Algorithms for Model Checking (2IW55)

Lecture 10

Parameterised Boolean Equation Systems (2)

Background material:

Model Checking Processes with Data,
J.F. Groote and T.A.C. Willemse (*Sc. Comp. Progr. 2005*)

Proof Graphs for Parameterised Boolean Equation Systems,
S. Cranen, B. Luttik and T.A.C. Willemse (*CONCUR 2013*)

Tim Willemse
(t.a.c.willemse@tue.nl)
http://www.win.tue.nl/∼timw
MF 6.073

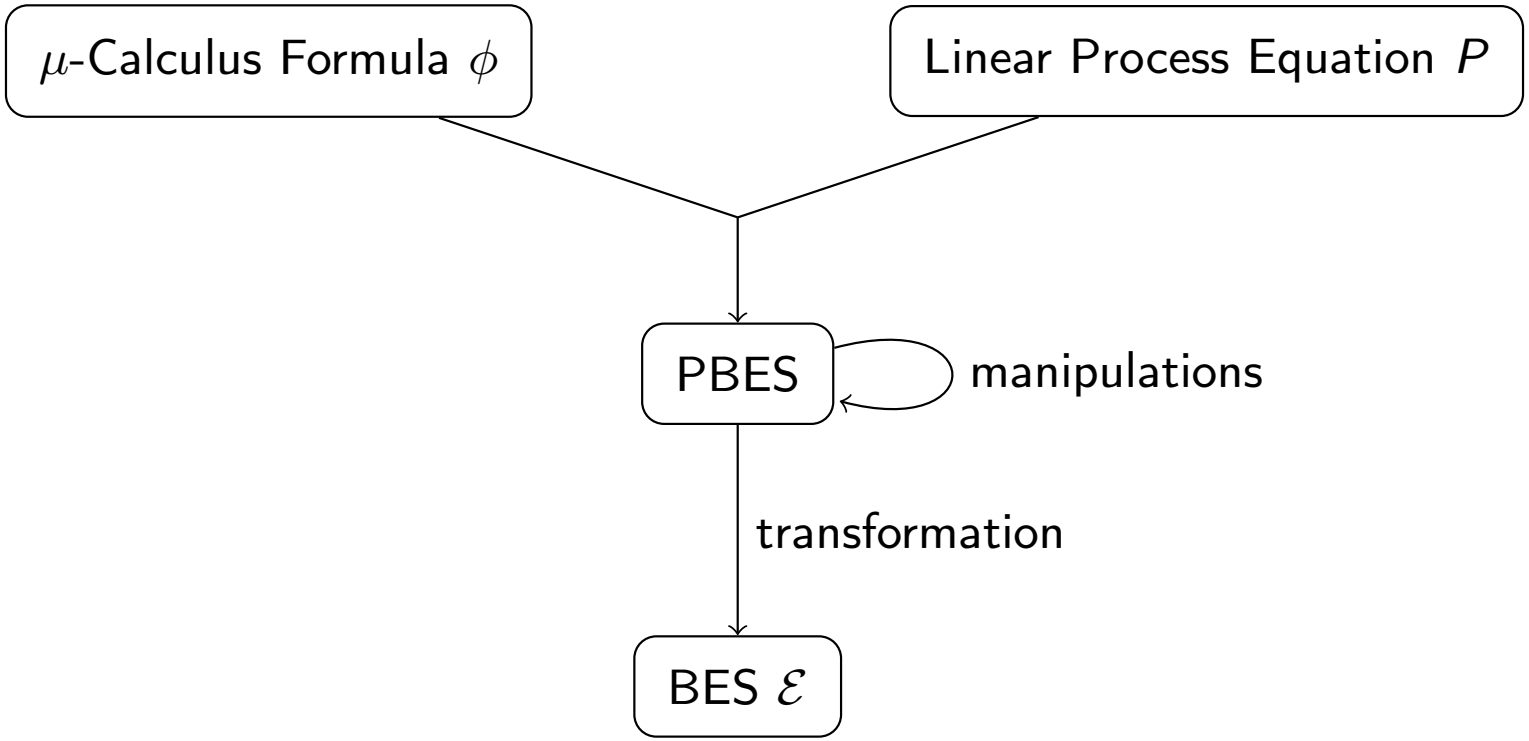
Department of Mathematics and Computer Science

TU/e Technische Universiteit Eindhoven University of Technology

Outline

Verification via PBESs

Verification Methodology:



Solving $\mathcal{E}$ answers $P \models \phi$

Problem Description

1. Given a process $X(e)$ described by an LPE X over Act
2. Given a first-order modal μ-calculus formula ϕ
3. Given environments η, ε
4. Check whether $X(e) \models \phi$ holds, where:

$$X(e) \models \phi \text{ iff } e \in [\![\phi]\!]\eta\varepsilon$$

- Decidable for finite data types
  - Compute LTS $[\![X(e)]\!]$
  - Evaluate ϕ on $[\![X(e)]\!]$ using standard model checking algorithms
- In general undecidable
- Transform problem to Parameterised Boolean Equation Systems (PBESs)

Parameterised Boolean Equation Systems

Grammar for predicate formulae

$$\phi, \psi ::= b \mid X(e) \mid \phi \wedge \psi \mid \phi \vee \psi \mid \forall d : D.\phi \mid \exists d : D.\phi$$

- b is a boolean expression ..... $n + m \geq 5$
- $X \in \mathcal{P}$ is a sorted predicate variable (or *relation*) ..... $X{:}2^D$
- e is an expression of sort D
- Interpreting ϕ requires two environments ..... ε (for data) and $\eta{:}\mathcal{P} \to 2^D$

Parameterised Boolean Equation Systems

Grammar for predicate formulae

$$\phi, \psi ::= b \mid X(e) \mid \phi \wedge \psi \mid \phi \vee \psi \mid \forall d : D.\phi \mid \exists d : D.\phi$$

- b is a boolean expression ..... $n + m \geq 5$
- $X \in \mathcal{P}$ is a sorted predicate variable (or *relation*) ..... $X{:}2^D$
- e is an expression of sort D
- Interpreting ϕ requires two environments ..... ε (for data) and $\eta{:}\mathcal{P} \to 2^D$

$$[\![b]\!]\eta\varepsilon = \begin{cases} \text{true} & \text{if } \varepsilon(b) \\ \text{false} & \text{else} \end{cases}$$

Parameterised Boolean Equation Systems

Grammar for predicate formulae

$$\phi, \psi ::= b \mid X(e) \mid \phi \wedge \psi \mid \phi \vee \psi \mid \forall d : D.\phi \mid \exists d : D.\phi$$

- b is a boolean expression ..... $n + m \geq 5$
- $X \in \mathcal{P}$ is a sorted predicate variable (or *relation*) ..... $X{:}2^D$
- e is an expression of sort D
- Interpreting ϕ requires two environments ..... ε (for data) and $\eta{:}\mathcal{P} \to 2^D$

$$[\![b]\!]\eta\varepsilon = \begin{cases} \text{true} & \text{if } \varepsilon(b) \\ \text{false} & \text{else} \end{cases} \qquad [\![X(e)]\!]\eta\varepsilon = \begin{cases} \text{true} & \text{if } \varepsilon(e) \in \eta(X) \\ \text{false} & \text{else} \end{cases}$$

Parameterised Boolean Equation Systems

Grammar for predicate formulae

$$\phi, \psi ::= b \mid X(e) \mid \phi \wedge \psi \mid \phi \vee \psi \mid \forall d : D.\phi \mid \exists d : D.\phi$$

- b is a boolean expression ..... $n + m \geq 5$
- $X \in \mathcal{P}$ is a sorted predicate variable (or *relation*) ..... $X{:}2^D$
- e is an expression of sort D
- Interpreting ϕ requires two environments ..... ε (for data) and $\eta{:}\mathcal{P} \to 2^D$

$$[\![b]\!]\eta\varepsilon = \begin{cases} \text{true} & \text{if } \varepsilon(b) \\ \text{false} & \text{else} \end{cases} \qquad [\![X(e)]\!]\eta\varepsilon = \begin{cases} \text{true} & \text{if } \varepsilon(e) \in \eta(X) \\ \text{false} & \text{else} \end{cases}$$

$$[\![\phi \wedge \psi]\!]\eta\varepsilon = [\![\phi]\!]\eta\varepsilon \text{ and } [\![\psi]\!]\eta\varepsilon \qquad [\![\phi \vee \psi]\!]\eta\varepsilon = [\![\phi]\!]\eta\varepsilon \text{ or } [\![\psi]\!]\eta\varepsilon$$

Parameterised Boolean Equation Systems

Grammar for predicate formulae

$$\phi, \psi ::= b \mid X(e) \mid \phi \wedge \psi \mid \phi \vee \psi \mid \forall d : D.\phi \mid \exists d : D.\phi$$

- b is a boolean expression ..... $n + m \geq 5$
- $X \in \mathcal{P}$ is a sorted predicate variable (or *relation*) ..... $X{:}2^D$
- e is an expression of sort D
- Interpreting ϕ requires two environments ..... ε (for data) and $\eta{:}\mathcal{P} \to 2^D$

$$[\![b]\!]\eta\varepsilon = \begin{cases} \text{true} & \text{if } \varepsilon(b) \\ \text{false} & \text{else} \end{cases} \qquad [\![X(e)]\!]\eta\varepsilon = \begin{cases} \text{true} & \text{if } \varepsilon(e) \in \eta(X) \\ \text{false} & \text{else} \end{cases}$$

$$[\![\phi \wedge \psi]\!]\eta\varepsilon = [\![\phi]\!]\eta\varepsilon \text{ and } [\![\psi]\!]\eta\varepsilon \qquad [\![\phi \vee \psi]\!]\eta\varepsilon = [\![\phi]\!]\eta\varepsilon \text{ or } [\![\psi]\!]\eta\varepsilon$$

$$[\![\forall d{:}D.\phi]\!]\eta\varepsilon = \text{for all } v{\in}D{:}\ [\![\phi]\!]\eta(\varepsilon[d := v]) \qquad [\![\exists d{:}D.\phi]\!]\eta\varepsilon = \text{for some } v{\in}D{:}\ [\![\phi]\!]\eta(\varepsilon[d := v])$$

Parameterised Boolean Equation Systems

A parameterised Boolean equation is an equation of the form $\sigma X(d : D) = \phi$

- σ is a least fixed point sign μ or a greatest fixed point sign ν.
- ϕ is a predicate formula, X a predicate variable
- a parameterised Boolean equation system is a sequence of such equations

- bound (bnd), free, well-formedness, open, close, rank as in BESs
- As in BESs, the order of equations is important.
- Assume, for simplicity, that all equations range over sort D only

Parameterised Boolean Equation Systems

The solution of a PBES $\mathcal{E}$ is an environment: $\eta : \mathcal{P} \to 2^D$;

We define $[\![\mathcal{E}]\!]\eta\varepsilon$ by recursion on $\mathcal{E}$.

$$\begin{cases} [\![\epsilon]\!]\eta\varepsilon & := \eta \\ [\![(\mu X(d:D) = \phi)\ \mathcal{E}]\!]\eta\varepsilon & := [\![\mathcal{E}]\!]\eta[X := \mu\Phi^{X,d}_{\mathcal{E},\eta,\varepsilon}]\varepsilon \\ [\![(\nu X(d:D) = \phi)\ \mathcal{E}]\!]\eta\varepsilon & := [\![\mathcal{E}]\!]\eta[X := \nu\Phi^{X,d}_{\mathcal{E},\eta,\varepsilon}]\varepsilon \end{cases}$$

Note: $\nu\Phi^{X,d}_{\mathcal{E},\theta,\varepsilon}$ is the greatest fixpoint to the following monotone functional:

$$\Phi^{X,d}_{\mathcal{E},\eta,\varepsilon}(S) := \{v \in D \mid [\![\phi]\!]([\![\mathcal{E}]\!]\eta[X := S]\varepsilon)\varepsilon[d := v]\}$$

Let:

- $\eta : \mathcal{P} \to 2^D$ be a predicate environment and ε a data environment
- $\mathsf{sig}(\mathcal{E}) = \{(X, v) \mid X \in \mathsf{bnd}(\mathcal{E}), v \in D\}$ be the signatures

Definition (Signature Environments)

Assume $S \subseteq \mathsf{sig}(\mathcal{E})$.

- S_{true} is the environment defined as $S_{\mathsf{true}}(X) = \{v \mid (X, v) \in S\}$ for all X
- S_{false} is the environment defined as $S_{\mathsf{false}}(X) = \{v \mid (X, v) \notin S\}$ for all X

Definition (Dependency Graphs)

Let b be a Boolean.
A structure $\langle S, R, L \rangle$ is a b-dependency graph for closed $\mathcal{E}$ and data environment ε if:

- $S \subseteq \mathsf{sig}(\mathcal{E})$
- $L(X, v) = \mathsf{rank}(X)$
- $R \subseteq S \times S$ such that: if for $\sigma X(d : D) = \phi$ in $\mathcal{E}$, $(X, v) \in S$ then
  - If $b = \mathsf{true}$, we require: $[\![\phi]\!]((X, v)^\bullet)_{\mathsf{true}}\varepsilon[d := v]$
  - If $b = \mathsf{false}$, we require: $\neg[\![\phi]\!]((X, v)^\bullet)_{\mathsf{false}}\varepsilon[d := v]$

Where

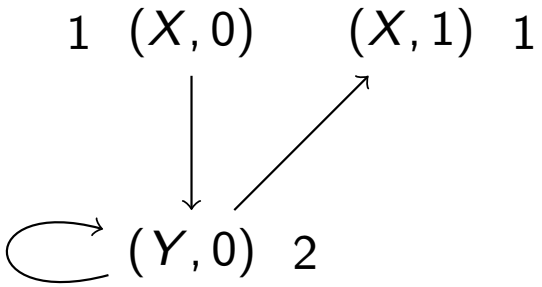
- $(X, v)^\bullet = \{(Z, w) \in S \mid (X, v)\ R\ (Z, w)\}$

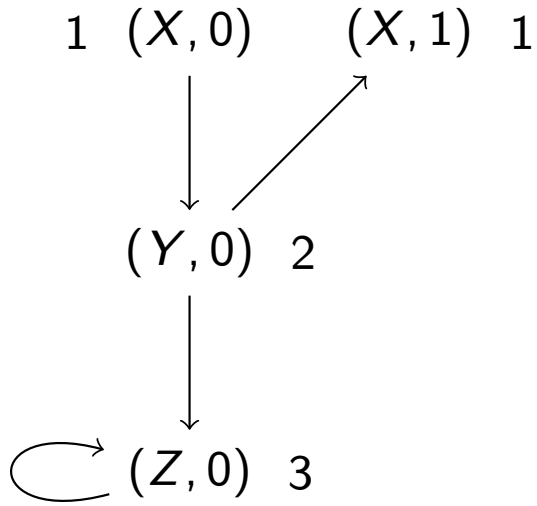
Example

Consider the following equation system $\mathcal{E}$.

$$\mu X(b : Bit) = Y(b) \vee b = 1$$
$$\nu Y(b : Bit) = Y(b) \vee (X(1) \wedge Z(b))$$
$$\mu Z(b : Bit) = Z(b)$$

Below are two true-dependency graphs $\langle S, R, L \rangle$ with $(X, 0) \in S$.
Note that $\text{sig}(\mathcal{E}) = \{(U, 0), (U, 1) \mid U = X, Y, Z\}$.





Definition (Proof Graphs)

A true-dependency graph $\langle S, R, L \rangle$ is a proof graph iff for all $s \in S$ and all infinite paths $\pi \in \text{path}(s)$:

$$\min\{r \mid \text{label } r \text{ occurs infinitely often on } \pi\} \text{ is even}$$

Theorem

For all closed PBESs $\mathcal{E}$ and all η, ε:

$$v \in [\![\mathcal{E}]\!]\eta\varepsilon(X) \textit{ iff there is a proof graph } \langle S, R, L \rangle \textit{ such that } (X, v) \in S$$

Example (Continued)

To see that the graph on the left satisfies the true-dependency graph property:

for $(X,0)$: $[\![Y(b) \vee b = 1]\!]\{(Y,0)\}_{\mathsf{true}}\delta[b := 0] = \mathsf{true}$

for $(X,1)$: $[\![Y(b) \vee b = 1]\!]\emptyset_{\mathsf{true}}\delta[b := 1] = \mathsf{true}$

for $(Y,0)$: $[\![Y(b) \vee X(1)]\!]\{(Y,0),(X,1)\}_{\mathsf{true}}]\delta[b := 0] = \mathsf{true}$

- Any infinite path goes through states with label 2, hence, it satisfies the proof graph property.
- Note that in this true-dependency graph, $(Y,0) \to (X,1)$ can be left out, because the right hand side of the equation for Y is disjunctive and the left disjunct is true.

Dually:

Definition (Refutation Graphs)

A false-dependency graph $\langle S, R, L \rangle$ is a refutation graph iff for all $s \in S$ and all infinite paths $\pi \in \text{path}(s)$:

$$\min\{r \mid \text{label } r \text{ occurs infinitely often on } \pi\} \text{ is odd}$$

Theorem

For all closed PBESs $\mathcal{E}$ and all η, ε:

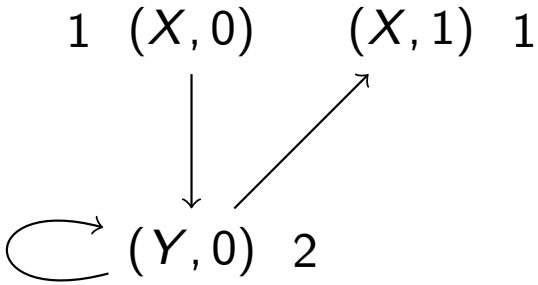
$$v \notin [\![\mathcal{E}]\!]\eta\varepsilon(X) \textit{ iff there is a refutation graph } \langle S, R, L \rangle \textit{ such that } (X, v) \in S$$

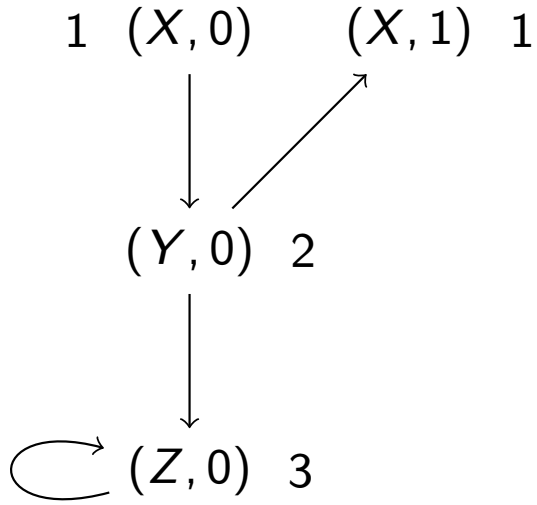
Example

Consider the following equation system $\mathcal{E}$.

$$\mu X(b : Bit) = Y(b) \vee b = 1$$
$$\nu Y(b : Bit) = Y(b) \vee (X(1) \wedge Z(b))$$
$$\mu Z(b : Bit) = Z(b)$$

Below are two true-dependency graphs $\langle S, R, L \rangle$ with $(X, 0) \in S$. The left one is a proof graph; the right one is not.





Outline

First-order Modal μ-Calculus model checking problem

- Given is a First-order Modal μ-Calculus formula $\sigma Z.\ \phi$
- Given a system described by an LPE $X(e)$

Compute whether $X(e) \models \sigma Z.\ \phi$

- Transform the model checking problem to solving a PBES $\mathcal{E}$
- The transformation is similar to the transformation to BES.
- Idea: for each fixed point subformula $\sigma' X.\ \psi$ of $\sigma Z.\phi$, add an equation

$$\sigma' \tilde{X}(d : D, \cdots) = RHS(\psi)$$

- The order of the equations respects the subterm ordering in $\sigma Z.\phi$
- Transformation is such that $X(e) \models \sigma Z.\ \phi$ iff $e \in [\![\mathcal{E}]\!]\eta\varepsilon(\tilde{Z})$

Verification via PBESs

- Identify a list of data variables bound outside the scope of a fixed point formula
- Given a formula ψ and some formal variable Z

Identify Bound Data Variables

$$Par(Z, b, l) = Par(Z, X, l) = []$$

$$Par(Z, \phi \wedge \psi, l) = Par(Z, \phi \vee \psi, l) = Par(Z, \phi, l) +\!\!+ Par(Z, \psi, l)$$

$$Par(Z, \forall d{:}D.\phi, l) = Par(Z, \exists d{:}D.\phi, l) = Par(Z, \phi, [d{:}D] +\!\!+ l)$$

$$Par(Z, [\alpha]\phi, l) = Par(Z, \langle\alpha\rangle\phi, l) = Par(Z, \phi, l)$$

$$Par(Z, \sigma X.\phi, l) = \begin{cases} l & \text{if } Z = X \\ Par(Z, \phi, l) & \text{otherwise} \end{cases}$$

Example

The one-place buffer system described by process B:

$$\begin{aligned} B(b : Bool, n : Nat) \quad &= \quad \sum_{m:Nat} \; b \longrightarrow r(m) \cdot B(\text{false}, m) \\ &+ \quad \neg b \longrightarrow s(n) \cdot B(\text{true}, n) \end{aligned}$$

- Property ψ: if the input stream is constant, so is the output stream:

$$\forall k : Nat.\ (\nu X.(\forall l : Nat.\ [r(l)](l = k \Rightarrow X) \wedge [s(l)](l = k \wedge X)))$$

- Transform ψ to a formula Ψ that starts with a dummy fixed point:

$$\nu A.\ \forall k : Nat.\ (\nu X.(\forall l : Nat.\ [r(l)](l = k \Rightarrow X) \wedge [s(l)](l = k \wedge X)))$$

- We have: $Par(A, \Psi, []) = []$ and $Par(X, \Psi, []) = [k : Nat]$

Verification via PBESs

- Let $\psi := \sigma Z.\ \phi$
- Given LPE $X(d{:}D) = \sum_{i \leq n} \sum_{e_i:D_i} c_i(d, e_i) \longrightarrow a_i(f_i(d, e_i)) \cdot X(g_i(d, e_i))$

Create Equation System Outline

$$\mathbf{E}(b) = \epsilon \qquad \mathbf{E}(Z) = \epsilon$$

$$\mathbf{E}(\phi \wedge \psi) = \mathbf{E}(\phi)\ \mathbf{E}(\psi) \qquad \mathbf{E}(\phi \vee \psi) = \mathbf{E}(\phi)\ \mathbf{E}(\psi)$$

$$\mathbf{E}(\forall d'{:}D'.\phi) = \mathbf{E}(\phi) \qquad \mathbf{E}(\exists d'{:}D'.\phi) = \mathbf{E}(\phi)$$

$$\mathbf{E}([\alpha]\phi) = \mathbf{E}(\phi) \qquad \mathbf{E}(\langle\alpha\rangle\phi) = \mathbf{E}(\phi)$$

$$\mathbf{E}(\sigma Z.\phi) = \left(\sigma \tilde{Z}(d{:}D, \mathit{Par}(Z, \psi, [])) = \mathbf{RHS}(\phi)\right)\ \mathbf{E}(\phi)$$

Example

Applying operator $\mathbf{E}$ on formula Ψ given the buffer process B:

$$
\begin{aligned}
& \mathbf{E}(\Psi) \\
= \quad & \mathbf{E}(\underline{\nu A.}\ \Psi_1) \\
= \quad & (\nu \tilde{A}(b : Bool, n : Nat) = \mathbf{RHS}(\Psi_1))\ \mathbf{E}(\Psi_1) \\
= \quad & (\nu \tilde{A}(b : Bool, n : Nat) = \mathbf{RHS}(\Psi_1))\ \mathbf{E}(\forall k : Nat.\ \Psi_2) \\
= \quad & (\nu \tilde{A}(b : Bool, n : Nat) = \mathbf{RHS}(\Psi_1))\ \mathbf{E}(\forall k : Nat.\ \Psi_2) \\
= \quad & (\nu \tilde{A}(b : Bool, n : Nat) = \mathbf{RHS}(\Psi_1))\ \mathbf{E}(\nu X.\Psi_3) \\
= \quad & (\nu \tilde{A}(b : Bool, n : Nat) = \mathbf{RHS}(\Psi_1)) \\
& (\nu \tilde{X}(b : Bool, n : Nat, k : Nat) = \mathbf{RHS}(\Psi_3))\ \mathbf{E}(\Psi_3) \\
= \quad & \ldots \\
& (\nu \tilde{A}(b : Bool, n : Nat) = \mathbf{RHS}(\Psi_1)) \\
& (\nu \tilde{X}(b : Bool, n : Nat, k : Nat) = \mathbf{RHS}(\Psi_3))
\end{aligned}
$$

So, $\mathbf{E}(\Psi)$ yields two equations.

Verification via PBESs

- Let $\psi := \sigma Y.\ \phi$
- Given LPE $X(d{:}D) = \sum_{i \leq n} \sum_{e_i:D_i} c_i(d, e_i) \longrightarrow a_i(f_i(d, e_i)) \cdot X(g_i(d, e_i))$

RHS:

Verification via PBESs

- Let $\psi := \sigma Y.\ \phi$
- Given LPE $X(d{:}D) = \sum_{i \leq n} \sum_{e_i:D_i} c_i(d, e_i) \longrightarrow a_i(f_i(d, e_i)) \cdot X(g_i(d, e_i))$

RHS:

$$\mathbf{RHS}(b) = b \qquad \mathbf{RHS}(Z) = \tilde{Z}(d, Par(Z, \psi, []))$$

Verification via PBESs

- Let $\psi := \sigma Y.\ \phi$
- Given LPE $X(d{:}D) = \sum_{i \leq n} \sum_{e_i:D_i} c_i(d, e_i) \longrightarrow a_i(f_i(d, e_i)) \cdot X(g_i(d, e_i))$

RHS:

$$\mathbf{RHS}(b) = b \qquad \mathbf{RHS}(Z) = \tilde{Z}(d, Par(Z, \psi, []))$$

$$\mathbf{RHS}(\phi \wedge \psi) = \mathbf{RHS}(\phi) \ \wedge \ \mathbf{RHS}(\psi) \qquad \mathbf{RHS}(\phi \vee \psi) = \mathbf{RHS}(\phi) \ \vee \ \mathbf{RHS}(\psi)$$

Verification via PBESs

- Let $\psi := \sigma Y.\ \phi$
- Given LPE $X(d{:}D) = \sum_{i \leq n} \sum_{e_i:D_i} c_i(d, e_i) \longrightarrow a_i(f_i(d, e_i)) \cdot X(g_i(d, e_i))$

RHS:

$$\mathbf{RHS}(b) = b \qquad \mathbf{RHS}(Z) = \tilde{Z}(d, Par(Z, \psi, []))$$

$$\mathbf{RHS}(\phi \wedge \psi) = \mathbf{RHS}(\phi) \ \wedge \ \mathbf{RHS}(\psi) \qquad \mathbf{RHS}(\phi \vee \psi) = \mathbf{RHS}(\phi) \ \vee \ \mathbf{RHS}(\psi)$$

$$\mathbf{RHS}(\forall d'{:}D'.\phi) = \forall d'{:}D'.\ \mathbf{RHS}(\phi) \qquad \mathbf{RHS}(\exists d'{:}D'.\phi) = \exists d'{:}D'.\ \mathbf{RHS}(\phi)$$

Verification via PBESs

- Let $\psi := \sigma Y.\ \phi$
- Given LPE $X(d{:}D) = \sum_{i \leq n} \sum_{e_i:D_i} c_i(d, e_i) \longrightarrow a_i(f_i(d, e_i)) \cdot X(g_i(d, e_i))$

RHS:

$$\mathbf{RHS}(b) = b \qquad \mathbf{RHS}(Z) = \tilde{Z}(d, Par(Z, \psi, []))$$

$$\mathbf{RHS}(\phi \wedge \psi) = \mathbf{RHS}(\phi) \ \wedge \ \mathbf{RHS}(\psi) \qquad \mathbf{RHS}(\phi \vee \psi) = \mathbf{RHS}(\phi) \ \vee \ \mathbf{RHS}(\psi)$$

$$\mathbf{RHS}(\forall d'{:}D'.\phi) = \forall d'{:}D'.\ \mathbf{RHS}(\phi) \qquad \mathbf{RHS}(\exists d'{:}D'.\phi) = \exists d'{:}D'.\ \mathbf{RHS}(\phi)$$

$$\mathbf{RHS}(\sigma Z.\phi) = \tilde{Z}(d, Par(Z, \psi, []))$$

Verification via PBESs

- Let $\psi := \sigma Y.\ \phi$
- Given LPE $X(d{:}D) = \sum_{i \leq n} \sum_{e_i:D_i} c_i(d, e_i) \longrightarrow a_i(f_i(d, e_i)) \cdot X(g_i(d, e_i))$

RHS:

$$\mathbf{RHS}(b) = b \qquad \mathbf{RHS}(Z) = \tilde{Z}(d, Par(Z, \psi, []))$$

$$\mathbf{RHS}(\phi \wedge \psi) = \mathbf{RHS}(\phi) \ \wedge \ \mathbf{RHS}(\psi) \qquad \mathbf{RHS}(\phi \vee \psi) = \mathbf{RHS}(\phi) \ \vee \ \mathbf{RHS}(\psi)$$

$$\mathbf{RHS}(\forall d'{:}D'.\phi) = \forall d'{:}D'.\ \mathbf{RHS}(\phi) \qquad \mathbf{RHS}(\exists d'{:}D'.\phi) = \exists d'{:}D'.\ \mathbf{RHS}(\phi)$$

$$\mathbf{RHS}(\sigma Z.\phi) = \tilde{Z}(d, Par(Z, \psi, []))$$

$$\mathbf{RHS}(\langle \alpha \rangle \phi) = \bigvee_{i \leq n} \exists e_i{:}D_i.\ \Big(c_i(d, e_i) \wedge a_i(f_i(d, e_i)) \text{ in } \alpha \wedge ((\mathbf{RHS}(\phi))[d := g_i(d, e_i)]) \Big)$$

Verification via PBESs

- Let $\psi := \sigma Y.\ \phi$
- Given LPE $X(d{:}D) = \sum_{i \leq n} \sum_{e_i:D_i} c_i(d, e_i) \longrightarrow a_i(f_i(d, e_i)) \cdot X(g_i(d, e_i))$

RHS:

$$\mathbf{RHS}(b) = b \qquad \mathbf{RHS}(Z) = \tilde{Z}(d, Par(Z, \psi, []))$$

$$\mathbf{RHS}(\phi \wedge \psi) = \mathbf{RHS}(\phi) \ \wedge \ \mathbf{RHS}(\psi) \qquad \mathbf{RHS}(\phi \vee \psi) = \mathbf{RHS}(\phi) \ \vee \ \mathbf{RHS}(\psi)$$

$$\mathbf{RHS}(\forall d'{:}D'.\phi) = \forall d'{:}D'.\ \mathbf{RHS}(\phi) \qquad \mathbf{RHS}(\exists d'{:}D'.\phi) = \exists d'{:}D'.\ \mathbf{RHS}(\phi)$$

$$\mathbf{RHS}(\sigma Z.\phi) = \tilde{Z}(d, Par(Z, \psi, []))$$

$$\mathbf{RHS}(\langle\alpha\rangle\phi) = \bigvee_{i \leq n} \exists e_i{:}D_i.\ \Big(c_i(d, e_i) \wedge a_i(f_i(d, e_i)) \text{ in } \alpha \wedge ((\mathbf{RHS}(\phi))[d := g_i(d, e_i)]) \Big)$$

$$\mathbf{RHS}([\alpha]\phi) = \bigwedge_{i \leq n} \forall e_i{:}D_i.\ \Big((c_i(d, e_i) \wedge a_i(f_i(d, e_i)) \text{ in } \alpha) \Rightarrow ((\mathbf{RHS}(\phi))[d := g_i(d, e_i)]) \Big)$$

Example (Verification of the Buffer process B, continued)

- Consider subformula $(\forall l : Nat.\ [r(l)](l = k \Rightarrow X) \wedge [s(l)](l = k \wedge X))$ of Ψ

$$
\begin{aligned}
& \mathbf{RHS}(\forall l : Nat.\ [r(l)](l = k \Rightarrow X) \wedge [s(l)](l = k \wedge X)) \\
= \quad & \forall l : Nat.\ \mathbf{RHS}([r(l)](l = k \Rightarrow X) \wedge [s(l)](l = k \wedge X)) \\
= \quad & \forall l : Nat.\ (\mathbf{RHS}([r(l)](l = k \Rightarrow X)) \wedge \mathbf{RHS}([s(l)](l = k \wedge X)))
\end{aligned}
$$

- Computing $\mathbf{RHS}([r(l)](l = k \Rightarrow X))$ requires process B.

$$
\begin{aligned}
& \mathbf{RHS}(([r(l)](l = k \Rightarrow X))) \\
= \quad & (\forall m : Nat.\ (b \wedge r(m) \text{ in } r(l)) \Rightarrow \mathbf{RHS}(l = k \Rightarrow X)[b := \text{false}, n := m]) \\
\wedge \quad & ((\neg b \wedge s(n) \text{ in } r(l)) \Rightarrow \mathbf{RHS}(l = k \Rightarrow X)[b := \text{true}, n := n]) \\
= \quad & (\forall m : Nat.\ (b \wedge r(m) \text{ in } r(l)) \Rightarrow (l = k \Rightarrow \tilde{X}(\text{false}, m, k))) \\
\wedge \quad & ((\neg b \wedge s(n) \text{ in } r(l)) \Rightarrow (l = k \Rightarrow \tilde{X}(\text{true}, n, k)))
\end{aligned}
$$

Matching parameterised actions with action formulae:

$$
\begin{array}{ll}
a(e) \text{ in } \mathsf{true} & = \mathsf{true} \\
a(e) \text{ in } a'(e') & = (a = a' \wedge e = e') \\
a(e) \text{ in } \neg\alpha & = \neg(a(e) \text{ in } \alpha) \\
a(e) \text{ in } (\alpha \wedge \beta) & = (a(e) \text{ in } \alpha) \ \wedge (a(e) \text{ in } \beta) \\
a(e) \text{ in } (\alpha \vee \beta) & = (a(e) \text{ in } \alpha) \ \vee (a(e) \text{ in } \beta)
\end{array}
$$

Observations:

- in yields a predicate formula
- in does not introduce predicate variables

Example

- The expression $r(m)$ in $r(l)$ yields $r = r \wedge m = l$, which simplifies to $m = l$
- The expression $s(n)$ in $r(l)$ yields $s = r \wedge n = l$, which simplifies to false

Example (Verification of the Buffer process, continued)

Buffer system and constant stream revisited

$$B(b : Bool, n : Nat) = \sum_{m:Nat} b \longrightarrow r(m) \cdot B(\text{false}, m) + \neg b \longrightarrow s(n) \cdot B(\text{true}, n)$$

Property Ψ: $\nu A.\ \forall k : Nat.\ (\nu X.(\forall l : Nat.\ [r(l)](l = k \Rightarrow X) \wedge [s(l)](l = k \wedge X)))$

Result after translation to PBES $\mathcal{E}$ (note: cleanup using ordinary first-order logic):

$$(\nu \tilde{A}(b : Bool, n : Nat) = \forall k : Nat.\ \tilde{X}(b, n, k))$$

$$(\nu \tilde{X}(b : Bool, n : Nat, k : Nat) = \forall l : Nat.\ ((\forall m : Nat.\ (b \wedge m = l) \Rightarrow (l = k \Rightarrow \tilde{X}(\text{false}, m, k))) \wedge ((\neg b \wedge n = l) \Rightarrow (l = k \wedge \tilde{X}(\text{true}, n, k)))))$$

For all $b : Bool$ and $n : Nat$, we have: $B(b, n) \models \Psi$ iff $(b, n) \in (\llbracket \mathcal{E} \rrbracket \theta\varepsilon)(\tilde{A}) = \text{true}$

Outline

How to solve PBESs

$$e \overset{?}{\in} X_i \text{ in } \mathcal{E} := \quad (\sigma_1 X_1(d_1 : D_1) = \phi_1) \cdots (\sigma_n X_n(d_n : D_n) = \phi_n)$$

Known techniques for solving/simplifying $\mathcal{E}$:

- Gauß Elimination on PBES + symbolic approximation of equations
- Instantiation to BES and subsequently solve the BES
- Using patterns
- Using under/over approximation
- Invariants

Definition (Logical Equivalence)

Let ϕ, ψ be two predicates. Then ψ is logically equivalent to ϕ, denoted $\phi \leftrightarrow \psi$ iff

$$\forall \varepsilon, \eta : \ [\![\phi]\!]\eta\varepsilon = [\![\psi]\!]\eta\varepsilon$$

- If $\phi \leftrightarrow \psi$, then equation $\nu X(d : D) = \phi$ has the same solution as $\nu X(d : D) = \psi$ (likewise for μ)
- Useful simplifications:
  - $\text{false} \wedge \phi \leftrightarrow \text{false}$
  - $\text{true} \vee \phi \leftrightarrow \text{true}$
  - if $d \notin \text{FV}(\phi)$, then $(\exists d : D.\ \phi) \leftrightarrow (\forall d : D.\ \phi) \leftrightarrow \phi$
  - One-point rule: $(\exists d : D. d = e \wedge \phi(d)) \leftrightarrow \phi(e)$
  - One-point rule: $(\forall d : D. d = e \Rightarrow \phi(d)) \leftrightarrow \phi(e)$
- Apply logical simplifications before applying PBES manipulations/solving techniques.

Gauß elimination on PBESs + Symbolic Approximation:

$$e \overset{?}{\in} X_i \text{ in } \mathcal{E} := \quad (\sigma_1 X_1(d_1 : D_1) = \phi_1) \cdots (\sigma_n X_n(d_n : D_n) = \phi_n)$$

- Local solution: eliminate X in its defining equation:

$$\mathcal{E}_0\ (\sigma X(d{:}D) = \phi)\ \mathcal{E}_1 \text{ becomes } \mathcal{E}_0\ (\sigma X(d{:}D) = X^\omega)\ \mathcal{E}_1$$

  - X^ω can be found by symbolic approximation:
  - $X^0 =$ false if $\sigma = \mu$, else $X^0 =$ true
  - $X^{n+1} = \phi[X := X^n]$
  - X^ω may require transfinite approximation; else $X^\omega = X^n$ for $X^n \leftrightarrow X^{n+1}$

Gauß elimination on PBESs + Symbolic Approximation:

$$e \overset{?}{\in} X_i \text{ in } \mathcal{E} := \quad (\sigma_1 X_1(d_1 : D_1) = \phi_1) \cdots (\sigma_n X_n(d_n : D_n) = \phi_n)$$

- Local solution: eliminate X in its defining equation:

$$\mathcal{E}_0\ (\sigma X(d{:}D) = \phi)\ \mathcal{E}_1 \text{ becomes } \mathcal{E}_0\ (\sigma X(d{:}D) = X^\omega)\ \mathcal{E}_1$$

  - X^ω can be found by symbolic approximation:
  - $X^0 =$ false if $\sigma = \mu$, else $X^0 =$ true
  - $X^{n+1} = \phi[X := X^n]$
  - X^ω may require transfinite approximation; else $X^\omega = X^n$ for $X^n \leftrightarrow X^{n+1}$

- Substitute definition backwards:

$$\begin{array}{rl} & \mathcal{E}_0\ (\sigma_1 X_1(d_1{:}D_1) = \phi_1)\ \mathcal{E}_1\ (\sigma_2 X_2(d_2{:}D_2) = \phi_2)\ \mathcal{E}_2 \\ \text{becomes:} & \mathcal{E}_0\ (\sigma_1 X_1(d_1{:}D_1) = \phi_1[X_2 := \phi_2])\ \mathcal{E}_1\ (\sigma_2 X_2(d_2{:}D_2) = \phi_2)\ \mathcal{E}_2 \end{array}$$

Gauß elimination on PBESs + Symbolic Approximation:

$$e \stackrel{?}{\in} X_i \text{ in } \mathcal{E} := \quad (\sigma_1 X_1(d_1 : D_1) = \phi_1) \cdots (\sigma_n X_n(d_n : D_n) = \phi_n)$$

- Local solution: eliminate X in its defining equation:

$$\mathcal{E}_0\ (\sigma X(d{:}D) = \phi)\ \mathcal{E}_1 \text{ becomes } \mathcal{E}_0\ (\sigma X(d{:}D) = X^\omega)\ \mathcal{E}_1$$

  - X^ω can be found by symbolic approximation:
  - $X^0 =$ false if $\sigma = \mu$, else $X^0 =$ true
  - $X^{n+1} = \phi[X := X^n]$
  - X^ω may require transfinite approximation; else $X^\omega = X^n$ for $X^n \leftrightarrow X^{n+1}$

- Substitute definition backwards:

$$\begin{array}{rl} & \mathcal{E}_0\ (\sigma_1 X_1(d_1{:}D_1) = \phi_1)\ \mathcal{E}_1\ (\sigma_2 X_2(d_2{:}D_2) = \phi_2)\ \mathcal{E}_2 \\ \text{becomes:} & \mathcal{E}_0\ (\sigma_1 X_1(d_1{:}D_1) = \phi_1[X_2 := \phi_2])\ \mathcal{E}_1\ (\sigma_2 X_2(d_2{:}D_2) = \phi_2)\ \mathcal{E}_2 \end{array}$$

- Substitute solved equations (i.e. not containing predicate variables) forward:

$$\begin{array}{rl} & \mathcal{E}_0\ (\sigma_1 X_1(d_1{:}D_1) = \phi_1)\ \mathcal{E}_1\ (\sigma_2 X_2(d_2{:}D_2) = \phi_2)\ \mathcal{E}_2 \\ \text{becomes:} & \mathcal{E}_0\ (\sigma_1 X_1(d_1{:}D_1) = \phi_1)\ \mathcal{E}_1\ (\sigma_2 X_2(d_2{:}D_2) = \phi_2[X_1 := \phi_1])\ \mathcal{E}_2 \end{array}$$

Solving PBESs

Example

PBES: $(\nu X(n : Nat) = n \leq 2 \wedge Y(n))\ (\mu Y(n : Nat) = \text{odd}(n) \vee X(n+1))$

1. Eliminate Y from $(\mu Y(n : Nat) = \text{odd}(n) \vee X(n+1))$ ..... done
2. Substitute definition of Y backwards:

$$\begin{array}{ll} & (\nu X(n : Nat) = n \leq 2 \wedge Y(n)) \\ \text{becomes} & (\nu X(n : Nat) = n \leq 2 \wedge (\text{odd}(n) \vee X(n+1))) \end{array}$$

3. Eliminate X from $(\nu X(n : Nat) = n \leq 2 \wedge (\text{odd}(n) \vee X(n+1)))$:

$$\begin{array}{lll} X^0 & \equiv \text{true} & \\ X^1 & \equiv n \leq 2 \wedge (\text{odd}(n) \vee \text{true}) & \leftrightarrow n \leq 2 \\ X^2 & \equiv n \leq 2 \wedge (\text{odd}(n) \vee n+1 \leq 2) & \leftrightarrow n \leq 2 \wedge (\text{odd}(n) \vee n \leq 1) \leftrightarrow n \leq 1 \\ X^3 & \equiv n \leq 2 \wedge (\text{odd}(n) \vee n+1 \leq 1) & \leftrightarrow n \leq 2 \wedge (\text{odd}(n) \vee n = 0) \leftrightarrow n \leq 1 \end{array}$$

So, solution to X is $n \leq 1$ (i.e., X semantically consists of the set $\{0, 1\}$)

Solving PBESs

Gauß Elimination terminates; symbolic approximation may not terminate

- Due to infinite data types, a transfinite approximation may be needed
- Evaluating predicates may be impossible: $\exists k, l, m : Nat . x^k + y^l = z^m$
- Theorem proving technology may be added in symbolic approximation

Outline

Exercise

Consider the lossy channel system described by the following LPE:

$$\begin{aligned} C(b : Bool, m : M) \quad &= \quad \sum_{k:M} b \longrightarrow r(k) \cdot C(\text{false}, k) \\ &+ \quad \neg b \longrightarrow s(m) \cdot C(\text{true}, m) \\ &+ \quad \neg b \longrightarrow l \cdot C(\text{true}, m) \end{aligned}$$

Action r stands for reading, s stands for sending and l stands for losing a message.

1. $\nu X.([\text{true}]X \wedge (\mu Y.[l]Y \wedge \forall m{:}M.[r(m)]Y \wedge \langle\text{true}\rangle\text{true}))$
2. $\nu X.\mu Y.\nu Z.(\forall m{:}M.[s(m)]X) \wedge ((\forall m{:}M.\ [s(m)]\text{false}) \vee ([l]Y \wedge \forall m{:}M.[r(m)]Y)) \wedge [l]Z \wedge \forall m{:}M.[r(m)]Z$

Questions:

- Explain the first formula in natural language
- Translate both formulae to PBESs given process C
- Use Gauß Elimination to solve the PBES
- For which initial states of C do the properties hold?