A logician is expecting a child. A friend asked: “Is it a boy or a girl?”

The logician replied: “Yes”.

Propositional Logic: Semantics (1/3)

CS402, Spring 2017

Shin Yoo

Overview

- Boolean Operators
- Propositional Formulas
- Interpretations

Propositions

A proposition is a declarative sentence. That is, it *can* be declared to be true or false. Examples:

- The sum of the numbers 3 and 5 is equal to 8.
- Jane reacted violently to Jack's accusations.
- Every even natural number greater than 2 is the sum of two prime numbers.
- All Martians like pepperoni on their pizza.

Propositionals are *atomic* and *indecomposable*. We use distinct symbols, $p.q.r.\ldots.$, to represent propositions.

Boolean Operators

Since propositions are of Boolean type, there are 2^{2^n} n-ary Boolean operators. Each of the n operands can be either true or false, resulting in 2^n Boolean tuples of operands. For each of 2^n tuples, the result of the operation can again be true or false. Hence 2^{2^n}.

For example, the following is the all possible unary Boolean opeartors, $o_1, \ldots, o_4$.

x	o_1	o_2	o_3	o_4
T	T	T	F	F
F	T	F	T	F

Operators o_1 and o_4 are constant, and do not operate on the operand; o_2 is the identity operator. Only o_3 is nontrivially interesting, and is called *negation*.

Binary Boolean Operators

There are 16 binary Boolean operators.

x_1	x_2	o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8
T	T	T	T	T	T	T	T	T	T
T	F	T	T	T	T	F	F	F	F
F	T	T	T	F	F	T	T	F	F
F	F	T	F	T	F	T	F	T	F

x_1	x_2	o_9	o_{10}	o_{11}	o_{12}	o_{13}	o_{14}	o_{15}	o_{16}
T	T	F	F	F	F	F	F	F	F
T	F	T	T	T	T	F	F	F	F
F	T	T	T	F	F	T	T	F	F
F	F	T	F	T	F	T	F	T	F

Trivial operators: o_1 and o_{16} (constant), o_4 and o_6 (projection), o_{11} and o_{13} (negated projection).

Interesting Operators

op	name	symbol	op	name	symbol
o_2	disjunction	$\vee$	o_{15}	nor	$\downarrow$
o_8	conjunction	$\wedge$	o_9	nand	$\uparrow$
o_5	implication	$\rightarrow$	o_{12}		
o_3	reverse implication	$\leftarrow$	o_{14}		
o_7	equivalence	$\leftrightarrow$	o_{10}	exclusive or	$\oplus$

x	y	$\wedge$	$\vee$	$\rightarrow$	$\leftrightarrow$	$\oplus$	$\uparrow$	$\downarrow$
T	T	T	T	T	T	F	F	F
T	F	F	T	F	F	T	T	F
F	T	F	T	T	F	T	T	F
F	F	F	F	T	T	F	T	T

Materialistic Implication

While $p \rightarrow q$ is often read "if p then q", it does not mean *causation*, i.e. it does not mean that p caused q. It only means "if p then q" such that $p \rightarrow q$ is false only when p is true but q is false (recall the truth table).

But this also means that $p \rightarrow q$ is equivalent to $\neg p \vee q$.
"2 is an odd number $\rightarrow$ 2 is an even number" is true.

The more philosophical branch of logic still has a problem with this. Outside mathematics, it is still easy to accept that when (p, q) is (T, F), $p \rightarrow q$ is also false. For cases $(T, T), (F, T)$ and (F, F), different accounts of the relationship accept that $p \rightarrow q$ is *sometimes* true, but they deny that the conditional is always true in each of these cases.

Redundancy

The first five binary operators ($\vee, \wedge, \rightarrow, \leftarrow, \leftrightarrow$) can all be defined in terms of any one of them plus negation ($\neg$). For example:

x	y	$x \wedge y$	$\neg y$	$x \rightarrow \neg y$	$\neg(x \rightarrow \neg y)$
T	T	T	F	F	T
T	F	F	T	T	F
F	T	F	F	T	F
F	F	F	T	T	F

x	y	$x \vee y$	$\neg x$	$\neg x \rightarrow y$
T	T	T	F	T
T	F	T	F	T
F	T	T	T	T
F	F	F	T	F

Redundancy

The choice of an interesting set of operators depends on the application.

- In digital circuit design, NAND($\uparrow$), NOR($\downarrow$), and NOT($\neg$) are commonly used to represent all Boolean formulas, mainly because these are more straightforward to implement at the physical, transistor level.
- In mathematics, we are generally interested in one-way logical deductions (from axioms to their implications), so we choose implication and negation.

Definition 1 (2.13)

Propositional Formula: a formula $fml \in \mathcal{F}$ is a word that can be derived from the following grammar, starting from the initial non-terminal *fml*:

1. $fml ::= p$ for any $p \in P$
2. $fml ::= \neg fml$
3. $fml ::= fml\ op\ fml$ where $op \in \{\vee, \wedge, \leftarrow, \rightarrow, \leftrightarrow, \downarrow, \uparrow, \oplus\}$

Each derivation of a formula from a grammar can be represented by a derivation tree that displays the application of the grammar rules to the non-terminals.

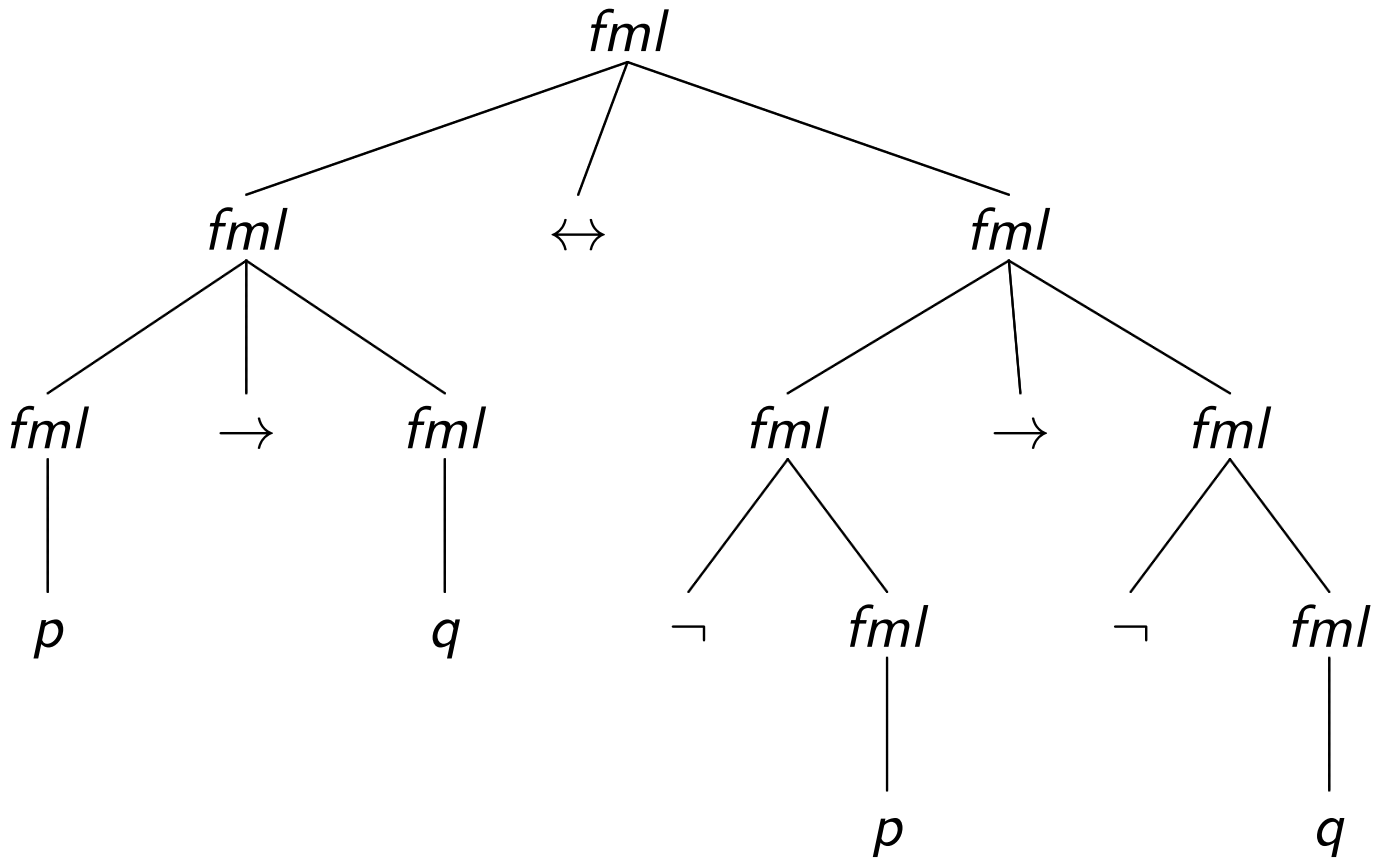
- Non-terminals: symbols that occur on the left-hand side of a rule
- Terminals: symbols that occur on only the right-hand side of a rule

From the derivation tree we can obtain a formation tree by replacing an fml non-terminal by the child that is an operator or an atom.

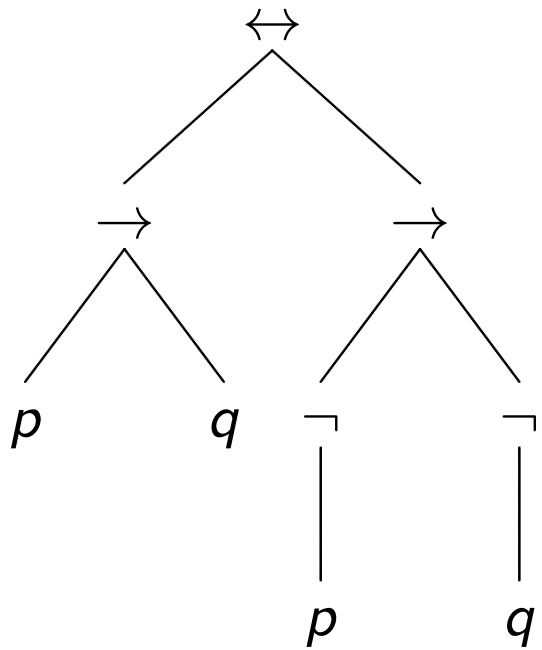
Derivation of $p \rightarrow q \leftrightarrow \neg p \rightarrow \neg q$ using grammar rules.

1. fml
2. $fml \leftrightarrow fml$
3. $fml \leftarrow fml \leftrightarrow fml$
4. $p \rightarrow fml \leftrightarrow fml$
5. $p \rightarrow q \leftrightarrow fml$
6. $p \rightarrow q \leftrightarrow fml \rightarrow fml$
7. $p \rightarrow q \leftrightarrow \neg fml \rightarrow fml$
8. $p \rightarrow q \leftrightarrow \neg p \rightarrow fml$
9. $p \rightarrow q \leftrightarrow \neg p \rightarrow \neg fml$
10. $p \rightarrow q \leftrightarrow \neg p \rightarrow \neg q$

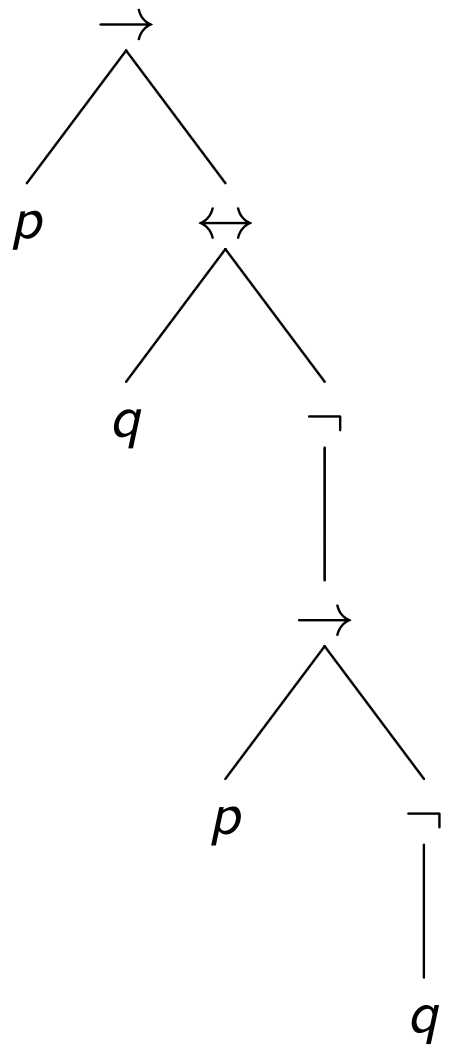
Derivation Tree: represents how non-terminals are expanded using which rules.



Formation Tree: shows the structure of the formula $p \rightarrow q \leftrightarrow \neg p \rightarrow \neg q$.



OK.



???

Removing Ambiguity: formation trees are unique, linear representation such as $p \rightarrow q \leftrightarrow \neg p \rightarrow \neg q$ are not. There are a few ways to resolve this ambiguity.

- **Polish Notation:** essentially, formulate linear representation by visiting the formation tree depth-first preorder (i.e. starting from the root, visit the current node, visit the left subtree, visit the right subtree, recursively).
  - $\leftrightarrow\rightarrow pq \rightarrow \neg p\neg q$
  - $\rightarrow p \leftrightarrow q\neg \rightarrow \neg p\neg q$
- **Use parentheses:** change the grammar slightly so that $fml ::= p$ for any $p \in P$, $fml ::= (\neg fml)$, and $fml ::= (fml\ op\ fml) \ldots$, etc.
  - $((p \rightarrow q) \leftrightarrow ((\neg p) \rightarrow (\neg q)))$
  - $(p \rightarrow (q \leftrightarrow (\neg(p \rightarrow (\neg q)))))$
- Define precedence and associativity: parentheses are needed only when the formula deviates from the precedence.

Removing Ambiguity: formation trees are unique, linear representation such as $p \rightarrow q \leftrightarrow \neg p \rightarrow \neg q$ are not. There are a few ways to resolve this ambiguity.

- Define **precedence** and **associativity**: parentheses are needed only when the formula deviates from the precedence. We naturally recognize $a * b * c + d * e$ as $(((a * b) * c) + (d * e))$. Similarly.
  - From high to low precedence: $\neg, \wedge, \uparrow, \vee, \downarrow, \rightarrow, \leftrightarrow$
  - Assume right associativity, i.e. $a \vee b \vee c$ means $(a \vee (b \vee c))$.

  With minimal use of parentheses, the previous two formulation trees can be represented as:
  - $p \rightarrow q \leftrightarrow \neg p \rightarrow \neg q$
  - $p \rightarrow (q \leftrightarrow \neg(p \rightarrow \neg q))$

Structural Induction

Theorem 1 (2.12)

To show property(A) for all formulas $A \in \mathcal{F}$, it suffices to show:

- *Base case: property(p) holds for all atoms $p \in \mathcal{P}$*
- *Induction step:*
  - *Assuming property(A), the property$(\neg A)$ holds.*
  - *Assuming property(A_1) and property(A_2), then property$(A_1$ op $A_2)$ hold, for each of the binary operators.*

Exercise: Prove that every propositional formula can be equivalently expressed using only $\uparrow$.

Interpretations

Definition 2 (2.15)

Let $A \in \mathscr{F}$ be a formula and let $\mathscr{P}_A$ be the set of atoms appearing in A. An *interpretation* for A is a total function $\mathscr{I}_A : \mathscr{P}_A \to \{T, F\}$ that assigns one of the *truth values* to *every* atom in $\mathscr{P}_A$.

Definition 3 (2.16)

Let $\mathscr{I}_A$ be an interpretation for $A \in \mathscr{F}$. $\nu_{\mathscr{I}_A}(A)$, the truth value of A under $\mathscr{I}_A$, is defined inductively on the structure of A.

$v_{\mathscr{I}}(A) = \mathscr{I}_A(A)$	if A is an atom
$v_{\mathscr{I}}(\neg A) = T$	if $v_{\mathscr{I}}(A) = F$
$v_{\mathscr{I}}(\neg A) = F$	if $v_{\mathscr{I}}(A) = T$
$v_{\mathscr{I}}(A_1 \vee A_2) = F$	if $v_{\mathscr{I}}(A_1) = F$ and $v_{\mathscr{I}}(A_2) = F$
$v_{\mathscr{I}}(A_1 \vee A_2) = T$	otherwise
$v_{\mathscr{I}}(A_1 \wedge A_2) = T$	if $v_{\mathscr{I}}(A_1) = T$ and $v_{\mathscr{I}}(A_2) = T$
$v_{\mathscr{I}}(A_1 \wedge A_2) = F$	otherwise
$v_{\mathscr{I}}(A_1 \rightarrow A_2) = F$	if $v_{\mathscr{I}}(A_1) = T$ and $v_{\mathscr{I}}(A_2) = F$
$v_{\mathscr{I}}(A_1 \rightarrow A_2) = T$	otherwise
$v_{\mathscr{I}}(A_1 \uparrow A_2) = F$	if $v_{\mathscr{I}}(A_1) = T$ and $v_{\mathscr{I}}(A_2) = T$
$v_{\mathscr{I}}(A_1 \uparrow A_2) = T$	otherwise
$v_{\mathscr{I}}(A_1 \downarrow A_2) = T$	if $v_{\mathscr{I}}(A_1) = F$ and $v_{\mathscr{I}}(A_2) = F$
$v_{\mathscr{I}}(A_1 \downarrow A_2) = F$	otherwise
$v_{\mathscr{I}}(A_1 \leftrightarrow A_2) = T$	if $v_{\mathscr{I}}(A_1) = v_{\mathscr{I}}(A_2)$
$v_{\mathscr{I}}(A_1 \leftrightarrow A_2) = F$	if $v_{\mathscr{I}}(A_1) \neq v_{\mathscr{I}}(A_2)$
$v_{\mathscr{I}}(A_1 \oplus A_2) = T$	if $v_{\mathscr{I}}(A_1) \neq v_{\mathscr{I}}(A_2)$
$v_{\mathscr{I}}(A_1 \oplus A_2) = F$	if $v_{\mathscr{I}}(A_1) = v_{\mathscr{I}}(A_2)$

Truth Tables

Definition 4 (2.20)

Let $A \in \mathscr{F}$ and suppose that there are n atoms in $\mathscr{P}_A$. A truth table is a table with $n+1$ columns and 2^n rows. There is a column for each atom in $\mathscr{P}_A$, plus a column for the formula A. The first n columns specify the interpretation $\mathscr{I}$ that maps atoms in $\mathscr{P}_A$ to $\{T, F\}$. The last column shows $\nu_{\mathscr{I}}(A)$, the truth value of A for the interpretation $\mathscr{I}$.

Example 1

Let $A = (p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p)$ and let $\mathscr{I}$ an interpretation such that $\mathscr{I}(p) = F$ and $\mathscr{I}(q) = T$, and $\mathscr{I}(p_i) = T$ for all other $p_i \in \mathcal{P}$. Extend $\mathscr{I}$ to $\nu_{\mathscr{I}}(A)$, the truth value of A.

1. $\nu_{\mathscr{I}}(p \rightarrow q) = T$
2. $\nu_{\mathscr{I}}(\neg q) = F$
3. $\nu_{\mathscr{I}}(\neg p) = T$
4. $\nu_{\mathscr{I}}(\neg q \rightarrow \neg p) = T$
5. $\nu_{\mathscr{I}}((p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p)) = T$

Example 2

$\nu_{\mathscr{I}}(p \rightarrow (q \rightarrow p)) = T$, but $\nu_{\mathscr{I}}((p \rightarrow q) \rightarrow p) = F$. This shows that $p \rightarrow q \rightarrow p$ is ambiguous.

Propositional Logic: Semantics (2/3)

CS402, Spring 2017

Shin Yoo

Overview

- Logical Equivalence and Substitution
- Satisfiability, Validity, and Consequence

Logical Equivalence

Definition 1 (2.26)

Let $A_1, A_2 \in \mathscr{F}$. If $\nu_{\mathscr{I}}(A_1) = \nu_{\mathscr{I}}(A_2)$ for *all* interpretations $\mathscr{I}$, then A_1 is *logically equivalent* to A_2, denoted $A_1 \equiv A_2$.

$\mathscr{I}(p)$	$\mathscr{I}(q)$	$\nu_{\mathscr{I}}(p \vee q)$	$\nu_{\mathscr{I}}(q \vee p)$
T	T	T	T
T	F	T	T
F	T	T	T
F	F	F	F

Logical Equivalence

We can extend the result of the previous example from atomic propositions to general formulas.

Theorem 1 (2.28)

Let A_1 and A_2 be any formulas. Then $A_1 \vee A_2 \equiv A_2 \vee A_1$.

Proof.

1. Let $\mathscr{I}$ be an arbitrary interpretation for $A_1 \vee A_2$. Then, $\mathscr{I}$ is also an interpretation for $A_2 \vee A_1$, because $\mathscr{P}_{A_1} \cup \mathscr{P}_{A_2} = \mathscr{P}_{A_2} \cup \mathscr{P}_{A_1}$.
2. Similarly, $\mathscr{I}$ is an interpretation for A_1 and A_2.
3. Therefore, $\nu_{\mathscr{I}}(A_1 \vee A_2) = T \leftrightarrow (\nu_{\mathscr{I}}(A_1) = T \vee \nu_{\mathscr{I}}(A_2) = T) \leftrightarrow \nu(A_2 \vee A_1) = T$.

Logical Equivalence

Theorem 2 (2.29)

$A_1 \equiv A_2$ if and only if $A_1 \leftrightarrow A_2$ is true in every interpretation.

- **Object Language:** the language we set out to study, i.e. propositional logic in our current case.
- **Metalanguage:** the language that is used to discuss an object language.

What is the difference between $\leftrightarrow$ and $\equiv$?

- **Material Equivalence** ($\leftrightarrow$): just another statement in the object language; truth value depends on the specific interpretation.
- **Logical Equivalance** ($\equiv$): semantic statement, i.e. if p is logically equivalent to q, it means that under every possible interpretation, p and q logically means the same thing. This is a statement in the metalanguage.

Logical Substitution

Logical equivalance justifies *substitution* of one formula for another that is equivalent.

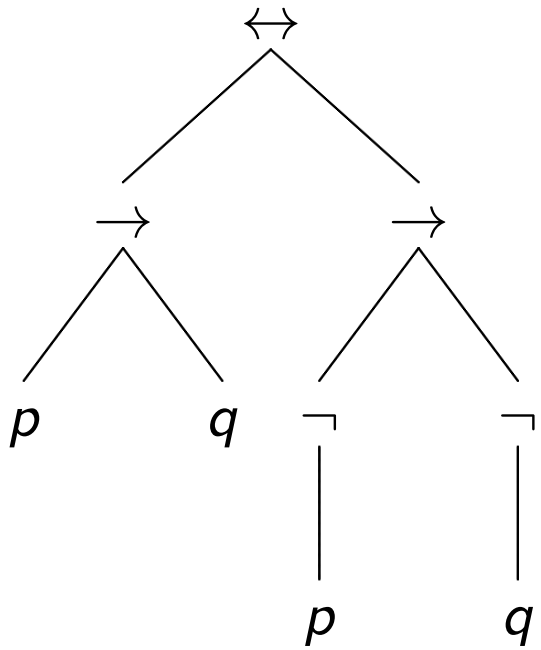
Let us present the intermediate steps first.

Definition 2 (2.30)

A is subformula of B if the formation tree for A occurs as a subtree of the formation tree for B. A is *proper* subformula of B if A is a subformula of B, but A is not identical to B.

Example 1 (2.31)

The formula $(p \rightarrow q) \leftrightarrow (\neg p \rightarrow \neg q)$ contains the following proper subformulas: $p \rightarrow q, \neg p \rightarrow \neg q, \neg p, \neg q, p$ and q



Definition 3 (2.32)

If A is a subformula of B, and A' is an arbitrary formula, then B', the *substitution* of A' for A in B, denoted $B\{A \leftarrow A'\}$, is the formula obtained by replacing all occurrences of the subtree for A in B by the tree for A'.

Theorem 3 (2.34)

Let A be a subformula of B and let A' be a formula such that $A \equiv A'$. Then $B \equiv B\{A \leftarrow A'\}$.

Substitution can be naturally used to *simplify* formulas.

$$p \wedge (\neg p \vee q) \equiv (p \wedge \neg p) \vee (p \wedge q) \equiv \mathit{false} \vee (p \wedge q) \equiv p \wedge q$$

Logical Equivalence

Definition 4 (2.35)

A binary operator, o, is *defined from* a set of operators, $O = \{o_1, \ldots, o_n\}$ iff there is a logical equivalence $A_1 \; o \; A_2 \equiv A$ where A is a formula constructed from occurrences of A_1, A_2, and operators in O.

Similarly, an unary operator o is *defined from* a set of operators, $O = \{o_1, \ldots, o_n\}$ iff there is a logical equivalence $o \; A_1 \equiv A$ where A is a formula constructed from occurrences of A_1, and operator o.

Example 2

- $\leftrightarrow$ is defined from $\{\rightarrow, \wedge\}$ because $A \leftrightarrow B \equiv (A \rightarrow B) \wedge (B \rightarrow A)$.
- $\rightarrow$ is defined from $\{\neg, \vee\}$ because $A \rightarrow B \equiv \neg A \vee B$.
- $\wedge$ is defined from $\{\neg, \vee\}$ because $A \wedge B \equiv \neg(\neg A \vee \neg B)$.

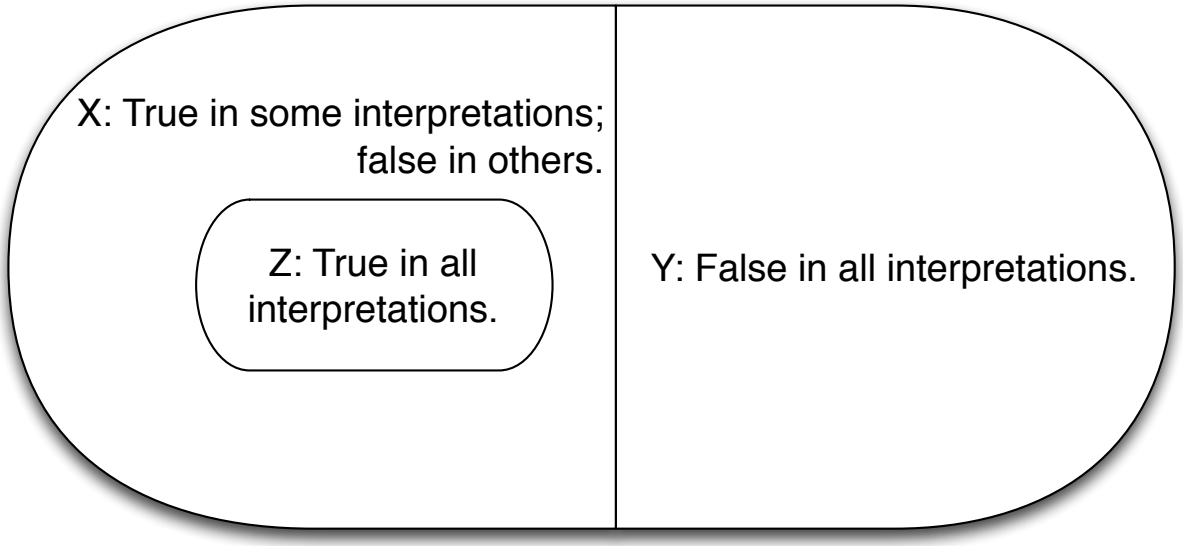
Satisfiability, Validity, and Consequences

Definition 5 (2.38)

- A propositional formula A is *satisfiable* iff $\nu_{\mathscr{I}}(A) = T$ for *some* interpretation $\mathscr{I}$.
- A satisfying interpretation is called a *model* for A.
- A is *valid*, denoted $\models A$, iff $\nu_{\mathscr{I}}(A) = T$ for *all* interpretation $\mathscr{I}$.
- A valid propositional formula is also called a *tautology*.
- A is *unsatisfiable* if and only if it is not satisfiable, that is, if $\nu_{\mathscr{I}}(A) = F$ for *all* interpretations $\mathscr{I}$.
- A is falsifiable, denoted $\not\models A$, if and only if it is not valid, that is, if $\nu_{\mathscr{I}}(A) = F$ for *some* interpretaion $\mathscr{I}$.

Theorem 4 (2.39)

A is valid iff $\neg A$ is unsatisfiable. A is satisfiable iff $\neg A$ is falsifiable.



- X (and, therefore, Z): Satisfiable.
- Y: Unsatisfiable.
- Z: Valid.
- $(X - Z) \bigcup Y$: Falsifiable (i.e. can be shown to be false).

Definition 6 (2.40)

Let $\mathscr{U} \subseteq \mathscr{F}$ be a set of formulas. An algorithm is a *decision procedure* for $\mathscr{U}$ if given an arbitrary formula $A \in \mathcal{F}$, it terminates and return the answer 'yes' if $A \in \mathscr{U}$ and the answer 'no' if $A \notin \mathscr{U}$.

By Theorem 2.39, a decision procedure for satisfiability can be used as a decision procedure for validity. Let $\mathcal{V}$ be the set of all satisfiable formulas. To decide the validity of A, we can apply the decision procesure for satisfiability of $\neg A$. This decision procedure is called a *refutation procedure*.

Example 3

Is $(p \rightarrow q) \leftrightarrow (\neg p \rightarrow \neg q)$ valid?

Example 4

$p \vee q$ is satisfiable but not valid.

Logical Consequence

Definition 7 (2.42)

Extension of satisfiability from a single formula to a set of formulas: a set of formulas $U = A_1, \ldots, A_n$ is (*simultaneously*) *satisfiable* iff there exists an interpretation $\mathscr{I}$ such that $\nu_{\mathscr{I}}(A_1) = \ldots = \nu_{\mathscr{I}}(A_n) = T$. The satisfying interpretation is called a *model* of U. U is *unsatisfiable* iff for every interpretation $\mathscr{I}$, there exists an i such that $\nu_{\mathscr{I}}(A_i) = F$.

Logical Consequence

Definition 8 (2.48)

Let U be a set of formulas and A a formula. A is a *logical consequence* of U, denoted $U \models A$, iff every model of U is a model of A.

Theorem 5 (2.50)

$U \models A$ *iff* $A_1 \wedge A_2 \ldots \wedge A_n \rightarrow A$, *where* $U = \{A_1, \ldots, A_n\}$.

If $U = \emptyset$, the logical consequence is the same as the validity.

Theories

Logical consequence is the central concept in the foundations of mathematics; validity is often trivial and not very interesting. For example, Euclidean geometry is an extensive set of logical consequences, all deduced from the five axioms.

Definition 9 (2.55)

Let $\mathscr{T}$ be a set of formulas. $\mathscr{T}$ is *closed under logical consequence* iff for all formulas A, if $\mathscr{T} \models A$ then $A \in \mathscr{T}$. A set of formulas that is closed under logical consequence is a *theory*. The elements of $\mathscr{T}$ are theorems.

Definition 10

Let $\mathscr{T}$ be a theory. $\mathscr{T}$ is said to be *axiomatizable* iff there exists a set of formulas U such that $\mathscr{T} = \{A | U \models A\}$. The set of formulas U are the axioms of $\mathscr{T}$. If U is finite, $\mathscr{T}$ is said to be *finitely axiomatizable*.

Examples of Theory

	p	q	r	$p \vee q \vee r$	$q \rightarrow r$	$r \rightarrow p$
$\mathscr{I}_1$	T	T	T	T	T	T
$\mathscr{I}_2$	T	T	F	T	F	T
$\mathscr{I}_3$	T	F	T	T	T	T
$\mathscr{I}_4$	T	F	F	T	T	T
$\mathscr{I}_5$	F	T	T	T	T	F
$\mathscr{I}_6$	F	T	F	T	F	T
$\mathscr{I}_7$	F	F	T	T	T	F
$\mathscr{I}_8$	T	F	F	F	T	T

- $U = \{p \vee q \vee r, q \rightarrow r, r \rightarrow p\}$
- Interpretation ν_1, ν_3, ν_4 are models of U (i.e. interpretations that make all formulas in U true).
- Which of the following are true?
  1. $U \models p$
  2. $U \models q \rightarrow r$
  3. $U \models r \vee \neg q$
  4. $U \models p \wedge \neg q$

Examples of Theory

	p	q	r	$p \vee q \vee r$	$q \rightarrow r$	$r \rightarrow p$
ν_1	T	T	T	T	T	T
ν_2	T	T	F	T	F	T
ν_3	T	F	T	T	T	T
ν_4	T	F	F	T	T	T
ν_5	F	T	T	T	T	F
ν_6	F	T	F	T	F	T
ν_7	F	F	T	T	T	F
ν_8	T	F	F	F	T	T

Theory of $U = \{p \vee q \vee r, q \rightarrow r, r \rightarrow p\}$, i.e. $\mathcal{T}$ (U):

- $U \subseteq \mathcal{T}(U)$ because for all formula $A \in \mathcal{F}$, $A \models A$.
- $p \in \mathcal{T}(U)$ because $U \models p$.
- $(q \rightarrow r) \in \mathcal{T}(U)$ because $U \models (q \rightarrow r)$.
- $p \wedge (q \rightarrow r) \in \mathcal{T}(U)$ because $U \models p \wedge (q \rightarrow r)$.

Theory of Euclidean Geometry is based on the set of 5 axioms, $U = A_1, A_2, A_3, A_4, A_5$ such that:

- A_1: Any two points can be joined by a unique straight line.
- A_2: Any straight line segment can be extended indefinitely in a straight line.
- A_3: Given any straight line segment, a circle can be drawn having the segment as radius and one endpoint as center.
- A_4: All right angles are congruent.
- A_5: For every line l and for every point P that does not lie on l, there exists a unique line m through P that is parallel to l.

The ancient Greeks suspected whether A_5 is a logical consequence of the other four. For about 2,000 years, various mathematicians tried to show $\{A_1, \ldots, A_4\} \models A_5$. Only in 1868, Beltrami showed that A_5 is independent from the rest. In other words, we accept A_5 as an axiom.

Beltrami also showed that non-Euclidean geometry (i.e. U with A_5 replaced with alternatives) is *consistent*.

Propositional Logic: Semantics (3/3)

CS402, Spring 2017

Shin Yoo

Overview

- Semantic Tableaux
- Soundness and completeness

Semantic Tableaux

A relatively efficient algorithm for deciding satisfiability in the propositional calculus.

- Search systematically for a model.
- If one is found, the formula is satisfiable; otherwise, it is unsatisfiable.

This method is the main tool for proving general theorems about the calculus.

Semantic Tableaux

Definition 1 (2.57)

A *literal* is an atom or a negation of an atom. An atom is a positive literal and the negation of an atom is a negative literal. For any atom p, $\{p, \neg p\}$ is a *complementary* pair of literals. For any formula A, $\{A, \neg A\}$ is a *complementary* pair of formulas. A is the complement of $\neg A$ and $\neg A$ is the complement of A.

Important observation: a set of litrals is *satisfiable* if and only if it does **not** contain a *complementary* pair of literals.

Semantic Tableaux

Analyze the satisfiablity of $A = p \wedge (\neg q \vee \neg p)$ in an arbitrary interpretation $\mathscr{I}$.

$\nu_{\mathscr{I}}(A) = T$ iff both $\nu_{\mathscr{I}}(p) = T$ and $\nu_{\mathscr{I}}(\neg q \vee \neg p) = T$.

Hence, $\nu_{\mathscr{I}}(A) = T$ if and only if either:

1. $\nu_{\mathscr{I}}(p) = T$ and $\nu_{\mathscr{I}}(\neg q) = T$ or
2. $\nu_{\mathscr{I}}(p) = T$ and $\nu_{\mathscr{I}}(\neg p) = T$

$\therefore$ A is satisfiable if and only if there exists an interpretation such that (1) holds or (2) holds.

The process is to reduce the question from one about the satisfiability of a formula to one about the satisfiability of sets of *literals*. Since any formula contains *finite* atoms, there are at most *finite* number of sets of literals. Then the decision on satisfiability becomes trivial.

Semantic Tableaux

Formula $B = (p \vee q) \wedge (\neg p \wedge \neg q)$ under an arbitrary interpretation $\mathscr{I}$.

$\nu_{\mathscr{I}}(B) = T$ iff $\nu_{\mathscr{I}}(p \vee q) = T$ and $\nu_{\mathscr{I}}(\neg p \wedge \neg q) = T$.

Hence, $\nu_{\mathscr{I}}(B) = T$ iff $\nu_{\mathscr{I}}(p \vee q) = \nu_{\mathscr{I}}(\neg p) = \nu_{\mathscr{I}}(\neg q) = T$.

Hence, $\nu_{\mathscr{I}}(B) = T$ iff either:

1. $\nu_{\mathscr{I}}(p) = \nu_{\mathscr{I}}(\neg p) = \nu_{\mathscr{I}}(\neg q) = T$, or
2. $\nu_{\mathscr{I}}(q) = \nu_{\mathscr{I}}(\neg p) = \nu(\neg q) = T$.

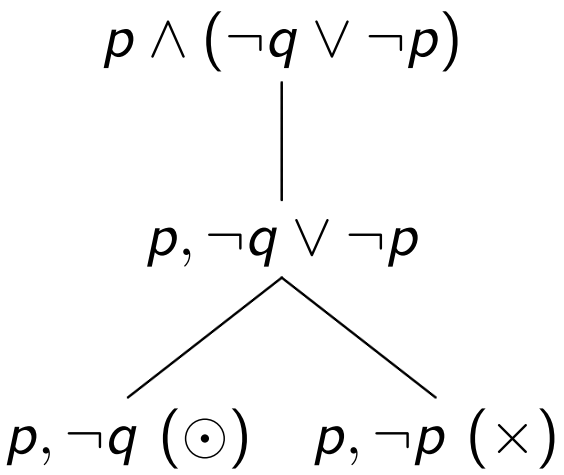
Since both $\{p, \neg p, \neg q\}$ and $\{q, \neg p, \neg q\}$ contain complementary pairs, B is unsatisfiable.

Semantic tableaux

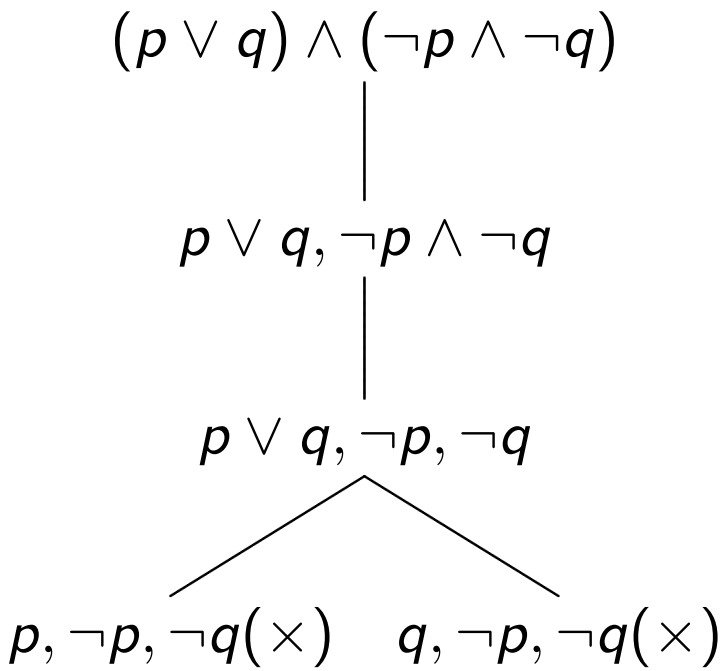
- This systematic search becomes easier if we use a suitable data structure to keep track of the assignments that must be made to subformulas.
- In semantic tableaux, trees are used.
- A leaf containing a complementary set of literals will marked with a $\times$ symbol, while a leaf containing a satisfiable set of literals will be marked with a $\odot$ symbol.

Semantic tableaux

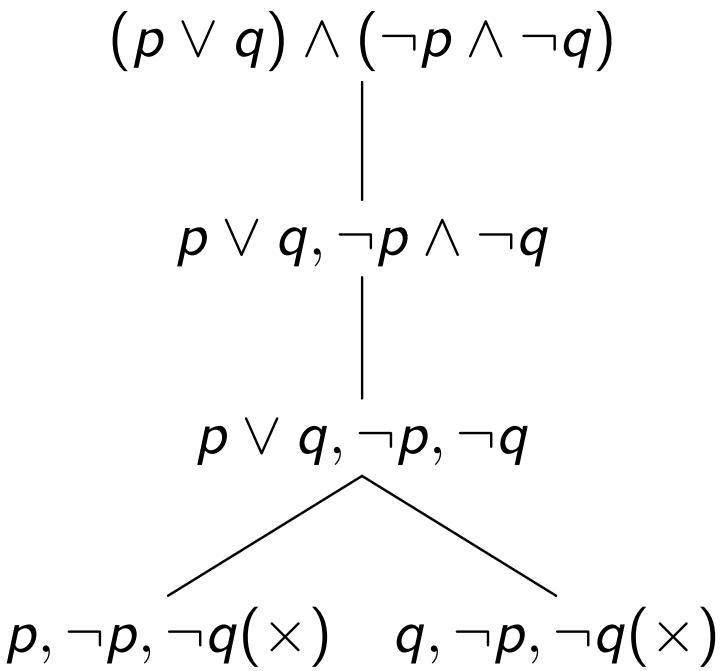
Is $p \wedge (\neg q \vee \neg p)$ satisfiable?

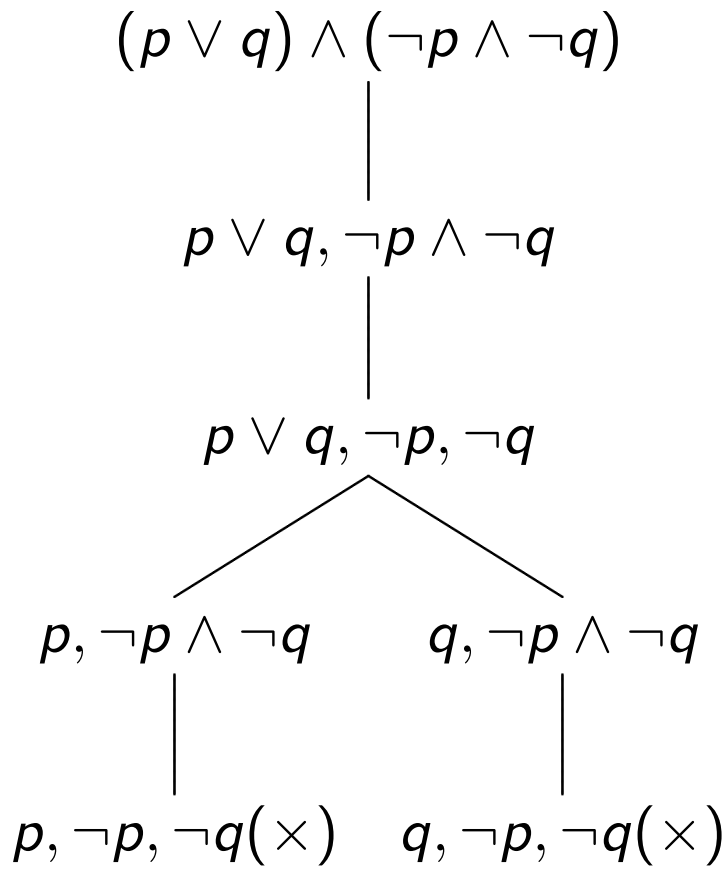


Is $(p \vee q) \wedge (\neg p \wedge \neg q)$ satisfiable?



The tableau construction is not unique.





Semantic tableaux

Classification of formulas according to their **principal operators**:

- α-formulas are conjunctive and are satisfiable only if both subformulas, α_1 and α_2, are satisfied.
- β-formulas are disjunctive and are satsified if at least one of the subformulas, β_1 or β_2, is satisfiable.

α	α_1	α_2
$\neg\neg A_1$	A_1	
$A_1 \wedge A_2$	A_1	A_2
$\neg(A_1 \vee A_2)$	$\neg A_1$	$\neg A_2$
$\neg(A_1 \rightarrow A_2)$	A_1	$\neg A_2$
$\neg(A_1 \uparrow A_2)$	A_1	A_2
$A_1 \downarrow A_2$	$\neg A_1$	$\neg A_2$
$A_1 \leftrightarrow A_2$	$A_1 \rightarrow A_2$	$A_2 \rightarrow A_1$
$\neg(A_1 \oplus A_2)$	$A_1 \rightarrow A_2$	$A_2 \rightarrow A_1$

β	β_1	β_2
$\neg(B_1 \wedge B_2)$	$\neg B_1$	$\neg B_2$
$B_1 \vee B_2$	B_1	B_2
$B_1 \rightarrow B_2$	$\neg B_1$	B_2
$B_1 \uparrow B_2$	$\neg B_1$	$\neg B_2$
$\neg(B_1 \downarrow B_2)$	B_1	B_2
$\neg(B_1 \leftrightarrow B_2)$	$\neg(B_1 \rightarrow B_2)$	$\neg(B_2 \rightarrow B_1)$
$B_1 \oplus B_2$	$\neg(B_1 \rightarrow B_2)$	$\neg(B_2 \rightarrow B_1)$

Semantic tableaux

Let $\mathcal{T}$ for a propositional formula A be a tree, whose nodes are all labeled with a set of formulas. Let $U(l)$ be the set of formulas of leaf l.

CONSTRUCTION OF SEM. TAB.(Algorithm 2.64)
Input: A propositional formula A
Output: A semantic tableaux $\mathcal{T}$ for A with marked leaves

(1) $\mathcal{T} \leftarrow$ a tree with a single node labeled $\{A\}$
(2) **while** there exists an unmarked leaf
(3) **foreach** unmarked leaf l
(4) **if** $U(l)$ is a set of lit.
(5) **if** a compl. lit. pair $\in U(l)$ **then** Mark l as $\times$
(6) **else** Mark l as $\oplus$
(7) **else**
(8) Choose $A \in U(l)$
(9) **if** $A == \alpha$ **then** Add l' to l s.t. $U(l') \leftarrow (U(l) - \{\alpha\} \cup \{\alpha_1, \alpha_2\}$
(10) **if** $A == \beta$ **then** Add l', l'' to l s.t. $U(l') \leftarrow (U(l) - \{\beta\}) \cup \{\beta_1\}$, $U(l'') \leftarrow (U(l) - \{\beta\}) \cup \{\beta_2\}$

This is not deterministic due to the choice of leaves in line (3).

Semantic tableaux

Definition 2 (2.65)

- A tableau whose construction has terminated is called a *completed tableau*.
- A completed tableau is *closed* if all leaves are marked closed (i.e. $\times$); otherwise, it is *open*.

Theorem 1 (2.66)

The construction of a semantic tableau terminates.

Soundness and Completeness

A tool operates on a formula ϕ at the syntactic level, i.e. it does not apply all possible interpretations.

- A tool is *sound* if whenever the tool says that a formula ϕ is valid (validity, not satisfiability), ϕ is really valid. That is, $\vdash \phi$ implies $\models \phi$.
- A tool is complete if whenever ϕ is valid, the tool does say that ϕ is valid. That is, $\models \phi$ implies $\vdash \phi$.
  - Writing in a contra-positive way: a tool (or method) is complete if whenever the tool says that ϕ is not valid, then ϕ is really not valid.
- Therefore, if a tool is sound and complete, then the tool says that ϕ is valid iff ϕ is really valid.

Note that:

- If a dumb tool always says that ϕ is not valid, then that tool is still sound.
- If a dumb tool always says that ϕ is valid, then that tool is still complete.

Soundness and Completeness

Theorem 2 (2.67)

Let $\mathcal{T}$ be a completed tableau for a formula A. A is unsatisfiable if and only if $\mathcal{T}$ is closed.

Corollary 1 (2.68)

A is satisfiable if and only if $\mathcal{T}$ is open.

Corollary 2 (2.69)

A is valid if and only if the tableau for $\neg A$ is closed.

Corollary 3 (2.70)

The method of semantic tableaux is a decision procedure for validity in the propositional calculus.

Soundness

Proof of soundness:

- If the tableau $\mathcal{T}$ for a formula A closes, then A is unsatisfiable.
- If a subtree rooted at node n of $\mathcal{T}$ closes, then the set of formulas $U(n)$ labeling n is unsatisfiable. Let h be the height of the node n in $\mathcal{T}$.
  - If $h = 0$, n is a leaf. Since $\mathcal{T}$ closes, $U(n)$ contains a complementary set of literals. Hence $U(n)$ is unsatisfiable.

Soundness

- If $h > 0$, either α- or β- rule was used in creating the child(ren) of n:
  - Case 1: α-rule. $U(n) = \{A_1 \wedge A_2\} \cup U_0$ and $U(n') = \{A_1, A_2\} \cup U_0$ for some set of formulas U_0.
  - The height of n' is $h - 1$; by induction, $U(n')$ is unsatisfiable since the subtree rooted at n' closes.
  - Let ν be an arbitrary interpretation. Since $U(n')$ is unsatisfiable, $\nu(A') = F$ for some $A' \in U(n')$. There are three possibilities:
    - For some $A_0 \in U_0$, $\nu(A_0) = F$. But $A_0 \in U_0 \subseteq U(n)$.
    - $\nu(A_1) = F, \nu(A_1 \wedge A_2) = F$. And $A_1 \wedge A_2 \in U(n)$.
    - $\nu(A_2) = F, \nu(A_1 \wedge A_2) = F$. And $A_1 \wedge A_2 \in U(n)$.

    In all three cases, $\nu(A) = F$ for some $A \in U(n)$. Therefore, $U(n)$ is unsatisfiable.

Soundness

- If $h > 0$, either α- or β- rule was used in creating the child(ren) of n:
  - Case 2: β-rule. $U(n) = \{B_1 \vee B_2\} \cup U_0$, $U(n) = \{B_1\} \cup U_0$ and $U(n'') = \{B_2\} \cup U_0$ for some set of formulas U_0.
  - By induction, both $U(n')$ and $U(n'')$ are unsatisfiable, since the subtrees rooted at n' and n'' close.
  - Let ν be an arbitrary interpretation. There are three possibilities:
    - $U(n')$ and $U(n'')$ are unsatisfiable, because $\nu(B_0) = F$ for some $B_0 \in U_0$. But $B_0 \in U_0 \subseteq U(n)$.
    - Otherwise, $\nu(B_0) = T$ for all $B_0 \in U_0$. Since both $U(n')$ and $U(n'')$ are unsatisfiable, $\nu(B_1) = \nu(B_2) = F$. By definition of ν on $\vee$, $\nu(B_1 \vee B_2) = F$, and $B_1 \vee B_2 \in U(n)$.

    Therefore $\nu(B) = F$ for some $B \in U(n)$; since ν was arbitrary, $U(n)$ is unsatisfiable.

Completeness

Proof of completeness:

- If A is unsatisfiable, then every tableau for A closes.
- Contrapositive statement (Cor 2.68): if some tableau for A is open (i.e., if some tableau for A has an open branch), then the formula A is satisfiable.

Completeness

Definition 3 (2.75)

Let U be a set of formulas. U is a **Hintikka**[a] set iff:

1. For all atoms p appearing in a formula of U, either $p \notin U$ or $\neg p \notin U$.
2. If $\alpha \in U$ is an α-formula, then $\alpha_1 \in U$ and $\alpha_2 \in U$.
3. If $\beta \in U$ is an β-formula, then either $\beta_1 \in U$ or $\beta_2 \in U$.

[a] Named after Finnish logician Jaakko Hintikka (1929-2015).

Completeness

Let us first deal with the following theorem, which we will then use to prove the completeness.

Theorem 3 (2.77)

Let l be an open leaf in a completed tableau $\mathcal{T}$. Let $U = \bigcup_i U(i)$, where i runs over the set of nodes on the branch from the root to l. Then U is a Hintikka set.

Proof.

Literal p or $\neg p$ cannot be decomposed. Thus, if a literal p or $\neg p$ appears for the first time in $U(n)$ for some n, the literal will be copied into $U(k)$ for all nodes k on the branch from n to l, in particular, $p \in U(l)$ or $\neg p \in U(l)$. This means that all literals in U appear in $U(l)$. Since the branch is open, no complementary pair of literals appears in $U(l)$, so Condition (1) for Hitikka set holds. $\square$

Proof for Theorem 2.77 Cont.

Suppose that $A \in U$ is an α-formula. Since the tableau is completed, A was the formula selected for decomposing at some node n in the branch from the root to l. Then $\{A_1, A_2\} \subseteq U(n') \subseteq U$, so Condition (2) holds.

Suppose that $B \in U$ is an β-formula. Since the tableau is completed, B was the formula selected for decomposing at some node n in the branch from the root to l. Then either $B_1 \in U(n') \subseteq U$ or $B_2 \in U(n') \subseteq U$, so Condition (3) holds. □

Completeness

Theorem 4 (2.78)

Hintikka's Lemma: *Let U be a Hintikka set. Then U is satisfiable.*

Proof.

Let us define an interpretation $\mathscr{I}$ based on the fact that U is a Hintikka set, and then show $\mathscr{I}$ is a model of U.

Let $\mathscr{I} : \mathscr{P}_U \rightarrow \{T, F\}$ be:

- $\mathscr{I}(p) = T$ if $p \in U$
- $\mathscr{I}(p) = F$ if $\neg p \in U$
- $\mathscr{I}(p) = T$ if $p \notin U$ and $\neg p \notin U$

Condition (1) in Definition 2.75 states that every literal is given exactly one value. The third case assigns arbitrary T to atoms that appear in U but not in literal form (i.e. $q \in \mathscr{P}_U$ but $q \notin U$ and $\neg q \notin U$). □

Proof for Theorem 2.78 Cont.

We use structural induction to show that for any $a \in U$, $\nu_{\mathscr{I}}(A) = T$. The base case is when A is an atom.

- A is an atom: $\nu_{\mathscr{I}}(A) = \nu_{\mathscr{I}}(p) = \mathscr{I}(p) = T$, because $p \in U$.
- A is a negated atom $\neg p$: $\neg p \in U$, therefore $\mathscr{I}(p) = F$, therefore $\nu_{\mathscr{I}}(A) = \nu_{\mathscr{I}}(\neg p) = T$.
- A is an α-formula: by Condition (2) of Def. 2.75, $A_1 \in U$ and $A_2 \in U$. By inductive hypothesis, $\nu_{\mathscr{I}}(A_1) = \nu_{\mathscr{I}}(A_2) = T$, so by definition of the conjunctive operator, $\nu_{\mathscr{I}}(A) = T$.
- A is an β-formula: by Condition (3) of Def. 2.75, either $B_1 \in U$ or $B_2 \in U$. By inductive hypothesis, either $\nu_{\mathscr{I}}(B_1) = T$ or $\nu_{\mathscr{I}}(B_2) = T$, so by definition of the disjunctive operator, $\nu_{\mathscr{I}}(A) = \nu_{\mathscr{I}}(B) = T$.

Completeness

Proof of Completeness for Semantic Tableaux.

Let $\mathcal{T}$ be a completed *open* tableau for A. Then U, the union of the labels of the nodes on *an open branch*, is a Hintikka set by Theorem 2.77, and a model can be found for U by Theorem 2.78. Since A is the formula labeling the root, $A \in U$, so the interpretation is a model of A. □

Propositional Logic: Normal Forms

CS402, Spring 2017

Shin Yoo

Overview

- Conjunctive Normal Forms and Validity
- Horn Clauses and Satisfiability

Note that the material corresponds to Chapter 1.5.2 and 1.5.3 of *Logic in Computer Science* by M. Huth and M. Ryan, the second reference book.

Normal Forms

Advantages of Normal Forms

- A mechanical tool can handle a formula of a normal form much easier.
- There are special algorithms to solve satisfiability of a formula very efficiently if the formula is written in some normal form.

We will cover two famous normal forms: Conjunctive normal form (CNF) and Horn clauses.

Conjunctive Normal Forms and Its Validity

Conjunctive Normal Form: a conjunction of clauses, where a clause is a disjunction of literals, i.e., **an AND of ORs**.

Any formula can be transformed into CNF.

- There exists a deterministic polynomial algorithm to convert a propositional formula into CNF[1].
- Structural induction over the formula ϕ.

Example 1

Translate the formula ϕ into CNF ϕ':

1. $\phi = (\neg p \wedge q) \rightarrow (p \wedge (r \rightarrow q))$
2. $\phi' = (p \vee \neg q \vee p) \wedge (p \vee \neg q \vee \neg r \vee q)$

[1] We are going to come back to this statement.

Translation to CNF

The translation algorithm consists of three parts:

- Transform ϕ into the implication-free form, ϕ_1.
- Transform the implication-free ϕ_1 into Negation Normal Form (NNF), ϕ_2.
- Transform the implicatio-free and NNF ϕ_2 into CNF ψ.

Implication-Free

Eliminate all implications in ϕ by replacing implication subformulas $\phi \rightarrow \psi$ with $\neg\phi \vee \psi$.

ImplFree(ϕ)
Input: a propositional formula, ϕ
Output: an implication-free formula, ϕ'

```
(1)  switch φ
(2)  case φ is a literal
(3)      return φ
(4)  case φ is φ1 → φ2
(5)      return ¬ImplFree(φ1) ∨ ImplFree(φ2)
(6)  case φ is ¬φ1
(7)      return ¬ImplFree(φ1)
(8)  case φ is φ1 op φ2, op ≠→
(9)      return ImplFree(φ1) op ImplFree(φ2)
```

Negation Normal Form

Eliminate all non-literal negations in ϕ using De Morgan's law.

$\mathrm{NNF}(\phi)$
Input: an implication-free formula, ϕ
Output: an implication-free, NNF formula, ϕ'

```
(1)   switch φ
(2)   case φ is a literal
(3)       return φ
(4)   case φ is ¬¬φ1
(5)       return NNF(φ1)
(6)   case φ is φ1 ∧ φ2
(7)       return NNF(φ1) ∧ NNF(φ2)
(8)   case φ is φ1 ∨ φ2
(9)       return NNF(φ1) ∨ NNF(φ2)
(10)  case φ is ¬(φ1 ∧ φ2)
(11)      return NNF(¬φ1 ∨ ¬φ2)
(12)  case φ is ¬(φ1 ∨ φ2)
(13)      return NNF(¬φ1 ∧ ¬φ2)
```

Conjunctive Normal Form

$\mathrm{CNF}(\phi)$
Input: an implication-free, NNF formula, ϕ
Output: a CNF formula, ϕ'

(1) **switch** ϕ
(2) **case** ϕ is a literal
(3) **return** ϕ
(4) **case** ϕ is $\phi_1 \wedge \phi_2$
(5) **return** $\mathrm{CNF}(\phi_1) \wedge \mathrm{CNF}(\phi_2)$
(6) **case** ϕ is $\phi_1 \vee \phi_2$
(7) **return** $\textsc{Distr}(\mathrm{CNF}(\phi_1), \mathrm{CNF}(\phi_2))$

DISTR Function

Essentially, recursively distribute $(p \wedge q) \vee r$ to $(p \vee r) \wedge (q \vee r)$:

$\textsc{Distr}(\eta_1, \eta_2)$
Input: CNF formulas, η_1, η_2
Output: a CNF formula for $\eta_1 \vee \eta_2$

(1) **switch** η_1, η_2
(2) **case** η_1 is $\eta_{11} \wedge \eta_{12}$
(3) **return** $\textsc{Distr}(\eta_{11}, \eta_2) \wedge \textsc{Distr}(\eta_{12}, \eta_2)$
(4) **case** η_2 is $\eta_{21} \wedge \eta_{22}$
(5) **return** $\textsc{Distr}(\eta_1, \eta_{21}) \wedge \textsc{Distr}(\eta_2, \eta_{22})$
(6) **default**
(7) **return** $\eta_1 \vee \eta_2$ //no conjunction

CNF Example

Transform $\phi = (\neg p \wedge q) \rightarrow (p \wedge (r \rightarrow q))$ into CNF.

Step 1. Elimindate implications. Let $I(\phi)$ denote $\textsc{ImplFree}(\phi)$:

$$
\begin{aligned}
I(\phi) &= \neg I(\neg p \wedge q) \vee I(p \wedge (r \rightarrow q)) \\
&= \neg (I(\neg p) \wedge I(q)) \vee I(p \wedge (r \rightarrow q)) \\
&= \neg (\neg p \wedge I(q)) \vee I(p \wedge (r \rightarrow q)) \\
&= \neg (\neg p \wedge q) \vee I(p \wedge (r \rightarrow q)) \\
&= \neg (\neg p \wedge q) \vee (I(p) \wedge I(r \rightarrow q)) \\
&= \neg (\neg p \wedge q) \vee (p \wedge I(r \rightarrow q)) \\
&= \neg (\neg p \wedge q) \vee (p \wedge (\neg I(r) \vee I(q))) \\
&= \neg (\neg p \wedge q) \vee (p \wedge (\neg r \vee I(q))) \\
&= \neg (\neg p \wedge q) \vee (p \wedge (\neg r \vee q))
\end{aligned}
$$

CNF Example

Transform $\phi = (\neg p \wedge q) \rightarrow (p \wedge (r \rightarrow q))$ into CNF.

Step 2. NNF. Let $N(\phi)$ denote $\mathrm{NNF}(\phi)$:

$$
\begin{aligned}
N(I(\phi)) &= N(\neg(\neg p \wedge q) \vee (p \wedge (\neg r \vee q))) \\
&= N(\neg(\neg p \wedge q)) \vee N(p \wedge (\neg r \vee q)) \\
&= N((\neg\neg p) \vee \neg q) \vee N(p \wedge (\neg r \vee q)) \\
&= (N((\neg\neg p)) \vee N(\neg q)) \vee N(p \wedge (\neg r \vee q)) \\
&= (p \vee N(\neg q)) \vee N(p \wedge (\neg r \vee q)) \\
&= (p \vee \neg q) \vee N(p \wedge (\neg r \vee q)) \\
&= (p \vee \neg q) \vee (N(p) \wedge N(\neg r \vee q)) \\
&= (p \vee \neg q) \vee (p \wedge N(\neg r \vee q)) \\
&= (p \vee \neg q) \vee (p \wedge (N(\neg r) \vee N(q)) \\
&= (p \vee \neg q) \vee (p \wedge (\neg r \vee N(q)) \\
&= (p \vee \neg q) \vee (p \wedge (\neg r \vee q))
\end{aligned}
$$

CNF Example

Transform $\phi = (\neg p \wedge q) \rightarrow (p \wedge (r \rightarrow q))$ into CNF.

Step 3. CNF. Let $C(\phi)$ denote $\mathrm{CNF}(\phi)$, $D(\phi_1, \phi_2)$ denote $\textsc{Distr}(\phi_1, \phi_2)$:

$$
\begin{aligned}
C(N(I(\phi))) &= C((p \vee \neg q) \vee (p \wedge (\neg r \vee q))) \\
&= D(C(p \vee \neg q), C(p \wedge (\neg r \vee q))) \\
&= D(p \vee \neg q, C(p \wedge (\neg r \vee q))) \\
&= D(p \vee \neg q, p \wedge (\neg r \vee q)) \\
&= D(p \vee \neg q, p) \wedge D(p \vee \neg q, \neg r \vee q) \\
&= (p \vee \neg q \vee p) \wedge D(p \vee \neg q, \neg r \vee q) \\
&= (p \vee \neg q \vee p) \wedge (p \vee \neg q \vee \neg r \vee q)
\end{aligned}
$$

Exercise: transform the following formula into CNF.

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

Validity of CNF Formulas

Why do we care about this particular normal form? CNF makes it very easy to check the validity of the given formula. Consider the following CNF formula: $(\neg q \vee p \vee r) \wedge (\neg p \vee r) \wedge q$. The semantic entailment holds if and only if:

$$\models \neg q \vee p \vee r, \models \neg p \vee r, \models q$$

Lemma 1 (1.43)

A disjunction of literals $L_1 \vee L_2 \vee \ldots \vee L_m$ is **valid** *if and only if there are $1 \leq i, j \leq m, i \neq j$ such that L_i is $\neg L_j$.*

Checking validity of a CNF formula boils down to searching for $L_i = \neg L_j$ in the constituent clauses: can be done in linear time.

Horn Clauses

Intuitively, a *Horn clause*[2] is a disjunction of literals with at most one positive (i.e. unnegated) literal. In other words, its disjunctive form is $\neg p \vee \neg q \vee \ldots \neg t \vee u$, which is $p \wedge q \wedge \ldots \wedge t \rightarrow u$.

Definition 1 (1.46)

A Horn formula is a formula ϕ of propositional logic if it can be generated as an instance of H in this grammar:

1. $P ::= \mathit{false} | \mathit{true} | p$
2. $A ::= P | P \wedge A$
3. $C ::= A \rightarrow P$
4. $H ::= C | C \wedge H$

That is, a *Horn formula* is a conjunction of *Horn clauses*.

[2] Named after American mathematician, Alfred Horn (1918 2001).

Examples of Horn formulas

Examples of Horn formulas

- $(p \wedge q \wedge s \rightarrow p) \wedge (q \wedge r \rightarrow p) \wedge (p \wedge s \rightarrow s)$
- $(p \wedge q \wedge s \rightarrow \mathit{false}) \wedge (q \wedge r \rightarrow p) \wedge (\mathit{true} \rightarrow s)$
- $(p_2 \wedge p_3 \wedge p_5 \rightarrow p_{13}) \wedge (\mathit{true} \rightarrow p_5) \wedge (p_5 \wedge p_{11} \rightarrow \mathit{false})$

Examples of formulas which are not Horn formulas

- $(p \wedge q \wedge s \rightarrow \neg p) \wedge (q \wedge r \rightarrow p) \wedge (p \wedge s \rightarrow s)$
- $(p \wedge q \wedge s \rightarrow \mathit{false}) \wedge (\neg q \wedge r \rightarrow p) \wedge (\mathit{true} \rightarrow s)$
- $(p_2 \wedge p_3 \wedge p_5 \rightarrow p_{13} \wedge p_{27}) \wedge (\mathit{true} \wedge p_5) \wedge (p_5 \wedge p_{11} \rightarrow \mathit{false})$
- $(p_2 \wedge p_3 \wedge p_5 \rightarrow p_{13} \wedge p_{27}) \wedge (\mathit{true} \wedge p_5) \wedge (p_5 \wedge p_{11} \vee \mathit{false})$

Horn Clauses and Satisfiability

We maintain a list of all occurrences of type P (remember: $P ::= \mathit{false} | \mathit{true} | p$, $A ::= P | P \wedge A$, $C ::= A \rightarrow P$) in formula ϕ, and iteratively *mark* each one of them as following:

1. Mark *true* if it occurs in the list.
2. If there is a conjunct $P_1 \wedge P_2 \wedge \ldots \wedge P_{k_i} \rightarrow P$ of ϕ such that all P_j with $1 \leq j \leq k_i$ are marked, mark P as well and repeat 2; otherwise, proceed to 3.
3. If *false* is marked, ϕ is unsatisfiable.
4. Else, ϕ is satisfiable.

Horn Algorithm

HORN(ϕ)
Input: A Horn formula, ϕ
Output: The satisfiability of ϕ
(1) Mark all occurrences of *true* in ϕ
(2) **while** there exists a conjunct $P_1 \wedge P_2 \wedge \ldots \wedge P_j \rightarrow P'$ of ϕ s.t. all P_js are marked but P' isn't
(3) Mark P'
(4) **if** *false* is marked **then return** UNSAT
(5) **else return** SAT

Correctness of the Horn Algorithm

Theorem 1 (1.47)

The algorithm HORN() *is correct for the satisfiability decision problem of Horn formulas and has no more than* $n + 1$ *cycles in its* `while` *statement if* n *is the number of atom is in* ϕ. *In particular,* HORN() *always terminates on correct input.*

Proof.

Termination: entering the body of the loop resulting in marking an yet-unmarked P that is not a *true* literal. Since there are only a finite number of atomic Ps in ϕ, HORN() terminates. □

Correctness of the Horn Algorithm

Corollary 1

After any number of executions of the `while` *loop, all marked P are true for all valuations in which ϕ evaluates to True.*

Proof.

When loop executes 0 times: we already marked all occurrences of *true*, which must be *True* in all valuations. Hence Corollary 1 holds.

Corollary 1 holds for k iterations: if we enter $k+1$-th iteration, the loop predicate is true, i.e., there exists a conjunct $P_1 \wedge \ldots \wedge P_{k_i} \rightarrow P$ such that all P_js are marked. Let ν be any interpretaion in which ϕ is true. By the induction hypothesis, $P_1 \wedge \ldots \wedge P_{k_i}$ is true, as well as $P_1 \wedge \ldots \wedge P_{k_i} \rightarrow P$ is true. Therefore, P' must be also true in ν. Therefore, Corollary 1 holds for $k+1$-th iteration. □

Correctness of the Horn Algorithm

Proof.

UNSAT: if *false* is marked, there exists a conjunct $P_1 \wedge \ldots \wedge P_{k_i} \rightarrow \mathit{false}$ of ϕ such that all P_is are marked. If ϕ is satisfiable, by Corollary 1, this means $(\mathit{true} \rightarrow \mathit{false}) = \mathit{false}$ whenever ϕ is true. This is impossible, so ϕ is unsatisfiable. Reductio ad absurdum.

SAT: if *false* is **NOT** marked, let ν be an interpretation that assign *true* to all marked atoms, and *false* to the others.

If ϕ is not true under ν, it means that there exists a conjunct $P_1 \wedge \ldots \wedge P_{k_i} \rightarrow P'$ of ϕ that is false. By the semantics, this can only mean that $P_1 \wedge \ldots \wedge P_{k_i}$ is true but P' is false. However, by the definition of ν, all P_is are marked, which means this conjunct has been processed by our `while` loop, resulting in P' being marked. By definition of ν, the conjunct becomes true; by Corollary 1, ϕ becomes true. Reductio ad absurdum. $\square$

Propositional Logic: Deductive Proof & Natural Deduction Part 1

CS402, Spring 2017

Shin Yoo

Deductive Proof

In propositional logic, a *valid* formula is a tautology. So far, we could show the validity of a formula ϕ in the following ways:

- Through the truth table for ϕ
- Obtain ϕ as a substitution instance of a formula known to be valid. That is, $q \rightarrow (p \rightarrow q)$ is valid, therefore $r \wedge s \rightarrow (p \vee q \rightarrow r \wedge s)$ is also valid.
- Obtain ϕ through interchange of equivalent formulas. That is, if $\phi \equiv \psi$ and ϕ is a subformula of a valid formula χ, χ' obtained by replacing all occurrences of ϕ in χ with ψ is also valid.

Deductive Proof

Goals of logic: (given U), is ϕ valid?

Theorem 1 (2.38, Ben-Ari)

$U \models \phi$ *iff* $\models A_1 \wedge \ldots \wedge A_n \rightarrow \phi$ *when* $U = \{A_1, \ldots, A_n\}$.

However, there are problems in semantic approach.

- Set of axioms may be *infinite*: for example, Peano and ZFC (Zermelo-Fraenkel set theory) theories cannot be finitely axiomatised. Hilbert system, $\mathcal{H}$, uses axiom schema, which in turn generates an infinite number of axioms. We cannot write truth tables for these.
- The truth table itself is not always there! Very few logical systems have decision procedures for validity. For example, predicate logic does not have any such decision procedure.

Semantic vs. Syntax

$\models \phi$	vs.	$\vdash \phi$
Truth		Tools
Semantics		Syntax
Validity		Proof
All Interpretations		Finite Proof Trees
Undecidable (except propositional logic)		Manual Heuristics

Deductive Proof

A deductive proof system relies on a set of proof rules (also *inference* rules), which are in themselves *syntactic transformations* following specific patterns.

- There may be an infinite number of axioms, but only a finite number of axioms will appear on any deductive proof.
- Any particular proof consists of a finite sequence of sets of formulas, and the legality of each individual deduction can be easily and efficiently determined from the syntax of the formulas.
- The proof of a formula clearly shows which axioms, theorems and rules are used and for what purposes.

Soundness and Completeness

- Given a logical system, its proof system is *sound* if and only if: $U \vdash \phi \rightarrow U \models \phi$.
- Given a logical system, its proof system is *complete* if and only if: $U \models \phi \rightarrow U \vdash \phi$.

Proof Calculus

Proof calculus refers to a family of formal systems that use a common style of formal inference for their inference rules. There are three classicl systems:

- Hilbert Systems, $\mathcal{H}$
- Gentzen Systems, $\mathcal{G}$. There are two variants:
  - Natural Deduction: every line has exactly one asserted propositions.
  - Sequent Calculus: every line has zero or more asserted propositions.

Natural Deduction

We have a collection of proof rules. Natural deduction does not have axioms.

- Suppose we have premises $\phi_1, \phi_2, \ldots, \phi_n$ and would like to prove a conclusion ψ. The intention is denoted by $\phi_1, \phi_2, \ldots, \phi_n \vdash \psi$ We call this expression a *sequent*; it is valid if a proof for it can be found.

Definition 1

A logical formula ϕ with the valid sequent $\vdash \phi$ is theorem.

Proof Rules

	Introduction	Elimination	
$\wedge$	$\dfrac{\phi \quad \psi}{\phi \wedge \psi} \wedge i$	$\dfrac{\phi \wedge \psi}{\phi} \wedge e_1$	$\dfrac{\phi \wedge \psi}{\psi} \wedge e_2$

- $\wedge_i$ (and-introduction): to prove $\phi \wedge \psi$, you must first prove ϕ and ψ separately and then use the rule $\wedge i$.
- $\wedge e_1$: (and-elimination) to prove ϕ, try proving $\phi \wedge \psi$ and then use the rule $\wedge e1$. Probably only useful when you already have $\phi \wedge \psi$ somwhere; otherwise, proving $\phi \wedge \psi$ may be harder than proving ϕ.

Proof Rules

	Introduction	Elimination
$\vee$	$\dfrac{\phi}{\phi \vee \psi} \vee i_1 \quad \dfrac{\psi}{\phi \vee \psi} \vee i_2$	$\dfrac{\phi \vee \psi \quad \begin{array}{c}\phi \\ \vdots \\ \chi\end{array} \quad \begin{array}{c}\psi \\ \vdots \\ \chi\end{array}}{\chi} \wedge e$

- $\vee i_1$ (or-introduction): to prove $\phi \vee \psi$, try proving ϕ. Again, in general it is harder to prove ϕ than it is to prove $\phi \vee \psi$, so this will usually be useful only if you have already managed to prove ϕ.
- $\vee e$ (or-elimination): has an excellent procedural interpretation. It says: if you have $\phi \vee \psi$, and you want to prove some χ, then try to prove χ from ϕ and from ψ in turn. In those subproofs, of course you can use the other prevailing premises as well.

Proof Rules

	Introduction	Elimination
$\rightarrow$	$\dfrac{\begin{array}{c}\phi \\ \vdots \\ \psi\end{array}}{\phi \rightarrow \psi}\ \rightarrow_i$	$\dfrac{\phi \quad \phi \rightarrow \psi}{\psi}\ \rightarrow_e$
$\neg$	$\dfrac{\begin{array}{c}\phi \\ \vdots \\ \bot\end{array}}{\neg\phi}\ \neg_i$	$\dfrac{\begin{array}{ccc}\neg\phi & \neg\phi & \neg\phi \\ & \vdots & \vdots \\ \neg\phi & \psi & \neg\psi\end{array}}{\phi}\ \neg_e$
$\bot$	(No introduction rule for $\bot$)	$\dfrac{\bot}{\phi}\ \bot_e$
$\neg\neg$		$\dfrac{\neg\neg\phi}{\phi}\ \neg\neg_e$

Derived Rules

$$\frac{\phi \rightarrow \psi \quad \neg\psi}{\neg\phi}\ MT \qquad \frac{\phi}{\neg\neg\phi}\ \neg\neg i$$

$$\begin{array}{c}\neg\phi \\ \vdots \\ \bot \\ \hline \phi\end{array}\ RAA \qquad \frac{}{\phi \vee \neg\phi}\ LEM$$

- Modus Tollens (MT): "If Abraham Lincoln was Ethiopian, then he was African. Abraham Lincoln was not African; therefore, he was not Ethiopian."
- Introduction of double negation.
- Reductio Ad Absurdum, i.e. Proof By Contradiction.
- Tertium Non Datur, or Law of the Excluded Middle.

How Proof Rules Work

Prove that $p \wedge q, r \vdash q \wedge r$.

Proof Tree

$$\dfrac{\dfrac{p \wedge q}{q}\ \wedge_{e_2} \qquad r}{q \wedge r}\ \wedge_i$$

Linear Form

1.	$p \wedge q$	premise
2.	r	premise
3.	q	$\wedge_{e_2}$, 1
4.	$q \wedge r$	$\wedge_i$, 3, 2

Scope Box

We can temporarily make any assumptions, and apply rules to them. We use scope boxes to represent their scope, i.e. to represent which other steps *depend* on them. For example, let us show that $p \rightarrow q \vdash \neg q \rightarrow \neg p$.

1.	$p \rightarrow q$	premise
2.	$\neg q$	assumption
3.	$\neg p$	modus tollens, 1, 2
4.	$\neg q \rightarrow \neg p$	$\rightarrow_i$, 2, 3

- Note that $\neg p$ *depends* on the assumption, $\neg q$. However, step 4 does not depends on step 2 or 3.
- The line immediately following a closed box has to match the pattern of the conclusion of the rule using the box.

Example 1

Prove that $p \wedge \neg q \rightarrow r, \neg r, p \vdash q$.

1.	$p \wedge \neg q \rightarrow r$	premise
2.	$\neg r$	premise
3.	p	premise
4.	$\neg q$	assumption
5.	$p \wedge \neg q$	$\wedge_i$, 3, 4
6.	r	$\rightarrow_i$, 5, 1
7.	$\bot$	$\neg_e$, 6, 2
8.	$\neg\neg q$	$\neg_i$, 4-7
9.	q	$\neg\neg_e$, 8

Example 2

Prove that $p \rightarrow q \vdash \neg p \vee q$.

1.	$p \rightarrow q$	premise
2.	$\neg p \vee p$	law of eliminated middle
3.	$\neg p$	assumption
4.	$\neg p \vee q$	$\vee_{i_3}$, 3
5.	p	assumption
6.	q	$\rightarrow_i$, 1, 5
7.	$\neg p \vee q$	$\vee_{i_2}$, 6
8.	$\neg p \vee q$	$\vee_e$, 2, 3-4, 5-7

Note that, earlier in the lecture, we also showed $p \rightarrow q \models \neg p \vee q$. Can you explain the differences?

Example 3: Law of Excluded Middle

Prove the law of excluded middle, i.e. $\dfrac{}{\phi \vee \neg\phi}\ LEM$.

1.	$\neg(\phi \vee \neg\phi)$	assumption
2.	ϕ	assumption
3.	$\phi \vee \neg\phi$	$\vee_{i_1}$, 2
4.	$\bot$	$\neg_e$, 3, 1
5.	$\neg\phi$	$\neg_i$, 2-4
6.	$\phi \vee \neg\phi$	$\vee_{i_2}$, 5
7.	$\bot$	$\neg_e$, 1, 6
8.	$\neg\neg(\phi \vee \neg\phi)$	$\neg_i$, 1-7
9.	$\phi \vee \neg\phi$	$\neg\neg_e$, 8

Proof Tips

- Write down the premises at the top.
- Write down the conclusion at the bottom.
- Observe the structure of the conclusion, and try to fit a rule backward.

Exercises

Prove the following:

- $\neg p \vee q \vdash p \rightarrow q$
- $p \rightarrow q, p \rightarrow \neg q \vdash \neg p$
- $p \rightarrow (q \rightarrow r), p, \neg r \vdash \neg q$

Propositional Logic: Deductive Proof & Natural Deduction Part 2

CS402, Spring 2017

Shin Yoo

Basis for Inference Rules

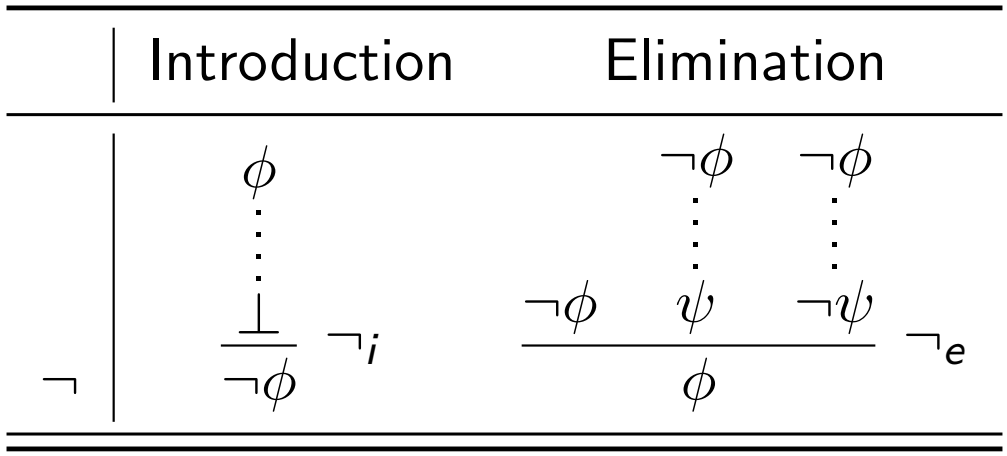
	Introduction	Elimination	
$\wedge$	$\dfrac{\phi \quad \psi}{\phi \wedge \psi}\ \wedge i$	$\dfrac{\phi \wedge \psi}{\phi}\ \wedge e_1$	$\dfrac{\phi \wedge \psi}{\psi}\ \wedge e_2$
$\vee$	$\dfrac{\phi}{\phi \vee \psi}\ \vee i_1$	$\dfrac{\psi}{\phi \vee \psi}\ \vee i_2$	$\dfrac{\phi \vee \psi \quad \begin{array}{c}\phi \\ \vdots \\ \chi\end{array} \quad \begin{array}{c}\psi \\ \vdots \\ \chi\end{array}}{\chi}\ \wedge e$
$\rightarrow$	$\dfrac{\begin{array}{c}\phi \\ \vdots \\ \psi\end{array}}{\phi \rightarrow \psi}\ \rightarrow_i$	$\dfrac{\phi \quad \phi \rightarrow \psi}{\psi}\ \rightarrow_e$	

How do we know the validity of these rules?

Truth tables. In other words, $\bigwedge$ *premise* $\models$ *consequent*

Basis for Inference Rules

How about the following?



These cannot be justified by truth tables. Rather, these are justified by the Reductio Principle of propositional logic.

Theorem 1 (Reductio Principle)

Let Γ be a set of formulas, ϕ and ψ a formula. If $\Gamma \cup \{\varphi\} \models \psi$ and $\Gamma \cup \{\varphi\} \models \neg\psi$, then $\Gamma \models \neg\varphi$. If $\Gamma \cup \{\neg\varphi\} \models \psi$ and $\Gamma \cup \{\neg\varphi\} \models \neg\psi$, then $\Gamma \models \varphi$.

Basis for Inference Rules

A related question: prove that $\wedge$ and $\vee$ cannot define $\neg$.

On Derived Rules

Repetition: since $\models P \rightarrow P$, we can derive an inference rule based on it.

$$\frac{\phi}{\phi}\ \textit{Repetition}$$

For example, this rule can be used to prove $Q \rightarrow (P \rightarrow Q)$:

1.	Q	Assumption
2.	P	Assumption
3.	Q	Repetition, 1
4.	$P \rightarrow Q$	$\rightarrow_i$, 2-3
5.	$Q \rightarrow (P \rightarrow Q)$	$\rightarrow_i$, 1-4

On Derived Rules

Now, introduction of double negation:

$$\frac{\phi}{\neg\neg\phi}$$

1.	ϕ	Assumption
2.	$\neg\phi$	Assumption
3.	ϕ	Repetition, 1
4.	$\neg\neg\phi$	$\neg_i$, 2-3 (i.e. $\neg\phi \rightarrow \neg\phi \wedge \neg\phi \rightarrow \phi$)
5.	$\phi \rightarrow \neg\neg\phi$	$\rightarrow_i$, 1-4

On Derived Rules

Elimination of double negation:

$$\frac{\neg\neg\phi}{\phi}$$

1.	$\neg\neg\phi$	Assumption
2.	$\neg\phi$	Assumption
3.	$\neg\neg\phi$	Repetition
4.	ϕ	$\neg_e$, 2-3
5.	$\neg\neg\phi \rightarrow \phi$	$\rightarrow_i$, 1-4

More Exercises

Prove the validity of the following sequents.

- $(s \rightarrow p) \vee (t \rightarrow q) \vdash (s \rightarrow q) \vee (t \rightarrow p)$
- $\vdash (p \rightarrow q) \vee (q \rightarrow r)$

Propositional Logic: Gentzen System, $\mathcal{G}$

CS402, Spring 2017

Shin Yoo

Quiz on Thursday, 6th April: 15 minutes, two questions.

Sequent Calculus in $\mathcal{G}$

In Natural Deduction, each line in the proof consists of exactly one proposition. That is, $A_1, A_2, \ldots, A_n \vdash B$.

In Sequent calculus, each line in the proof consists of zero or more propositions. That is, $A_1, A_2, \ldots, A_n \vdash B_1, B_2, \ldots, B_k$. The standard semantic is, "whenever every A_i is true, at least one B_j will also be true".

Axioms in $\mathcal{G}$

Definition 1 (3.2, Ben-Ari)

An axiom of $\mathcal{G}$ is a set of literals U containing a complementary pair.

Note that sets in $\mathcal{G}$ are implicitly disjunctive. For example, $\{\neg p, q, p\}$ is an axiom, i.e. $\vdash \neg p, q, p$ in $\mathcal{G}$.

Inference Rules in $\mathcal{G}$

Definition 2 (3.2, Ben-Ari)

There are two types of inference rules, defined with reference to tables below:

- Let $\{\alpha_1, \alpha_2\} \subseteq U_1$ and let $U_1' = U_1 - \{\alpha_1, \alpha_2\}$. Then $U = U_1' \cup \{\alpha\}$ can be inferred.
- Let $\{\beta_1\} \subseteq U_1, \{\beta_2\} \subseteq U_2$ and let $U_1' = U_1 - \{\beta_1\}, U_2' = U_2 - \{\beta_2\}$. Then $U = U_1' \cup U_2' \cup \{\beta\}$ can be inferred.

Inference Rules in $\mathcal{G}$

$$\frac{\vdash U_1' \cup \{\alpha_1, \alpha_2\}}{\vdash U_1' \cup \{\alpha\}}\ \alpha$$

α	α_1	α_2
$\neg\neg\alpha$	α	
$\neg(\alpha_1 \wedge \alpha_2)$	$\neg\alpha_1$	$\neg\alpha_2$
$\alpha_1 \vee \alpha_2$	α_1	α_2
$\alpha_1 \rightarrow \alpha_2$	$\neg\alpha_1$	α_2
$\alpha_1 \uparrow \alpha_2$	$\neg\alpha_1$	$\neg\alpha_2$
$\neg(\alpha_1 \downarrow \alpha_2)$	α_1	α_2
$\neg(\alpha_1 \leftrightarrow \alpha_2)$	$\neg(\alpha_1 \rightarrow \alpha_2)$	$\neg(\alpha_2 \rightarrow \alpha_1)$
$\alpha_1 \oplus \alpha_2$	$\neg(\alpha_1 \rightarrow \alpha_2)$	$\neg(\alpha_2 \rightarrow \alpha_1)$

That is, $\alpha-$rules build up disjunctions.

$$\frac{\vdash U_1' \cup \{\beta_1\} \quad \vdash U_2' \cup \{\beta_2\}}{\vdash U_1' \cup U_2' \cup \{\beta\}}\ \beta$$

β	β_1	β_2
$\beta_1 \wedge \beta_2$	β_1	β_2
$\neg(\beta_1 \vee \beta_2)$	$\neg\beta_1$	$\neg\beta_2$
$\neg(\beta_1 \rightarrow \beta_2)$	β_1	$\neg\beta_2$
$\neg(\beta_1 \uparrow \beta_2)$	β_1	β_2
$\beta_1 \downarrow \beta_2$	$\neg\beta_1$	$\neg\beta_2$
$\beta_1 \leftrightarrow \beta_2$	$\beta_1 \rightarrow \beta_2$	$\beta_2 \rightarrow \beta_1$
$\neg(\beta_1 \oplus \beta_2)$	$\beta_1 \rightarrow \beta_2$	$\beta_2 \rightarrow \beta_1$

That is, $\beta-$rules build up conjuntions (consider $(a \vee b) \wedge (c \vee d) \models a \vee c \vee (b \wedge d)$).

Example Proof

Prove that $\vdash p \vee (q \wedge r) \rightarrow (p \vee q) \wedge (p \vee r)$ in $\mathcal{G}$.

1.	$\vdash \neg p, p, q$	Axiom
2.	$\vdash \neg p, (p \vee q)$	$\alpha\vee$, 1
3.	$\vdash \neg p, p, r$	Axiom
4.	$\vdash \neg p, (p \vee r)$	$\alpha\vee$, 3
5.	$\vdash \neg p, (p \vee q) \wedge (p \vee r)$	$\beta\wedge$, 2, 4
6.	$\vdash \neg q, \neg r, p, q$	Axiom
7.	$\vdash \neg q, \neg r, (p \vee q)$	$\alpha\vee$, 6
8.	$\vdash \neg q, \neg r, p, r$	Axiom
9.	$\vdash \neg q, \neg r, (p \vee r)$	$\alpha\vee$, 8
10.	$\vdash \neg q, \neg r, (p \vee q) \wedge (p \vee r)$	$\beta\wedge$, 7, 9
11.	$\vdash \neg(q \wedge r), (p \vee q) \wedge (p \vee r)$	$\alpha\wedge$, 10
12.	$\vdash \neg(p \vee (q \wedge r)), (p \vee q) \wedge (p \vee r)$	$\beta\vee$, 5, 11
13.	$\vdash p \vee (q \wedge r) \rightarrow (p \vee q) \wedge (p \vee r)$	$\alpha\rightarrow$, 12

Wait...

- How do you magically come up with the axioms $\{\neg p, p, q\}$, $\{\neg p, p, r\}$, $\{\neg q, \neg r, p, q\}$, and $\{\neg q, \neg r, p, r\}$?
- Haven't we seen something like this before?

$$\vdash (p \vee q) \rightarrow (q \vee p)$$

Proof in $\mathcal{G}$

$$\dfrac{\dfrac{\neg p, q, p \quad \neg q, q, p}{\neg(p \vee q), q, p}}{\neg(p \vee q), (q \vee p)}$$
$$(p \vee q) \rightarrow (q \vee p)$$

$$\dfrac{\dfrac{\dfrac{\neg((p \vee q) \rightarrow (q \vee p))}{(p \vee q), \neg(q \vee p)}}{(p \vee q), \neg q, \neg p}}{\dfrac{p, \neg q, \neg p \quad q, \neg q, \neg p}{UNSAT \quad UNSAT}}$$

Semantic Tableau
(Sets are conjunctive)

$\mathcal{G}$ and Semantic Tableau

Theorem 1 (3.6, Ben-Ari)

Let A be a formula in propositional logic. Then $\vdash A$ in $\mathcal{G}$ if and only if there is a closed semantic tableaux for $\neg A$.

Theorem 2 (3.7, Ben-Ari)

Let U be a set of formulas and let $\bar{U}$ be the set of complements of formulas in U. Then, $\vdash U$ in $\mathcal{G}$ if and only if there is a closed semantic tableau for $\bar{U}$.

$\mathcal{G}$ and Semantic Tableau

We prove that, if there exists a closed semantic tableau for $\bar{U}$, then $\vdash U$ in $\mathcal{G}$. The opposite direction is left for you.

Proof.

Let $\mathcal{T}$ be a closed semantic tableau for $\bar{U}$. We prove $\vdash U$ by induction on h, the height of $\mathcal{T}$.

- If $h = 0$, then $\mathcal{T}$ consists of a single node labeled by $\bar{U}$. By assumption, $\mathcal{T}$ is closed, so it contains a complementary pair of literals $\{p, \neg p\}$, that is, $\bar{U} = \bar{U'} \cup \{p, \neg p\}$. Obviously, $U = U' \cup \{\neg p, p\}$ is an axiom in $\mathcal{G}$, hence $\vdash U$.

Proof. Cont.

- If $h > 0$, then some tableau rule was used on an α- or β-formula at the root of $\mathcal{T}$ on a formula $\bar{\phi} \in \bar{U}$, that is, $\bar{U} = \bar{U'} \cup \bar{\phi}$. The proof proceeds by cases, where you must be careful to distinguish between applications of the tableau rules and applications of the Gentzen rules of the same name.
  - Case 1: ϕ is an α-formula (such as) $\neg(A_1 \vee A_2)$. The tableau rule created a child node labeled by the set of formulas $\bar{U'} \cup \{\neg A_1, \neg A_2\}$. By assumption, the subtree rooted at this node is a closed tableau, so by the inductive hypothesis, $\vdash U' \cup \{A_1, A_2\}$. Using the appropriate rule of inference from $\mathcal{G}$, we obtain $\vdash U' \cup \{A_1 \vee A_2\}$, that is, $\vdash U' \cup \{\phi\}$, which is $\vdash U$.

Proof.

- If $h > 0$, then some tableau rule was used on an α- or β-formula at the root of $\mathcal{T}$ on a formula $\bar{\phi} \in \bar{U}$, that is, $\bar{U} = \bar{U'} \cup \bar{\phi}$. The proof proceeds by cases, where you must be careful to distinguish between applications of the tableau rules and applications of the Gentzen rules of the same name.
  - Case 2: ϕ is a β-formula (such as) $\neg(B_1 \wedge B_2)$. The tableau rule created two child nodes labeled by the sets of formulas $\bar{U'} \cup \{\neg B_1\}$ and $\bar{U'} \cup \{\neg B_2\}$. By assumption, the subtrees rooted at this node are closed, so by the inductive hypothesis $\vdash U' \cup \{B_1\}$ and $\vdash U' \cup \{B_2\}$. Using the appropriate rule of inference from $\mathcal{G}$, we obtain $\vdash U' \cup \{B_1 \wedge B_2\}$, that is, $\vdash U' \cup \{\phi\}$, which is $\vdash U$.

Why $\mathcal{G}$ and not natural deduction?

Taste. Or, more appropriately, aesthetics.

Natural deduction feels more, umm, natural. It is also more simplistic; having multiple disjunct on the right hand side, in $\mathcal{G}$, is clearly cumbersome and adds complexity.

$\mathcal{G}$ shows the symmetric nature of negation more vividly.

$$A_1, \ldots, A_n \vdash B_1, \ldots, B_k$$

$$\vdash (A_1 \wedge \cdots \wedge A_n) \rightarrow (B_1 \vee \cdots \vee B_k)$$

$$\vdash \neg A_1 \vee \neg A_2 \vee \cdots \vee \neg A_n \vee B_1 \vee B_2 \vee \cdots \vee B_k$$

$$\vdash \neg(A_1 \wedge A_2 \wedge \cdots \wedge A_n \wedge \neg B_1 \wedge \neg B_2 \wedge \cdots \wedge \neg B_k)$$

Soundness and Completeness of $\mathcal{G}$

Theorem 3 (3.8 in Ben-Ari)

$\models A$ *if and only if* $\vdash A$ *in* $\mathcal{G}$.

Proof.

A is valid iff $\neg A$ is unsatisfiable iff there is a closed semanti tableau for $\neg A$ iff there is a proof of A in $\mathcal{G}$. $\square$

Exercises

Prove the following in $\mathcal{G}$:

- $\vdash (A \rightarrow B) \rightarrow (\neg B \rightarrow \neg A)$
- $\vdash (A \rightarrow B) \rightarrow ((\neg A \rightarrow B) \rightarrow B)$
- $\vdash ((A \rightarrow B) \rightarrow A) \rightarrow A$

Propositional Logic: Hilbert System, $\mathcal{H}$

CS402, Spring 2017

Shin Yoo

Hilbert System, $\mathcal{H}$

Unlike $\mathcal{G}$, which deals with sets of formulas, $\mathcal{H}$ is a deductive system for single formulas. In $\mathcal{G}$, there is one definition of axioms, and multiple rules. In $\mathcal{H}$, there are many axioms, but only one rule.

Definition 1 (3.9, Ben-Ari)

$\mathcal{H}$ is a deductive system with three axiom schemes and one rule of inference. For any formulas, A, B and C, the following formulas are axioms:

- $\vdash (A \rightarrow (B \rightarrow A)$
- $\vdash ((A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$
- $\vdash (\neg B \rightarrow \neg A) \rightarrow (A \rightarrow B)$

$\mathcal{H}$ uses *Modus Ponens* (MP) as the single inference rule:

$$\frac{\vdash A \quad \vdash A \rightarrow B}{\vdash B}\ MP$$

Theorem 1 (3.10, Ben-Ari)

For any formula ϕ, $\phi \vdash \phi$.

Proof.

Think $A : \phi, B : \phi \rightarrow \phi, C : \phi$ when we refer to axiom schemes.

1.	$\vdash (\phi \rightarrow ((\phi \rightarrow \phi) \rightarrow \phi)) \rightarrow ((\phi \rightarrow (\phi \rightarrow \phi) \rightarrow (\phi \rightarrow \phi))$	Axiom 2
2.	$\vdash \phi \rightarrow ((\phi \rightarrow \phi) \rightarrow \phi)$	Axiom 1
3.	$\vdash (\phi \rightarrow (\phi \rightarrow \phi)) \rightarrow (\phi \rightarrow \phi)$	MP, 1, 2
4.	$\vdash \phi \rightarrow (\phi \rightarrow \phi)$	Axiom 1 $(A, B : \phi)$
5.	$\vdash \phi \rightarrow \phi$	MP, 3, 4

□

Hilbert System, $\mathcal{H}$

Note that $\{\rightarrow, \neg\}$ is an adequate set of opeartors, i.e. can replace all other binary operators through semantic equivalence.

However, for the sake of expressiveness, we introduce new rules of inference, called *derived rules*, to $\mathcal{H}$. We then use use derived rules to transform a proof into another (usually longer) proof, which uses just the original axioms and MP.

Consequently, derived rules should be proven to be *sound* with respect to $\mathcal{H}$. That is, the use of the derived rule does not increase the set of provable theorems in $\mathcal{H}$. That is, it should be possible to prove a derived rule of interest, without using itself.

Deduction Rule

Definition 2 (3.12)

Let U be a set of formulas, and A a formula. The notation $U \vdash A$ means that the formulas in U are *assumptions* in the proof of A. A *proof* is a sequence of lines $U_i \vdash \phi_i$, such that for each i, $U_i \subseteq U$, and ϕ_i is an axiom, a previously proved theorem, a member of U_i or can be derived by MP from previous lines $U_i' \vdash \phi_i'$, $U_i'' \vdash \phi_i''$, where $i', i'' < i$.

Definition 3 (3.13)

Deduction rule: suppose you want to prove $A \rightarrow B$. First, we assume A, that is, treat A as if it is an additional axiom, in addition to the given ones, U. Then prove $U \cup \{A\} \vdash B$. This conclusion discharges our initial assumption A. That is, we have now proved that $A \rightarrow B$. In other words, $\dfrac{U \cup \{A\} \vdash B}{U \vdash A \rightarrow B}$.

Soundness of the Deduction Rule in $\mathcal{H}$

Theorem 2 (3.14, Ben-Ari)

The deduction rule is a **sound** *derived rule.*

Proof.

We show, by induction on the length n of the proof of $U \cup A \vdash B$, how to obtain a proof of $U \vdash A \rightarrow B$ that does not use the deduction rule (i.e. show *soundness*).

For $n = 1$, B is proved in a single step. Consequently, B is either an element of $U \cup \{A\}$, an axiom in $\mathcal{H}$, or a previously proved theorem.

- If B is actually A, then $\vdash A \rightarrow A$ by Theorem 1, so naturally $U \vdash A \rightarrow A$.
- If $B \in U$ (i.e. B is a proven theorem), or B is an axiom, then $U \vdash B$. Then B is proved in a single application of MP as follows:

1.	$U \vdash B$	Axiom or Theorem
2.	$U \vdash B \rightarrow (A \rightarrow B)$	Axiom #1
3.	$U \vdash A \rightarrow B$	MP, 1, 2

Soundness of the Deduction Rule in $\mathcal{H}$

Proof.

If $n > 1$, the last step in the proof of $U \cup \{A\} \vdash B$ is either a one-step inference of B or an inference of B using MP.

In the first case, the result holds by the proof for n $= 1$.

Otherwise, MP was used, so there is a formula C and lines $i, j < n$ in the proof such that line i in the proof is $U \cup \{A\} \vdash C$ and line j is $U \cup \{A\} \vdash C \rightarrow B$. By the inductive hypothesis, $U \vdash A \rightarrow C$ and $U \vdash A \rightarrow (C \rightarrow B)$. Based on these, the proof of $U \vdash A \rightarrow B$ is given by:

1.	$U \vdash A \rightarrow C$	Inductive Hypothesis
2.	$U \vdash A \rightarrow (C \rightarrow B)$	Inductive Hypothesis
3.	$U \vdash (A \rightarrow (C \rightarrow B)) \rightarrow ((A \rightarrow C) \rightarrow (A \rightarrow B))$	Axiom #2
4.	$U \vdash (A \rightarrow C) \rightarrow (A \rightarrow B)$	MP, 2, 3
5.	$U \vdash A \rightarrow B$	MP, 1, 4

Derived Rules in $\mathcal{H}$

Theorem 3 (3.16, Ben-Ari)

$$\vdash (A \to B) \to ((B \to C) \to (A \to C))$$

Proof.

1.	$\{A \to B, B \to C, A\} \vdash A$	Assumption
2.	$\{A \to B, B \to C, A\} \vdash A \to B$	Assumption
3.	$\{A \to B, B \to C, A\} \vdash B$	MP, 1, 2
4.	$\{A \to B, B \to C, A\} \vdash B \to C$	Assumption
5.	$\{A \to B, B \to C, A\} \vdash C$	MP, 3, 4
6.	$\{A \to B, B \to C\} \vdash A \to C$	Deduction, 5
7.	$\{A \to B\} \vdash ((B \to C) \to (A \to C))$	Deduction, 6
8.	$\vdash (A \to B) \to ((B \to C) \to (A \to C))$	Deduction, 7

□

Definition 4 (Rule of Transtivity)

$$\frac{U \vdash A \to B \quad U \vdash B \to C}{U \vdash A \to C}$$

Derived Rules in $\mathcal{H}$

Theorem 4

$$\vdash (A \to (B \to C)) \to (B \to (A \to C))$$

Proof.

1.	$\{A \to (B \to C), B, A\} \vdash A$	Assumption
2.	$\{A \to (B \to C), B, A\} \vdash A \to (B \to C)$	Assumption
3.	$\{A \to (B \to C), B, A\} \vdash B \to C$	MP, 1, 2
4.	$\{A \to (B \to C), B, A\} \vdash B$	Assumption
5.	$\{A \to (B \to C), B, A\} \vdash C$	MP, 4, 3
6.	$\{A \to (B \to C), B\} \vdash A \to C$	Deduction, 5
7.	$\{A \to (B \to C)\} \vdash B \to (A \to C)$	Deduction, 6
8.	$\vdash (A \to (B \to C)) \to (B \to (A \to C))$	Deduction, 7

□

Definition 5 (Rule of Exchanged Antecedent)

$$\frac{A \to (B \to C)}{U \vdash B \to A \to C}$$

Theorems for other operators in $\mathcal{H}$

Theorem 5

$$\vdash A \rightarrow (B \rightarrow (A \wedge B))$$

Proof.

1.	$\{A, B\} \vdash (A \rightarrow \neg B) \rightarrow (A \rightarrow \neg B)$	Theorem 1
2.	$\{A, B\} \vdash A \rightarrow ((A \rightarrow \neg B) \rightarrow \neg B)$	Exchange of Antecedent
3.	$\{A, B\} \vdash A$	Assumption
4.	$\{A, B\} \vdash (A \rightarrow \neg B) \rightarrow \neg B$	MP, 3, 2
5.	$\{A, B\} \vdash \neg\neg B \rightarrow \neg(A \rightarrow \neg B)$	Contrapositive
6.	$\{A, B\} \vdash B$	Assumption
7.	$\{A, B\} \vdash \neg\neg B$	Double Negation
8.	$\{A, B\} \vdash \neg(A \rightarrow \neg B)$	MP, 5, 7
9.	$\{A\} \vdash B \rightarrow \neg(A \rightarrow \neg B)$	Deduction
10.	$\vdash A \rightarrow (B \rightarrow \neg(A \rightarrow \neg B))$	Deduction
11.	$\vdash A \rightarrow (B \rightarrow (A \wedge B))$	$A \wedge B \models \neg(A \rightarrow \neg B)$

Exercise

Prove the following theorems in $\mathcal{H}$:

- $\vdash (A \rightarrow B) \rightarrow (\neg B \rightarrow \neg A)$ (Contraposition)
- $\vdash \neg\neg A \rightarrow A$ (Double Negation)
- given that $\vdash \mathit{true}$ and $\vdash \neg \mathit{false}$, prove $\vdash (\neg A \rightarrow \mathit{false}) \rightarrow A$ (Reductio Ad Absurdum)

Propositional Logic: Soundness and Completeness of $\mathcal{H}$

CS402, Spring 2017

Shin Yoo

Goals of Logic

- To check whether a given formula ϕ is valid (semantic)
- To prove a given formula ϕ (syntactic)

$\models \phi$	$\equiv$	Semantic	$\equiv$	$\vdash \phi$
Semantic		Tableau		Syntactic
Methods				Methods
(Truth table)				($\mathcal{G}$, $\mathcal{H}$, etc)

Soundness and Completeness of $\mathcal{H}$

We can use what we have proved so far.

$\vdash \phi$	$\equiv$	$\mathcal{G}$	$\equiv$	Semantic	$\equiv$	$\models \phi$
Hilbert System, $\mathcal{H}$				Tableau		Semantic Methods (Truth table)

Soundness of $\mathcal{H}$

Theorem 1 (3.34, Ben-Ari)

$\mathcal{H}$ is sound, that is, if $\vdash A$ in $\mathcal{H}$ then $\models A$.

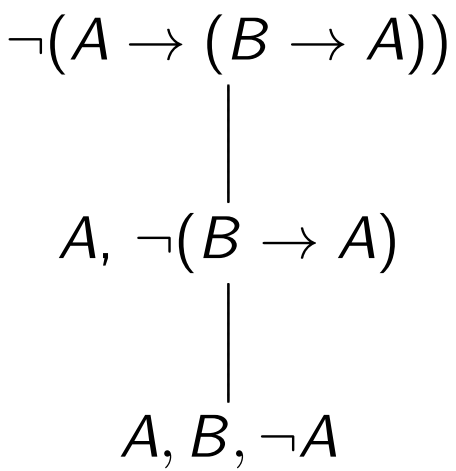
How do we prove this? Structural Induction, that is:

1. Show that three axioms of $\mathcal{H}$ are all valid, and
2. Show that if the premises of Modus Ponens rule is valid, then so is the conclusion.

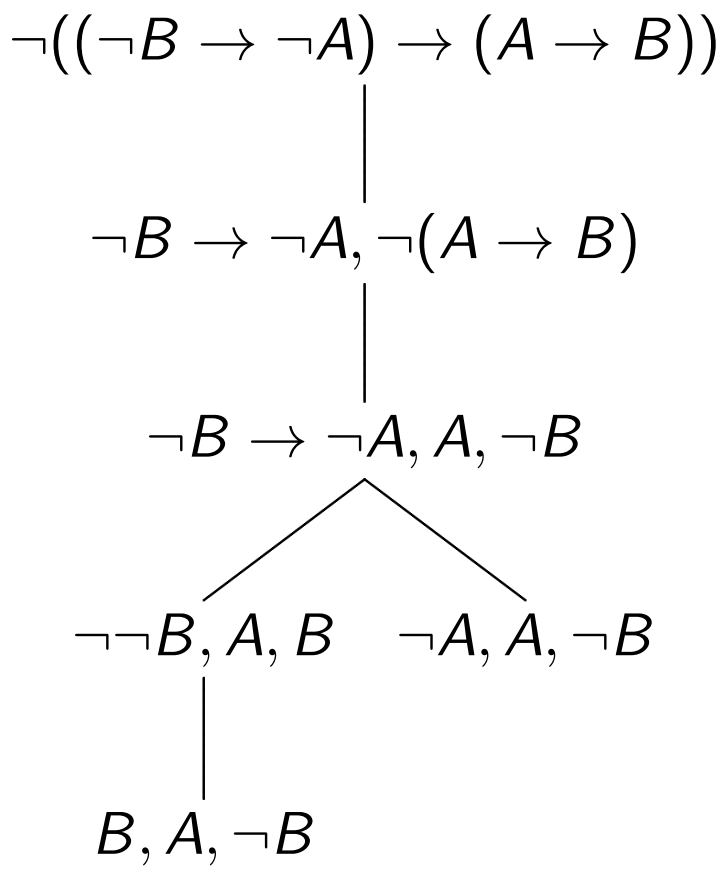
Soundness of $\mathcal{H}$

Show that three axioms of $\mathcal{H}$ are all valid. To show that A is valid, we can show $\neg A$ is not satisfiable, i.e., that the semantic tableau of $\neg A$ is *closed*.

Axiom 1



Axiom 3



Soundness of $\mathcal{H}$

Tableau for Axiom 2:
$\vdash (A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$?

Soundness of $\mathcal{H}$

Show that if MP is sound. We use Reductio Ad Absurdum.

Proof.

Suppose that MP is not sound. There would be a set of formulas $\{A, A \rightarrow B, B\}$ such that A and $A \rightarrow B$ are valid but B is not.If B is not valid, there is an interpretation ν such that $\nu(B) = F$. Since A and $A \rightarrow B$ are valid, for **any** interpretation, **including** ν, resulting in $\nu(A) = \nu(A \rightarrow B) = T$. The truth table then states $\nu(B) = T$, which results in a contradiction in terms of our choice of ν. □

$\mathcal{H}$ is sound.

Completeness of $\mathcal{H}$

Theorem 2 (3.35, Ben-Ari)

$\mathcal{H}$ is complete, that is, if $\models A$ then $\vdash A$ in $\mathcal{H}$.

- Any valid formula can be proved in $\mathcal{G}$ (Thm 3.8). We will show how a proof in $\mathcal{G}$ can be mechanically transformed into a proof in $\mathcal{H}$.
- A set of formulas U is provable in $\mathcal{G} \equiv$ a single formula $\bigvee U$ is provable in $\mathcal{H}$.
- Certain axioms of $\mathcal{G}$ are trivial: $\{p, \neg p\}$ is an axiom in $\mathcal{G}$, then $\vdash (p \vee \neg p)$ in $\mathcal{H}$, using Thm 3.10 ($\vdash A \rightarrow A$). (Note that $A \vee B \leftrightarrow \neg A \vee B$)
- How about $\{q, \neg p, r, p, s\}$? Not trivial.

Completeness of $\mathcal{H}$

Lemma 1 (3.36, Ben-Ari)

If $U' \subseteq U$ *and* $\vdash \bigvee U'$, *then* $\vdash \bigvee U$ *in* $\mathcal{H}$.

Proof.

Suppose we have a proof of $\bigvee U'$. By repeated application of Thm 3.31, we can transform this into a proof of $\bigvee U''$, where U'' is a permutation of the elements of U (Thm 3.31 Weakening: $\vdash A \rightarrow A \vee B$). Now by repeated applications of the commutativity and associativity of disjunction, we can move the elements of U'' to their proper places (Thm 3.32 is the Commutativity Rule, Thm 3.33 is the Associativity Rule). □

Completeness of $\mathcal{H}$

Now we use the induction on the structure of proofs in $\mathcal{G}$ in order to prove the completeness of $\mathcal{H}$. That is, we show that for any proof in $\mathcal{G}$, there exists a mechanically corresponding proof in $\mathcal{H}$.

Proof.

Case 1: If U is an **axiom**, it contains a pair of complementary literals, and $\vdash p \vee \neg p$ is provable in $\mathcal{H}$. By Lemma 1, this can be transformed into a proof of $\bigvee U$.

Lem 3.36: If $U' \subseteq U$ and $\vdash \bigvee U'$, then $\vdash \bigvee U$ in $\mathcal{H}$.

Completeness of $\mathcal{H}$

Proof. Cont.

Case 2: If U is not an axiom in $\mathcal{G}$, the last step in the proof of G is the application of either $\alpha-$ or $\beta-$ rule.

$$\frac{\vdash U_1 \cup \{A_1, A_2\}}{\vdash U_1 \cup \{A_1 \vee A_2\}}$$

- $\alpha-$rule: $\vdash U_1 \cup \{A_1 \vee A_2\}$. By the inductive hypothesis, $\vdash (\bigvee U_1 \vee A_1) \vee A_2$ in $\mathcal{H}$, from which we get $\vdash \bigvee U_1 \vee (A_1 \vee A_2)$ by associativity.

Completeness of $\mathcal{H}$

Proof.

- $\beta-$rule: $\dfrac{\vdash U_1 \cup \{A_1\} \quad \vdash U_2 \cup \{A_2\}}{\vdash U_1 \cup U_2 \cup \{A_1 \wedge A_2\}}$. By the inductive hypothesis, $\vdash \bigvee U_1 \vee A_1$ and $\vdash \bigvee U_2 \vee A_2$ in $\mathcal{H}$. From these, we need to find a proof of $\vdash \bigvee U_1 \vee \bigvee U_2 \vee (A_1 \wedge A_2)$.

1.	$\vdash \bigvee U_1 \vee A_1$	Induction Hypothesis
2.	$\vdash \neg \bigvee U_1 \rightarrow A_1$	$A \vee B \models \neg A \rightarrow B$
3.	$\vdash A_1 \rightarrow (A_2 \rightarrow (A_1 \wedge A_2))$	Derived rule on $\wedge$
4.	$\vdash \neg \bigvee U_1 \rightarrow (A_2 \rightarrow (A_1 \wedge A_2))$	MP, 2, 3
5.	$\vdash A_2 \rightarrow (\neg \bigvee U_1 \rightarrow (A_1 \wedge A_2))$	Exchanged Antecedents
6.	$\vdash \bigvee U_2 \vee A_2$	Induction Hypothesis
7.	$\vdash \neg \bigvee U_2 \rightarrow A_2$	$A \vee B \models \neg A \rightarrow B$
8.	$\vdash \neg \bigvee U_2 \rightarrow (\neg \bigvee U_1 \rightarrow (A_1 \wedge A_2))$	MP, 7, 5
9.	$\vdash \neg \bigvee U_2 \rightarrow (\bigvee U_1 \vee (A_1 \wedge A_2))$	$A \vee B \models \neg A \rightarrow B$
10.	$\vdash \bigvee U_2 \vee (\bigvee U_1 \vee (A_1 \wedge A_2))$	$A \vee B \models \neg A \rightarrow B$
11.	$\vdash \bigvee U_1 \vee \bigvee U_2 \vee (A_1 \wedge A_2))$	Associativity

Consistency of $\mathcal{H}$

Definition 1 (3.38, Ben-Ari)

A set of formulas U is inconsistent iff for some formula A, $U \vdash A$ and $U \vdash \neg A$. U is consistent iff U is not inconsistent.

Theorem 3 (3.39, Ben-Ari)

U is inconsistent iff for all A, $U \vdash A$.

Proof.

Let A be an arbitrary formula. Since U is inconsistent, for some formula B, $U \vdash B$ and $U \vdash \neg B$. By Thm 3.21, $\vdash B \rightarrow (\neg B \rightarrow A)$. Using MP twice, $U \vdash A$. The converse is trivial. $\square$

Consistency of $\mathcal{H}$

Corollary 1 (3.40)

U is consistent iff for some A, $U \nvdash A$.

Theorem 4 (3.41)

$U \vdash A$ iff $U \cup \{\neg A\}$ is inconsistent.

Variants of $\mathcal{H}$

Variant Hilbert Systems almost always have MP as the single rule, while having different choice of primitive operators and axioms. For example, a variant $\mathcal{H}$' replaces the third axiom with:

$$\text{Axiom 3': } \vdash (\neg B \rightarrow \neg A) \rightarrow ((\neg B \rightarrow A) \rightarrow B)$$

Theorem 5

$\mathcal{H}$ and $\mathcal{H}$' are equivalent.

Variants of $\mathcal{H}$

Proof of Axiom 3' in $\mathcal{H}$:

1.	$\{\neg B \to \neg A, \neg B \to A, \neg B\} \vdash \neg B$	Assumption
2.	$\{\neg B \to \neg A, \neg B \to A, \neg B\} \vdash \neg B \to A$	Assumption
3.	$\{\neg B \to \neg A, \neg B \to A, \neg B\} \vdash A$	MP, 1, 2
4.	$\{\neg B \to \neg A, \neg B \to A, \neg B\} \vdash \neg B \to \neg A$	Assumption
5.	$\{\neg B \to \neg A, \neg B \to A, \neg B\} \vdash A \to B$	Contrapositive, 4
6.	$\{\neg B \to \neg A, \neg B \to A, \neg B\} \vdash B$	MP, 3, 5
7.	$\{\neg B \to \neg A, \neg B \to A\} \vdash \neg B \to B$	Deduction, 7
8.	$\{\neg B \to \neg A, \neg B \to A\} \vdash (\neg B \to B) \to B$	Theorem 3.29
9.	$\{\neg B \to \neg A, \neg B \to A\} \vdash B$	MP, 8, 9
10.	$\{\neg B \to \neg A\} \vdash (\neg B \to A) \to B$	Deduction, 9
11.	$\vdash (\neg B \to \neg A) \to (\neg B \to A) \to B$	Deduction, 10

Variants of $\mathcal{H}$

$\mathcal{H}$" has the same MP rule, but a different set of axioms:

1. $\vdash A \vee A \rightarrow A$
2. $\vdash A \rightarrow A \vee B$
3. $\vdash A \vee B \rightarrow B \vee A$
4. $\vdash B \rightarrow C \rightarrow (A \vee B \rightarrow A \vee C)$

$\mathcal{H}$"' has only one axiom called Meredith's Axiom:

$$(\{[(A \rightarrow B) \rightarrow (\neg C \rightarrow \neg D)] \rightarrow C\} \rightarrow E) \rightarrow [(E \rightarrow A) \rightarrow (D \rightarrow A)]$$

Arrrrgh.

Subformula Property

Definition 2 (3.48, Ben-Ari)

A deductive system has the **subformula property** if any formula appearing in a proof of A is either a subformula of A or the negation of a subformula of A.

$\mathcal{G}$ has the subformula property; $\mathcal{H}$ does not. Why?

If a deductive system has the subformula property, then mechanical proof may become possible, since

- there exist only a finite number of subformulas for a finite formula ϕ
- there exist only a finite number of inference rules

The rest is the machine's work.

Automated Proof

One desirable property of a deductive system is to generate an automated/mechanical proof.

- We have decision procedures to check validity of a propositional formula automatically (i.e., truth table and semantic tableau).
- Note that decision procedures require knowledge on all interpretations (i.e., infinite number of interpretations, in general) which is not feasible except for propositional logic.

A deductive proof requires only a finite set of sets of formulas, because a deductive proof system analyses the target formula only, not its interpretations.

- Many research works to develop (semi)automated theorem prover.
- No obvious connection between the formula and its proof in $\mathcal{H}$; makes a proof in $\mathcal{H}$ difficult (no mechanical proof).
- One has to rely on one's brain to select proper axioms.