Learning Boolean Specifications

Hans Kleine Büning

VL

University of Paderborn

April 4, 2013
Outline

1. Configuration Systems
2. Specification Problem
3. Query Learning
4. Learning Terms and Clauses
5. Experimental Results

joined work with Uwe Bubeck
Configuration

Components

\[ C_1, C_i, C_n \]

System

\[ \approx \text{Customer demands} \]

- automotive industry (car configuration)
- bill of materials
- ...
Rule-based Configuration:

Rules: IF park heating and Engine=Diesel THEN Battery=strong

Components: $a^i_j$

Query: $\exists x_1 \in \{a_1^1, a_2^1\} \ldots x_n \in \{a_1^n, a_2^n\} : \land_j x_j \land \alpha \models p$
**Rule-based Configuration:**

**Rules:** IF park heating and Engine=Diesel THEN Battery=strong

**Components:** $a_j^i$

Query: $\exists x_1 \in \{a_1^1, a_2^1\} \ldots x_n \in \{a_1^n, a_2^n\} : \bigwedge_j x_j \land \alpha \models p$

**Lemma**

*Configuration problem is NP-complete*
Challenge: Complex components with functional behavior

\[ f : D_1 \times \ldots \times D_n \rightarrow D_0 \]

- hard/software components
- software components
Challenge: Complex components with functional behavior

\[ f : D_1 \times \ldots \times D_n \rightarrow D_0 \]

- hard/software components
- software components

Collaborative Research Centre (12 years)
’On the Fly Computing’

Central project: configuration of software components
Challenge: Complex components with functional behavior

\[ f : D_1 \times \ldots \times D_n \rightarrow D_0 \]

Components: Set of Boolean functions \( h, f_0, \ldots, f_n, \ldots \)

Target function: Boolean function \( g \)

Product structure: Circuit \( C \), the nodes are the components
Components are Boolean functions: \( C_i = \{ f_i(x_1, \ldots, x_m) \} \)

Representation: **Propositional formulas**

Quantified Boolean formulas

\[ \forall x \exists y [(x \lor y \lor z) \land (x \lor \neg y)] \approx z \]

Worst-Case:
Function over \( n \) variables \( \Longrightarrow \) exponential size of the formulas
Specification Problem

- component not longer available
- no solution, but partial solution

⇒ specification of the missing component
Specification Problem

- component not longer available
- no solution, but partial solution

\[ \Rightarrow \text{specification of the missing component} \]

**Given**: partial configuration, single unknown component \( h \)
Specification Problem

**Example:** partial configuration

\[ C(h)(z_1, z_2) = \{ \text{out} = \neg y, \ y = h(x_1, x_2), \ x_1 = (z_1 \lor z_2), \ x_2 = (z_1 \lor z_2) \} \]

\[ \approx \ \neg z_1 \land \neg z_2 \]
Specification Problem

Example: partial configuration
\[ C(h)(z_1, z_2) = \{ \text{out} = \neg y, \ y = h(x_1, x_2), \ x_1 = (z_1 \lor z_2), \ x_2 = (z_1 \lor z_2) \} \]

multiple solutions: \( h = x_1, h = x_1 \lor x_2, h = x_2, h = x_1 \land x_2 \)
Definition

(Specification Problem)

**Instance**: partial configuration $C(h)(z)$, target function $g(z)$.

**Query**: $\exists H : C(h/H)(z) \approx g(z)$?

The specification problem $S = \langle g(z) \approx ? C(h)(z) \rangle$ and the solution space is defined as $SP(S) := \{H : H \text{ is a solution of } S\}$. 
Definition

(Specification Problem)

**Instance**: partial configuration $C(h)(z)$, target function $g(z)$.

**Query**: $\exists H : C(h/H)(z) \approx g(z)$?

The specification problem $S = \langle g(z) \approx C(h)(z) \rangle$ and the solution space is defined as $SP(S) := \{ H : H$ is a solution of $S \}$.

Theorem

*For a solvable specification problem $S = \langle g(z) \approx C(h)(z) \rangle$ with unknown function $h(x)$ it holds for $h_1, h_2 \in SP(S)$:
1. $h_1 \land h_2, h_1 \lor h_2 \in SP(S)$.
2. For every formula $\alpha$ over $x$: $h_1 \models \alpha \models h_2$ iff $\alpha \in SP(S)$.***
Let $h_1, \ldots, h_k$ be the solutions

minimal solution $h_{\text{min}} := \bigwedge_{1 \leq i \leq k} h_i(x)$

maximum solution $h_{\text{max}}(x) := \bigvee_{1 \leq i \leq k} h_i(x)$
Let $h_1, \ldots, h_k$ be the solutions

minimal solution $h_{\min} := \bigwedge_{1 \leq i \leq k} h_i(x)$

maximum solution $h_{\max}(x) := \bigvee_{1 \leq i \leq k} h_i(x)$

reachable input values for the unknown function $h$

$(x_1 = 1, x_2 = 0)$ not possible $\implies$ partial specification
Let \( h_1, \ldots, h_k \) be the solutions

minimal solution \( h_{\min}(x) := \bigwedge_{1 \leq i \leq k} h_i(x) \)

maximum solution \( h_{\max}(x) := \bigvee_{1 \leq i \leq k} h_i(x) \)

\[ h_{\min} \models h_{\max} \]

\[ D(S) = \{ x : h_{\min}(x) = h_{\max}(x) \} \]

**Definition**

The domain is defined as \( D(S) := \{ x : \neg h_{\max}(x) \lor h_{\min}(x) \} \).

For every solution \( h \) the partial function \( h|_{D(S)} \) is called a *prime* solution.
Let $h_1, \ldots, h_k$ be the solutions
minimal solution $h_{min} := \land_{1 \leq i \leq k} h_i(x)$
maximum solution $h_{max}(x) := \lor_{1 \leq i \leq k} h_i(x)$

**Definition**

The domain is defined as $D(S) := \{x : \neg h_{max}(x) \lor h_{min}(x)\}$. For every solution $h$ the partial function $h|_{D(S)}$ is called a *prime* solution.

**Example:** solutions $x_1, x_2, (x_1 \land x_2), (x_1 \lor x_2)$

$\neg h_{max}(x) \lor h_{min}(x) \approx (\neg x_1 \land \neg x_2) \lor (x_1 \land x_2) \approx (x_1 \leftrightarrow x_2)$

$\implies D(S) = \{(0,0),(1,1)\}$

$\implies$ specification only for $D(S)$
QBF-hypothesis: \( H(x) \)
\[ := \exists z[ C_0(z, y/1) = g(z) \land C_0(z, y/0) \neq g(z) \land \land_{1 \leq i \leq n} x_i = C_i(z) ] \]

Problems
1. multiple solutions
2. equivalent propositional formulas (length, running time)
3. solutions in a class \( K \): CNF, Horn, 2-CNF, TERM, CLAUSE, restricted length
Concept Learning

**Black-Box**: Hidden propositional formula $h$ over the variables $x_1, \ldots, x_n$

**Goal**: Learn an equivalent formula

- supervised learning: teacher and learner
- classes of formulas: term, clause, DNF, monotone DNF, Horn, ..
  (Angluin et. al.)

**Example**

variables $x_1, x_2, x_3$

formula is a term (conjunction of literals)

for example: hidden formula $t = (x_1 \land \neg x_3)$
1 **Membership Query:** for a truth assignment $\nu(x_i) \in \{0, 1\}$: 
$\nu(h) = 1$?

2 **Equivalence Query:** for a formula $f$:
$h \approx f$?

answer: *yes*, if $f \approx h$, 
otherwise, *no* and a separating truth assignment $\nu$
$(\nu(h) = 1, \nu(f) = 0)$ or $(\nu(h) = 0, \nu(f) = 1)$

$\implies$ algorithms for learning propositional formulas
Example: formula is a term, variables $x_1, x_2, x_3$:

1. equivalence query: $f = x_1 \land \neg x_1$ equivalent to $h$?
2. no and $v = (1, 1, 0)$ satisfies $h$
3. membership query: $v = (0, 1, 0)$; no $\implies x_1 \in h$
4. membership query: $v = (1, 0, 0)$; yes $\implies x_2, \neg x_2 \not\in h$
5. membership query: $v = (1, 1, 1)$; no $\implies \neg x_3 \in h$
6. $h \approx x_1 \land \neg x_3$
Example: formula is a term, variables $x_1, x_2, x_3$:

1. equivalence query: $f = x_1 \land \neg x_1$ equivalent to $h$?
2. no and $v = (1, 1, 0)$ satisfies $h$
3. membership query: $v = (0, 1, 0)$; no $\implies x_1 \in h$
4. membership query: $v = (1, 0, 1)$; yes $\implies x_2, \neg x_2 \not\in h$
5. membership query: $v = (1, 1, 1)$; no $\implies \neg x_3 \in h$
6. $h \approx x_1 \land \neg x_3$

$\implies$ learning prime implicants (DNF, monotone DNF)
Specification versus query learning

1. borrow some ideas
2. length of the solutions
3. no direct access to the unknown component
4. more information: target function, partial configuration
5. multiple non-equivalent solutions
Specification versus query learning

1. borrow some ideas
2. length of the solutions
3. no direct access to the unknown component
4. more information: target function, partial configuration
5. multiple non-equivalent solutions

Ad 2: target function $g(z)$ is in DNF,
unknown function $h$ must be in CNF
$\exists h \in CNF : h(z) \approx g(z)$,
but in worst-case of super-polynomial length
Algorithm-Term

**Input:** Partial configuration $C(h)(z)$, target function $g(z)$

**Output:** $H \in TERM$: with $C(h/H)(z) \approx g(z)$ if solution exists, no otherwise

$$C(h)(z) = \{out = C_0(z, y), y = h(x_1, \ldots, x_n), x_1 = C_1(z), \ldots, x_n = C_n(z)\}$$

```plaintext
begin
if $\exists$ truth assignment $v$:
$g(v(z)) = C_0(v(z), y = 1) \land g(v(z)) \neq C_0(v(z), y = 0)$
then $H = \{x^{b_j} : b_j = C_j(v(z)), 1 \leq j \leq n\} \ldots$
```

...
Experimental results

1. SAT-solver: PicoSAT, version 936
2. delete in a circuit $C(z)$ the function of a node: $C(h)(z)$
3. target function $C(z)$
4. try to learn a term $t$ with $C(h/t)(z) \approx C(z)$
5. benchmark circuits: ISCAS’85 and Tommi Juntila’s collection
Future work

1. Components with multiple functions
   Example: component \( \{ (h_1(x), \ldots, h_r(x)) \} \), each function \( h_i \) must be a term
   includes the problem of learning \( DNF(t) \), DNF formulas with at most \( t \) terms.

2. Learning specifications for Horn formulas, monotone DNF,..

3. Components with single output, but the functions are encoded as QBF

4. Transformations and implementations, SAT solver or QSAT solver, equivalence solver?

5. Systems with relative few but complex functions
Dynamic Configuration

1. Given: components $f_i$ (initial functions)
2. compose new functions: $g(z_1, z_2) = f_1(f_3(z_1, 0), f_2(z_2))$
3. compose new functions: $h(y, x) := g(f_2(x, x), g(x, y))$
4. ... $t$ times
5. $\implies$ fixed programm

Problem:
Instance: Initial functions, target function $G$
Query: Does there exist a composition equivalent to $G$?

Equivalent to the problem whether for a quantified Boolean formula a composition over a fixed set of initial functions exists.