Multi-Agent Railway Scheduling via Expert-Guided Maac with Attention-Based Critic

SSN College of Engineering, Kalavakkam

Department of Computer Science and Engineering

IV Semester – B.E (CSE)

R2024

UCS3461 - Foundations of Artificial Intelligence (TCP-Lab)

Academic Year: 2025-2026 Even **Batch: 2024-2028**

Exercise 4: Propositional Logic: Proof by Resolution using the Refutation Method (CO3, K4, PIs: 1.3.1,1.4.1,2.1.3,12.3.2)

Given Rules and Facts

Let A_i, B_i, and $D_i$be propositional symbols. The knowledge base consists of the following rules and facts:

R1: $\neg A_1$
R2: $\neg D_1$
R3: $\neg B_1$
R4: B_2

R5: $B_1 \leftrightarrow (A_2 \vee A_3)$
R6: $B_2 \leftrightarrow (A_1 \vee A_4 \vee A_5)$
R7: $D_1 \rightarrow B_1$
R8: $D_2 \rightarrow B_2$
R9: $A_2 \rightarrow D_1$

Query: $\neg A_2$

Using the above rules, develop a Python program to prove the query $\neg A_2$ from the given knowledge base using resolution by refutation method.

Implement a class named **ResolutionProver** with the following methods:

_init__(self) - to initialize the knowledge base
add_rule(self, rule: str) - to add rules to the knowledge base
to_cnf(self, rule: str) - to convert each rule into Conjunctive Normal Form (CNF)
resolve() - to apply the resolution rule between two clauses
resolution_procedure() - to repeatedly apply resolution until either an empty clause is derived or no new clauses can be generated.

The program should represent each clause as a set of literals, first convert all rules into CNF, add the negation of the query to the clause set, apply the resolution procedure, and finally report whether the query is logically entailed by the knowledge base. The program should support propositional logic operators AND (∧), OR (∨), NEGATION (¬), IMPLICATION (→), and BICONDITIONAL (↔).