Unification of predicate so that they can be resolved using the resolution principle

- *Axiom 1* P (x , f(x))
- *Axiom 2* Q (y , g(y))
- *Axiom 3* ¬ P (A , w) v ¬Q (y, z)

Question

- *Could you use resolution principle?*
    - *Look at 1 and 3 and try to resolve them*

      P (x , f(x)) ¬ P (A , w)

    - *Do you have equal number of argument?*
    - *Are these argument the same? NO*

Note

- *Need to unify these two predicate*
  1. *First, if the names do not agree then cannot do anything* P (x , f(x)) ¬ P (A , w)
  2. *Second the number of argument has to agree* P (x , f(x)) ¬ P (A , w)
  3. *Third, it has to be some agreements in term of what these argument are.*

Three substitution rule for the unification process

1. *For a variable, you can substitute an object (constant)* < **x , A** > **x ← A**
2. *For a variable, you can substitute another variable* < **y , z** >
3. *For a variable, you can substitute a function as long as the function does not use this variable as its argument* **w = f (x)**

P (x , f(x)) ¬ P (A , w)

< x , A >

P (A , f(A)) ¬ P (A , w)

< w , f(A) >

P (A , f(A)) ¬ P (A , f(A))

NIL

Symbolic processing

On (monitor, table) $\quad\quad$ *¬ On (monitor, table)*

NIL

How to unify

- *An iterative algorithm to unify predicate*

Concepts

1. *Ensure that predate names are the same, so is the number of arguments and the arguments are unified*
2. *Scan* $\quad P(x, y, f(w)) \quad\quad \neg P(A, z, f(w))$

Posing question

Is here agreement or disagreement?

- *Disagreement set* $D=\{x, A\}$

Resolve: substation α *(alpha)* $= < x, A > \quad \alpha . A_1 \quad\quad \alpha . A_2$

Update: predicate $P(A, y, f(w)) \quad\quad \neg P(A, z, f(w))$

Update process is taking substation and applying to expression

Now looking at the second argument and asking the same question

Is here agreement or disagreement? Yes

- *Disagreement set* $D=\{y, z\}$
- α *(alpha)* $= < y, z >$
- α *(new alpha)*$=$ α *(old alpha)* $\cup < y, z >$
- $\boxed{\alpha \text{ (new alpha)}}$ $= \{ < x, A >, \; < y, z > \}$

$P(A, z, f(w)) \quad\quad \neg P(A, z, f(w))$

MGU: most general unifier

The unification algorithm

1. *Start off with an empty set substitute α_0 ={} given axioms A_1 , A_2, A_3, ………… A_n*
2. *Step K+1 assume that at step k, α_k has been found if α_k o A_i for i=1 …..n are the same then α_k is MGU*
   else
   determine D_k disagreement set of α_k o A_i *O mean apply*
   modify α_k to get $\alpha_{k+1} = \alpha_k \cup$ { substitutions } that resolve the disagreement set D_k.
3. *Contain k=k+1*

Example

Axioma 1 → $P(A, x, f(g(y)))$

Axioma 2 → $\neg P(z, h(z, w), f(w))$

- α_0 ={}
- D_0 = {A , z} α= <z , A>
- $\alpha_1 = \alpha_0 \cup$ {<z , A>}={<z , A>
- α_1 o A_1 → $P(A, x, f(g(y)))$
- α_1 o A_2 → $P(A, h(z, w), f(w))$

Are these predicate the same?

- D_1={x , h(A , w)} α= <x , h(A,w)>
- $\alpha_2 = \alpha_1 \cup$ {< x , h(A,w) >}={<z , A> , <x , h(A,w)>}
- α_2 o A_1 → $P\boxed{(A, h(A, w)}\boxed{, f(g(y)))}$
- α_2 o A_2 → $P\boxed{(A, h(A, w)}\boxed{, f(w))}$
- D_2={w , g(y)} α= <w, g(y)>
- $\alpha_3 = \alpha_2 \cup$ {< x , g(y) >}={<z , A> , <x , h(A,w)>}
- α_3= {< z , A> , <x ,h(A, g(y))> , <w , h(A, , g(y))>}
- α_3 o A_1 → $P(A, h(A, g(y)), f(g(y)))$
- α_3 o A_1 → $P(A, h(A, g(y)), f(g(y)))$

Finish logic

Unification

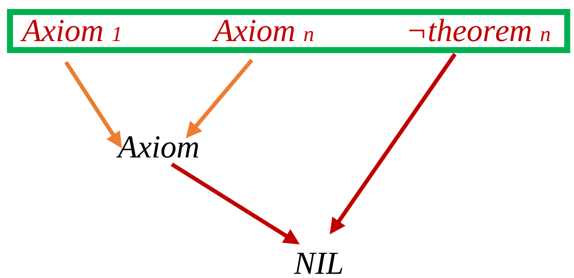
1. ***Do you understand why we do it?***

Clause form

2. ***If we have axioms and negated theorem and there are in the clause form, can be unified?***

- *Axiom $_1$* *clause*
- *.*
- *Axiom $_n$* *clause* *can be unified?*
-
- *¬ theorem* *clause*

○ ***need to resolve***



How does this correspond to AI production system?

- *States → database*
- *initial state → axioms and negated theorem*
- *The final goal state → NIL*
- *Resolution → operators*

What would be the control strategy to resolve?

Example

1. ***whoever can read is literate***

   $\forall \mathbf{x} \, (R(x) \Rightarrow L(x))$ R→ *Read*

   L→ *Literate*

2. ***Dolphins are not literate***

   $\forall \mathbf{x} \, (D(x) \Rightarrow \neg L(x))$

3. ***Some dolphins are intelligent***

   $\boxed{\exists \mathbf{x} \, (D(x) \Rightarrow I(x))}$

need to use Skolem function

Theorem

- ***Some who intelligent cannot read***

  $\exists x \, (I(x) \wedge \neg R(x))$

- ***Convert 1, 2, 3, TH → clause form***
  - ***Drop the quantifier***

1. $\neg R(x) \; \mathbf{v} \; L(x)$
2. $\neg D(x) \; \mathbf{v} \; \neg L(x)$
3. $D(x) \wedge I(x)$

   3a. $D(A)$
   3.b $I(A)$

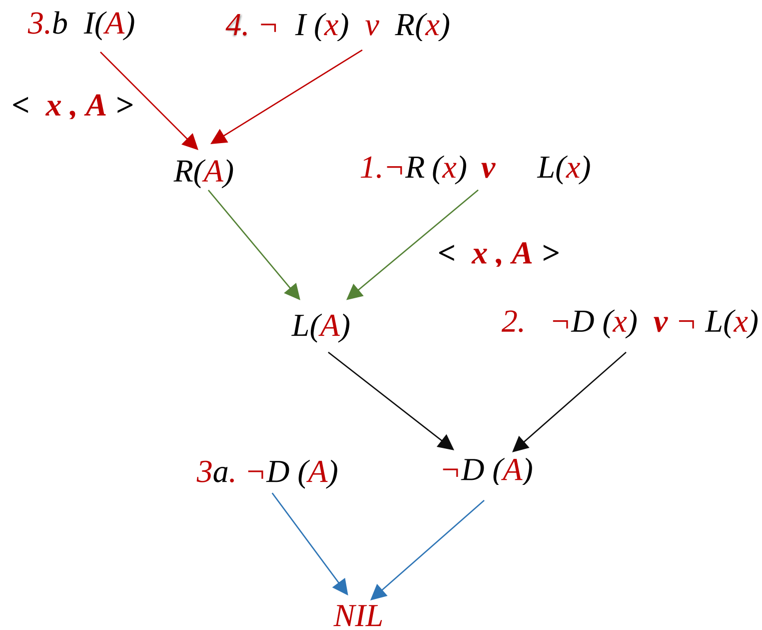
$\neg$ ***TH :*** $\neg(\exists \mathbf{x} \, (I(x) \wedge \neg R(x))$

$\forall \mathbf{x} \; \neg (I(x) \wedge \neg R(x))$

$\cancel{\forall \mathbf{x}} \; (\neg I(x) \; v \; R(x))$

4. $\neg I(x) \; v \; R(x)$

Substitute **< *x* , *A* >**



What are these control strategies?

<u>Control strategy for resolution proof</u>

1. ***Breath first search (BFS)***

   All first level resolvents are computed first then the second level resolvents……..etc.

2. ***Set of support***

   At least one of each resolvent is selected from the clauses resulting then ¬(negate) of the goal or from their descendants.

3. ***Linear input***

   Each resolvent has at least one parents in the base set (initial state)

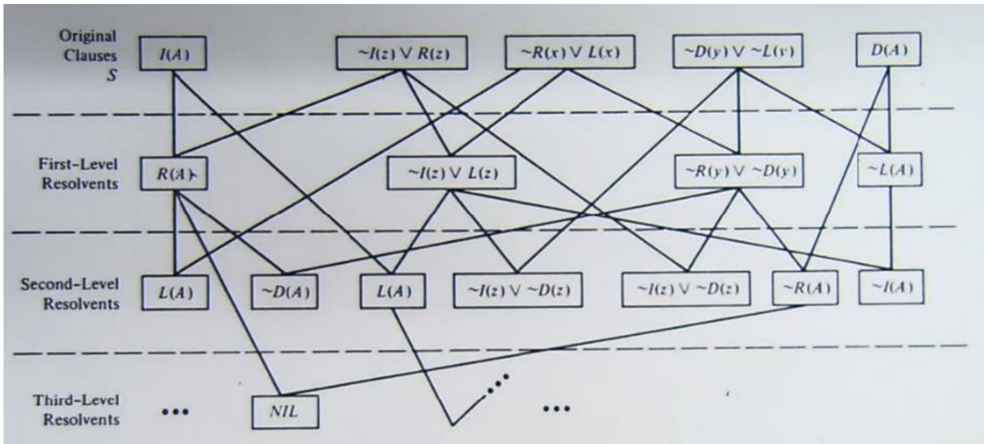


Fig. 5.2 Illustration of a breadth-first strategy.

These strategy is complete

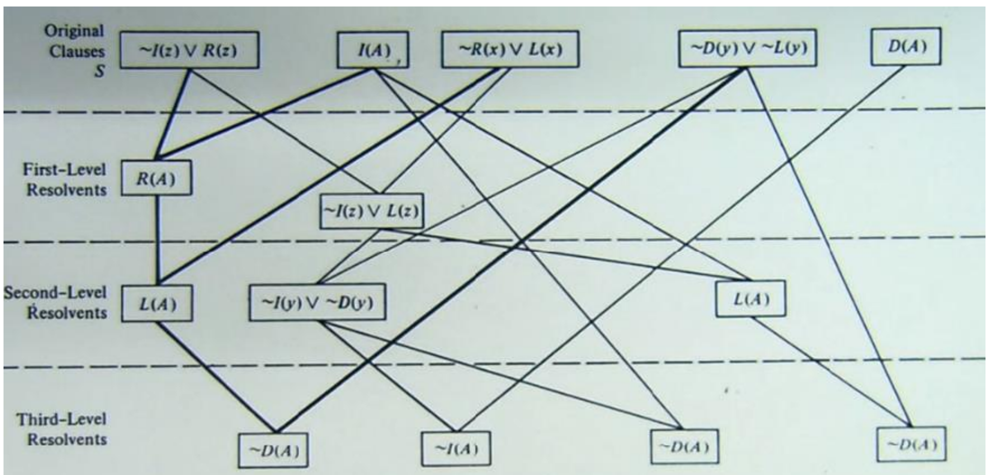


Fig. 5.3 Illustration of a set-of-support strategy.

These strategy is heuristic

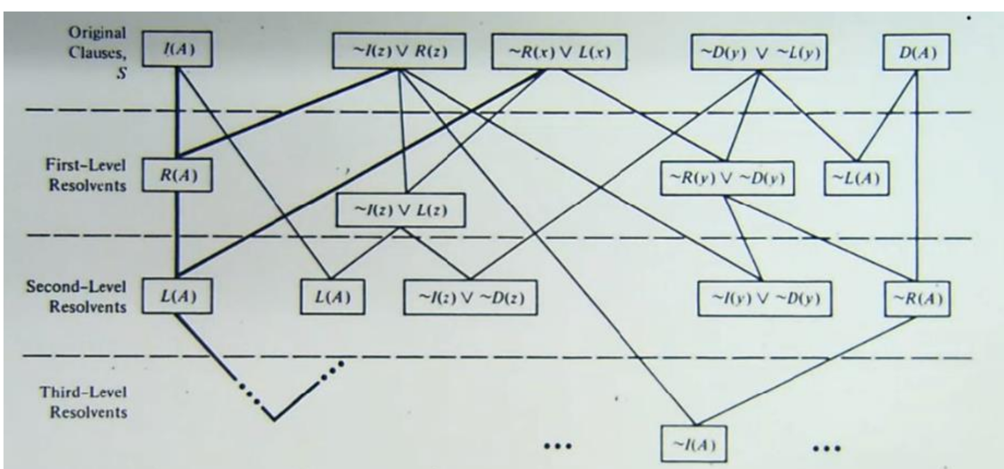


Fig. 5.4 Illustration of a linear-input form strategy.

Finding answer to "Questions" using resolution refutation process

Example:

If Fido goes wherever John goes and if John is at school where is Fido?

Axioms

- ***A1*** $\forall \mathbf{x}\ (AT\ (John\ ,\ x\ ) => AT(\ Fido\ ,\ x\ ))$
- ***A2*** $AT\ (\ John\ ,\ school\ )$

Question is

$$\exists\ x\ (AT\ (\ Fido\ ,\ x)) \rightarrow x\ =\ ?$$

Sol

A1 $\neg\ AT\ (\ John\ ,\ x\ )\ v\ \ AT(\ Fido\ ,\ x\ )$

A2 $AT\ (\ John\ ,\ school\ )$

$\neg\ \ \ \exists\ x\ (\ AT\ (\ Fido\ ,\ x)) \rightarrow \cancel{\forall}\ \mathbf{x}\ \ (\neg\ AT\ (\ Fido\ ,\ x))$

Drop

Resolve

$\neg\ AT\ (\ John\ ,\ x\ )\ v\ \ AT(\ Fido\ ,\ x\ )$ $AT\ (\ John\ ,\ school\ )$

< ***x , school*** >

$AT\ (\ Fido\ ,\ school\ )$ $\neg AT\ (\ Fido\ ,\ x\ )$

< ***x , school*** >

NIL

$\neg \quad \exists x\, (AT(Fido, x)) \rightarrow \forall x\, (\neg AT(Fido, x)) \vee AT(Fido, x)$

$\neg AT(John, x) \vee AT(Fido, x)$ $\qquad$ $AT(John, school)$

$< x, school >$

$AT(Fido, school)$ $\qquad$ $\neg AT(Fido, x) \vee AT(Fido, x)$

$< x, school >$

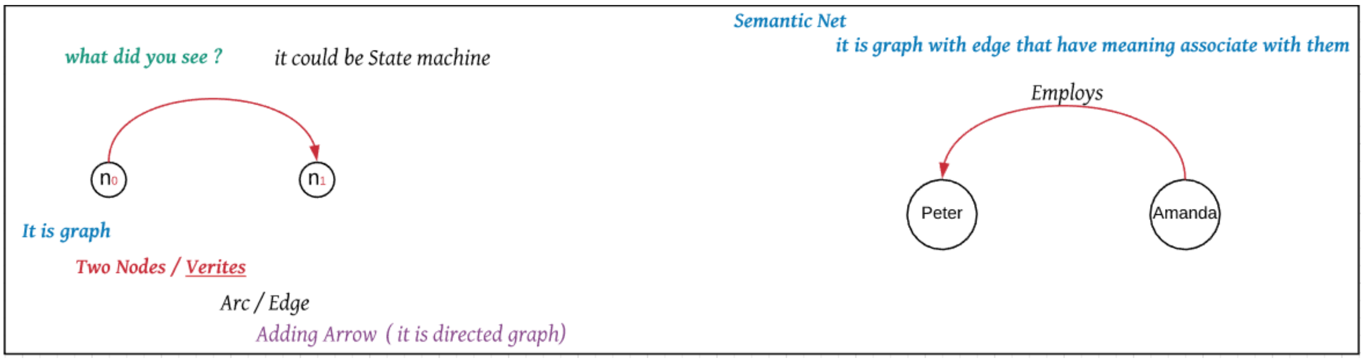
$AT(Fido, school)$

Knowledge Representation

Logic

Structured schemes

What is the structured scheme? What is knowledge representation? Why people come it with AI?



What is the semantic net?

Having graph and nodes. The nodes could be object / entities /processes

Edge represents relation /explicit meaning

What is the typical relation?

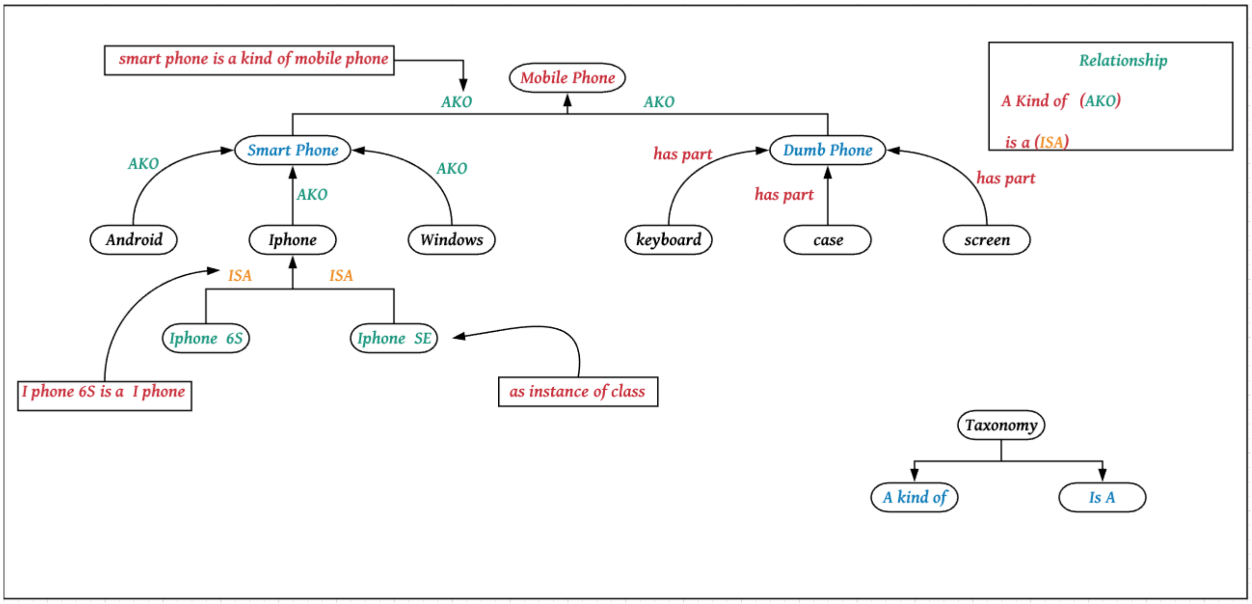
Typical relation

a) Decomposition

b) Taxonomy (away of classifying object, entity that presents in the node)

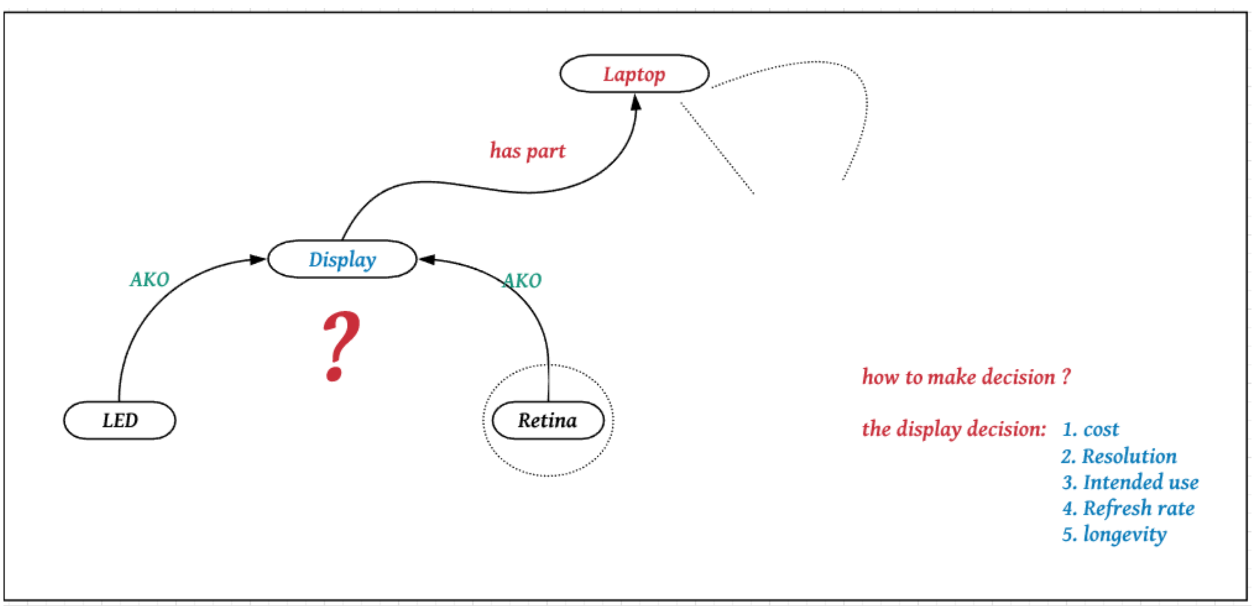
Example

Mobile phone



Advantages

- *Ability to specify a range of variants for system components and their decomposition*



- *Inheritance reuse take advantage of properties that span a class hierarchy*