Oct 8 2013

LECTURE #6

41

- Satisfiability
  $X \models \alpha$
  (Model Checking)

- Provability
  $X \vdash \alpha$
  (Theorem Proving)

What's the relationship between them?

SOUNDNESS

$X \vdash \alpha$ implies $X \models \alpha$.
(Proof by induction on formula and rule.)

COMPLETENESS

$X \nvdash \alpha$ implies $X \not\models \alpha$.

First, strengthen the statement

$X \nvdash \bot \wedge X \nvdash \alpha$ implies $X \not\models \alpha$.

(X = consistent).

(A)

$$\dfrac{\dfrac{\dfrac{X \vdash \bot}{X \vdash \beta \wedge \neg\beta}}{X \vdash \beta \qquad X \vdash \neg\beta}}{X \vdash \alpha} \ (\forall \alpha)$$

(B)

$$\dfrac{\dfrac{X, \neg\alpha \vdash \bot}{X, \neg\alpha \vdash \alpha} \qquad \dfrac{\alpha \vdash \alpha}{X, \alpha \vdash \alpha}}{X \vdash \alpha}$$

$X \nvdash \alpha,\ X \nvdash \bot \Rightarrow X \nvdash \bot,\ X, \neg\alpha \nvdash \bot$

$\Rightarrow X \cup \{\neg\alpha\}$ = consistent

implies $X \not\models \alpha$.

$Y \supseteq X \cup \{\neg\alpha\}$

Maximally consistent superset of X containing $\neg\alpha$

Y = Satisfiable $\Rightarrow$ X = Satisfiable.

But also $X \cup \{\neg\alpha\}$ = satisfiable.

$\boxed{\text{Completeness Proof}}$

$X \nvdash \alpha$ ($\wedge$ X = consistent)

$\Rightarrow$ $X \cup \{\neg\alpha\}$ = consistent

$\Rightarrow$ $\exists Y$ $Y \supseteq X \cup \{\neg\alpha\}$ & Y = Maximally consistent

$\boxed{\text{Lindenbaum's Lemma}}$ $\boxed{\text{TBS}}$ To be shown!

$\boxed{\text{TBS}}$

$\Rightarrow$ Y = Satisfiable

$\Rightarrow$ $X \cup \{\neg\alpha\}$ = Satisfiable

$\Rightarrow$ $X \not\models \alpha$ $\square$

Defn. (a) $X \subseteq \mathcal{F}$ is called <u>inconsistent</u> if $X \vdash \alpha$ $\forall_{\alpha \in \mathcal{F}}$; otherwise, <u>consistent</u>.

(b) $Y \subseteq \mathcal{F}$ is called <u>maximally</u> <u>consistent</u> if Y = consistent but each $Z \supsetneq Y$ is inconsistent.

$\boxed{\text{Lindenbaum's Lemma:}}$

Every consistent set $X \subseteq \mathcal{F}$ can be extended to a maximally consistent set $X' \supseteq X$.

$\boxed{\text{Satisfiability Lemma:}}$

Every maximally consistent set Y is satisfiable.

$\square$

Lindenbaum's Lemma:

proof: Let H be the set of all consistent $Y \supseteq X$, partially ordered with respect to $\subseteq$ relation.

$$H = \{ Y \mid Y \supseteq X \ \& \ Y \nvdash \bot \}$$

(a) $H \neq \phi$ $\quad (\because X \in H)$

(b) $\exists_{K \subseteq H}$ $\quad K =$ chain. $\quad \{$ i.e. $\forall Y, Z \in K \quad Y \subseteq Z$ or $Z \subseteq Y$.

$\mathcal{U} := \bigcup K =$ upperbound for the chain K.

- $\mathcal{U} \nvdash \bot$ $\quad$ ($\mathcal{U}$ = consistent)

  suppose not: $\mathcal{U} \vdash \bot$

  $\Rightarrow \mathcal{U}_0 \vdash \bot \quad \mathcal{U}_0 =$ finite $\quad \therefore \mathcal{U}_0 = \{\alpha_1, \dots, \alpha_n\}$

  $\therefore \quad \alpha_1, \dots, \alpha_n \vdash \bot$

  $\alpha_i \in Y_i \in K$

  Let Y be the biggest among $Y_1, \dots, Y_n$.

  $\Rightarrow \{\alpha_1, \dots, \alpha_n\} \subseteq Y$

  $$\frac{\alpha_1 \dots \alpha_n \vdash \bot}{Y \vdash \bot} \ \text{(by MR)}$$

  $\Rightarrow Y \notin H \Rightarrow \#$

(c) By Zorn's Lemma: H has a maximal element, X'.

$X' \supseteq X$ and X' = maximally consistent.

$\square$

Satisfiability Lemma:

Υ = maximally consistent
$\Rightarrow \Upsilon$ = satisfiable.

proof:
Define a propositional valuation $\omega : PV \to \{0,1\}$
$\omega \models p$ iff $\Upsilon \vdash p$ $\forall p$ = prime variable.

Show that $\forall_\alpha$ $\Upsilon \vdash \alpha$ iff $\omega \models \alpha$.
(i.e. ω = model for Υ)
(i.e. Υ = satisfiable).

$\Upsilon \vdash \alpha \wedge \beta$ iff $\Upsilon \vdash \alpha, \beta$ ($\because \wedge 1$ & $\wedge 2$)
iff $\omega \models \alpha$ and $\omega \models \beta$
iff $\omega \models \alpha \wedge \beta$.

$\Upsilon \vdash \neg\alpha$ iff $\Upsilon \nvdash \alpha$ ($\because$ maximality of Υ)
iff $\omega \not\models \alpha$
iff $\omega \models \neg\alpha$

$\Rightarrow$ $\dfrac{\Upsilon \vdash \neg\alpha \ \wedge\ \Upsilon \vdash \alpha}{\Upsilon \vdash \bot}$

$\Leftarrow$ $\Upsilon \nvdash \alpha$
$\Rightarrow \Upsilon \cup \{\neg\alpha\}$
is a consistent extension of Υ
$\Rightarrow \neg\alpha \in \Upsilon$
$\Rightarrow \Upsilon \vdash \neg\alpha$.
$\square$

Horn Clauses:

A Horn clause is a disjunction of literals in which all or nearly all of the literals are complemented.

(At most one of its literals is pure.)

Example: $x \equiv \quad T \Rightarrow x \qquad (\omega : x \mapsto 1)$

$\bar{x} \vee \bar{y} \equiv x \wedge y \Rightarrow \bot \qquad (\omega : x \mapsto 0 \text{ or } y \mapsto 0)$

$x \vee \bar{y} \vee \bar{z} \equiv y \wedge z \Rightarrow x \qquad (\omega : x \mapsto 1 \text{ or } y \mapsto 0 \text{ or } z \mapsto 0)$

HORNSAT ALGORITHM (GREEDY)

(1) Initialize Assign all variables false.
(Thus initially all clauses of the following form will be satisfied.

$$x \wedge y \Rightarrow \bot$$

$$u \wedge v \wedge w \wedge x \wedge y \Rightarrow z.$$

But not

$$T \Rightarrow x$$

(2) Update

For a clauses whose r.h.s ~~are~~ not satisfied choose one & FLIP the truth assignment to true.

(3) Reevalute all clauses and repeat (2), until no ~~varis~~ variable can be FLIPPED (unsatisfiable) OR a satisfiable assignment has been found. □

Clauses $= m$, # variables $= n$.
Complexity $= O(mn)$.

PROLOG PROGRAM

Defn: (1) A Horn clause is a clause that contains at most one positive literal.

(2) A program clause is one that contains exactly one positive literal.

$$A :- B_1, B_2, \dots, B_n.$$

(3) If a program clause contains some negative literals it is called a rule $(n > 0)$

(4) A unit clause (fact) is one that consists of exactly one positive literal.

$$A. \quad \text{or} \quad A :-$$

(5) A goal clause is one that contains no positive literals.

$$?-$$

(6) A PROLOG program is a set of clauses containing only program clauses. $\{$ Rules or Facts.

Lemma: If a set of Horn clauses S is unsatisfiable then S must contain at least one fact and one goal clause.

proof: If S contains no fact, then assign every prime variable false. $S \Rightarrow$ Satisfiable.

If S contains no goal clause then assign every prime variable true, $S \Rightarrow$ Satisfiable.

$\square$

General view of a Prolog Program:

<u>Given</u>: A collection of facts and rules. ≡ Program P.

<u>Deduce</u>: If a conjunction of some facts $\{q_1 \wedge q_2 \wedge \cdots \wedge q_n\}$

?- $q_1, q_2, \ldots q_n$

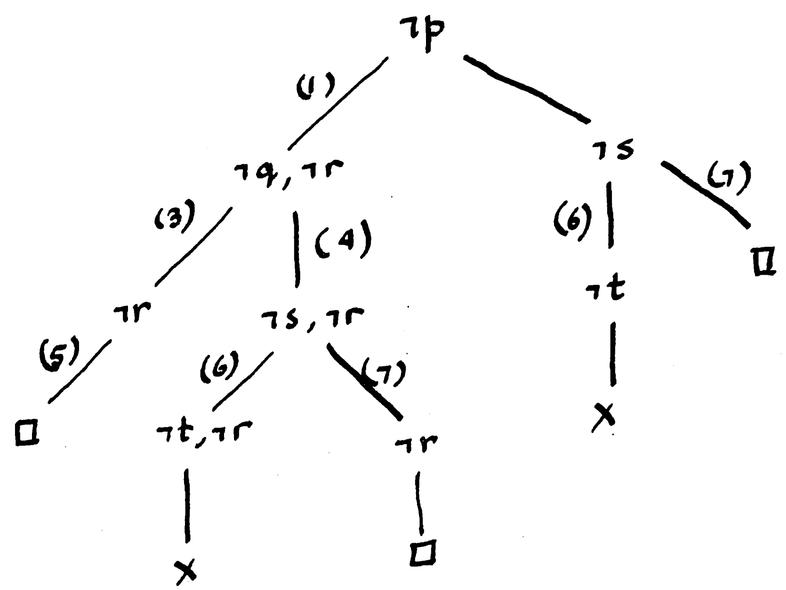
is a consequence of P.

$G = \{\neg q_1, \neg q_2, \cdots, \neg q_n\}$

<u>Lemma</u>: q_i's are consequences of P iff $P \cup \{G\}$ is unsatisfiable.

<u>proof</u>: H.W.

```
(1)   p :- q, r.
(2)   p :- s.
(3)   q.
(4)   q :- s.
(5)   r.
(6)   s :- t.
(7)   s.
      ?- p
```



More Complex Prolog Program (Need 1st order logic).

```
parent (X,Y) :- mother (X,Y).                     male(sam).
parent (X,Y) :- father (X,Y).                     male (tom).
daughter (X,Y) :- mother (Y,X), female(X).        female (kim).
son (X,Y) :- mother (Y,X), male(X).               father(bud, kim).
child (X,Y) :- son(X,Y).                          female (jane).
child (X,Y) :- daughter(X,Y).                     mother (jane, sam).
daughter(X,Y) :- father (Y,X), female (X)              :
son (X,Y) :- father (Y,X), male(X)                ?- parent (bud, sam)
```