Adapted from AIMA slides

Logic: exercises

Peter Antal
antal@mit.bme.hu

Truth tables for connectives

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
$false$	$false$	$true$	$false$	$false$	$true$	$true$
$false$	$true$	$true$	$false$	$true$	$true$	$false$
$true$	$false$	$false$	$false$	$true$	$false$	$false$
$true$	$true$	$false$	$true$	$true$	$true$	$true$

Logical equivalence

- Two sentences are logically equivalent} iff true in same models: $\alpha \equiv ß$ iff $\alpha \models \beta$ and $\beta \models \alpha$
-
-

$$
\begin{aligned}
(\alpha \wedge \beta) &\equiv (\beta \wedge \alpha) \quad \text{commutativity of } \wedge \\
(\alpha \vee \beta) &\equiv (\beta \vee \alpha) \quad \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) &\equiv (\alpha \wedge (\beta \wedge \gamma)) \quad \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) &\equiv (\alpha \vee (\beta \vee \gamma)) \quad \text{associativity of } \vee \\
\neg(\neg\alpha) &\equiv \alpha \quad \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) \quad \text{contraposition} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \vee \beta) \quad \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) \quad \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) &\equiv (\neg\alpha \vee \neg\beta) \quad \text{de Morgan} \\
\neg(\alpha \vee \beta) &\equiv (\neg\alpha \wedge \neg\beta) \quad \text{de Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) &\equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) \quad \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) &\equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) \quad \text{distributivity of } \vee \text{ over } \wedge
\end{aligned}
$$

Resolution

Conjunctive Normal Form (CNF)

conjunction of disjunctions of literals

clauses

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

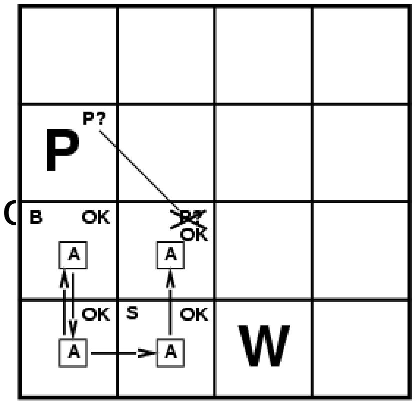
- Resolution inference rule (for CNF):
-

$$\frac{l_i \vee \ldots \vee l_k, \qquad m_1 \vee \ldots \vee m_n}{l_i \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k \vee m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n}$$

where l_i and m_j are complementary literals.

E.g., $$\frac{P_{1,3} \vee P_{2,2}, \qquad \neg P_{2,2}}{P_{1,3}}$$

- Resolution is sound and complete for propositional lo
-



Resolution

Soundness of resolution inference rule:

$$\frac{\begin{array}{c}\neg(l_i \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k) \Rightarrow l_i \\ \neg m_j \Rightarrow (m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n)\end{array}}{\neg(l_i \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k) \Rightarrow (m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n)}$$

Conversion to CNF

$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})\beta$

1. Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta)\wedge(\beta \Rightarrow \alpha)$.
2.

$(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$

2. Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha\vee \beta$.

$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$

3. Move $\neg$ inwards using de Morgan's rules and double-negation:

$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \vee \neg P_{2,1}) \vee B_{1,1})$

4. Apply distributivity law ($\wedge$ over $\vee$) and flatten:

$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

Resolution algorithm

- Proof by contradiction, i.e., show $KB \land \neg\alpha$ unsatisfiable

```
function PL-RESOLUTION(KB, α) returns true or false
    clauses ← the set of clauses in the CNF representation of KB ∧ ¬α
    new ← { }
    loop do
        for each Ci, Cj in clauses do
            resolvents ← PL-RESOLVE(Ci, Cj)
            if resolvents contains the empty clause then return true
            new ← new ∪ resolvents
        if new ⊆ clauses then return false
        clauses ← clauses ∪ new
```

Resolution strategies (heuristics for clause selection)

1. **Unit clause preference**: P, ¬P ∨ [.....] ==> [.....] shorter!

2. **'Set of Support'**

   resolution (a clause from a 'Set of Support' and an external clause), rezolvent into 'Set of Support'-ba,

   complete, if clauses not in 'Set of Support' are satisfiable

   in practice: 'Set of Support' = the negated question (the rest is assumed to be true)

3. **Input resolution**

   The resolvent in step i. is one of the clause in step i+1 (it starts with the question). **Complete in Horn KBs.**

4. **Linear resolution**

   P and Q can be resolved, if P is in the KB or P is the ancestor of Q in the proof tree. **Complete.**

5. **Pruning**

   Eliminate all rules more specific than in the knowledge base.

Resolution: brief summary

- Full first-order version:

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{(\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n)\theta}$$

where $\texttt{Unify}(\ell_i, \neg m_j) = \theta$.

- The two clauses are assumed to be standardized apart so that they share no variables.
- For example,

$$\frac{\neg Rich(x) \vee Unhappy(x) \qquad Rich(Ken)}{Unhappy(Ken)}$$

with $\theta = \{x/Ken\}$

- Apply resolution steps to CNF(KB $\wedge \neg\alpha$); complete for FOL

Definitions & conversions

- What is the definition of the satisfiability of a logical expression?
- Compare the deductive and the abductive inference. How and when can we use them?
- A propositional knowledge base contains the following statements below.Convert to clause form and prove with resolution that T is true.
  - $P \rightarrow (R \vee S)$, $\neg P \rightarrow (R \vee S)$, $\neg S$, $(R \vee U) \rightarrow T$
- Decide with truth-table that the following statement is not satisfiable, satisfiable, or valid:
  - $(A \rightarrow \neg B) \rightarrow (C \rightarrow B)$
- Show the type of the following expression using truth-tables!
  - $(A \rightarrow B)$ XOR $(B \rightarrow A)$
- What is the type of the expression (valid, not satisfiable, satisfiable)?
  - $(\neg A \vee B) \rightarrow (C \rightarrow \neg B)$
  - $(A \rightarrow \neg B) \vee (C \vee \neg B)$
  - $(A \rightarrow \neg B) \vee (\neg A \rightarrow B)$
- What is the type of the next statement (valid, satisfiable, not satisfiable, none of these).
  - $(A \rightarrow \neg B) \rightarrow (C \rightarrow B)$.
- What is the resolution inference step? Show its soundness with truth-tables for three variables.

Conversion to CNF

- Everyone who loves all animals is loved by someone:
  $\forall x\ [\forall y\ Animal(y) \Rightarrow Loves(x,y)] \Rightarrow [\exists y\ Loves(y,x)]$

- 1. Eliminate biconditionals and implications
-
  $\forall x\ [\neg\forall y\ \neg Animal(y) \vee Loves(x,y)] \vee [\exists y\ Loves(y,x)]$

- 2. Move $\neg$ inwards: $\neg\forall x\ p \equiv \exists x\ \neg p$, $\neg\exists x\ p \equiv \forall x\ \neg p$
-
  $\forall x\ [\exists y\ \neg(\neg Animal(y) \vee Loves(x,y))] \vee [\exists y\ Loves(y,x)]$
  $\forall x\ [\exists y\ \neg\neg Animal(y) \wedge \neg Loves(x,y)] \vee [\exists y\ Loves(y,x)]$
  $\forall x\ [\exists y\ Animal(y) \wedge \neg Loves(x,y)] \vee [\exists y\ Loves(y,x)]$

Conversion to CNF contd.

3. Standardize variables: each quantifier should use a different one

$\forall x\ [\exists y\ Animal(y) \land \neg Loves(x,y)] \lor [\exists z\ Loves(z,x)]$

4. Skolemize: a more general form of existential instantiation.

Each existential variable is replaced by a Skolem function of the enclosing universally quantified variables:

$$\forall x\ [Animal(F(x)) \land \neg Loves(x,F(x))] \lor Loves(G(x),x)$$

5. Drop universal quantifiers:

$$[Animal(F(x)) \land \neg Loves(x,F(x))] \lor Loves(G(x),x)$$

6. Distribute $\lor$ over $\land$:

$$[Animal(F(x)) \lor Loves(G(x),x)] \land [\neg Loves(x,F(x)) \lor Loves(G(x),x)]$$

CNF, Skolemization

- The $\forall$, $\exists$, $\rightarrow$, $\Leftrightarrow$, $\wedge$ symbols are not present in clauses in conjunctive normal form. Describe their eliminitation.
- Explain the skolemization step and its purpose using the following expression:
  - $\neg\exists x.\ A(x) \rightarrow \neg\forall .y\ B(x,y)$!

Representation&inference in FOL I.

$\forall x\, P(x) \rightarrow \exists\, x\, P(x)$?

Valid?
negation:

$$\neg\ (\forall\, x\, P(x) \rightarrow \exists\, x\, P(x))$$

In CNF:

$$\neg\ (\neg\forall\, x\, P(x) \vee \exists\, x\, P(x))$$
$$\neg\ (\exists\, x\, \neg\, P(x) \vee \exists\, x\, P(x))$$
$$\forall\, x\, P(x) \wedge \forall\, y\, \neg P(y)$$
$$P(x) \wedge \neg P(y)$$

a1. $P(x)$
a2. $\neg P(y)$

Null➔valid!

BARBARA:

$$\forall x.\ B(x) \rightarrow A(x)$$
$$\underline{\forall x.\ C(x) \rightarrow B(x)}$$
$$\forall x.\ C(x) \rightarrow A(x)$$

1. $\forall x.\ B(x) \rightarrow A(x)$
2. $\forall x.\ C(x) \rightarrow B(x)$

Q. $\forall x.\ C(x) \rightarrow A(x)$

¬Q. $\neg (\forall x.\ C(x) \rightarrow A(x))$

1. $\neg B(x1) \vee A(x1)$
2. $\neg C(x2) \vee B(x2)$
3. $\neg (\forall x.\ \neg C(x) \vee A(x))$

   $\exists x.\ \neg(\neg C(x) \vee A(x))$

   $\exists x.\ C(x) \wedge \neg A(x)$

   $C(S1) \wedge \neg A(S1)$

   $C(S1)\ ,\ \neg A(S1)$

3a. $C(S1)$

3b. $\neg A(S1)$

1. $\neg B(x1) \vee A(x1)$
2. $\neg C(x2) \vee B(x2)$

3a. $C(S1)$

3b. $\neg A(S1)$

4. 1+3b. x1/S1 $\neg B(S1)$
5. 4+2 x2/S1 $\neg C(S1)$
6. 5.+3a. null

DARAPTI: ??

$\forall x.\ C(x) \rightarrow A(x)$

$\underline{\forall x.\ C(x) \rightarrow B(x)}$

$\exists x.\ B(x) \wedge A(x)$

1. $\forall x.\ C(x) \rightarrow A(x)$
2. $\forall x.\ C(x) \rightarrow B(x)$

Q. $\exists x.\ B(x) \wedge A(x)$

¬Q. $\neg (\exists x.\ B(x) \wedge A(x))$

1. $\neg C(x1) \vee A(x1)$
2. $\neg C(x2) \vee B(x2)$
3. $\neg (\exists x.\ B(x) \wedge A(x))$

   $\forall x.\ \neg(B(x) \wedge A(x))$

   $\forall x.\ \neg B(x) \vee \neg A(x)$

3. $\neg B(x3) \vee \neg A(x3)$

1. $\neg C(x1) \vee A(x1)$
2. $\neg C(x2) \vee B(x2)$
3. $\neg B(x3) \vee \neg A(x3)$

4. 1+3 x1/x3 $\neg B(x1) \vee \neg C(x1)$
5. 4+2 x2/x1 $\neg C(x1)$

Null???

Példa

DARAPTI: ??????

$\forall x.\ C(x) \rightarrow A(x)$

$\forall x.\ C(x) \rightarrow B(x)$

($\exists x.\ C(x)$ hidden assumption about existence)

$\exists x.\ B(x) \wedge A(x)$

1. $\forall x.\ C(x) \rightarrow A(x)$
2. $\forall x.\ C(x) \rightarrow B(x)$
3. $\exists x.\ C(x)$

Q. $\exists x.\ B(x) \wedge A(x)$

¬Q. $\neg (\exists x.\ B(x) \wedge A(x))$

1. $\neg C(x1) \vee A(x1)$
2. $\neg C(x2) \vee B(x2)$
3. $C(S1)$
4. $\neg (\exists x.\ B(x) \wedge A(x))$

   $\forall x.\ \neg(B(x) \wedge A(x))$

   $\forall x.\ \neg B(x) \vee \neg A(x)$

4. $\neg B(x3) \vee \neg A(x3)$

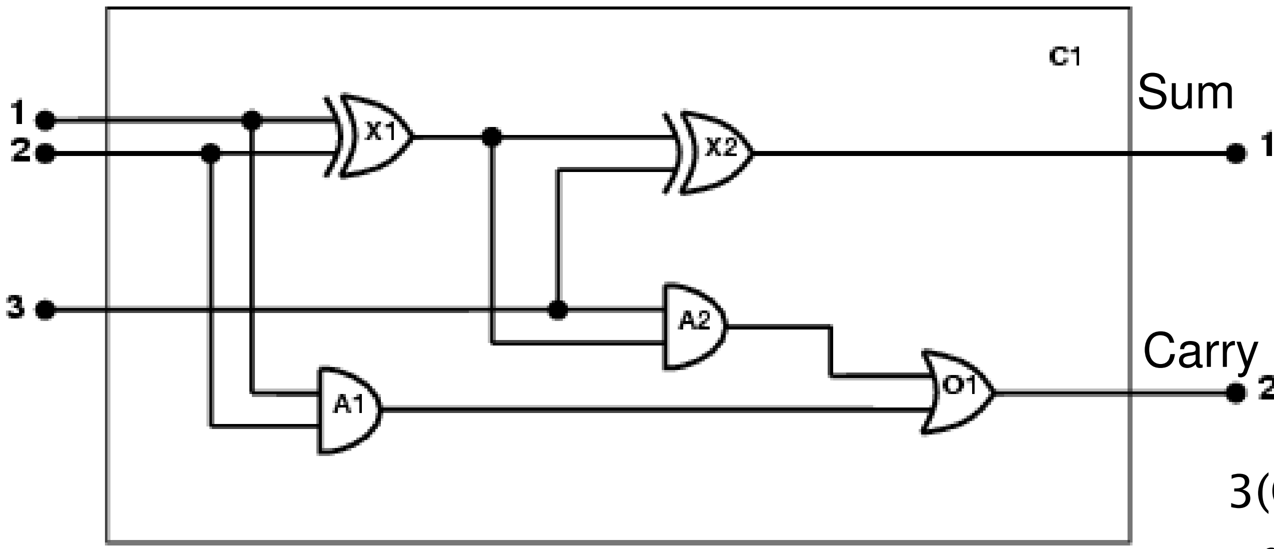
---

1. $\neg C(x1) \vee A(x1)$
2. $\neg C(x2) \vee B(x2)$
3. $C(S1)$
4. $\neg B(x3) \vee \neg A(x3)$

4. 1+3 x1/x3 $\neg B(x1) \vee \neg C(x1)$
5. 4+2 x2/x1 $\neg C(x1)$
6. 5+3 x1/S1 null

The electronic circuits domain

One-bit full adder: I1+I2 + I3 (as carry)



Input			Output	
3(C)	1	2	O(1) Sum	O(2) Carr
0	0	0	0	0
1	0	0	1	0
0	1	0	1	0
1	1	0	0	1
0	0	1	1	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1

The electronic circuits domain

1. Identify the task

2.

- Does the circuit actually add properly? (circuit verification)
-

2. Assemble the relevant knowledge

3.

- Composed of wires and gates; Types of gates (AND, OR, XOR, NOT)
-
- Irrelevant: size, shape, color, cost of gates
-

3. Decide on a vocabulary

4.

- Alternatives:
-

Type(X_1) = XOR

Type(X_1, XOR)
XOR(X_1)

The electronic circuits domain

4. Encode general knowledge of the domain
5.
   - $\forall t_1,t_2\ Connected(t_1, t_2) \Rightarrow Signal(t_1) = Signal(t_2)$
   - $\forall t\ Signal(t) = 1 \vee Signal(t) = 0$
   -
   - $1 \neq 0$
   -
   - $\forall t_1,t_2\ Connected(t_1, t_2) \Rightarrow Connected(t_2, t_1)$
   -
   - $\forall g\ Type(g) = OR \Rightarrow Signal(Out(1,g)) = 1 \Leftrightarrow \exists n\ Signal(In(n,g)) = 1$
   -
   - $\forall g\ Type(g) = AND \Rightarrow Signal(Out(1,g)) = 0 \Leftrightarrow \exists n\ Signal(In(n,g)) = 0$
   -
   - $\forall g\ Type(g) = XOR \Rightarrow Signal(Out(1,g)) = 1 \Leftrightarrow Signal(In(1,g)) \neq Signal(In(2,g))$
   -
   - $\forall g\ Type(g) = NOT \Rightarrow Signal(Out(1,g)) \neq Signal(In(1,g))$
   -

The electronic circuits domain

5. Encode the specific problem instance
6.

$Type(X_1) = XOR$
$Type(X_2) = XOR$
$Type(A_1) = AND$
$Type(A_2) = AND$
$Type(O_1) = OR$

$Connected(Out(1,X_1),In(1,X_2))$
$Connected(Out(1,X_1),In(2,A_2))$
$Connected(Out(1,A_2),In(1,O_1))$
$Connected(Out(1,A_1),In(2,O_1))$
$Connected(Out(1,X_2),Out(1,C_1))$
$Connected(Out(1,O_1),Out(2,C_1))$

$Connected(In(1,C_1),In(1,X_1))$
$Connected(In(1,C_1),In(1,A_1))$
$Connected(In(2,C_1),In(2,X_1))$
$Connected(In(2,C_1),In(2,A_1))$
$Connected(In(3,C_1),In(2,X_2))$
$Connected(In(3,C_1),In(1,A_2))$

The electronic circuits domain

6. Pose queries to the inference procedure
7.

What are the possible sets of values of all the terminals for the adder circuit?

$\exists i_1,i_2,i_3,o_1,o_2$ Signal(In(1,C_1)) $= i_1 \wedge$ Signal(In(2,C_1)) $= i_2 \wedge$ Signal(In(3,C_1)) $= i_3 \wedge$ Signal(Out(1,C_1)) $= o_1 \wedge$ Signal(Out(2,C_1)) $= o_2$

Input			Output	
3(C)	1	2	O(1) Sum	O(2) Carr
0	0	0	0	0
1	0	0	1	0
0	1	0	1	0
1	1	0	0	1
0	0	1	1	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1

Set theory in FOL

- $\forall s\ Set(s) \Leftrightarrow (s = \{\}) \vee (\exists x,s_2\ Set(s_2) \wedge s = \{x|s_2\})$
-
- $\neg\exists x,s\ \{x|s\} = \{\}$
-
- $\forall x,s\ x \in s \Leftrightarrow s = \{x|s\}$
-
- $\forall x,s\ x \in s \Leftrightarrow [\ \exists y,s_2\}\ (s = \{y|s_2\} \wedge (x = y \vee x \in s_2))]$
-
- $\forall s_1,s_2\ s_1 \subseteq s_2 \Leftrightarrow (\forall x\ x \in s_1 \Rightarrow x \in s_2)$
- $\forall s_1,s_2\ (s_1 = s_2) \Leftrightarrow (s_1 \subseteq s_2 \wedge s_2 \subseteq s_1)$
-
- $\forall x,s_1,s_2\ x \in (s_1 \cap s_2) \Leftrightarrow (x \in s_1 \wedge x \in s_2)$
-
- $\forall x,s_1,s_2\ x \in (s_1 \cup s_2) \Leftrightarrow (x \in s_1 \vee x \in s_2)$

- (For a naiv definition, Russell's paradox: Let *R* be the set of all sets that are not members of themselves. If *R* is not a member of itself, then its definition dictates that it must contain itself, and if it contains itself, then it contradicts its own definition as the set of all sets that are not members of themselves….)

Predicate Logic: Caesar&Marcus

1. Marcus was a Man.
2. Marcus was a Pompeian.
3. All Pompeians were Roman.
4. Caesar was a ruler.
5. All Pompeians were either loyal to Caesar or hated him.
6. Every one is loyal to someone.
7. People only try to assassinate rulers they are not loyal to.
8. Marcus tried to assassinate Caesar.

Adapted from Rich&Knight: Artificial intelligence, 1991

Caesar&Marcus: formalization

1. Marcus was a Man.

Man(Marcus)

2. Marcus was a Pompeian.

Pompeian(Marcus)

3. All Pompeians were Roman.

$\forall x$ Pompeian(x) $\Rightarrow$ Roman (x)

4. Caesar was a ruler.

Ruler(Caesar)

5. All Romans were either loyal to Caesar or hated him.

$\forall x$ Romans(x) $\Rightarrow$ Loyalto(x,Caesar) $\vee$ Hate(x,Caesar)

6. Every one is loyal to someone.

$\forall x \exists y$ Loyalto(x,y)

7. People only try to assassinate rulers they are not loyal to.

$\forall x \forall y$ Roman(x) $\wedge$ Ruler(y) $\wedge$ Tryassassinate(x,y) $\Rightarrow \neg$Loyalto(x,y)

8. Marcus tried to assassinate Caesar.

Tryassassinate(Marcus,Caesar)

Caesar&Marcus: possible questions

A. Was Marcus loyal to Caesar?
B. Did Marcus hate Caesar?
C. Who hate Caesar?
D. …

Caesar&Marcus: **Modus Ponens** Loyalty

Modus Ponens (backward):

$\neg$ *Loyalto* (*Marcus*, *Caesar*)
==(from 7, it follows)==>
Pompeian (*Marcus*) $\wedge$ *Ruler* (*Caesar*) $\wedge$ *Tryassassinate* (*Marcus*, *Caesar*)

==(4)==> *Pompeian* (*Marcus*) $\wedge$ *Tryassassinate* (*Marcus*, *Caesar*)

==(8)==> *Pompeian* (*Marcus*)

==(9)==> *Man* (*Marcus*)

==(1)==> *TRUE*

Caesar&Marcus: beyond Horn-clauses (and Modus Ponens)

9. Romans, who know Marcus, either hate Caesar or believe that if somebody hates someone then he is fool.

$\forall x\, [Roman\,(x) \wedge Know\,(x, M)] \rightarrow$
$[Hate\,(x, C) \vee (\forall y\, (\exists\, z\, Hate\,(y, z)) \rightarrow ThinkToBeFool(x, y))]$

Not a Horn-clause! ➔ Modus Ponens is not complete!

Caesar&Marcus: conversion to CNF

$\forall x([Roman(x) \wedge Know(x, Marcus)] \rightarrow$
$[Hate(x, Caesar) \vee (\forall y [(\exists z\, Hate(y, z)) \rightarrow ThinkToBeFool(x, y)])])$

$\forall x \neg [Roman(x) \wedge Know(x, Marcus)] \vee$
$[Hate(x, Caesar) \vee (\forall y \neg (\exists z\, Hate(y, z)) \vee ThinkToBeFool(x, y))]$

$\forall x [\neg Roman(x) \vee \neg Know(x, Marcus)] \vee$
$[Hate(x, Caesar) \vee (\forall y \forall z \neg Hate(y, z) \vee ThinkToBeFool(x, y))]$

$\forall x \forall y \forall z [\neg Roman(x) \vee \neg Know(x, Marcus)] \vee$
$[Hate(x, Caesar) \vee [\neg Hate(y, z) \vee ThinkToBeFool(x, y))]$

$[\neg Roman(x) \vee \neg Know(x, Marcus)] \vee [Hate(x, Caesar)$
$\vee [\neg Hate(y, z) \vee ThinkToBeFool(x, y))]$

$\neg Roman(x) \vee \neg Know(x, Marcus) \vee Hate(x, Caesar) \vee$
$\neg Hate(y, z) \vee ThinkToBeFool(x, y))$

Caesar&Marcus: possible questions

A. Was Marcus loyal to Caesar?
B. Did Marcus hate Caesar?
C. Who hate Caesar?
D. …

¬ ***Roman*** **(x2) ∨** ***LoyalTo*** **(x2,** ***C*****) ∨** ***Hate*** **(x2,** ***C*****)**

¬ ***Hate*** **(*****M*****,** ***C*****)**

M **/x2**

¬***Roman*** **(*****M*****) ∨** ***LoyalTo*** **(*****M*****,** ***C*****)**

¬ ***Pompeian*****(x1) ∨** ***Roman*** **(x1)**

M **/x1**

¬ ***Pompeian*** **(*****M*****) ∨** ***LoyalTo*** **(*****M*****,** ***C*****)**

Pompeian **(*****M*****)**

¬ ***Man*** **(x4) ∨ ¬** ***Ruler*** **(y1) ∨**
¬ ***Tryassassinate*** **(x4, y1) ∨ ¬** ***LoyalTo*** **(x4, y1)**

LoyalTo **(*****M*****,** ***C*****)**

M **/x4,** ***C*** **/y1**

Man **(*****M*****)**

¬ ***Man*** **(*****M*****) ∨ ¬** ***Ruler*** **(*****C*****) ∨ ¬** ***Tryassassinate*** **(*****M*****,** ***C*****)**

Ruler **(*****C*****)**

¬ ***Ruler*** **(*****C*****) ∨ ¬** ***Tryassassinate*** **(*****M*****,** ***C*****)**

Tryassassinate **(*****M*****,** ***C*****)**

¬ ***Tryassassinate*** **(*****M*****,** ***C*****)**

nil

Caesar&Marcus: possible questions

A. Was Marcus loyal to Caesar?
B. Did Marcus hate Caesar?
C. Who hates Caesar?

➔ Is there anybody who hates Caesar? (+ Instantiation)

- Query:

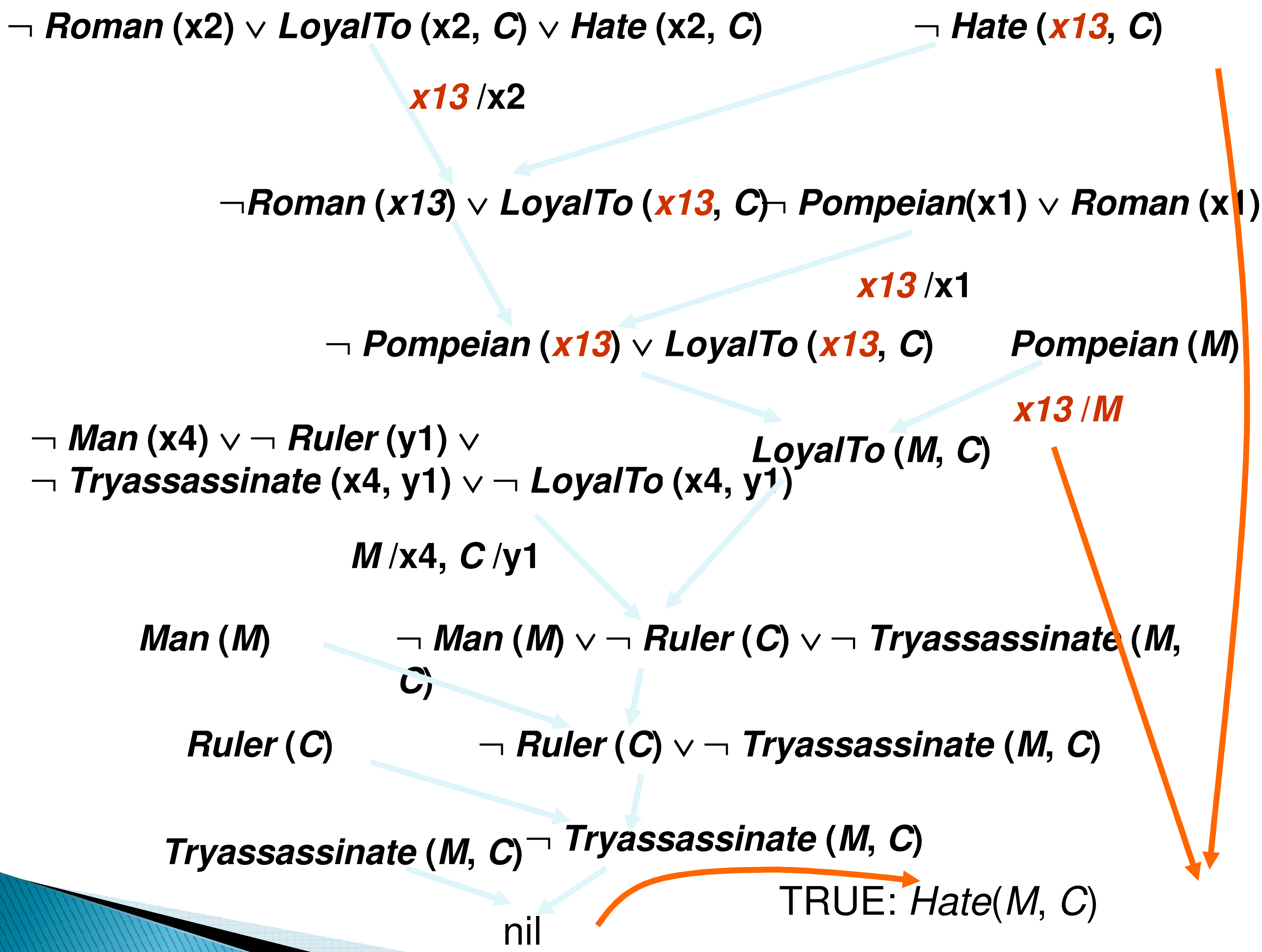
$\exists$ x Hate (x, Caesar)

Negation

$\neg$ ($\exists$ x Hate (x, Caesar))

$\forall$ x $\neg$Hate (x, Caesar)

$\neg$Hate (x13, Caesar)


¬ Roman (x2) ∨ LoyalTo (x2, C) ∨ Hate (x2, C)
¬ Hate (x13, C)
x13 /x2
¬Roman (x13) ∨ LoyalTo (x13, C)
¬ Pompeian(x1) ∨ Roman (x1)
x13 /x1
¬ Pompeian (x13) ∨ LoyalTo (x13, C)
Pompeian (M)
x13 /M
¬ Man (x4) ∨ ¬ Ruler (y1) ∨
¬ Tryassassinate (x4, y1) ∨ ¬ LoyalTo (x4, y1)
LoyalTo (M, C)
M /x4, C /y1
Man (M)
¬ Man (M) ∨ ¬ Ruler (C) ∨ ¬ Tryassassinate (M, C)
Ruler (C)
¬ Ruler (C) ∨ ¬ Tryassassinate (M, C)
Tryassassinate (M, C)
¬ Tryassassinate (M, C)
nil
TRUE: Hate(M, C)

Representation&inference in FOL III.

- Tim, Neal és Elisabeth are Club mMans. Club mMans are hiker or skier. Hikers do not like rain, skiers like snow. Elisabeth does not like what Tim likes, and she likes what Tim does not like. Tim likes rain and snow.
  - Is there any club mMan who is hiker, but not skier?
  - Answer this question with resolution!
-
- A solution:
- H(x) denotes that x is Hiker, S(x) denotes that x is a skier, L(x,y) denotes that x likes y (if x is a club mMan and y denotes rain or snow):
  - $\forall$x. S(x) v H(x)
  - $\neg\exists$x. H(x) $\wedge$ L(x, Rain), which is $\forall$x. $\neg$H(x) v $\neg$L(x, Rain)
  - $\forall$x. S(x) $\rightarrow$ L(x, Snow), which is $\forall$x. $\neg$S(x) v L(x, Snow)
  - $\forall$y. L(Elisabeth, y) $\leftrightarrow$ $\neg$L(Tim, y), which is 4a: $\forall$y. $\neg$L(Elisabeth, y) v $\neg$L(Tim, y) and 4b...
  - L(Tim, Rain)
  - L(Tim, Snow)
  - **question**: **$\exists$x. H(x) $\wedge$ $\neg$S(x), which is $\forall$x $\neg$H(x) v S(x)**

FERISON

- The FERISON syllogism is as follows:
  - $\forall x.\ C(x) \rightarrow \neg A(x)$
  - $\underline{\exists x.\ C(x) \wedge B(x)}$
  - $\exists x.\ C(x) \wedge \neg A(x)$

- Is it sound?

FESTIMO

- The FESTIMO syllogism is as follows:
  - $\forall x.\ B(x) \rightarrow \neg A(x)$
  - $\underline{\exists x.\ C(x) \wedge A(x)}$
  - $\exists x.\ C(x) \wedge \neg B(x)$
- Is it sound?

DARII

- The DARII syllogism is as follows:
  - $\forall x.\ B(x) \rightarrow A(x)$
  - $\underline{\exists x.\ C(x) \wedge B(x)}$
  - $\exists x.\ C(x) \wedge A(x)$
- Prove its validity by resolution.

Representation&inference in FOL II.

- Prove with resolution that the statements
  - $\forall x.\ Mobile(x) \rightarrow Gadget\ (x)$,
  - $\exists x.\ Mobile(x) \wedge Intelligent(x)$
  - entail that $\exists x.\ Gadget\ (x) \wedge Intelligent(x)$

The Secret Chamber

While excavating an ancient Puzzlanian crypt, you discover an unusual column. The column has four narrow holes bored into it, all at the same height, evenly spaced around the column, just large enough for a huPompeian hand. Reading the inscriptions above and below the holes, you realize that this column is part of a complex mechanism that will open a secret chamber. Out of sight within each hole is a switch that can either be up or down; when all the switches are in the same position, the secret chamber will open before you. The column is small enough that you can reach all the way around, so using both hands you could flip any two switches at the same time. Here's the tricky part: As soon as your hand leaves a hole, the column will rapidly spin for a random number of quarter rotations. If you're not careful, you might lose a hand. But you can flip two switches at once, then quickly pull both hands out at the same time. What strategy can you use to open the chamber in a finite, and preferably small, number of attempts? Assume you cannot discriminate between holes after a spin without reaching into the column.

http://www.greylabyrinth.com/puzzle/puzzle102