Solvers Principles and Architecture (SPA)

Part 2

SMT Solvers

Master Sciences Informatique (Sif)
September, 2019
Rennes

Khalil Ghorbal
khalil.ghorbal@inria.fr

Recall that **logic** is a pair of **syntax** and **semantics**.

Syntax

- Alphabet: set of symbols
- Expressions: sequences of symbols
- Rules: identifying **well-formed** expressions

Semantics

- **Meaning**: what is meant by well-formed expressions
- Rules: infer the meaning from subexpressions

In addition to Logical symbols: $\neg$, $\wedge$, $\longrightarrow$, etc. (alphabet of propositional logic)
We will be adding:

- variables symbols: x, y, etc.
- **parameters**, or non-logical symbols: $\exists$, f, $\leq$, $=$, $+$, π, etc.

Quantifiers

- Exists: $\exists$
- Forall: $\forall$

Functions

- Symbol (or name)
- Output type (or kind) – (Co-domain)
- Inputs arity (or cardinality) and their respective types – (Domain)

Predicates

- Sets described by some relations
- n-arity functions with co-domain $\{\mathtt{F}, \mathtt{T}\}$ (False/True in PL)
- Predicate symbols: $=$, $<$, $\in$, etc.

Constants

- Functions with **arity zero**
- Usual symbols: π, 1, $\emptyset$, etc.
- Predicates with arity zero are the propositional constants ($\mathtt{F}$, $\mathtt{T}$).

First-order means **quantifiers** are only allowed over **variables**: $Q_i x_i$.

- **Each quantifier is necessarily related to a variable**.
- A variable is either **free** or bound by a quantifier.

Examples

- Function $+ : (x, y) \mapsto x + y$
- Predicate: $f(x) = f(y)$ (for some function f)
- Predicate: $x \leq f(y)$

Example of First-order Languages

Basic Set Language

- Relationship predicate: $\mathcal{R}$
- Constant: $\emptyset$

Elementary Number Language

- Constant: 0
- Function: $\mathtt{Succ}$
- Equality predicate: $=$

Terms

Built **inductively** from functions' symbols **applied** to constants and variables.

- A variable v is a term
- A constant 0 is a term
- The function f applied to terms t_1 and t_2 is a term named $f(t_1, t_2)$

Atomic Formulas

Built by applying **predicates on terms**.

- $\mathtt{F}/\mathtt{T}$ are atomic predicates
- $\leq\ v\,0$ is an atomic predicate (prefix notation)
- $t_1 = t_2$ is an atomic predicate (infix notation)

Built **inductively** from atomic formulas with logic connectives and quantifiers.

- $\neg\phi$ is a formula
- $\phi_1 \longrightarrow \phi_2$ is a formula
- $Q_1 v_1 . Q_2 v_2 . \phi(t, g(t))$ is a formula
- Terms t and $g(t)$ may or may not contain the variables v_1 and v_2

A variable in a wff is either **free** or bound to a quantifier.

- $\exists v_1.f(v_1) < v_2$: v_2 is free
- $\forall v_1.\exists v_2.P(v_1, g(v_1, v_2))$: both variables are bound

A wff with no free variables is called a **sentence**.

A **signature** (Σ) contains the parameters of the language, that is all its non-logical **symbols**: constants, functions, and predicates.

Example: Elementary Numbers Signatures

- $(0, \mathtt{Succ}, =)$
- $(0, 1, +, -, >)$

An **interpretation** (M) of a signature is twofold:

- An underlying domain $\mathcal{D}_M$ (e.g. natural numbers)
- An interpretation of all the symbols of Σ over $\mathcal{D}_M$

Example: $\Sigma := (0, 1, +, -, >)$

- $\mathcal{D}$ is $\mathbb{N}$ or $\mathbb{Z}$
- 0 and 1 are the natural numbers *zero* and *one*
- $+ : (x, y) \mapsto x + y$, $- : (x, y) \mapsto x - y$
- $> : (x, y) \mapsto x > y$
- wff $w : \exists x.\forall y.\neg(x > y)$ (sentence)

Let $\mathcal{V}$ denote the set of variables.
Given an interpretation M, an **assignement** is a map $\sigma : \mathcal{V} \to \mathcal{D}_M$.
The assignement σ depends on the interpretation M.
The interpretation M associates

- Functions' symbols (f) of arity n to actual mathematical functions ($f_M : \mathcal{D}_M^n \to \mathcal{D}_M$)
- Terms to elements in $\mathcal{D}_M$
- Predicates' symbols (P) of arity n to subsets P_M in $\mathcal{D}_M^n$

Inductive Interpretation of wff

- $[\![P t_1 t_2]\!]_{M,\sigma} \triangleq ([\![t_1]\!]_\sigma, [\![t_2]\!]_\sigma) \in P_M$.
- $[\![\forall v.w]\!]_{M,\sigma} \triangleq (\forall m \in \mathcal{D}_M.\ [\![w[v \setminus m]]\!]_\sigma = 1)$ (m is a fresh variable not appearing in w).

Let Σ be a signature. A $\Sigma-$Theory T is a set of sentences over Σ.
The interpretation M is a *model* of T if M satisfies all the sentences of T.
Let T denote a theory, and $\sigma : \mathcal{V} \rightarrow \mathcal{D}_M$ an assignement.

- σ **satisfies** w w.r.t. M (model of T) if and only if $[\![w]\!]_{M,\sigma} = 1$
- w is T-**satisfiable** w.r.t. M if there exist M (model of T), σ such that σ satisfies w w.r.t. M
- w is T-**unsatisfiable** if and only if for all models M of T

$$\forall \sigma.\ ([\![w]\!]_{M,\sigma} = 0) \quad .$$

- The **validity problem** for T is the problem of deciding, for each Σ-formula w, if w is T-valid.
- The **satisfiability problem** for T is the problem of deciding, for each Σ-formula w, if w is T-satisfiable.

Proving Validity

w is T-valid if and only if $\neg w$ is T-unsatisfiable.

Examples

- $\forall v_1.Pv_1 \models Pv_2$
- $\forall v_1.Pv_1 \models \exists v_2.Pv_2$
- $\exists v_1.\forall v_2.Qv_1v_2 \models \forall v_2.\exists v_1.Qv_1v_2$
- $\models \exists v_1(Pv_1 \longrightarrow \forall v_2.Pv_2)$
- $\forall v_1.\exists v_2.Qv_1v_2 \not\models \exists v_2.\forall v_1.Qv_1v_2$
- $Pv_1 \not\models \forall v_1.Pv_1$ (Depends on M)

Examples

- $\forall v_1.Pv_1 \models Pv_2$
- $\forall v_1.Pv_1 \models \exists v_2.Pv_2$
- $\exists v_1.\forall v_2.Qv_1v_2 \models \forall v_2.\exists v_1.Qv_1v_2$
- $\models \exists v_1(Pv_1 \longrightarrow \forall v_2.Pv_2)$
- $\forall v_1.\exists v_2.Qv_1v_2 \not\models \exists v_2.\forall v_1.Qv_1v_2$
- $Pv_1 \not\models \forall v_1.Pv_1$ (Depends on M)

Examples

- $\forall v_1.Pv_1 \models Pv_2$
- $\forall v_1.Pv_1 \models \exists v_2.Pv_2$
- $\exists v_1.\forall v_2.Qv_1v_2 \models \forall v_2.\exists v_1.Qv_1v_2$
- $\models \exists v_1(Pv_1 \longrightarrow \forall v_2.Pv_2)$
- $\forall v_1.\exists v_2.Qv_1v_2 \not\models \exists v_2.\forall v_1.Qv_1v_2$
- $Pv_1 \not\models \forall v_1.Pv_1$ (Depends on M)

Examples

- $\forall v_1.Pv_1 \models Pv_2$
- $\forall v_1.Pv_1 \models \exists v_2.Pv_2$
- $\exists v_1.\forall v_2.Qv_1v_2 \models \forall v_2.\exists v_1.Qv_1v_2$
- $\models \exists v_1(Pv_1 \longrightarrow \forall v_2.Pv_2)$
- $\forall v_1.\exists v_2.Qv_1v_2 \not\models \exists v_2.\forall v_1.Qv_1v_2$
- $Pv_1 \not\models \forall v_1.Pv_1$ (Depends on M)

Examples

- $\forall v_1.Pv_1 \models Pv_2$
- $\forall v_1.Pv_1 \models \exists v_2.Pv_2$
- $\exists v_1.\forall v_2.Qv_1v_2 \models \forall v_2.\exists v_1.Qv_1v_2$
- $\models \exists v_1(Pv_1 \longrightarrow \forall v_2.Pv_2)$
- $\forall v_1.\exists v_2.Qv_1v_2 \not\models \exists v_2.\forall v_1.Qv_1v_2$
- $Pv_1 \not\models \forall v_1.Pv_1$ (Depends on M)

Examples

- $\forall v_1.Pv_1 \models Pv_2$
- $\forall v_1.Pv_1 \models \exists v_2.Pv_2$
- $\exists v_1.\forall v_2.Qv_1v_2 \models \forall v_2.\exists v_1.Qv_1v_2$
- $\models \exists v_1(Pv_1 \longrightarrow \forall v_2.Pv_2)$
- $\forall v_1.\exists v_2.Qv_1v_2 \not\models \exists v_2.\forall v_1.Qv_1v_2$
- $Pv_1 \not\models \forall v_1.Pv_1$ (Depends on M)

Quantifier free formula: $(x \leq 0 \vee x + y \leq 0) \wedge y \geq 1 \wedge x \geq 1$
Translated into a CNF: $(a \vee b) \wedge c \wedge d$

SAT gives $(a, b, c, d) = (1, 0, 1, 1)$
But $x \leq 0 \wedge x \geq 1$ is a contradiction:
Learn $\bar{a} \vee \bar{d}$

SAT gives $(a, b, c, d) = (0, 1, 1, 1)$
But $x + y \leq 0 \wedge y \geq 1 \wedge x \geq 1$ is a contradiction:
Learn $\bar{b} \vee \bar{c} \vee \bar{d}$
The problem is UNSAT.