Warm Up Problem

What is the definition of a truth valuation? How did we extend this to well-formed formulas?

Propositional Logic: Semantics

Carmen Bruni

With thanks to A. Gao for these slides!

Based on slides by Jonathan Buss, Lila Kari, Anna Lubiw and Steve Wolfman with thanks to B. Bonakdarpour, A. Gao, D. Maftuleac, C. Roberts, R. Trefler, and P. Van Beek

Lecture 4

Learning goals

By the end of this lecture, you should be able to

- Evaluate the truth value of a formula
  - Define a (truth) valuation.
  - Determine the truth value of a formula by using truth tables.
  - Determine the truth value of a formula by using valuation trees.
- Determine and prove whether a formula has a particular property
  - Define tautology, contradiction, and satisfiable formula.
  - Compare and contrast the three properties (tautology, contradiction, and satisfiable formula).
  - Prove whether a formula is a tautology, a contradiction, or satisfiable, using a truth table and/or a valuation tree.
  - Describe strategies to prove whether a formula is a tautology, a contradiction or a satisfiable formula.

The meaning of well-formed formulas

To interpret a formula, we have to give meanings to the propositional variables and the connectives.

A propositional variable has no intrinsic meaning; it gets a meaning via a valuation.

A (truth) valuation is a function $t : \mathcal{P} \mapsto \{\mathtt{F}, \mathtt{T}\}$ from the set of all proposition variables $\mathcal{P}$ to $\{\mathtt{F}, \mathtt{T}\}$. It assigns true/false to every propositional variable.

Two notations: $t(p)$ and p^t both denote the truth value of p under the truth valuation t.

Truth value of a formula

Fix a truth valuation t. Every formula α has a value under t, denoted α^t, determined as follows.

1. $p^t = t(p)$.
2. $(\neg\alpha)^t = \begin{cases} \mathtt{T} & \text{if } \alpha^t = \mathtt{F} \\ \mathtt{F} & \text{if } \alpha^t = \mathtt{T} \end{cases}$
3. $(\alpha \wedge \beta)^t = \begin{cases} \mathtt{T} & \text{if } \alpha^t = \beta^t = \mathtt{T} \\ \mathtt{F} & \text{otherwise} \end{cases}$
4. $(\alpha \vee \beta)^t = \begin{cases} \mathtt{T} & \text{if } \alpha^t = \mathtt{T} \text{ or } \beta^t = \mathtt{T} \\ \mathtt{F} & \text{otherwise} \end{cases}$
5. $(\alpha \to \beta)^t = \begin{cases} \mathtt{T} & \text{if } \alpha^t = \mathtt{F} \text{ or } \beta^t = \mathtt{T} \\ \mathtt{F} & \text{otherwise} \end{cases}$
6. $(\alpha \leftrightarrow \beta)^t = \begin{cases} \mathtt{T} & \text{if } \alpha^t = \beta^t \\ \mathtt{F} & \text{otherwise} \end{cases}$

Truth tables for connectives

The unary connective $\neg$:

α	$(\neg\alpha)$
T	F
F	T

The binary connectives $\wedge$, $\vee$, $\rightarrow$, and $\leftrightarrow$:

α	β	$(\alpha \wedge \beta)$	$(\alpha \vee \beta)$	$(\alpha \rightarrow \beta)$	$(\alpha \leftrightarrow \beta)$
F	F	F	F	T	T
F	T	F	T	T	F
T	F	F	T	F	F
T	T	T	T	T	T

Evaluating a formula using a truth table

Example. The truth table of $((p \vee q) \rightarrow (q \wedge r))$.

Evaluating a formula using a truth table

Example. The truth table of $((p \vee q) \rightarrow (q \wedge r))$.

p	q	r	$(p \vee q)$	$(q \wedge r)$	$((p \vee q) \rightarrow (q \wedge r))$
F	F	F	F	F	T
F	F	T	F	F	T
F	T	F	T	F	F
F	T	T	T	T	T
T	F	F	T	F	F
T	F	T	T	F	F
T	T	F	T	F	F
T	T	T	T	T	T

Evaluating a formula using a truth table

Build the truth table of $\big((p \to (\neg q)) \to (q \vee (\neg p))\big)$.

Understanding the disjunction and the biconditional

α	β	$(\alpha \vee \beta)$	Exclusive OR	Biconditional
F	F	F	F	T
F	T	T	T	F
T	F	T	T	F
T	T	T	F	T

- What is the difference between an inclusive OR (the disjunction) and an exclusive OR?
- What is the relationship between the exclusive OR and the biconditional?

A Small Theorem

Theorem: Fix a truth valuation t. Every formula α has a value α^t in $\{\mathtt{F}, \mathtt{T}\}$.

A Small Theorem

Theorem: Fix a truth valuation t. Every formula α has a value α^t in $\{\mathtt{F}, \mathtt{T}\}$.

Proof: By structural induction. Let $R(\alpha)$ be "α has a value α^t in $\{\mathtt{F}, \mathtt{T}\}$".

A Small Theorem

Theorem: Fix a truth valuation t. Every formula α has a value α^t in $\{\mathtt{F}, \mathtt{T}\}$.

Proof: By structural induction. Let $R(\alpha)$ be "α has a value α^t in $\{\mathtt{F}, \mathtt{T}\}$".

1. If α is a propositional variable, then t assigns it a value of T or F (by the definition of a truth valuation).
2. If α has a value in $\{\mathtt{F}, \mathtt{T}\}$, then $(\neg\alpha)$ also does, as shown by the truth table of $(\neg\alpha)$.
3. If α and β each has a value in $\{\mathtt{F}, \mathtt{T}\}$, then $(\alpha \star \beta)$ also does for every binary connective $\star$, as shown by the corresponding truth tables.

By the principle of structural induction, every formula has a value.

By the unique readability of formulas, we have proved that a formula has **only one** truth value under any truth valuation t. QED

Tautology, Contradiction, Satisfiable

A formula α is a *tautology* if and only if
for every truth valuation t, $\alpha^t = \mathtt{T}$.

A formula α is a *contradiction* if and only if
for every truth valuation t, $\alpha^t = \mathtt{F}$.

A formula α is *satisfiable* if and only if
there exists a truth valuation t such that $\alpha^t = \mathtt{T}$.

Examples

Let p be a Propositional variable.

- The formula $(p \vee (\neg p))$ is a tautology (and satisfiable)
- The formula $(p \wedge (\neg p))$ is a contradiction (and not satisfiable)
- The formula $(p \to (\neg p))$ is satisfiable (but not a tautology)

How to determine the properties of a formula

- Truth table
- Valuation tree
- Reasoning

Valuation Tree

Rather than fill out an entire truth table, we can analyze what happens if we plug in a truth value for one variable.

$\neg T$	F
$\neg F$	T

$(p \wedge T)$	p
$(p \wedge F)$	F
$(p \wedge p)$	p

$(p \vee T)$	T
$(p \vee F)$	p
$(p \vee p)$	p

$(p \rightarrow T)$	T
$(p \rightarrow F)$	$(\neg p)$
$(T \rightarrow p)$	p
$(F \rightarrow p)$	T
$(p \rightarrow p)$	T

We can evaluate a formula by using these rules to construct a *valuation tree*.

Example of a valuation tree

Example. Show that $\Big(\big(((p \wedge q) \to (\neg r)) \wedge (p \to q)\big) \to (p \to (\neg r))\Big)$ is a tautology by using a valuation tree.

Example of a valuation tree

Example. Show that $\Big(\big(((p \wedge q) \to (\neg r)) \wedge (p \to q)\big) \to (p \to (\neg r))\Big)$ is a tautology by using a valuation tree.

Suppose $t(p) = \mathtt{T}$. We put $\mathtt{T}$ in for p:

$$\big(((\mathtt{T} \wedge q) \to (\neg r)) \wedge (\mathtt{T} \to q)\big) \to (\mathtt{T} \to (\neg r)) \ .$$

Based on the truth tables for the connectives, the formula becomes $\Big(((q \to (\neg r)) \wedge q) \to (\neg r)\Big)$.

If $t(q) = \mathtt{T}$, this yields $((\neg r) \to (\neg r))$ and then $\mathtt{T}$. (Check!).
If $t(q) = \mathtt{F}$, it yields $(\mathtt{F} \to (\neg r))$ and then $\mathtt{T}$. (Check!).

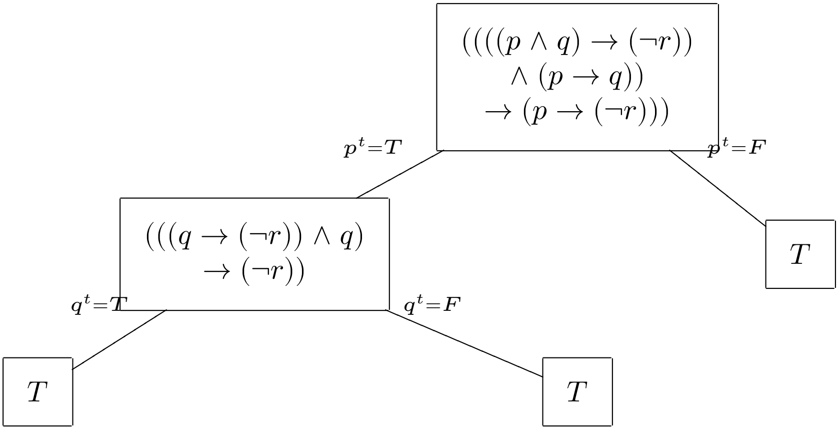
Suppose $t(p) = \mathtt{F}$. We get

$$\big(((\mathtt{F} \wedge q) \to (\neg r)) \wedge (\mathtt{F} \to q)\big) \to (\mathtt{F} \to (\neg r)) \ ,$$

Simplification yields $\big((\mathtt{F} \to (\neg r)) \wedge \mathtt{T}\big) \to \mathtt{T}$ and eventually $\mathtt{T}$.

Thus every valuation makes the formula true, as required.

Valuation Tree (Do not break on multiple lines!)



Determine which of the following are satisfiable, form a tautology or form a contradiction. Use a truth table. Repeat using a valuation tree.

1. $\Big((((p \vee q) \leftrightarrow (\neg r)) \wedge (p \to q)\big) \wedge (p \to (\neg r))\Big)$
2. $\Big((((p \wedge q) \to r) \wedge (p \to q)\big) \to (p \to r)\Big)$
3. $((p \vee q) \leftrightarrow ((p \wedge (\neg q)) \vee ((\neg p) \to q)))$
4. $(p \wedge (\neg p))$

Definition of logical equivalence

Two formulas α and β are logically equivalent if and only if they have the same value under any valuation.

- $\alpha^t = \beta^t$, for every valuation t.
- α and β must have the same final column in their truth tables.
- $(\alpha \leftrightarrow \beta)$ is a tautology.

Why do we care about logical equivalence?

- Will I lose marks if I provide a solution that is syntactically different but logically equivalent to the provided solution?
- Do these two circuits behave the same way?
- Do these two pieces of code fragments behave the same way?

You already know one way of proving logical equivalent. What is it?

Theorem: $(((\neg p) \wedge q) \vee p) \equiv (p \vee q)$.

Logical Identities

Commutativity

$$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha)$$
$$(\alpha \vee \beta) \equiv (\beta \vee \alpha)$$

Associativity

$$\big(\alpha \wedge (\beta \wedge \gamma)\big) \equiv \big((\alpha \wedge \beta) \wedge \gamma\big)$$
$$\big(\alpha \vee (\beta \vee \gamma)\big) \equiv \big((\alpha \vee \beta) \vee \gamma\big)$$

Distributivity

$$\big(\alpha \vee (\beta \wedge \gamma)\big) \equiv \big((\alpha \vee \beta) \wedge (\alpha \vee \gamma)\big)$$
$$\big(\alpha \wedge (\beta \vee \gamma)\big) \equiv \big((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)\big)$$

Idempotence

$$(\alpha \vee \alpha) \equiv \alpha$$
$$(\alpha \wedge \alpha) \equiv \alpha$$

Double Negation

$$\big(\neg(\neg\alpha)\big) \equiv \alpha$$

De Morgan's Laws

$$\big(\neg(\alpha \wedge \beta)\big) \equiv \big((\neg\alpha) \vee (\neg\beta)\big)$$
$$\big(\neg(\alpha \vee \beta)\big) \equiv \big((\neg\alpha) \wedge (\neg\beta)\big)$$

Logical Identities, cont'd

Simplification I (Absorption)

$$(\alpha \wedge \texttt{T}) \equiv \alpha$$
$$(\alpha \vee \texttt{T}) \equiv \texttt{T}$$
$$(\alpha \wedge \texttt{F}) \equiv \texttt{F}$$
$$(\alpha \vee \texttt{F}) \equiv \alpha$$

Simplification II

$$(\alpha \vee (\alpha \wedge \beta)) \equiv \alpha$$
$$(\alpha \wedge (\alpha \vee \beta)) \equiv \alpha$$

Implication

$$(\alpha \to \beta) \equiv ((\neg\alpha) \vee \beta)$$

Contrapositive

$$(\alpha \to \beta) \equiv ((\neg\beta) \to (\neg\alpha))$$

Equivalence

$$(\alpha \leftrightarrow \beta) \equiv ((\alpha \to \beta) \wedge (\beta \to \alpha))$$

Excluded Middle

$$(\alpha \vee (\neg\alpha)) \equiv \texttt{T}$$

Contradiction

$$(\alpha \wedge (\neg\alpha)) \equiv \texttt{F}$$

A logical equivalence proof

Theorem: $(((\neg p) \wedge q) \vee p) \equiv (p \vee q)$.

A logical equivalence proof

Theorem: $(((\neg p) \wedge q) \vee p) \equiv (p \vee q)$.

Proof.

$(((\neg p) \wedge q) \vee p)$	
$\equiv (((\neg p) \vee p) \wedge (q \vee p))$	Distributivity
$\equiv (\mathtt{T} \wedge (q \vee p))$	Excluded Middle
$\equiv (q \vee p)$	Simplification I
$\equiv (p \vee q)$	Commutativity

□

A practice problem

"If it is sunny, I will play golf, provided that I am relaxed."

s: it is sunny. g: I will play golf. r: I am relaxed.

A few translations:

1. $(s \to (r \to g))$
2. $(r \to (s \to g))$
3. $((s \land r) \to g)$

Theorem: All three translations are logically equivalent.
Proof: Done in class.

Collected Wisdom

- Try getting rid of $\rightarrow$ and $\leftrightarrow$.
- Try moving negations inward. $(\neg(p \vee q)) \equiv ((\neg p) \wedge (\neg q))$.
- Work from the more complex side first, BUT
- Switch to different strategies/sides when you get stuck.
- In the end, write the proof in clean ' one-side-to-the-other' form and double-check steps.

```
if ( (input > 0) or (not output) ) {
    if ( not (output and (queuelength < 100) ) ) {
        P1
    } else if ( output and (not (queuelength < 100)) ) {
        P2
    } else { P3 }
} else { P4 }
```

When does each piece of code get executed?

Let i: `input > 0`,
u: `output`,
q: `queuelength < 100`.

A Code Example, cont'd

i	u	q	$(i \vee (\neg u))$	$(\neg(u \wedge q))$	$(u \wedge (\neg q))$	
T	T	T	T			
T	T	F	T			
T	F	T	T			
T	F	F	T			
F	T	T	F			P_4
F	T	F	F			P_4
F	F	T	T			
F	F	F	T			

A Code Example, cont'd

i	u	q	$(i \vee (\neg u))$	$(\neg(u \wedge q))$	$(u \wedge (\neg q))$	
T	T	T	T	F	F	P_3
T	T	F	T	T		P_1
T	F	T	T	T		P_1
T	F	F	T	T		P_1
F	T	T	F			P_4
F	T	F	F			P_4
F	F	T	T	T		P_1
F	F	F	T	T		P_1

Finding Live Code

Prove that P_3 is live code. That is, the conditions leading to P_3 is satisfiable.

Theorem:
$$\Big((i \vee (\neg u)) \wedge \Big((\neg(\neg(u \wedge q))) \wedge \big(\neg(u \wedge (\neg q))\big)\Big)\Big) \equiv ((i \wedge u) \wedge q)$$

Proof: In class

Two pieces of code: Are they equivalent?

Prove that the two code fragments are equivalent.

Listing 1: Your code

```
if (i || !u) {
  if (!(u && q)) {
    P1
  } else if (u && !q) {
    P2
  } else { P3 }
} else { P4 }
```

Listing 2: Your friend's code

```
if ((i && u) && q) {
  P3
} else if (!i && u) {
  P4
} else {
  P1
}
```

Simplifying Code

To prove that the two fragments are equivalent, show that each block of code P_1, P_2, P_3, and P_4 is executed under equivalent conditions.

Block	Fragment 1	Fragment 2
P_1	$(i \lor (\neg u)) \land (\neg(u \land q))$	$(\neg(i \land u \land q)) \land (\neg((\neg i) \land u))$
P_2	$(i \lor (\neg u)) \land (\neg(\neg(u \land q))) \land (u \land (\neg q))$	$\mathtt{F}$
P_3	$(i \lor (\neg u)) \land (\neg(\neg(u \land q))) \land (\neg(u \land (\neg q)))$	$(i \land u \land q)$
P_4	$(\neg(i \lor (\neg u)))$	$(\neg(i \land u \land q)) \land ((\neg i) \land u)$