Cheatsheet Mechanical Theorem Proving

July 17, 2019

by Constantin Ruhdorfer

1 Mechanical Theorem Proving

This file contains a number of relevant formulas and theorems to remember when working in the field of automated reasoning/mechanical theorem proving. This is updated while I introduce myself to the field.

2 Content

3 Propositional Logic

A proposition is a declarative sentence that can be either true ***T*** or false ***F***. The symbols (G, H, P etc.) used to denote a proposition are called *atoms*. From propositions we can build computed propositions by using logical connectives. They are:

- $\neg$ "Not"
- $\wedge$ "And"
- $\vee$ "Or"
- $\implies$ "If .. then" or "implies"
- $\iff$ "If and only if"

If you have forgotten how they map to truth values or want to refresh your memories take a look here. Also useful will be De Morgan's laws:

$$\neg (F \wedge G) = \neg F \vee \neg G$$

$$\neg (F \vee G) = \neg F \wedge \neg G$$

3.1 Well formatted formulas

Well formatted formulas are defined recursively:

1. An atom is a formula.
2. If G is a formula so is $(\neg G)$.
3. If G and H are formulas so are $(G \wedge H)$, $(G \vee H)$, $(G \implies H)$ and $(G \iff H)$.
4. All formulas are generated using above rules.

3.2 Omitting braces

Braces can be omitted by assigning decreasing ranks to the propositional connectives

$$\iff, \implies, \wedge, \vee, \neg$$

and requiring that the connective with greater rank always reaches further.

3.3 Interpretations

Given a propositional formula G and $A_1, A_2, ...A_n$ the atoms occurring in the formula then the assignment of truth values ($\boldsymbol{T}$ or $\boldsymbol{F}$) to $A_1, A_2, ...A_n$ is called an *interpretation*. A formula G is said to be *true under an interpretation* if G is evaluated to $\boldsymbol{T}$ in the interpretation.

3.4 Validity and Consistency

A formula is said to be *valid* if and only if it is true under all its interpretations. Otherwise it is said to be *invalid*. A formula is said to be *inconsistent* (or *unsatisfiable*) if and only if it is false under all its interpretations. Otherwise it is said to be *consistent*.

3.5 Equality

Two formulas G and H, denoted as $G = H$, are considered equal if the truth values of G and H are the same under every interpretation of G and H.

3.6 Normal forms

A *literal* is an atom or the negation of an atom.

3.6.1 Conjunctive normal form

A formula F is said to be in *conjunctive normal form* if F has the form $F := F_1 \wedge ... \wedge F_n, n \geq 1$ where each of $F_1, ..., F_n$ is a disjunction of literals.

3.6.2 Disjunctive normal form

A formula F is said to be in *disjunctive normal form* if F has the form $F := F_1 \vee ... \vee F_n, n \geq 1$ where each of $F_1, ..., F_n$ is a conjunction of literals.

3.7 Logical Consequence

A formula G *logically follows from* $F_1, ..., F_n$ if and only if any interpretation I in which $F_1 \wedge ... \wedge F_n$ is true G is also true. $F_1, ..., F_n$ are called *axioms* of G. Two practical ways of showing logical consequence:

Given $F_1, ..., F_n$ and a formula G, G is a logical consequence of $F_1, ..., F_n$ if the formula $(F_1, ..., F_n) \implies G$ is valid or if the formula $F_1 \wedge ... \wedge F_n \wedge \neg G$ is inconsistent.

3.8 More

For a quick refresher on all of this consult your favorite resource or for instance this.

4 First Order Logic

Sometimes also referred to as "predicate logic".

4.1 Terms, Predicates and Atoms

Terms are defined recursively:

1. A constant is a term.
2. A variable is a term.
3. If f is an n-place function symbol, and $t_1, ..., t_n$ are terms, then $f(t_1, ..., t_n)$ is a term.
4. All terms are generated by applying the above rules.

A *predicate* is a mapping from a list of terms to a truth value. If P is an n-place predicate symbol, and $t_1, ..., t_n$ are terms, then $P(t_1, ..., t_n)$ is an *atom*. Using the same five logical connectives to build formulas from atoms.

4.2 Quantifiers

First order logic deals with variables. In order to characterize those we use the symbols:

1. The *universal quantifier*: $(\forall x)$ "For all x..."
2. The *existential quantifier*: $(\exists x)$ "There exists an x..."

4.3 Variable Scope

The occurrence of a variable in a formula is called *bound* if the the occurrence is within the scope of a quantifier employing the variable or the occurrence is the quantifier. Otherwise it is considered *free*.

A variable in a formula is said to be *free* if at least one occurrence of it is free in the formula and is *bound* if at least one occurrence of it is bound. Therefore a variable can both be considered free and bound in a formula. For instance consider $(\forall x) P(x, y) \vee (\exists y) Q(y)$.

4.4 Well formatted formulas

Well formatted formulas are defined recursively:

1. An atom is a formula.
2. If G and H are formulas, then so are $(\neg G)$, $(G \vee H)$, $(G \wedge H)$, $(G \implies H)$ and $(G \iff H)$.
3. If G is a formula and x is a free variable in G, then $(\forall x)\, G$ and $(\exists x)\, G$ are formulas.
4. Formulas are generated only by a *finite number* of applications of the rules above.

4.5 Interpretations

Since first order logic uses variables assign truth values to atoms is not sufficient. The domain and an assignment to constants, predicate and function symbols needs to be specified first. Then we can define an interpretation as:

A *interpretation* of a first order logic formula F consists of a nonempty domain D, and an assignment of "values":

1. Constants are assigned an element in D
2. n-place function symbols are assigned a mapping from $D^n = \{(x_1, ..., x_n) \,|x_1 \in D, ..., x_n \in D\}$ to D.
3. n-place predicate symbols are assigned a mapping from D^n to $\{\boldsymbol{T}, \boldsymbol{F}\}$.

4.6 Evaluation

An evaluation for a formula G can be computed by using these rules:

1. $\iff$, $\implies$, $\wedge$, $\vee$, $\neg$ are evaluated as usual.
2. $(\forall x)\, G$ is $\boldsymbol{T}$ if G is $\boldsymbol{T}$ for every $d \in D$.
3. $(\exists x)\, H$ is $\boldsymbol{T}$ if H is $\boldsymbol{T}$ for at least one $d \in D$

Formulas containing free variables cannot be evaluated.

4.7 Validity and Consistency

Validity and consistency are defined similar to the definitions in the propositional logic. This means:

A formula G is consistent if there exists an interpretation I under which G is assigned the value $\boldsymbol{T}$. If this is the case we say that I is a *model* of G and *satisfies* it. Again: If there is no interpretation that satisfies G then the formula is considered inconsistent.

As before, a formula is considered valid if every interpretation of the formula satisfies the formula.

4.8 Logical consequence

Logical consequence is defined exactly the same as above, meaning that:

A formula G *logically follows from* $F_1, ..., F_n$ if and only if any interpretation I in which $F_1 \wedge ... \wedge F_n$ is true G is also true. $F_1, ..., F_n$ are called *axioms* of G. Two practical ways of showing logical consequence:

Given $F_1, ..., F_n$ and a formula G, G is a logical consequence of $F_1, ..., F_n$ if the formula $(F_1, ..., F_n) \implies G$ is valid or if the formula $F_1 \wedge ... \wedge F_n \wedge \neg G$ is inconsistent.

4.9 Prenex normal form

Prenex normal form is used to simplify prove procedures (handy if you consider the topic...).

A formula is considered to be in prenex normal form if it is in the form of:

$$(Q_1 x_1) ... (Q_n x_n) (M)$$

where in every $(Q_i x_i), i = 1, ..., n$ the Q is one of the quantifiers discussed above and M is a formula free of quantifiers. In this case $(Q_1 x_1) ... (Q_n x_n)$ is called the prefix and M the matrix of the formula.

Or less formal: All quantifiers are placed at the start of the formula.

4.10 Equality

Equality is the same as discussed before but with some added rules around quantifiers and free variables. Let $F[x]$ denote a formula F with a free variable x in it. Let G be a formula without the variable x. It follows:

$$(Qx) F[x] \vee G = (Qx) (F[x] \vee G)$$
$$(Qx) F[x] \wedge G = (Qx) (F[x] \wedge G)$$
$$\neg ((\forall x) F[x]) = (\exists x) (\neg F[x])$$
$$\neg ((\exists x) F[x]) = (\forall x) (\neg F[x])$$

And for $F[x]$ and $G[x]$:

$$(\forall x) F[x] \wedge (\forall x) G[x] = (\forall x) (F[x] \wedge G[x])$$
$$(\exists x) F[x] \vee (\exists x) G[x] = (\exists x) (F[x] \vee G[x])$$

Sadly the opposite is not true:

$$(\forall x) F[x] \vee (\forall x) G[x] \neq (\forall x) (F[x] \vee G[x])$$

$$(\exists x) F[x] \wedge (\exists x) G[x] \neq (\exists x) (F[x] \wedge G[x])$$

The proves for these are trivial and can be looked up in every textbook.

4.11 Transforming into Prenex normal form

First eliminate $\iff$ and $\implies$:

$$F \iff G = (F \implies G) \wedge (G \implies F)$$

$$F \implies G = \neg F \vee G$$

Then eliminate not needed $\neg$s:

$$\neg (\neg G) = G$$

in combination with De Morgan's (see above) laws and additionally the laws:

$$\neg\left((\forall x)\, F[x]\right) = (\exists x)\left(\neg G[x]\right)$$

$$\neg\left((\exists x)\, F[x]\right) = (\forall x)\left(\neg G[x]\right)$$

To move all quantifiers to the left use the equalities from the chapter before in addition to:

$$(Q_1 x)\, F[x] \vee (Q_2 x)\, H[x] = (Q_1 x)\,(Q_2 z)\,(F[x] \vee H[z])$$

$$(Q_3 x)\, F[x] \wedge (Q_4 x)\, H[x] = (Q_3 x)\,(Q_4 z)\,(F[x] \wedge H[z])$$