CS389L: Automated Logical Reasoning

Lecture 1: Introduction and Review of Basics

Işıl Dillig

Course staff

- Instructor: Işıl Dillig
- E-mail: isil@cs.utexas.edu
- Office hours: Thursday after class until 6:30 pm in GDC 5.726
- TA: Kostas Ferles (kferles@cs.utexas.edu)
- Office hours: Wed 1-2:30 pm in GDC 5.710B

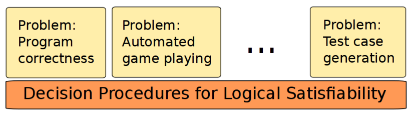
What is this Course About?

- 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
  - Hoare Logic
  - Automated verification using SMT
  - Invariant inference
  - Predicate abstraction

Logistics

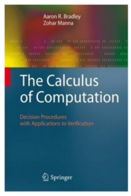
- All class material (slides, relevant reading etc.) posted on the course website:

  http://www.cs.utexas.edu/~idillig/cs389L

- Also have a Piazza page: piazza.com/utexas/spring2019/cs389l
- Please post all non-personal questions on Piazza instead of emailing us!

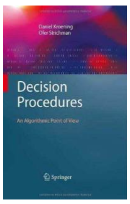
Optional Reference #1

- 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

- My philosophy: Maximize your knowledge without taking too much time
- No programming assignments or big projects
- Expect problem set once every 2 weeks (35% grade)
- 2 in-class, closed-book exams (30% of grade each)
- Class attendance and participation (5% of grade)

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 $\bot$ ("false")
propositional variables $p, q, r, p_1, q_1, r_1, \cdots$

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

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

$\neg F$	"not"	(negation)
$F_1 \wedge F_2$	"and"	(conjunction)
$F_1 \vee F_2$	"or"	(disjunction)
$F_1 \rightarrow F_2$	"implies"	(implication)
$F_1 \leftrightarrow F_2$	"if and only if"	(iff)

PL Semantics

- Interpretation I: mapping from each propositional variables in F to exactly one truth value

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

- Formula F + Interpretation I = Truth value
- We write $I \models F$ if F evaluates to $\top$ under I (satisfying interpretation or model)
- Similarly, $I \not\models F$ if F evaluates to $\bot$ under I (falsifying interpretation or counter-model).

Inductive Definition of PL Semantics

Base Cases:

$I \models \top \qquad I \not\models \bot$

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

Inductive Cases:

$I \models \neg F$ iff $I \not\models F$
$I \models F_1 \wedge F_2$ iff $I \models F_1$ and $I \models F_2$
$I \models F_1 \vee F_2$ iff $I \models F_1$ or $I \models F_2$
$I \models F_1 \rightarrow F_2$ iff, $I \not\models F_1$ or $I \models F_2$
$I \models F_1 \leftrightarrow F_2$ iff, $I \models F_1$ and $I \models F_2$ or $I \not\models F_1$ and $I \not\models F_2$

Simple Example

- Consider formula $F_1 : (p \wedge q) \rightarrow (p \vee \neg q)$
- 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 valid iff for all interpretations I, $I \models F$.
- F is contingent if it is satisfiable but not valid.
- Duality between satisfiability and validity:

$$\boxed{F \text{ is valid iff } \neg F \text{ is unsatisfiable}}$$

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

Examples

- Sat, unsat, or valid?
  - $(p \wedge q) \rightarrow \neg p$
  - $(p \rightarrow q) \rightarrow (\neg(p \wedge \neg q))$
  - $(p \rightarrow (q \rightarrow r)) \wedge \neg((p \wedge q) \rightarrow r)$

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
- Modern SAT solvers combine search and deduction!

Method 1: Truth Tables

Example $F : (p \wedge q) \rightarrow (p \vee \neg q)$

p q	$p \wedge q$	$\neg q$	$p \vee \neg q$	F
0 0	0	1	1	1
0 1	0	0	0	1
1 0	0	1	1	1
1 1	1	0	1	1

Thus F is valid.

Another Example

$F : (p \vee q) \rightarrow (p \wedge q)$

p q	$p \vee q$	$p \wedge q$	F	
0 0	0	0	1	$\leftarrow$ satisfying I
0 1	1	0	0	$\leftarrow$ falsifying I
1 0	1	0	0	
1 1	1	1	1	

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 $I \not\models F$
- 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 $I \models \neg F$, 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:

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

The Proof Rules (II)

- According to semantics of conjunction, from $I \models F \wedge G$, we can deduce:

$$\begin{array}{l} I \models F \wedge G \\ \hline I \models F \\ I \models G \end{array} \leftarrow\text{and}$$

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

$$\frac{I \not\models F \wedge G}{I \not\models F \mid I \not\models G}$$

- The second deduction results in a branch in the proof, so each case has to be examined separately!

The Proof Rules (III)

- According to semantics of disjunction, from $I \models F \vee G$, we can deduce:

$$\frac{I \models F \vee G}{I \models F \mid I \models G}$$

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

$$\begin{array}{l} I \not\models F \vee G \\ \hline I \not\models F \\ I \not\models G \end{array}$$

The Proof Rules (IV)

- According to semantics of implication:

$$\frac{I \models F \rightarrow G}{I \not\models F \mid I \models G}$$

- And:

$$\begin{array}{l} I \not\models F \rightarrow G \\ \hline I \models F \\ I \not\models G \end{array}$$

The Proof Rules (V)

- According to semantics of iff:

$$\frac{I \models F \leftrightarrow G}{I \models F \wedge G \mid I \models \neg F \wedge \neg G}$$

- And:

$$\frac{I \not\models F \leftrightarrow G}{I \models F \wedge \neg G \mid I \models \neg F \wedge G}$$

The Proof Rules (Contradiction)

- Finally, we derive a contradiction, when I both entails F and does not entail F:

$$\frac{\begin{array}{c} I \models F \\ I \not\models F \end{array}}{I \models \bot}$$

An Example

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

Another Example

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

$$F: \quad ((p \rightarrow q) \land (q \rightarrow r)) \rightarrow (p \rightarrow r)$$

Equivalence

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

$$\boxed{F_1 \Leftrightarrow F_2 \text{ iff } F_1 \leftrightarrow F_2 \text{ is valid}}$$

- 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, $I \models F_1 \rightarrow F_2$

$$\boxed{F_1 \Rightarrow F_2 \text{ iff } F_1 \rightarrow F_2 \text{ is valid}}$$

- 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.

Summary

- Next lecture:

  Normal forms and algorithms for deciding satisfiability

- Optional reading:

  Bradley & Manna texbook until Section 1.6