Jay Urbain

(

Propositional Theorem Proving

Entailment - one sentence logically follows from another.

- In logic, entailment is a relation between sets of sentences & a sentence.
- Defined in terms of truth preserving.
- Some sentence(s) T entails sentence A iff it is necessary that A be true whenever T is true.

- Apply rules of inference directly to sentences in KB to construct a proof of entailment. (w/o having to construct complete model, i.e., truth table).

Inference and Proofs

Modus Ponens (mode that affirms)

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

Meaning: If sentences of form $\alpha \Rightarrow \beta$ & α are given, then β can be inferred

Proof by resolution - ~~Model Checking~~

Unit Resolution Rule

$$\frac{l_1 \vee \cdots l_k, \quad m}{l_1 \vee \cdots \vee l_{k-1} \vee l_{i+1} \cdots l_k.}$$

- l_i & m are complementary literals

e.g $\dfrac{\cancel{P11} \vee P31, \quad \neg \cancel{P11} \vee \neg P22}{P31 \vee \neg P22}$

To use resolution require Conjunctive Normal Form

- Every sentence in PL is logically equivalent to a conjunction of clauses i.e., CNF.

e.g. Convert $B11 \Leftrightarrow (P12 \vee P21)$ into CNF

1.) Biconditional Elimination $\alpha \Leftrightarrow \beta$ w/ $\alpha \Rightarrow \beta, \beta \Rightarrow \alpha$

$(B11) \Rightarrow (P12 \vee P21) \wedge (P12 \vee P21) \Rightarrow B11$

2.) Implication elimination. $\Rightarrow$, replace $\alpha \Rightarrow \beta$ w/ $\neg\alpha \vee \beta$

$(\neg B11 \vee P12 \vee P21) \wedge (\neg(P12 \vee P21) \vee B11)$

3) CNF requires $\neg$ to appear adj. to literals.

$$\begin{cases} \neg(\neg\alpha) \equiv \alpha \\ \neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta) & \text{Demorgan } \wedge \\ \neg(\alpha \vee \beta \equiv (\neg\alpha \wedge \neg\beta) & \text{DeMorgan } \vee \end{cases}$$

$(\neg B11 \vee P12 \vee P21) \wedge ((\neg P12 \wedge \neg P21) \vee B11)$

$(\neg B11 \vee P12 \vee P21) \wedge (\neg P12 \vee B11) \wedge (\neg P21 \vee B1$

Resolution Algo.

- Proof by contradiction.
- To show $KB \models \alpha$, show $KB \wedge \neg\alpha$ is <u>not</u> satisfiable.
- Convert $KB \wedge \neg\alpha$ to CNF
- Apply resolution algo. to eliminate complementary literals.

E.g. $KB = [B11 \Leftrightarrow (P12 \vee P21)] \wedge \neg B11$, show $\neg P12$

- convert to CNF, bicond elimination, ~~implic~~ implic

$[\neg B11 \vee (P12 \vee P21)] \wedge [\neg(P12 \vee P21) \vee B11] \wedge \neg B11$

De Morgan

$(\neg P12 \wedge \neg P21) \vee B11$

Distributivity of $\vee$ over $\wedge$

$[\cancel{\neg B11} \vee P12 \vee \cancel{P21}] \wedge [\neg P12 \vee B11] \wedge [\neg P21 \vee \cancel{B11}] \wedge [\neg B11]$

$\boxed{P12}$ $\quad \boxed{\neg P21}$ $\quad \boxed{\neg P12} \longleftrightarrow \boxed{P12}$

unsatisfiable

$\neg \cancel{P12} \vee$

Horn Clauses & Definite Clauses

Definite clause: restricted form of CNF to ↑ perf. & ↓ complexity.

- disjunction of literals in wh/ exactly one is positive.

  e.g. $\neg L11 \vee \neg Breeze \vee B11$ is DC

  $\neg B11 \vee P12 \vee P21$ is not

Horn Clause: slightly more general

disjunction of literals of wh/ at most one is positive. Allows facts

So all definite clauses are Horn clauses, as are clauses w/ no positive literals. i.e., goal clauses.

1.) Every definite clause can be written as an implication whose premise is a conjunction of pos. literals and whose conclusion is a pos. literal

$\alpha \Rightarrow \beta \equiv \neg\alpha \vee \beta$

E.g. $(\neg L11 \vee \neg Breeze \vee B11)$

$\equiv (L11 \wedge Breeze) \Rightarrow B11$

i.e. ~~$\alpha \Rightarrow \beta$~~

$\alpha \Rightarrow \beta \equiv \neg\alpha \vee \beta$

Single literal, L11 is a fact.

2) Inference in Horn clauses can be performed w/ FC & BC

3.) Deciding entailment w/ Horn clauses can be done in linear time!

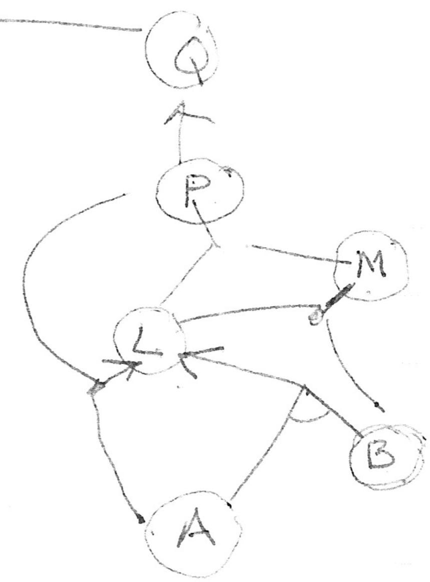
Forward and Backward Chaining

- FC determines if a single propositional symbol q – the query – is entailed by the KB of definite clauses.
- Starts w/ known facts
- If all the premises of an implication are known, then its conclusion is added to the list of known facts.

  Eg. If $B_{01} \wedge B_{10} \Rightarrow P_{00}$

  if B_{01} and B_{10} known, P_{00} can be added as fact to KB.
- Process continues until the query q is added ~~of~~ until no further inferences can be made.
- Note: Just because q is not entailed by KB does not mean it is ~~not~~ FALSE i.e ~~$\neg q$~~ you need $\neg q$
- Runs in linear time wrt length of KB

	count	
$P \Rightarrow Q$	1	
$L \wedge M \Rightarrow P$	2 ∅	
$B \wedge L \Rightarrow M$	2 1 ∅	
$A \wedge P \Rightarrow L$	2 1	
$A \wedge B \Rightarrow L$	~~2~~ 1 ∅	
A	∅	
B	∅	



PL FCEntails (KB, q) returns T or F

Count ← table where count[c] is number of symbols in c's premise

inferred ← table of symbols initially F

agenda ← Queue of symbols known to be T in KB (A and B.) but not yet processed

<u>while</u> agenda not empty. <u>do</u>

p ← PoP(agenda)

<u>if</u> p == q <u>return</u> true

<u>if</u> inferred [p] == false then

inferred [p] = true

for each clause C in KB

where p is in C. Premise do

decrement count [c].

<u>if</u> count [c] == ∅ <u>then</u>

add c. conclusion to agenda.

return <u>false</u>!

agenda	inferred
A	A = ~~F~~ T
B L	B = ~~F~~ T
	L = ~~F~~ T
	P = F
	M = F T