ECE750T-28: Computer-aided Reasoning for Software Engineering

Lecture 1: Introduction to Logic in SE

Vijay Ganesh
(Original notes from Isil Dillig)
This course is about computational logic and its application to software engineering.
What is this Course About?

- This course is about computational logic and its application to software engineering.
- Explore various logical theories widely used in computer science.
What is this Course About?

- This course is about computational logic and its application to software engineering.

- Explore various logical theories widely used in computer science.

- Learn about decision procedures, provers, solvers.
What is this Course About?

- This course is about computational logic and its application to software engineering.
- Explore various logical theories widely used in computer science.
- Learn about decision procedures, provers, solvers.
- Learn about applications such as concolic testing, model checking, analysis, fault localization, synthesis and programming languages.
Why Should You Care?

Logic is a fundamental part of computer science:
Logic is a fundamental part of computer science:

- Computation, irrespective of representation, can be very complex to understand/process in all its gory detail.
Why Should You Care?

Logic is a fundamental part of computer science:

- Computation, irrespective of representation, can be very complex to understand/process in all its gory detail.

- Hence, we need abstractions.
Why Should You Care?

Logic is a fundamental part of computer science:

- Computation, irrespective of representation, can be very complex to understand/process in all its gory detail.

- Hence, we need abstractions.

- Logics are precise languages that allow us to represent/manipulate/process/morph abstractions of computations.
Why Should You Care?

Logic is a fundamental part of computer science:

- Computation, irrespective of representation, can be very complex to understand/process in all its gory detail.

- Hence, we need abstractions.

- Logics are precise languages that allow us to represent/manipulate/process/morph abstractions of computations.

- Examples include Boolean logic (aka propositional or sentential calculus), predicate logic, first-order theories,...
Why Should You Care?

Logic is a fundamental part of computer science:

- Artificial intelligence: constraint satisfaction, automated game playing, planning, . . .
- Programming Languages: logic programming, type systems, programming language theory . . .
- Hardware verification and synthesis: correctness of circuits, ATPG, . . .
- Program analysis, verification and synthesis: Static analysis, software verification, test case generation, program understanding, . . .
Why Should You Care?

Logic is a fundamental part of computer science:

- **Artificial intelligence**: constraint satisfaction, automated game playing, planning, . . .
Why Should You Care?

Logic is a fundamental part of computer science:

- **Artificial intelligence:** constraint satisfaction, automated game playing, planning, . . .

- **Programming Languages:** logic programming, type systems, programming language theory . . .
Why Should You Care?

Logic is a fundamental part of computer science:

▶ **Artificial intelligence:** constraint satisfaction, automated game playing, planning, . . .

▶ **Programming Languages:** logic programming, type systems, programming language theory . . .

▶ **Hardware verification and synthesis:** correctness of circuits, ATPG, . . .
Why Should You Care?

Logic is a fundamental part of computer science:

▶ **Artificial intelligence**: constraint satisfaction, automated game playing, planning, . . .

▶ **Programming Languages**: logic programming, type systems, programming language theory . . .

▶ **Hardware verification and synthesis**: correctness of circuits, ATPG, . . .

▶ **Program analysis, verification and synthesis**: Static analysis, software verification, test case generation, program understanding, . . .
Why Should You Care?

- No matter what your research area or interest is, the techniques we cover in this course are likely to be relevant.
Why Should You Care?

- No matter what your research area or interest is, the techniques we cover in this course are likely to be relevant.

- Very good tool kit because many difficult problems can be reduced deciding satisfiability of formulas in logic.
Topics Covered in the Course

- Review of propositional logic
Topics Covered in the Course

- Review of propositional logic
- Modern SAT solvers
Topics Covered in the Course

- Review of propositional logic
- Modern SAT solvers
- Complexity theory basics, reductions, classes,...
Topics Covered in the Course

- Review of propositional logic
- Modern SAT solvers
- Complexity theory basics, reductions, classes,…
- First-order theorem provers
Topics Covered in the Course

- Review of propositional logic
- Modern SAT solvers
- Complexity theory basics, reductions, classes,…
- First-order theorem provers
- Theory of uninterpreted functions
Topics Covered in the Course

- Review of propositional logic
- Modern SAT solvers
- Complexity theory basics, reductions, classes,…
- First-order theorem provers
- Theory of uninterpreted functions
- Linear inequalities over reals (Simplex) and integers
Topics Covered in the Course

▶ Review of propositional logic

▶ Modern SAT solvers

▶ Complexity theory basics, reductions, classes, ...

▶ First-order theorem provers

▶ Theory of uninterpreted functions

▶ Linear inequalities over reals (Simplex) and integers

▶ Theories of bit-vectors, arrays and strings
Topics Covered in the Course

- Combining decision procedures (Nelson-Oppen)
Topics Covered in the Course

- Combining decision procedures (Nelson-Oppen)
- SMT Solvers and the DPLL(T) framework
Topics Covered in the Course

▶ Combining decision procedures (Nelson-Oppen)
▶ SMT Solvers and the DPLL(T) framework
▶ Constraint Simplification
Topics Covered in the Course

▶ Combining decision procedures (Nelson-Oppen)
▶ SMT Solvers and the DPLL(T) framework
▶ Constraint Simplification
▶ Quantifier elimination
Topics Covered in the Course

- Combining decision procedures (Nelson-Oppen)
- SMT Solvers and the DPLL(T) framework
- Constraint Simplification
- Quantifier elimination
- Applications: concolic testing, analysis, formal methods
Logistics

- Class meets every Friday from 11:30 AM to 2:20 PM

All lectures will be held in EIT 3141. All the material for the class (lecture slides, homework, reading, announcements) will be posted on the course website: https://ece.uwaterloo.ca/~vganesh/teaching.html
Logistics

- Class meets every Friday from 11:30 AM to 2:20 PM
- All lectures will be held in EIT 3141

All the material for the class (lecture slides, homework, reading, announcements) will be posted on the course website: [https://ece.uwaterloo.ca/~vganesh/teaching.html](https://ece.uwaterloo.ca/~vganesh/teaching.html)
Logistics

- Class meets every Friday from 11:30 AM to 2:20 PM
- All lectures will be held in EIT 3141
- All the material for the class (lecture slides, homework, reading, announcements) will be posted on the course website:

  https://ece.uwaterloo.ca/~vganesh/teaching.html
Recommended Books

- *The Calculus of Computation* by Aaron Bradley and Zohar Manna

Warning: Will cover many topics not in the Bradley & Manna textbook and will skip some chapters of this textbook
Recommended Books

- **The Calculus of Computation** by Aaron Bradley and Zohar Manna

  ![The Calculus of Computation book cover]

  **Warning**: Will cover many topics not in the Bradley & Manna textbook and will skip some chapters of this textbook
Another Recommended Book

- Decision Procedures: An Algorithmic Point of View by Daniel Kroening and Ofer Strichman
Another Recommended Book

- Decision Procedures: An Algorithmic Point of View by Daniel Kroening and Ofer Strichman

- Mostly I will follow papers, and these papers will be cited on the website.
Requirements

- Two homework assignments (15% of the final grade)
Requirements

- Two homework assignments (15% of the final grade)
- You get 1-2 weeks to complete the assignment from the day it is handed out
Requirements

- Two homework assignments (15% of the final grade)
- You get 1-2 weeks to complete the assignment from the day it is handed out
- No late submissions
Requirements

- Two homework assignments (15% of the final grade)
- You get 1-2 weeks to complete the assignment from the day it is handed out
- No late submissions
- All assignments should be done individually
Final Exam

- One final exam (50% of the final grade)
Final Exam

- One final exam (50% of the final grade)

- No mid-term exam
Final Exam

- One final exam (50% of the final grade)
- No mid-term exam
- All exams closed-book, closed-notes, closed-laptop, closed-phone etc.
Final Exam

- One final exam (50% of the final grade)
- No mid-term exam
- All exams closed-book, closed-notes, closed-laptop, closed-phone etc.
- Date fixed by registrar. Non-negotiable.
Research Projects

- Research Project (35% of the final grade)

- Maximum 2 people per research project group

- Ideally, new research that is publishable. Both theoretical or practical projects are acceptable

- Examples include: Novel solving technique, decidability/complexity result, feature in solver/prover, application of logics

- Must get approval of the project idea from instructor by October 4th, 2013

- Must submit 2-page project proposal with title, names, abstract, problem statement, solution description, impact
Research Projects

- Research Project (35% of the final grade)
- Maximum 2 people per research project group
Research Projects

- Research Project (35% of the final grade)

- Maximum 2 people per research project group

- Ideally, new research that is publishable. Both theoretical or practical projects are acceptable
Research Projects

- Research Project (35% of the final grade)

- Maximum 2 people per research project group

- Ideally, new research that is publishable. Both theoretical or practical projects are acceptable

- Examples include: Novel solving technique, decidability/complexity result, feature in solver/prover, application of logics
Research Projects

- Research Project (35% of the final grade)
- Maximum 2 people per research project group
- Ideally, new research that is publishable. Both theoretical or practical projects are acceptable
- Examples include: Novel solving technique, decidability/complexity result, feature in solver/prover, application of logics
- Must get approval of the project idea from instructor by October 4th, 2013
Research Projects

- Research Project (35% of the final grade)
- Maximum 2 people per research project group
- Ideally, new research that is publishable. Both theoretical or practical projects are acceptable
- Examples include: Novel solving technique, decidability/complexity result, feature in solver/prover, application of logics
- Must get approval of the project idea from instructor by October 4th, 2013
- Must submit 2-page project proposal with title, names, abstract, problem statement, solution description, impact
Grading

▸ Final exam: 50%

▸ Homeworks and class participation: 15%

▸ Respect honor code on exams and homework

▸ You can consult other students on the homework, but write-up must be your own

▸ Also, write-up must mention names of consultants/collaborators

▸ Research project: 35%
Grading

- Final exam: 50%
- Homeworks and class participation: 15%

Respect honor code on exams and homework. You can consult other students on the homework, but write-up must be your own. Also, write-up must mention names of consultants/collaborators.
Grading

- Final exam: 50%
- Homeworks and class participation: 15%
- Respect honor code on exams and homework
Grading

- Final exam: 50%
- Homeworks and class participation: 15%
- Respect honor code on exams and homework
- You can consult other students on the homework, but write-up must be your own
Grading

- Final exam: 50%
- Homeworks and class participation: 15%
- Respect honor code on exams and homework
- You can consult other students on the homework, but write-up must be your own
- Also, write-up must mention names of consultants/collaborators
Grading

- Final exam: 50%
- Homeworks and class participation: 15%
- Respect honor code on exams and homework
- You can consult other students on the homework, but write-up must be your own
- Also, write-up must mention names of consultants/collaborators
- Research project: 35%
Let’s get started!
What are Logics?

- Precise mathematical languages with well-defined syntax and semantics
What are Logics?

- Precise mathematical languages with well-defined syntax and semantics
- Syntax: Meta-rules for defining well-formed formulas
What are Logics?

- Precise mathematical languages with well-defined syntax and semantics
- Syntax: Meta-rules for defining well-formed formulas
- Semantics: Interpretation/models
What are Logics?

- Precise mathematical languages with well-defined syntax and semantics
- Syntax: Meta-rules for defining well-formed formulas
- Semantics: Interpretation/models
- Forms of valid reasoning: Deductive, inductive, abductive,…
What are Logics?

- Precise mathematical languages with well-defined syntax and semantics
- Syntax: Meta-rules for defining well-formed formulas
- Semantics: Interpretation/models
- Forms of valid reasoning: Deductive, inductive, abductive, ...
- Proof systems: Intuitionistic vs. classical
What are Logics?

▶ Precise mathematical languages with well-defined syntax and semantics

▶ Syntax: Meta-rules for defining well-formed formulas

▶ Semantics: Interpretation/models

▶ Forms of valid reasoning: Deductive, inductive, abductive,…

▶ Proof systems: Intuitionistic vs. classical

▶ Fields of study: proof, model, set, recursion, and type theory
What are Logics?

- Precise mathematical languages with well-defined syntax and semantics
- Syntax: Meta-rules for defining well-formed formulas
- Semantics: Interpretation/models
- Forms of valid reasoning: Deductive, inductive, abductive,…
- Proof systems: Intuitionistic vs. classical
- Fields of study: proof, model, set, recursion, and type theory
- Questions studied: Proof and truth, Provability, Decidability, Complexity, Foundations,…
What are Logics?

- Precise mathematical languages with well-defined syntax and semantics
- Syntax: Meta-rules for defining well-formed formulas
- Semantics: Interpretation/models
- Forms of valid reasoning: Deductive, inductive, abductive,…
- Proof systems: Intuitionistic vs. classical
- Fields of study: proof, model, set, recursion, and type theory
- Questions studied: Proof and truth, Provability, Decidability, Complexity, Foundations,…
- Properties of logics: Soundness, completeness, compactness, expressive power, decidability,…
Review of Propositional Logic: Syntax

**Atom**

truth symbols $\top$ ("true") and $\bot$ ("false")

propositional variables $p, q, r, p_1, q_1, r_1, \cdots$
Review of Propositional Logic: Syntax

Atom
truth symbols $\top$ ("true") and $\bot$ ("false")
propositional variables $p, q, r, p_1, q_1, r_1, \cdots$

Literal
atom $\alpha$ or its negation $\neg \alpha$
Review of Propositional Logic: Syntax

Atom  
truth symbols \( \top \) (“true”) and \( \bot \) (“false”)
propositional variables \( p, q, r, p_1, q_1, r_1, \ldots \)

Literal  
atom \( \alpha \) or its negation \( \neg \alpha \)

Formula  
literal or application of a
logical connective to formulae \( F, F_1, F_2 \)
Review of Propositional Logic: Syntax

Atom 
- truth symbols \( \top \) ("true") and \( \bot \) ("false")
- propositional variables \( p, q, r, p_1, q_1, r_1, \ldots \)

Literal
- atom \( \alpha \) or its negation \( \neg \alpha \)

Formula
- literal or application of a logical connective to formulae \( F, F_1, F_2 \)

\[ \neg F \quad \text{"not"} \quad \text{(negation)} \]
Review of Propositional Logic: Syntax

Atom  truth symbols $\top$ ("true") and $\bot$ ("false")
propositional variables $p, q, r, p_1, q_1, r_1, \cdots$

Literal atom $\alpha$ or its negation $\neg \alpha$

Formula literal or application of a
logical connective to formulae $F, F_1, F_2$

$$\neg F$$ “not” (negation)
$$F_1 \land F_2$$ “and” (conjunction)
Review of Propositional Logic: Syntax

Atom  truth symbols $\top$ ("true") and $\bot$ ("false")
propositional variables $p, q, r, p_1, q_1, r_1, \cdots$

Literal  atom $\alpha$ or its negation $\neg\alpha$

Formula  literal or application of a
logical connective to formulae $F, F_1, F_2$

$\neg F$  "not"  (negation)

$F_1 \land F_2$  "and"  (conjunction)

$F_1 \lor F_2$  "or"  (disjunction)
Review of Propositional Logic: Syntax

Atom truth symbols $\top$ ("true") and $\bot$ ("false")
propositional variables $p, q, r, p_1, q_1, r_1, \ldots$

Literal atom $\alpha$ or its negation $\neg\alpha$

Formula literal or application of a
logical connective to formulae $F, F_1, F_2$

- $\neg F$ "not" (negation)
- $F_1 \land F_2$ "and" (conjunction)
- $F_1 \lor F_2$ "or" (disjunction)
- $F_1 \rightarrow F_2$ "implies" (implication)
Review of Propositional Logic: Syntax

Atom  truth symbols $\top$ ("true") and $\bot$ ("false")
propositional variables $p, q, r, p_1, q_1, r_1, \cdots$

Literal  atom $\alpha$ or its negation $\neg\alpha$

Formula  literal or application of a
logical connective to formulae $F, F_1, F_2$

$\neg F$  "not"  (negation)
$F_1 \land F_2$  "and"  (conjunction)
$F_1 \lor F_2$  "or"  (disjunction)
$F_1 \rightarrow F_2$  "implies"  (implication)
$F_1 \leftrightarrow F_2$  "if and only if"  (iff)
Interpretations in Propositional Logic

- An interpretation $I$ for a formula $F$ in propositional logic is a mapping from each propositional variable in $F$ to exactly one truth value

\[ I : \{ p \mapsto \top, q \mapsto \bot, \cdots \} \]
Interpretations in Propositional Logic

- An interpretation $I$ for a formula $F$ in propositional logic is a mapping from each propositional variables in $F$ to exactly one truth value

  $$I : \{ p \mapsto \top, q \mapsto \bot, \cdots \}$$

- For a formula $F$ with 2 propositional variables, how many interpretations are there?
Interpretations in Propositional Logic

- An interpretation $I$ for a formula $F$ in propositional logic is a mapping from each propositional variable in $F$ to exactly one truth value

$$I : \{p \mapsto \top, q \mapsto \bot, \cdots \}$$

- For a formula $F$ with 2 propositional variables, how many interpretations are there?

- In general, for formula with $n$ propositional variables, how many interpretations?
Entailment

- Under an interpretation, every propositional formula evaluates to $T$ or $F$

Formula $F$ + Interpretation $I = \text{Truth value}$
Entailment

- Under an interpretation, every propositional formula evaluates to $T$ or $F$

  \[ F + \text{Interpretation } I = \text{Truth value} \]

- We write $I \models F$ if $F$ evaluates to $\top$ under $I$ (satisfying interpretation)
Entailment

- Under an interpretation, every propositional formula evaluates to $T$ or $F$

  Formula $F$ + Interpretation $I = \text{Truth value}$

- We write $I \models F$ if $F$ evaluates to $\top$ under $I$ (satisfying interpretation)

- Similarly, $I \not\models F$ if $F$ evaluates to $\bot$ under $I$ (falsifying interpretation).
Inductive Definition of Propositional Semantics

\textbf{Base Cases:}

\[ I \models \top \]

\[ I \not\models \bot \]

\[ I \models p \iff I \models \llbracket p \rrbracket = \top \]

\[ I \not\models p \iff I \models \llbracket p \rrbracket = \bot \]
Inductive Definition of Propositional Semantics

**Base Cases:**

\[
\begin{align*}
I \models \top & \quad I \not\models \bot
\end{align*}
\]
Inductive Definition of Propositional Semantics

**Base Cases:**

\[ I \models \top \quad I \not\models \bot \]

\[ I \models p \quad \text{iff} \quad I[p] = \top \]
Inductive Definition of Propositional Semantics

Base Cases:

\[ I \models \top \quad I \not\models \bot \]
\[ I \models p \quad \text{iff} \quad I[p] = \top \]
\[ I \not\models p \quad \text{iff} \quad I[p] = \bot \]
Inductive Definition of Propositional Semantics

Base Cases:

\[ I \models \top \quad I \not\models \bot \]

\[ I \models p \iff I[p] = \top \]

\[ I \not\models p \iff I[p] = \bot \]

Inductive Cases:
Inductive Definition of Propositional Semantics

Base Cases:

\[ I \models \top \quad I \not\models \bot \]
\[ I \models p \quad \text{iff} \quad I[p] = \top \]
\[ I \not\models p \quad \text{iff} \quad I[p] = \bot \]

Inductive Cases:

\[ I \models \neg F \quad \text{iff} \quad I \not\models F \]
Inductive Definition of Propositional Semantics

**Base Cases:**
- \( I \models \top \) \quad \text{iff} \quad I \not\models \bot \\
- \( I \models p \) \quad \text{iff} \quad I[p] = \top \\
- \( I \not\models p \) \quad \text{iff} \quad I[p] = \bot \\

**Inductive Cases:**
- \( I \models \neg F \) \quad \text{iff} \quad I \not\models F \\
- \( I \models F_1 \land F_2 \) \quad \text{iff} \quad I \models F_1 \text{ and } I \models F_2
Inductive Definition of Propositional Semantics

Base Cases:
\[ I \models \top \quad I \not\models \bot \]
\[ I \models p \quad \text{iff} \quad I[p] = \top \]
\[ I \not\models p \quad \text{iff} \quad I[p] = \bot \]

Inductive Cases:
\[ I \models \neg F \quad \text{iff} \quad I \not\models F \]
\[ I \models F_1 \land F_2 \quad \text{iff} \quad I \models F_1 \quad \text{and} \quad I \models F_2 \]
\[ I \models F_1 \lor F_2 \quad \text{iff} \quad I \models F_1 \quad \text{or} \quad I \models F_2 \]
Inductive Definition of Propositional Semantics

**Base Cases:**
\[
I \models \top \quad I \not\models \bot \\
I \models p \quad \text{iff} \quad I[p] = \top \\
I \not\models p \quad \text{iff} \quad I[p] = \bot
\]

**Inductive Cases:**
\[
I \models \neg F \quad \text{iff} \quad I \not\models F \\
I \models F_1 \wedge F_2 \quad \text{iff} \quad I \models F_1 \quad \text{and} \quad I \models F_2 \\
I \models F_1 \vee F_2 \quad \text{iff} \quad I \models F_1 \quad \text{or} \quad I \models F_2 \\
I \models F_1 \rightarrow F_2
\]
Inductive Definition of Propositional Semantics

**Base Cases:**

\[
\begin{align*}
I & \models \top & I & \not\models \bot \\
I & \models p & \text{iff} & I[p] = \top \\
I & \not\models p & \text{iff} & I[p] = \bot
\end{align*}
\]

**Inductive Cases:**

\[
\begin{align*}
I & \models \neg F & \text{iff} & I \not\models F \\
I & \models F_1 \land F_2 & \text{iff} & I \models F_1 \ \text{and} \ I \models F_2 \\
I & \models F_1 \lor F_2 & \text{iff} & I \models F_1 \ \text{or} \ I \models F_2 \\
I & \models F_1 \rightarrow F_2 & \text{iff, } & I \not\models F_1 \ \text{or} \ I \models F_2
\end{align*}
\]
Inductive Definition of Propositional Semantics

**Base Cases:**

\[ I \models \top \quad I \not\models \bot \]

\[ I \models p \quad \text{iff} \quad I[p] = \top \]

\[ I \not\models p \quad \text{iff} \quad I[p] = \bot \]

**Inductive Cases:**

\[ I \models \neg F \quad \text{iff} \quad I \not\models F \]

\[ I \models F_1 \land F_2 \quad \text{iff} \quad I \models F_1 \text{ and } I \models F_2 \]

\[ I \models F_1 \lor F_2 \quad \text{iff} \quad I \models F_1 \text{ or } I \models F_2 \]

\[ I \models F_1 \rightarrow F_2 \quad \text{iff, } I \not\models F_1 \text{ or } I \models F_2 \]

\[ I \models F_1 \leftrightarrow F_2 \]
Inductive Definition of Propositional Semantics

**Base Cases:**

\[ I \models \top \quad I \not\models \bot \]

\[ I \models p \iff I[p] = \top \]

\[ I \not\models p \iff I[p] = \bot \]

**Inductive Cases:**

\[ I \models \neg F \iff I \not\models F \]

\[ I \models F_1 \land F_2 \iff I \models F_1 \text{ and } I \models F_2 \]

\[ I \models F_1 \lor F_2 \iff I \models F_1 \text{ or } I \models F_2 \]

\[ I \models F_1 \rightarrow F_2 \iff, I \not\models F_1 \text{ or } I \models F_2 \]

\[ I \models F_1 \leftrightarrow F_2 \iff, I \models F_1 \text{ and } I \models F_2 \]
Inductive Definition of Propositional Semantics

**Base Cases:**
\[ I \models \top \quad I \not\models \bot \]
\[ I \models p \quad \text{iff} \quad I[p] = \top \]
\[ I \not\models p \quad \text{iff} \quad I[p] = \bot \]

**Inductive Cases:**
\[ I \models \neg F \quad \text{iff} \quad I \not\models F \]
\[ I \models F_1 \land F_2 \quad \text{iff} \quad I \models F_1 \text{ and } I \models F_2 \]
\[ I \models F_1 \lor F_2 \quad \text{iff} \quad I \models F_1 \text{ or } I \models F_2 \]
\[ I \models F_1 \rightarrow F_2 \quad \text{iff, } I \not\models F_1 \text{ or } I \models F_2 \]
\[ I \models F_1 \leftrightarrow F_2 \quad \text{iff, } I \models F_1 \text{ and } I \models F_2 \]
\[ \quad \text{or } I \not\models F_1 \text{ and } I \not\models F_2 \]
Simple Example

\[ F : (p \land q) \rightarrow (p \lor \neg q) \]

\[ I : \{ p \mapsto \top, \ q \mapsto \bot \} \]
Simple Example

\[ F : (p \land q) \rightarrow (p \lor \neg q) \]
\[ I : \{ p \mapsto \top, \quad q \mapsto \bot \} \]

1. \[ I \models p \]
Simple Example

\[ F : (p \land q) \rightarrow (p \lor \neg q) \]

\[ I : \{ p \mapsto \top, \ q \mapsto \bot \} \]

1. \( I \models p \) since \( I[p] = \top \)
Simple Example

\[ F : (p \land q) \rightarrow (p \lor \neg q) \]

\[ I : \{ p \mapsto \top, \ q \mapsto \bot \} \]

1. \( I \models p \) since \( I[p] = \top \)
2. \( I \models ? q \)
Simple Example

\[ F : (p \land q) \rightarrow (p \lor \neg q) \]

\[ I : \{ p \leftrightarrow \top, \ q \leftrightarrow \bot \} \]

1. \( I \models p \) \quad \text{since} \quad I[p] = \top
2. \( I \not\models q \) \quad \text{since} \quad I[q] = \bot

Thus, \( F \) is true under \( I \).
Simple Example

\[ F : (p \land q) \rightarrow (p \lor \neg q) \]
\[ I : \{ p \mapsto \top, \ q \mapsto \bot \} \]

1. \( I \models p \) \quad \text{since} \ I[p] = \top
2. \( I \not\models q \) \quad \text{since} \ I[q] = \bot
3. \( I \models ? \) \( \neg q \)
Simple Example

\[ F : (p \land q) \rightarrow (p \lor \neg q) \]

\[ I : \{ p \mapsto \top, \ q \mapsto \bot \} \]

1. \( I \models p \) since \( I[p] = \top \)
2. \( I \not\models q \) since \( I[q] = \bot \)
3. \( I \models \neg q \) by 2 and \( \neg \)
Simple Example

\[ F : (p \land q) \rightarrow (p \lor \neg q) \]
\[ I : \{ p \mapsto \top, q \mapsto \bot \} \]

1. \[ I \models p \quad \text{since } I[p] = \top \]
2. \[ I \not\models q \quad \text{since } I[q] = \bot \]
3. \[ I \models \neg q \quad \text{by 2 and } \neg \]
4. \[ I \models ? \quad p \land q \]
Simple Example

\[ F : (p \land q) \rightarrow (p \lor \neg q) \]

\[ I : \{ p \mapsto \top, \ q \mapsto \bot \} \]

1. \( I \models p \) since \( I[p] = \top \)
2. \( I \not\models q \) since \( I[q] = \bot \)
3. \( I \models \neg q \) by 2 and \( \neg \)
4. \( I \not\models p \land q \) by 2 and \( \land \)
Simple Example

\[ F : (p \land q) \rightarrow (p \lor \neg q) \]

\[ I : \{ p \mapsto \top, \ q \mapsto \bot \} \]

1. \( I \models p \) since \( I[p] = \top \)
2. \( I \not\models q \) since \( I[q] = \bot \)
3. \( I \models \neg q \) by 2 and \( \neg \)
4. \( I \not\models p \land q \) by 2 and \( \land \)
5. \( I \models? p \lor \neg q \)
Simple Example

\[ F : (p \land q) \to (p \lor \neg q) \]

\[ I : \{ p \mapsto \top, \ q \mapsto \bot \} \]

1. \( I \models p \) since \( I[p] = \top \)
2. \( I \not\models q \) since \( I[q] = \bot \)
3. \( I \models \neg q \) by 2 and \( \neg \)
4. \( I \not\models p \land q \) by 2 and \( \land \)
5. \( I \models p \lor \neg q \) by 1 and \( \lor \)
Simple Example

\[ F : (p \land q) \rightarrow (p \lor \neg q) \]

\[ I : \{ p \mapsto \top, \ q \mapsto \bot \} \]

1. \( I \models p \) since \( I[p] = \top \)
2. \( I \not\models q \) since \( I[q] = \bot \)
3. \( I \models \neg q \) by 2 and \( \neg \)
4. \( I \not\models p \land q \) by 2 and \( \land \)
5. \( I \models p \lor \neg q \) by 1 and \( \lor \)
6. \( I \models? \ F \)
Simple Example

\[ F : (p \land q) \rightarrow (p \lor \neg q) \]

\[ I : \{ p \mapsto \top, \ q \mapsto \bot \} \]

1. \( I \models p \) since \( I[p] = \top \)
2. \( I \not\models q \) since \( I[q] = \bot \)
3. \( I \models \neg q \) by 2 and \( \neg \)
4. \( I \not\models p \land q \) by 2 and \( \land \)
5. \( I \models p \lor \neg q \) by 1 and \( \lor \)
6. \( I \models F \) by 4 and \( \rightarrow \)
Simple Example

$$F : (p \land q) \rightarrow (p \lor \neg q)$$

$$I : \{ p \mapsto \top, \quad q \mapsto \bot \}$$

1. $I \models p$ since $I[p] = \top$
2. $I \not\models q$ since $I[q] = \bot$
3. $I \models \neg q$ by 2 and $\neg$
4. $I \not\models p \land q$ by 2 and $\land$
5. $I \models p \lor \neg q$ by 1 and $\lor$
6. $I \models F$ by 4 and $\rightarrow$

Thus, $F$ is true under $I$. 
Another Example

What does the formula

\[ F : (p \leftrightarrow \neg q) \rightarrow (q \rightarrow \neg r) \]

evaluate to under this interpretation?

\[ I = \{ p \mapsto \bot, \ q \mapsto \top, \ r \mapsto \top \} \]
Another Example

- What does the formula
  \[ F : (p \iff \neg q) \to (q \to \neg r) \]
evaluate to under this interpretation?
  \[ I = \{ p \mapsto \bot, \; q \mapsto \top, \; r \mapsto \top \} \]

- \( I \not\models F \)
Satisfiability and Validity

- \( F \) is **satisfiable** iff there exists an interpretation \( I \) such that \( I \models F \).
- \( F \) is **valid** iff for all interpretations \( I \), \( I \models F \).
- \( F \) is **contingent** if it is satisfiable but not valid.
- Duality between satisfiability and validity: \( F \) is valid iff \( \neg F \) is unsatisfiable.

Thus, if we have a procedure for checking satisfiability, this also allows us to decide validity.
Satisfiability and Validity

- $F$ is **satisfiable** iff there exists an interpretation $I$ such that $I \models F$.

- $F$ **valid** iff for all interpretations $I$, $I \models F$.
Satisfiability and Validity

- **$F$ is satisfiable** iff there exists an interpretation $I$ such that $I \models F$.
- **$F$ valid** iff for all interpretations $I$, $I \models F$.
- **$F$ is contingent** if it is satisfiable but not valid.
Satisfiability and Validity

- \( F \) is **satisfiable** iff there exists an interpretation \( I \) such that \( I \models F \).

- \( F \) **valid** iff for all interpretations \( I \), \( I \models F \).

- \( F \) is **contingent** if it is satisfiable but not valid.

- Duality between satisfiability and validity:

  \[
  F \text{ is valid iff } \neg F \text{ is unsatisfiable}
  \]
Satisfiability and Validity

- \( F \) is satisfiable iff there exists an interpretation \( I \) such that \( I \models F \).

- \( F \) valid iff for all interpretations \( I \), \( I \models F \).

- \( F \) is contingent if it is satisfiable but not valid.

- Duality between satisfiability and validity:

  \[
  \text{\( F \) is valid iff \( \neg F \) is unsatisfiable}
  \]

- Thus, if we have a procedure for checking satisfiability, this also allows us to decide validity
Before we talk about practical algorithms for deciding satisfiability, let’s review some simple techniques.
Before we talk about practical algorithms for deciding satisfiability, let’s review some simple techniques.

Two very simple techniques:
Before we talk about practical algorithms for deciding satisfiability, let’s review some simple techniques

Two very simple techniques:

- Truth table method: essentially a search-based technique
Deciding Satisfiability and Validity

- Before we talk about practical algorithms for deciding satisfiability, let’s review some simple techniques

- Two very simple techniques:
  - Truth table method: essentially a search-based technique
  - Semantic argument method: deductive way of deciding satisfiability
Before we talk about practical algorithms for deciding satisfiability, let’s review some simple techniques.

Two very simple techniques:

- **Truth table method**: essentially a search-based technique
- **Semantic argument method**: deductive way of deciding satisfiability

Completely different, but complementary techniques
Deciding Satisfiability and Validity

- Before we talk about practical algorithms for deciding satisfiability, let’s review some simple techniques

- Two very simple techniques:
  - Truth table method: essentially a search-based technique
  - Semantic argument method: deductive way of deciding satisfiability

- Completely different, but complementary techniques

- In fact, as we’ll see later, modern SAT solvers combine both search-based and deductive techniques!
Example: \[ F : (p \land q) \rightarrow (p \lor \neg q) \]
## Method 1: Truth Tables

**Example**  
\[ F : (p \land q) \rightarrow (p \lor \neg q) \]

<table>
<thead>
<tr>
<th>(p)</th>
<th>(q)</th>
<th>(p \land q)</th>
<th>(\neg q)</th>
<th>(p \lor \neg q)</th>
<th>(F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Method 1: Truth Tables

Example \( F : (p \land q) \rightarrow (p \lor \neg q) \)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Thus \( F \) is valid.
Another Example

\[ F : (p \lor q) \rightarrow (p \land q) \]

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>p</th>
<th>q</th>
<th>p \lor q</th>
<th>p \land q</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

← satisfying \( I \)
← falsifying \( I \)
Another Example

\[ F : (p \lor q) \rightarrow (p \land q) \]

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \lor q$</th>
<th>$p \land q$</th>
<th>$F$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

← satisfying $I$
← falsifying $I$

Thus $F$ is satisfiable, but invalid.
Summary: Truth Tables

- List all interpretations ⇒ If all interpretations satisfy formula, then valid.
  If no interpretation satisfies it, unsatisfiable.

Completely brute-force, impractical: requires explicitly listing all $2^n$ interpretations in the worst-case!

Method does not work for any logic where domain is not finite (e.g., first-order logic)
Summary: Truth Tables

- List all interpretations $\Rightarrow$ If all interpretations satisfy formula, then valid. If no interpretation satisfies it, unsatisfiable.

- Completely brute-force, impractical: requires explicitly listing all $2^n$ interpretations in the worst-case!
Summary: Truth Tables

- List all interpretations $\Rightarrow$ If all interpretations satisfy formula, then valid. If no interpretation satisfies it, unsatisfiable.

- Completely brute-force, impractical: requires explicitly listing all $2^n$ interpretations in the worst-case!

- Method does not work for any logic where domain is not finite (e.g., first-order logic)
Method 2: Semantic Argument

- Semantic argument method is essentially a proof by contradiction, and is also applicable for theories with non-finite domain.
Method 2: Semantic Argument

- Semantic argument method is essentially a proof by contradiction, and is also applicable for theories with non-finite domain.

- Main idea: Assume $F$ is not valid $\Rightarrow$ there exists some falsifying interpretation $I$ such that $I \not|= F$
Method 2: Semantic Argument

- Semantic argument method is essentially a proof by contradiction, and is also applicable for theories with non-finite domain.

- **Main idea**: Assume $F$ is not valid $\Rightarrow$ there exists some falsifying interpretation $I$ such that $I \nvDash F$

- Apply proof rules.
Method 2: Semantic Argument

- Semantic argument method is essentially a proof by contradiction, and is also applicable for theories with non-finite domain.

- Main idea: Assume $F$ is not valid $\Rightarrow$ there exists some falsifying interpretation $I$ such that $I \not\models F$

- Apply proof rules.

- If we derive a contradiction in every branch of the proof, then $F$ is valid.
Method 2: Semantic Argument

- Semantic argument method is essentially a proof by contradiction, and is also applicable for theories with non-finite domain.

- Main idea: Assume \( F \) is not valid \( \Rightarrow \) there exists some falsifying interpretation \( I \) such that \( I \models \neg F \)

- Apply proof rules.

- If we derive a contradiction in every branch of the proof, then \( F \) is valid.

- If there exists some branch where we cannot derive a contradiction (after exhaustively applying all proof rules), then \( F \) is not valid.
The Proof Rules (I)

- According to semantics of negation, from $I \models \neg F$, we can deduce $I \not\models F$:

$$
\begin{align*}
I & \models \neg F \\
\therefore & I \not\models F
\end{align*}
$$
According to semantics of negation, from $I \models \neg F$, we can deduce $I \not\models F$:

$$
rac{I \models \neg F}{I \not\models F}
$$

Similarly, from $I \not\models \neg F$, we can deduce:
According to semantics of negation, from $I \models \neg F$, we can deduce $I \not\models F$:

$$
\frac{I \models \neg F}{I \not\models F}
$$

Similarly, from $I \not\models \neg F$, we can deduce:

$$
\frac{I \not\models \neg F}{I \models F}
$$
According to semantics of conjunction, from $I \models F \land G$, we can deduce:

$$
\begin{align*}
I & \models F \land G \\
\therefore I & \models F \\
\therefore I & \models G
\end{align*}
$$

The second deduction results in a branch in the proof, so each case has to be examined separately!
According to semantics of conjunction, from \( I \models F \land G \), we can deduce:

\[
\begin{align*}
  & I \models F \land G \\
  & I \models F \\
  & I \models G \quad \text{← and}
\end{align*}
\]

Similarly, from \( I \not\models F \land G \), we can deduce:
According to semantics of conjunction, from \( I \models F \land G \), we can deduce:

\[
\begin{align*}
I & \models F \land G \\
I & \models F \\
I & \models G
\end{align*}
\]

Similarly, from \( I \not\models F \land G \), we can deduce:

\[
\begin{align*}
I & \not\models F \land G \\
I & \not\models F \\
I & \not\models G
\end{align*}
\]
According to semantics of conjunction, from $I \models F \land G$, we can deduce:

$$
\begin{align*}
I & \models F \land G \\
\quad & \implies I \models F \\
\quad & \implies I \models G
\end{align*}
$$

Similarly, from $I \not\models F \land G$, we can deduce:

$$
\begin{align*}
I & \not\models F \land G \\
\quad & \implies I \not\models F \\
\quad & \implies I \not\models G
\end{align*}
$$

The second deduction results in a branch in the proof, so each case has to be examined separately!
According to semantics of disjunction, from $I \models F \lor G$, we can deduce:

\[
\frac{I \models F \lor G}{I \models F \quad I \models G}
\]
According to semantics of disjunction, from \( I \models F \lor G \), we can deduce:

\[
\begin{align*}
I & \models F \\
I & \models G
\end{align*}
\]

Similarly, from \( I \not\models F \lor G \), we can deduce:
According to semantics of disjunction, from \( I \models F \lor G \), we can deduce:

\[
\frac{I \models F \lor G}{I \models F \quad I \models G}
\]

Similarly, from \( I \not\models F \lor G \), we can deduce:

\[
\frac{I \not\models F \lor G}{I \not\models F \quad I \not\models G}
\]
The Proof Rules (IV)

- According to semantics of implication:

\[ I \models F \rightarrow G \]
According to semantics of implication:

\[
\frac{\frac{I \models F \rightarrow G}{I \not\models F} \quad I \models G}{I \not\models F} \quad I \models G
\]
The Proof Rules (IV)

According to semantics of implication:

\[
\begin{align*}
I \models F \rightarrow G \\
\neg I \models F \mid I \models G
\end{align*}
\]

And:

\[
\neg I \not\models F \rightarrow G
\]
According to semantics of implication:

\[ \frac{I \models F \rightarrow G}{I \not\models F \mid I \models G} \]

And:

\[ \frac{I \not\models F \rightarrow G}{I \models F} \]

\[ I \not\models G \]
The Proof Rules (V)

- According to semantics of iff:

\[ I \models F \leftrightarrow G \]
The Proof Rules (V)

- According to semantics of iff:

\[
\frac{I \models F \leftrightarrow G}{I \models F \land G}
\]
The Proof Rules (V)

- According to semantics of iff:

\[
\begin{align*}
I & \models F \leftrightarrow G \\
I & \models F \land G & I & \models \neg F \land \neg G
\end{align*}
\]
According to semantics of iff:

\[
\begin{align*}
  I & \models F \leftrightarrow G \\
  I \models F \land G & \quad | \\
  I \models \neg F \land \neg G
\end{align*}
\]

And:

\[
I \not\models F \leftrightarrow G
\]
The Proof Rules (V)

- According to semantics of iff:

\[
\begin{align*}
I \models F \leftrightarrow G & \quad \text{implies} \quad I \models F \land G \quad \text{and} \quad I \models \neg F \land \neg G.
\end{align*}
\]

- And:

\[
\begin{align*}
I \nvdash F \leftrightarrow G & \quad \text{implies} \quad I \models F \land \neg G \quad \text{and} \quad I \models \neg F \land G.
\end{align*}
\]
Finally, we derive a contradiction, when $I$ both entails $F$ and does not entail $F$:

\[
\begin{align*}
I & \models F \\
I & \not\models F \\
I & \models \bot
\end{align*}
\]
An Example

Prove  \( F : (p \land q) \rightarrow (p \lor \neg q) \) is valid.
An Example

Prove $F : (p \land q) \rightarrow (p \lor \neg q)$ is valid.

Let’s assume that $F$ is not valid and that $I$ is a falsifying interpretation.
An Example

Prove \( F : (p \land q) \to (p \lor \neg q) \) is valid.

Let’s assume that \( F \) is not valid and that \( I \) is a falsifying interpretation.

1. \( I \not\models (p \land q) \to (p \lor \neg q) \) assumption
An Example

Prove \( F : (p \land q) \rightarrow (p \lor \lnot q) \) is valid.

Let’s assume that \( F \) is not valid and that \( I \) is a falsifying interpretation.

1. \( I \not\models (p \land q) \rightarrow (p \lor \lnot q) \) assumption
2. \( I \models p \land q \) 1 and \( \rightarrow \)
3. \( I \not\models p \lor \lnot q \) 1 and \( \rightarrow \)

Thus \( F \) is valid.
An Example

Prove $F : (p \land q) \rightarrow (p \lor \neg q)$ is valid.

Let’s assume that $F$ is not valid and that $I$ is a falsifying interpretation.

1. $I \nvdash (p \land q) \rightarrow (p \lor \neg q)$ assumption
2. $I \models p \land q$ 1 and $\rightarrow$
3. $I \nvdash p \lor \neg q$ 1 and $\rightarrow$
4. $I \models p$ 2 and $\land$
5. $I \models q$ 2 and $\land$
An Example

Prove \( F : (p \land q) \rightarrow (p \lor \neg q) \) is valid.

Let’s assume that \( F \) is not valid and that \( I \) is a falsifying interpretation.

1. \( I \not \models (p \land q) \rightarrow (p \lor \neg q) \)  assumption
2. \( I \models p \land q \)  1 and \( \rightarrow \)
3. \( I \not \models p \lor \neg q \)  1 and \( \rightarrow \)
4. \( I \models p \)  2 and \( \land \)
5. \( I \models q \)  2 and \( \land \)
6. \( I \not \models p \)  3 and \( \lor \)
7. \( I \not \models \neg q \)  3 and \( \lor \)

\( \therefore \) Thus \( F \) is valid.
An Example

Prove \( F : (p \land q) \rightarrow (p \lor \neg q) \) is valid.

Let’s assume that \( F \) is not valid and that \( I \) is a falsifying interpretation.

1. \( I \not\models (p \land q) \rightarrow (p \lor \neg q) \) \hspace{0.5cm} \text{assumption}
2. \( I \models p \land q \) \hspace{1.5cm} 1 \text{ and } \rightarrow
3. \( I \not\models p \lor \neg q \) \hspace{1.5cm} 1 \text{ and } \rightarrow
4. \( I \models p \) \hspace{1.5cm} 2 \text{ and } \land
5. \( I \models q \) \hspace{1.5cm} 2 \text{ and } \land
6. \( I \not\models p \) \hspace{1.5cm} 3 \text{ and } \lor
7. \( I \not\models \neg q \) \hspace{1.5cm} 3 \text{ and } \lor
8. \( I \models \bot \) \hspace{1.5cm} 4 \text{ and } 6 \text{ are contradictory}
An Example

Prove \( F : (p \land q) \rightarrow (p \lor \neg q) \) is valid.

Let’s assume that \( F \) is not valid and that \( I \) is a falsifying interpretation.

1. \( I \not\models (p \land q) \rightarrow (p \lor \neg q) \) assumption
2. \( I \models p \land q \) 1 and \( \rightarrow \)
3. \( I \not\models p \lor \neg q \) 1 and \( \rightarrow \)
4. \( I \models p \) 2 and \( \land \)
5. \( I \models q \) 2 and \( \land \)
6. \( I \not\models p \) 3 and \( \lor \)
7. \( I \not\models \neg q \) 3 and \( \lor \)
8. \( I \models \bot \) 4 and 6 are contradictory

\( \Rightarrow \) Thus \( F \) is valid.
Another Example

- Prove that the following formula is valid using semantic argument method:

\[ F : ((p \rightarrow q) \land (q \rightarrow r)) \rightarrow (p \rightarrow r) \]
Formulas $F_1$ and $F_2$ are equivalent (written $F_1 \iff F_2$) iff for all interpretations $I$, $I \models F_1 \iff F_2$

$F_1 \iff F_2$ iff $F_1 \iff F_2$ is valid
Formulas $F_1$ and $F_2$ are equivalent (written $F_1 \iff F_2$) iff for all interpretations $I$, $I \models F_1 \leftrightarrow F_2$

- Thus, if we have a procedure for checking satisfiability, we can also check equivalence.
Implication

- Formula $F_1$ implies $F_2$ (written $F_1 \implies F_2$) iff for all interpretations $I$, $I \models F_1 \rightarrow F_2$

\[ F_1 \implies F_2 \text{ iff } F_1 \rightarrow F_2 \text{ is valid} \]
**Implication**

- Formula $F_1$ implies $F_2$ (written $F_1 \Rightarrow F_2$) iff for all interpretations $I$,
  
  \[ I \models F_1 \rightarrow F_2 \]

- Thus, if we have a procedure for checking satisfiability, we can also check implication.

  \[ F_1 \Rightarrow F_2 \text{ iff } F_1 \rightarrow F_2 \text{ is valid} \]
Implication

- **Formula** \( F_1 \) **implies** \( F_2 \) (written \( F_1 \Rightarrow F_2 \)) iff for all interpretations \( I \),
  \[ I \models F_1 \rightarrow F_2 \]

\[
F_1 \Rightarrow F_2 \text{ iff } F_1 \rightarrow F_2 \text{ is valid}
\]

- Thus, if we have a procedure for checking satisfiability, we can also check implication

- **Caveat:** \( F_1 \Leftrightarrow F_2 \) and \( F_1 \Rightarrow F_2 \) are not formulas (they are not part of PL syntax); they are semantic judgments!
Example

- Prove that $F_1 \land (\neg F_1 \lor F_2)$ implies $F_2$ using semantic argument method.
Today:

Review of basic concepts underlying propositional logic
Summary

▶ Today:

Review of basic concepts underlying propositional logic

▶ Next lecture:

Normal forms and algorithms for deciding satisfiability
Summary

▶ Today:

Review of basic concepts underlying propositional logic

▶ Next lecture:

Normal forms and algorithms for deciding satisfiability

▶ Reading:

Bradley & Manna textbook until Section 1.6