אמות אוטומטי

תרגול 8

Satisfiability Modulo Theories (SMT)

based on slides by Ofer Strichman
Satisfiability

• The classic SAT problem: given a propositional formula \( \varphi \), is \( \varphi \) satisfiable?

• Example:
  
  – Let \( x_1, x_2 \) be propositional variables
  
  – Let \( \varphi := (x_1 \rightarrow x_2) \land (x_1 \land \neg \neg x_2) \)
  
  – Is \( \varphi \) satisfiable?
Satisfiability Modulo Theories

• Now let the predicates be equalities

• Example:
  
  – Let $x_1, x_2, x_3 \in \mathbb{Z}$
  
  – Let $\varphi := ((x_1 = x_2) \rightarrow (x_2 = x_3)) \land x_1 \neq x_3$
  
  – Is $\varphi$ satisfiable?
Satisfiability Modulo Theories

• Now let the predicates be from linear arithmetic

• Example:
  - Let $x_1, x_2, x_3 \in \mathbb{R}$
  - Let $\varphi \equiv ((x_1 + 2x_3 < 5) \lor \neg(x_3 \leq 1) \land (x \geq 3))$
  - Is $\varphi$ satisfiable?
Satisfiability Modulo Theories

• Now let the predicates represent arrays

• Example:
  
  – Let $i, j, a[\ ] \in \mathbb{Z}$
  
  – Let $\varphi := (i = j \land a[j] = 1) \land a[i] \neq 1$

  – Is $\varphi$ satisfiable?
Generalization

• Equalities, linear predicates, arrays, ...
• Is there a general framework to define them ?

• Yes ! It is called first-order logic

• Each of the above examples is a first-order theory.
What lies ahead...

• As in the case of propositional logic, we need to define
  – Syntax
  – Semantics

• But it is more difficult, because:
  – First-order logic is a template for defining theories
  – Semantics cannot be defined with truth-tables
    • The domains can be infinite!
First order logic

• We distinguish between
  – Logical symbols: $\lor \land \neg \forall \exists ()$
  – Non-logical Symbols $\Sigma$:
    • function symbols (incl. constants)
    • predicate symbols

• $\Sigma$ is the ‘signature’ of the theory
  – $\Sigma$ restricts the syntax
Example 1

• $\Sigma = \{0,1,f,p\}$
  – ‘0’, ‘1’ are constant symbols
  – ‘f’ is a binary function symbol
  – ‘p’ is a binary predicate

• An example of a $\Sigma$-formula:

  \[ \exists y \forall x. p(x, f(y)) \]

• Is it true?
  – depends on the interpretation we give to $p$ and $f$.
  – Does there exists a satisfying interpretation?
An interpretation is given by a structure

A structure is given by:

1. A **domain** to each variable

2. An **interpretation** of the nonlogical symbols: i.e.,
   - Maps each **function symbol** to a function of the same arity
   - Maps each **constant symbol** to a domain element
   - Maps each **predicate symbol** to a predicate of the same arity

3. An **assignment** of a domain element to each free (unquantified) variable
Structures

• \( \phi = \exists x. f(x, 0) = 1 \)

• Consider the structure S:
  – Domain: \( N_0 \)
  – Interpretation:
    • ‘f’ \( \mapsto * \) (multiplication)
    • ‘0’ and ‘1’ are mapped to 0 and 1 in \( N_0 \)
    • ‘=’ \( \mapsto = \) (equality)

• Now, is \( \phi \) true in S?
Satisfying structures

• **Definition**: A formula is satisfiable if there exists a structure that satisfies it

• **Example**: $\phi = \exists x. f(x, 0) = 1$ is satisfiable

• Consider the structure $S'$:
  - Domain: $\mathbb{N}_0$
  - Interpretation:
    • ‘0’ and ‘1’ are mapped to 0 and 1 in $\mathbb{N}_0$
    • ‘=’ $\mapsto =$ (equality)
    • ‘f’ $\mapsto +$ (addition)

• $\phi$ is satisfied by $S'$. $S'$ is a **model** of $\phi$. 
Two ways to define a theory

• We can define a theory

1. The easy, “direct” way,
   ■ Define the grammar,
   ■ Assume standard interpretation of the symbols.

2. The hard, “formal” way.
   ■ Define the signature
   ■ Restrict possible interpretations with sentences/axioms

Very useful for defining algorithms and proving general theorems
1. Defining a theory: the easy way

• Example: The quantifier-free theory of *equalities*

• Grammar: A Boolean combination of predicates of the form “$x = y$”, where $x, y \in R$.
  
  – Example: $(x = y) \lor \lnot(y = z \rightarrow x = z)$

• Note that here...
  
  – we implicitly say that ‘$=$‘ is the equality predicate, and
  
  – we redefine the domain of the variables
Fragments

• So far we only restricted the nonlogical symbols.
• Sometimes we want to
  – restrict the grammar
  – restrict the allowed logical symbols.

• These are called **logic fragments**.

• Examples:
  – The **quantifier-free fragment** over $\sum = \{=,+,0,1\}$
  – The **conjunctive fragment** over $\sum = \{=,+,0,1\}$
Fragments

• Let $\Sigma = \emptyset$
  – (T must be empty: no nonlogical symbols to interpret)
• Q: What is the quantifier-free fragment of $T$?
• A: propositional logic

• Thus, propositional logic is also a first-order theory.
  – A very degenerate one.
Theories

• Let $\sum = \{\}$
  – (T must be empty: no nonlogical symbols to interpret)

• Q: What is T?

• A: Quantified Boolean Formulas (QBF)

• Example:
  – $\forall x_1 \exists x_2 \forall x_3. x_1 \rightarrow (x_2 \lor x_3)$
Some famous theories

- Presburger arithmetic: $\Sigma = \{0, 1, +, =\}$
- Peano arithmetic: $\Sigma = \{0, 1, +, *, =\}$
- Theory of reals
- Theory of integers
- Theory of arrays
- Theory of pointers
- Theory of sets
- Theory of recursive data structures
- ...
Expressiveness of a theory

• Each formula defines a **language**: the set of satisfying assignments (‘models’) are the words accepted by this language.

• Consider the fragment ‘2-CNF’

  \[
  \text{formula : } (\text{ literal } \lor \text{ literal }) \mid \text{ formula } \land \text{ formula }
  \]

  \[
  \text{literal: } \text{Boolean \text{ -- variable } \mid \neg\text{Boolean \text{ -- variable}}}
  \]

  \[(x_1 \lor \neg x_2) \land (\neg x_3 \lor x_2)\]
Expressiveness of a theory

• Now consider a Propositional Logic formula \( \phi: (x_1 \lor x_2 \lor x_3) \)

• \textbf{Q}: Can we express this language with 2-CNF?

• \textbf{A}: No.

• Proof:
  – The language accepted by \( \phi \) has 7 words: all assignments other than \( x_1 = x_2 = x_3 = F \).
  – The first 2-CNF clause removes \( \frac{1}{4} \) of the assignments, which leaves us with 6 accepted words. Additional clauses only remove more assignments.
Expressiveness of a theory

**Claim:** $2CNF \prec \text{Propositional Logic}$

**Denote:** $L_1 \prec L_2$

$L_2$ is more expressive than $L_1$.

- **Claim:** $2CNF \prec \text{Propositional Logic}$
- Generally there is only a partial order between theories.
The tradeoff

• So we see that theories can have different expressive power

• Q: why would we want to restrict ourselves to a theory or a fragment? why not take some ‘maximal theory’...

• A: Adding axioms to the theory may make it harder to decide or even undecidable.
Example: First Order Peano Arithmetic

- $\sum = \{0,1, '+', '*', '='\}$
- Domain: Natural numbers

Undecidable!
Example: First Order Presburger Arithmetic

- $\sum = \{0, 1, '+', '*', '='\}$
- Domain: Natural numbers

decidable!
Tradeoff: expressiveness/computational hardness.

• Assume we are given theories $L_1 < \cdots < L_n$
When is a specific theory useful?

1. Expressible enough to state something interesting.

2. Decidable (or semi-decidable) and more efficiently solvable than richer theories.

3. More expressive, or more natural for expressing some models in comparison to ‘leaner’ theories.
Some useful first-order theories

<table>
<thead>
<tr>
<th>Theory name</th>
<th>Example formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equality</td>
<td>$y_1 = y_2 \land \neg(y_1 = y_3) \rightarrow \neg(y_1 = y_3)$</td>
</tr>
<tr>
<td>Equality + UF</td>
<td>$y_1 = y_2 \land \neg(F(y_1) = F(y_3)) \rightarrow \neg(y_1 = y_3)$</td>
</tr>
<tr>
<td>Linear arithmetic</td>
<td>$(2z_1 + 3z_2 \leq 5) \lor (z_2 + 5z_2 - 10z_3 \geq 6)$</td>
</tr>
<tr>
<td>Bit vectors</td>
<td>$((a &gt;&gt; b) &amp; c) &lt; c$</td>
</tr>
<tr>
<td>Arrays</td>
<td>$(i = j \land a[j] = 1) \rightarrow a[i] = 1$</td>
</tr>
<tr>
<td>Pointer logic</td>
<td>$p = q \land *p = 5 \rightarrow *q = 5$</td>
</tr>
<tr>
<td>Quantified Boolean</td>
<td>$\forall x. \exists y. y \rightarrow x$</td>
</tr>
<tr>
<td>Formulas</td>
<td>Combined theories&lt;br&gt;$ (i \leq j \land a[j] = 1) \rightarrow a[i] &lt; 2$</td>
</tr>
</tbody>
</table>

The SMT web page: [http://combination.cs.uiowa.edu/smtlib/](http://combination.cs.uiowa.edu/smtlib/)