CSE 494/598, Fall 2008
Logic Programming

1. Review of Propositional Logic

In logic, we distinguish between two languages: the one that is the object of our study and the one that we use to talk about that object. The former is called the *object language*; the latter is the *metalanguage*. In the first part of this course, the object language is the formal language of propositional formulas defined below. The metalanguage is the usual informal language of mathematics and theoretical computer science, which is a mixture of the English language and mathematical notation.

Propositional Formulas: Syntax

A *propositional signature* is a set of symbols called *atoms*. (In examples, we will assume that p, q, r are atoms.) The symbols

$$\wedge \quad \vee \quad \rightarrow \quad \leftrightarrow \quad \neg \quad \bot \quad \top$$

are called *propositional connectives*. Among them, the symbols $\wedge$ *(conjunction)*, $\vee$ *(disjunction)*, $\rightarrow$ *(implication)* and $\leftrightarrow$ *(equivalence)* are called *2-place*, or *binary* connectives; $\neg$ *(negation)* is a *1-place*, or *unary* connective; $\bot$ *(false)* and $\top$ *(true)* are *0-place*.

Take a propositional signature σ which contains neither the propositional connectives nor the parentheses (,). The alphabet of propositional logic consists of the atoms from σ, the propositional connectives, and the parentheses. By a *string* we understand a finite string of symbols in this alphabet. We define when a string is a *(propositional) formula* recursively, as follows:

- every atom is a formula,
- both 0-place connectives are formulas,
- if F is a formula then $\neg F$ is a formula,
- for any binary connective $\odot$, if F and G are formulas then $(F \odot G)$ is a formula.

For instance,

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

and

$$(\neg p \to q) \tag{1}$$

are formulas; the string

$$\neg p \to q \tag{2}$$

is not a formula. But very soon (see next page) we are going to introduce a convention according to which (2) can be used as an abbreviation for (1).

Properties of formulas can be often proved by induction. One useful method is strong induction on length (number of symbols). In such a proof, the induction hypothesis is that every formula which is shorter than F has the property P that we want to prove. From this assumption we need to derive that F has property P also. Then it follows that all formulas have property P.

In another useful form of induction, we check that all atoms and 0-place connectives have property P, and that the property is preserved when a new formula is formed using a unary or binary connective. More precisely, we show that

- every atom has property P,
- both 0-place connectives have property P,
- if a formula F has property P then so does $\neg F$,
- for any binary connective $\odot$, if formulas F and G have property P then so does $(F \odot G)$.

Then we can conclude that property P holds for all formulas. This is called "structural induction."

1.1 Show how to use each kind of induction to prove that the number of left parentheses in any formula is equal to the number of right parentheses.

From now on, we will abbreviate formulas of the form $(F \odot G)$ by dropping the outermost parentheses in them. We will also agree that $\leftrightarrow$ has a lower binding power than the other binary connectives. For instance,

$$p \vee q \leftrightarrow p \to r$$

will be viewed as shorthand for

$$((p \vee q) \leftrightarrow (p \to r)).$$

Finally, for any formulas $F_1, F_2, \dots, F_n$ $(n > 2)$,

$$F_1 \wedge F_2 \wedge \cdots \wedge F_n$$

will stand for

$$(\cdots(F_1 \wedge F_2) \wedge \cdots \wedge F_n).$$

The abbreviation $F_1 \vee F_2 \vee \cdots \vee F_n$ will be understood in a similar way.

Propositional Formulas: Semantics

The symbols f and t are called *truth values*. An *interpretation* of a propositional signature σ is a function from σ into $\{\mathsf{f}, \mathsf{t}\}$. If σ is finite then an interpretation can be defined by the table of its values, for instance:

p	q	r
f	f	t

(3)

The semantics of propositional formulas that we are going to introduce defines which truth value is assigned to a formula F by an interpretation I.

As a preliminary step, we need to associate functions with all unary and binary connectives: a function from $\{\mathsf{f}, \mathsf{t}\}$ into $\{\mathsf{f}, \mathsf{t}\}$ with the unary connective $\neg$, and a function from $\{\mathsf{f}, \mathsf{t}\} \times \{\mathsf{f}, \mathsf{t}\}$ into $\{\mathsf{f}, \mathsf{t}\}$ with each of the binary connectives. These functions are denoted by the same symbols as the corresponding connectives, and defined by the following tables:

x	$\neg(x)$
f	t
t	f

x	y	$\wedge(x, y)$	$\vee(x, y)$	$\rightarrow(x, y)$	$\leftrightarrow(x, y)$
f	f	f	f	t	t
f	t	f	t	t	f
t	f	f	t	f	f
t	t	t	t	t	t

For any formula F and any interpretation I, the truth value F^I that is *assigned* to F by I is defined recursively, as follows:

- for any atom F, $F^I = I(F)$,
- $\bot^I = \mathsf{f}$, $\top^I = \mathsf{t}$,
- $(\neg F)^I = \neg(F^I)$,
- $(F \odot G)^I = \odot(F^I, G^I)$ for every binary connective $\odot$.

If $F^I = \mathsf{t}$ then we say that the interpretation I *satisfies* F (symbolically, $I \models F$).

If the underlying signature is finite then the set of interpretations is finite also, and the values of F^I for all interpretations I can be represented by a finite table. This table is called the *truth table* of F.

A propositional formula F is a *tautology* if every interpretation satisfies F.

Equivalent Formulas

A formula F is *equivalent* to a formula G (symbolically, $F \Leftrightarrow G$) if, for every interpretation I, $F^I = G^I$. In other words, $F \Leftrightarrow G$ means that formula $F \leftrightarrow G$ is a tautology.

Satisfiability and Entailment

A set Γ of formulas is *satisfiable* if there exists an interpretation that satisfies all formulas in Γ, and *unsatisfiable* otherwise.

For any atom A, the literals A, $\neg A$ are said to be *complementary* to each other. Thus the assertion of the last problem can be expressed as follows: A set of literals is satisfiable iff it does not contain complementary pairs.

A set Γ of formulas *entails* a formula F (symbolically, $\Gamma \models F$), if every interpretation that satisfies all formulas in Γ satisfies F also. Note that the notation for entailment uses the same symbol as the notation for satisfaction introduced earlier, the difference being that the expression on the left is an interpretation (I) in one case and a set of formulas (Γ) in the other. The formulas entailed by Γ are also called the *logical consequences* of Γ.