AUTOMATA, FORMAL LANGUAGES AND LOGIC

Dr Pooja Agarwal

Professor,

Department of Computer Science & Engineering

AUTOMATA, FORMAL LANGUAGES AND LOGIC

MODULE 5

Propositional Logic

Dr Pooja Agarwal

Department of Computer Science & Engineering

Outline

- Proof by resolution
  - Unit resolution
  - Conjunctive normal form
  - Resolution algorithm

Proof by Resolution

- **Proof by resolution**
  - Resolution yield completeness for the inference algorithms when coupled with any complete search algorithm.

Proof by Resolution

Literal

- A literal is an assignment of a value to a Boolean variable

Axiom

- An assumption or statement that is assumed to be true.

Clause

- A clause is satisfied or true in a possible world if and only if at least one of the literals that makes up the clause is true in that possible world.
- A clause is a set of literals considered to be an implicit disjunction.
- A clause with exactly one literal is known as **unit clause**, it can be a positive or a negative unit clause.

Proof by Resolution

Clashing Clauses

- Let C1 and C2 be two clauses and L and L’ are two literals. where L Ɛ C1 and L’ Ɛ C2 and L’ is complement of L, then C1 and C2 are said to be clashing clauses and clash on complementary literals L and L’. The resultant clause C is given by

$$Res(C1, C2) = (C1-\{L\}) \vee (C2-\{L'\})$$

Proof by Resolution

Resolution

- Resolution is a simple iterative process where two clauses are resolved yielding a new clause that has been inferred from them.

Example

- Playing Tennis or Raining
- Not Raining or working

Proof by Resolution

Resolution

- Resolution is a simple iterative process where two clauses are resolved yielding a new clause that has been inferred from them.

Example

- Playing Tennis **(P)** **or** Raining**(Q)**
- Not Raining or working

Proof by Resolution

Resolution

- Resolution is a simple iterative process where two clauses are resolved yielding a new clause that has been inferred from them.

Example

- Playing Tennis **(P) or** Raining**(Q)**
- Not Raining (¬Q) **or** working **(R)**

Two statements:-

P V Q ¬Q V R

P V R

AUTOMATA FORMAL LANGUAGES AND LOGIC

Existing Knowledge Base

R_1: $\neg P_{1,1}$

R_2: $B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$

R_3: $B_{2,1} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{3,1})$

R4: $\neg B_{1,1}$

R5: $B_{2,1}$

$R_6 : (B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$ by bicond. elim R_2

$R_7 : ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})$ by And-Elimination to R_6

$R_8 : (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \lor P_{2,1}))$ by contrapositives

$R_9 : \neg(P_{1,2} \lor P_{2,1})$ by Modus Ponens with R_8 and R_4

R10 : ¬ P1,2 ∧ ¬P2,1 by De Morgan’s rule

Proof by Resolution

- Let's say agent returns from [2,1] to [1,1] and goes to [1,2]

- We add:

R11 : $\neg B1,2$

R12 : $B1,2 \Leftrightarrow (P1,1 \lor P2,2 \lor P1,3)$

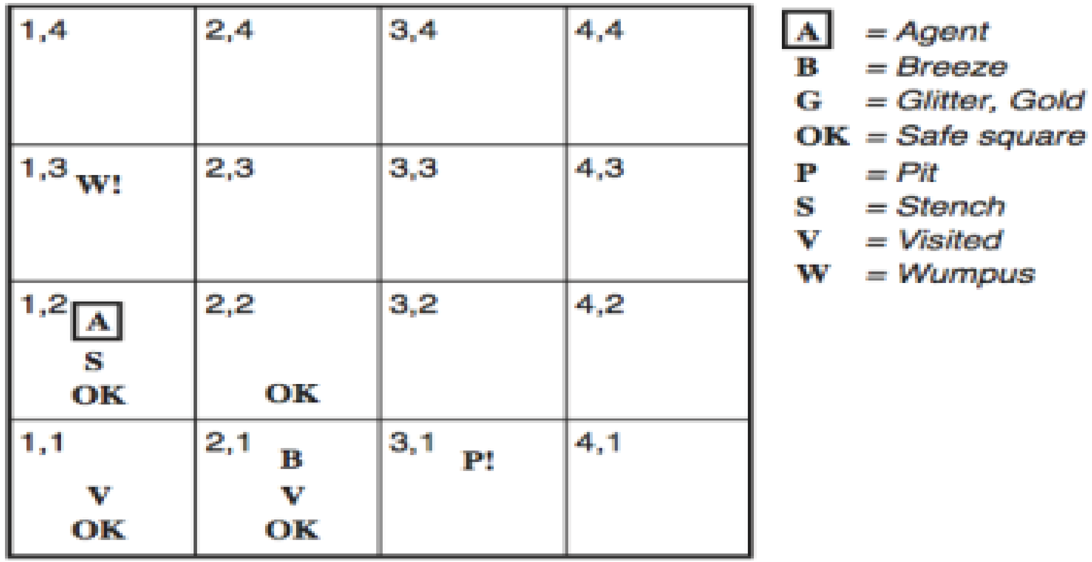
R13 : $\neg P2,2$

R14 : $\neg P1,3$

R15: $P1,1 \lor P2,2 \lor P3,1$

R16: $P1,1 \lor P3,1$

R17: $P3,1$



R_3: $B_{2,1} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{3,1})$

$B_{2,1} \Rightarrow (P_{1,1} \lor P_{2,2} \lor P_{3,1}) \land (P_{1,1} \lor P_{2,2} \lor P_{3,1}) \Rightarrow B_{2,1}$

R5: $B_{2,1}$

$$\frac{\alpha \Rightarrow \beta,\ \alpha}{\beta}$$

Proof by Resolution

Factoring

One more technical aspect of resolution rule : the resulting clause should contain only one copy of each literal.

The removal of multiple copies of literals is called **Factoring**.

Example:- If we resolve
$A \vee B$ with **$A \vee \neg B$,** we obtain

$A \vee A$

Which is reduced to just 'A'.

THANK YOU

Dr Pooja Agarwal
Department of Computer Science and Engineering
poojaagarwal@pes.edu