Course staff

- **Instructor**: İsil Dillig
- **E-mail**: isil@cs.utexas.edu
- **Office hours**: Thursday 3-4 pm
- **TA**: Shankara Pailoor (spailoor@cs.utexas.edu)
- **Office hours**: Monday 4-5 pm

What is this Course About?

- This course is about **computational logic** and its applications in reasoning about software correctness.
- Explore logical theories widely used in computer science.
- Learn about **decision procedures** that allow us to automatically decide satisfiability and validity of logical formulas.

Why Should You Care?

Logic is a fundamental part of computer science:

- **Artificial intelligence**: planning, automated game playing, ...
- **Programming languages**: Static analysis, software verification, program synthesis, ...
- **Software engineering**: automated test generation, automated program repair, ...

Overview of the Course

- **Part I: Propositional logic**
  - SAT solvers
  - Applications and variations (e.g., MaxSAT)
  - Binary Decision Diagrams

Overview, cont

- **Part II: First-order theorem proving**
  - Semantics of FOL and theoretical properties
  - Basics of first-order theorem proving
  - Decidable fragments of FOL
Overview, cont.

▶ Part III: SMT Solving
  ▶ Decision procedures for commonly used theories (e.g., equality, linear arithmetic)
  ▶ Combining theories, Nelson-Oppen method
  ▶ DPLL(T) and practical SMT solvers

Overview, cont.

▶ Part IV: Applications in formal methods
  ▶ Program verification
  ▶ Program synthesis

Logistics

▶ All class material (slides, relevant reading etc.) posted on the course website:
  http://www.cs.utexas.edu/~idillig/cs389L
▶ Also have a Piazza page: piazza.com/utexas/spring2021/cs389L
▶ Please post all non-personal questions on Piazza instead of emailing us!

Optional Reference #1

▶ The Calculus of Computation
  by Aaron Bradley and Zohar Manna

Optional Reference #2

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

Workload and Grading

▶ No exams or big projects
▶ Combination of problem sets and programming assignments
▶ Collaboration on homeworks is not allowed
▶ You can have 2 day “late days” total that you can use throughout the semester
Exams

- **Exam dates**: February 28, May 2 – put these dates on your calendar! (free during finals week)
- All exams closed-book, closed-notes, closed-laptop, closed-phone etc, but can bring 3 cheat sheets
- Please introduce yourself!

Let’s get started!

- **Today**: Review of basic propositional logic
- Should already know this stuff – quick refresher!

### Review of Propositional Logic: PL Syntax

- **Atom**: truth symbols \( \top \) ("true") and \( \perp \) ("false")
- **Propositional variables**: \( p, q, r, p_1, q_1, r_1, \ldots \)
- **Literal**: \( \alpha \) 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)
  - \( F_1 \iff F_2 \): "if and only if" (iff)

### Inductive Definition of PL Semantics

#### Base Cases:

- \( I \models \top \)
- \( I \nmodels \perp \)
- \( I \models \alpha \iff I[\alpha] = \top \)
- \( I \nmodels \alpha \iff I[\alpha] = \perp \)

#### Inductive Cases:

- \( I \models \neg F \iff I \nmodels F \)
- \( I \models F_1 \land F_2 \iff I \models F_1 \text{ and } I \models F_2 \)
- \( I \models F_1 \lor F_2 \iff I \models F_1 \text{ or } I \models F_2 \)
- \( I \models F_1 \Rightarrow F_2 \iff I \models F_1 \text{ and } I \nmodels F_2 \)
- \( I \models F_1 \iff F_2 \iff I \models F_1 \text{ and } I \nmodels F_2 \)

### PL Semantics

- **Interpretation** \( I \): mapping from each propositional variables in \( F \) to exactly one truth value
  \[ I : \{ p \mapsto \top, q \mapsto \perp, \cdots \} \]
- **Formula** \( F \) + Interpretation \( I = \text{Truth value} \)

- We write \( I \models F \) if \( F \) evaluates to \( \top \) under \( I \) (satisfying interpretation or model)
- Similarly, \( I \nmodels F \) if \( F \) evaluates to \( \perp \) under \( I \) (falsifying interpretation or counter-model).

### Simple Example

- Consider formula \( F_1 : (p \land q) \rightarrow (p \lor \neg q) \)
- What is its truth value under interpretation \( I_1 : \{ p \mapsto \top, q \mapsto \bot \} \)?
- What about formula \( F_2 : (p \leftrightarrow \neg q) \rightarrow (q \rightarrow \neg r) \) and interpretation \( I_2 = \{ p \mapsto \bot, q \mapsto \top, r \mapsto \top \} \)?
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:**
  
  \[ F \text{ is valid if and only if } \neg F \text{ is unsatisfiable} \]

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

Examples

- Sat, unsat, or valid?
  
  \[ (p \land q) \rightarrow \neg p \]
  
  \[ (p \rightarrow q) \land \neg (p \land \neg q) \]
  
  \[ (p \rightarrow (q \rightarrow r)) \land \neg ((p \land q) \rightarrow r) \]

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>

Thus \( F \) is valid.

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>

Thus \( F \) is satisfiable, but invalid.

Bad Idea!

- Truth tables are completely brute-force, impractical \( \Rightarrow \) must list all \( 2^n \) interpretations!
- Does not work for any other 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.
- **Main idea**: Assume $F$ is not valid $\Rightarrow$ there exists some falsifying interpretation $I$ such that $I \not|= F$
- Apply **proof rules**.
- If we derive a contradiction in every branch of the proof, then $F$ is valid.

The Proof Rules (I)

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

The Proof Rules (II)

- According to semantics of conjunction, from $I |= F \land G$, we can deduce:
  $$I |= F \land G$$  $$I |= F$$  $$I |= G$$
- Similarly, from $I \not|= F \land G$, we can deduce:
  $$I \not|= F \land G$$  $$I \not|= F$$  $$I \not|= G$$
- The second deduction results in a branch in the proof, so each case has to be examined separately!

The Proof Rules (IV)

- According to semantics of implication:
  $$I |= F \rightarrow G$$  $$I \not|= F \mid I |= G$$
- And:
  $$I \not|= F \rightarrow G$$  $$I |= F$$  $$I \not|= G$$

The Proof Rules (V)

- According to semantics of iff:
  $$I |= F \leftrightarrow G$$  $$I |= F \land \neg G \mid I |= \neg F \land G$$
- And:
  $$I \not|= F \leftrightarrow G$$  $$I |= F \land \neg G$$  $$I |= \neg F \land G$$
The Proof Rules (Contradiction)

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

Another Example

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

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

Equivalence

- Formulas \( F_1 \) and \( F_2 \) are equivalent (written \( F_1 \Leftrightarrow F_2 \)) iff for all interpretations \( I \), \( I \models F_1 \Leftrightarrow F_2 \)

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

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

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 \( F_1 \land (\lnot F_1 \lor F_2) \) implies \( F_2 \) using semantic argument method.
Summary

- Next lecture:
  Normal forms and algorithms for deciding satisfiability

- Optional reading:
  Bradley & Manna textbook until Section 1.6