# **Backward Chaining Analogy — Detective Solving a Mystery**

## **AIM**

To simulate a detective using **backward chaining** reasoning, where the detective starts with a conclusion (the crime) and works backward by verifying clues (premises) to solve the case.

## **PROCEDURE**

1. **Define clues and evidence** (facts the detective already has).
2. **Define logical connections** (rules) linking evidence to possible conclusions.
3. To solve (prove) the crime:  
   * If the conclusion is in evidence (facts), declare solved.
   * Otherwise, find rules that conclude the crime.
   * For each such rule, check if all evidence (premises) hold by recursive checking.
   * If all premises are proven, conclude the crime is solved.
4. If no evidence or rules support the crime, the detective cannot solve the case.

## **CODE :**

class Detective:

def \_\_init\_\_(self, rules, evidence):

"""

rules: dict {conclusion: [premises]}

evidence: set of known clues/facts

"""

self.rules = rules

self.evidence = set(evidence)

def solve\_case(self, conclusion):

if conclusion in self.evidence:

return True # Case solved directly by evidence

for possible\_conclusion, premises in self.rules.items():

if possible\_conclusion == conclusion:

# Check all premises recursively

if all(self.solve\_case(p) for p in premises):

self.evidence.add(conclusion) # Add solved case

return True

return False

# Detective's knowledge base

rules = {

'Murder committed': ['Suspect had motive', 'Suspect was at crime scene'],

'Suspect had motive': ['Suspect was in financial trouble'],

'Suspect was at crime scene': ['Witness saw suspect near crime scene']

}

evidence = ['Suspect was in financial trouble', 'Witness saw suspect near crime scene']

detective = Detective(rules, evidence)

print(f"Can detective solve the 'Murder committed' case? {detective.solve\_case('Murder committed')}")

print(f"Evidence after investigation: {detective.evidence}")

## **OUTPUT :**

Can detective solve the 'Murder committed' case? True

Evidence after investigation: {'Suspect had motive', 'Suspect was in financial trouble', 'Witness saw suspect near crime scene', 'Suspect was at crime scene', 'Murder committed'}

## **EXPLANATION**

* The detective wants to prove the conclusion: "Murder committed".
* To prove it, must verify two premises: "Suspect had motive" and "Suspect was at crime scene".
* The detective recursively checks if these premises hold:  
  + "Suspect had motive" depends on "Suspect was in financial trouble" (known evidence).
  + "Suspect was at crime scene" depends on "Witness saw suspect near crime scene" (known evidence).
* Since both premises can be verified from evidence, the detective proves "Murder committed".
* The knowledge base (evidence) is updated with new proven facts.

## **CONCLUSION**

* Backward chaining can be thought of as a detective working backward from a crime to clues.
* It breaks down big conclusions into smaller verifiable facts.
* When all supporting facts are found, the conclusion is accepted.
* This analogy makes backward chaining intuitive for goal-driven problem solving.