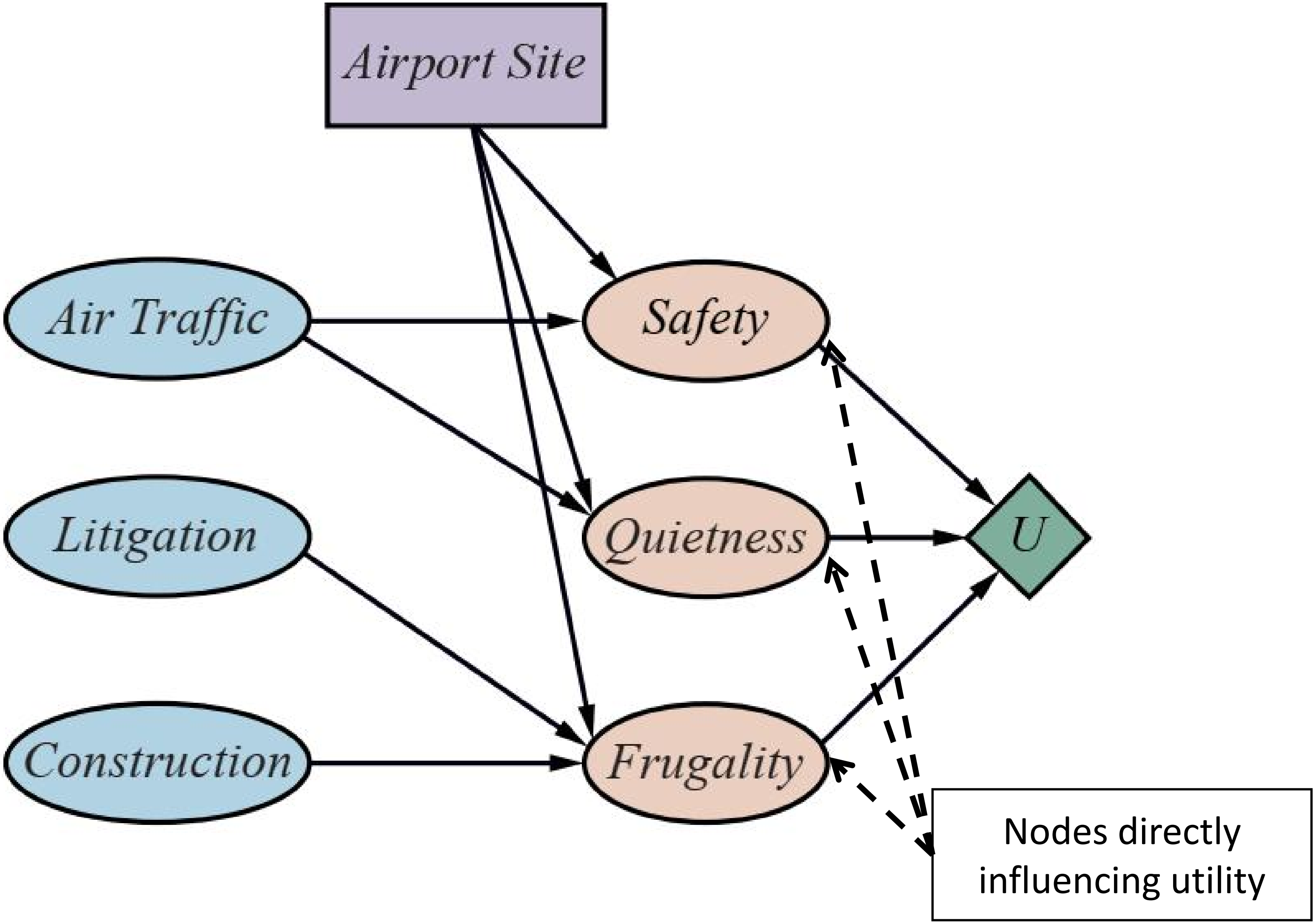
Decision Network: Example



Decision Network: Example



Decision Network: Example

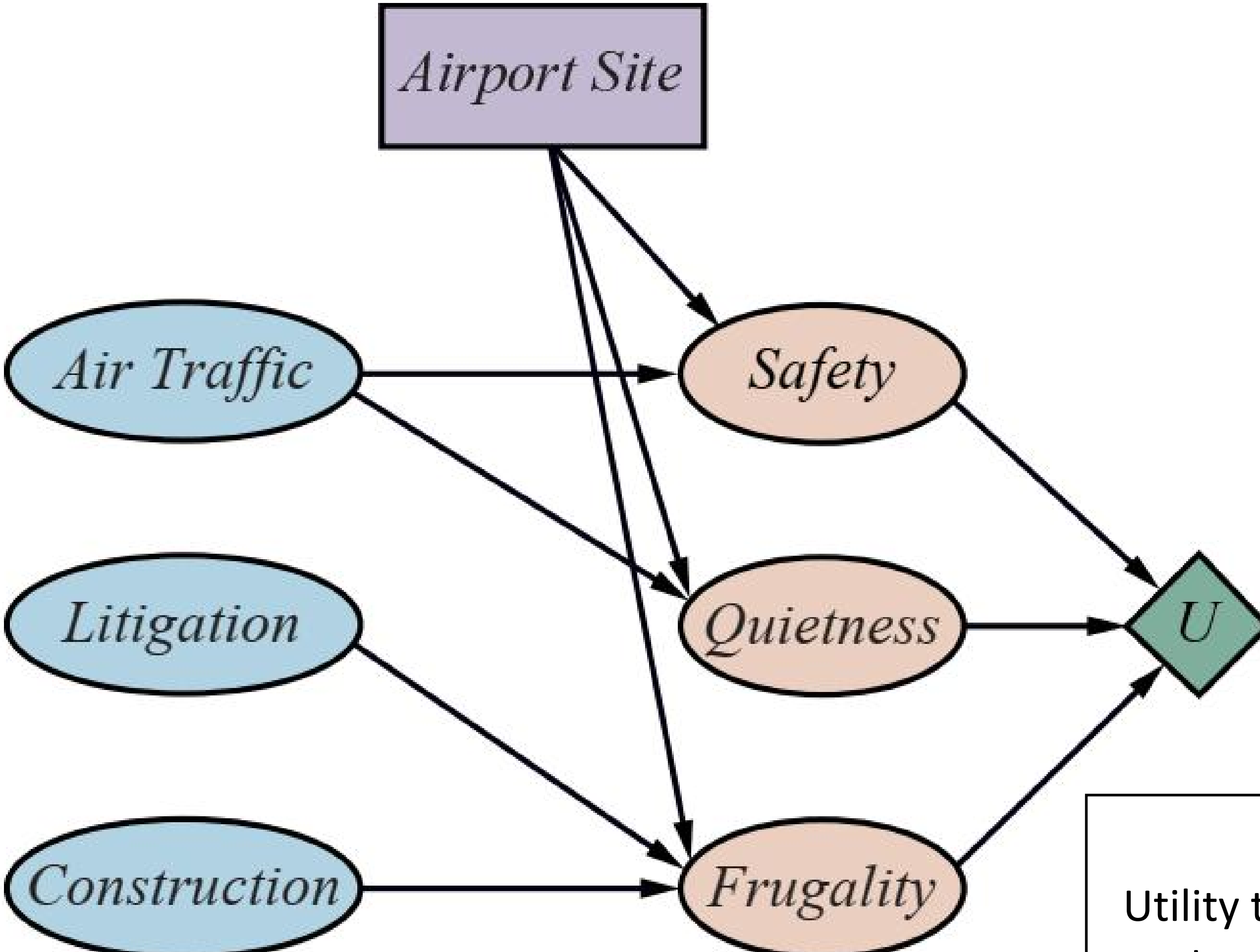


Decision Network: Evaluation

The algorithm for decision network evaluation is as follows:

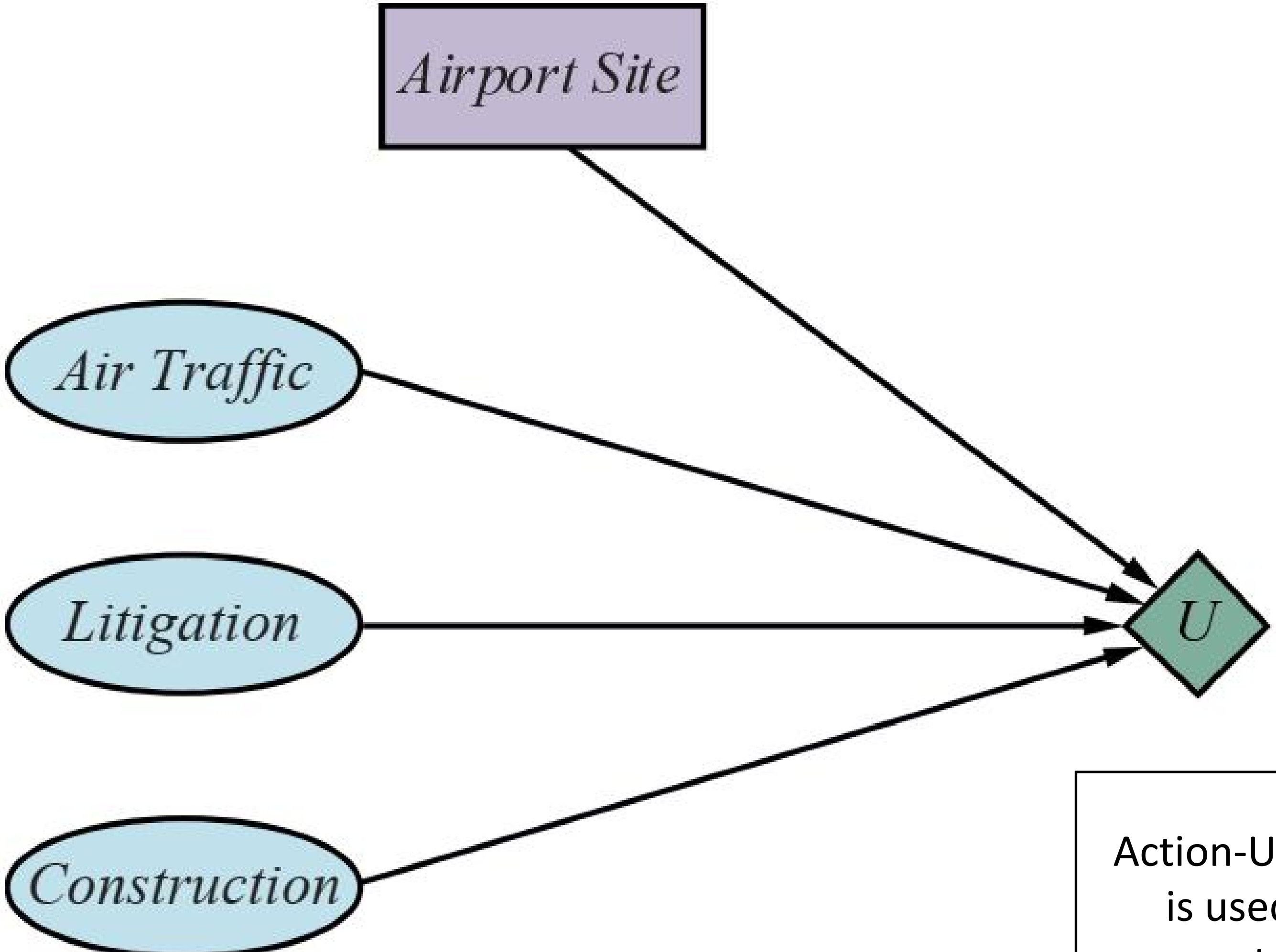
1. Set the evidence variables for the current state
2. For each possible value a of decision node:
   a. Set the decision node to that value
   b. Calculate the posterior probabilities for the parent nodes of the utility node
   c. Calculate the utility for the action / value a
3. Return the action with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