Algorithms for Model Checking (2IW55)

Lecture 9
Parameterised Boolean Equation Systems
Background material:
"Model-checking processes with data" and
"Parameterised Boolean Equation Systems",
J.F. Groote and T.A.C. Willemse

Tim Willemse
(timw@win.tue.nl)
http://www.win.tue.nl/~timw
HG 6.81

Symbolic System Specification

Today: System specification represented by Linear Process Equations:

- abstract data types for reasoning about data
  - data sorts ..... $Bool, Nat$
  - function symbols ..... $and : Bool \times Bool \to Bool$
  - equations ..... $and(x, \text{true}) = x$
- process algebra for reasoning about dynamic behaviour
  - parameterised atomic actions ..... $read(n), write(n)$
  - process operators ..... $+, \sum_{n:Nat}, \cdot$
  - parameterised recursion ..... $X(n : Nat) = a \cdot X(n) + \sum_{m:Nat} b(m) \cdot X(m)$

Linear Process Equation format

$$\begin{aligned} X(d:D) = & \sum_{e_1:D_1} c_1(d,e_1) \longrightarrow a_1(f_1(d,e_1)) \cdot X(g_1(d,e_1)) \\ & + \quad \ldots \\ & + \sum_{e_n:D_n} c_n(d,e_n) \longrightarrow a_n(f_n(d,e_n)) \cdot X(g_n(d,e_n)) \end{aligned}$$

- d is a vector of state variables

For every summand i:

- e_i is the vector of local variables
- c_i is the enabling condition; free variables in c_i are d and e_i
- $a_i \in Act$ is the action label; a_i carries parameters of sort D_{a_i}.
- f_i is the parameter for action a_i; free variables in f_i are d and e_i
- g_i is the next-state; free variables in c_i are d and e_i

Linear Process Equation format

$$\begin{aligned} X(d:D) = & \sum_{e_1:D_1} c_1(d,e_1) \longrightarrow a_1(f_1(d,e_1)) \cdot X(g_1(d,e_1)) \\ & + \quad \ldots \\ & + \sum_{e_n:D_n} c_n(d,e_n) \longrightarrow a_n(f_n(d,e_n)) \cdot X(g_n(d,e_n)) \end{aligned}$$

Semantics: $[X(e)]$ defines the Labelled Transition System $[X(e)] = \langle S, s_0, Act', \rightarrow \rangle$:

- $S = D$ is the state space
- $s_0 = e$ is the initial state
- $Act' = \{a_i(d) \mid 1 \leq i \leq n \wedge d \in D_{a_i}\}$ is the set of actions
- $d \xrightarrow{a} d'$ iff for some i: $\exists e_i{:}D_i.\ c_i(d,e_i) \wedge d' = g_i(d,e_i) \wedge a = a_i(f_i(d,e_i))$

Symbolic System Specification

Example

Consider the system $X(0, \text{true})$ given by the following LPE:

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

Intuition:

- if b holds, then an arbitrary natural number can be read through action r
- if $\neg b$ holds, then value n is sent through action s

Formally:

- State space: $Nat \times Bool$
- Transitions: for all $n, m \in Nat$:
  $(n, \text{true}) \xrightarrow{r(m)} (m, \text{false})$ and $(n, \text{false}) \xrightarrow{s(n)} (n, \text{true})$

Extended Hennessy-Milner Logic

- Hennessy-Milner Logic = μ-calculus - recursion
- Grammar:

$$\phi, \psi ::= \text{true} \mid \text{false} \mid \phi \wedge \psi \mid \phi \vee \psi \mid \langle a \rangle \phi \mid [a]\phi$$

Problem

Consider the process $X(0, \text{true})$, given by:

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

Specify that every natural number n can be read through action r.

- $\langle r(0) \rangle \text{true} \wedge \langle r(1) \rangle \text{true} \wedge \langle r(2) \rangle \text{true} \wedge \ldots$
- Hennessy-Milner formulae are finite: the above is not a Hennessy-Milner formula

Extended Hennessy-Milner Logic

Solving infinity problems:

- Introduce first-order quantification ..... $\forall d{:}D.\phi(d)$
- Inject data in modalities ..... $[a(d)]\phi$

Extended Hennessy-Milner formulae:

$$\phi, \psi ::= b \mid \phi \wedge \psi \mid \phi \vee \psi \mid \forall d{:}D.\phi \mid \exists d{:}D.\phi \mid \langle \alpha \rangle \phi \mid [\alpha]\phi$$

- b is a Boolean expression ..... e.g. $d + e \geq 5$
- d is a sorted data variable

α is an action formula:

$$\alpha, \beta ::= a(e) \mid \text{true} \mid \neg\alpha \mid \alpha \wedge \beta \mid \alpha \vee \beta$$

- e is a data expression for action label a

Extended Hennessy-Milner Logic

Example

Consider the process $X(0, \text{true})$, given by:

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

- Some number can be read ..... $\exists n{:}Nat.\ \langle r(n) \rangle \text{true}$
- Every natural number can be read ..... $\forall n{:}Nat.\ \langle r(n) \rangle \text{true}$
- Any number that is read can subsequently be sent ..... $\forall n{:}Nat.\ [r(n)]\langle s(n) \rangle \text{true}$
- Only odd numbers can be sent ..... $\forall n{:}Nat.\ [s(n)]odd(n)$

Extended Hennessy-Milner Logic — Semantics

Given LPE X with $[X(e)] = \langle S, s_0, Act', \rightarrow \rangle$

- a state formula ϕ characterises a set of states in S
- state formulae contain data variables . . . . . e.g. $n + 3 \geq m$
- values for data variables are given by an environment . . . . . e.g. $\varepsilon(n) = 5$

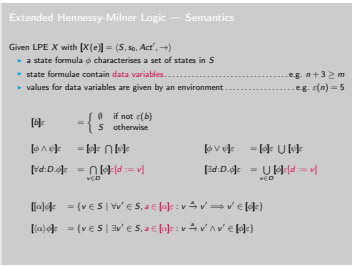
$$[b]\varepsilon = \begin{cases} \emptyset & \text{if not } \varepsilon(b) \\ S & \text{otherwise} \end{cases}$$

$$[\phi \wedge \psi]\varepsilon = [\phi]\varepsilon \cap [\psi]\varepsilon \qquad [\phi \vee \psi]\varepsilon = [\phi]\varepsilon \cup [\psi]\varepsilon$$

$$[\forall d{:}D.\phi]\varepsilon = \bigcap_{v \in D} [\phi]\varepsilon[d := v] \qquad [\exists d{:}D.\phi]\varepsilon = \bigcup_{v \in D} [\phi]\varepsilon[d := v]$$

$$[[\alpha]\phi]\varepsilon = \{v \in S \mid \forall v' \in S, a \in [\alpha]\varepsilon : v \xrightarrow{a} v' \Longrightarrow v' \in [\phi]\varepsilon\}$$

$$[\langle\alpha\rangle\phi]\varepsilon = \{v \in S \mid \exists v' \in S, a \in [\alpha]\varepsilon : v \xrightarrow{a} v' \wedge v' \in [\phi]\varepsilon\}$$



Example (Reconsider the example)

Consider the toggle $X(\text{true})$, given by:

$$X(b : Bool) = t(b) \cdot X(\neg b)$$

with $[X(\text{true})] = \langle S, s_0, Act, \rightarrow \rangle$.

state s satisfies $\exists c{:}Bool.\ \langle t(c)\rangle\text{true}$ iff s satisfies $\langle t(c)\rangle\text{true}$ for some value for c:

$$\begin{aligned} & [\exists c{:}Bool.\langle t(c)\rangle\text{true}]\varepsilon \\ = & \bigcup_{b \in \{\text{true},\text{false}\}} [\langle t(c)\rangle\text{true}]\varepsilon[c := b] \\ = & \bigcup_{b \in \{\text{true},\text{false}\}} \{v \in S \mid \exists v' \in S, a \in [t(c)]\varepsilon[c := b] : v \xrightarrow{a} v' \wedge v' \in [\text{true}]\varepsilon[c := b]\} \\ = & \bigcup_{a \in [\![t(c)]\!]\varepsilon[c:=\text{true}] \cup [\![t(c)]\!]\varepsilon[c:=\text{false}]} \{v \in S \mid v \xrightarrow{a}\} \end{aligned}$$

Extended Hennessy-Milner Logic — Semantics

Given LPE X with $[X(e)] = \langle S, s_0, Act', \rightarrow \rangle$

- an action formula characterises a set of actions

$$
\begin{aligned}
[a(e)]\varepsilon &= \{a(\varepsilon(e))\} \\
[\text{true}]\varepsilon &= Act' \\
[\neg\alpha]\varepsilon &= Act' \setminus [\alpha]\varepsilon \\
[\alpha \wedge \beta]\varepsilon &= [\alpha]\varepsilon \bigcap [\beta]\varepsilon \\
[\alpha \vee \beta]\varepsilon &= [\alpha]\varepsilon \bigcup [\beta]\varepsilon
\end{aligned}
$$

Examples

- any action but $read(3)$ . . . . . $\neg read(3)$
- any action other than $read(d)$, for quantified d . . . . . $\neg read(d)$

First-order Modal mu-Calculus

- First-order Modal mu-Calculus = Extended Hennessy-Milner logic + fixed points
- State formulae directly in Positive Normal Form:

$$
\begin{aligned}
\phi, \psi ::= \quad & b \mid \phi \vee \psi \mid \phi \wedge \psi \mid \exists d{:}D.\ \phi \mid \forall d{:}D.\ \phi \mid [\alpha]\phi \mid \langle\alpha\rangle\phi \mid \\
& Z \mid \mu Z.\ \phi \mid \nu Z.\ \phi
\end{aligned}
$$

- Z is a formal variable
- $\mu Z.\phi$ is the least fixed point; $\nu Z.\phi$ is the greatest fixed point
- α is an action formula (see extended Hennessy-Milner logic)

First-order Modal mu-Calculus

Given LPE X with $[X(e)] = \langle S, s_0, Act', \rightarrow \rangle$

- Extended Hennessy-Milner logic is interpreted in one environment ε
- First-order Modal μ-calculus requires two ..... ε (for data) and $\eta : Var \rightarrow 2^S$
- Semantics of ϕ is given by $[\phi]\eta\varepsilon$ ..... $\subseteq S$
- The set $(2^S, \subseteq)$ is a complete lattice

For formulae ϕ and variable Z, define $\Phi^Z_{\eta\varepsilon}(S') := [\phi]\ \eta[Z := S']\varepsilon$

- $\Phi^Z_{\eta\varepsilon}$ is monotone: $S' \subseteq T'$ implies $\Phi^Z_{\eta\varepsilon}(S') \subseteq \Phi^Z_{\eta\varepsilon}(T')$
- The existence of least and greatest fixed points of $\Phi^Z_{\eta\varepsilon}$ in $(2^S, \subseteq)$ is thus guaranteed
- Notation: $\nu\Phi^Z_{\eta\varepsilon}$ and $\mu\Phi^Z_{\eta\varepsilon}$

Semantics of First-order Modal μ-calculus formulae:

$$[Z]\eta\varepsilon = \eta(Z) \qquad [\nu Z.\ \phi]\eta\varepsilon = \nu\Phi^Z_{\eta\varepsilon} \qquad [\mu Z.\ \phi]\eta\varepsilon = \mu\Phi^Z_{\eta\varepsilon}$$

First-order Modal mu-Calculus

Example

1. Some action is enabled ..... $\langle \text{true} \rangle \text{true}$
2. Absence of deadlock ..... $\nu X.\ [\text{true}]X \wedge \langle \text{true} \rangle \text{true}$
3. Reading value n is always inevitably followed by sending value n
   $\nu X.\ \forall n{:}Nat.\ [\text{true}]X \wedge [r(n)]\mu Y.([\neg s(n)]Y \wedge \langle \text{true} \rangle \text{true})$
4. Every stream of numbers is increasing. $\nu X.\forall n{:}Nat.[r(n)](X \wedge \forall m{:}Nat.[r(m)](m > n))$

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 algorithms
- In general undecidable
- Transform problem to Parameterised Boolean Equation Systems (PBESs)

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 ..... $X{:}D \to Bool$
- e is an expression of sort D
- Interpreting ϕ requires two environments ..... ε (for data) and $\eta{:}\mathcal{P} \to (D \to Bool)$

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

$$[\![\phi \wedge \psi]\!]\eta\varepsilon = [\![\phi]\!]\eta\varepsilon \wedge [\![\psi]\!]\eta\varepsilon \qquad [\![\phi \vee \psi]\!]\eta\varepsilon = [\![\phi]\!]\eta\varepsilon \vee [\![\psi]\!]\eta\varepsilon$$

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

Parameterised Boolean Equation Systems

- The set of functions $f{:}D \to \mathit{Bool}$ is denoted Bool^D
- For $f, g \in \mathit{Bool}^D$, let $f \sqsubseteq g$ iff for all $d \in D$, $f(d) \Rightarrow g(d)$
- $(\mathit{Bool}^D, \sqsubseteq)$ is a complete lattice

To formulae ϕ, variables Z and d, associate $\Phi^Z_{\eta\varepsilon}(f) := \lambda v \in D.[\phi]\eta[Z := f]\varepsilon[d := v]$

- $\Phi^{Z,d}_{\eta\varepsilon}$ is monotone: $f \sqsubseteq g$ implies $\Phi^{Z,d}_{\eta\varepsilon}(f) \sqsubseteq \Phi^{Z,d}_{\eta\varepsilon}(g)$
- Least and greatest fixed points of $\Phi^{Z,d}_{\eta\varepsilon}$ in $(\mathit{Bool}^D, \sqsubseteq)$ are guaranteed to exist
- Least fixed point is denoted $\mu\Phi^{Z,d}_{\eta\varepsilon}$; dually: $\nu\Phi^{Z,d}_{\eta\varepsilon}$

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

Parameterised Boolean Equation Systems

- As in BESs, the order of equations is important.
- bounded, free, well-formedness, open, close as in BESs
- The solution of a PBES is an environment: $\eta : \mathcal{P} \to (D \to \mathit{Bool})$

Given a PBES $\mathcal{E}$, we define $[\mathcal{E}]\eta\varepsilon$ by recursion on $\mathcal{E}$.

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

Note: $\Phi^{X,d}_{\theta\varepsilon}$ is the monotone functional associated to ϕ, X and d

Teaser: Example Transformation

- Consider the process X, given by:

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

- $X(n, b) \models \nu Y.[\text{true}]Y \wedge \langle \text{true} \rangle \text{true}$ is encoded by the PBES:

$$\nu \bar{Y}(n : Nat, b : Bool) = \forall m : Nat.\ b \Longrightarrow \bar{Y}(m, \neg b) \wedge \neg b \Longrightarrow \bar{Y}(n, \neg b) \wedge ((\exists m : Nat.b) \vee (\neg b))$$

- for all n, b, $\bar{Y}(n, b) = \text{true}$ iff $X(n, b) \models \nu Y.[\text{true}]Y \wedge \langle \text{true} \rangle \text{true}$
- Intuition: the outcome should not depend on the value of n
- Reduced and simplified PBES:

$$\nu \bar{Y}(b : Bool) = (b \Longrightarrow \bar{Y}(\neg b)) \wedge (\neg b \Longrightarrow \bar{Y}(\neg b))$$

- Translates to the following BES: $(\nu \bar{Y}_{\text{true}} = \bar{Y}_{\text{false}})$ $(\nu \bar{Y}_{\text{false}} = \bar{Y}_{\text{true}})$

Outlook

Next week:

- Translate $X(d) \models \phi$ to a PBES
- Reduce complexity of PBES
- Procedures for solving PBESs

Exercise

Let X be an LPE with $Act = \{r, s, i\}$, where:

- r (read) and s (send) take natural numbers as parameters
- i (internal activity) is a parameterless action

Specify the following requirements in the First-order modal μ-calculus

- No infinite sequence of i actions is reachable
- In all states, every r action is inevitably followed by a s action
- In all states, every $r(n)$ action can eventually be followed by a $s(n)$ action
- There is a path on which infinitely many r actions occur

Explain the following requirements in natural language

- $\nu Y.[\text{true}]Y \wedge \forall n{:}Nat.[r(n)]\mu Z.[i]Z \wedge \langle s(n)\rangle\text{true}$
- $\nu Y.[i]Y \wedge \langle\text{true}\rangle\text{true}$