Lecture 10: Resolution

Artificial Intelligence

CS-6364

Resolution = Refutation

- Definition - *one complete inference procedure using resolution!*

  also known as - proof by contradiction

  - reductio ad absurdum

- The idea: to prove P, assume P is false (i.e. add $\neg P$ to KB) and prove by contradiction

  $(KB \wedge \neg P \Rightarrow False) \Leftrightarrow (KB \Rightarrow P)$

Example Proof

- Refutation on a more complex example:

- In English:

  Everyone who loves all animals is loved by someone.
  Anyone who kills an animal is loved by no one.
  Jack loves all animals.
  Either Jack or Curiosity killed the cat who is named Tuna.
  Did Curiosity kill the cat?

Translation in FOL

A. $\forall x\, [\forall y\, \text{Animal}(y) \Rightarrow \text{Loves}(x, y)] \Rightarrow [\exists y\, \text{Loves}(y, x)]$

B. $\forall x\, [\exists y\, \text{Animal}(y) \wedge \text{Kills}(x,y)] \Rightarrow [\forall z\, \neg \text{Loves}(z,x)]$

C. $\forall x\, \text{Animal}(x) \Rightarrow \text{Loves}(\text{Jack}, x)$

D. $\text{Kills}(\text{Jack}, \text{Tuna}) \vee \text{Kills}(\text{Curiosity}, \text{Tuna})$

E. $\text{Cat}(\text{Tuna})$

F. $\forall x\, \text{Cat}(x) \Rightarrow \text{Animal}(x)$

$\neg$G. $\neg \text{Kills}(\text{Curiosity}, \text{Tuna})$

Conversion to CNF

A. $\forall x\, [\forall y\, Animal(y) \Rightarrow Loves(x, y)] \Rightarrow [\exists y\, Loves(y, x)]$
B. $\forall x\, [\exists y\, Animal(y) \wedge Kills(x,y)] \Rightarrow [\forall z\, \neg Loves(z,x)]$
C. $\forall x\, Animal(x) \Rightarrow Loves(Jack, x)$
D. $Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$
E. $Cat(Tuna)$
F. $\forall x\, Cat(x) \Rightarrow Animal(x)$
¬G. $\neg Kills(Curiosity, Tuna)$

A1. $Animal(F(x)) \vee Loves(G(x), x)$
A2. $\neg Loves(x, F(x)) \vee Loves(G(x), x)$
B. $\neg Animal(y) \vee \neg Kills(x,y) \vee \neg Loves(z,x)]$
C. $\neg Animal(x) \vee Loves(Jack, x)$
D. $Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$
E. $Cat(Tuna)$
F. $\neg Cat(x) \vee Animal(x)$
¬G. $\neg Kills(Curiosity, Tuna)$

Proof

A1. $Animal(F(x)) \vee Loves(G(x), x)$

A2. $\neg Loves(x, F(x)) \vee Loves(G(x), x)$

B. $\neg Animal(y) \vee \neg Kills(x,y) \vee \neg Loves(z,x)]$

C. $\neg Animal(x) \vee Loves(Jack, x)$

D. $Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$

E. $Cat(Tuna)$

F. $\neg Cat(x) \vee Animal(x)$

$\neg$G. $\neg Kills(Curiosity, Tuna)$

R1: E & F $\Rightarrow$ $Animal(Tuna)$ $\{x/Tuna\}$

R2: R1&B $\Rightarrow$ $\neg Loves(z, x) \vee \neg Kills(x, Tuna)$ $\{y/Tuna\}$

R3: D & $\neg$G $\Rightarrow$ $Kills(Jack, Tuna)$

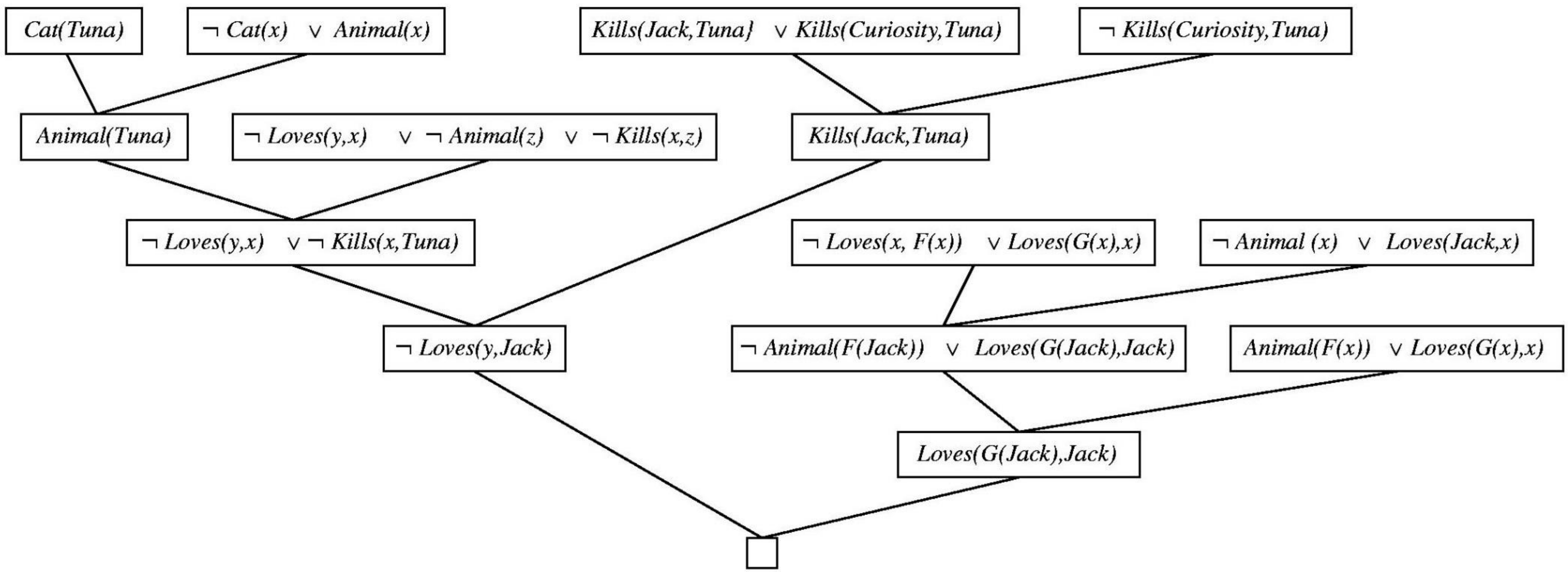
R4: R3 & R2 $\Rightarrow$ $\neg Loves(z,Jack)$ $\{x/Jack\}$

R5: A2 & C $\Rightarrow$ $\neg Animal(Jack) \vee Loves(G(Jack), Jack)$ $\{x/Jack,$ $F(x) = Identity\_Function\}$

R6: R5 & A1 $\Rightarrow$ $Loves(G(Jack), Jack)$ $\{x/Jack; F(x) = Identity\_Function\}$

R7: R4 & R6 $\Rightarrow$ FALSE $\{z/G(Jack)\}$

Resolution Proof



Example 2

In English:
The custom officials searched everyone who entered the country and was not a VIP. Some of the drug pushers entered this country and they were only searched by drug pushers. No drug pusher was a VIP.

At least one of the custom officials was a drug pusher.

Translation in FOL

"The custom officials searched everyone who entered the country and was not a VIP"

$$\forall x\ \exists y\ (entered\_country(x) \wedge \neg VIP(x)) \Rightarrow (official(y) \wedge searched(y,x))$$

with VIP(x) - x is a VIP
official(y) - y is a custom official
searched(x,y) x has searched y

"Some of the drug pushers entered this country and they were only searched by drug pushers"

$$\exists x\ \forall y\ [entered\_country(x) \wedge d\_pusher(x)] \wedge [searched(y,x) \Rightarrow d\_pusher(y)]$$

More translations!

"No drug pusher was a VIP"

$\forall x\ [\text{d\_pusher}(x) \Rightarrow \neg\ \text{VIP}(x)]$

Goal: "At least one of the custom officials is a drug pusher"

$\exists x\ (\text{official}(x) \land \text{d\_pusher}(x))$

More details

Convert the translation in FOL into CNF:

"The custom officials searched everyone who entered the country and was not a VIP"

$\forall x\ \exists y\ (entered\_country(x) \wedge \neg VIP(x)) \Rightarrow (official(y) \wedge searched(y,x))$

Eliminate "$\Rightarrow$"

$\forall x\ \exists y\ (\neg(entered\_country(x) \wedge \neg VIP(x)) \vee (official(y) \wedge searched(y,x))$

Eliminate Exist. Quantifiers

$\forall x\ (\neg entered\_country(x) \vee VIP(x) \vee (official(f(x)) \wedge searched(f(x),x))$

Skolem function

The axioms

We obtain the first 2 sentences in CNF:

1. $\neg$entered_country(x) $\vee$ VIP(x) $\vee$ official(f(x))
2. $\neg$entered_country(x) $\vee$ VIP(x) $\vee$ searched(f(x),x)

Next sentence in FOL:

$\exists$x $\forall$y [entered_country(x) $\wedge$ d_pusher(x)] $\wedge$ [searched(y,x) $\Rightarrow$ d_pusher(y)]

eliminate "$\Rightarrow$"

$\exists$x $\forall$y [entered_country(x) $\wedge$ d_pusher(x)] $\wedge$ [$\neg$searched(y,x) $\vee$ d_pusher(y)]

Then what? When eliminating the exist. Quantifier:

x=a (a constant) + standardize variables (change y into x)

We obtain the next 3 sentences in CNF:

3. entered_country(a)
4. d_pusher(a)
5. $\neg$searched(x,a) $\vee$ d_pusher(x)

Last axioms

From statement $\forall x\ (d\_pusher(x) \Rightarrow \neg VIP(x))$

6. $\neg d\_pusher(x) \vee \neg VIP(x)$

Finally:

The goal: $\underline{\exists x\ (official(x) \wedge d\_pusher(x))}$

Negated: $\forall x\ (\neg official(x) \vee \neg d\_pusher(x))$

Generates the sentence in CNF:

7. $\neg official(x) \vee \neg d\_pusher(x)$

Axioms in CNF

1. $\neg$entered_country(x) $\vee$ VIP(x) $\vee$ official(f(x))
2. $\neg$entered_country(x) $\vee$ VIP(x) $\vee$ searched(f(x),x)
3. entered_country(a)
4. d_pusher(a)
5. $\neg$searched(x,a) $\vee$ d_pusher(x)
6. $\neg$d_pusher(x) $\vee$ $\neg$VIP(x)
7. $\neg$official(x) $\vee$ $\neg$d_pusher(x)

Prove (by refutation)

The refutation

2: ¬entered_country(x) ∨ VIP(x) ∨ searched(f(x),x)

6. ¬d_pusher(x) ∨ ¬VIP(x)

$\theta = \{\}$

8: ¬d_pusher(x) ∨ ¬entered_country(x) ∨ searched(f(x),x)

4. d_pusher(a)

$\theta = \{x/a\}$

9: ¬entered_country(a) ∨ searched(f(a),a)

$\theta = \{\}$

3. entered_country(a)

10: searched(f(a),a)

The refutation-2

1: ¬entered_country(x) ∨ VIP(x) ∨ official(f(x))

6. ¬d_pusher(x) ∨ ¬VIP(x)

$\theta = \{\}$

11: ¬d_pusher(x) ∨ ¬entered_country(x) ∨ official(f(x))

4. d_pusher(a)

$\theta = \{x/a\}$

12: ¬entered_country(a) ∨ official(f(a))

3. entered_country(a)

$\theta = \{\}$

13: official(f(a))

The refutation-3

7: ¬official(x) ∨ ¬d_pusher(x)

4. d_pusher(a)

$\theta = \{x/a\}$

14: ¬official(f(a))

13. official(f(a))

NIL

The statement in the goal is valid.

Example 3

John likes all kinds of food.
Apple is food.
Chicken is food.
Anything anyone eats and isn't killed by is food.
Bill eats peanuts and is alive.
Sue eats anything Bill eats.

Show that John likes peanuts.
What food does Sue eat?

Translation in FOL

"John likes all kinds of food."

$\forall x \; food(x) \Rightarrow likes(John, x)$

"Apple is food." → $food(Apple)$

"Chicken is food." → $food(Chicken)$

"Anything anyone eats and isn't killed by is food."

$\forall x \; \forall y \; (eats(x,y) \land \neg \; killed(x,y)) \Rightarrow food(y)$

More translations

"Bill eats peanuts and is alive."

We need some commonsense knowledge:

I assume $alive(x) \equiv \forall y\ (\neg killed(x,y))$ is too general for the context of the sentence

I prefer the conversion:

$$eats(Bill,Peanuts) \land \neg\ killed(Bill,\ Peanuts)$$

I could have used:

$$eats(Bill,Peanuts) \land \forall x\ \neg killed(Bill,\ x)$$

"Sue eats anything Bill eats."

$$\forall x\ eats(Bill,x) \Rightarrow eats(Sue,x)$$

Refute "John likes peanuts"

Transformation to CNF:

1. $\neg$food(x) $\vee$ likes(John, x)
2. food(Apple)
3. food(Chicken)
4. $\neg$eats(x,y) $\vee$ killed(x,y)) $\vee$ food(y)
5. eats(Bill, Peanuts)
6. $\neg$killed(Bill, Peanuts)
7. $\neg$eats(Bill,x) $\vee$ eats(Sue,x)

+Goal: 8. $\neg$likes(John,Peanuts)

The proof

9. $\neg$food(Peanuts) from 1&8 $\theta = \{x/Peanuts\}$

10. killed(Bill, Peanuts) $\vee$ food(Peanuts)
from 4&5 $\theta = \{x/Bill; y/Peanuts\}$

11. food(Peanuts) from 6&10 $\theta = \{\}$

12. NIL from 9&11 $\theta = \{\}$

$\Rightarrow$ This resolution theorem proving showed that the clause “John likes peanuts” is valid

"What food does Sue eat?"

We do again theorem proving, changing the goal clause to:

$\neg$likes(John, Peanuts) $\vee$ $\neg$eats(Sue,z)

Axioms

1. $\neg$food(x) $\vee$ likes(John, x)
2. food(Apple)
3. food(Chicken)
4. $\neg$eats(x,y) $\vee$ killed(x,y) $\vee$ food(y)
5. eats(Bill, Peanuts)
6. $\neg$killed(Bill, x)
7. $\neg$eats(Bill,x) $\vee$ eats(Sue,x)
8. $\neg$likes(John,Peanuts) $\vee$ $\neg$eats(Sue,z)

Proof

9. $\neg$food(Peanuts) $\vee$ $\neg$eats(Sue,z)
from 1&8 $\theta = \{x/Peanuts\}$

10. eats(Sue, Peanuts) from 5&7 $\theta = \{x/Peanuts\}$

11. $\neg$food(Peanuts) from 9&10 $\theta = \{z/Peanuts\}$

12. killed(Bill, Peanuts) $\vee$ food(Peanuts)
from 4&5 $\theta = \{x/Bill;\ y/Peanuts\}$

13. food(Peanuts) from 6&12 $\theta = \{x/Peanuts\}$

14. NIL from 11&13

From the substitution in 11, we see that z=Peanuts $\Rightarrow$ Sue eats Peanuts

Another proof

9. ¬food(Peanuts) ∨ ¬eats(Sue,z)
from 1&8 $\theta = \{x/Peanuts\}$
10. killed(Bill, Peanuts) ∨ food(Peanuts)
from 4&5 $\theta = \{x/Bill;\ y/Peanuts\}$

11. food(Peanuts) from 6&10 $\theta = \{x/Peanuts\}$
12. ¬eats(Sue, z) from 9&11 $\theta = \{\}$
13. eats(Sue, Peanuts) from 5&7 $\theta = \{x/Peanuts\}$
14. NIL from 12&13 $\theta = \{z/Peanuts\}$

$\downarrow$

eats(Sue, Peanuts)

Answering questions

IMPORTANT: when answering questions that are not YES/NO questions, the goal needs to have the negated predication with the argument(s) that will be substituted by the answer.

Example:

"What food does Sue eat?"

Set the goal to:

8′. $\forall x\ (eats(Sue,x) \vee \neg eats(Sue,x))$

Lessons learned

- 3 examples
- Several kinds of proofs
- Formal method of answering questions

Resolution Strategies

- Strategies that help find proofs efficiently. We know that repeated applications of the resolution rule will find the proof if one exists → *<u>what about the efficiency of this process?</u>*
- *We look back at 4 strategies used to guide the search for the proof*

- <u>Unit Preference</u>
  - Prefers sentences that are a single literal (unit clauses)
  - The idea is that we are trying to produce *an empty clause*, so it might be a good idea to prefer inferences that produce shorter clauses
  - <u>For example</u>, resolving a unit sentence **A** with any other sentence **B** ∨ ¬**A** ∨ **C** always yields a shorter clause: **B** ∨ **C**

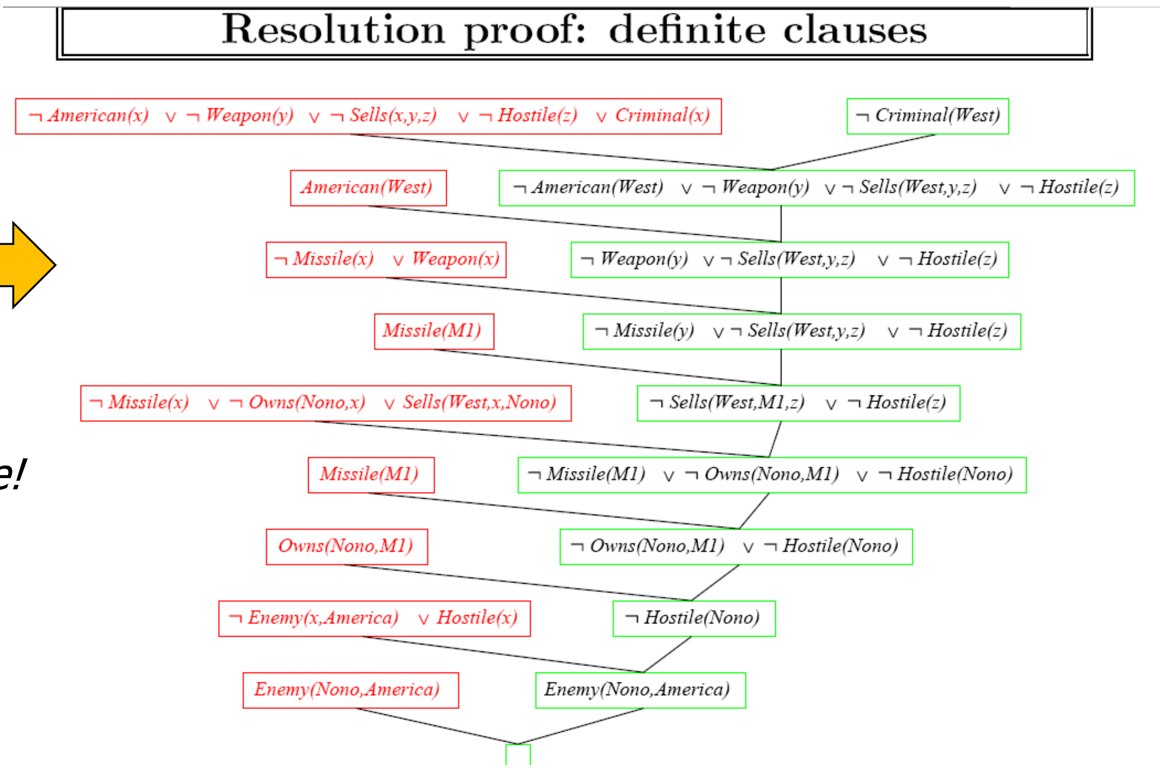
Set of Support

- *The set of support* is a sub-set of sentences such that one sentence from this set should be used in each resolution step! The result of resolution is also added to the set of support.
  - *If we chose a set of support that is small enough when compared with the rest of the KB, the search space is reduced dramatically.*
  - Common approach: use the negated query as the set of support, on the assumption that the original knowledge base is consistent.

Input Resolution

Every application of resolution combines one of the input sentences (from the KB or the query) with some other sentence (generated by a prior application of resolution)

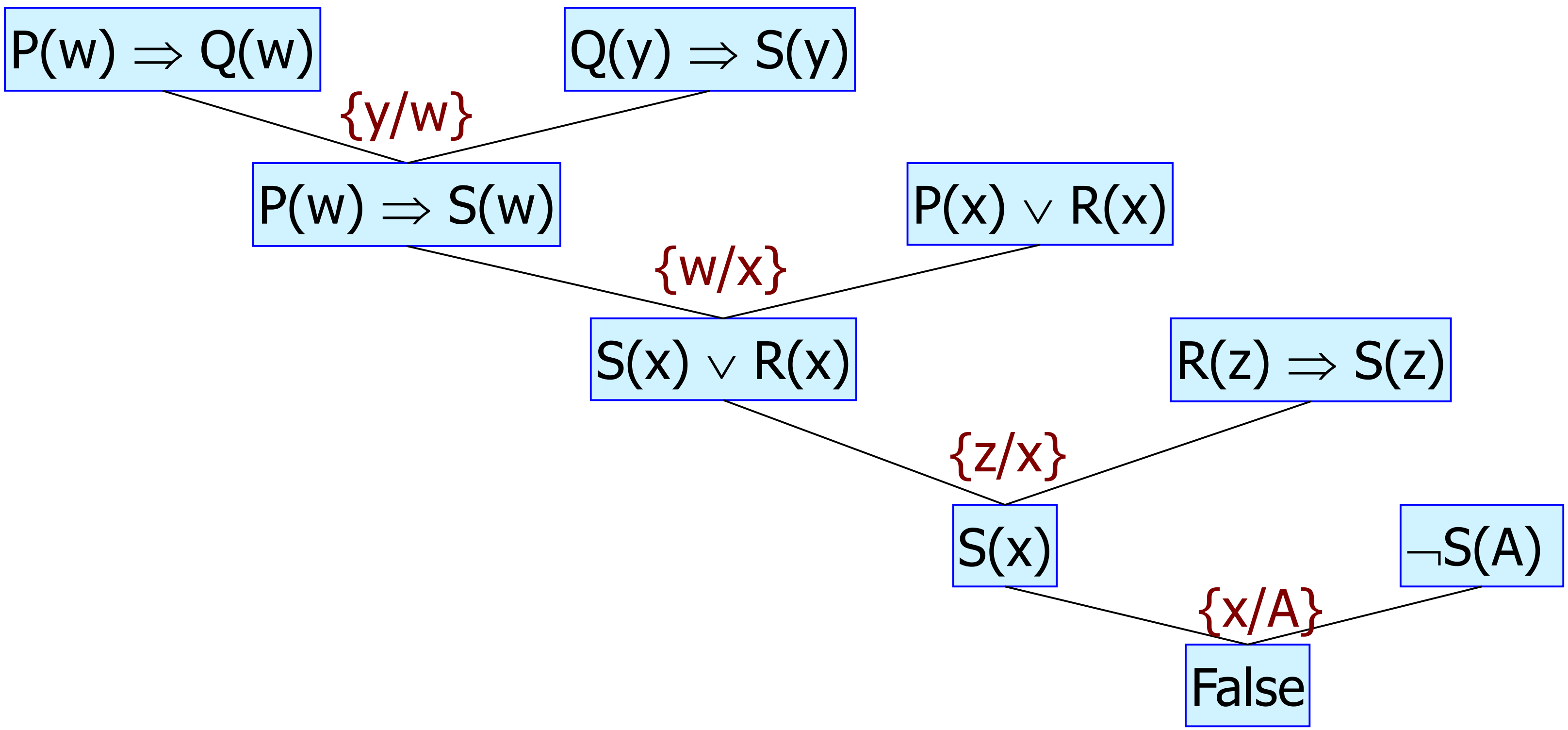
The proof looks like this



Shape of a single spine
With sentences coming in the spine!

Input Resolution

- Every resolution combines one of the input sentences with some other sentences (from the KB or the query)



Subsumption

Eliminates all sentences that are subsumed by an existing sentence in the KB.

For example, if *P(x)* is in the KB, then there is no sense in adding *P(A)* and even less sense in adding *P(A)* $\vee$ *Q(B)*. It helps keep the KB small, and thus helps keep the search space small.

- Subsumption keeps the KB small

Demodulation Rule → the other way of dealing with equality

□ Definition:

Informally:

For equality *x=y*
+ any sentence with a nested term that unifies with *x*, it derives the same sentence with *y* substituted for the nested term

Formally:

$\forall$x,y,z where $UNIFY(x,z) = \theta$:

$$\frac{x=y,\ (\ldots z \ldots)}{(\ \ldots SUBST(\theta, y) \ldots\ )}$$

Theorem Provers

OTTER (Organized Techniques for Theorem-proving and Effective Research) (McCune 1992) PROVER 9

In preparing a problem for Otter, the user must divide the knowledge into four parts:

1. A set of clauses known as the set of support (SoS) which define the important facts about the problem. Every resolution step resolves a member of the set of support against another axiom, so the search is focused on the set of support.
2. A set of usable axioms that are outside the set of support. These provide background knowledge about the problem area. The boundary between what is part of the problem and what is background (thus in usable axioms) is up to the user's judgment.
3. A set of equations known as rewrites or demodulators.
4. A set of parameters and clauses that defines the control strategy. The user specifies a *heuristic function* to control the search and a *filtering function* to eliminate some sub-goals as un-interesting.

procedure OTTER(*sos, usable*)
  inputs: *sos*, a set of support—clauses defining the problem (a global variable)
    usable, background knowledge potentially relevant to the problem

  repeat
    clause ← the lightest member of *sos*
    move *clause* from *sos* to *usable*
    PROCESS(INFER(*clause, usable*), *sos*)
  until *sos* = [ ] **or** a refutation has been found

---

function INFER(*clause, usable*) **returns** clauses

  resolve *clause* with each member of *usable*
  return the resulting clauses after applying FILTER

---

procedure PROCESS(*clauses, sos*)

  for each *clause* **in** *clauses* **do**
    clause ← SIMPLIFY(*clause*)
    merge identical literals
    discard clause if it is a tautology
    sos ← [*clause*–*sos*]
    if *clause* has no literals **then** a refutation has been found
    if *clause* has one literal **then** look for unit refutation