Algorithms for Model Checking (2IW55)

Lecture 5

Boolean Equation Systems

Background material: Chapter 3 and 6 of
A. Mader, "Verification of Modal Properties using Boolean Equation Systems", Ph.D. thesis, 1997

Tim Willemse
(timw@win.tue.nl)
http://www.win.tue.nl/∼timw
MF 7.073

Technische Universiteit Eindhoven University of Technology

Outline

Boolean Equation Systems

- Boolean Equation Systems are systems of fixed point equations.

Given a set *Var* of propositional variables. A Boolean Expression is defined by:

$$f ::= X \mid \text{true} \mid \text{false} \mid f \wedge f \mid f \vee f$$

A Boolean Equation is an equation of the form $\mu X = f$ or $\nu X = f$ where $X \in Var$ and f is a Boolean Expression.

A Boolean Equation System is a sequence of Boolean Equations:

$$\mathcal{E} ::= \varepsilon \mid (\mu X = f)\ \mathcal{E} \mid (\nu X = f)\ \mathcal{E}$$

Note:

- Negation is not allowed, in order to ensure monotonicity.
- The order of equations is important. The leftmost sign will be given priority.

TU/e Technische Universiteit Eindhoven University of Technology

Boolean Equation Systems

- A variable W that occurs in a Boolean Expression of a BES $\mathcal{E}$ is called bound, if there is an equation for W in $\mathcal{E}$, otherwise W is called free.
- If propositional variables are bound uniquely (i.e., at most once), the BES is well-formed; we only consider well-formed BESs.
- If $\mathcal{E}$ contains no free variables, $\mathcal{E}$ is closed, otherwise it is open.
- Henceforth, σ represents either μ or ν if we wish to abstract from its actual polarity.

Example

An example of a closed BES $\mathcal{E}$ with three propositional variables X, Y and Z:

$$(\mu X = (X \wedge Y) \vee Z)\ (\nu Y = X \wedge Y)\ (\mu Z = Z \wedge X)$$

An example of an open BES $\mathcal{F}$ with three propositional variables X, Y and Z:

$$(\mu X = Y \vee Z)\ (\nu Y = X \wedge Y)$$

An example of a BES that is not well-formed:

$$(\mu X = X)\ (\nu X = X)$$

Boolean Equation Systems

- Let *Val* be the set of all functions $\eta : \mathit{Var} \to \{\text{false}, \text{true}\}$
- The solution of a BES is a valuation: $\eta : \mathit{Val}$
- Let $[f](\eta)$ denote the value of boolean expression f under valuation η.
- For the solution η of a BES $\mathcal{E}$, we wish $\eta(X) = [f](\eta)$ for all equations $\sigma X = f$ in $\mathcal{E}$.
- Also, we want the smallest (for μ) or greatest (for ν) solution, where leftmost fixed point signs take priority over fixed point signs that follow.

Given a BES $\mathcal{E}$, we define $[\![\mathcal{E}]\!] : \mathit{Val} \to \mathit{Val}$ by recursion on $\mathcal{E}$.

$$\begin{cases} [\![\varepsilon]\!](\eta) & := \eta \\ [\![(\mu X = f)\ \mathcal{E}]\!](\eta) & := [\![\mathcal{E}]\!](\eta[X := [f](\eta_\mu)]) \text{ where } \eta_\mu := [\![\mathcal{E}]\!](\eta[X := \text{false}]) \\ [\![(\nu X = f)\ \mathcal{E}]\!](\eta) & := [\![\mathcal{E}]\!](\eta[X := [f](\eta_\nu)]) \text{ where } \eta_\nu := [\![\mathcal{E}]\!](\eta[X := \text{true}]) \end{cases}$$

Note: for closed BESs we have $[\![\mathcal{E}]\!](\eta)(X) = [\![\mathcal{E}]\!](\eta')(X)$ for all η, η' and all bound X

Outline

Model Checking using BESs

Transformation of the μ-calculus model checking problem to BES

- Given is the following model checking problem: $M, s \models \sigma X.\ f$
  - a closed μ-calculus formula $\sigma X.\ f$ in Positive Normal Form and,
  - a Mixed Kripke Structure $M = \langle S, s_0, Act, R, L \rangle$.
  - $s \in S$ is a state
- We define a BES $\mathcal{E}$ with the following property:

$$([\![\mathcal{E}]\!](\eta))(X_s) = \text{true iff } M, s \models \sigma X.\ f$$

  i.e. formula $\sigma X.\ f$ holds in state s if and only if the solution for X_s yields true.
- This BES is defined as follows:
  - For each subformula $\sigma' Y.g$, we add the following equation for each state $s \in S$:

$$\sigma' Y_s \ = \ RHS(s, g)$$

  - Important: The order of the equations respects the subterm ordering in the original formula $\sigma X.\ f$.

Model Checking using BESs

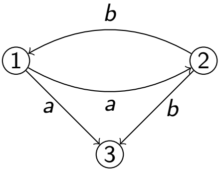
The Right-Hand Side of an equation is defined inductively on the structure of the μ-calculus formula:

$$
\begin{array}{lcl}
RHS(s, \mathrm{true}) & = & \mathrm{true} \\
RHS(s, \mathrm{false}) & = & \mathrm{false} \\
RHS(s, p) & = & \begin{cases} \mathrm{true} & \text{if } p \in L(s) \\ \mathrm{false} & \text{otherwise} \end{cases} \\
RHS(s, X) & = & X_s \\
 & & \\
RHS(s, f \wedge g) & = & RHS(s, f) \wedge RHS(s, g) \\
RHS(s, f \vee g) & = & RHS(s, f) \vee RHS(s, g) \\
 & & \\
RHS(s, [a]f) & = & \bigwedge_{t \in S} \{RHS(t, f) \mid s \xrightarrow{a} t\} \\
RHS(s, \langle a \rangle f) & = & \bigvee_{t \in S} \{RHS(t, f) \mid s \xrightarrow{a} t\} \\
 & & \\
RHS(s, \mu X.\ f) & = & X_s \\
RHS(s, \nu X.\ f) & = & X_s
\end{array}
$$

conventions: $\bigwedge_{t \in S} \emptyset = \mathrm{true}$ and $\bigvee_{t \in S} \emptyset = \mathrm{false}$

Technische Universiteit Eindhoven University of Technology

Example

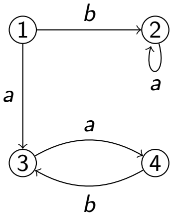


- $RHS(1,\ [a]X) = RHS(2,\ X) \wedge RHS(3,\ X) = X_2 \wedge X_3$.
- $RHS(2,\ \langle b\rangle Y) = RHS(1,\ Y) \vee RHS(3,\ Y) = Y_1 \vee Y_3$.
- $RHS(3, \langle b\rangle Y) = \text{false}$ (empty disjunction!)
- $$\begin{array}{rl} & RHS(1,\ [a]\langle b\rangle \mu Z.\ Z) \\ = & RHS(2,\ \langle b\rangle \mu Z.\ Z) \wedge RHS(3,\ \langle b\rangle \mu Z.\ Z) \wedge \\ = & (RHS(1,\ \mu Z.Z) \vee RHS(3,\ \mu Z.Z)) \wedge \text{false} \\ = & (Z_1 \vee Z_3) \wedge \text{false} \end{array}$$
- Translation of $\mu X.\langle b\rangle \text{true} \vee \langle a\rangle X$ to BES:

$$(\mu X_1 = X_3 \vee X_2)\ (\mu X_2 = \text{true})\ (\mu X_3 = \text{false})$$

TU/e Technische Universiteit Eindhoven University of Technology

Example



μ-calculus formula: $\nu X.\big([a]X \wedge \nu Y.\mu Z.(\langle b\rangle Y \vee \langle a\rangle Z)\big)$
Translates to the following BES:

$$
\begin{array}{lcl}
\nu X_1 & = & X_3 \wedge Y_1 \\
\nu X_2 & = & X_2 \wedge Y_2 \\
\nu X_3 & = & X_4 \wedge Y_3 \\
\nu X_4 & = & \text{true} \wedge Y_4 \\
\nu Y_1 & = & Z_1 \\
\nu Y_2 & = & Z_2 \\
\nu Y_3 & = & Z_3 \\
\nu Y_4 & = & Z_4 \\
\mu Z_1 & = & Y_2 \vee Z_3 \\
\mu Z_2 & = & \text{false} \vee Z_2 \\
\mu Z_3 & = & \text{false} \vee Z_4 \\
\mu Z_4 & = & Y_3 \vee \text{false}
\end{array}
$$

Outline

Solving BESs

- We reduced the model checking problem $M, s \models f$ to the solution of a BES with $\mathcal{O}(|M| \times |f|)$ equations.
- We now want a fast procedure to solve such BESs.
- An extremely tedious way to solve a BES is to unfold its semantics.
- A very appealing solution is to solve it by Gauß Elimination.

TU/e Technische Universiteit Eindhoven University of Technology

Solving BESs

Gauß Elimination uses the following 4 basic operations to solve a BES:

- local solution: eliminate X in its defining equation:

$$\begin{array}{lll}\mathcal{E}_0\ (\mu X = f)\ \mathcal{E}_1 & \text{becomes} & \mathcal{E}_0\ (\mu X = f[X := \text{false}])\ \mathcal{E}_1 \\ \mathcal{E}_0\ (\nu X = f)\ \mathcal{E}_1 & \text{becomes} & \mathcal{E}_0\ (\nu X = f[X := \text{true}])\ \mathcal{E}_1\end{array}$$

- Substitute definitions to the left:

$$\begin{array}{rl} & \mathcal{E}_0\ (\sigma_1 X = X \vee Y)\ \mathcal{E}_1\ (\sigma_2 Y = Y \wedge X)\ \mathcal{E}_2 \\ \text{becomes:} & \mathcal{E}_0\ (\sigma_1 X = X \vee (Y \wedge X))\ \mathcal{E}_1\ (\sigma_2 Y = Y \wedge X)\ \mathcal{E}_2\end{array}$$

- Substitute closed equations to the right:

$$\begin{array}{rl} & \mathcal{E}_0\ (\sigma_1 X = \text{true})\ \mathcal{E}_1\ (\sigma_2 Y = Y \wedge X)\mathcal{E}_2 \\ \text{becomes:} & \mathcal{E}_0\ (\sigma_1 X = \text{true})\ \mathcal{E}_1\ (\sigma_2 Y = Y \wedge \text{true})\ \mathcal{E}_2\end{array}$$

- Boolean simplication: At least the following:

$$b \wedge \text{true} \rightarrow b \qquad b \vee \text{true} \rightarrow \text{true} \qquad b \wedge \text{false} \rightarrow \text{false} \qquad b \vee \text{false} \rightarrow b$$

Solving BESs

Example

$$(\mu X = X \vee Y)\ (\nu Y = X \vee (Y \wedge Z))\ (\mu Z = Y \wedge Z)$$

local $\rightarrow$

$$(\mu X = \text{false} \vee Y)\ (\nu Y = X \vee (\text{true} \wedge Z))\ (\mu Z = Y \wedge \text{false})$$

simplifications $\rightarrow$

$$(\mu X = Y)\ (\nu Y = X \vee Z))\ (\mu Z = \text{false})$$

substitution backwards $\rightarrow$

$$(\mu X = Y)\ (\nu Y = X \vee \text{false})\ (\mu Z = \text{false})$$

simplifications $\rightarrow$

$$(\mu X = Y)\ (\nu Y = X)\ (\mu Z = \text{false})$$

substitution backwards $\rightarrow$

$$(\mu X = X)\ (\nu Y = X)\ (\mu Z = \text{false})$$

local $\rightarrow$

$$(\mu X = \text{false})\ (\nu Y = X)\ (\mu Z = \text{false})$$

substitution to the right $\rightarrow$

$$(\mu X = \text{false})\ (\nu Y = \text{false})\ (\mu Z = \text{false})$$

Solving BESs

Gauß Elimination is a decision procedure for computing the solution to a BES.

Input: a BES $(\sigma_1 X_1 = f_1)$ ... $(\sigma_n X_n = f_n)$. Returns: the solution for X_1.

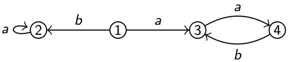
for $i = n$ **downto** 1 **do**
  if $\sigma_i = \mu$ **then** $f_i := f_i[X_i := \text{false}]$
  else $f_i := f_i[X_i := \text{true}]$
  end if
  for $j = i - 1$ **downto** 1 **do** $f_j := f_j[X_i := f_i]$
  end for
end for

Note:

- Invariants of the outer loop:
  - f_i contains only variables X_j with $j \leq i$.
  - for all $i < j \leq n$, X_j does not occur in f_j.
- Upon termination ($i = 0$), $\sigma_1 X_1 = f_1$ is closed and evaluates to true or false.
- One could substitute the solution for X_1 to the right and repeat the procedure to solve X_2, *etcetera*.

Technische Universiteit Eindhoven University of Technology

Example



Encoding the μ-calculus formula: $\nu X.([a]X \wedge \nu Y.\mu Z.(\langle b\rangle Y \vee \langle a\rangle Z))$ leads to the below BES; solving using Gauß Elimination (each column is one iteration of the algorithm):

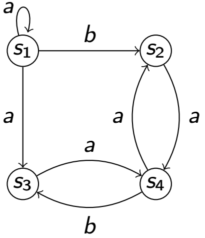
νX_1	$=$	$X_3 \wedge Y_1$	$X_3 \wedge Y_1$	$X_3 \wedge Y_1$	$X_3 \wedge Y_1$	$X_3 \wedge Y_1$	$X_3 \wedge Y_1$	$\cdots$	true
νX_2	$=$	$X_2 \wedge Y_2$	$X_2 \wedge Y_2$	$X_2 \wedge Y_2$	$X_2 \wedge Y_2$	$X_2 \wedge Y_2$	$X_2 \wedge Y_2$	$\cdots$	false
νX_3	$=$	$X_4 \wedge Y_3$	$X_4 \wedge Y_3$	$X_4 \wedge Y_3$	$X_4 \wedge Y_3$	$X_4 \wedge Y_3$	$X_4 \wedge Y_3$	$\cdots$	true
νX_4	$=$	Y_4	Y_4	Y_4	Y_4	Y_4	Y_3	$\cdots$	true
νY_1	$=$	Z_1	Z_1	Z_1	Z_1	$Y_2 \vee Y_3$	$Y_2 \vee Y_3$	$\cdots$	true
νY_2	$=$	Z_2	Z_2	Z_2	false	false	false	$\cdots$	false
νY_3	$=$	Z_3	Z_3	Y_3	Y_3	Y_3	Y_3	$\cdots$	true
νY_4	$=$	Z_4	Y_3	Y_3	Y_3	Y_3	Y_3*	$\cdots$	true
μZ_1	$=$	$Y_2 \vee Z_3$	$Y_2 \vee Z_3$	$Y_2 \vee Y_3$	$Y_2 \vee Y_3$	$Y_2 \vee Y_3*$	$Y_2 \vee Y_3*$	$\cdots$	true
μZ_2	$=$	Z_2	Z_2	Z_2	false$*$	false$*$	false$*$	$\cdots$	false
μZ_3	$=$	Z_4	Y_3	Y_3*	Y_3*	Y_3*	Y_3*	$\cdots$	true
μZ_4	$=$	Y_3	Y_3*	Y_3*	Y_3*	Y_3*	Y_3*	$\cdots$	true

Technische Universiteit Eindhoven University of Technology

Complexity of Gauß Elimination.

- Note that in $\mathcal{O}(n^2)$ substitutions, we obtain the final answer for X_1.
- However, f_1 can have $\mathcal{O}(2^n)$ different copies of e_n as subterms, so intermediate expressions could become exponentially big.
- Practical efficiency increases a lot if one keeps all intermediate terms simplified all the time.
- Gauß Elimination can be sped up if a forward dependency analysis is conducted (so-called local model checking).
- Precise efficiency depends heavily on the set of simplification rules.
- Precise complexity of solving Boolean Equation Systems is still unknown.
- Complexity of Gauß Elimination is independent of the alternation depth (see Proposition 6.4 [Mader]).

Outline



Consider the following μ-Calculus formula f:

$$\nu X.([a]X \wedge \nu Y.\mu Z.(\langle b\rangle Y \vee \langle a\rangle Z))$$

- Use the Emerson-Lei algorithm for computing whether $M, s_1 \models f$.
- Translate the model checking question $M \models f$ to a BES; indicate how $M, s \models \phi$ corresponds to the variables in the BES.
- Solve the BES by Gauß Elimination.