# Automated Reasoning

**Harald Ganzinger**

**Viorica Sofronie-Stokkermans**

**Uwe Waldmann**

**Summer Term 2004**



## Topics of the Course

Propositional logic
    syntax, semantics
    OBDDs, DPLL-procedure

First-order predicate logic
    syntax, semantics, model theory, . . .
    resolution, tableaux

First-order predicate logic with equality
    term rewriting systems
    Knuth-Bendix completion, superposition

Implementation techniques
    indexing data structures



## Emphasis in this Course

- introduction into logics and *deductive services* underlying important domains of application
- proof systems: soundness, completeness, complexity, implementation
- implementation of theoretical constructions
- efficient algorithms for specific deduction problems



## Literature

Schöning: Logik für Informatiker, Spektrum

Fitting: First-Order Logic and Automated Theorem Proving, Springer

Baader and Nipkow: Term Rewriting and All That, Cambridge Univ. Press

## Part 1: Propositional Logic

Propositional logic

- logic of truth values
- decidable (but NP-complete)
- can be used to describe functions over a finite domain
- important for hardware applications (e.g., model checking)



## 1.1 Syntax

- propositional variables
- logical symbols
  $\Rightarrow$ Boolean combinations



## Propositional Variables

Let $\Pi$ be a set of propositional variables.

We use letters $P$, $Q$, $R$, $S$, to denote propositional variables.



## Propositional Formulas

$F_\Pi$ is the set of propositional formulas over $\Pi$ defined as follows:

$$
\begin{array}{lcll}
F, G, H & ::= & \bot & \text{(falsum)} \\
 & \mid & \top & \text{(verum)} \\
 & \mid & P, \quad P \in \Pi & \text{(atomic formula)} \\
 & \mid & \neg F & \text{(negation)} \\
 & \mid & (F \wedge G) & \text{(conjunction)} \\
 & \mid & (F \vee G) & \text{(disjunction)} \\
 & \mid & (F \rightarrow G) & \text{(implication)} \\
 & \mid & (F \leftrightarrow G) & \text{(equivalence)}
\end{array}
$$

## Notational Conventions

- We omit brackets according to the following rules:
  - $\neg \;>_p\; \vee \;>_p\; \wedge \;>_p\; \rightarrow \;>_p\; \leftrightarrow$ (binding precedences)
  - $\vee$ and $\wedge$ are associative and commutative
  - $\rightarrow$ is right-associative



## 1.2 Semantics

In classical logic (dating back to Aristoteles) there are “only” two truth values “true” and “false” which we shall denote, respectively, by 1 and 0.

There are multi-valued logics having more than two truth values.



## Valuations

A propositional variable has no intrinsic meaning. The meaning of a propositional variable has to be defined by a valuation.

A $\Pi$-valuation is a map

$$\mathcal{A} : \Pi \rightarrow \{0, 1\}.$$

where $\{0, 1\}$ is the set of truth values.



## Truth Value of a Formula in $\mathcal{A}$

Given a $\Pi$-valuation $\mathcal{A}$, the function $\mathcal{A}^* : \Sigma\text{-formulas} \rightarrow \{0, 1\}$ is defined inductively over the structure of $F$ as follows:

$$\begin{aligned}
\mathcal{A}^*(\bot) &= 0 \\
\mathcal{A}^*(\top) &= 1 \\
\mathcal{A}^*(P) &= \mathcal{A}(P) \\
\mathcal{A}^*(\neg F) &= 1 - \mathcal{A}^*(F) \\
\mathcal{A}^*(F \rho G) &= \mathrm{B}_\rho(\mathcal{A}^*(F), \mathcal{A}^*(G)) \\
&\quad \text{with } \mathrm{B}_\rho \text{ the Boolean function associated with } \rho
\end{aligned}$$

For simplicity, we write $\mathcal{A}$ instead of $\mathcal{A}^*$.

## 1.3 Models, Validity, and Satisfiability

$F$ is valid in $\mathcal{A}$ ($\mathcal{A}$ is a model of $F$; $F$ holds under $\mathcal{A}$):

$$\mathcal{A} \models F :\Leftrightarrow \mathcal{A}(F) = 1$$

$F$ is valid (or is a tautology):

$$\models F :\Leftrightarrow \mathcal{A} \models F \text{ for all } \Pi\text{-valuations } \mathcal{A}$$

$F$ is called satisfiable iff there exists an $\mathcal{A}$ such that $\mathcal{A} \models F$.
Otherwise $F$ is called unsatisfiable (or contradictory).



## Entailment and Equivalence

$F$ entails (implies) $G$ (or $G$ is a consequence of $F$), written $F \models G$, if for all $\Pi$-valuations $\mathcal{A}$, whenever $\mathcal{A} \models F$ then $\mathcal{A} \models G$.

$F$ and $G$ are called equivalent if for all $\Pi$-valuations $\mathcal{A}$ we have $\mathcal{A} \models F \Leftrightarrow \mathcal{A} \models G$.

Proposition 1.1:
$F$ entails $G$ iff $(F \rightarrow G)$ is valid

Proposition 1.2:
$F$ and $G$ are equivalent iff $(F \leftrightarrow G)$ is valid.



## Entailment and Equivalence

Extension to sets of formulas $N$ in the "natural way", e.g., $N \models F$ if for all $\Pi$-valuations $\mathcal{A}$: if $\mathcal{A} \models G$ for all $G \in N$, then $\mathcal{A} \models F$.



## Validity vs. Unsatisfiability

Validity and unsatisfiability are just two sides of the same medal as explained by the following proposition.

Proposition 1.3:

$$F \text{ valid} \Leftrightarrow \neg F \text{ unsatisfiable}$$

Hence in order to design a theorem prover (validity checker) it is sufficient to design a checker for unsatisfiability.

$Q$: In a similar way, entailment $N \models F$ can be reduced to unsatisfiability. How?

## Checking Unsatisfiability

Every formula $F$ contains only finitely many propositional variables. Obviously, $\mathcal{A}(F)$ depends only on the values of those finitely many variables in $F$ under $\mathcal{A}$.

If $F$ contains $n$ distinct propositional variables, then it is sufficient to check $2^n$ valuations to see whether $F$ is satisfiable or not.
$\Rightarrow$ truth table.

So the satisfiability problem is clearly deciadable (but, by Cook's Theorem, NP-complete).

Nevertheless, in practice, there are (much) better methods than truth tables to check the satisfiability of a formula. (later more)



## Substitution Theorem

Proposition 1.4:
Let $F$ and $G$ be equivalent formulas, let $H$ be a formula in which $F$ occurs as a subformula.

Then $H$ is equivalent to $H'$ where $H'$ is obtained from $H$ by replacing the occurrence of the subformula $F$ by $G$.
(Notation: $H = H[F]$, $H' = H[G]$.)

Proof: By induction over the formula structure of $H$.



## Some Important Equivalences

Proposition 1.5:
The following equivalences are valid for all formulas $F, G, H$:

$$
\begin{aligned}
(F \wedge F) &\leftrightarrow F \\
(F \vee F) &\leftrightarrow F && \text{(Idempotency)} \\
(F \wedge G) &\leftrightarrow (G \wedge F) \\
(F \vee G) &\leftrightarrow (G \vee F) && \text{(Commutativity)} \\
(F \wedge (G \wedge H)) &\leftrightarrow ((F \wedge G) \wedge H) \\
(F \vee (G \vee H)) &\leftrightarrow ((F \vee G) \vee H) && \text{(Associativity)} \\
(F \wedge (G \vee H)) &\leftrightarrow ((F \wedge G) \vee (F \wedge H)) \\
(F \vee (G \wedge H)) &\leftrightarrow ((F \vee G) \wedge (F \vee H)) && \text{(Distributivity)}
\end{aligned}
$$



## Some Important Equivalences

The following equivalences are valid for all formulas $F, G, H$:

$$
\begin{aligned}
(F \wedge (F \vee G)) &\leftrightarrow F \\
(F \vee (F \wedge G)) &\leftrightarrow F && \text{(Absorption)} \\
(\neg\neg F) &\leftrightarrow F && \text{(Double Negation)} \\
\neg(F \wedge G) &\leftrightarrow (\neg F \vee \neg G) \\
\neg(F \vee G) &\leftrightarrow (\neg F \wedge \neg G) && \text{(De Morgan's Laws)} \\
(F \wedge G) &\leftrightarrow F, \text{ if } G \text{ is a tautology} \\
(F \vee G) &\leftrightarrow \top, \text{ if } G \text{ is a tautology} && \text{(Tautology Laws)} \\
(F \wedge G) &\leftrightarrow \bot, \text{ if } G \text{ is unsatisfiable} \\
(F \vee G) &\leftrightarrow F, \text{ if } G \text{ is unsatisfiable} && \text{(Tautology Laws)}
\end{aligned}
$$