Introduction To Artificial Intelligence

Lab 2 – Logic

1. Evaluation:

In this lab, I am required to evaluate disadvantages of resolution method for propositional logic, propose my own solution for specific problem.

Resolution method for Propositional logic:

+ Advantages:

1. Soundness:
   To establish the logical soundness of the resolution theorem, we employ a proof by contradiction. Assume a set of clauses S entails a goal α. To prove this, we initially assume the negation of α (¬α), suggesting S entails ¬α. We assign truth values to propositions in α and claim that under this assignment, the resolution of any two clauses from S will always yield a true result. This implies that, even after exhaustively applying resolution to all clauses, the resulting clause will not be false. Consequently, the assumption that S entails ¬α leads to a contradiction, establishing the logical soundness of the resolution theorem.
2. Completeness:
   The Resolution Theorem is considered complete, implying that any complete search algorithm through resolution can derive any conclusion entailed by a knowledge base. However, this completeness is limited, as resolution cannot generate consequences for a true statement like P v Q from the truth of P alone. It excels in refutation completeness, allowing it to confirm or refute a sentence but not enumerate true sentences.

+ Disadvantages

1. Limited Expressiveness:

   Consider a classic argument:
   All men are mortal
   Socrates is a man
   Conclusion: Socrates is mortal.
   Then we have:
   p = All men are mortal
   q = Socrates is a man
   r = Socrates is mortal,
   Each premise and conclusion in the argument schema must use distinct logical variables, as there are no logical connectives involved. In propositional logic, the absence of quantifiers limits the representation of "all," requiring three separate and independent variables for the three components: two premises and the conclusion.

Propositional logic fails to capture the nuanced relationship between an individual being a man and that individual being mortal. The need for variables and quantification arises unless one is prepared to create separate statements for the mortality of each known man.

Propositional logic lacks the expressiveness required for effective knowledge representation. It is too coarse to describe object properties easily and lacks the structure to express relations between entities, hindering reasoning about real-world entities. Additionally, it doesn't allow making generalized statements about classes of similar objects, presenting serious limitations in real-world reasoning.

5 databases for example:
Example 1:
Consider the following Knowledge Base:
1. -A or B
2. -C or B
3. A or C or B
4. -B

Check if C or -A

Negate the conclusion: C or -A => -C and A (*)
5. Resolve (1) and (4): -A
6. Resolve (5) and (3): C or B
7. Resolve (1) and (6): C or -A
8. Resolve (*) and (7): Hence proved C or -A

Example 2:
Consider the following Knowledge Base:
1. The humidity is high or the sky is cloudy.
2. If the sky is cloudy, then it will rain.
3. If the humidity is high, then it is hot.
4. It is not hot.
Goal: It will rain.
Let P: Humidity is high; Q: Sky is cloudy; R: It will rain and S: It is hot.
KB will become:
1. P or Q
2. Q => R = -Q or R
3. P => S = -P or S
4. -S
5. -R (Negate the conclusion)
6. Resolve (1) and (2): P or R
7. Resolve (3) and (6): R or S
8. Resolve (7) and (4): R
9. Resolve (8) and (5): null => Hence KB entails R

Example 3:

Consider the following Knowledge Base:
1. Gita likes all kinds of food
2. Mango and chapati are food.
3. Gita eats almond and is still alive.
4. It is not hot.

Example 4:
Anyone passing his logic exams and winning the lottery is happy. But anyone who studies or is lucky can pass all his exams. John did not study but he is lucky. Anyone who is lucky wins the lottery.
Is John happy?
1. Anyone passing his logic exams and winning the lottery is happy.
∀x ((Pass(x, History) ∧ Win(x, Lottery)) → Happy(x))
2. Anyone who studies or is lucky can pass all his exams.
∀x ∀y ((Study(x) ∨ Lucky(x)) → Pass(x, y))
3. John did not study but he is lucky
(¬Study (John) ∧ Lucky (John))
4. Anyone who is lucky wins the lottery.
∀x ((Lucky (x)) → Win(x, Lottery))
The KB will become:
1. ¬Pass(x, History) ∨ ¬Win(x, Lottery)) ∨ Happy(x))
2. Study(y) ∨ Pass (y, z)
3. ¬Lucky (w) ∨ Pass (w, v)
4. ¬Study(John)
5. Lucky (John)
6. ¬Lucky (u)) ∨ Win(u, Lottery)
7. ¬Happy(John) (Negate the conclusion)
8. Resolve (1) and (6): ¬Pass(v, History) ∨ ¬Lucky (v) ∨ Happy(v))
9. Resolve (8) and (7): ¬Pass(John, History) ∨ ¬Lucky (John)
10. Resolve (9) and (5): ¬Pass(John, History)
11. Resolve (10) and (3): ¬Lucky (John)
12. Resolve (11) and (5): null
=> John is happy

Example 5:
Those people who read are not stupid. John can read and is wealthy. All people who are not poor and are smart are happy. Happy people have exciting lives. Can anyone be found with an exciting life?
1. Assume: ∀x Wealthy(x) → ¬Poor(x), ∀x Stupid(x) → ¬Smart(x)
2. ∀y (Read(y) → Smart(y))
3. Read (John) ∧ ¬Poor(John)
4. ∀x Smart (x) ∧ ¬Poor(x) → Happy(x))
5. ∀z Happy (z) → Exciting(z)
6. ∃w Exciting(w)
The KB becomes:
1. ∀y ¬Read(y) ∨ Smart(y)

2. Read (John)
3. ¬Poor(John)
4. ¬Smart (x) ∨ Poor(x) ∨ Happy(x))
5. ¬Happy(z) ∨ Exciting(z)
6. ¬Exciting(w) (Negate conclusion)
7. Resolve (5) and (6): ¬Happy (a)
8. Resolve (7) and (4): ¬Smart (b) ∨ Poor(b)
9. Resolve (8) and (1): ¬Read (c) ∨ Poor(c)
10. Resolve (9) and (3): ¬Read (John)
11. Resolve (10) and (2): null

Hence anyone can be found with an exciting life.