H250: Honors Colloquium – Introduction to Computation

Axiomatization of Logic. Truth and Proof. Resolution

Marius Minea
marius@cs.umass.edu

Motivation: Determining Truth

In CS250, we started with truth table proofs.

Propositional formula: can always do truth table (even if large)

Predicate formula: can't do truth tables (infinite possibilities)
$\Rightarrow$ must use other *proof rules*

Motivation: Determining Truth

In CS250, we started with truth table proofs.

Propositional formula: can always do truth table (even if large)

Predicate formula: can't do truth tables (infinite possibilities)
$\Rightarrow$ must use other *proof rules*

How many do we need? (best: few)
Are they enough?
Can we prove everything?

Axiomatization helps answer these questions

First, Define Syntax

We define a *language* by its *symbols*
and the *rules* to correctly combine symbols (the *syntax*)

First, Define Syntax

We define a *language* by its *symbols*
and the *rules* to correctly combine symbols (the *syntax*)

Symbols of propositional logic:

propositions: p, q, r (usually lowercase letters)

operators (logical connectives): negation $\neg$, implication $\rightarrow$

parentheses ( )

First, Define Syntax

We define a *language* by its *symbols*
and the *rules* to correctly combine symbols (the *syntax*)

Symbols of propositional logic:
propositions: p, q, r (usually lowercase letters)
operators (logical connectives): negation $\neg$, implication $\rightarrow$
parentheses ( )

Formulas of propositional logic: defined by *structural induction*
(how to build complex formulas from simpler ones)

A formula (compound proposition) is:
any *proposition* (aka atomic formula or variable)
$(\neg\alpha)$ where α is a formula
$(\alpha \rightarrow \beta)$ if α and β are formulas

First, Define Syntax

We define a *language* by its *symbols*
and the *rules* to correctly combine symbols (the *syntax*)

Symbols of propositional logic:
propositions: p, q, r (usually lowercase letters)
operators (logical connectives): negation $\neg$, implication $\rightarrow$
parentheses ( )

Formulas of propositional logic: defined by *structural induction*
(how to build complex formulas from simpler ones)

A formula (compound proposition) is:
any *proposition* (aka atomic formula or variable)
$(\neg\alpha)$ where α is a formula
$(\alpha \rightarrow \beta)$ if α and β are formulas

Implication and negation suffice!

Definitions Should Be Minimal

Fewest cases $\Rightarrow$ simplicity
(all future reasoning must cover all cases)

Can define all other connectives in terms of $\neg$ and $\rightarrow$:

$\alpha \wedge \beta \stackrel{def}{=} \neg(\alpha \rightarrow \neg\beta)$ (AND)

$\alpha \vee \beta \stackrel{def}{=} \neg\alpha \rightarrow \beta$ (OR)

$\alpha \leftrightarrow \beta \stackrel{def}{=} (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$ (equivalence)

What is a Proof? (First Try)

Let H be a set of formulas (hypotheses, premises).
A *deduction* (or *proof*) from H is a sequence of formulas (statements) $S_1, \ldots, S_n$, such that every formula S_k is

- a premise : $S_k \in H$
- or follows from previous statements by a *inference rule*

What is a Proof? (First Try)

Let H be a set of formulas (hypotheses, premises).
A *deduction* (or *proof*) from H is a sequence of formulas (statements) $S_1, \ldots, S_n$, such that every formula S_k is

- a premise : $S_k \in H$
- or follows from previous statements by a *inference rule*

Which inference rules to use? (again, few is good!)

$$\frac{A \qquad A \to B}{B} \qquad \textit{modus ponens}$$

(from A and $A \to B$ we derive/infer B

What is a Proof? (First Try)

Let H be a set of formulas (hypotheses, premises).
A *deduction* (or *proof*) from H is a sequence of formulas (statements) $S_1, \ldots, S_n$, such that every formula S_k is

- a premise : $S_k \in H$
- or follows from previous statements by a *inference rule*

Which inference rules to use? (again, few is good!)

$$\frac{A \qquad A \to B}{B} \qquad \textit{modus ponens}$$

(from A and $A \to B$ we derive/infer B

Modus Ponens is enough for propositional (and predicate) logic

Proof Axioms

What about statements that need no premises at all ?
Need a *base case* for our reasoning: *axioms*
Actual definition: A *deduction* (or *proof*) from H is a sequence of formulas $S_1, \ldots, S_n$, such that every formula S_k is

- an *axiom*
- a premise : $S_k \in H$
- or follows from previous statements by a *deduction rule*

Notation: $H \vdash S_n$ (S_n can be derived from H)

Axioms of propositional logic:
A1: $\alpha \rightarrow (\beta \rightarrow \alpha)$
A2: $(\alpha \rightarrow (\beta \rightarrow \gamma)) \rightarrow ((\alpha \rightarrow \beta) \rightarrow (\alpha \rightarrow \gamma))$
A3: $(\neg\beta \rightarrow \neg\alpha) \rightarrow (\alpha \rightarrow \beta)$

α, β, γ are any formulas

A Sample Deduction

We prove $A \to A$ for any formula A
(this is an axiom in some systems, but we can do without it)
(1) $A \to ((A \to A) \to A))$ A1, $\alpha = A, \beta = A \to A$

A Sample Deduction

We prove $A \to A$ for any formula A

(this is an axiom in some systems, but we can do without it)

(1) $A \to ((A \to A) \to A))$ A1, $\alpha = A, \beta = A \to A$

((2) $A \to ((A \to A) \to A)) \to ((A \to (A \to A)) \to (A \to A))$

A2, $\alpha = \gamma = A, \beta = A \to A$

A Sample Deduction

We prove $A \to A$ for any formula A

(this is an axiom in some systems, but we can do without it)

(1) $A \to ((A \to A) \to A))$ A1, $\alpha = A, \beta = A \to A$

((2) $A \to ((A \to A) \to A)) \to ((A \to (A \to A)) \to (A \to A))$

A2, $\alpha = \gamma = A, \beta = A \to A$

(3) $(A \to (A \to A)) \to (A \to A)$ MP(1,2)

A Sample Deduction

We prove $A \to A$ for any formula A

(this is an axiom in some systems, but we can do without it)

(1) $A \to ((A \to A) \to A))$ A1, $\alpha = A, \beta = A \to A$

((2) $A \to ((A \to A) \to A)) \to ((A \to (A \to A)) \to (A \to A))$

A2, $\alpha = \gamma = A, \beta = A \to A$

(3) $(A \to (A \to A)) \to (A \to A)$ MP(1,2)

(4) $A \to (A \to A)$ A1, $\alpha = \beta = A$

A Sample Deduction

We prove $A \to A$ for any formula A
(this is an axiom in some systems, but we can do without it)

(1) $A \to ((A \to A) \to A))$ A1, $\alpha = A, \beta = A \to A$

((2) $A \to ((A \to A) \to A)) \to ((A \to (A \to A)) \to (A \to A))$
A2, $\alpha = \gamma = A, \beta = A \to A$

(3) $(A \to (A \to A)) \to (A \to A)$ MP(1,2)

(4) $A \to (A \to A)$ A1, $\alpha = \beta = A$

(5) $A \to A$ MP(3,4)

A Sample Deduction

We prove $A \to A$ for any formula A
(this is an axiom in some systems, but we can do without it)

(1) $A \to ((A \to A) \to A))$ A1, $\alpha = A, \beta = A \to A$

((2) $A \to ((A \to A) \to A)) \to ((A \to (A \to A)) \to (A \to A))$ A2, $\alpha = \gamma = A, \beta = A \to A$

(3) $(A \to (A \to A)) \to (A \to A)$ MP(1,2)

(4) $A \to (A \to A)$ A1, $\alpha = \beta = A$

(5) $A \to A$ MP(3,4)

We can also show the *Deduction Theorem*

$$\text{If } H \cup \{A\} \vdash B \text{ then } H \vdash A \to B$$

(assuming premise and proving conclusion shows implication)

Proofs are purely syntactic!

We have done this without defining truth values, truth tables, etc.

Finding a proof may be difficult (creativity, heuristics/tactics, etc.)

Checking a proof is mechanical, based on simple string operations
(check that formulas "pattern match" structure of axioms/rules)

Semantics of Logic

Truth assignment (valuation): gives a value (T, F) to any formula:
$v(p)$ defined for any atomic proposition p

Semantics of Logic

Truth assignment (valuation): gives a value (T, F) to any formula:

$v(p)$ defined for any atomic proposition p

$$v(\neg\alpha) = \begin{cases} \mathsf{T} & \text{if } v(\alpha) = \mathsf{F} \\ \mathsf{F} & \text{if } v(\alpha) = \mathsf{T} \end{cases}$$

Semantics of Logic

Truth assignment (valuation): gives a value (T, F) to any formula:

$v(p)$ defined for any atomic proposition p

$$v(\neg\alpha) = \begin{cases} \text{T} & \text{if } v(\alpha) = \text{F} \\ \text{F} & \text{if } v(\alpha) = \text{T} \end{cases}$$

$$v(\alpha \to \beta) = \begin{cases} \text{F} & \text{if } v(\alpha) = \text{T and } v(\beta) = \text{F} \\ \text{T} & \text{otherwise} \end{cases}$$

Semantics of Logic

Truth assignment (valuation): gives a value (T, F) to any formula:

$v(p)$ defined for any atomic proposition p

$$v(\neg\alpha) = \begin{cases} \text{T} & \text{if } v(\alpha) = \text{F} \\ \text{F} & \text{if } v(\alpha) = \text{T} \end{cases}$$

$$v(\alpha \to \beta) = \begin{cases} \text{F} & \text{if } v(\alpha) = \text{T and } v(\beta) = \text{F} \\ \text{T} & \text{otherwise} \end{cases}$$

An *interpretation* is a truth assignment for the propositions of a formula.

Truth and Semantic Consequence

An *interpretation* is a truth assignment for the propositions of a formula.

A formula may be true or false in an interpretation.
An interpretation *satisfies* the formula or not.
A formula true in all interpretations is *valid* (a *tautology*).

How do we relate the truth values of different formulas?

Truth and Semantic Consequence

An *interpretation* is a truth assignment for the propositions of a formula.

A formula may be true or false in an interpretation.
An interpretation *satisfies* the formula or not.
A formula true in all interpretations is *valid* (a *tautology*).

How do we relate the truth values of different formulas?

A set of formulas $H = \{H_1, \ldots, H_n\}$ *entails* (semantically implies) a formula C if any interpretation that satisfies H satisfies C.
We say C is a semantic consequence (entailed by) H.

Notation: $H \models C$

This corresponds to our truth table proofs.

Soundness and Completess

$H \vdash C$: *deduction* (purely syntactic, using inference rules)
$H \models C$: *entailment* (semantic, using truth values)
Ideally, we'd like these notions to match.

Soundness and Completess

$H \vdash C$: *deduction* (purely syntactic, using inference rules)
$H \models C$: *entailment* (semantic, using truth values)
Ideally, we'd like these notions to match.

Soundness: If $H \vdash C$, then $H \models C$: every theorem is valid. (Everything that we prove is true. Our logic is sound, not contradictory).

Soundness and Completess

$H \vdash C$: *deduction* (purely syntactic, using inference rules)
$H \models C$: *entailment* (semantic, using truth values)
Ideally, we'd like these notions to match.

Soundness: If $H \vdash C$, then $H \models C$: every theorem is valid.
(Everything that we prove is true. Our logic is sound, not contradictory).

Completeness: If $H \models C$, then $H \vdash C$: everything true is provable

Propositional logic is both *sound* and *complete*.

Predicate (First Order) Logic: Syntax

Terms:

variable v

$f(t_1, \cdots, t_n)$ $\quad$ f is an n-ary *function*, $t_1, \cdots, t_n$ are *terms*

Example: $parent(x)$, $\quad gcd(x, y)$, $\quad \max(\min(x, y), z)$

constant c: special case, zero-argument function

Predicate (First Order) Logic: Syntax

Terms:

variable v

$f(t_1, \cdots, t_n)$ f is an n-ary *function*, $t_1, \cdots, t_n$ are *terms*

Example: $parent(x)$, $gcd(x, y)$, $\max(\min(x, y), z)$

constant c: special case, zero-argument function

Formulas (well-formed formulas):

$P(t_1, \cdots, t_n)$ P is an n-ary *predicate*, $t_1, \cdots, t_n$ *terms*

Example: $contains(empty, x)$, $divide(gcd(x, y), x))$

proposition p: special case, zero-argument predicate

$\neg\alpha$ α is a formula

$\alpha \to \beta$ α, β formulas

Predicate (First Order) Logic: Syntax

Terms:

variable v

$f(t_1, \cdots, t_n)$ $\quad$ f is an n-ary *function*, $t_1, \cdots, t_n$ are *terms*

Example: $parent(x)$, $\quad gcd(x, y)$, $\quad \max(\min(x, y), z)$

constant c: special case, zero-argument function

Formulas (well-formed formulas):

$P(t_1, \cdots, t_n)$ $\quad$ P is an n-ary *predicate*, $t_1, \cdots, t_n$ *terms*

Example: $contains(empty, x)$, $\quad divide(gcd(x, y), x))$

proposition p: special case, zero-argument predicate

$\neg\alpha$ $\quad$ α is a formula

$\alpha \rightarrow \beta$ $\quad$ α, β formulas

$\forall v\, \alpha$ $\quad$ v *variable*, α formula: *universal quantification*

Example: $\forall x\ \neg contains(empty, x)$, $\ \forall x \forall y\ divide(gcd(x, y), x)$

Proofs in Predicate Logic

Definition of deduction or proof is the same.

Need new axioms related to predicates and quantifiers

A1: $\alpha \to (\beta \to \alpha)$ (A1-A3 from propositional logic)

A2: $(\alpha \to (\beta \to \gamma)) \to ((\alpha \to \beta) \to (\alpha \to \gamma))$

A3: $(\neg\beta \to \neg\alpha) \to (\alpha \to \beta)$

Proofs in Predicate Logic

Definition of deduction or proof is the same.

Need new axioms related to predicates and quantifiers

A1: $\alpha \to (\beta \to \alpha)$ (A1-A3 from propositional logic)

A2: $(\alpha \to (\beta \to \gamma)) \to ((\alpha \to \beta) \to (\alpha \to \gamma))$

A3: $(\neg\beta \to \neg\alpha) \to (\alpha \to \beta)$

A4: $\forall x(\alpha \to \beta) \to (\forall x\alpha \to \forall x\beta)$

A5: $\forall x\alpha \to \alpha[x \leftarrow t]$ if x can be substituted with t in α

A6: $\alpha \to \forall x\alpha$ if x is not free in α

Structures

A *structure* (interpretation) I consists of:

a nonempty set U called *universe* or *domain* of I

Structures

A *structure* (interpretation) I consists of:

a nonempty set U called *universe* or *domain* of I

(the set of values that the variables can take)

Structures

A *structure* (interpretation) I consists of:

a nonempty set U called *universe* or *domain* of I

(the set of values that the variables can take)

a value $c_I \in U$ for every constant symbol c,

Structures

A *structure* (interpretation) I consists of:

- a nonempty set U called *universe* or *domain* of I
  (the set of values that the variables can take)
- a value $c_I \in U$ for every constant symbol c,
- a function $f_I : U^n \to U$ for every n-ary function symbol f,

Structures

A *structure* (interpretation) I consists of:

- a nonempty set U called *universe* or *domain* of I
  (the set of values that the variables can take)
- a value $c_I \in U$ for every constant symbol c,
- a function $f_I : U^n \to U$ for every n-ary function symbol f,
- a relation $P_I \subseteq U^n$ for every n-ary predicate symbol P

Structures

A *structure* (interpretation) I consists of:

a nonempty set U called *universe* or *domain* of I
(the set of values that the variables can take)

a value $c_I \in U$ for every constant symbol c,

a function $f_I : U^n \to U$ for every n-ary function symbol f,

a relation $P_I \subseteq U^n$ for every n-ary predicate symbol P

An interpretation does *not* assign values to variables.

Structures

A *structure* (interpretation) I consists of:

a nonempty set U called *universe* or *domain* of I
(the set of values that the variables can take)

a value $c_I \in U$ for every constant symbol c,

a function $f_I : U^n \to U$ for every n-ary function symbol f,

a relation $P_I \subseteq U^n$ for every n-ary predicate symbol P

An interpretation does *not* assign values to variables.

An *assignment* is a function $s : V \to U$ that assigns to every variable a value from the universe.

Truth, Models and Tautologies

Given a structure and an assignment, we can *evaluate* any formula.

we know constants, functions, variable values, predicate relations

$\forall x \varphi$ is true iff φ true when substituting x with any value $d \in U$.

Truth, Models and Tautologies

Given a structure and an assignment, we can *evaluate* any formula.
we know constants, functions, variable values, predicate relations
$\forall x \varphi$ is true iff φ true when substituting x with any value $d \in U$.

A *model* for a formula φ is a structure for which the formula is true *for any* variable assignment.
Notation: $I \models \varphi$

Truth, Models and Tautologies

Given a structure and an assignment, we can *evaluate* any formula.
we know constants, functions, variable values, predicate relations
$\forall x \varphi$ is true iff φ true when substituting x with any value $d \in U$.

A *model* for a formula φ is a structure for which the formula is true *for any* variable assignment.
Notation: $I \models \varphi$

A *tautology* is a formula that is true in *any* interpretation.

Truth, Models and Tautologies

Given a structure and an assignment, we can *evaluate* any formula.
we know constants, functions, variable values, predicate relations
$\forall x \varphi$ is true iff φ true when substituting x with any value $d \in U$.

A *model* for a formula φ is a structure for which the formula is true *for any* variable assignment.
Notation: $I \models \varphi$

A *tautology* is a formula that is true in *any* interpretation.

The number of interpretations (and assignment) is infinite!
$\Rightarrow$ can't check truth exhaustively $\Rightarrow$ *deductive proofs* essential

Soundness and Completess

Let H be a set of premises, and I an interpretation.

We say $I \models H$ if I is a model for *every* formula in H.

We say $H \models C$ (read: H (semantically) implies C) if for every interpretation I

$$I \models H \text{ implies } I \models C$$

(C is true in all interpretations that satisfy all premises in H).

Soundness and Completess

Let H be a set of premises, and I an interpretation.

We say $I \models H$ if I is a model for *every* formula in H.

We say $H \models C$ (read: H (semantically) implies C) if for every interpretation I

$$I \models H \text{ implies } I \models C$$

(C is true in all interpretations that satisfy all premises in H).

Soundness: If $H \vdash C$, then $H \models C$: every theorem is valid.

Completeness: If $H \models C$, then $H \vdash C$: everything true is provable

Predicate logic is both *sound* and *complete*.

Soundness and Completess

Let H be a set of premises, and I an interpretation.

We say $I \models H$ if I is a model for *every* formula in H.

We say $H \models C$ (read: H (semantically) implies C) if for every interpretation I

$$I \models H \text{ implies } I \models C$$

(C is true in all interpretations that satisfy all premises in H).

Soundness: If $H \vdash C$, then $H \models C$: every theorem is valid.

Completeness: If $H \models C$, then $H \vdash C$: everything true is provable

Predicate logic is both *sound* and *complete*.

Important: Completeness says we can prove something that's true.
We may not be able to disprove something false
(thus decide if something unknown is true or false).

Proof by Resolution

A formula is *valid* iff its *negation* is a *contradiction*.

We can prove a theorem by contradiction showing that its negation is *unsatisfiable*

Proof by Resolution

A formula is *valid* iff its *negation* is a *contradiction*.

We can prove a theorem by contradiction showing that its negation is *unsatisfiable*

Consider hypotheses $A_1, A_2, \ldots, A_n$, conclusion C and the theorem

$$A_1 \wedge A_2 \ldots \wedge A_n \rightarrow C$$

i.e., together, premises $A_1, A_2, \ldots A_n$ imply conclusion C

Proof by Resolution

A formula is *valid* iff its *negation* is a *contradiction*.

We can prove a theorem by contradiction showing that its negation is *unsatisfiable*

Consider hypotheses $A_1, A_2, \ldots, A_n$, conclusion C and the theorem

$$A_1 \wedge A_2 \ldots \wedge A_n \rightarrow C$$

i.e., together, premises $A_1, A_2, \ldots A_n$ imply conclusion C

Negation of implication: $\neg(H \rightarrow C) = \neg(\neg H \vee C) = H \wedge \neg C$

So we show $\quad A_1 \wedge A_2 \ldots \wedge A_n \wedge \neg C \quad$ is a contradiction
We can systematically do this by the *resolution method*.

Resolution in propositional logic

Resolution is an *inference rule* that produces a *new clause* from two clauses with *complementary literals* (p and $\neg p$).

$$\boxed{\dfrac{p \vee A \qquad \neg p \vee B}{A \vee B} \qquad \textit{resolution}}$$

"From clauses $p \vee A$ and $\neg p \vee B$ we derive clause $A \vee B$"

Recall: *clause* = *disjunction* $\vee$ of *literals* (propositions or negations)

Resolution in propositional logic

Resolution is an *inference rule* that produces a *new clause* from two clauses with *complementary literals* (p and $\neg p$).

$$\boxed{\frac{p \vee A \qquad \neg p \vee B}{A \vee B} \qquad \textit{resolution}}$$

"From clauses $p \vee A$ and $\neg p \vee B$ we derive clause $A \vee B$"

Recall: *clause* = *disjunction* $\vee$ of *literals* (propositions or negations)

New clause = *resolvent* of the two clauses with respect to p

Example: $res_p(p \vee q \vee \neg r, \neg p \vee s) = q \vee \neg r \vee s$

Resolution in propositional logic

Resolution is an *inference rule* that produces a *new clause* from two clauses with *complementary literals* (p and $\neg p$).

$$\boxed{\dfrac{p \vee A \qquad \neg p \vee B}{A \vee B} \qquad \textit{resolution}}$$

"From clauses $p \vee A$ and $\neg p \vee B$ we derive clause $A \vee B$"

Recall: *clause* = *disjunction* $\vee$ of *literals* (propositions or negations)

New clause = *resolvent* of the two clauses with respect to p

Example: $res_p(p \vee q \vee \neg r, \neg p \vee s) = q \vee \neg r \vee s$

Modus ponens can be seen as a special case of resolution:

$$\frac{p \vee \mathit{false} \qquad \neg p \vee q}{\mathit{false} \vee q}$$

Likewise, hypothetical syllogism (rewrite implication using $\vee$)

Resolution is a valid proof rule

$$\frac{p \vee A \qquad \neg p \vee B}{A \vee B} \qquad \textit{resolution}$$

$$\{p \vee A, \neg p \vee B\} \models A \vee B$$

A *valid inference rule*:
any assignment making premises true also makes conclusion true

Resolution is a valid proof rule

$$\frac{p \vee A \qquad \neg p \vee B}{A \vee B} \qquad \textit{resolution}$$

$$\{p \vee A, \neg p \vee B\} \models A \vee B$$

A *valid inference rule*:
any assignment making premises true also makes conclusion true

Proof by cases: for $p = T$, we must show $B \models A \vee B$:
if $B = T$, then also $A \vee B = T$ (valid)
case $p = F$ is symmetric, so the rule is valid

Resolution is a valid proof rule

$$\frac{p \lor A \qquad \neg p \lor B}{A \lor B} \qquad \textit{resolution}$$

$$\{p \lor A, \neg p \lor B\} \models A \lor B$$

A *valid inference rule*:
any assignment making premises true also makes conclusion true

Proof by cases: for $p = T$, we must show $B \models A \lor B$:
if $B = T$, then also $A \lor B = T$ (valid)
case $p = F$ is symmetric, so the rule is valid

Corollary: if $A \lor B$ is a contradiction, so is $(p \lor A) \land (\neg p \lor B)$
if resolution reaches contradiction, we started from a contradiction