

Algorithms for Model Checking (2IW55)

Lecture 9

Parameterised Boolean Equation Systems

Background material:

Ploeger, Wesselink & Willemse: "Verification of reactive systems via instantiation of Parameterised Boolean Equation Systems" and

Groote & Willemse: "Parameterised Boolean Equation Systems"

Tim Willemse

(timw@win.tue.nl)

<http://www.win.tue.nl/~timw>

MF 6.073

Linear Process Equations

Extended Hennessy-Milner Logic

First-order Modal μ -Calculus

Parameterised Boolean Equation Systems

Linear Process Equation format

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

- ▶ 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
- ▶ $a_i \in \text{Act}$ 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 g_i are d and e_i

Linear Process Equation format

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

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

- ▶ $S = D$ 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 : $\exists e_i : D_i. c_i(d, e_i) \wedge d' = g_i(d, e_i) \wedge a = a_i(f_i(d, e_i))$

Example

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

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

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

Linear Process Equations

Extended Hennessy-Milner Logic

First-order Modal μ -Calculus

Parameterised Boolean Equation Systems

- ▶ 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:

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

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

- ▶ 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:

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

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

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

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

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

- ▶ values for data variables are given by an environment e.g. $\varepsilon(n) = 5$
- ▶ an action formula characterises a set of **actions**

$$\llbracket a(e) \rrbracket_{\varepsilon} = \{a(\varepsilon(e))\}$$

$$\llbracket \text{true} \rrbracket_{\varepsilon} = Act'$$

$$\llbracket \neg\alpha \rrbracket_{\varepsilon} = Act' \setminus \llbracket \alpha \rrbracket_{\varepsilon}$$

$$\llbracket \alpha \wedge \beta \rrbracket_{\varepsilon} = \llbracket \alpha \rrbracket_{\varepsilon} \cap \llbracket \beta \rrbracket_{\varepsilon}$$

$$\llbracket \alpha \vee \beta \rrbracket_{\varepsilon} = \llbracket \alpha \rrbracket_{\varepsilon} \cup \llbracket \beta \rrbracket_{\varepsilon}$$

Examples

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

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

- ▶ a state formula ϕ characterises a set of states in S
- ▶ state formulae contain **data variables** e.g. $n + 3 \geq m$

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

- ▶ a state formula ϕ characterises a set of states in S
- ▶ state formulae contain **data variables** e.g. $n + 3 \geq m$

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

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

- ▶ a state formula ϕ characterises a set of states in S
- ▶ state formulae contain **data variables** e.g. $n + 3 \geq m$

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

$$\llbracket \phi \wedge \psi \rrbracket_\varepsilon = \llbracket \phi \rrbracket_\varepsilon \cap \llbracket \psi \rrbracket_\varepsilon$$

$$\llbracket \phi \vee \psi \rrbracket_\varepsilon = \llbracket \phi \rrbracket_\varepsilon \cup \llbracket \psi \rrbracket_\varepsilon$$

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

- ▶ a state formula ϕ characterises a set of states in S
- ▶ state formulae contain **data variables** e.g. $n + 3 \geq m$

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

$$\llbracket \phi \wedge \psi \rrbracket_\varepsilon = \llbracket \phi \rrbracket_\varepsilon \cap \llbracket \psi \rrbracket_\varepsilon$$

$$\llbracket \phi \vee \psi \rrbracket_\varepsilon = \llbracket \phi \rrbracket_\varepsilon \cup \llbracket \psi \rrbracket_\varepsilon$$

$$\llbracket \forall d:D. \phi \rrbracket_\varepsilon = \bigcap_{v \in D} \llbracket \phi \rrbracket_\varepsilon[d := v]$$

$$\llbracket \exists d:D. \phi \rrbracket_\varepsilon = \bigcup_{v \in D} \llbracket \phi \rrbracket_\varepsilon[d := v]$$

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

- ▶ a state formula ϕ characterises a set of states in S
- ▶ state formulae contain **data variables** e.g. $n + 3 \geq m$

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

$$\llbracket \phi \wedge \psi \rrbracket_\varepsilon = \llbracket \phi \rrbracket_\varepsilon \cap \llbracket \psi \rrbracket_\varepsilon$$

$$\llbracket \phi \vee \psi \rrbracket_\varepsilon = \llbracket \phi \rrbracket_\varepsilon \cup \llbracket \psi \rrbracket_\varepsilon$$

$$\llbracket \forall d:D. \phi \rrbracket_\varepsilon = \bigcap_{v \in D} \llbracket \phi \rrbracket_\varepsilon[d := v]$$

$$\llbracket \exists d:D. \phi \rrbracket_\varepsilon = \bigcup_{v \in D} \llbracket \phi \rrbracket_\varepsilon[d := v]$$

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

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

- ▶ a state formula ϕ characterises a set of states in S
- ▶ state formulae contain **data variables** e.g. $n + 3 \geq m$

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

$$\llbracket \phi \wedge \psi \rrbracket_\varepsilon = \llbracket \phi \rrbracket_\varepsilon \cap \llbracket \psi \rrbracket_\varepsilon$$

$$\llbracket \phi \vee \psi \rrbracket_\varepsilon = \llbracket \phi \rrbracket_\varepsilon \cup \llbracket \psi \rrbracket_\varepsilon$$

$$\llbracket \forall d:D.\phi \rrbracket_\varepsilon = \bigcap_{v \in D} \llbracket \phi \rrbracket_\varepsilon[d := v]$$

$$\llbracket \exists d:D.\phi \rrbracket_\varepsilon = \bigcup_{v \in D} \llbracket \phi \rrbracket_\varepsilon[d := v]$$

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

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

Example

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

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

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

Linear Process Equations

Extended Hennessy-Milner Logic

First-order Modal μ -Calculus

Parameterised Boolean Equation Systems

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

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

- ▶ 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)

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

- ▶ 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 $\llbracket \phi \rrbracket_{\eta \varepsilon}$ $\subseteq S$
- ▶ The set $(2^S, \subseteq)$ is a **complete lattice**

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

- ▶ 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 $\llbracket \phi \rrbracket_{\eta\varepsilon}$ $\subseteq S$
- ▶ The set $(2^S, \subseteq)$ is a **complete lattice**

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

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

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

- ▶ 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 $\llbracket \phi \rrbracket_{\eta\varepsilon}$ $\subseteq S$
- ▶ The set $(2^S, \subseteq)$ is a **complete lattice**

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

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

Semantics of First-order Modal μ -calculus formulae:

$$\llbracket Z \rrbracket_{\eta\varepsilon} = \eta(Z)$$

$$\llbracket \nu Z. \phi \rrbracket_{\eta\varepsilon} = \nu\Phi_{\eta\varepsilon}^Z$$

$$\llbracket \mu Z. \phi \rrbracket_{\eta\varepsilon} = \mu\Phi_{\eta\varepsilon}^Z$$

Example

- ▶ Given LPE $P(n:\text{Nat}) = n > 0 \rightarrow a \cdot P(n-1)$.
- ▶ $\llbracket \mu X.[a]X \rrbracket_{\eta \varepsilon}$ yields a subset of natural numbers;
- ▶ By definition: $\llbracket \mu X.[a]X \rrbracket_{\eta \varepsilon} = \mu \Phi_{\eta \varepsilon}^X$
- ▶ The transformer $\Phi_{\eta \varepsilon}^X$ can be rewritten:

$$\begin{aligned}
 & \Phi_{\eta \varepsilon}^X(S) \\
 = & \llbracket [a]X \rrbracket_{\eta} [X := S]_{\varepsilon} \\
 = & \{n \in \mathbb{N} \mid \forall n' \in \mathbb{N}, a' \in \llbracket [a] \rrbracket_{\varepsilon}: n \xrightarrow{a'} n' \Rightarrow n' \in \llbracket X \rrbracket_{\eta} [X := S]_{\varepsilon}\} \\
 = & \{n \in \mathbb{N} \mid n > 0 \Rightarrow n-1 \in S\}
 \end{aligned}$$

- ▶ $\mu \Phi_{\eta \varepsilon}^X = Z$, such that:
 1. $Z = \{n \in \mathbb{N} \mid n > 0 \Rightarrow n-1 \in Z\}$
 2. For all $Y = \{n \in \mathbb{N} \mid n > 0 \Rightarrow n-1 \in Y\}$ we have $Z \subseteq Y$.

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
 $\nu X. \forall n: \text{Nat}. [\text{true}]X \wedge [r(n)]\mu Y. ([\neg s(n)]Y \wedge \langle \text{true} \rangle \text{true})$
4. Every stream of numbers is increasing. $\nu X. \forall n: \text{Nat}. [r(n)](X \wedge \forall m: \text{Nat}. [r(m)](m > n))$

Linear Process Equations

Extended Hennessy-Milner Logic

First-order Modal μ -Calculus

Parameterised Boolean Equation Systems

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:

$$X(e) \models \phi \text{ iff } e \in \llbracket \phi \rrbracket_{\eta\varepsilon}$$

- ▶ Decidable for **finite data types**
 - Compute LTS $\llbracket X(e) \rrbracket$
 - Evaluate ϕ on $\llbracket X(e) \rrbracket$ using standard model checking 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 (or *relation*) $X : 2^D$
- ▶ e is an expression of sort D
- ▶ Interpreting ϕ requires **two** environments ε (for data) and $\eta : \mathcal{P} \rightarrow 2^D$

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 $\eta: \mathcal{P} \rightarrow 2^D$

$$\llbracket b \rrbracket_{\eta \varepsilon} = \begin{cases} \text{true} & \text{if } \varepsilon(b) \\ \text{false} & \text{else} \end{cases}$$

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 $\eta: \mathcal{P} \rightarrow 2^D$

$$\llbracket b \rrbracket_{\eta \varepsilon} = \begin{cases} \text{true} & \text{if } \varepsilon(b) \\ \text{false} & \text{else} \end{cases} \quad \llbracket X(e) \rrbracket_{\eta \varepsilon} = \begin{cases} \text{true} & \text{if } \varepsilon(e) \in \eta(X) \\ \text{false} & \text{else} \end{cases}$$

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 $\eta: \mathcal{P} \rightarrow 2^D$

$$\llbracket b \rrbracket_{\eta \varepsilon} = \begin{cases} \text{true} & \text{if } \varepsilon(b) \\ \text{false} & \text{else} \end{cases} \quad \llbracket X(e) \rrbracket_{\eta \varepsilon} = \begin{cases} \text{true} & \text{if } \varepsilon(e) \in \eta(X) \\ \text{false} & \text{else} \end{cases}$$

$$\llbracket \phi \wedge \psi \rrbracket_{\eta \varepsilon} = \llbracket \phi \rrbracket_{\eta \varepsilon} \text{ and } \llbracket \psi \rrbracket_{\eta \varepsilon} \quad \llbracket \phi \vee \psi \rrbracket_{\eta \varepsilon} = \llbracket \phi \rrbracket_{\eta \varepsilon} \text{ or } \llbracket \psi \rrbracket_{\eta \varepsilon}$$

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 $\eta : \mathcal{P} \rightarrow 2^D$

$$\llbracket b \rrbracket_{\eta \varepsilon} = \begin{cases} \text{true} & \text{if } \varepsilon(b) \\ \text{false} & \text{else} \end{cases}$$

$$\llbracket X(e) \rrbracket_{\eta \varepsilon} = \begin{cases} \text{true} & \text{if } \varepsilon(e) \in \eta(X) \\ \text{false} & \text{else} \end{cases}$$

$$\llbracket \phi \wedge \psi \rrbracket_{\eta \varepsilon} = \llbracket \phi \rrbracket_{\eta \varepsilon} \text{ and } \llbracket \psi \rrbracket_{\eta \varepsilon}$$

$$\llbracket \phi \vee \psi \rrbracket_{\eta \varepsilon} = \llbracket \phi \rrbracket_{\eta \varepsilon} \text{ or } \llbracket \psi \rrbracket_{\eta \varepsilon}$$

$$\llbracket \forall d : D. \phi \rrbracket_{\eta \varepsilon} = \text{for all } v \in D: \\ \llbracket \phi \rrbracket_{\eta(\varepsilon[d := v])}$$

$$\llbracket \exists d : D. \phi \rrbracket_{\eta \varepsilon} = \text{for some } v \in D: \\ \llbracket \phi \rrbracket_{\eta(\varepsilon[d := v])}$$

To formulae ϕ , variables Z and d , associate $\Phi_{\eta,\varepsilon}^Z(S) := \{v \in D \mid \llbracket \phi \rrbracket_{\eta}[Z := S]\varepsilon[d := v]\}$

- ▶ $\Phi_{\eta,\varepsilon}^{Z,d}$ is **monotone**: $S \subseteq T$ implies $\Phi_{\eta,\varepsilon}^{Z,d}(S) \subseteq \Phi_{\eta,\varepsilon}^{Z,d}(T)$
- ▶ Least and greatest fixed points of $\Phi_{\eta,\varepsilon}^{Z,d}$ in $(2^S, \subseteq)$ are guaranteed to exist
- ▶ Least fixed point is denoted $\mu\Phi_{\eta,\varepsilon}^{Z,d}$; dually: $\nu\Phi_{\eta,\varepsilon}^{Z,d}$

To formulae ϕ , variables Z and d , associate $\Phi_{\eta,\varepsilon}^Z(S) := \{v \in D \mid \llbracket \phi \rrbracket_{\eta}[Z := S]\varepsilon[d := v]\}$

- ▶ $\Phi_{\eta,\varepsilon}^{Z,d}$ is **monotone**: $S \subseteq T$ implies $\Phi_{\eta,\varepsilon}^{Z,d}(S) \subseteq \Phi_{\eta,\varepsilon}^{Z,d}(T)$
- ▶ Least and greatest fixed points of $\Phi_{\eta,\varepsilon}^{Z,d}$ in $(2^S, \subseteq)$ are guaranteed to exist
- ▶ Least fixed point is denoted $\mu\Phi_{\eta,\varepsilon}^{Z,d}$; dually: $\nu\Phi_{\eta,\varepsilon}^{Z,d}$

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

- ▶ 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: $\eta : \mathcal{P} \rightarrow 2^D$

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

$$\left\{ \begin{array}{l} \llbracket \varepsilon \rrbracket \eta \varepsilon \quad \quad \quad := \eta \\ \llbracket (\mu X(d : D) = \phi) \mathcal{E} \rrbracket \eta \varepsilon \quad := \llbracket \mathcal{E} \rrbracket \eta [X := \mu \Phi_{\mathcal{E}, \eta, \varepsilon}^{X, d}] \varepsilon \\ \llbracket (\nu X(d : D) = \phi) \mathcal{E} \rrbracket \eta \varepsilon \quad := \llbracket \mathcal{E} \rrbracket \eta [X := \nu \Phi_{\mathcal{E}, \eta, \varepsilon}^{X, d}] \varepsilon \end{array} \right.$$

Note: $\Phi_{\mathcal{E}, \theta, \varepsilon}^{X, d}$ is the monotone functional associated to ϕ , X and d in the context of \mathcal{E} , defined as follows:

$$\Phi_{\mathcal{E}, \eta, \varepsilon}^Z(S) := \{v \in D \mid \llbracket \phi \rrbracket (\llbracket \mathcal{E} \rrbracket \eta [Z := S] \varepsilon) \varepsilon [d := v]\}$$

- ▶ Consider the process X , given by:

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

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

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

- ▶ for all n, b , $(n, b) \in \bar{Y}$ 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:

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

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

Next lecture:

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

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

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