The Boolean Solution Problem from the Perspective of Predicate Logic

Christoph Wernhard

TU Dresden

FroCoS 2017

Brasília, 28 September 2017

Supported by DFG grant WE 5641/1-1

The Solution Problem: Example

Given $F[p_1 \ldots p_n]$, find $G_1 \ldots G_n$ s.th. $\models F[G_1 \ldots G_n]$

$$
\begin{aligned}
F[p_1 p_2] \;=\; & \forall x\,(a(x) \rightarrow b(x)) \;\rightarrow \\
& \quad (\forall x\,(p_1(x) \rightarrow p_2(x)) \land \\
& \quad\; \forall x\,(a(x) \rightarrow p_2(x)) \land \forall x\,(p_2(x) \rightarrow b(x))).
\end{aligned}
$$

$$
\begin{aligned}
& p_1(x)\{p_1 \mapsto \lambda x_1.a(x_1)\} \\
= \; & (\lambda x_1.a(x_1))x \\
= \; & a(x)
\end{aligned}
$$

$$
\begin{aligned}
\models \; & \forall x\,(a(x) \rightarrow b(x)) \;\rightarrow \\
& \quad (\forall x\,(a(x) \rightarrow b(x)) \land \\
& \quad\; \forall x\,(a(x) \rightarrow b(x)) \land \forall x\,(b(x) \rightarrow b(x))).
\end{aligned}
$$

Some solutions

G_1	G_2
$\lambda x_1.a(x_1)$	$\lambda x_1.b(x_1)$
$a(x_1)$	$a(x_1)$
$\bot$	$a(x_1)$
$a(x_1) \land b(x_1)$	$a(x_1) \lor b(x_1)$

Some non-solutions

G_1	G_2
$b(x_1)$	$a(x_1)$
$a(x_1)$	$\bot$
$\bot$	$\top$

The Solution Problem: Historic Context and Related Works

Given $F[p_1 \ldots p_n]$, find $G_1 \ldots G_n$ s.th. $\models F[G_1 \ldots G_n]$

- In the **Algebra of Logic** a core reasoning mode, together with elimination
  - Boole, Peirce, **Schröder** („*Auflösungsproblem*”), Löwenheim
- **Boolean algebra: [Rudeanu, 1974]** *Boolean Functions and Equations*
- **Boolean unification** on term level added to Prolog [Martin and Nipkow, 1986, Büttner and Simonis, 1987], Complexity: [Kanellakis et al., 1990, Baader, 1998], Implementation: [Carlsson, 1991]
- **Towards predicate logic:**
  - relational monadic formulas [Behmann, 1950] ([Löwenheim, 1908]?)
  - quantifier-free formulas [Eberhard et al., 2017] *Boolean unif. with predic.*
- Here: formalization with **predicate logic**, targeted at automated reasoning
  - transfer of **classical material**
  - **second-order quantification** is essential: $\forall, \exists$ instead of Π, $\bigcup$
  - **abstractions** from classical special techniques
  - relations to Craig **interpolation**, **definability** (**“query reformulation”**) and **second-order quantifier elimination** (**“forgetting”**)

The Solution Problem: Applications

Given $F[p_1 \dots p_n]$, find $G_1 \dots G_n$ s.th. $\models F[G_1 \dots G_n]$

- Collections of **examples**: [Schröder, 1890, Rudeanu, 1974]
- Vast number of **applications** in graph theory, automata theory, circuit design, query optimization, marketing problems, medical diagnosis, biochemistry, ... [Rudeanu, 2001, Brown, 2003]
- Proof compression [Eberhard et al., 2017]
- Detecting similar concepts [Baader and Narendran, 2001]

The Solution Problem: Formal Characterization

Given $F[p_1 \ldots p_n]$, find $G_1 \ldots G_n$ s.th. $\models F[G_1 \ldots G_n]$

Definition.

- A ***solution problem (SP)*** is a pair $F[\boldsymbol{p}]$ of a formula F and a sequence $\boldsymbol{p}$ of distinct predicates, called unknowns.
- A ***unary SP*** is a SP with a single unknown.
- A ***(particular) solution*** of a SP is a sequence $\boldsymbol{G}$ of formulas s.th. $\mathrm{substitutible}(\boldsymbol{G}, \boldsymbol{p}, F)$ and $\models F[\boldsymbol{G}]$.

- *Particular* in contrast to *general* solutions
- substitutible captures i.a.:
  - no free variables in $\boldsymbol{G}$ get bound if inserted for $\boldsymbol{p}$ into $F[\boldsymbol{p}]$
  - no member of $\boldsymbol{p}$ occurs free in a member of $\boldsymbol{G}$
- Dual characterization based on unsatisfiability is possible

The Solution Problem for First-Order Logic is Recursively Enumerable

Given $F[p_1 \ldots p_n]$, find $G_1 \ldots G_n$ s.th. $\models F[G_1 \ldots G_n]$

- Assuming a countable vocabulary, the **set of solutions** of a **first-order** SP is **recursively enumerable**

Enumerated sequences of formulas	Validity test
$G_{11} \ldots G_{n1}$	$\models F[G_{11} \ldots G_{n1}]$?
$G_{12} \ldots G_{n2}$	$\models F[G_{12} \ldots G_{n2}]$?
$G_{13} \ldots G_{n3}$	$\models F[G_{12} \ldots G_{n3}]$?
$\vdots$	$\vdots$

Different Views on the Solution Problem

Given $F[p_1 \ldots p_n]$, find $G_1 \ldots G_n$ s.th. $\models F[G_1 \ldots G_n]$

Boolean Unification

Definientia and Interpolants

Construction of Witnesses
for Second-Order Quantifier Elimination

The Solution Problem as Boolean Unification

Given $F[p_1 \ldots p_n]$, find $G_1 \ldots G_n$ s.th. $\models F[G_1 \ldots G_n]$

- E-unification in the theory of Boolean algebra (or logical equivalence)

$\Rightarrow$ A solution represents a unifier $\sigma = \{p_1 \mapsto G_1, \ldots, p_n \mapsto G_n\}$:

$$\begin{array}{ll} & \models F[G_1 \ldots G_n] \\ \text{iff} & F[p_1 \ldots p_n]\sigma =_E \top\sigma \end{array}$$

$\Leftarrow$ A unifier represents a solution:

$$\begin{array}{ll} & L[\boldsymbol{p}]\sigma =_E R[\boldsymbol{p}]\sigma \\ \text{iff} & \models L[\boldsymbol{G}] \leftrightarrow R[\boldsymbol{G}] \end{array}$$

- can be generalized to a finite sets of equations
- σ must be ground because members of $\boldsymbol{p}$ are not allowed in $\boldsymbol{G}$

The Solution Problem as Construction of SOQE-Witnesses

Given $F[p_1 \ldots p_n]$, find $G_1 \ldots G_n$ s.th. $\models F[G_1 \ldots G_n]$

- **Second-order quantifier elimination (SOQE):**

  Given $\exists \boldsymbol{p}\, F[\boldsymbol{p}]$, find a first-order equivalent

Definition. An ***SOQE-witness*** of $\boldsymbol{p}$ in $\exists \boldsymbol{p}\, F[\boldsymbol{p}]$ is a sequence $\boldsymbol{G}$ of formulas such that $\mathrm{substitutible}(...)$ and

$$\exists \boldsymbol{p}\, F[\boldsymbol{p}] \equiv F[\boldsymbol{G}].$$

- Computation of SOQE-witnesses is **restricted SOQE**, where resultants must have the form $F[\boldsymbol{G}]$

$\Rightarrow$ $\boldsymbol{G}$ is is a solution of $F[\boldsymbol{p}]$ iff

1. $\boldsymbol{G}$ is an SOQE-witness of $\boldsymbol{p}$ in $\exists \boldsymbol{p}\, F[\boldsymbol{p}]$, and
2. $\models \exists \boldsymbol{p}\, F[\boldsymbol{p}]$.

$\Leftarrow$ $\boldsymbol{G}$ is an SOQE-witness of $\boldsymbol{p}$ in $\exists \boldsymbol{p}\, F[\boldsymbol{p}]$ iff
$\boldsymbol{G}$ is a solution of $(\neg F[\boldsymbol{p}'] \lor F[\boldsymbol{p}])[\boldsymbol{p}]$ ($\boldsymbol{p}'$ fresh)

The Solution Problem, Definientia and Interpolants

Given $F[p_1 \ldots p_n]$, find $G_1 \ldots G_n$ s.th. $\models F[G_1 \ldots G_n]$

- Assume substitutible(...)

$$
\begin{array}{llcll}
 & G \text{ is a solution of } F[p] & & & \\
\text{iff} & & \models & F[G] & \\
\text{iff} & p \leftrightarrow G & \models & F[p] & \\
\text{iff} & \neg F[p] & \models & \neg(p \leftrightarrow G) & \\
\hline
\text{iff} & \neg F[p] & \models & p \leftrightarrow \neg G & \text{if } p \text{ is nullary} \\
\text{iff} & \multicolumn{3}{l}{G \text{ is a negated definiens of } p \text{ within } \neg F[p]} & \\
\text{iff} & \exists p\,(\neg F[p] \land \neg p) & \models G \models & \neg\exists p\,(\neg F[p] \land p) & \\
\text{iff} & \neg F[\bot] & \models G \models & F[\top] & \\
\text{iff} & \neg F[p] \land \neg p & \models G \models & \neg(\neg F[p'] \land p') & \\
\text{iff} & G \text{ is a Craig interpolant} & & &
\end{array}
$$

Reduction from n-ary to Unary Solution Problems

- Equivalence of the solutions of an **n-ary SP** and the solutions of **n unary SPs** on **existential second-order** formulas:

Theorem. For a SP $F[p_1 \ldots p_n]$ that has a solution the following statements are equivalent:

(a) $G_1 \ldots G_n$ is a solution of the SP.

(b) For $i \in \{1, \ldots, n\}$: G_i is a solution of the unary SP

$$(\exists p_{i+1} \ldots \exists p_n \, F[G_1 \ldots G_{i-1} p_i p_{i+1} \ldots p_n])[p_i]$$

s.th. no member of $p_1 \ldots p_n$ occurs free in G_i.

Solving on the Basis of Second-Order Formulas

(b) For $i \in \{1, \ldots, n\}$: G_i is a solution of the unary SP

$$(\exists p_{i+1} \ldots \exists p_n \, F[G_1 \ldots G_{i-1} p_i p_{i+1} \ldots p_n])[p_i]$$

Algorithm *SOLVE-ON-SECOND-ORDER($SOLVE\text{-}UNARY$)*.
For $i := 1$ to n do: Assign to G_i an output of $SOLVE\text{-}UNARY$ applied to the unary SP as specified in the reduction theorem.

- This algorithm **inherits properties from its $SOLVE\text{-}UNARY$ parameter:**
  - **nondeterministic or deterministic**
  - in the nondet. case: **each solution is reached by an execution path**
  - in the det. case: the output is a most general (i.e. reproductive) solution

Abstraction from the Method of Successive Eliminations

(b) For $i \in \{1, \ldots, n\}$: G_i is a solution of the unary SP

$$(\exists p_{i+1} \ldots \exists p_n \, F[G_1 \ldots G_{i-1} p_i p_{i+1} \ldots p_n])[p_i]$$

- The **method of successive eliminations**, aka **Boole's method**, in the variant of **[Büttner and Simonis, 1987]**:

1. For $i := n$ to 1: Create auxiliary formulas F_i by SOQE s.th.
   $F_i[p_1, \ldots p_i] \equiv \exists p_{i+1} \ldots \exists p_n \, F[p_1, \ldots p_i p_{i+1}, \ldots p_n]$
2. For $i := 1$ to n do: Assign to G_i a solution of $(F_i[G_1 \ldots G_{i-1} p_i])[p_i]$

- The **reduction theorem** abstracts from this:
  - second-order quantification (in place of original algebraic operators)
  - not just for propositional logic, *xor* is not necessary
  - alternate ways to handle intermediate second-order formulas
  - alternate methods to solve the unary SPs
  - deterministic / nondeterministic variants
  - for particular / most general solutions

Solving by Inside-Out SOQE-Witness Construction

Algorithm *SOLVE-BY-WITNESSES*.
Repeatedly eliminate from $\exists p_1 \ldots \exists p_n\, F[p_1 \ldots p_n]$ the innermost quantifier $\exists p_i$ by replacing p_i with an SOQE-witness.

$$\begin{array}{lll} & \exists p_1 \exists p_2 & F[p_1 p_2] \\ \equiv & \exists p_1 & F[p_1 G'_2[p_1]] \qquad \text{because } \exists p_2\, F[p_1 p_2] \;\equiv\; F[p_1 G'_2[p_1]] \\ \equiv & & F[G_1 G'_2[G_1]] \end{array}$$

- ISSUE: Are all solutions reachable in a nondeterministic variant?

Solution Existence Theorem

Theorem. The following statements are equivalent:

(a) There exists a solution of the SP $F[p_1 \ldots p_n]$.

(b) $\models \exists p_1 \ldots \exists p_n \, F[p_1 \ldots p_n]$.

- Holds for n-ary SPs under the **assumption that it holds for unary SPs** with the considered formulas and unknowns
  - expressed in the paper via formula and predicate classes
  - holds for example for propositional formulas: $\exists p \, F[p] \equiv F[F[\top]]$

Π_2^P-Completeness of Solution Existence in the Propositional Case

Theorem. The following statements are equivalent:

(a) There exists a solution of the SP $F[p_1 \ldots p_n]$.

(b) $\models \exists p_1 \ldots \exists p_n \, F[p_1 \ldots p_n]$.

- The following tasks are both Π_2^P-complete:
  - determining the **existence of a solution of a propositional SP**
  - determining the **validity of an existential QBF**
- Shown for Boolean unification: [Kanellakis et al., 1990, Baader, 1998]

SOQE Resultant as Precondition for Solution Existence

- Relationship between SOQE and solution existence expanded on by [Schröder, 1890]:

  The resultant of eliminating the unknowns of a SP is the unique weakest precondition under which the SP has a solution

- (We continue to assume solution existence of unary SPs)

Theorem. Let A be a formula s.th. no member of $\boldsymbol{p}$ occurs in A and $A \equiv \exists \boldsymbol{p}\, F[\boldsymbol{p}]$. Then

(i) $(A \rightarrow F[\boldsymbol{p}])[\boldsymbol{p}]$ has a solution.

(ii) If B is a formula s.th. no member of $\boldsymbol{p}$ occurs in B, and $(B \rightarrow F[\boldsymbol{p}])[\boldsymbol{p}]$ has a solution, then $B \models A$.

SOQE Resultant as Precondition for Solution Existence: Example

$$F[p_1 p_2] = \forall x\,(p_1(x) \rightarrow p_2(x)) \land \forall x\,(a(x) \rightarrow p_2(x)) \land \forall x\,(p_2(x) \rightarrow b(x)).$$

$$\exists p_1 \exists p_2\, F[p_1 p_2] \equiv \forall x\,(a(x) \rightarrow b(x)). \qquad \text{by SOQE}$$

$$\not\equiv \top.$$

$$\forall x\,(a(x) \rightarrow b(x)) \rightarrow F[p_1 p_2] = \forall x\,(a(x) \rightarrow b(x)) \;\rightarrow (\forall x\,(p_1(x) \rightarrow p_2(x)) \land \forall x\,(a(x) \rightarrow p_2(x)) \land \forall x\,(p_2(x) \rightarrow b(x))).$$

1. The Solution Problem
2. The Method of Successive Eliminations – Abstracted
3. Solution Existence
4. Conclusion

Further Steps

- **Reproductive solutions** as most general solutions ✓
- Solution constructions based on **Craig interpolation** ✓
- Solution constructions related to **SOQE**
  - [Eberhard et al., 2017] generalized to relational monadic formulas ✓
  - relaxed notion of substitutible?
  - adaption of SOQE methods (SCAN, Ackermann approach)?
- Application relevant **generalizations of the problem**
  - solutions in restricted vocabularies ✓
  - expressing synthesis of definitional equivalences ✓
  - weakest / strongest / simplest solution?
- Consideration of particular **formula classes**
- Understanding and systematizing **methods and results** from the literature
  - Boole, Jevons, Pierce, Schröder, Löwenheim, Behmann, ...
- Reviewing the many **applications** mentioned in the literature

Summary

Given $F[p_1 \ldots p_n]$, find $G_1 \ldots G_n$ s.th. $\models F[G_1 \ldots G_n]$

- **Different views**
  - unification
  - r.e. case of SOQE
  - definitions that entail a formula
- **Nice core theory, rooted in the Algebra of Logic**
  - reduction of n-ary to unary SPs
  - solution existence as validity of an existential second-order formula
  - SOQE to express preconditions for solution existence
  - reproductive solutions
- Adaption from historic and algebraic setting to **predicate logic**
  - **second-order quantification** is essential
  - **abstractions**, towards systematizing methods, beyond propositional logic
  - new aspects, e.g.: deterministic / **nondeterministic** / most general
  - placement in **context of modern SOQE** (forgetting), **Craig interpolation** and **definability** (query reformulation)

References I

[Baader, 1998] Baader, F. (1998).
On the complexity of Boolean unification.
Inf. Process. Lett., 67(4):215–220.

[Baader and Narendran, 2001] Baader, F. and Narendran, P. (2001).
Unification of concept terms in description logics.
J. Symb. Comput., 31(3):277–305.

[Behmann, 1950] Behmann, H. (1950).
Das Auflösungsproblem in der Klassenlogik.
Archiv für Philosophie, 4(1):97–109.
(First of two parts, also published in *Archiv für mathematische Logik und Grundlagenforschung*, 1.1 (1950), pp. 17-29).

[Behmann, 1951] Behmann, H. (1951).
Das Auflösungsproblem in der Klassenlogik.
Archiv für Philosophie, 4(2):193–211.
(Second of two parts, also published in *Archiv für mathematische Logik und Grundlagenforschung*, 1.2 (1951), pp. 33-51).

References II

[Brown, 2003] Brown, F. M. (2003).
Boolean Reasoning.
Dover Publications, Mineola, NY, second edition.
(first edition 1990, Kluwer, Norwell, MA).

[Büttner and Simonis, 1987] Büttner, W. and Simonis, H. (1987).
Embedding Boolean expressions into logic programming.
J. Symb. Comput., 4(2):191–205.

[Carlsson, 1991] Carlsson, M. (1991).
Boolean constraints in SICStus Prolog.
Technical Report SICS T91:09, Swedish Institute of Computer Science, Kista.

[Eberhard et al., 2017] Eberhard, S., Hetzl, S., and Weller, D. (2017).
Boolean unification with predicates.
J. Logic and Comput., 27(1):109–128.

[Kanellakis et al., 1990] Kanellakis, P. C., Kuper, G. M., and Revesz, P. Z. (1990).
Constraint query languages.
In *PODS'90*, pages 299–313. ACM Press.

References III

[Löwenheim, 1908] Löwenheim, L. (1908).
Über das Auflösungsproblem im logischen Klassenkalkül.
In *Sitzungsberichte der Berliner Mathematischen Gesellschaft*, volume 7, pages 89–94. B. G. Teubner.

[Martin and Nipkow, 1986] Martin, U. and Nipkow, T. (1986).
Unification in Boolean rings.
In *CADE-8*, volume 230 of *LNCS (LNAI)*, pages 506–513. Springer.

[Rudeanu, 1974] Rudeanu, S. (1974).
Boolean Functions and Equations.
Elsevier.

[Rudeanu, 2001] Rudeanu, S. (2001).
Lattice Functions and Equations.
Springer.

[Schröder, 1890] Schröder, E. (1890).
Vorlesungen über die Algebra der Logik, volume 1.
Teubner, Leipzig.

References IV

[Schröder, 1895] Schröder, E. (1895).
Vorlesungen über die Algebra der Logik, volume 3.
Teubner, Leipzig.

[Wernhard, 2017] Wernhard, C. (2017).
The Boolean solution problem from the perspective of predicate logic – extended version.
Technical Report KRR 17–01, TU Dresden.