ASSIGNMENT-3

RESOLUTION-REFUTATION AS SEARCH

Drithi Davuluri
B21AI055

1) Experimental Details:

The main question addressed by this experiment is "Does the given Knowledge Base entail the provided query?"

a) Input Specifications: The input format consists of two integers, 'n' and 'm', representing the number of formulas in the knowledge base (KB) and the mode parameter, respectively followed by n formulas and one query. M is used to decide if the resolution steps are to be seen or not.

b) CNF Conversion **(Part-0)**
Parsing and Preprocessing: Formulas are parsed into a format suitable for resolution-refutation. Parsing involves handling logical operators (AND, OR, IMPLICATION, IFF), parentheses, and negation. The parsed formulas are then converted into CNF. Formulas are converted into Conjunctive Normal Form (CNF) using the sympy library.

c) Reformation of KB:
Then the KB was reformed by separating the formula at &. This results in the same KB as all the formulas in KB are joined using & only. This is done in the following way- If KB : (A|B)&(C|D) then the KB will be now 1.(A|B) 2. (C|D). Then after further rearrangements we reached an array of KB as follows- [[A,B],[C,D]]
Then the negation of query is added to this KB.

d) Uninformed Search (Uniform) Resolution **(Part-A)**
Resolution Algorithm:
For each pair of clauses in the knowledge base, attempted resolution using the is_resolvent function.
Is_resolvent - For each literal in one clause, check if its negation is present in the other clause. If a negation is found, resolve the literals by removing

both the positive and negated literals. Form the resolvent by combining the remaining literals from both clauses.

If the is_resolvent function returns a non-empty resolvent, append it to the knowledge base. If the resolvent is an empty clause, return 1 to indicate that the query is entailed by the knowledge base. After each iteration, remove duplicate literals within clauses to maintain the correctness of the resolution process. If the maximum number of iterations is reached without deriving a contradiction, return 0 to indicate that the query is not entailed.

e) Greedy Search Resolution **(Part-B)**:

The heuristic function used in the greedy search strategy determines the priority of resolution based on the number of literals in a clause. The lower the number of literals, the higher the priority. This heuristic aims to resolve pairs that result in smaller resolvents first. The correctness of heuristic is justified-

Fewer Clauses Tend to Be More Informative: Clauses with fewer literals tend to be more informative as they express more specific constraints. Resolving pairs with fewer literals may lead to more significant deductions.

Smaller Resolvents May Lead to Faster Convergence: Resolving clauses with fewer literals can result in smaller resolvents. Smaller resolvents may lead to faster convergence towards contradictions, as they represent more direct and specific conflicts.

Potential for Early Detection of Contradictions: Prioritizing pairs with fewer literals increases the chances of early detection of contradictions. If a contradiction exists, resolving pairs with fewer literals may reveal it sooner in the resolution process.

Reduction of Search Space: Resolving smaller clauses first might help in reducing the overall search space by quickly eliminating certain possibilities. This reduction in the search space can contribute to the efficiency of the algorithm

Then for the resolution prioritize the pairs based on a heuristic function (heuristic), sorting the knowledge base to resolve pairs with fewer literals first Then do the same process as in uniform search.

f) Output:

The program outputs the result of the resolution-refutation algorithm using the search strategy used.

Optionally, it can print the resolution steps if the mode parameter is set based on the input m.

2) Results:

When m=0, the algorithm prints 1 if the query entails and it also prints the number of nodes traversed. The time of execution is also printed for analysis. When m=1 along with the above printed the steps of resolution are also printed. Tested the code on 8 different test cases each for uniform search and greedy search.

KB	**Query**	**Result**	**Uniform**	**Greedy**
P>Q !Q	Q	0	Time:1.2602	Time: 1.2865
(P&Q)>R (S&T)>Q S T P	R	1	Node: 515 Time: 0.0097	Nodes: 190 Time: 0.0028
A=(B|C) !A	!B	1	Node:38 Time:0.000999	Node:14 Time:0.00097
P|Q P>R Q>R	R	1	Node:98 Time:0.0030	Node:42 Time:0.0018
P !P	R	1	Node: 1 Time: 0.000998	Node:1 Time: 0.001
!A|B !C	!A	0	Node: - Time: 0.5813	Node: - Time: 0.566
(A&B)|C C E	!E	0	Node: - Time: 0.0069	Node: - Time: 0.0050
P&(Q>R) P>T Q>(R=T)	R	0	Node: - Time: 4.933	Node: - Time: 4.862
P|(Q&(R>T)) P>R Q>T Q>(R=T)	R	1	Node: 255 Time: 0.0055	Node: 186 Time: 0.0024

3) Observation

Query Entailment:

The "Result" column indicates whether the query is entailed by the knowledge base (KB) or not. A result of 1 indicates entailment, while 0 indicates non-entailment.

Comparison of Uniform and Greedy Search:

The "Uniform" and "Greedy" columns provide information on the performance of the resolution-refutation algorithm using uniform and greedy search strategies, respectively.

4) Analysis

Time Complexity:

The "Time" column represents the time taken for each resolution attempt. Greedy search tends to have slightly higher time complexity in some cases. For example, in the case of (A&B)|C, the uniform search takes less time compared to the greedy search.

Node Exploration:

The "Nodes" column represents the number of nodes explored during the resolution process. In most cases, the greedy search explores fewer nodes compared to the uniform search. This reduction in the number of explored nodes is beneficial for efficiency.

Effectiveness of Greedy Heuristic:

The experiment demonstrates that the heuristic used in the greedy search strategy, which prioritizes pairs with fewer literals, is effective in certain cases. For instance, in the case of P|(Q&(R>T)), the greedy search explores fewer nodes compared to the uniform search, indicating the heuristic's success in guiding the search towards faster convergence.