Consistent Consequence for Boolean Equation Systems

M.W. Gazda & T.A.C. Willemse

Eindhoven University of Technology

Formal verification

- model checking:

  Various modal logics (LTL, CTL, μ-calculus) and corresponding model checking algorithms (notably Parity Game solvers).
- equivalence checking: broad spectrum of equivalences and algorithms
- static analysis
- specific approaches to specific systems and types of data inolved...

One framework to encompass them all...

(**P**arameterised) **B**oolean **E**quation **S**ystems

Parameterised Boolean Equation Systems

- a common framework in which we can encode and solve various problems from the area of formal verification

  (including model checking using arbitrary μ-calculus formulae)
- essentially, PBESs are fixpoint equation systems over lattices of first-order predicates
- in general undecidable
- equation systems involving only finite domains are decidable; in particular Boolean Equation Systems are in NP$\cap$coNP

Advantages of PBESs

- a uniform and very general framework for solving different formal verification problems
- problems encoded in a compact, symbolic way
- PBESs are amenable to various **manipulations**, also on a symbolic level.
  - invariants
  - minimisation
  - static analysis techniques
  - abstraction (work in progress)
- in case of model checking PBESs encode both model and a property, so certain techniques may become more powerful and/or efficient (abstraction)

Equation systems over lattices

Let $X_1, \ldots, X_n$ be variables ranging over certain lattices $L_1, \ldots, L_n$ and let $f_1, \ldots, f_n$ be monotone functions.

$$\begin{aligned} \sigma_1 X_1 &= f_1(X_1, \ldots, X_n) \\ \ldots & \\ \sigma_n X_n &= f_n(X_1, \ldots, X_n) \end{aligned}$$

$\sigma_i \in \{\mu, \nu\}$ for $i \in \{1, \ldots, n\}$ specifies the *fixpoint sign* of an equation (μ - smallest, ν - largest)

A *solution* of an equation system is a valid *fulfillment* of equations, such that the the smallest/largest fixpoint is taken according to the fixpoint sign, and the earlier fixpoint signs have precedence over the latter.

Alternation depth & blocks

$$
\begin{aligned}
\nu X_1 &= f_1(X_1, \ldots, X_n) \\
\nu X_2 &= f_2(X_1, \ldots, X_n) \\
\mu X_3 &= f_3(X_1, \ldots, X_n) \\
\mu X_4 &= f_4(X_1, \ldots, X_n) \\
\mu X_5 &= f_5(X_1, \ldots, X_n) \\
\nu X_6 &= f_6(X_1, \ldots, X_n)
\end{aligned}
$$

We are interested in two kinds of equation systems:

- *Boolean Equation Systems (BESs)*: all equations are over the boolean lattice $\{\bot, \top\}$

$$\begin{aligned} \nu X &= X \wedge Z \\ \mu Y &= X \wedge (Y \vee Z) \\ \nu Z &= Z \end{aligned}$$

- *Parameterised Boolean Equation Systems (PBESs)*: equations over first-order predicates

$$\begin{aligned} \nu X(n : \mathbb{N}) &= Y(n > 10) \wedge X(n + 1) \\ \mu Y(b : \mathit{Bool}) &= Y(\neg b) \vee X(5) \end{aligned}$$

Boolean Equation Systems

Boolean equation systems are closely related to *parity games*, both problems are mututally reducible in linear time.

This means that BES solution problem is in $NP \cap coNP$ with respect to alternation depth, but no polynomial algorihtm is known.

Fortunately, in practical applications (model checking), the alternation depth is usually very low (≤ 3).

In practice, it may be worthwile to explore certain heuristics to efficiently solve BESs. In particular, we can try to come up with an (efficiently computable) equivalence that respects the solution, and which allows quotienting. We can then first compute the quotient, and then solution on a quotient.

Parameterised Boolean Equation Systems

- in general undecidable (first-order formulae)
- various techniques have been developed in order to facilitate solving PBESs
  - enumeration
  - invariant strenghtening:
    $\sigma X(d : D) = f_X$ vs. $\sigma X'(d : D) = f_X \wedge I(d)$

  Proving correctness of the above techniques was extremely tedious and notationally involved (e.g. proving correctness of enumeration took about 10 pages...).
- a novel technique: abstraction in the setting of PBESs (work in progress): we would need a relation which resembles simulation on Kripke Structures, on which the abstraction theory is based

Equivalence relations & preorders on PBESs

It might be beneficial to investigate relations that approximate solution of equation systems. The two main purposes are:

- minimisation
- facilitate proving correctness of PBES manipulations

Contributions:

- (Willemse, CONCUR 2010): consistent correlation equivalence
- (Gazda & Willemse, SOFSEM 2012):
  - more results on consistent correlation (complexity, proof system)
  - consistent consequence preorder (coarser than consistent correlation)
- current work: developing an abstraction technique for PBESs, using the consistent consequence framework

PBESs vs. BESs

In our current work, we restricted ourselves to plain Boolean Equation Systems.

- ease of presentation; results concerning BESs carry over to PBESs, but are much more notationally involved
- PBES can be seen as a compact notation for (possibly infinite) BESs (we can associate a separate boolean variable with each data value)
- while computing PBESs, we often ultimately end up computing BESs (enumeration)
- BESs are interesting in their own right, esp. because of their connection with parity games

Solution-preserving relations on BESs

$$\begin{aligned} \nu X &= X \wedge Z \\ \mu Y &= X \wedge (Y \vee W) \\ \mu Z &= Z \\ \mu U &= U \wedge W \\ \mu V &= V \wedge W \\ \nu W &= W \end{aligned}$$

We are after a relation on BES variables which approximates the solution.

R approximates the solution if $X \, R \, Y$ implies that:
if solution for X is 'true' ($\top$), then it is 'true' for Y as well

Consistent consequence & consistent correlation

An assignment (environment) η *respects* a relation R if, whenever $X\,R\,Y$, then $\eta(X) \Rightarrow \eta(Y)$.

Consistent consequence

Consistent consequence is a relation R on BES variables with the following properties:

- only variables from the same block (having the same rank) can be related
- for all assignments that respect R, the values of the right-hand sides also respect R

A *consistent correlation* is a symmetric consistent consequence.

The largest consistent consequence is denoted with $\lessdot$, the largest cons. correlation with $\doteqdot$.

Example

If we assume some implication relation on variables, then we must be able to prove the same about the corresponding right-hand sides.

$$\begin{aligned} \nu X &= X \wedge Z \\ \mu Y &= X \wedge (Y \vee W) \\ \mu Z &= Z \\ \mu U &= U \wedge W \\ \mu V &= V \wedge W \\ \nu W &= W \end{aligned}$$

- U and V are consistent consequence of each other, in fact they are consistently correlated ($U \doteqdot V$)
- Z is a consistent consequence of U and V ($U, V \lessdot Z$)
- it can be shown that Y cannot be related to any variable, because of its dependency on X and W

Soundness

Basic requirement: relation must be a valid approximation of the solution.

Theorem

Let $\mathcal{E}$ be an equation system. If R is a consistent consequence on $\mathcal{E}$, then the boolean values in the solution of $\mathcal{E}$ respect R.

Complexity

Theorem

Deciding $\lessdot$ and $\div$ is coNP-complete.

Proof.

Use reduction to logical consequence of two negation-free propositional formulale, known to be coNP-complete. □

Thus for arbitrary BESs, computing $\div$ and $\lessdot$ turns out to be (very likely) more difficult than computing the solution...

- definition of both consistent consequence and correlation are essentially semantic and involve quantification over all consistent environments

- we propose a more accessible syntactic alternative in the form of a proof system

- proof system: axiomatisation of negation-free propositional logic, augmented with a single coinductive rule.

$$\text{CC}\ \frac{\Gamma, X \subset Y \vdash_c f_X \subset f_Y \qquad \text{block}_{\mathcal{E}}(X) = \text{block}_{\mathcal{E}}(Y)}{\Gamma \vdash_c X \subset Y}$$

Axioms of propositional logic

rules of the form $\frac{}{\Gamma \vdash_c A}$, where A ranges over the following laws:

AS1	$\alpha \wedge (\beta \wedge \gamma) \subset (\alpha \wedge \beta) \wedge \gamma$	DS1	$\alpha \vee (\beta \wedge \gamma) \subset (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$
AS2	$(\alpha \wedge \beta) \wedge \gamma \subset \alpha \wedge (\beta \wedge \gamma)$	DS2	$(\alpha \vee \beta) \wedge (\alpha \vee \gamma) \subset \alpha \vee (\beta \wedge \gamma)$
AS3	$\alpha \vee (\beta \vee \gamma) \subset (\alpha \vee \beta) \vee \gamma$	DS3	$\alpha \wedge (\beta \vee \gamma) \subset (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$
AS4	$(\alpha \vee \beta) \vee \gamma \subset \alpha \vee (\beta \vee \gamma)$	DS4	$(\alpha \wedge \beta) \vee (\alpha \wedge \gamma) \subset \alpha \wedge (\beta \vee \gamma)$
COM1	$\alpha \wedge \beta \subset \beta \wedge \alpha$	AB1	$\alpha \vee (\alpha \wedge \beta) \subset \alpha$
COM2	$\alpha \vee \beta \subset \beta \vee \alpha$	AB2	$\alpha \subset \alpha \wedge (\alpha \vee \beta)$
ID1	$\alpha \subset \alpha \wedge \alpha$	ID2	$\alpha \vee \alpha \subset \alpha$
SUP	$\alpha \subset \alpha \vee \beta$	INF	$\alpha \wedge \beta \subset \alpha$
TOP	$\alpha \subset \alpha \wedge \top$	BOT	$\alpha \vee \bot \subset \alpha$

Inequality logic rules

SUB $\quad \dfrac{\Gamma \vdash_c \alpha \subset \beta}{\Gamma \vdash_c \alpha\varsigma \subset \beta\varsigma}$

CTX $\quad \dfrac{\Gamma \vdash_c \alpha \subset \beta}{\Gamma \vdash_c \gamma[X := \alpha] \subset \gamma[X := \beta]}$

TRA $\quad \dfrac{\Gamma \vdash_c \alpha \subset \beta \quad \beta \subset \gamma}{\Gamma \vdash_c \alpha \subset \gamma}$

REF $\quad \dfrac{}{\Gamma \vdash_c \alpha \subset \alpha}$

Consistent consequence rules

CC $\quad \dfrac{\Gamma, X \subset Y \vdash_c f_X \subset f_Y \qquad \mathrm{rank}(X) = \mathrm{rank}(Y)}{\Gamma \vdash_c X \subset Y}$

CNT $\quad \dfrac{}{\Gamma \vdash_c X \subset Y} \; (X \subset Y) \in \Gamma$

An application: abstraction

In many practical instances the equation system resulting from a verification problem is impossible to solve using standard methods (it can be computationally intractable or the computation may not terminate, in case of aribtrary PBESs with infinite domains).

In those cases we would like to adopt to the PBES setting one of the fundamental techniques in formal methods: the abstract interpretation.

We are mainly inspired by the work of Clarke *et al.*: *Model checking and abstraction*, 1994.

We now move to the setting of Parameterised BESs.

- variables (*predicate variables*) may carry *data parameters*
- we allow in addition first-order predicate expressions on the right-hand sides

Formally, a PBES $\mathcal{E}$ is defined by the following grammar:

$$\begin{array}{lll} \mathcal{E} & ::= & \epsilon \mid (\nu X(\mathbf{d_X}{:}\mathbf{D_X}) = \phi) \mid (\mu X(\mathbf{d_X}{:}\mathbf{D_X}) = \phi) \\ \phi, \psi & ::= & b \mid \phi \wedge \psi \mid \phi \vee \psi \mid \forall d{:}D.\ \phi \mid \exists d{:}D.\ \phi \mid X(\mathbf{e}) \end{array}$$

where ϕ, ψ are predicate formulae; b is a simple predicate (without predicate variables); $\mathbf{e}$ is a data expression (vector of data expressions).

Assume that we have, for each X, a surjection $h_X{:}\mathbf{D_X} \to \widehat{\mathbf{D}}_\mathbf{X}$, mapping a *concrete domain* $\mathbf{D_X}$ to an *abstract domain* $\widehat{\mathbf{D}}_\mathbf{X}$.

The *abstraction* $\mathcal{F}^m(\phi)$ of ϕ, for $m \in \{\sqcup, \sqcap\}$, in the context of h is defined inductively:

$$\mathcal{F}^m(b(\mathbf{d})) = \begin{cases} \exists \mathbf{d}.\ h(\mathbf{d}) = \widehat{\mathbf{d}} \wedge b & \text{if } m = \sqcup \\ \forall \mathbf{d}.\ h(\mathbf{d}) \neq \widehat{\mathbf{d}} \vee b & \text{otherwise} \end{cases}$$

$$\mathcal{F}^m((Y(e))(\mathbf{d})) = \begin{cases} \exists \mathbf{d}.\ h(\mathbf{d}) = \widehat{\mathbf{d}} \wedge \widehat{Y}(h(e)) & \text{if } m = \sqcup \\ \forall \mathbf{d}.\ h(\mathbf{d}) \neq \widehat{\mathbf{d}} \vee \widehat{Y}(h(e)) & \text{otherwise} \end{cases}$$

$$\mathcal{F}^m((\phi \oplus \psi)(\mathbf{d})) = \mathcal{F}^m(\phi(\mathbf{d})) \oplus \mathcal{F}^m(\psi(\mathbf{d})) \quad \text{for } \oplus \in \{\wedge, \vee\}$$

$$\mathcal{F}^m((\mathsf{Q}\, d'{:}D.\ \phi)(\mathbf{d})) = \mathsf{Q}\, \widehat{d}'{:}\widehat{D}.\ \mathcal{F}^m(\phi(d', \mathbf{d})) \quad \text{for } \mathsf{Q} \in \{\forall, \exists\}$$

The abstraction is pushed to the simple predicate expressions...

- our abstraction technique can be used to verify arbitrary properties, expressed in μ-calculus formulae
- the abstraction operator as defined here involves quantification over a concrete domain, hence its application is limited to solving PBESs 'by hand'
- *h* can be an arbitrary surjection, no homomorphism / safety condition required
- in order to automate the abstraction, we need to deal with proper homomorphisms or Galois connections
- work in progress: we have developed an automated abstraction technique that uses homomorphism, and proved that it is capable of proving at least as many properties as another technique (Espada & van de Pol) for μCRL toolset, but which uses homomorphisms lifted to Galois connections (much more overhead)

Future work

- use the preorders to devise new techniques for solving (Parameterised) Boolean Equation Systems
  - abstraction
  - detection of irrelevant parameters in PBESs (J. Keiren)
- investigate new, coarser preorders on BES variables, e.g. allow to make substitution at right-hand sides
- in the setting of parity games: work on preorders that are efficiently computable and can be used to reduce PGs before applyign stadard algorithms (bigstep, recursive algorithm)

- Parameterised Boolean Equation Systems are a general framework in which we can encode various verification problems
- advantages: uniform framework, amenable to manipulations, combine model and property
- special case: BESs, closely related to parity games ($NP \cap coNP$)
- it is worthwile to consider preorders (P)BESs:
  - speeds up computation of solution
  - facilitates reasoning about (P)BES transformations
- consistent consequence preorder & consistent correlation
  - difficult to compute on general BESs
  - facilitate proving correctness of: invariants, enumeration, abstraction
- abstraction techniques:
  - not limited to just safety or liveness properties
  - efficient (combination of model and property in PBESs)