ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

Lab2

Môn : Cơ sở trí tuệ nhân tạo

Họ và tên: Nguyễn Vũ Hiếu

MSSV:20120478

Thành phố Hồ Chí Minh-2023

1. **Ưu và nhược điểm của phương pháp phân giải (Resolution)**
    a. Ưu điểm
    -Toàn vẹn :Khi áp dụng cho các tập hợp hữu hạn các mệnh đề, phương pháp phân giải là toàn vẹn, có thể chứng minh hoặc bác bỏ bất kỳ câu lệnh nào được suy luận logic nếu có đủ thời gian.
    -Độ chính xác : Phương pháp này đảm bảo tính logic của mọi kết luận được rút ra từ quy tắc phân giải, dựa trên các giả thiết ban đầu.
    -Áp dụng rộng rãi: Đây là một phương pháp rất linh hoạt, có thể tự động hóa và triển khai trong phần mềm, phù hợp cho nhiều ứng dụng giải quyết vấn đề khác nhau.
    b. Nhược điểm
    -Số lượng mệnh đề tăng mạnh: Khi số lượng mệnh đề tăng lên, không gian tìm kiếm mở rộng, dẫn đến vấn đề "nổ quy mô" (combinatorial explosion). Điều này làm cho phương pháp trở nên không hiệu quả với các vấn đề lớn hơn.
    -Không hiệu quả trong một số trường hợp: Ở một số trường hợp, phương pháp phân giải có thể không kết thúc hoặc mất thời gian không thực tế để tìm ra một giải pháp.

2. **Giải pháp**
    -Heuristics: Áp dụng các chiến lược thông minh để hướng dẫn quá trình phân giải. Điều này có thể bao gồm ưu tiên xử lý các mệnh đề cụ thể hơn dựa trên khả năng dẫn đến kết quả phân giải nhanh hơn.
    -Kỹ thuật Giảm mệnh đề: Phát triển các phương pháp giảm số lượng mệnh đề trong khi vẫn giữ nguyên tính đúng đắn logic. Có thể bao gồm các kỹ thuật đơn giản hóa hoặc xác định và loại bỏ các mệnh đề dư thừa hoặc không cần thiết.
    -Xử lý Song Song: Sử dụng các kỹ thuật tính toán song song để xử lý đồng thời các nhánh của không gian tìm kiếm, có thể giảm thời gian cần để tìm ra một giải pháp.

3. **Source code**
    -Giải thích cách phân giải và các hàm quan trọng

A. Hàm pl_resolution(alpha, KB):
    i. Khởi tạo các biến và danh sách:
        - new_clauses_list: Danh sách để lưu trữ các mệnh đề mới phát sinh trong quá trình giải quyết.
        - solution: Biến đánh dấu xem có giải pháp được tìm thấy hay không.
        - cnf_clause_list: Danh sách các mệnh đề trong dạng chuẩn của Knowledge Base (KB), được sao chép từ KB ban đầu để không ảnh hưởng đến KB gốc.
        - neg_alpha: Biểu diễn phủ định của mệnh đề alpha.
    ii. Thêm phủ định của alpha vào KB: Mệnh đề alpha được chuyển sang dạng chuẩn CNF và sau đó được phủ định. Mệnh đề phủ định này được thêm vào KB nếu chưa tồn tại trong danh sách mệnh đề dạng chuẩn.

iii. Vòng lặp chính: Hàm sử dụng một vòng lặp vô hạn while True để tiến hành quá trình phân giải. Mỗi lần lặp, một danh sách mới (new_clauses_list) được thêm vào để lưu trữ các mệnh đề mới phát sinh.

iv. Quá trình phân giải: Với mỗi cặp mệnh đề trong cnf_clause_list, hàm resolve được sử dụng để tìm resolvent (kết quả phân giải). Nếu resolvent là mệnh đề rỗng ([]), điều này ngụ ý rằng đã tìm thấy giải pháp. Quá trình dừng lại và solution được đặt thành True. Nếu resolvent không hợp lệ và chưa tồn tại trong cnf_clause_list hoặc new_clauses_list, nó sẽ được thêm vào new_clauses_list.

v. Kiểm tra điều kiện dừng: Nếu không có mệnh đề mới được thêm vào new_clauses_list (tức là không có thay đổi), quá trình dừng lại. Nếu new_clauses_list cuối cùng trở thành trống, điều này ngụ ý rằng không thể tìm thấy giải pháp và solution được đặt thành False.

vi. Mở rộng KB và lặp lại quá trình: Nếu có mệnh đề mới được tạo ra, chúng sẽ được thêm vào cnf_clause_list để mở rộng Knowledge Base. Quá trình sẽ tiếp tục với các mệnh đề mới này, cố gắng tìm ra một giải pháp hoặc chứng minh.

B. Hàm resolve(clause_1: list, clause_2: list):

i. Lặp qua từng literal trong hai mệnh đề: Hàm sử dụng hai vòng lặp for để duyệt qua từng literal trong clause_1 và clause_2.

ii. Kiểm tra tính bổ sung của hai literals: Mỗi literal trong clause_1 sẽ được kiểm tra với tất cả các literals trong clause_2. Nếu hai literals là bổ sung của nhau (complementary literals), nghĩa là một là dạng phủ định của một cái kia, ví dụ: p và ¬p. Trong trường hợp này, chúng sẽ được sử dụng để tạo ra một resolvent mới.

iii. Tạo resolvent từ các literals tương thích: Khi tìm thấy các literals bổ sung, chúng sẽ được loại bỏ khỏi mệnh đề gốc. Resolvent mới sẽ được tạo ra bằng cách kết hợp các phần còn lại của clause_1 và clause_2. Resolvent này sau đó được thêm vào danh sách resolvents sau khi được chuyển đổi thành dạng chuẩn bằng cách sử dụng hàm standard_clause.

iv. Trả về danh sách resolvent: Kết quả cuối cùng của hàm là một danh sách các resolvent được tạo ra từ việc phân giải giữa clause_1 và clause_2.

4. Test case

- 5 test case được đính kèm trong thư mục Project02_logic - SRC/INPUT

❖ Tài liệu tham khảo

https://github.com/Puravnisar/Logic-Programming-Resolution
https://www.geeksforgeeks.org/method-resolution-order-in-python-inheritance/
Artificial Intelligence: A Modern Approach, Third Edition, Chapter 7, Figure 7.12