PROPOSITIONAL LOGIC

formula

f	$::=$	v	symbol
	$\mid$	$f \wedge f$	and
	$\mid$	$f \vee f$	or
	$\mid$	$f \leftrightarrow f$	if and only if
	$\mid$	$f \rightarrow f$	implies
	$\mid$	$\neg f$	not

TRUTH VALUES

To assign a truth value to a propositional formula, we have to assign truth values to each of its atoms (symbols).

a	b	$a \wedge b$	$a \vee b$	$a \leftrightarrow b$	$a \rightarrow b$	$\neg a$
false	false	F	F	T	T	T
false	true	F	T	F	T	T
true	false	F	T	F	F	F
true	true	T	T	T	T	F

A tautology is a formula, true for all possible assignments.

PROPOSITIONAL LAWS (EXAMPLES)

The contrapositive law:

$$(p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p)$$

De Morgan's law:

$$\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$$

First-Order Predicate Calculus

$$
\begin{array}{rcll}
f & ::= & p(v_1, \ldots, v_n) & \text{predicate} \\
 & \mid & v = f(v_1, \ldots, v_n) & \text{equality} \\
 & \mid & v_1 = v_2 & \\
 & \mid & f_1 \wedge f_2 \mid f_1 \vee f_2 \mid f_1 \leftrightarrow f_2 \mid f_1 \rightarrow f_2 \mid \neg f & \\
 & \mid & \forall v.\, f & \text{universal} \\
 & \mid & \exists v.\, f & \text{existential}
\end{array}
$$

PREDICATE CALCULUS

In mathematical logic, a predicate is a function that maps constants or variables to true and false.

Predicate calculus enables reasoning about propositions. e.g.:

$$\forall c\,[\text{rainy}(c) \wedge \text{cold}(c) \rightarrow \text{snowy}(c)]$$

Quantifier
$(\forall, \exists)$

Operators
$(\wedge, \vee, \neg, \rightarrow, \leftrightarrow)$

QUANTIFIERS

Universal ($\forall$) quantifier indicates that the proposition is true for all variable's values.

Existential ($\exists$) quantifier indicates that the proposition is true for at least one value of the variable.

$$\forall A\, \forall B\, [(\exists C\, [takes(A,C) \wedge takes(B,C)]) \rightarrow classmates(A,B)]$$

STRUCTURAL CONGRUENCE

$$(P_1 \rightarrow P_2) \equiv (\neg P_1 \vee P_2)$$

$$(\neg \exists x [P(x)]) \equiv (\forall x [\neg P(x)])$$

$$\neg (P_1 \wedge P_2) \equiv \neg P_1 \vee \neg P_2$$

$$\neg (P_1 \vee P_2) \equiv \neg P_1 \wedge \neg P_2$$

$$\neg \neg P \equiv P$$

$$(P_1 \leftrightarrow P_2) \equiv (P_1 \rightarrow P_2) \wedge (P_2 \rightarrow P_1)$$

$$(\neg \forall x [P(x)]) \equiv \exists x [\neg P(x)]$$

$$(P_1 \leftarrow P_2) \equiv (P_1 \vee \neg P_2)$$

$$\neg (\neg P_1 \vee \neg P_2) \equiv P_1 \wedge P_2$$

CLAUSAL FORM

- Looking for a minimal kernel appropriate for theorem proving.
- Propositions are transformed into normal form by using structural congruence relationship.
- One popular normal form candidate is clausal form.
- Clocksin and Mellish ('94) 5-step procedure

EXAMPLE

$$\forall A\,[\neg student(A) \rightarrow (\neg dorm\_resident(A) \wedge \neg \exists B\,[takes(A,B) \wedge class(B)])\,]$$

① eliminate implication ($\rightarrow$) and equivalence ($\leftrightarrow$

$$\forall A\,[student(A) \vee (\neg dorm\_resident(A) \wedge \neg \exists B\,[takes(A,B) \wedge class(B)]\,)\,]$$

② move negation inward, to individual terms

$$\forall A\,[student(A) \vee (\neg dorm\_resident(A) \wedge \forall B\,[\neg(takes(A,B) \wedge class(B))]\,)\,]$$

$$\equiv \forall A\,[student(A) \vee (\neg dorm\_resident(A) \wedge \forall B\,[\neg takes(A,B) \vee \neg class(B)]\,)\,]$$

③ Skolemization : eliminate existential quantifiers ($\exists$).

EXAMPLE (continued)

(4) Move univerial quantifiers to top-level and make implicit (all variables are universally quantified).

$$student(A) \vee (\neg dorm\_resident(A) \wedge (\neg takes(A,B) \vee \neg class(B)))$$

(5) Use distributive, associative, and commutative rules of $\vee$, $\wedge$, $\neg$, to move into conjuctive normal form

$$(student(A) \vee \neg dorm\_resident(A)) \wedge$$

$$(student(A) \vee \neg takes(A,B) \vee \neg class(B))$$

(a conjuction of disjunctions) or "clauses".

clauses

CLAUSAL FORM TO PROLOG

6. Use commutativity of $\vee$ to move negated terms to the right of each clause.

7. Use $P \vee \neg Q \equiv Q \rightarrow P \equiv P \leftarrow Q$

   $(student(A) \leftarrow \neg(\neg dorm\_resident(A))) \wedge$

   $(student(A) \leftarrow \neg(\neg takes(A,B) \vee \neg class(B)))$

8. Move Horn clauses to Prolog:

```
student(A) :- dorm_resident(A).
student(A) :- takes(A,B), class(B).
```

SKOLENIZATION

$\exists X[\text{takes}(X, cs101) \wedge \text{class\_year}(X, 2)]$

introduce a Skolem constant to get rid of existential quantifier

takes(x, cs101), class_year(x, 2)

$\forall X[\neg \text{dorm\_resident}(X) \vee \exists A[\text{campus\_address\_of}(X, A)]]$

introduce a Skolem function (address depends on X)

$\forall X[\neg \text{dorm\_resident}(X) \vee \text{campus\_address\_of}(X, f(X))]$

what if we do not introduce a function, what is the new meaning?

LIMITATIONS

- If more than one non-negated term in a clause, it cannot be moved to a Horn clause (only one head term).

- If zero non-negated terms, the same problem arises.

"every living thing is an animal or plant"

e.g. $animal(x) \lor plant(x) \lor \neg living(x)$

$animal(x) \lor plant(x) \leftarrow living(x).$