ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA: CÔNG NGHỆ THÔNG TIN

BÁO CÁO LAB 2 – LOGIC VÀ MỆNH ĐỀ
MÔN: CƠ SỞ TRÍ TUỆ NHÂN TẠO

Sinh viên thực hiện: Nguyễn Hải Đăng
Mã số sinh viên: 20120049

Giảng viên lý thuyết: GS. TS. Lê Hoài Bắc
Giảng viên thực hành: Nguyễn Ngọc Đức
Lớp lý thuyết: 20_1
Học kỳ - Niên khoá: HK2 - 2022-2023

Thành phố Hồ Chí Minh, ngày 16 tháng 04 năm 2023

MỤC LỤC

I. HỢP GIẢI

Hệ thống hợp giải là một hệ thống chứng minh đúng và đủ đối với bài toán suy dẫn. Hệ thống này dựa trên một luật duy nhất được giỏi là luật hợp giải được phát biểu như sau: cho các mệnh đề α, β, γ, khi đó:

$$\frac{\alpha \vee \neg\beta \quad \beta \vee \gamma}{\alpha \vee \gamma}$$

Hay $(\alpha \vee \neg\beta) \wedge (\beta \vee \gamma) \Rightarrow (\alpha \vee \gamma)$

Luật hợp giải trên là đúng, Ta có thể kiểm chứng như sau: nếu như α đúng thì rõ ràng suy luận trên là đúng và luật là đúng, còn nếu α sai thì β cũng phải sai thì $\alpha \vee \neg\beta$ mới đúng, từ đó γ phải đúng để $\beta \vee \gamma$ đúng, từ đó kết luận đúng vì γ đúng.

Có thể thấy, một hệ thống chỉ cần sử dụng một mình luật hợp giải cũng là đủ để chứng minh một bài toán suy dẫn. Tuy nhiên, để áp dụng được phương pháp hợp giải, cơ sở tri thức cần được biến đổi đưa về dạng chuẩn hội.

II. DẠNG HỘI CHUẨN

Mọi logic mệnh đề đều tương đương logic với một phép nối liền các phép hội của từ. Một câu logic được biểu diễn là nối liền các phép hội của từ được gọi là **dạng hội chuẩn** (Conjunctive Normal Form) hay CNF.

Ví dụ câu sau đây là dạng hội chuẩn CNF:

$$A \vee (B \vee \neg C) \wedge (C \vee D) \wedge \neg E$$

Trong câu trên, mỗi từ là một ký hiệu mệnh đề hay phủ định của ký hiệu mệnh đề. Các phép hội được thực hiện trên các từ và cuối cùng được nối với nhau bằng phép nối liền.

Ta có thể mô tả quá trình biến đổi một mệnh đề về dạng hội chuẩn như sau:

- Bước 1: Loại bỏ $\Leftrightarrow$, thay $A \Leftrightarrow B$ bằng $(A \Rightarrow B) \wedge (B \Rightarrow A)$
- Bước 2: Loại bỏ $\Rightarrow$, thay $A \Rightarrow B$ bằng $\neg A \vee B$.
- Bước 3: CNF đòi hỏi dấu $\neg$ chỉ được xuất hiện trên các từ, do đó cần phân phối dấu phủ định $\neg$ vào trong các từ:
    - $\neg(\neg A) \equiv A$
    - $\neg(A \wedge B) \equiv (\neg A \vee \neg B)$ (luật De Morgan)
    - $\neg(A \vee B) \equiv (\neg A \wedge \neg B)$ (luật De Morgan)
- Bước 4: Áp dụng luật phân phối $\vee$ vào $\wedge$ cho tất cả trường hợp có thể.

Một số luật biến đổi logic có thể sử dụng:

Basic statement	Equivalent
$p \vee q$	$q \vee p$
$p \wedge q$	$q \wedge p$
$\neg(p \wedge q)$	$\neg p \vee \neg q$
$\neg(p \vee q)$	$\neg p \wedge \neg q$
$p \rightarrow q$	$\neg p \vee q$
	$\neg q \rightarrow \neg p$
$p \leftrightarrow q$	$(p \rightarrow q) \wedge (q \rightarrow p)$
	$(\neg p \vee q) \wedge (\neg q \vee p)$
$p \wedge (q \wedge r)$	$(p \wedge q) \wedge r$
$p \vee (q \vee r)$	$(p \vee q) \vee r$
$p \wedge (q \vee r)$	$(p \wedge q) \vee (p \wedge r)$
$p \vee (q \wedge r)$	$(p \vee q) \wedge (p \vee r)$
$p \rightarrow (q \vee r)$	$(p \wedge \neg q) \rightarrow r$

III. THUẬT TOÁN HỢP GIẢI

1. *Ý tưởng và mã giả của thuật toán hợp giải*

- Thủ tục suy diễn dựa trên hợp giải hoạt động dựa trên nguyên tắc chứng minh phản chứng. Nghĩa là, để chứng minh câu $KB \vDash \alpha$ (KB suy dẫn hay khéo theo α), ta chứng minh phủ định của nó $(KB \wedge \neg\alpha)$ không thỏa. Việc này được chứng minh bằng cách chỉ ra một mâu thuẫn bên trong các mệnh đề.
- Đầu tiên, $(KB \wedge \neg\alpha)$ được biến đổi về dạng CNF. Sau đó, luật hợp giải được áp dụng lên các mệnh đề kết quả. Mỗi cặp mệnh đề chứa các từ bù nhau được hợp giải tạo ra mệnh đề mới và được thêm vào tập câu nếu nó chưa xuất hiện. Quá trình này sẽ tiếp tục cho đến khi một trong hai điều sau xảy ra:
  - Không có mệnh đề mới nào có thể được thêm vào, trong trường hợp đó KB không suy dẫn được α.
  - Hai mệnh đề hợp giải nhận được một mệnh đề rỗng, KB suy dẫn được α.
- Đây là mã giả của thuật toán hợp giải:

```
function PL-RESOLUTION(KB, α) returns true or false
  inputs: KB, the knowledge base, a sentence in propositional logic
          α, the query, a sentence in propositional logic

  clauses ← the set of clauses in the CNF representation of KB ∧ ¬α
  new ← { }
  while true do
    for each pair of clauses C_i, C_j in clauses do
      resolvents ← PL-RESOLVE(C_i, C_j)
      if resolvents contains the empty clause then return true
      new ← new ∪ resolvents
    if new ⊆ clauses then return false
    clauses ← clauses ∪ new
```

Figure 7.13 A simple resolution algorithm for propositional logic. PL-RESOLVE returns the set of all possible clauses obtained by resolving its two inputs.

2. *Ưu điểm, nhược điểm của giải thuật hợp giải và giải pháp khắc phục vấn đề*

- Ưu điểm:

    - Độ hoàn thiện cao: Nếu một tập hợp các mệnh đề không thỏa mãn, thì kết thúc quá trình hợp giải của các mệnh đề đó chứa mệnh đề trống, đảm bảo tính hoàn thiện của kết quả.
    - Độ chính xác cao: Kết quả trả về luôn là hệ quả được suy ra từ cơ sở tri thức ban đầu, đảm bảo tính đúng đắn của kết quả.
    - Là phương pháp đơn giản và dễ hiểu, phù hợp với các vấn đề logic mệnh đề cơ bản.
    - Được sử dụng rộng rãi trong lĩnh vực trí tuệ nhân tạo và hình thành cơ sở cho nhiều phương pháp giải quyết vấn đề khác.
    - Có thể được kết hợp với các phương pháp khác để tăng tốc độ giải quyết vấn đề.
- Nhược điểm:
    - Mỗi cặp mệnh đề hợp giải với nhau có thể tạo ra nhiều kết quả và các kết quả này không đóng góp vào quá trình chứng minh đúng. Điều này có thể dẫn đến việc phải xử lý một lượng lớn các kết quả không cần thiết, gây lãng phí thời gian và tài nguyên tính toán.
    - Hệ thống đòi hỏi vét cạn sẽ duyệt qua hết các trường hợp hợp giải, điều này có thể gây mất nhiều thời gian và không khả thi đối với các vấn đề có số lượng biến và mệnh đề lớn.
    - Giải thuật hợp giải có thể trở nên rất chậm và tốn nhiều tài nguyên tính toán khi áp dụng cho các vấn đề lớn với số lượng biến và mệnh đề lớn.
    - Có thể không thể giải quyết được các vấn đề với các mệnh đề phức tạp hoặc có tính khó xác định cao.
    - Có thể dẫn đến nhiều kết quả trùng lặp hoặc không cần thiết, dẫn đến lãng phí tài nguyên và thời gian tính toán.
- Giải pháp khắc phục:
    - Phương pháp Davis-Putnam:

      Thuật toán giải quyết SAT (Satisfiability) ban đầu của Davis và Putnam, hiện được gọi là thủ tục Davis-Putnam, có thể nhanh hơn nhiều so với việc chỉ "thử tất cả". Thủ tục này dựa trên việc chúng ta thường có thể xác định giá trị đúng/sai của một mệnh đề hoặc công thức mà không cần biết tất cả các giá trị của các biến của nó, vì một đối số đúng đơn lẻ sẽ làm cho mệnh đề đó là đúng và một mệnh đề sai đơn lẻ sẽ làm cho công thức là sai. Dưới đây là mã giả của thuật toán,

Algorithm 1 Davis-Putnam Procedure

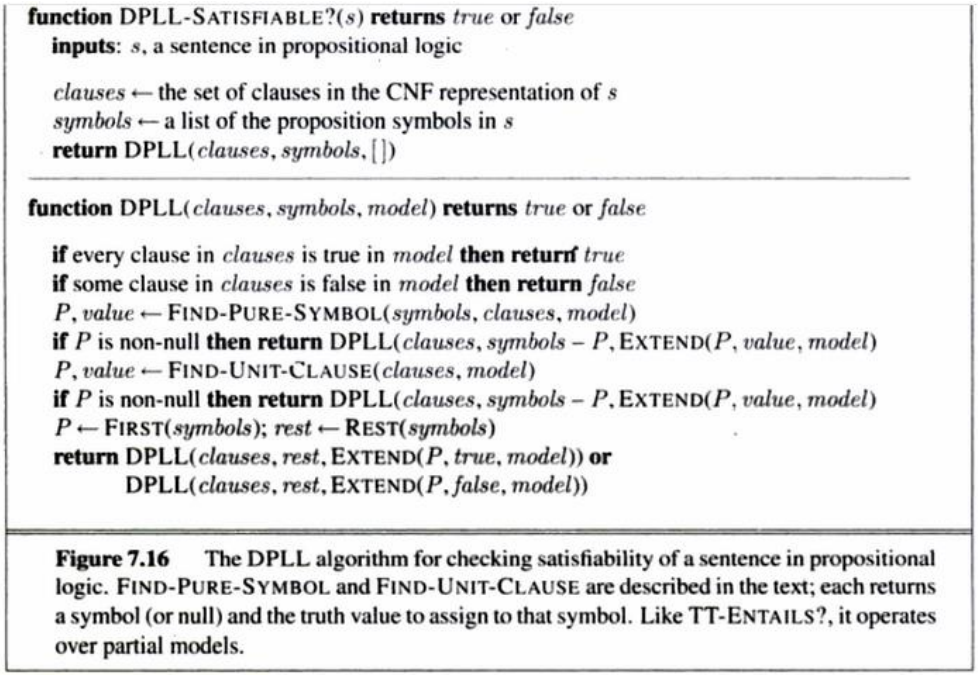
```
1: procedure DAVIS-PUTNAM(φ)
2:     S ← {(φ, [])}                ▷ Initially, S contains only a pairing of φ with the empty truth assignment.
3:     while S is not empty do
4:         (φ′, t) ← an arbitrary element of S
5:         x ← an arbitrary literal in φ′
6:         S ← (S − {(φ′, t)}) ∪ CHASE(φ, x, t) ∪ CHASE(φ, x̄, t)
7:     end while
8: end procedure

9: function CHASE(φ, x, t)
10:    Set x to true in t
11:    Delete all clauses from φ that contain the literal x
12:    Delete the literal x̄ from all clauses in φ
13:    if φ is empty then abort from DAVIS-PUTNAM with t        ▷ t is a satisfying assignment
14:    elseif φ contains an empty clause then return {}         ▷ t contains bad decisions
15:    elseif φ contains a unit clause (y) then return CHASE(φ, y, t)   ▷ Continue the chase
16:    else return {(φ, t)}
17: end function
```

- Phương pháp DPLL:

  Thuật toán này là phiên bản được mô tả bởi Davis, Logemann, Hilary Putnam (1960) và Loveland (1962), vì vậy nó được gọi là DPLL theo tên viết tắt của cả bốn tác giả. DPLL lấy đầu vào là một câu ở dạng thông thường liên kết—một tập hợp các mệnh đề. Giống như BACKTRACKING-SEARCH và TT-ENTAILS, nó thực chất là một phép liệt kê đệ quy, theo chiều sâu đầu tiên của các mô hình có thể. Nó thể hiện ba cải tiến so với sơ đồ đơn giản của TT-ENTAILS:

  - Chấm dứt sớm: Thuật toán phát hiện xem câu phải đúng hay sai, ngay cả với mô hình đã hoàn thành một phần. Một mệnh đề là đúng nếu bất kỳ literal nào cũng đúng, ngay cả khi các literal khác chưa có chân trị. Do đó, toàn bộ suy luận có thể được đánh giá là đúng ngay cả trước khi mô hình hoàn thành.
  - Pure symbol heuristic: ký hiện thuần túy (pure symbol) là một ký hiệu luôn xuất hiện với cùng "một dấu" trong tất cả mệnh đề.
  - Mệnh đề đơn vị heuristic: Mệnh đề đơn vị được định nghĩa trước đó là một mệnh đề chỉ có một đối số. Trong ngữ cảnh của DPLL, nó cũng có nghĩa là các mệnh đề trong đó tất cả các đối số ngoại trừ một trong số chúng đã được gán FALSE bởi mô hình.



```
function DPLL-SATISFIABLE?(s) returns true or false
  inputs: s, a sentence in propositional logic

  clauses ← the set of clauses in the CNF representation of s
  symbols ← a list of the proposition symbols in s
  return DPLL(clauses, symbols, [])

function DPLL(clauses, symbols, model) returns true or false

  if every clause in clauses is true in model then return true
  if some clause in clauses is false in model then return false
  P, value ← FIND-PURE-SYMBOL(symbols, clauses, model)
  if P is non-null then return DPLL(clauses, symbols – P, EXTEND(P, value, model)
  P, value ← FIND-UNIT-CLAUSE(clauses, model)
  if P is non-null then return DPLL(clauses, symbols – P, EXTEND(P, value, model)
  P ← FIRST(symbols); rest ← REST(symbols)
  return DPLL(clauses, rest, EXTEND(P, true, model)) or
         DPLL(clauses, rest, EXTEND(P, false, model))
```

Figure 7.16 The DPLL algorithm for checking satisfiability of a sentence in propositional logic. FIND-PURE-SYMBOL and FIND-UNIT-CLAUSE are described in the text; each returns a symbol (or null) and the truth value to assign to that symbol. Like TT-ENTAILS?, it operates over partial models.

3. Kịch bản kiểm thử

- Test-case 1:

Input	Output	Ghi chú
-P OR -U	5	
4	-P OR R	(-P OR Q) hợp giải với (-Q OR R)
-P OR Q	Q	(-P OR Q) hợp giải với (P)
-Q OR R	-Q OR S	(-Q OR R) hợp giải với (Q OR S)
-R OR S	-R OR -U	(-R OR S) hợp giải với (-U OR -S)
-U OR -S	-S	(-U OR -S) hợp giải với (U)
	7	
	-P OR S	(-P OR Q) hợp giải với (-Q OR S)
	R	(-Q OR R) hợp giải với (Q)
	-Q OR -U	(-Q OR R) hợp giải với (-R OR -U)
	-R	(-R OR S) hợp giải với (-S)
	-P OR -U	(-P OR R) hợp giải với (-R OR -U)
	S	(Q) hợp giải với (-Q OR S)
	-Q	(-Q OR S) hợp giải với (-S)
	3	
	-U	(-U OR -S) hợp giải với (S)
	-P	(U) hợp giải với (-P OR -U)
	{}	(Q) hợp giải với (-Q)
	YES	

- Test-case 2:

Input	Output	Ghi chú
C	3	

4	-B OR C	(-A OR -B OR C) hợp giải với (A)
-A OR -B OR C	-A OR C	(-A OR -B OR C) hợp giải với (B)
-B OR -C	-A OR -B	(-A OR -B OR C) hợp giải (-C)
A	3	
B	-B	(-B OR -C) hợp giải với (-B OR C)
	C	(A) hợp giải với (-A OR C)
	-A	(B) hợp giải với (-A OR -B)
	1	
	{}	(-C) hợp giải với (C)
	YES	

- Test-case 3:

Input	Output	Ghi chú
R OR S	4	
3	Q OR R	(-P OR R) hợp giải với (P OR Q)
-P OR R	-P	(-P OR R) hợp giải với (-R)
P OR Q	P OR S	(P OR Q) hợp giải với (-Q OR S)
-Q OR S	-Q	(-Q OR S) hợp giải với (-S)
	5	
	R OR S	(-P OR R) hợp giải với (P OR S)
	P	(P OR Q) hợp giải với (-Q)
	Q	(-R) hợp giải với (Q OR R)
	R	(Q OR R) hợp giải với (-Q)
	S	(-P) hợp giải với (P OR S)
	1	
	{}	(-R) hợp giải với (R)
	YES	

- 'Test-case 4:

Input	Output	Ghi chú
S OR -T	3	
3	Q OR -R	(P OR Q) hợp giải với (-P OR -R)
P OR Q	-P OR -T	(-P OR -R) hợp giải với (-T OR R)
-P OR -R	R	(-T OR R) hợp giải với (T)
-T OR R	3	
-S	Q OR -T	(P OR Q) hợp giải với (-P OR -T)
T	-P	(-P OR -R) hợp giải với (R)
	Q	(Q OR -R) hợp giải với (R)
	0	
	NO	

- Test-case 5:

Input	Output	Ghi chú

-A	6	
5	-A OR -D	(-A OR C) hợp giải với (-C OR -D)
-A OR C	C OR D	(-A OR C) hợp giải với (A OR D)
-B OR D	C	(-A OR C) hợp giải với (A)
B OR C	-B OR -C	(-B OR D) hợp giải với (-C OR -D)
-C OR -D	B OR -D	(B OR C) hợp giải với (-C OR -D)
A OR D	A OR -C	(-C OR -D) hợp giải với (A OR D)
	4	
	-A OR -B	(-A OR C) hợp giải với (-B OR -C)
	-D	(-C OR -D) hợp giải với (C)
	A OR B	(A OR D) hợp giải với (B OR -D)
	-B	(C) hợp giải với (-B OR -C)
	0	
	NO	

IV. TIÊU CHÍ ĐÁNH GIÁ

Tiêu chí	Điểm tự đánh giá
Read input data and store in suitable data structure (0.5)	0.5
Implementation of resolution method (1.0)	1.0
Inference process and results (2.5)	2.5
Testcases, report, evaluations (1.0)	0.5

V. TÀI LIỆU THAM KHẢO

- CS270 Combinatorial Algorithms & Data Structures Spring 2006 - Lecture 1: 1.17.06 - Lecturer: Satish - Scribe: Jason Wolfe
- Artificial Intelligence A Modern Approach (4th Edition)
- Logic.py of arizona.edu
- DPLL algorithm - Wikipedia
- Giáo trình Cơ sở Trí tuệ nhân tạo – Lê Hoài Bắc, Tô Hoài Việt (2014)