جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

كلية العلـــوم
قـســـــــم علوم الذكاء الاصطناعي

المحاضرة الرابعة

المادة : *Knowledge representation*
المرحلة : الأولى
اسم الاستاذ: م.م. عبدالله فاضل شبر

Resolution Theorem Proving

Resolution is a technique for proving theorems in the predicate logic using the ***resolution by refutation algorithm***. The resolution refutation proof procedure answers a query or deduces a new result by reducing the set of clauses to a contradiction.

The **Resolution by Refutation Algorithm** includes the following steps:-

a) Convert the statements to **predicate calculus** (predicate logic).

b) Convert the statements from **predicate calculus** to **clause form.**

c) Add the negation of what is to be proved to the clause form.

d) Resolve the clauses to producing new clauses and producing a contradiction by generating the empty clause.

Clause Form

The statements that produced from **predicate calculus** method are nested and very complex to understand, so this will lead to more complexity in resolution stage, therefore the following steps are used to convert the **predicate calculus** to **clause form**:-

ملاحظة/ يتم تطبيق الخطوات الخاصة Clause form (9 خطوات) بالتسلسل لكون الخطوة اللاحقة تعتمد على نتيجة الخطوة السابقة.

1. **Eliminate (→)** by replacing each instance of the form **(P → Q)** by expression **(┐P∨Q)**

2. **Reduce the scope of negation.**

- ┐(┐a) ≡ a
- ┐(∀X) b(X) ≡ ∃X ┐ b(X)
- ┐(∃X) b(X) ≡ ∀X ┐ b(X)
- ┐(a∧b) ≡ ┐a ∨ ┐b
- ┐(a∨b) ≡ ┐a ∧ ┐b

3. **Standardize variables:** rename all variables so that each quantifier has its own unique variable name. ***For example***,
   - $\forall X\ a(X) \vee \forall X\ b(X) \equiv \forall X\ a(X) \vee \forall Y\ b(Y)$

4. **Move all quantifiers to the left** without changing their order. ***For example***,
   - $\forall X\ a(X) \vee \forall Y\ b(Y) \equiv \forall X\ \forall Y\ a(X) \vee b(Y)$

5. **Eliminate existential quantification** by using the equivalent function. ***For example***,
   - $\forall X\ \exists Y\ mother(X,Y) \equiv \forall X\ mother(X,m(X))$
   - $\forall X\ \forall Y\ \exists Z\ p(X,Y,Z) \equiv \forall X\ \forall Y\ p(X,Y, f(X,Y))$

6. **Remove universal quantification** symbols. ***For example***,
   - $\forall X\ \forall Y\ p(X,Y, f(X,Y)) \equiv p(X,Y, f(X,Y))$

7. **Use the associative and distributive properties** to get a conjunction of disjunctions called **conjunctive Normal Form (CNF)**. ***For example,***

- $a \vee (b \vee c) \equiv (a \vee b) \vee c$
- $a \wedge (b \wedge c) \equiv (a \wedge b) \wedge c$
- $a \vee (b \wedge c) \equiv (a \vee b) \wedge (a \vee c)$
- $a \wedge (b \vee c) \equiv (a \wedge b) \vee (a \wedge c)$

8. **Split each conjunct** into a separate clause. ***For example,***

- $(\neg a(X) \vee \neg b(X) \vee e(W)) \wedge (\neg b(X) \vee \neg d(X,f(X)) \vee e(W))$
  - ✓ $\neg a(X) \vee \neg b(X) \vee e(W)$
  - ✓ $\neg b(X) \vee \neg d(X,f(X)) \vee e(W)$

9. **Standardize variables** again so that each clause contains variable names that do not occur in any other clause. ***For example,***

- $(\neg a(X) \vee \neg b(X) \vee e(W)) \wedge (\neg b(X) \vee \neg d(X,f(X)) \vee e(W))$
  - ✓ $\neg a(X) \vee \neg b(X) \vee e(W)$
  - ✓ $\neg b(Y) \vee \neg d(Z,f(Z)) \vee e(V)$

Example1: ***Use the Resolution Algorithm for proving that John is happy with regard the following story:***

Everyone passing his AI exam and winning the lottery is happy. But everyone who studies or lucky can pass all his exams. John did not study but he is lucky. Everyone who is lucky wins the lottery.

Solution:

A. Convert all statements to predicate calculus.

- ∀X pass(X,ai_exam)∧win(X,lottery)→happy(X)
- ∀X study(X) ∨ lucky(X)→ ∀E pass(X,E)
- ┐study(john) ∧ lucky(john)
- ∀X lucky(X) →win(X,lottery)
- happy (john) **Required to prove it**

B. Convert the statements from predicate calculus to clause form.

$\forall X \ pass(X,ai\_exam) \land win(X,lottery) \rightarrow happy(X)$

$\forall X \ study(X) \lor lucky(X) \rightarrow \forall E \ pass(X,E)$

$\neg study(john) \land lucky(john)$

$\forall X \ lucky(X) \rightarrow win(X,lottery)$

1. Remove $(\rightarrow)$

- $\forall X \quad \neg(pass(X,ai\_exam) \land win(X,lottery)) \lor happy(X)$
- $\forall X \quad \neg(study(X) \lor lucky(X)) \lor \forall E \ pass(X,E)$
- $\neg study(john) \land lucky(john)$
- $\forall X \quad \neg(lucky(X)) \lor win(X,lottery)$

2. Reduce $\neg$

- $\forall X \quad (\neg pass(X,ai\_exam) \lor \neg win(X,lottery)) \lor happy(X)$
- $\forall X \quad (\neg study(X) \land \neg lucky(X)) \lor \forall E \ pass(X,E)$
- $\neg study(john) \land lucky(john)$
- $\forall X \quad \neg lucky(X) \lor win(X,lottery)$

3. Standardize Variables

- ∀X (¬pass(X,ai_exam)∨¬win(X,lottery))∨ happy(X)
- ∀Y (¬study(Y)∧¬lucky(Y))∨ ∀E pass(Y,E)
- ¬study(john)∧lucky(john)
- ∀Z ¬lucky(Z)∨win(Z,lottery)

4. Move all quantifiers to the left

- ∀X (¬pass(X,ai_exam)∨¬win(X,lottery))∨ happy(X)
- ∀Y ∀E (¬study(Y)∧¬lucky(Y))∨ pass(Y,E)
- ¬study(john)∧lucky(john)
- ∀Z ¬lucky(Z)∨win(Z,lottery)

5. Remove ∃

- Nothing to do here.

6. Remove ∀

- (┐pass(X,ai_exam) ∨ ┐win(X,lottery))∨ happy(X)
- (┐study(Y)∧┐lucky(Y)) ∨ pass(Y,E)
- ┐study(john) ∧ lucky(john)
- ┐lucky(Z) ∨ win(Z,lottery)

7. CNF

- ┐pass(X,ai_exam)∨┐win(X,lottery)∨ happy(X)
- (┐study(Y)∧┐lucky(Y))∨ pass(Y,E)   ≡   **(a∧b)∨c ≡ c∨(a∧b)**

The second statement becomes:

pass(Y,E)∨┐study(Y)∧pass(Y,E)∨┐lucky(Y)

- ┐study(john)∧lucky(john)
- ┐lucky(Z)∨win(Z,lottery)

8. Split ∧

- ┐pass(X,ai_exam)∨┐win(X,lottery)∨ happy(X)
- pass(Y,E)∨┐study(Y)
- pass(Y,E)∨┐lucky(Y)
- ┐study(john)
- lucky(john)
- ┐lucky(Z)∨win(Z,lottery)

9. Standardize Variables

- ┐pass(X,ai_exam)∨┐win(X,lottery)∨ happy(X)
- pass(Y,E)∨┐study(Y)
- pass(M,G)∨┐lucky(M)
- ┐study(john)
- lucky(john)
- ┐lucky(Z)∨win(Z,lottery)

C. Add the negation of what is to be proved to the clause form.

- ┐ happy(john).

added it to the other six clauses and shown below:

- ┐pass(X,ai_exam)∨┐win(X,lottery)∨ happy(X)
- pass(Y,E)∨┐study(Y)
- pass(M,G)∨┐lucky(M)
- ┐study(john)
- lucky(john)
- ┐lucky(Z)∨win(Z,lottery)
- ┐ happy(john).

D. Resolve the clauses to producing new clauses and producing a contradiction by generating the empty clause.

- There are two types of resolution, the first one is *backward resolution* and the second is *forward resolution*.