Chapter 4

Knowledge & Reasoning

A knowledge-based agent

- A knowledge-based agent includes a knowledge base and an inference system.
- A knowledge base is a set of representations of facts of the world.
- Each individual representation is called a **sentence**.
- The sentences are expressed in a **knowledge representation language**.
- **Inference** is deriving new sentences from old.
- The agent operates as follows:
  1. It TELLs the knowledge base what it perceives.
  2. It ASKs the knowledge base what action it should perform.
  3. It performs the chosen action.

Structure of knowledge based agent

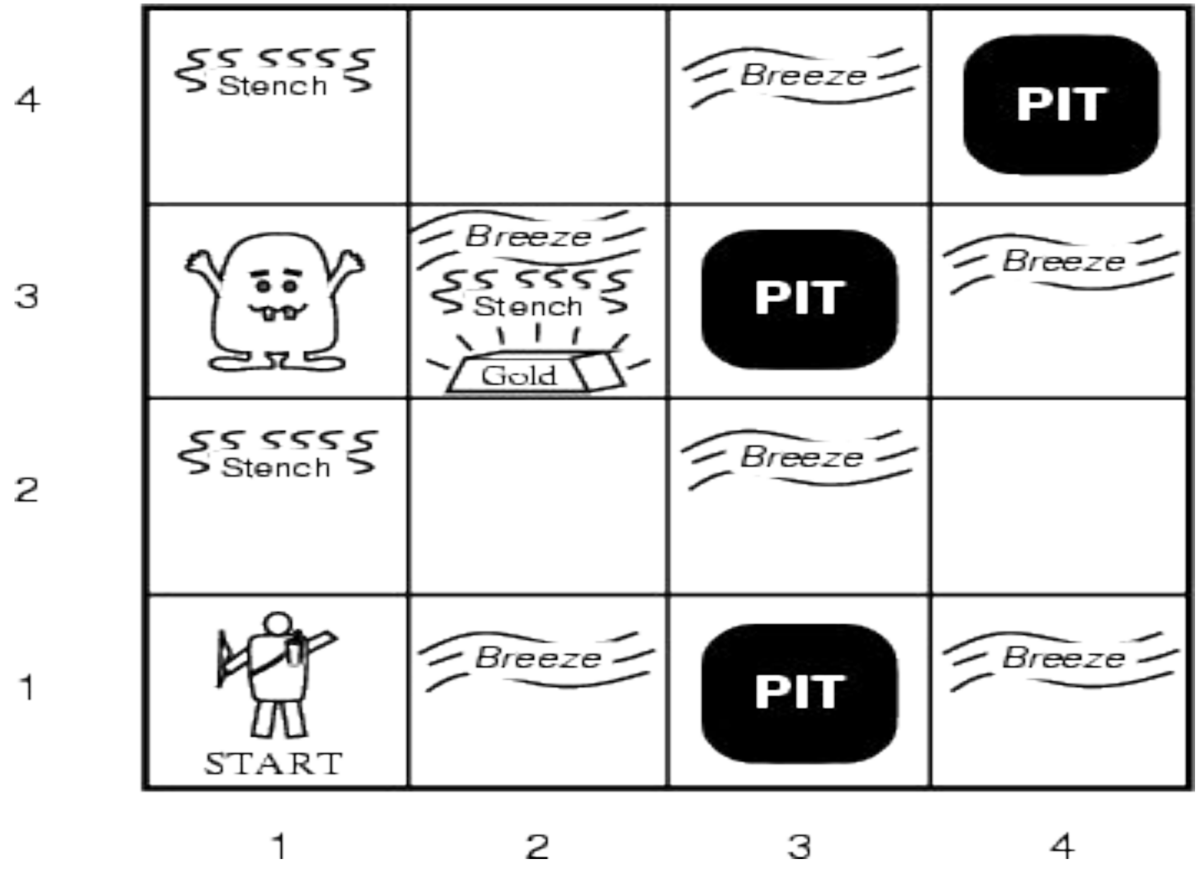
```
function KB-AGENT(percept):
persistent: KB, a knowledge base
         t, a counter, initially 0, indicating time
TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
Action = ASK(KB, MAKE-ACTION-QUERY(t))
TELL(KB, MAKE-ACTION-SENTENCE(action, t))
 t = t + 1
 return action
```

The Wumpus World environment

- The Wumpus world computer game
- The agent explores a cave consisting of rooms connected by passageways.
- Lurking somewhere in the cave is the Wumpus, a beast that eats any agent that enters its room.
- Some rooms contain bottomless pits that trap any agent that wanders into the room.
- Occasionally, there is a heap of gold in a room.
- The goal is to collect the gold and exit the world without being eaten

A typical Wumpus world

- The agent always starts in the field [1,1].
- The task of the agent is to find the gold, return to the field [1,1] and climb out of the cave.



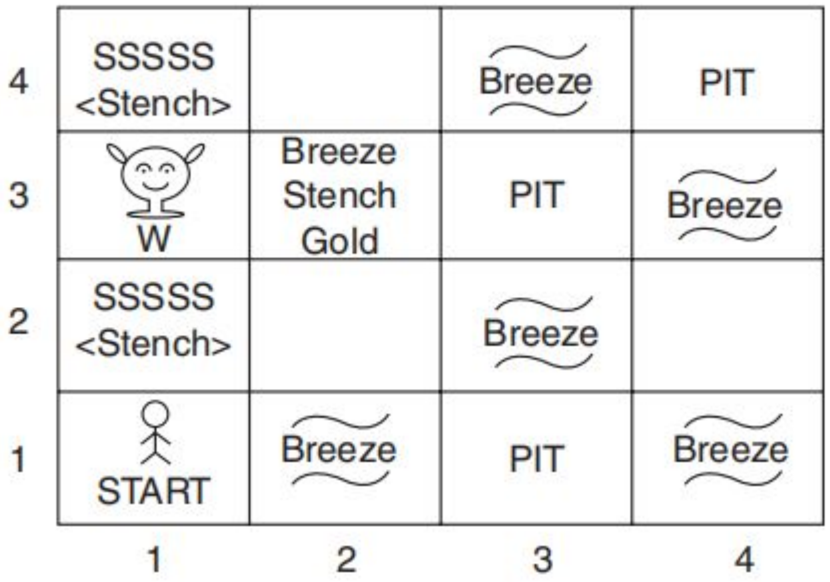
Wumpus goal

The agent's goal is to find the gold and bring it back to the start square as quickly as possible, without getting killed

- 1000 points reward for climbing out of the cave with the gold
- 1 point deducted for every action taken
- 1000 points penalty for getting killed

Specifying the environment

Like the vacuum world, the Wumpus world is a grid of squares surrounded by walls, where each square can contain agents and objects. The agent always starts in the lower left corner, a square that we will label [1, 1]. The agent's task is to find the gold, return to [1, 1], and climb out of the cave.



To specify the agent's task, we specify its performance, environment, actions, sensors [PEAS]. In the Wumpus world, these are as follows:

Specifying the environment (PEAS descriptor)

Performance measure:

- 1,000 for picking up the gold.
- -1000 for falling into a pit or being eaten by Wumpus.
- One for each action taken.
- -10 for using up the arrow.
- The game ends if either agent dies or came out of the cave.

Environment:

- A 4 × 4 grid of rooms.
- The agent always starts in the square labeled [1, 1], facing to the right.
- The locations of the gold and Wumpus are chosen randomly, with a uniform distribution, from the squares other than the start square.
- Each square other than the start can be a pit, with probability 0.2.

Specifying the environment (PEAS descriptor)

Actuators:

- Left turn,
- Right turn
- Move forward
- Grab
- Release
- Shoot.

Specifying the environment (PEAS descriptor)

Sensors:

- The agent has five sensors, each of which gives a single bit of information.
- In the square containing the Wumpus and in the directly (not diagonally) adjacent squares, the agent will perceive a **stench.**
- In the squares directly adjacent to pit, the agent will perceive a **breeze**.
- In the square where the gold is, the agent will perceive a **glitter**.
- When an agent walks into a wall, it will perceive a **bump.**
- When the Wumpus is killed, it emits a woeful **scream** that can be perceived anywhere in the cave.
- The percepts will be given to the agent in the form of list of five symbols; for example, if there is a stench and a breeze, but no glitter, bump, or scream, the agent will receive the percept

Properties of the task environment

- **Partially observable:** The Wumpus world is partially observable because the agent can only perceive the close environment such as an adjacent room.
- **Deterministic:** It is deterministic, as the result and outcome of the world are already known.
- **Sequential:** The order is important, so it is sequential.
- **Static:** It is static as Wumpus and Pits are not moving.
- **Discrete:** The environment is discrete.
- **Single agent:** The environment is a single agent as we have one agent only and Wumpus is not considered as an agent.

- Atomic proposition variable for Wumpus world:
- Let $\mathbf{P}_{\mathbf{i,j}}$ be true if there is a Pit in the room [i, j].
- Let $\mathbf{B}_{\mathbf{i,j}}$ be true if agent perceives breeze in [i, j], (dead or alive).
- Let $\mathbf{W}_{\mathbf{i,j}}$ be true if there is wumpus in the square[i, j].
- Let $\mathbf{S}_{\mathbf{i,j}}$ be true if agent perceives stench in the square [i, j].
- Let $\mathbf{V}_{\mathbf{i,j}}$ be true if that square[i, j] is visited.
- Let $\mathbf{G}_{\mathbf{i,j}}$ be true if there is gold (and glitter) in the square [i, j].
- Let $\mathbf{OK}_{\mathbf{i,j}}$ be true if the room is safe.

Wumpus world sentences

Let $P_{i,j}$ be true if there is a pit in [i, j]; $P_{i,j}$

Let $B_{i,j}$ be true if there is a breeze in [i, j]; $B_{i,j}$

$\neg P_{1,1}$, $\neg B_{1,1}$, $B_{2,1}$

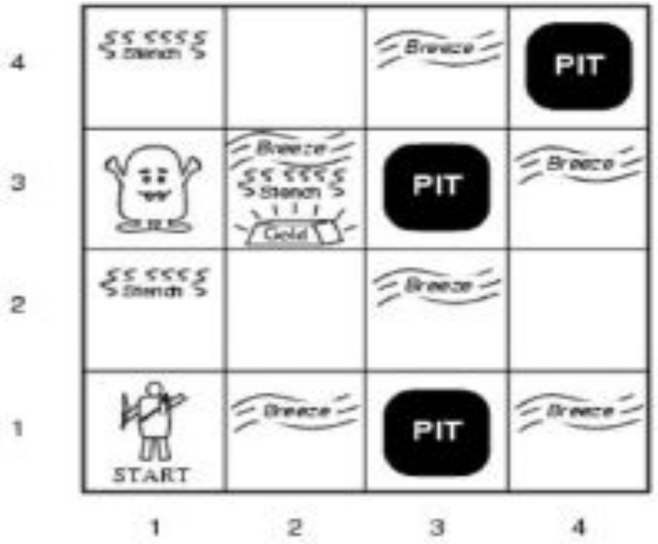
- "Pits cause breezes in adjacent squares"

$B_{i,\ j} \Leftrightarrow (P_{i-1,\ j} \vee P_{i,\ j-1} \vee P_{i,\ j+1} \vee P_{j+1,\ j})$

$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

$B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$

- "Wumpus causes stench in adjacent squares" (???)



Wumpus world sentences

- "No Stench means there is no wumpus in adjacent Square"

$$\neg S_{i,\ j} \Rightarrow \neg(W_{i-1,\ j} \wedge W_{i,\ j-1} \wedge W_{i,\ j+1} \wedge W_{j+1,\ j})$$

$$\neg S_{2,2} \Rightarrow \neg(W_{1,2} \wedge W_{2,1} \wedge W_{2,3} \wedge W_{3,2})$$

- "No Breeze means there is no pits in adjacent Square"

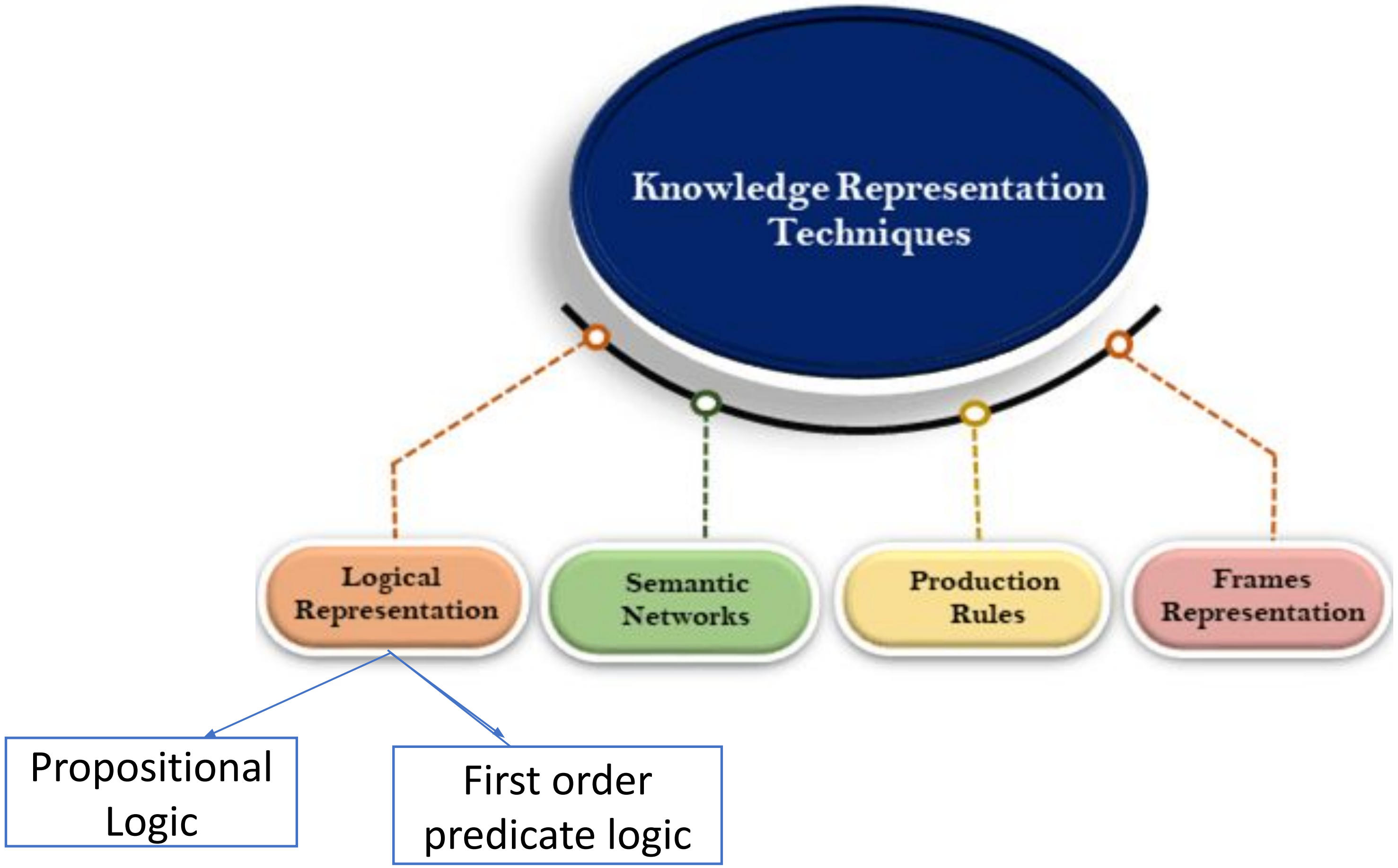
$$\neg B_{i,\ j} \Rightarrow \neg(P_{i-1,\ j} \wedge P_{i,\ j-1} \wedge P_{i,\ j+1} \wedge P_{j+1,\ j})$$

$$\neg B_{2,2} \Rightarrow \neg(P_{1,2} \wedge P_{2,1} \wedge P_{2,3} \wedge P_{3,2})$$

Fundamental property of logical reasoning-

In each case for which, the agent draws conclusion based on the available information, that conclusion is guaranteed to be correct if the available information is correct.

Knowledge representation Techniques



What is a Logic?

- A language with concrete rules
  - No ambiguity in representation
  - Allows unambiguous communication and processing
  - Very unlike natural languages e.g. English
- Many ways to translate between languages
  - A statement can be represented in different logics
  - And perhaps differently in same logic
- **Expressiveness** of a logic
  - How much can we say in this language?

Logic

- We used logical reasoning to find the gold.
- Logics are formal languages for representing information such that conclusions can be drawn
- Syntax defines the sentences in the language
- Semantics define the "meaning" of sentences;
  - i.e., define truth of a sentence in a world

- E.g., the language of arithmetic
  - $x+2 \geq y$ is a sentence; $x2+y >$ {} is not a sentence → syntax
  -
  - $x+2 \geq y$ is true in a world where $x = 7$, $y = 1$
  - $x+2 \geq y$ is false in a world where $x = 0$, $y = 6$

semantics

Logical Entailment

- *Model:* an assignment of (*True*/*False*) values to each of the symbols. If the knowledge base is built from *n* symbols, there are 2^n possible models.
- *Evaluation:* A sentence *s* is evaluated on a model *m* by setting each symbol to its corresponding value in *m*. The result of the evaluation is a value in {*True*,*False*}

- "KB logically entails *S*" if all the models that evaluate KB to *True* also evaluate *S* to *True*.
- Denoted by: KB |= *S*
- Note: We do not care about those models that evaluate KB to *False*. The result of evaluating *S* for these models is irrelevant.

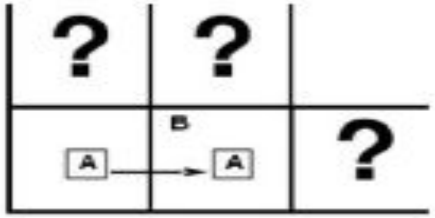
- *Valid:* A sentence is *valid* if it is true for all models.

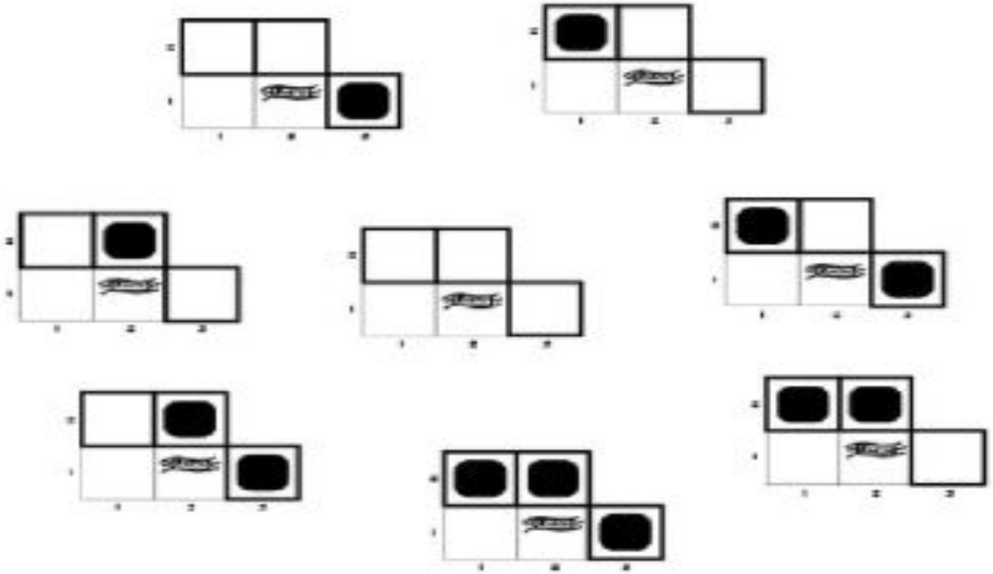
$$\boxed{\alpha \vee \neg\alpha}$$

- *Satisfiable:* A sentence is *satisfiable* if it is true for some models.
- *Unsatisfiable:* A sentence is *unsatisfiable* if it is true for no models.

$$\boxed{\alpha \wedge \neg\alpha}$$

Wumpus models





All possible models in this reduced Wumpus world.

Propositional Logic

- Propositional logic is also called **Boolean logic** as it works on 0 and 1.
- In propositional logic, we use **symbolic variables** to represent the logic, and we can use any symbol for a representing a proposition, such A, B, C, P, Q, R, etc.
- Propositions can be either true or false, but it cannot be both.
- Propositional logic consists of an object, relations or function, and **logical connectives**.
- These connectives are also called logical operators.
- The propositions and connectives are the basic elements of the propositional logic.
- Connectives can be said as a logical operator which connects two sentences.
- A proposition formula which is always true is called **tautology**, and it is also called a valid sentence.
- A proposition formula which is always false is called **Contradiction**.
- Statements which are questions, commands, or opinions are not propositions such as "**Where is Rohini**", "**How are you**", "**What is your name**", are not propositions.

Logical Connectives

- **Negation:** A sentence such as ¬ P is called negation of P. A literal can be either Positive literal or negative literal.
- **Conjunction:** A sentence which has $\wedge$ connective such as, **P $\wedge$ Q** is called a conjunction.
  Example: Rohan is intelligent and hardworking. It can be written as,
  P= Rohan is intelligent,
  Q= Rohan is hardworking. → P$\wedge$ Q.
- **Disjunction:** A sentence which has $\vee$ connective, such as **P $\vee$ Q**. is called disjunction, where P and Q are the propositions.
  Example: "Richa is a doctor or Engineer",
-

  Here P= Ritika is Doctor. Q= Ritika is Doctor, so we can write it as **P $\vee$ Q**.
- **Implication:** A sentence such as P → Q, is called an implication. Implications are also known as if-then rules. It can be represented as
  If it is raining, then the street is wet.
  Let P= It is raining, and Q= Street is wet, so it is represented as P → Q
- **Biconditional:** A sentence such as **P⇔ Q is a Biconditional sentence, example If I am breathing, then I am alive**
  P= I am breathing, Q= I am alive, it can be represented as P ⇔ Q.

Propositional logic: Semantics

Each model/world specifies true or false for each proposition symbol

E.g. $P_{1,2}$ $P_{2,2}$ $P_{3,1}$

	$P_{1,2}$	$P_{2,2}$	$P_{3,1}$
E.g.	false	true	false

With these symbols, 8 possible models, can be enumerated automatically.

Rules for evaluating truth with respect to a model *m*:

$\neg S$	is true iff	S is false	
$S_1 \wedge S_2$	is true iff	S_1 is true and	S_2 is true
$S_1 \vee S_2$	is true iff	S_1 is true or	S_2 is true
$S_1 \Rightarrow S_2$	is true iff	S_1 is false or	S_2 is true
i.e.,	is false iff	S_1 is true and	S_2 is false
$S_1 \Leftrightarrow S_2$	is true iff	$S_1 \Rightarrow S_2$ is true and	$S_2 \Rightarrow S_1$ is true

First Order Logic

- More expressive logic than propositional
- **Constants** are objects: john, apples
- **Predicates** are properties and relations:
  - likes(john, apples)
- **Functions** transform objects:
  - likes(john, fruit_of(apple_tree))
- **Variables** represent any object: likes(X, apples)
- **Quantifiers** qualify values of variables
  - True for all objects (Universal): $\forall X.$ likes(X, apples)
  - Exists at least one object (Existential): $\exists X.$ likes(X, apples)

Translating English to FOL

- Every gardener likes the sun.
  - ∀x gardener(x) => likes(x,Sun)
- All purple mushrooms are poisonous.
  - ∀x (mushroom(x) ^ purple(x)) => poisonous(x)
- No purple mushroom is poisonous.
  - ~∃x purple(x) ^ mushroom(x) ^ poisonous(x)
  - ∀x (mushroom(x) ^ purple(x)) => ~poisonous(x)

Propositional Logic	First Order Predicate Logic
Cannot represent small worlds	Very well represent small world
Weak knowledge representation language	Strong knowledge representation language
Uses symbols to represent propositions	Uses predicates which involves constants, variables, functions, relations
Cannot express specialization, generalization.	It can express specialization, Generalization, using relations
Foundational level logic	Higher level logic
Not sufficiently expressive to represent complex statement	sufficiently expressive to represent complex statement
World contain facts	World contain objects, relations, functions

a. Some students took French in spring 2001.
$\exists x \;\; Student(x) \wedge Takes(x, F, Spring2001)$.

b. Every student who takes French passes it.
$\forall x, s \;\; Student(x) \wedge Takes(x, F, s) \;\Rightarrow\; Passes(x, F, s)$.

c. Only one student took Greek in spring 2001.
$\exists x \;\; Student(x) \wedge Takes(x, G, Spring2001) \wedge \forall y \;\; y \neq x \Rightarrow \neg Takes(y, G, Spring2001)$.

d. The best score in Greek is always higher than the best score in French.
$\forall s \;\; \exists x \;\; \forall y \;\; Score(x, G, s) > Score(y, F, s)$.

e. Every person who buys a policy is smart.
$\forall x \;\; Person(x) \wedge (\exists y, z \;\; Policy(y) \wedge Buys(x, y, z)) \;\Rightarrow\; Smart(x)$.

f. No person buys an expensive policy.
$\forall x, y, z \;\; Person(x) \wedge Policy(y) \wedge Expensive(y) \;\Rightarrow\; \neg Buys(x, y, z)$.

g. There is an agent who sells policies only to people who are not insured.
$\exists x \;\; Agent(x) \wedge \forall y, z \;\; Policy(y) \wedge Sells(x, y, z) \;\Rightarrow\; (Person(z) \wedge \neg Insured(z))$.

Logical equivalence

- To manipulate logical sentences we need some rewrite rules.
- Two sentences are logically equivalent iff they are true in same models: $\alpha \equiv \beta$ iff $\alpha \models \beta$ and $\beta \models \alpha$

$(\alpha \land \beta)$	$\equiv$	$(\beta \land \alpha)$	commutativity of $\land$
$(\alpha \lor \beta)$	$\equiv$	$(\beta \lor \alpha)$	commutativity of $\lor$
$((\alpha \land \beta) \land \gamma)$	$\equiv$	$(\alpha \land (\beta \land \gamma))$	associativity of $\land$
$((\alpha \lor \beta) \lor \gamma)$	$\equiv$	$(\alpha \lor (\beta \lor \gamma))$	associativity of $\lor$
$\neg(\neg\alpha)$	$\equiv$	α	double-negation elimination
$(\alpha \Rightarrow \beta)$	$\equiv$	$(\neg\beta \Rightarrow \neg\alpha)$	contraposition
$(\alpha \Rightarrow \beta)$	$\equiv$	$(\neg\alpha \lor \beta)$	implication elimination
$(\alpha \Leftrightarrow \beta)$	$\equiv$	$((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha))$	biconditional elimination
$\neg(\alpha \land \beta)$	$\equiv$	$(\neg\alpha \lor \neg\beta)$	de Morgan
$\neg(\alpha \lor \beta)$	$\equiv$	$(\neg\alpha \land \neg\beta)$	de Morgan
$(\alpha \land (\beta \lor \gamma))$	$\equiv$	$((\alpha \land \beta) \lor (\alpha \land \gamma))$	distributivity of $\land$ over $\lor$
$(\alpha \lor (\beta \land \gamma))$	$\equiv$	$((\alpha \lor \beta) \land (\alpha \lor \gamma))$	distributivity of $\lor$ over $\land$

Inference Rules

1.Modus Ponens

$$\frac{\alpha \Rightarrow \beta, \quad \alpha}{\beta}.$$

The notation means that, whenever any sentences of the form $\alpha \Rightarrow \beta$ and α are given, then the sentence β can be inferred. For example, if $(WumpusAhead \wedge WumpusAlive) \Rightarrow Shoot$ and $(WumpusAhead \wedge WumpusAlive)$ are given, then $Shoot$ can be inferred.

2.And Elimination

$$\frac{\alpha \wedge \beta}{\alpha}.$$

For example, from $(WumpusAhead \wedge WumpusAlive)$, $WumpusAlive$ can be inferred.

3.Biconditional Elimination

$$\frac{\alpha \Leftrightarrow \beta}{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)} \quad \text{and} \quad \frac{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)}{\alpha \Leftrightarrow \beta}$$

Simple Knowledge base for wumpus world

- There is no pit in [1,1]:

$$R_1 : \quad \neg P_{1,1} .$$

- A square is breezy if and only if there is a pit in a neighboring square. This has to be stated for each square; for now, we include just the relevant squares:

$$R_2 : \quad B_{1,1} \quad \Leftrightarrow \quad (P_{1,2} \vee P_{2,1}) .$$
$$R_3 : \quad B_{2,1} \quad \Leftrightarrow \quad (P_{1,1} \vee P_{2,2} \vee P_{3,1}) .$$

- The preceding sentences are true in all wumpus worlds. Now we include the breeze percepts for the first two squares visited in the specific world the agent is in, leading up to the situation in Figure 7.3(b).

$$R_4 : \quad \neg B_{1,1} .$$
$$R_5 : \quad B_{2,1} .$$

The knowledge base, then, consists of sentences R_1 through R_5. It can also be considered as a single sentence—the conjunction $R_1 \wedge R_2 \wedge R_3 \wedge R_4 \wedge R_5$—because it asserts that all the individual sentences are true.

Inference in wumpus world

We start with the knowledge base containing R_1 through R_5, and show how to prove $\neg P_{1,2}$, that is, there is no pit in [1,2]. First, we apply biconditional elimination to R_2 to obtain

$$R_6: \quad (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1}).$$

Then we apply And-Elimination to R_6 to obtain

$$R_7: \quad ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1}).$$

Logical equivalence for contrapositives gives

$$R_8: \quad (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1})).$$

Now we can apply Modus Ponens with R_8 and the percept R_4 (i.e., $\neg B_{1,1}$), to obtain

$$R_9: \quad \neg(P_{1,2} \vee P_{2,1}).$$

Finally, we apply de Morgan's rule, giving the conclusion

$$R_{10}: \quad \neg P_{1,2} \wedge \neg P_{2,1}.$$

That is, neither [1,2] nor [2,1] contains a pit.

Conversion to CNF

$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

1. Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.

   $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$

2. Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$.

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$

3. Move $\neg$ inwards using de Morgan's rules and double-negation:

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$

4. Apply distributivity law ($\wedge$ over $\vee$) and flatten:

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

Resolution Inference algorithm

Conjunctive Normal Form (CNF) conjunction of disjunctions of literals E.g., $(A \lor \neg B) \land (B \lor \neg C \lor \neg D)$:
Basic intuition, resolve B, ¬Bto get $(A) \lor (\neg C \lor \neg D)$

Resolution inference rule (for CNF):

$$\frac{A_1 \lor \ldots \lor A_i \lor \ldots \lor A_n \quad B_1 \lor \ldots \lor \neg A_i \lor \ldots \lor B_m}{A_1 \lor \ldots \lor A_{i-1} \lor A_{i+1} \lor \ldots \lor A_n \lor B_1 \lor \ldots \lor B_{l-1} \lor B_{l+1} \lor \ldots \lor B_m}$$

E.g., $\dfrac{P_{1,3} \lor P_{2,2}, \qquad \neg P_{2,2}}{P_{1,3}}$

Horn Clauses

- Resolution can be exponential in space and time.

- If we can reduce all clauses to "Horn clauses" resolution is linear in space and time

A clause with at most 1 positive literal.

e.g. $A \vee \neg B \vee \neg C$

- Every Horn clause can be rewritten as an implication with a conjunction of positive literals in the premises and a single positive literal as a conclusion.

e.g. $B \wedge C \Rightarrow A$

Unification:

It is the process used to find substitutions that make different logical expressions look identical. **Unification** is a key component of all first-order Inference algorithms.

UNIFY (p, q) = θ where SUBST (θ, p) = SUBST (θ, q) θ is our unifier value (if one exists).

Ex: "Who does John know?"

UNIFY (Knows (John, x), Knows (John, Jane)) = {x/ Jane}.

UNIFY (Knows (John, x), Knows (y, Bill)) = {x/Bill, y/ John}.

UNIFY (Knows (John, x), Knows (y, Mother(y))) = {x/Bill, y/ John}

UNIFY (Knows (John, x), Knows (x, Elizabeth)) = FAIL

- The last unification fails because both use the same variable, X. X can't equal both John and Elizabeth. To avoid this change the variable X to Y (or any other value) in Knows(X, Elizabeth)

Knows(X, Elizabeth) → Knows(Y, Elizabeth)

Still means the same. This is called **standardizing apart.**

Skolemization

- Process for removing existential quantifiers is called Skolemization , after the logician Skolem.
- A constant that replaces an existentially quantified variable is called a Skolem constant and a function that is used in replacing a variable is called a Skolem function .
- A skolemized sentence is satisfiable exactly when the original sentence is satisfiable.

$\exists x \forall y\ P(x,y)$ becomes $\forall y\ P(c, y)$

$\forall x \exists y\ P(x,y)$ becomes $\forall x\ P(x, f(x))$

Steps for Resolution:

1. Conversion of facts into first-order logic.
2. Convert FOL statements into CNF
3. Negate the statement which needs to prove (proof by contradiction)
4. Draw resolution graph (unification).

Convert the given sentences into FOPL.

- Every student who studies Artificial Intelligence (AI) is intelligent.
- Some students study AI.
- John is a student.
- If a person is intelligent, then they get a good job.

Prove by resolution: Does John get a good job?

Example:

1.John likes all kind of food.

2.Apple and vegetable are food

3.Anything anyone eats and not killed is food.

4.Anil eats peanuts and still alive

5.Harry eats everything that Anil eats.

Prove by resolution that:

6.John likes peanuts.

Step-1: Conversion of Facts into FOL

In the first step we will convert all the given statements into its first order logic.

a. $\forall x$: food(x) $\rightarrow$ likes(John, x)

b. food(Apple) Λ food(vegetables)

c. $\forall x\ \forall y$: eats(x, y) $\Lambda \neg$ killed(x) $\rightarrow$ food(y)

d. eats (Anil, Peanuts) Λ alive(Anil).

e. $\forall x$: eats(Anil, x) $\rightarrow$ eats(Harry, x)

f. $\forall x$: $\neg$ killed(x) $\rightarrow$ alive(x)

g. $\forall x$: alive(x) $\rightarrow \neg$ killed(x)

added predicates.

h. likes(John, Peanuts)

Step-2: Conversion of FOL into CNF

In First order logic resolution, it is required to convert the FOL into CNF as CNF form makes easier for resolution proofs.

- **Eliminate all implication (→) and rewrite**
    - ∀x ¬ food(x) V likes(John, x)
    - food(Apple) Λ food(vegetables)
    - ∀x ∀y ¬ [eats(x, y) Λ ¬ killed(x)] V food(y)
    - eats (Anil, Peanuts) Λ alive(Anil)
    - ∀x ¬ eats(Anil, x) V eats(Harry, x)
    - ∀x¬ [¬ killed(x)] V alive(x)
    - ∀x ¬ alive(x) V ¬ killed(x)
    - likes(John, Peanuts).

- **Move negation (¬)inwards and rewrite**
    - ∀x ¬ food(x) V likes(John, x)
    - food(Apple) Λ food(vegetables)
    - ∀x ∀y ¬ eats(x, y) V killed(x) V food(y)
    - eats (Anil, Peanuts) Λ alive(Anil)
    - ∀x ¬ eats(Anil, x) V eats(Harry, x)
    - ∀x ¬killed(x)] V alive(x)
    - ∀x ¬ alive(x) V ¬ killed(x)
    - likes(John, Peanuts).

- **Rename variables or standardize variables**
    - ∀x ¬ food(x) V likes(John, x)
    - food(Apple) Λ food(vegetables)
    - ∀y ∀z ¬ eats(y, z) V killed(y) V food(z)
    - eats (Anil, Peanuts) Λ alive(Anil)
    - ∀w¬ eats(Anil, w) V eats(Harry, w)
    - ∀g ¬killed(g)] V alive(g)
    - ∀k ¬ alive(k) V ¬ killed(k)
    - likes(John, Peanuts).

- **Eliminate existential instantiation quantifier by elimination.**

  In this step, we will eliminate existential quantifier ∃, and this process is known as **Skolemization**. But in this example problem since there is no existential quantifier so all the statements will remain same in this step.

- **Drop Universal quantifiers.**

  - ¬ food(x) V likes(John, x)
  - food(Apple)
  - food(vegetables)
  - ¬ eats(y, z) V killed(y) V food(z)
  - eats (Anil, Peanuts)
  - alive(Anil)
  - ¬ eats(Anil, w) V eats(Harry, w)
  - killed(g) V alive(g)
  - ¬ alive(k) V ¬ killed(k)
  - likes(John, Peanuts).

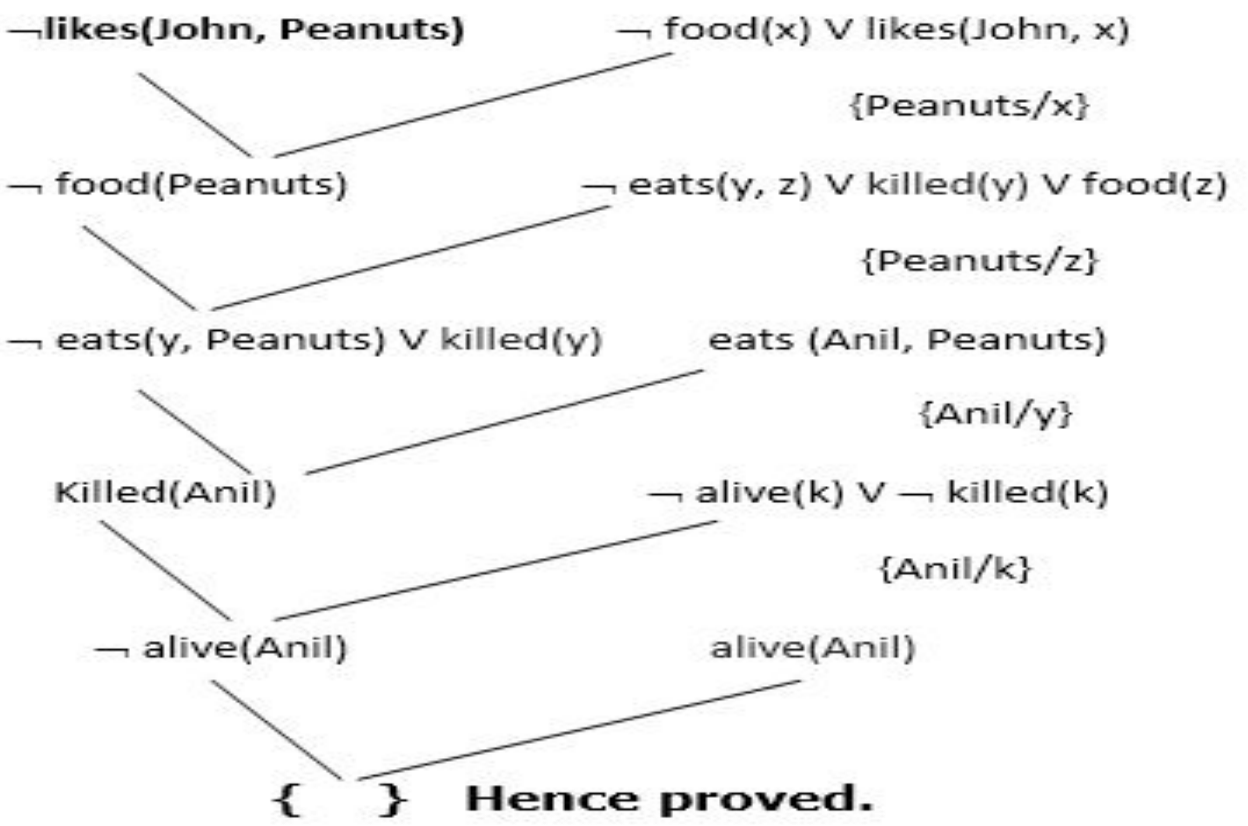
Step-3: Negate the statement to be proved

In this statement, we will apply negation to the conclusion statements, which will be written as ¬likes(John, Peanuts)

Step-4: Draw Resolution graph:

Now in this step, we will solve the problem by resolution tree using substitution. For the above problem, it will be given as f

Hence the negation of the conclusion has been proved as a complete contradiction with the given set of statements.



- **Hence the negation of the conclusion has been proved as a complete contradiction with the given set of statements.**

The law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American. • Prove that Col. West is a

... it is a crime for an American to sell weapons to hostile nations:

$$American(x) \wedge Weapon(y) \wedge Sells(x,y,z) \wedge Hostile(z) \Rightarrow Criminal(x)$$

Nono … has some missiles, i.e., $\exists x\ Owns(Nono,x) \wedge Missile(x)$:

$$Owns(Nono,M_1) \text{ and } Missile(M_1)$$

… all of its missiles were sold to it by Colonel West

$$Missile(x) \wedge Owns(Nono,x) \Rightarrow Sells(West,x,Nono)$$

Missiles are weapons:

$$Missile(x) \Rightarrow Weapon(x)$$

An enemy of America counts as "hostile“:

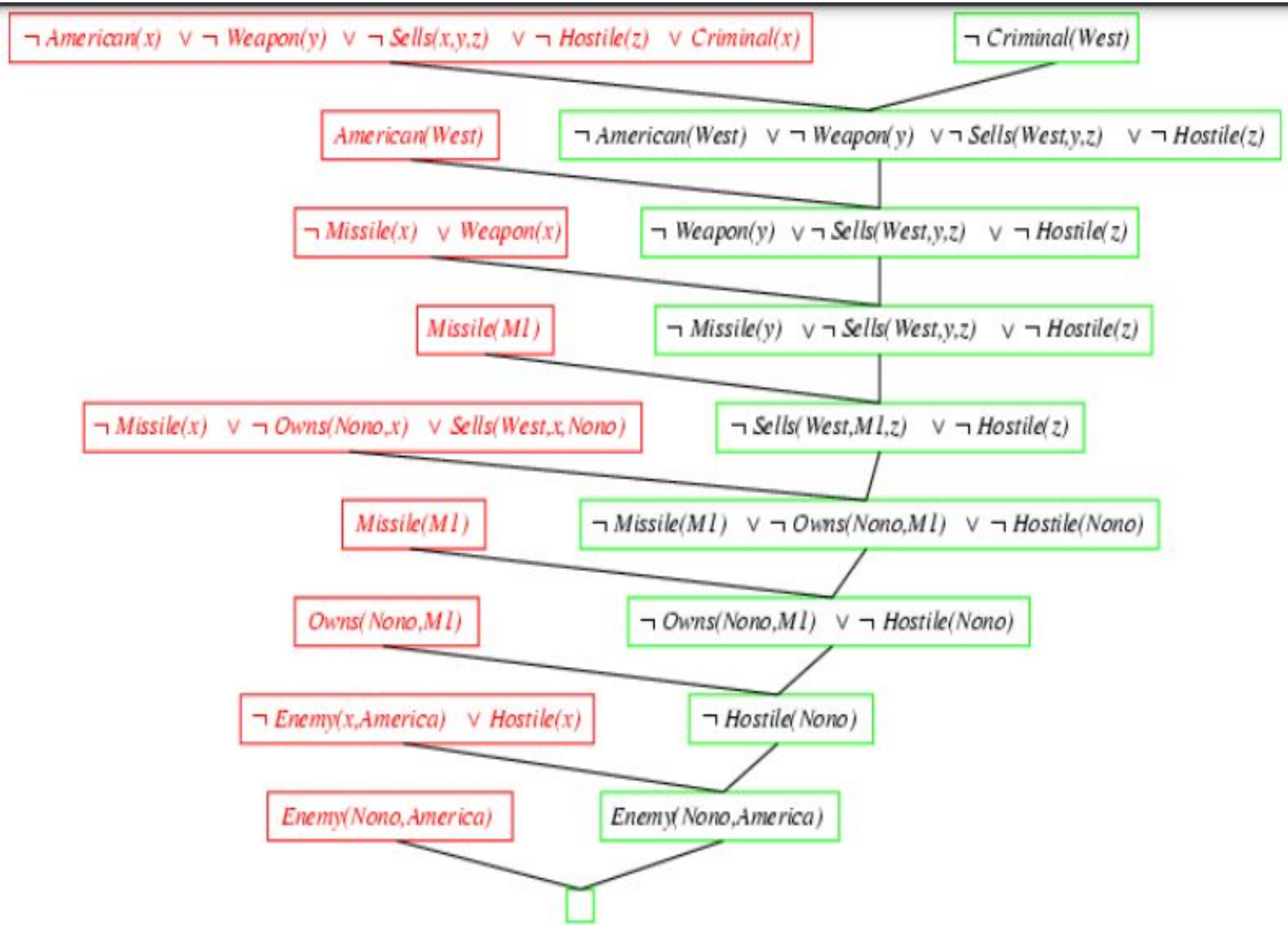
$$Enemy(x,America) \Rightarrow Hostile(x)$$

West, who is American …

$$American(West)$$

The country Nono, an enemy of America …

$$Enemy(Nono,America)$$



¬ American(x) ∨ ¬ Weapon(y) ∨ ¬ Sells(x,y,z) ∨ ¬ Hostile(z) ∨ Criminal(x)
¬ Criminal(West)
American(West)
¬ American(West) ∨ ¬ Weapon(y) ∨ ¬ Sells(West,y,z) ∨ ¬ Hostile(z)
¬ Missile(x) ∨ Weapon(x)
¬ Weapon(y) ∨ ¬ Sells(West,y,z) ∨ ¬ Hostile(z)
Missile(M1)
¬ Missile(y) ∨ ¬ Sells(West,y,z) ∨ ¬ Hostile(z)
¬ Missile(x) ∨ ¬ Owns(Nono,x) ∨ Sells(West,x,Nono)
¬ Sells(West,M1,z) ∨ ¬ Hostile(z)
Missile(M1)
¬ Missile(M1) ∨ ¬ Owns(Nono,M1) ∨ ¬ Hostile(Nono)
Owns(Nono,M1)
¬ Owns(Nono,M1) ∨ ¬ Hostile(Nono)
¬ Enemy(x,America) ∨ Hostile(x)
¬ Hostile(Nono)
Enemy(Nono,America)
Enemy(Nono,America)

Forward Chaining

- -When a decision is taken based on available data , the process is called forward chaining.
- It is **data driven inference technique.**
- **For example: It is raining**
- **Starting from the known facts, it triggers all the rules whose premises are satisfied, adding their concusion to the known facts.**
- -Reasoning in which the focus of attention starts with the known data. It can be used within an agent to derive conclusions from incoming percepts often without a specific query in mind.
- In this type of chaining, the inference engine starts by evaluating existing facts, derivations, and conditions before deducing new information. An endpoint (goal) is achieved through the manipulation of knowledge that exists in the knowledge base.
- Forward chaining can be used in planning, monitoring, controlling, and interpreting applications.

Advantages of Forward chaining

- It can be used to draw multiple conclusions.
- It provides a good basis for arriving at conclusions.
- It’s more flexible than backward chaining because it does not have a limitation on the data derived from it.

Disadvantages Forward chaining

- The process of forward chaining may be time-consuming. It may take a lot of time to eliminate and synchronize available data.
- Unlike backward chaining, the explanation of facts or observations for this type of chaining is not very clear. The former uses a goal-driven method that arrives at conclusions efficiently.

Backward chaining

Backward Chaining

- -based on the decision the initial data is fetched the process is called backward reasoning.
- It is **Goal driven reasoning technique.**
- **For example: where are my keys?**
- **It useful for answering specific questions**

- Backward chaining is a concept in artificial intelligence that involves backtracking from the endpoint or goal to steps that led to the endpoint. This type of chaining starts from the goal and moves backward to comprehend the steps that were taken to attain this goal.
- The backtracking process can also enable a person establish logical steps that can be used to find other important solutions.
- Backward chaining can be used in debugging, diagnostics, and prescription applications.

Properties of backward chaining

- The process uses an up-down approach (top to bottom).
- It’s a goal-driven method of reasoning.
- The endpoint (goal) is subdivided into sub-goals to prove the truth of facts.
- A backward chaining algorithm is employed in inference engines and complex database systems.
- The modus ponens inference rule is used as the basis for the backward chaining process. This rule states that if both the conditional statement (p->q) and the antecedent (p) are true, then we can infer the subsequent (q).

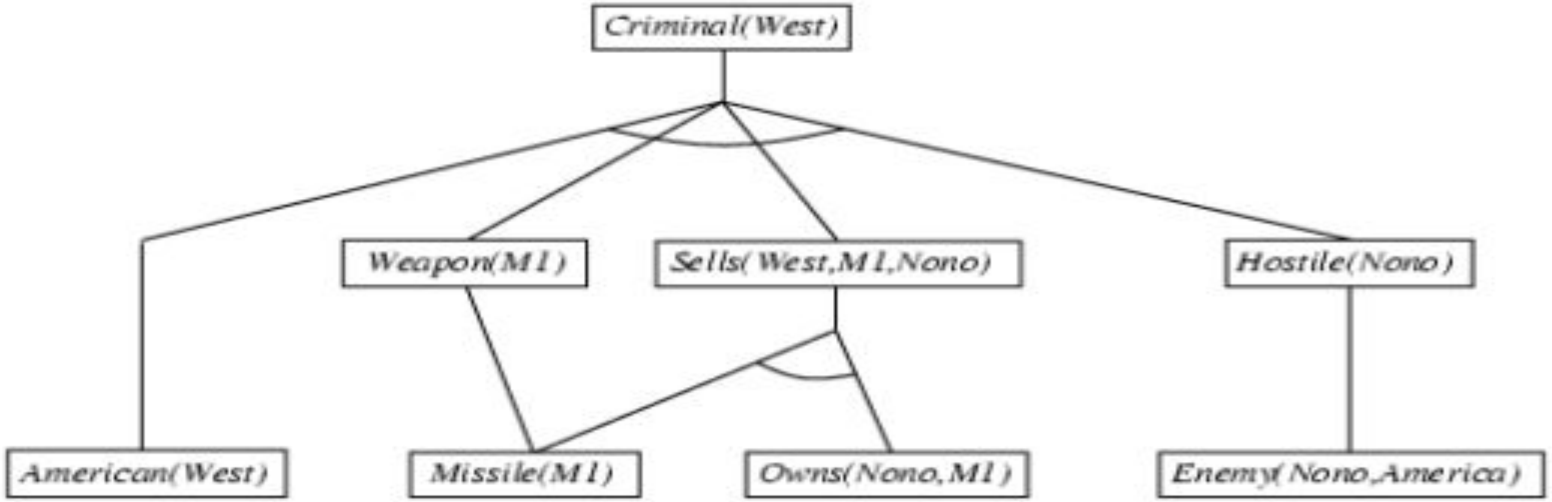
Advantages of backward chaining

- The result is already known, which makes it easy to deduce inferences.
- It's a quicker method of reasoning than forward chaining because the endpoint is available.
- In this type of chaining, correct solutions can be derived effectively if pre-determined rules are met by the inference engine.

Disadvantages of backward chaining

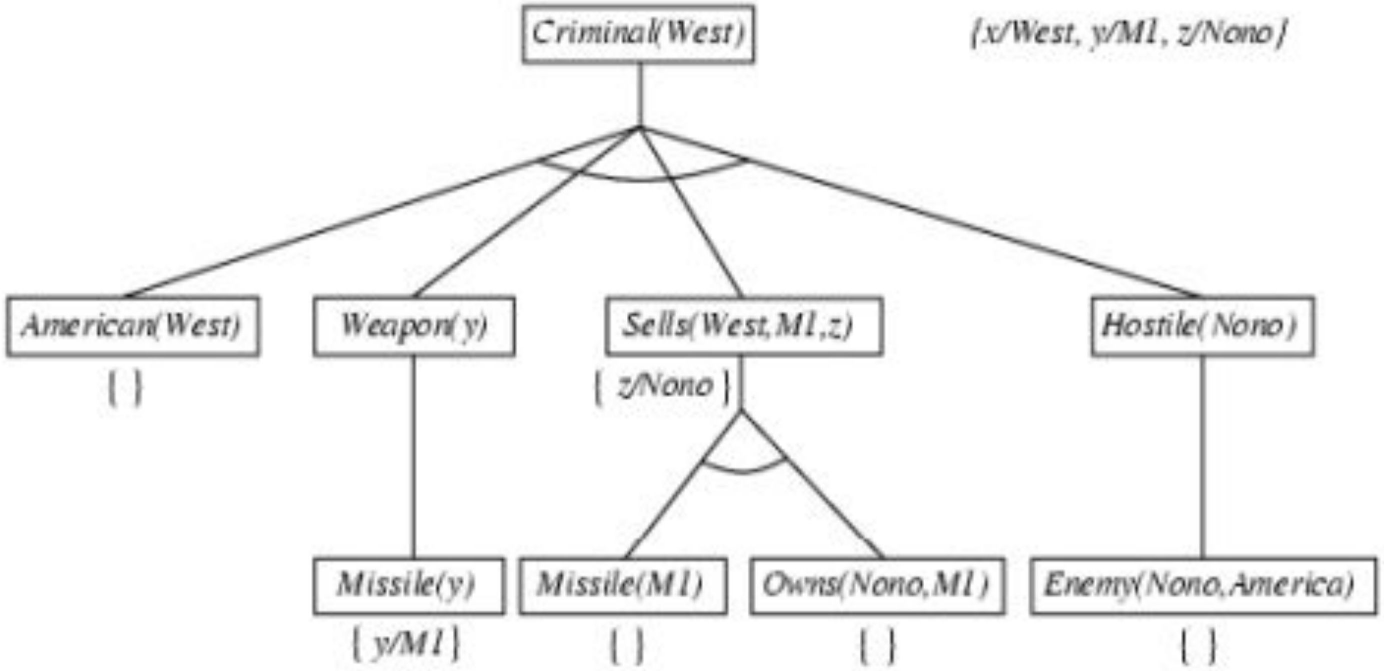
- The process of reasoning can only start if the endpoint is known.
- It doesn't deduce multiple solutions or answers.
- It only derives data that is needed, which makes it less flexible than forward chaining.

Forward chaining proof



*American(x) $\wedge$ Weapon(y) $\wedge$ Sells(x,y,z) $\wedge$ Hostile(z) $\Rightarrow$ Criminal(x)
*Owns(Nono,M1) and Missile(M1)
*Missile(x) $\wedge$ Owns(Nono,x) $\Rightarrow$ Sells(West,x,Nono)
*Missile(x) $\Rightarrow$ Weapon(x)
*Enemy(x,America) $\Rightarrow$ Hostile(x)
*American(West)
*Enemy(Nono,America)

Backward chaining example



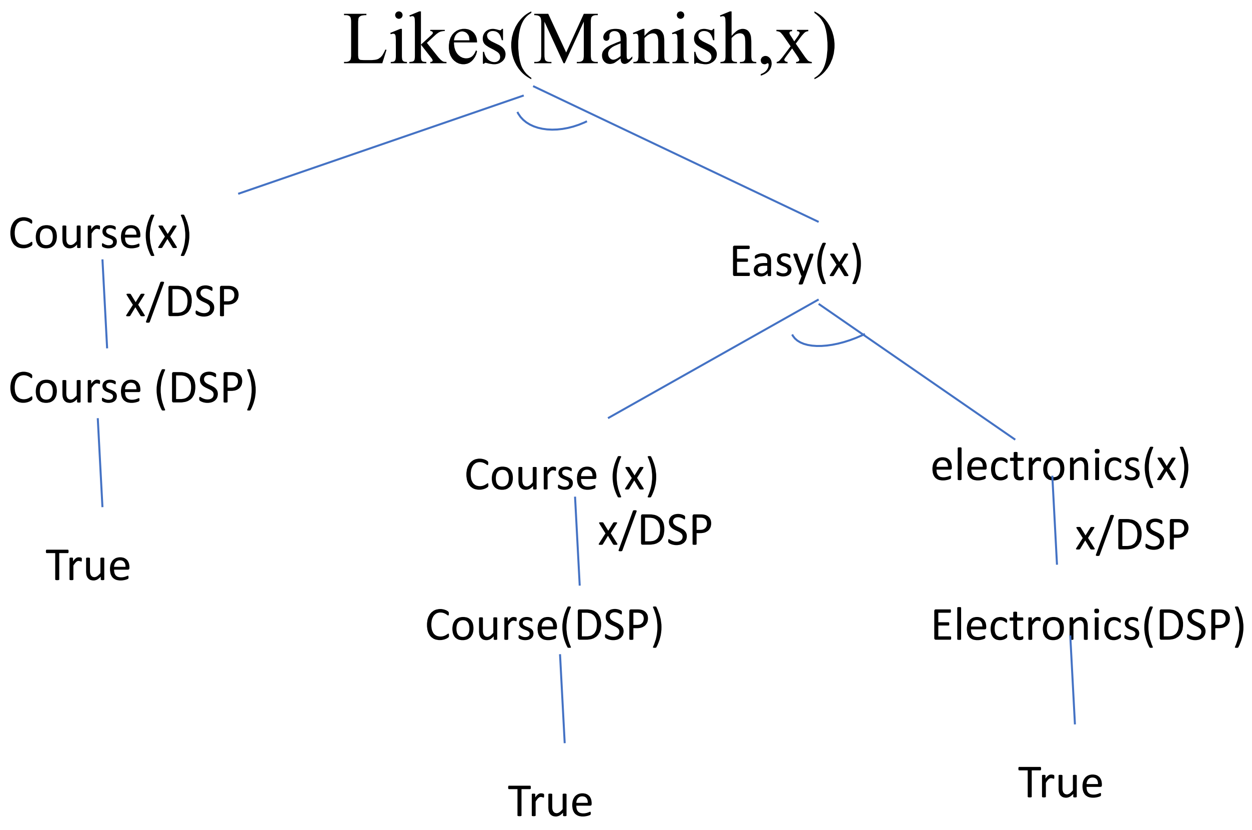
- Using predicate logic find the course of Manish’s liking for the following
- 1. Manish only likes easy courses
- 2. Computer courses are hard
- 3. All electronics courses are easy.
- 4. DSP is an electronics course.

Convert to FOL

Proof by backward chaining

- 1. $\forall x$: course(x) $\wedge$ easy(x) -> likes (Manish,x)
- 2. $\forall x$: course(x) $\wedge$ computer(x) -> hard (x)
- 3. $\forall x$: course(x) $\wedge$ electronics(x) -> easy (x)
- 4. Electronics (DSP)
- 5. course(DSP)

•


Likes(Manish,x)
Course(x)
x/DSP
Course (DSP)
True
Easy(x)
Course (x)
x/DSP
Course(DSP)
True
electronics(x)
x/DSP
Electronics(DSP)
True

Knowledge Engineering process

1. **Identify the task-** find range of questions that KB supports and facts available

2. **Assemble the relevant knowledge-** knowledge acquisistion from domain experts

3. **Defining the vocabulary-** translate domain concepts to logic level names. Ontology (vocabulary) is created which determines what things existes but not their reationships

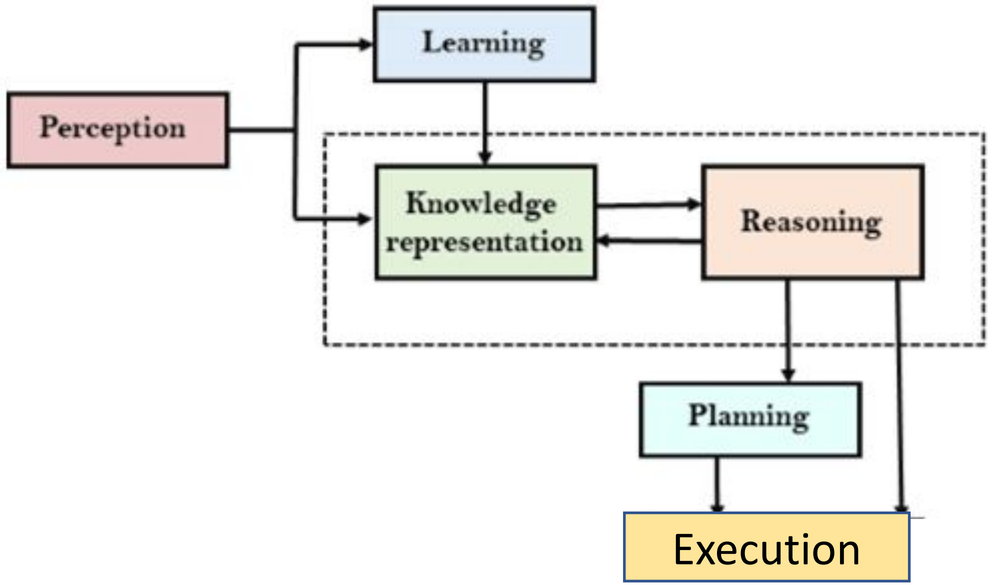
4. **Encoding the general knowledge about the domain-** axioms are written for all vocabulary terns and gaps if any identified by domain experts

5. **Encode the problem-** writing simple atomic sentences about instances of concepts that are already a part of ontology.

6. **Query the Knowledge base-** apply inference procedure on axioms and problem specific facts to derive facts we are interested in knowing

7. **Debug the knowledge base-** check for correctness of inference procedures, missing axioms, incorrect axioms etc..

AI knowledge cycle



Uncertainty

- Due to partial observability, nondeterminism or combination of two
- Agent may not know where it may land up after a sequence of action.

Handling uncertain knowledge

- ❑ $\forall p$ Symptom(p, Toothache) $\Rightarrow$ Disease(p, Cavity)

  - Not correct – toothache can be caused in many other cases
  - $\forall p$ Symptom(p, Toothache)

    $\Rightarrow$ Disease(p, Cavity) $\vee$
    Disease(p, GumDisease) $\vee$
    Disease(p, ImpactedWisdom) $\vee$ …

- ❑ $\forall p$ Disease(p, Cavity) $\Rightarrow$ Symptom(p, Toothache)

  - This is not correct either, since all cavities do not cause toothache

Trying to use logic in situations like medical diagnosis fails for three reasons and hence we go for probability theory:

1. **Laziness:** It is too much work to list the complete set of antecedents and consequents to ensure an exception less rule and too hard to use such rule.

2. **Theoretical ignorance:** medical science has no complete theory for domain.

3. **Practical Ignorance**: even if we know all the rules, we might be uncertain about a particular patient because not all the necessary tests have been or can be run

Use of probability theory

- Probability provides a way of summarizing the uncertainty that comes from our laziness and ignorance.
- Agents knowledge at best can provide only degree of belief in the relevant sentences
- To make choices an agent must have preferences between the different possible outcomes of the various plans
- So we use utility theory to represent and reason with preferences

Decision Theory= probability theory + Utility Theory

Prior & Posterior Probability

- Unconditional / Prior Probability
- E.g. P(cavity)=0.2

- Conditional / Posterior Probability
- E.g. P(cavity | toothache)=0.6

- Variables in probability theory called random variables. Their names begins with first letter capitalize.
- Random variable can be Boolean, discrete or continuous.

Axioms of Probability

1. All prob are between 0 and 1: $0 \le P(A) \le 1$
2. $P(True) = 1$ and $P(False) = 0$
3. $P(A \vee B) = P(A) + P(B) - P(A \wedge B)$

Bayes' Rule

$P(A \wedge B) = P(A \mid B)\, P(B)$

$P(A \wedge B) = P(B \mid A)\, P(A)$ ………**Product rules**

$$P(B \mid A) = \frac{P(A \mid B) P(B)}{P(A)}$$ ….**Bayes rule/ Bayes theorem/ Bayes law**

Applying Bayes rule

$$P(cause \mid effect) = \frac{P(effect \mid cause)P(cause)}{P(effect)}.$$

example, a doctor knows that the disease meningitis causes the patient to have a stiff neck, say, 70% of the time. The doctor also knows some unconditional facts: the prior probability that a patient has meningitis is 1/50,000, and the prior probability that any patient has a stiff neck is 1%. Letting s be the proposition that the patient has a stiff neck and m be the proposition that the patient has meningitis, we have

$$\begin{aligned} P(s \mid m) &= 0.7 \\ P(m) &= 1/50000 \\ P(s) &= 0.01 \\ P(m \mid s) &= \frac{P(s \mid m)P(m)}{P(s)} = \frac{0.7 \times 1/50000}{0.01} = 0.0014 . \end{aligned} \qquad (13.14)$$

P(cause|effect) describes diagnostic direction

Joint Probability Distribution

- Distribution of probability of multiple variables.
- For ea. Probability distribution of cavity and weather.

- Full joint Probability Distribution
- It is joint distribution for all the random variables.
- If variables are Cavity, toothache, Weather the full joint distribution atable ill have 2x2x4 =16 entries.

Full joint distribution for the toothache, Cavity, catch world

	Toothache		¬Toothache	
	Catch	¬Catch	Catch	¬Catch
Cavity	0.108	0.012	0.072	0.008
¬Cavity	0.016	0.064	0.144	0.576

$$P(cavity \mid toothache) = \frac{P(cavity \wedge toothache)}{P(toothache)} = \frac{0.108 + 0.012}{0.108 + 0.012 + 0.016 + 0.064} = 0.6\,.$$

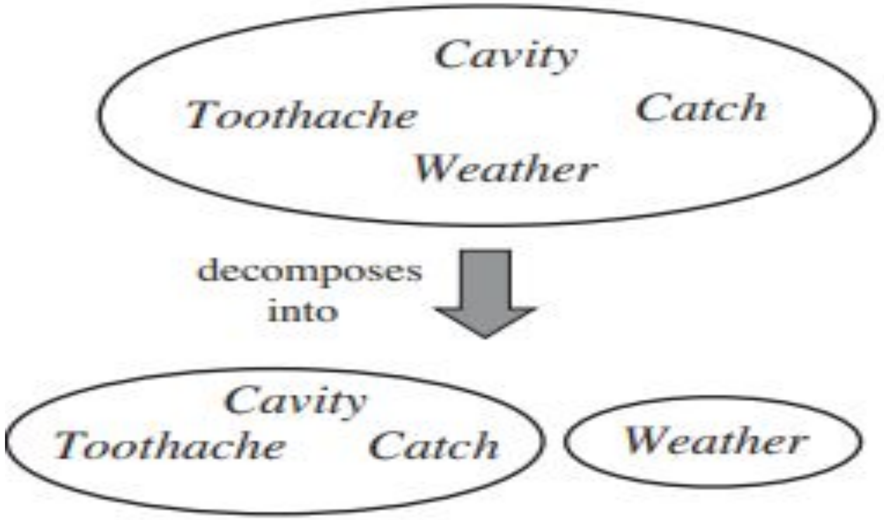
$$P(\neg cavity \mid toothache) = \frac{P(\neg cavity \wedge toothache)}{P(toothache)} = \frac{0.016 + 0.064}{0.108 + 0.012 + 0.016 + 0.064} = 0.4\,.$$

Inference using Full joint distribution table:

P(cavity)=0.108+0.012+0.072+0.008=0.2

P(cavity ^ toothache)= 0.108+0.012+0.072+0.008+0.016+0.064=0.28

Factoring a large joint distribution into smaller distributions using absolute independence



- Here weather is independent of ones dental problems.
- Independence between two propositions a and b can be given by,

$P(a|b)=P(a)$

$P(b|a)=P(b)$

$P(a \wedge b)=P(a)P(b)$

Representing knowledge in an uncertain domain

- **The full joint probability distribution** can answer any question about the domain, but can become intractably large as the number of variables grows.
- Furthermore, specifying probabilities for possible worlds one by one is unnatural and tedious

- **independence and conditional independence relationships** among variables can greatly reduce the number of probabilities that need to be specified in order to define the full joint distribution.

- **A data structure called a Bayesian network BAYESIAN NETWORK** to represent the dependencies among variables. Bayesian networks can represent essentially any full joint probability distribution and in many cases can do so very concisely.

Bayesian network / Belief Network / Probabilistic network

A Bayesian network is a directed graph in which each node is annotated with quantitative probability information. The full specification is as follows:

1. Each **node corresponds to a random variable**, which may be discrete or continuous.

2. A set of **directed links or arrows connects pairs of nodes**. If there is an arrow from node X to node Y , X is said to be a parent of Y. The graph has **no directed cycles** (and hence is a directed acyclic graph, or DAG.

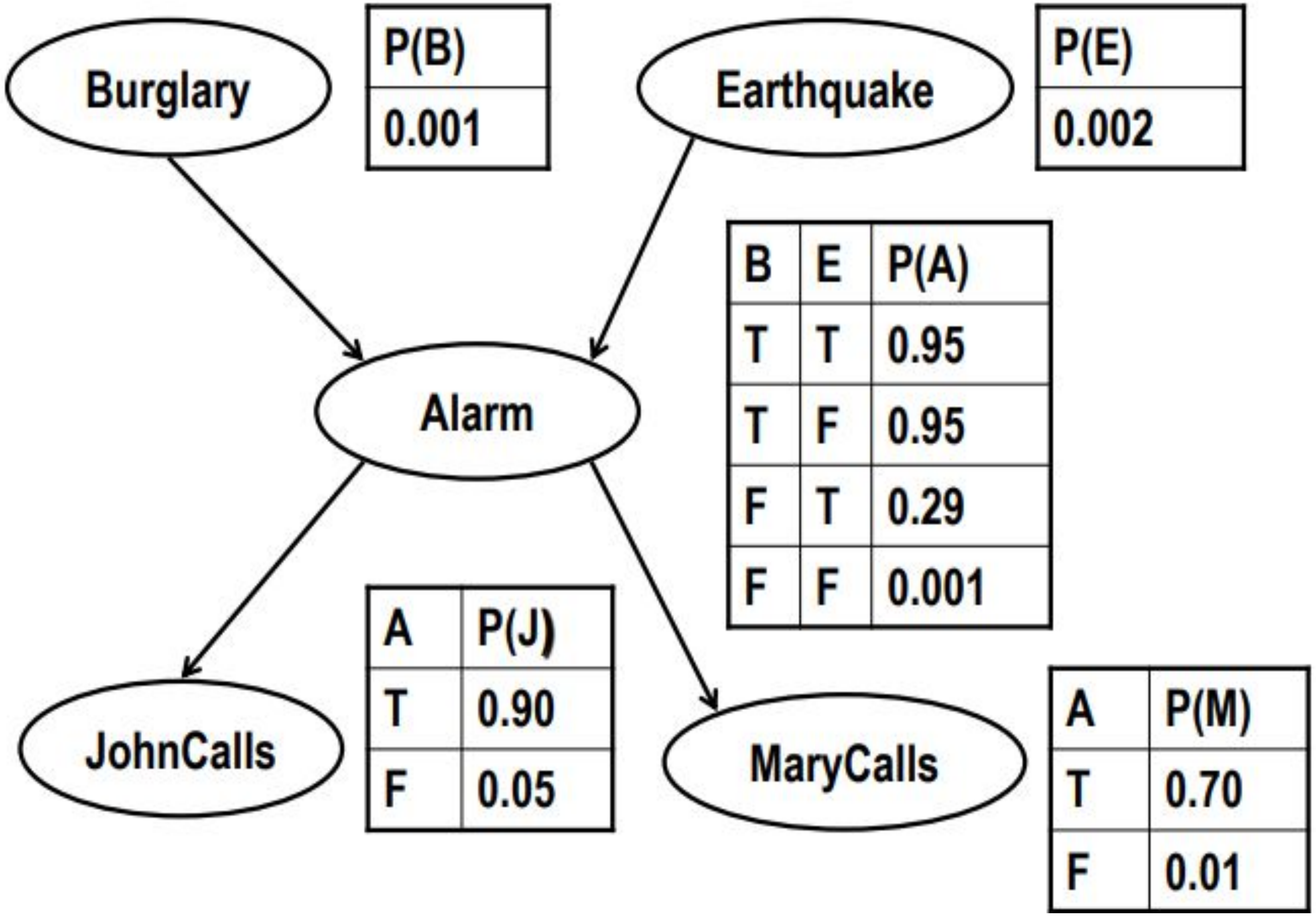
3. **Each node Xi has a conditional probability distribution**

P(Xi | Parents(Xi))that quantifies the effect of the parents on the node

Example

- ❑ Burglar alarm at home
  - Fairly reliable at detecting a burglary
  - Responds at times to minor earthquakes
- ❑ Two neighbors, on hearing alarm, calls police
  - John always calls when he hears the alarm, but sometimes confuses the telephone ringing with the alarm and calls then, too.
  - Mary likes loud music and sometimes misses the alarm altogether

Belief Network Example



P(B)
0.001

P(E)
0.002

B	E	P(A)
T	T	0.95
T	F	0.95
F	T	0.29
F	F	0.001

A	P(J)
T	0.90
F	0.05

A	P(M)
T	0.70
F	0.01

The joint probability distribution

- A generic entry in the joint probability distribution $P(x_1, \ldots, x_n)$ is given by:

$$P(x_1,\ldots,x_n) = \prod_{i=1}^{n} P(x_i \mid Parents(X_i))$$

The joint probability distribution

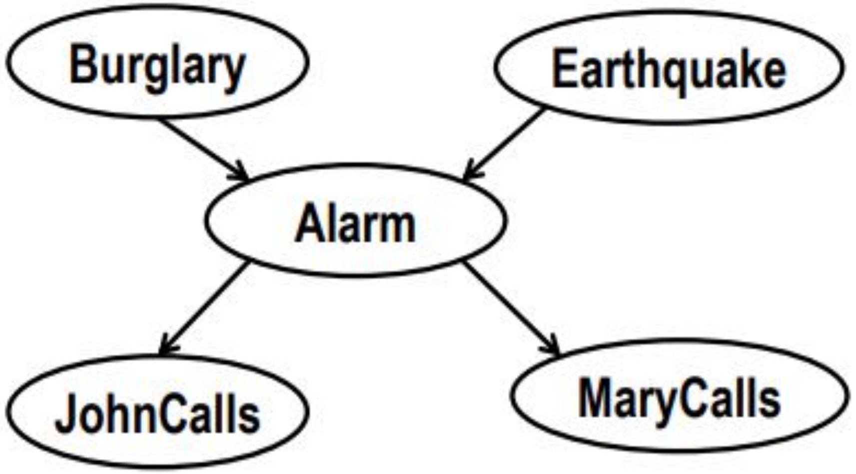
- Probability of the event that the alarm has sounded but neither a burglary nor an earthquake has occurred, and both Mary and John call:

$$P(J \wedge M \wedge A \wedge \neg B \wedge \neg E)$$

$$= P(J \mid A)\ P(M \mid A)\ P(A \mid \neg B \wedge \neg E)\ P(\neg B)\ P(\neg E)$$

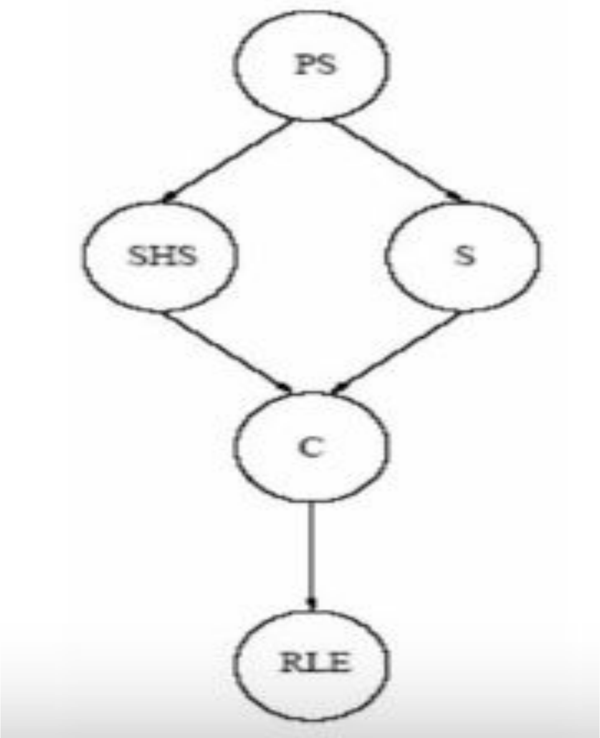
$$= 0.9 \times 0.7 \times 0.001 \times 0.999 \times 0.998$$

$$= 0.00062$$



Consider a situation in which we want to reason about the relationship between smoking and lung cancer. Intuitively, we know that whether or not a person has cancer is directly influenced by whether she is exposed to second-hand smoke and whether she smokes. Both of these things are affected by whether her parents smoke. Cancer reduces a person's life expectancy. (i) Draw the Bayesian network. (ii) How many independent values are required to specify all the conditional probability tables (CPTs) for your network? [10]

(i)



(ii) No. of independent values required to specify all the CPT s of the network

1+2+2+4+2=11

i.e.

1(PS)+2(SHS)+2(S)+4(C)+2(RLE)=11