CSE-AI TY A div

Student Name: Prachee Prasad

Roll No.: 381060

Assignment 8

Implement Backward Chaining Algorithm (Medical Diagnosis)

Problem Statement:

To implement a **Backward Chaining Algorithm** for **medical diagnosis**, aiming to prove whether a patient has the **Flu** by working backward from the goal using a set of rules and initial facts.

Objective:

To understand **goal-driven reasoning** in AI, where the system starts from a query (goal) and works backward to verify whether the goal can be inferred from known facts.

Requirements:

- Input: Initial facts (observed symptoms) – Fever, Cough, Body Ache.
- Output: Step-by-step backward reasoning showing which rules are applied to prove the goal (Flu).
- Approach: **Backward Chaining** — recursively check if goal can be satisfied using rules and known facts.

Operating System:

Windows / Linux / macOS

Libraries and Packages Used:

- **C++ iostream**, **string**, **map**, **set** for managing rules and facts.
- No external libraries required.

Theory:

Definition:

Backward Chaining is a **goal-driven reasoning** technique in AI, where the inference starts from the **goal** and works backward to determine if it can be derived from known facts using available rules.

Structure:

- **Facts:** Observed symptoms provided by the doctor.
- **Rules:** Knowledge base linking symptoms to diagnoses:
    1. Fever $\wedge$ Cough $\Rightarrow$ ViralInfection
    2. ViralInfection $\wedge$ BodyAche $\Rightarrow$ FluSymptoms
    3. FluSymptoms $\Rightarrow$ Flu
- **Inference Engine:** Recursively checks if the goal can be proved by evaluating antecedents of rules.

Methodology:

1. Start from the **goal** (Flu).
2. Check if the goal is already a **known fact**.
3. If not, find rules where the **goal is the consequent**.
4. Recursively attempt to **prove all antecedents** of those rules.
5. If all antecedents are true or can be proven, the goal is satisfied.
6. Display the **step-by-step reasoning** until the goal is proved or disproved.

Advantages:

- Focused on a specific **goal**, avoiding unnecessary computations.
- Efficient for **query-based reasoning** in expert systems.
- Useful when the goal is known and facts are sparse.

Limitations:

- Can be inefficient if the rule base is large with many backward paths.
- Requires **complete and correct rules** for accurate inference.
- Does not handle uncertainty without extensions.

Working / Algorithm:

Algorithm Steps:

1. Input initial facts: Fever, Cough, BodyAche.
2. Define goal: Flu.
3. Call `backward_prove(goal)` function:
    - Check if the goal is already a fact.
    - If not, find rules where the goal is the consequent.
    - Recursively check if all antecedents of the rule can be proved.
4. If all antecedents are satisfied, mark the goal as **proved**.
5. Continue recursively until the goal is proved or no rules apply.
6. Display each step, showing which rule is being applied and which antecedents are being checked.

Example Step-by-Step Inference:

```
Goal: Flu
```

```
Step 1: Check if Flu is a fact → No
Step 2: Look for rules producing Flu → Rule 3: FluSymptoms ⇒ Flu
Step 3: Prove FluSymptoms
        - Check facts → No
        - Look for rules → Rule 2: ViralInfection ∧ BodyAche ⇒
FluSymptoms
Step 4: Prove ViralInfection
        - Check facts → No
        - Look for rules → Rule 1: Fever ∧ Cough ⇒ ViralInfection
Step 5: Prove Fever and Cough → Both are facts → ViralInfection proved
Step 6: BodyAche → Already a fact
Step 7: FluSymptoms proved → Apply Rule 3 → Flu proved
```

Conclusion:

The **Backward Chaining Algorithm** demonstrates **goal-driven reasoning**, efficiently proving whether a goal can be inferred from known facts using a set of rules.
It is widely used in **expert systems**, **diagnostic systems**, and AI applications where the **query or goal is known in advance**, showcasing logical reasoning and inference mechanisms.