**Assignment 8**

**Title:**

Implementation of Backward Chaining Algorithm for Reasoning in Knowledge-Based Systems

**Aim:**

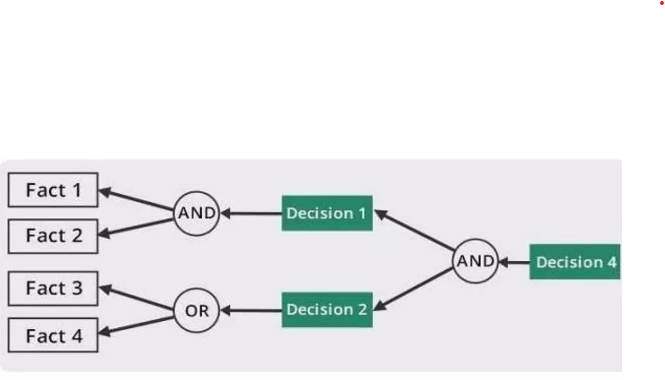
To implement the **Backward Chaining Algorithm** in a knowledge-based system and demonstrate its use in deriving facts or conclusions starting from a goal.

**Objectives:**

1. To understand the working of the **Backward Chaining** inference technique.
2. To implement a rule-based system that uses **Backward Chaining** to check the validity of a hypothesis or conclusion based on predefined rules and facts.
3. To apply **Backward Chaining** for efficiently deriving facts or verifying goals in knowledge-based reasoning.
4. To observe how conclusions are derived by tracing the rules back from the goal to the known facts.

**Theory:**

**Backward Chaining Overview:**



**Backward Chaining** is a goal-driven reasoning method used in **rule-based expert systems**. It works by starting with a **goal (or hypothesis)** and trying to prove it by recursively looking for facts or rules that support it. This process continues until either the goal is proven true or no supporting rules or facts are found.

**How Backward Chaining Works:**

1. **Goal**: Start with a goal that needs to be proven.
2. **Rules**: A set of **if-then** rules are defined, where the **if** part represents the condition (premises) and the **then** part represents the action (conclusion).
3. **Inference Process**:
   * The algorithm begins with the goal and searches for rules where the **then** part matches the goal.
   * For each rule that matches, the algorithm checks if the **if** part (premises) can be satisfied by facts or by proving other sub-goals recursively.
   * This process continues until the premises are verified using facts, or it is determined that the goal cannot be achieved.

**Example Scenario:**

Consider a knowledge-based system for medical diagnosis:

* **Goal**: Determine if the patient has the flu.
* **Rules**:
  + If the patient has a fever and a cough, then the patient has the flu.
  + If the patient has the flu, recommend bed rest.
* **Facts**:
  + The patient has a fever.
  + The patient has a cough.

Using backward chaining, the system will try to prove the goal (flu diagnosis) by checking if there are rules that support it, then verifying the premises (fever and cough).

**Algorithm Steps:**

1. **Input**: A goal to be proven and a knowledge base consisting of if-then rules.
2. **Match**: Search for rules where the **then** part (conclusion) matches the goal.
3. **Recursively Verify Premises**: For each matching rule, try to prove the **if** part (premises) by checking facts or recursively proving sub-goals.
4. **Prove or Disprove**: Continue the process until the premises are satisfied (goal proven) or no rules can support the goal (goal disproven).

**Procedure:**

**1. Define the Knowledge Base:**

* The knowledge base consists of a list of **if-then rules**.
* Each rule has two parts:
  + **Premise** (if): A condition that must be proven true.
  + **Conclusion** (then): A goal that is derived if the premise is true.

**2. Set the Goal:**

* Identify the goal (or hypothesis) that needs to be proven.

**3. Implement the Backward Chaining Algorithm:**

* The algorithm recursively tries to prove the goal by tracing the rules backward.
* **Backward Chaining Algorithm Steps**:
  1. **Start** with a goal that needs to be proven.
  2. **Search** for rules where the conclusion (then part) matches the goal.
  3. **For each rule**, recursively try to prove the premise (if part) using facts or sub-goals.
  4. **If all premises are satisfied**, the goal is proven.
  5. **If no rules can prove the goal**, the goal is disproven.

**4. Stop Condition:**

* The process stops when:
  + The goal is proven (all premises of supporting rules are satisfied).
  + The goal cannot be proven (no supporting rules or facts are found).

**Implementation Details:**

* **Input**:
  + A goal to be proven.
  + A set of if-then rules representing the knowledge base.
* **Processing**:
  + The algorithm searches for rules where the conclusion matches the goal.
  + For each matching rule, the premises are recursively verified using known facts or by trying to prove other sub-goals.
* **Output**:
  + A boolean result indicating whether the goal was proven or not.
  + If the goal is proven, the facts supporting the conclusion are returned.

**Sample Knowledge Base and Goal**:

**Goal**:

Goal: Flu

**Rules**:

Rule 1: If (Fever and Cough), then Flu.

Rule 2: If Flu, then Recommend bed rest.

Rule 3: If (Sore throat and Cough), then Cold.

**Facts:**

1. Fever

2. Cough

**Expected Output**:

Goal (Flu): Proven

Supporting Facts: [Fever, Cough]

**Expected Output:**

The algorithm will output whether the goal was proven or not, along with the supporting facts if the goal was proven.

For example:

Copy code

Goal: Flu

Result: Proven

Supporting Facts: [Fever, Cough]

**Conclusion:**

In this lab, we implemented the **Backward Chaining Algorithm** to reason in a knowledge-based system. The algorithm started with a goal and traced back through the rules to verify the premises. **Backward Chaining** is particularly useful for situations where a specific hypothesis or conclusion needs to be verified, and it efficiently works by focusing only on the goal, reducing unnecessary computations.