Introduction to Logic

Carlos Areces

carlos.areces@gmail.com

INRIA Nancy Grand Est
Nancy, France

2010 - Bloomington - US

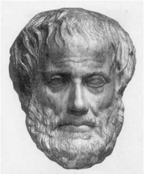
What is Logic? a Personal Perspective

What is Logic? a Personal Perspective

- What is Logic?

What is Logic? a Personal Perspective

- What is Logic?
  - A tool to model how we think ???



What is Logic? a Personal Perspective

- What is Logic?
  - A tool to model how we think ???
  - The fundamental basis of Mathematics ???

What is Logic? a Personal Perspective

- What is Logic?
  - A tool to model how we think ???
  - The fundamental basis of Mathematics ???
  - The origins of Computer Science ???

What is Logic? a Personal Perspective

- What is Logic?
  - A tool to model how we think ???
  - The fundamental basis of Mathematics ???
  - The origins of Computer Science ???
- Perhaps we are asking the wrong question. Let's try with this one:

  What is Logic used for?

What is Logic? a Personal Perspective

- What is Logic?
  - A tool to model how we think ???
  - The fundamental basis of Mathematics ???
  - The origins of Computer Science ???
- Perhaps we are asking the wrong question. Let's try with this one:

  What is Logic used for?

- I believe that a fair and acurate answer is:

  Logics are used to describe.

Next question then: To describe what?

Next question then: To describe what?

- To describe what?

Next question then: To describe what?

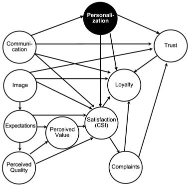
- To describe what? Anything!!!!

Next question then: To describe what?

- To describe what? **Anything!!!!**
- Agreed: With more or less detail.
  But we can use logics to describe (almost?) anything we can dream of.

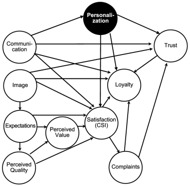
Next question then: To describe what?

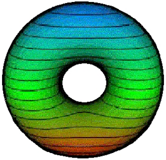
- To describe what? Anything!!!!
- Agreed: With more or less detail.
  But we can use logics to describe (almost?) anything we can dream of.



Next question then: To describe what?

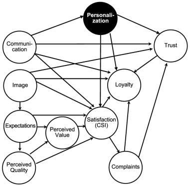
- To describe what? Anything!!!!
- Agreed: With more or less detail.
  But we can use logics to describe (almost?) anything we can dream of.

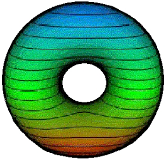


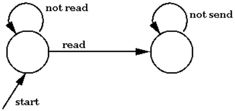


Next question then: To describe what?

- To describe what? Anything!!!!
- Agreed: With more or less detail.
  But we can use logics to describe (almost?) anything we can dream of.

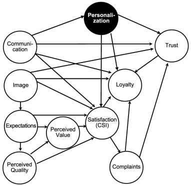


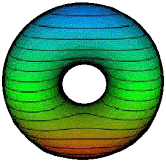


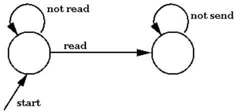


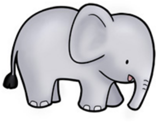
Next question then: To describe what?

- To describe what? Anything!!!!
- Agreed: With more or less detail.
  But we can use logics to describe (almost?) anything we can dream of.









Logics and Drawings

Logics and Drawings

- We say that

  Logics are used to describe anything!

Logics and Drawings

- We say that

  Logics are used to describe anything!

- But then, drawings, for example, are also used to describe things.

Logics and Drawings

- We say that

  Logics are used to describe anything!

- But then, drawings, for example, are also used to describe things.
- And they are, in many ways, much better than logics
  - Drawings can be, arguably, more beautiful than logics

Logics and Drawings

- We say that

  Logics are used to describe anything!

- But then, drawings, for example, are also used to describe things.
- And they are, in many ways, much better than logics
  - Drawings can be, arguably, more beautiful than logics
  - Drawings can be done by 5 year olds
  - ...

Logics and Drawings

- We say that

  Logics are used to describe anything!

- But then, drawings, for example, are also used to describe things.
- And they are, in many ways, much better than logics
  - Drawings can be, arguably, more beautiful than logics
  - Drawings can be done by 5 year olds
  - ...
- When/for what are logics better than drawing?

Querying

Querying

- There are at least one aspect in which logics are better than drawings.

We can ask questions to a logic
and, more interestingly, obtain answers

Querying

- There are at least one aspect in which logics are better than drawings.

  We can ask questions to a logic
  and, more interestingly, obtain answers

- Which are the questions that we can pose a Logic?

Querying

- There are at least one aspect in which logics are better than drawings.

We can ask questions to a logic
and, more interestingly, obtain answers

- Which are the questions that we can pose a Logic?
- The most common one is probably satisfiability:

Is φ true in some model?

Querying

- There are at least one aspect in which logics are better than drawings.

We can ask questions to a logic
and, more interestingly, obtain answers

- Which are the questions that we can pose a Logic?
- The most common one is probably satisfiability:

Is φ true in some model?

- Let's write $M \models \varphi$ (and we say "M satisfies φ") if the model M makes the formula φ true.

What can we say with 'satisfies'?

What can we say with 'satisfies'?

- We say then that a formula φ is satisfiable if there is a model M such that $M \models \varphi$

What can we say with 'satisfies'?

- We say then that a formula φ is satisfiable if there is a model M such that $M \models \varphi$
- We say that a formula φ is unsatisfiable (a contradiction) if there is no model M such that $M \models \varphi$

What can we say with 'satisfies'?

- We say then that a formula φ is satisfiable if there is a model M such that $M \models \varphi$
- We say that a formula φ is unsatisfiable (a contradiction) if there is no model M such that $M \models \varphi$
- We say that a formula φ is valid if for all models M, $M \models \varphi$

What can we say with 'satisfies'?

- We say then that a formula φ is satisfiable if there is a model M such that $M \models \varphi$
- We say that a formula φ is unsatisfiable (a contradiction) if there is no model M such that $M \models \varphi$
- We say that a formula φ is valid if for all models M, $M \models \varphi$

A property of most logics: $M \models \varphi$ iff $M \not\models \neg\varphi$

What can we say with 'satisfies'?

- We say then that a formula φ is satisfiable if there is a model M such that $M \models \varphi$
- We say that a formula φ is unsatisfiable (a contradiction) if there is no model M such that $M \models \varphi$
- We say that a formula φ is valid if for all models M, $M \models \varphi$

A property of most logics: $M \models \varphi$ iff $M \not\models \neg\varphi$

Exercise: If φ is a valid, what is $\neg\varphi$?

What can we say with 'satisfies'?

- We say then that a formula φ is satisfiable if there is a model M such that $M \models \varphi$
- We say that a formula φ is unsatisfiable (a contradiction) if there is no model M such that $M \models \varphi$
- We say that a formula φ is valid if for all models M, $M \models \varphi$

A property of most logics: $M \models \varphi$ iff $M \not\models \neg\varphi$

Exercise: If φ is a valid, what is $\neg\varphi$? $\neg\varphi$ is a contradiction

What can we say with 'satisfies'?

- We say then that a formula φ is satisfiable if there is a model M such that $M \models \varphi$
- We say that a formula φ is unsatisfiable (a contradiction) if there is no model M such that $M \models \varphi$
- We say that a formula φ is valid if for all models M, $M \models \varphi$

A property of most logics: $M \models \varphi$ iff $M \not\models \neg\varphi$

Exercise: If φ is a valid, what is $\neg\varphi$? $\neg\varphi$ is a contradiction
If φ is satisfiable, what is $\neg\varphi$?

What can we say with 'satisfies'?

- We say then that a formula φ is satisfiable if there is a model M such that $M \models \varphi$
- We say that a formula φ is unsatisfiable (a contradiction) if there is no model M such that $M \models \varphi$
- We say that a formula φ is valid if for all models M, $M \models \varphi$

A property of most logics: $M \models \varphi$ iff $M \not\models \neg\varphi$

Exercise: If φ is a valid, what is $\neg\varphi$? $\neg\varphi$ is a contradiction
If φ is satisfiable, what is $\neg\varphi$? We don't know. It can be satisfiable or it can be a contradiction.

What can we say with 'satisfies'?

- We say then that a formula φ is satisfiable if there is a model M such that $M \models \varphi$
- We say that a formula φ is unsatisfiable (a contradiction) if there is no model M such that $M \models \varphi$
- We say that a formula φ is valid if for all models M, $M \models \varphi$

A property of most logics: $M \models \varphi$ iff $M \not\models \neg\varphi$

Exercise: If φ is a valid, what is $\neg\varphi$? $\neg\varphi$ is a contradiction

If φ is satisfiable, what is $\neg\varphi$? We don't know. It can be satisfiable or it can be a contradiction.

Can it be a validity?

What can we say with 'satisfies'?

- We say then that a formula φ is satisfiable if there is a model M such that $M \models \varphi$
- We say that a formula φ is unsatisfiable (a contradiction) if there is no model M such that $M \models \varphi$
- We say that a formula φ is valid if for all models M, $M \models \varphi$

A property of most logics: $M \models \varphi$ iff $M \not\models \neg\varphi$

Exercise: If φ is a valid, what is $\neg\varphi$? $\neg\varphi$ is a contradiction

If φ is satisfiable, what is $\neg\varphi$? We don't know. It can be satisfiable or it can be a contradiction.

Can it be a validity? No.

What is $\models$ anyway?

What is $\models$ anyway?

- $\models$ is a binary relation between models and formulas.

What is $\models$ anyway?

- $\models$ is a binary relation between models and formulas.
- To see it more clearly instead of writing $M \models \varphi$ we could write $(M, \varphi) \in \models$.

What is $\models$ anyway?

- $\models$ is a binary relation between models and formulas.
- To see it more clearly instead of writing $M \models \varphi$ we could write $(M, \varphi) \in \models$.

In most of the rest of this tutorial we will talk about this relation between models and formulas for one particular logic:

Propositional Logic

Propositional Logic

Propositional Logic

- A proposition is a sentence that can be either true or false.

Propositional Logic

- A proposition is a sentence that can be either true or false.
- Examples:
  - 3 is greater than 4.

Propositional Logic

- A proposition is a sentence that can be either true or false.
- Examples:
  - 3 is greater than 4.
  - Carlos was born in Argentina.

Propositional Logic

- A proposition is a sentence that can be either true or false.
- Examples:
  - 3 is greater than 4.
  - Carlos was born in Argentina.
  - It will rain tomorrow.

Propositional Logic

- A proposition is a sentence that can be either true or false.
- Examples:
  - 3 is greater than 4.
  - Carlos was born in Argentina.
  - It will rain tomorrow.
- Propositions are the smallest (atomic) formulas of propositional logics. They can be combined using logical operators into complex formulas.

Propositional Logics: Syntax

Propositional Logics: Syntax

- Logical Operators or Connectives:

$\neg$	$\neg p$	negation

Propositional Logics: Syntax

- Logical Operators or Connectives:

$\neg$	$\neg p$	negation
$\wedge$	$p \wedge q$	conjunction

Propositional Logics: Syntax

- Logical Operators or Connectives:

$\neg$	$\neg p$	negation
$\wedge$	$p \wedge q$	conjunction
$\vee$	$p \vee q$	disjunction

Propositional Logics: Syntax

- Logical Operators or Connectives:

$\neg$	$\neg p$	negation
$\wedge$	$p \wedge q$	conjunction
$\vee$	$p \vee q$	disjunction
$\rightarrow$	$p \rightarrow q$	implication

Propositional Logics: Syntax

- Logical Operators or Connectives:

$\neg$	$\neg p$	negation
$\wedge$	$p \wedge q$	conjunction
$\vee$	$p \vee q$	disjunction
$\rightarrow$	$p \rightarrow q$	implication
$\leftrightarrow$	$p \leftrightarrow q$	equivalency

Propositional Logics: Syntax

- Logical Operators or Connectives:

$\neg$	$\neg p$	negation
$\wedge$	$p \wedge q$	conjunction
$\vee$	$p \vee q$	disjunction
$\rightarrow$	$p \rightarrow q$	implication
$\leftrightarrow$	$p \leftrightarrow q$	equivalency
$\oplus$	$p \oplus q$	difference

Propositional Logics: Syntax

- Logical Operators or Connectives:

$\neg$	$\neg p$	negation
$\wedge$	$p \wedge q$	conjunction
$\vee$	$p \vee q$	disjunction
$\rightarrow$	$p \rightarrow q$	implication
$\leftrightarrow$	$p \leftrightarrow q$	equivalency
$\oplus$	$p \oplus q$	difference

- Do we need them all? How many of them are there?

Propositional Logics: Syntax

- Logical Operators or Connectives:

$\neg$	$\neg p$	negation
$\wedge$	$p \wedge q$	conjunction
$\vee$	$p \vee q$	disjunction
$\rightarrow$	$p \rightarrow q$	implication
$\leftrightarrow$	$p \leftrightarrow q$	equivalency
$\oplus$	$p \oplus q$	difference

- Do we need them all? How many of them are there? We'll answer that in a moment...

Propositional Logics: Syntax

Propositional Logics: Syntax

- We can use a grammar to define the set of well formed formulas FORM of propositional logic

Propositional Logics: Syntax

- We can use a grammar to define the set of well formed formulas FORM of propositional logic

$$\text{FORM} := A \mid (\neg\varphi) \mid (\varphi \star \varphi')$$

Propositional Logics: Syntax

- We can use a grammar to define the set of well formed formulas FORM of propositional logic

$$\text{FORM} := A \mid (\neg\varphi) \mid (\varphi \star \varphi')$$

  where A is a propositional symbol (i.e., $A \in \{p, q, r, s, t, \ldots\}$), $\varphi, \varphi' \in \text{FORM}$ and $\star$ is a binary logical operator (i.e., $\star \in \{\wedge, \vee, \rightarrow, \leftrightarrow, \oplus\}$).
- Examples:

$$(p \wedge q)$$

Propositional Logics: Syntax

- We can use a grammar to define the set of well formed formulas FORM of propositional logic

$$\text{FORM} := A \mid (\neg\varphi) \mid (\varphi \star \varphi')$$

where A is a propositional symbol (i.e., $A \in \{p, q, r, s, t, \ldots\}$), $\varphi, \varphi' \in$ FORM and $\star$ is a binary logical operator (i.e., $\star \in \{\wedge, \vee, \rightarrow, \leftrightarrow, \oplus\}$).

- Examples:

$$(p \wedge q) \qquad \checkmark$$

Propositional Logics: Syntax

- We can use a grammar to define the set of well formed formulas FORM of propositional logic

$$\text{FORM} := A \mid (\neg\varphi) \mid (\varphi \star \varphi')$$

where A is a propositional symbol (i.e., $A \in \{p, q, r, s, t, \ldots\}$), $\varphi, \varphi' \in \text{FORM}$ and $\star$ is a binary logical operator (i.e., $\star \in \{\wedge, \vee, \rightarrow, \leftrightarrow, \oplus\}$).

- Examples:

$(p \wedge q)$ ✓

$\neg p$

Propositional Logics: Syntax

- We can use a grammar to define the set of well formed formulas FORM of propositional logic

  $$\text{FORM} := A \mid (\neg\varphi) \mid (\varphi \star \varphi')$$

  where A is a propositional symbol (i.e., $A \in \{p, q, r, s, t, \ldots\}$), $\varphi, \varphi' \in$ FORM and $\star$ is a binary logical operator (i.e., $\star \in \{\wedge, \vee, \rightarrow, \leftrightarrow, \oplus\}$).
- Examples:

$(p \wedge q)$	✓
$\neg p$	×

Propositional Logics: Syntax

- We can use a grammar to define the set of well formed formulas FORM of propositional logic

$$\text{FORM} := A \mid (\neg\varphi) \mid (\varphi \star \varphi')$$

  where A is a propositional symbol (i.e., $A \in \{p, q, r, s, t, \ldots\}$), $\varphi, \varphi' \in \text{FORM}$ and $\star$ is a binary logical operator (i.e., $\star \in \{\wedge, \vee, \rightarrow, \leftrightarrow, \oplus\}$).
- Examples:

$$(p \wedge q) \qquad \checkmark$$

$$\neg p \qquad \times$$

$$(p \rightarrow (p \oplus q))$$

Propositional Logics: Syntax

- We can use a grammar to define the set of well formed formulas FORM of propositional logic

$$\text{FORM} := A \mid (\neg\varphi) \mid (\varphi \star \varphi')$$

where A is a propositional symbol (i.e., $A \in \{p, q, r, s, t, \ldots\}$), $\varphi, \varphi' \in$ FORM and $\star$ is a binary logical operator (i.e., $\star \in \{\wedge, \vee, \rightarrow, \leftrightarrow, \oplus\}$).

- Examples:

$(p \wedge q)$	✓
$\neg p$	×
$(p \rightarrow (p \oplus q))$	✓

Too Many Parentheses. . .

Too Many Parentheses...

- Parentheses are needed to ensure a unique mapping between formula and formation tree.

Too Many Parentheses...

- Parentheses are needed to ensure a unique mapping between formula and formation tree.
- If we drop parenthesis in $(\neg(p \to q))$ we obtain $\neg p \to q$. The same formula obtained if we drop parenthesis in $((\neg p) \to q)$.

$$
\begin{array}{ccc}
(\neg(p \to q)) & \searrow & \\
 & & \neg p \to q \\
((\neg p) \to q) & \nearrow &
\end{array}
$$

Too Many Parentheses...

- Parentheses are needed to ensure a unique mapping between formula and formation tree.
- If we drop parenthesis in $(\neg(p \to q))$ we obtain $\neg p \to q$. The same formula obtained if we drop parenthesis in $((\neg p) \to q)$.

$$
\begin{array}{ccc}
(\neg(p \to q)) & \searrow & \\
 & & \neg p \to q \\
((\neg p) \to q) & \nearrow &
\end{array}
$$

- But some parentheses can be avoided:
  - Outer parentheses: we write $\neg p$ instead of $(\neg p)$

Too Many Parentheses...

- Parentheses are needed to ensure a unique mapping between formula and formation tree.
- If we drop parenthesis in $(\neg(p \to q))$ we obtain $\neg p \to q$. The same formula obtained if we drop parenthesis in $((\neg p) \to q)$.

$$
\begin{array}{ccc}
(\neg(p \to q)) & \searrow & \\
 & & \neg p \to q \\
((\neg p) \to q) & \nearrow &
\end{array}
$$

- But some parentheses can be avoided:
  - Outer parentheses: we write $\neg p$ instead of $(\neg p)$
  - Equivalents: we write $p \wedge q \wedge r$ instead of $p \wedge (q \wedge r)$ or $(p \wedge q) \wedge r$

Too Many Parentheses...

- Parentheses are needed to ensure a unique mapping between formula and formation tree.
- If we drop parenthesis in $(\neg(p \to q))$ we obtain $\neg p \to q$. The same formula obtained if we drop parenthesis in $((\neg p) \to q)$.

$$\begin{tikzcd}[column sep=small]
(\neg(p \to q)) \arrow[dr] & \\
& \neg p \to q \\
((\neg p) \to q) \arrow[ur] &
\end{tikzcd}$$

- But some parentheses can be avoided:
  - Outer parentheses: we write $\neg p$ instead of $(\neg p)$
  - Equivalents: we write $p \wedge q \wedge r$ instead of $p \wedge (q \wedge r)$ or $(p \wedge q) \wedge r$
  - Precedence: if we declare that $\neg$ has higher precedence than $\wedge$ then $\neg p \wedge q$ represents $(\neg p) \wedge q$

Propositional Logic: Semantics

Propositional Logic: Semantics

- We want to assign meaning to propositional formulas. I.e., we will define the relation $\models$.

Propositional Logic: Semantics

- We want to assign meaning to propositional formulas. I.e., we will define the relation $\models$.
- First we have to decide which are going to be our models.

Propositional Logic: Semantics

- We want to assign meaning to propositional formulas. I.e., we will define the relation $\models$.
- First we have to decide which are going to be our models.
- We said that our atomic, not further analyzed, formulas were the propositional symbols. Hence, our models should explicitly decide whether propositional formulas are true or false.

Propositional Logic: Semantics

- We want to assign meaning to propositional formulas. I.e., we will define the relation $\models$.
- First we have to decide which are going to be our models.
- We said that our atomic, not further analyzed, formulas were the propositional symbols. Hence, our models should explicitly decide whether propositional formulas are true or false.
- Def.: A propositional model is a mapping of propositional symbols to elements of the set $\{\text{True}, \text{False}\}$ (an assignment or valuation).

Propositional Logic: Semantics

Propositional Logic: Semantics

- Examples of Models:

	M_1
p	True
q	False
r	True

Propositional Logic: Semantics

- Examples of Models:

	M_1	M_2
p	True	True
q	False	True
r	True	True

Propositional Logic: Semantics

- Examples of Models:

	M_1	M_2
p	True	True
q	False	True
r	True	True

We sometimes write these models as

$M_1 = \{p \mapsto \text{True}, q \mapsto \text{False}, r \mapsto \text{True}\}$

Propositional Logic: Semantics

- Examples of Models:

	M_1	M_2
p	True	True
q	False	True
r	True	True

We sometimes write these models as

$M_1 = \{p \mapsto \text{True}, q \mapsto \text{False}, r \mapsto \text{True}\}$

or

$M_1(p) = \text{True}$

$M_1(q) = \text{False}$

$M_1(r) = \text{True}$

Propositional Logic: Semantics

- Examples of Models:

	M_1	M_2
p	True	True
q	False	True
r	True	True

We sometimes write these models as

$M_1 = \{p \mapsto \text{True}, q \mapsto \text{False}, r \mapsto \text{True}\}$

or

$M_1(p) = \text{True}$

$M_1(q) = \text{False}$

$M_1(r) = \text{True}$

- Q 1: If we only have 3 propositional symbols, how many different models are possible?

Propositional Logic: Semantics

- Examples of Models:

	M_1	M_2
p	True	True
q	False	True
r	True	True

We sometimes write these models as

$M_1 = \{p \mapsto \text{True}, q \mapsto \text{False}, r \mapsto \text{True}\}$

or

$M_1(p) = \text{True}$

$M_1(q) = \text{False}$

$M_1(r) = \text{True}$

- Q 1: If we only have 3 propositional symbols, how many different models are possible? 2^3

Propositional Logic: Semantics

- Examples of Models:

	M_1	M_2
p	True	True
q	False	True
r	True	True

We sometimes write these models as

$M_1 = \{p \mapsto \text{True}, q \mapsto \text{False}, r \mapsto \text{True}\}$

or

$M_1(p) = \text{True}$

$M_1(q) = \text{False}$

$M_1(r) = \text{True}$

- Q 1: If we only have 3 propositional symbols, how many different models are possible? 2^3
- Q 2: If we only have a finite number of propositional symbols, how many different models are possible?

Propositional Logic: Semantics

- Examples of Models:

	M_1	M_2
p	True	True
q	False	True
r	True	True

We sometimes write these models as

$M_1 = \{p \mapsto \text{True}, q \mapsto \text{False}, r \mapsto \text{True}\}$

or

$M_1(p) = \text{True}$

$M_1(q) = \text{False}$

$M_1(r) = \text{True}$

- Q 1: If we only have 3 propositional symbols, how many different models are possible? 2^3
- Q 2: If we only have a finite number of propositional symbols, how many different models are possible? Finitely many

Propositional Logic: Semantics

- Examples of Models:

	M_1	M_2
p	True	True
q	False	True
r	True	True

We sometimes write these models as

$M_1 = \{p \mapsto \text{True}, q \mapsto \text{False}, r \mapsto \text{True}\}$

or

$M_1(p) = \text{True}$

$M_1(q) = \text{False}$

$M_1(r) = \text{True}$

- Q 1: If we only have 3 propositional symbols, how many different models are possible? 2^3
- Q 2: If we only have a finite number of propositional symbols, how many different models are possible? Finitely many
- Q 3: How many propositional symbols can appear in an arbitrary formula?

Propositional Logic: Semantics

- Examples of Models:

	M_1	M_2
p	True	True
q	False	True
r	True	True

We sometimes write these models as

$M_1 = \{p \mapsto \text{True}, q \mapsto \text{False}, r \mapsto \text{True}\}$

or

$M_1(p) = \text{True}$

$M_1(q) = \text{False}$

$M_1(r) = \text{True}$

- Q 1: If we only have 3 propositional symbols, how many different models are possible? 2^3
- Q 2: If we only have a finite number of propositional symbols, how many different models are possible? Finitely many
- Q 3: How many propositional symbols can appear in an arbitrary formula? Finitely many

Propositional Logic: Back to the definition of $\models$

Propositional Logic: Back to the definition of $\models$

- Assume we are given a model M, we define

$$M \models p \quad \text{iff} \quad M(p) = \text{True}$$

Propositional Logic: Back to the definition of $\models$

- Assume we are given a model M, we define

$$
\begin{array}{rcl}
M \models p & \text{iff} & M(p) = \text{True} \\
M \models \neg\varphi & \text{iff} & M \not\models \varphi
\end{array}
$$

Propositional Logic: Back to the definition of $\models$

- Assume we are given a model M, we define

$$\begin{aligned} M \models p \quad &\text{iff} \quad M(p) = \text{True} \\ M \models \neg\varphi \quad &\text{iff} \quad M \not\models \varphi \\ M \models \varphi \wedge \psi \quad &\text{iff} \quad M \models \varphi \text{ and } M \models \psi \end{aligned}$$

Propositional Logic: Back to the definition of $\models$

- Assume we are given a model M, we define

$$
\begin{array}{rcl}
M \models p & \text{iff} & M(p) = \text{True} \\
M \models \neg\varphi & \text{iff} & M \not\models \varphi \\
M \models \varphi \wedge \psi & \text{iff} & M \models \varphi \text{ and } M \models \psi \\
M \models \varphi \vee \psi & \text{iff} & M \models \varphi \text{ or } M \models \psi
\end{array}
$$

Propositional Logic: Back to the definition of $\models$

- Assume we are given a model M, we define

$$
\begin{array}{rcl}
M \models p & \text{iff} & M(p) = \text{True} \\
M \models \neg\varphi & \text{iff} & M \not\models \varphi \\
M \models \varphi \wedge \psi & \text{iff} & M \models \varphi \text{ and } M \models \psi \\
M \models \varphi \vee \psi & \text{iff} & M \models \varphi \text{ or } M \models \psi \\
& \dots &
\end{array}
$$

- Example: Given $M = \{p \mapsto \text{True}, q \mapsto \text{False}\}$ then

$$M \models p \vee \neg q \text{ iff}$$

Propositional Logic: Back to the definition of $\models$

- Assume we are given a model M, we define

$$
\begin{array}{rcl}
M \models p & \text{iff} & M(p) = \text{True} \\
M \models \neg\varphi & \text{iff} & M \not\models \varphi \\
M \models \varphi \wedge \psi & \text{iff} & M \models \varphi \text{ and } M \models \psi \\
M \models \varphi \vee \psi & \text{iff} & M \models \varphi \text{ or } M \models \psi \\
& \dots &
\end{array}
$$

- Example: Given $M = \{p \mapsto \text{True}, q \mapsto \text{False}\}$ then

$$M \models p \vee \neg q \text{ iff}$$
$$M \models p \text{ or } M \models \neg q \text{ iff}$$

Propositional Logic: Back to the definition of $\models$

- Assume we are given a model M, we define

$$
\begin{array}{rcl}
M \models p & \text{iff} & M(p) = \text{True} \\
M \models \neg\varphi & \text{iff} & M \not\models \varphi \\
M \models \varphi \wedge \psi & \text{iff} & M \models \varphi \text{ and } M \models \psi \\
M \models \varphi \vee \psi & \text{iff} & M \models \varphi \text{ or } M \models \psi \\
 & \ldots &
\end{array}
$$

- Example: Given $M = \{p \mapsto \text{True}, q \mapsto \text{False}\}$ then

$$
\begin{array}{l}
M \models p \vee \neg q \text{ iff} \\
M \models p \text{ or } M \models \neg q \text{ iff} \\
M(p) = \text{True or } \ldots
\end{array}
$$

Truth Tables

Truth Tables

The definition of $\models$ can also be given in terms of Truth Tables:

$\wedge$	1	0
1	1	0
0	0	0

Truth Tables

The definition of $\models$ can also be given in terms of Truth Tables:

$\wedge$	1	0
1	1	0
0	0	0

$\wedge$	$M \models \varphi$	$M \not\models \varphi$
$M \models \psi$	$M \models \varphi \wedge \psi$	$M \not\models \varphi \wedge \psi$
$M \not\models \psi$	$M \not\models \varphi \wedge \psi$	$M \not\models \varphi \wedge \psi$

Truth Tables

The definition of $\models$ can also be given in terms of Truth Tables:

$\wedge$	1	0
1	1	0
0	0	0

$\wedge$	$M \models \varphi$	$M \not\models \varphi$
$M \models \psi$	$M \models \varphi \wedge \psi$	$M \not\models \varphi \wedge \psi$
$M \not\models \psi$	$M \not\models \varphi \wedge \psi$	$M \not\models \varphi \wedge \psi$

$\vee$	1	0
1	1	1
0	1	0

Truth Tables

The definition of $\models$ can also be given in terms of Truth Tables:

$\wedge$	1	0
1	1	0
0	0	0

$\wedge$	$M \models \varphi$	$M \not\models \varphi$
$M \models \psi$	$M \models \varphi \wedge \psi$	$M \not\models \varphi \wedge \psi$
$M \not\models \psi$	$M \not\models \varphi \wedge \psi$	$M \not\models \varphi \wedge \psi$

$\vee$	1	0
1	1	1
0	1	0

$\vee$	$M \models \varphi$	$M \not\models \varphi$
$M \models \psi$	$M \models \varphi \vee \psi$	$M \models \varphi \vee \psi$
$M \not\models \psi$	$M \models \varphi \vee \psi$	$M \not\models \varphi \vee \psi$

How Many Operators?

How Many Operators?

- Now we can answer the question How many binary logical operators exists in propositional logic?

How Many Operators?

- Now we can answer the question How many binary logical operators exists in propositional logic?
- Each binary logical operator is in correspondence with a 2×2 truth table.

How Many Operators?

- Now we can answer the question How many binary logical operators exists in propositional logic?
- Each binary logical operator is in correspondence with a 2×2 truth table. Hence there are exactly 16 different binary logical operators in propositional logic.

How Many Operators?

- Now we can answer the question How many binary logical operators exists in propositional logic?
- Each binary logical operator is in correspondence with a 2×2 truth table. Hence there are exactly 16 different binary logical operators in propositional logic.
- Do we need all of them?

How Many Operators?

- Now we can answer the question How many binary logical operators exists in propositional logic?
- Each binary logical operator is in correspondence with a 2×2 truth table. Hence there are exactly 16 different binary logical operators in propositional logic.
- Do we need all of them? No, many of them can be defined in terms of others.

How Many Operators?

- Now we can answer the question How many binary logical operators exists in propositional logic?
- Each binary logical operator is in correspondence with a 2×2 truth table. Hence there are exactly 16 different binary logical operators in propositional logic.
- Do we need all of them? No, many of them can be defined in terms of others.
- Examples:

$$p \to q \equiv \neg p \vee q$$

How Many Operators?

- Now we can answer the question How many binary logical operators exists in propositional logic?
- Each binary logical operator is in correspondence with a 2×2 truth table. Hence there are exactly 16 different binary logical operators in propositional logic.
- Do we need all of them? No, many of them can be defined in terms of others.
- Examples:

$p \rightarrow q \equiv \neg p \vee q$

$\rightarrow$	1	0
1	1	0
0	1	1

How Many Operators?

- Now we can answer the question How many binary logical operators exists in propositional logic?
- Each binary logical operator is in correspondence with a 2×2 truth table. Hence there are exactly 16 different binary logical operators in propositional logic.
- Do we need all of them? No, many of them can be defined in terms of others.
- Examples:

$$p \to q \equiv \neg p \vee q$$

$\to$	1	0
1	1	0
0	1	1

p	q	$p \to q$
1	1	1
1	0	0
0	1	1
0	0	1

How Many Operators?

- Now we can answer the question How many binary logical operators exists in propositional logic?
- Each binary logical operator is in correspondence with a 2×2 truth table. Hence there are exactly 16 different binary logical operators in propositional logic.
- Do we need all of them? No, many of them can be defined in terms of others.
- Examples:

$$p \to q \equiv \neg p \vee q$$

$\to$	1	0
1	1	0
0	1	1

p	q	$p \to q$	$\neg p$
1	1	1	0
1	0	0	0
0	1	1	1
0	0	1	1

How Many Operators?

- Now we can answer the question How many binary logical operators exists in propositional logic?
- Each binary logical operator is in correspondence with a 2×2 truth table. Hence there are exactly 16 different binary logical operators in propositional logic.
- Do we need all of them? No, many of them can be defined in terms of others.
- Examples:

$$p \to q \equiv \neg p \vee q$$

$\to$	1	0
1	1	0
0	1	1

p	q	$p \to q$	$\neg p$	$\neg p \vee q$
1	1	1	0	1
1	0	0	0	0
0	1	1	1	1
0	0	1	1	1

One Operator to Rule them All

One Operator to Rule them All

- Actually one single operator, called NAND, is enough to define all others

NAND	1	0
1	0	1
0	1	1

One Operator to Rule them All

- Actually one single operator, called NAND, is enough to define all others

NAND	1	0
1	0	1
0	1	1

- A set of operators that can define all others is called (truth) functionally complete.

One Operator to Rule them All

- Actually one single operator, called NAND, is enough to define all others

NAND	1	0
1	0	1
0	1	1

- A set of operators that can define all others is called (truth) functionally complete.
- Writing things with NAND only can be cumbersome. But it is very much used, for example, to build logical circuits.

One Operator to Rule them All

- Actually one single operator, called NAND, is enough to define all others

NAND	1	0
1	0	1
0	1	1

- A set of operators that can define all others is called (truth) functionally complete.
- Writing things with NAND only can be cumbersome. But it is very much used, for example, to build logical circuits.
- Other examples of functionally complete sets are $\{\neg, \vee\}$ and $\{\neg, \wedge\}$.

'Official' definition of $\models$

'Official' definition of $\models$

- Now that we know that $\{\neg, \vee\}$, for example, are functionally complete, we can give a final definition of $\models$.

'Official' definition of $\models$

- Now that we know that $\{\neg, \vee\}$, for example, are functionally complete, we can give a final definition of $\models$.
  Assume we are given a model M, we define

$$\begin{array}{rcl} M \models p & \text{iff} & M(p) = \text{True} \\ M \models \neg\varphi & \text{iff} & M \not\models \varphi \\ M \models \varphi \wedge \psi & \text{iff} & M \models \varphi \text{ and } M \models \psi \end{array}$$

- Which we can trivially turn into an algorithm Eval to check whether a formula φ is true in a model M.

Eval(M,φ)

Eval(M,φ)

- Let M be a model and φ a formula. Define

  Eval(M,φ) $=$
  if (φ is atomic) **then** return $M(\varphi)$

Eval(M,φ)

- Let M be a model and φ a formula. Define

  Eval(M,φ) $=$
  if (φ is atomic) **then** return $M(\varphi)$
  if ($\varphi = \neg\psi$) **then**
  if Eval(M,ψ) $=$ False **then** return True **else** return False

Eval(M,φ)

- Let M be a model and φ a formula. Define

  Eval(M,φ) $=$
    if (φ is atomic) **then** return $M(\varphi)$
    if ($\varphi = \neg\psi$) **then**
      if Eval(M,ψ) $=$ False **then** return True **else** return False
    if ($\varphi = \psi_1 \wedge \psi_2$) **then**
      if Eval(M,ψ_1) $=$ False **then** return False
      else if Eval(M,ψ_2) $=$ False **then** return False
        else return True

Eval(M,φ)

- Let M be a model and φ a formula. Define

  Eval(M,φ) $=$
    if (φ is atomic) **then** return $M(\varphi)$
    if ($\varphi = \neg\psi$) **then**
      if Eval(M,ψ) $=$ False **then** return True **else** return False
    if ($\varphi = \psi_1 \wedge \psi_2$) **then**
      if Eval(M,ψ_1) $=$ False **then** return False
      else if Eval(M,ψ_2) $=$ False **then** return False
        else return True

- Q 1: How long does it take to run Eval?

Eval(M,φ)

- Let M be a model and φ a formula. Define

  Eval(M,φ) $=$
  if (φ is atomic) **then** return $M(\varphi)$
  if ($\varphi = \neg\psi$) **then**
  if Eval(M,ψ) $=$ False **then** return True **else** return False
  if ($\varphi = \psi_1 \wedge \psi_2$) **then**
  if Eval(M,ψ_1) $=$ False **then** return False
  else if Eval(M,ψ_2) $=$ False **then** return False
  else return True

- Q 1: How long does it take to run Eval? It needs at most as many steps as operators are in the formula.

Eval(M,φ)

- Let M be a model and φ a formula. Define

  Eval(M,φ) =
  if (φ is atomic) **then** return $M(\varphi)$
  if ($\varphi = \neg\psi$) **then**
  if Eval(M,ψ) = False **then** return True **else** return False
  if ($\varphi = \psi_1 \wedge \psi_2$) **then**
  if Eval(M,ψ_1) = False **then** return False
  else if Eval(M,ψ_2) = False **then** return False
  else return True

- Q 1: How long does it take to run Eval? It needs at most as many steps as operators are in the formula.
- Q 2: Which propositional symbols are inspected during a run of Eval on M and φ?

Eval(M,φ)

- Let M be a model and φ a formula. Define

  Eval(M,φ) =
  if (φ is atomic) **then** return $M(\varphi)$
  if ($\varphi = \neg\psi$) **then**
  if Eval(M,ψ) = False **then** return True **else** return False
  if ($\varphi = \psi_1 \wedge \psi_2$) **then**
  if Eval(M,ψ_1) = False **then** return False
  else if Eval(M,ψ_2) = False **then** return False
  else return True

- Q 1: How long does it take to run Eval? It needs at most as many steps as operators are in the formula.
- Q 2: Which propositional symbols are inspected during a run of Eval on M and φ? Only those appearing in φ.

Counting Models

Counting Models

- The last question is very important. Let's repeat the answer:

  When we are evaluating a formula φ in a model M
  only the propositional variables in φ are checked.

Counting Models

- The last question is very important. Let's repeat the answer:

  When we are evaluating a formula φ in a model M
  only the propositional variables in φ are checked.

- In other words, if two models M_1 and M_2 assign the same truth values to the variables in φ they will both agree on the truth value of φ (independently of the other values).

Counting Models

- The last question is very important. Let's repeat the answer:

  When we are evaluating a formula φ in a model M
  only the propositional variables in φ are checked.

- In other words, if two models M_1 and M_2 assign the same truth values to the variables in φ they will both agree on the truth value of φ (independently of the other values).
- Third try: when looking for a model for φ we only need to inspect a finite number of models (at most 2^n for n the number of propositional variables in φ).

Solving Satisfiability

Solving Satisfiability

- Let's define an algorithm Sat(φ) to solve the following problems: Given a formula φ answer YES if the formula is satisfiable, and NO otherwise.

Solving Satisfiability

- Let's define an algorithm Sat(φ) to solve the following problems: Given a formula φ answer YES if the formula is satisfiable, and NO otherwise.

  Sat(φ) =
  let S = the set of 2^n different models defined
  over the n propositional letters appearing in φ

Solving Satisfiability

- Let's define an algorithm Sat(φ) to solve the following problems: Given a formula φ answer YES if the formula is satisfiable, and NO otherwise.

  Sat(φ) =
    let S = the set of 2^n different models defined
          over the n propositional letters appearing in φ
    for each $M \in S$ **do**
      if Eval(M,φ) = True **then** return YES

Solving Satisfiability

- Let's define an algorithm Sat(φ) to solve the following problems: Given a formula φ answer YES if the formula is satisfiable, and NO otherwise.

Sat(φ) =
  let S = the set of 2^n different models defined
        over the n propositional letters appearing in φ
  for each $M \in S$ **do**
    if Eval(M,φ) = True **then** return YES
  return NO

Solving Satisfiability

- Let's define an algorithm Sat(φ) to solve the following problems: Given a formula φ answer YES if the formula is satisfiable, and NO otherwise.

  Sat(φ) =
    let S = the set of 2^n different models defined
          over the n propositional letters appearing in φ
    for each $M \in S$ **do**
      if Eval(M,φ) = True **then** return YES
    return NO

- This algorithm makes, in the worst case, exponentially many steps (and that is the best we can do for the moment).

Solving Satisfiability

- Let's define an algorithm Sat(φ) to solve the following problems: Given a formula φ answer YES if the formula is satisfiable, and NO otherwise.

  Sat(φ) =
  let S = the set of 2^n different models defined
  over the n propositional letters appearing in φ
  for each $M \in S$ **do**
  if Eval(M,φ) = True **then** return YES
  return NO

- This algorithm makes, in the worst case, exponentially many steps (and that is the best we can do for the moment).
- The algorithm Sat is called a decision procedure for satisfiability, because it always terminate with the correct answer to the question "Is φ satisfiable?"

Improving the Sat Algorithm...

- We don't know how to improve the worst case complexity of the Sat algorithm, but we can do better in the average.

Improving the Sat Algorithm...

- We don't know how to improve the worst case complexity of the Sat algorithm, but we can do better in the average.
- An important weakness of the Sat algorithm we defined is that we don't learn anything from each evaluation to Eval.

Improving the Sat Algorithm...

- We don't know how to improve the worst case complexity of the Sat algorithm, but we can do better in the average.
- An important weakness of the Sat algorithm we defined is that we don't learn anything from each evaluation to Eval.

  Suppose we are evaluating the formula $(\neg p \wedge \neg q) \wedge \neg r$

p	q	r	$(\neg p \wedge \neg q) \wedge \neg q$
1	1	1	0
1	1	0	0
1	0	1	0
1	0	0	0
0	1	1	0
0	1	0	0
0	0	1	0
0	0	0	1

Improving the Sat Algorithm...

- We don't know how to improve the worst case complexity of the Sat algorithm, but we can do better in the average.
- An important weakness of the Sat algorithm we defined is that we don't learn anything from each evaluation to Eval.

  Suppose we are evaluating the formula $(\neg p \wedge \neg q) \wedge \neg r$

p	q	r	$(\neg p \wedge \neg q) \wedge \neg q$
1	1	1	0
1	1	0	0
1	0	1	0
1	0	0	0
0	1	1	0
0	1	0	0
0	0	1	0
0	0	0	1

- For example, our algorithm could realize already after the first step, that if we want a model for the whole formula then r should be set to False, and don't try setting it to True any more.

Semantic Tableaux

- Intuitively, tableaux are ways to systematically organize the search for a model of a formula.
- Tableaux are built by applying rules to an input formula. These rules break down the formula to detect all possible ways of building a model.
- Each branch of a tableaux records one way of trying to build a model. Some branches ("closed branches") don't lead to models. Others branching ("open branches") do.
- The best way to learn is via an example...

Tableaux for Propositional Logics

Tableaux for Propositional Logics

Let's see if we can build a model for $(\neg(p \wedge q) \wedge \neg\neg r) \wedge p$.

Tableaux for Propositional Logics

Let's see if we can build a model for $(\neg(p \wedge q) \wedge \neg\neg r) \wedge p$.

Rules for $\neg$ and $\wedge$

Tableaux for Propositional Logics

Let's see if we can build a model for $(\neg(p \wedge q) \wedge \neg\neg r) \wedge p$.

Rules for $\neg$ and $\wedge$

$$\frac{(\varphi \wedge \psi)}{\begin{array}{c}\varphi\\ \psi\end{array}}\ (\wedge)$$

Tableaux for Propositional Logics

Let's see if we can build a model for $(\neg(p \wedge q) \wedge \neg\neg r) \wedge p$.

Rules for $\neg$ and $\wedge$

$$\frac{(\varphi \wedge \psi)}{\begin{array}{c}\varphi \\ \psi\end{array}}\ (\wedge)$$

$$\frac{\neg(\varphi \wedge \psi)}{\neg\varphi \qquad\qquad \neg\psi}\ (\neg\wedge)$$

Tableaux for Propositional Logics

Let's see if we can build a model for $(\neg(p \wedge q) \wedge \neg\neg r) \wedge p$.

Rules for $\neg$ and $\wedge$

$$\frac{(\varphi \wedge \psi)}{\begin{array}{c}\varphi \\ \psi\end{array}} \ (\wedge)$$

$$\frac{\neg(\varphi \wedge \psi)}{\neg\varphi \qquad\qquad \neg\psi} \ (\neg\wedge)$$

$$\frac{\neg\neg\varphi}{\varphi} \ (\neg\neg)$$

Tableaux for Propositional Logics

Let's see if we can build a model for $(\neg(p \wedge q) \wedge \neg\neg r) \wedge p$.

Rules for $\neg$ and $\wedge$

$$\frac{(\varphi \wedge \psi)}{\begin{array}{c}\varphi\\ \psi\end{array}}\ (\wedge)$$

$$\frac{\neg(\varphi \wedge \psi)}{\neg\varphi \qquad\qquad \neg\psi}\ (\neg\wedge)$$

$$\frac{\neg\neg\varphi}{\varphi}\ (\neg\neg)$$

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

Tableaux for Propositional Logics

Let's see if we can build a model for $(\neg(p \wedge q) \wedge \neg\neg r) \wedge p$.

Rules for $\neg$ and $\wedge$

$$\frac{(\varphi \wedge \psi)}{\begin{array}{c}\varphi\\ \psi\end{array}}\ (\wedge)$$

$$\frac{\neg(\varphi \wedge \psi)}{\neg\varphi \qquad\qquad \neg\psi}\ (\neg\wedge)$$

$$\frac{\neg\neg\varphi}{\varphi}\ (\neg\neg)$$

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

Tableaux for Propositional Logics

Let's see if we can build a model for $(\neg(p \wedge q) \wedge \neg\neg r) \wedge p$.

Rules for $\neg$ and $\wedge$

$$\frac{(\varphi \wedge \psi)}{\begin{array}{c}\varphi \\ \psi\end{array}} \ (\wedge)$$

$$\frac{\neg(\varphi \wedge \psi)}{\neg\varphi \qquad\qquad \neg\psi} \ (\neg\wedge)$$

$$\frac{\neg\neg\varphi}{\varphi} \ (\neg\neg)$$

$$\begin{array}{c}
(\neg(p \wedge q) \wedge \neg\neg r) \wedge p \\
\neg(p \wedge q) \wedge \neg\neg r \\
p \\
\neg(p \wedge q) \\
\neg\neg r
\end{array}$$

Tableaux for Propositional Logics

Let's see if we can build a model for $(\neg(p \wedge q) \wedge \neg\neg r) \wedge p$.

Rules for $\neg$ and $\wedge$

$$\frac{(\varphi \wedge \psi)}{\begin{array}{c}\varphi \\ \psi\end{array}}\ (\wedge)$$

$$\frac{\neg(\varphi \wedge \psi)}{\neg\varphi \qquad\qquad \neg\psi}\ (\neg\wedge)$$

$$\frac{\neg\neg\varphi}{\varphi}\ (\neg\neg)$$

$$\begin{array}{c}
(\neg(p \wedge q) \wedge \neg\neg r) \wedge p \\
\neg(p \wedge q) \wedge \neg\neg r \\
p \\
\neg(p \wedge q) \\
\neg\neg r \\
r
\end{array}$$

Tableaux for Propositional Logics

Let's see if we can build a model for $(\neg(p \wedge q) \wedge \neg\neg r) \wedge p$.

Rules for $\neg$ and $\wedge$

$$\frac{(\varphi \wedge \psi)}{\begin{array}{c}\varphi \\ \psi\end{array}} \ (\wedge)$$

$$\frac{\neg(\varphi \wedge \psi)}{\neg\varphi \qquad\qquad \neg\psi} \ (\neg\wedge)$$

$$\frac{\neg\neg\varphi}{\varphi} \ (\neg\neg)$$

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

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

$$p$$

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

$$\neg\neg r$$

$$r$$

$$\neg p \qquad\qquad\qquad\qquad \neg q$$

Tableaux for Propositional Logics

Let's see if we can build a model for $(\neg(p \wedge q) \wedge \neg\neg r) \wedge p$.

Rules for $\neg$ and $\wedge$

$$\frac{(\varphi \wedge \psi)}{\begin{array}{c}\varphi \\ \psi\end{array}} \ (\wedge)$$

$$\frac{\neg(\varphi \wedge \psi)}{\neg\varphi \qquad\qquad \neg\psi} \ (\neg\wedge)$$

$$\frac{\neg\neg\varphi}{\varphi} \ (\neg\neg)$$

$$\begin{array}{ccc}
 & (\neg(p \wedge q) \wedge \neg\neg r) \wedge p & \\
 & \neg(p \wedge q) \wedge \neg\neg r & \\
 & p & \\
 & \neg(p \wedge q) & \\
 & \neg\neg r & \\
 & r & \\
\neg p & & \neg q
\end{array}$$

Contradiction!!!

Tableaux for Propositional Logics

Let's see if we can build a model for $(\neg(p \wedge q) \wedge \neg\neg r) \wedge p$.

Rules for $\neg$ and $\wedge$

$$\frac{(\varphi \wedge \psi)}{\begin{array}{c}\varphi \\ \psi\end{array}}\ (\wedge)$$

$$\frac{\neg(\varphi \wedge \psi)}{\neg\varphi \qquad\qquad \neg\psi}\ (\neg\wedge)$$

$$\frac{\neg\neg\varphi}{\varphi}\ (\neg\neg)$$

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

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

$$p$$

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

$$\neg\neg r$$

$$r$$

$\neg p$	$\neg q$
Contradiction!!!	*Model*

Describing and Answering Questions

Describing and Answering Questions

- We started this lecture saying that logics can be used to describe things.

Describing and Answering Questions

- We started this lecture saying that logics can be used to describe things.
- And that they were better than pictures because we could query this descriptions and obtain answers.

Describing and Answering Questions

- We started this lecture saying that logics can be used to describe things.
- And that they were better than pictures because we could query this descriptions and obtain answers.
- Let's see a concrete example of this. Suppose that you are charged with the sitting arrangement for the dinner tonight.
  - There are three chairs arranged in a line on one side of a table.
  - These chairs should be occupied by your Aunt, your Father and your Sister.
  - Sister doesn't want to sit in the middle chair
  - Aunty doesn't want to sit next to your Father.

Describing and Answering Questions

Describing and Answering Questions

- In this case it is easy to see that there is no way to arrange the seats.

Describing and Answering Questions

- In this case it is easy to see that there is no way to arrange the seats. But what if Grandma and the neighbors are also coming for dinner!!!

Describing and Answering Questions

- In this case it is easy to see that there is no way to arrange the seats. But what if Grandma and the neighbors are also coming for dinner!!!
- It is easy to describe the problem using propositional logic, and any model of the description (e.g., obtained using tableaux) will give us a suitable table arrangement.

Describing and Answering Questions

- In this case it is easy to see that there is no way to arrange the seats. But what if Grandma and the neighbors are also coming for dinner!!!
- It is easy to describe the problem using propositional logic, and any model of the description (e.g., obtained using tableaux) will give us a suitable table arrangement.
- In the course that begins tomorrow I will start again with Propositional Logic, and show that many difficult problems can be modeled using Propositional Logic, and we will talk more in detail about improvements of the Sat algorithm.

Describing and Answering Questions

- In this case it is easy to see that there is no way to arrange the seats. But what if Grandma and the neighbors are also coming for dinner!!!
- It is easy to describe the problem using propositional logic, and any model of the description (e.g., obtained using tableaux) will give us a suitable table arrangement.
- In the course that begins tomorrow I will start again with Propositional Logic, and show that many difficult problems can be modeled using Propositional Logic, and we will talk more in detail about improvements of the Sat algorithm.
- We will also discuss the limitations of propositional logics, and in which cases its expressiveness is too limited and present ways in which we can extend it.