# (Concrete) Coalgebraic Logics and Synthesis of Mealy Machines

Marcello Bonsangue[1,2], Jan Rutten[1,3] and Alexandra Silva[1]

[1]Centrum voor Wiskunde en Informatica
[2]LIACS - Leiden University
[3]Vrije Universiteit Amsterdam

CIC workshop, October 2007

# Motivation

## Design of Sequential systems

- (Binary) Mealy machines $\leftrightarrow$ digital circuits
- There is no formal way of specifying Mealy machines
- Typically they are "defined" in a natural language, such as English

  **Source of ambiguities**
- We need a formal way of specifying Mealy machines

# Motivation

## Design of Sequential systems

- (Binary) Mealy machines $\leftrightarrow$ digital circuits
- There is no formal way of specifying Mealy machines
- Typically they are “defined” in a natural language, such as English

  **Source of ambiguities**
- We need a formal way of specifying Mealy machines

# Motivation

## Design of Sequential systems

- (Binary) Mealy machines $\leftrightarrow$ digital circuits
- There is no formal way of specifying Mealy machines
- Typically they are "defined" in a natural language, such as English

**Source of ambiguities**

- We need a formal way of specifying Mealy machines

# Motivation

## Design of Sequential systems

- (Binary) Mealy machines $\leftrightarrow$ digital circuits
- There is no formal way of specifying Mealy machines
- Typically they are "defined" in a natural language, such as English

  **Source of ambiguities**
- We need a formal way of specifying Mealy machines

# What will we show?

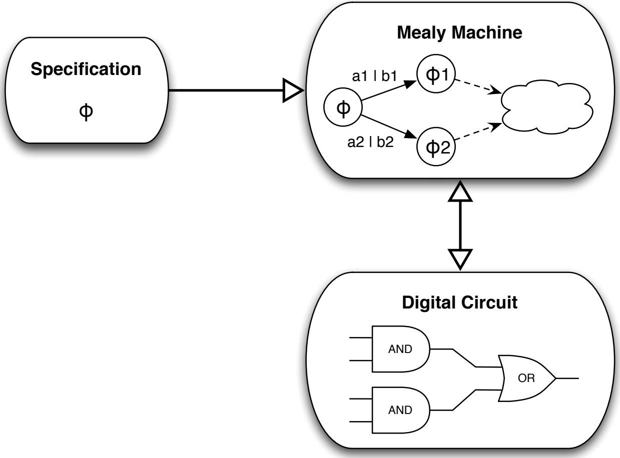
We will show ...

- ... a logic for **specifying** Mealy machines
- ... synthesis of Mealy machines from a specification formula

# What will we show?

We will show ...

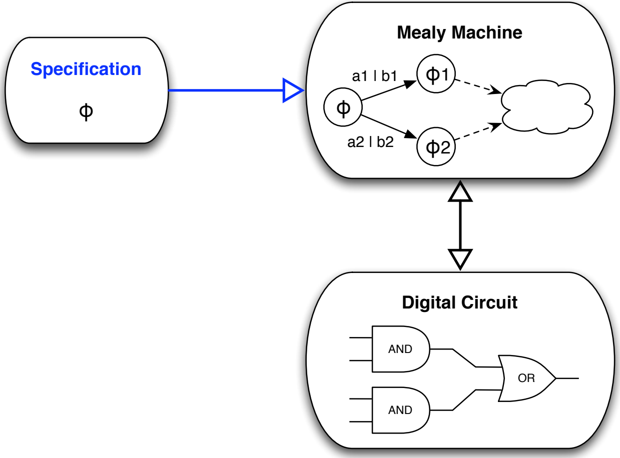
- ... a logic for **specifying** Mealy machines
- ... synthesis of Mealy machines from a specification formula



# What will we show?

We will show . . .

- . . . a logic for **specifying** Mealy machines
- . . . synthesis of Mealy machines from a specification formula



# But first...

What do we mean by **Binary Mealy Machine?**

- Mealy machines = **Deterministic Mealy machines**
- Mealy machine = set of states $S$ + transition function $f$

$$\begin{aligned} f &: S \to (B \times S)^A \\ f(s)(a) &= \; < b, s' > \end{aligned}$$

$$s \xrightarrow{a|b} s'$$

  $A$ is the input alphabet and $B$ is the output alphabet
- Binary Mealy machines : $A = 2$ and

$B = \mathcal{B} =$

# But first. . .

What do we mean by **Binary Mealy Machine?**

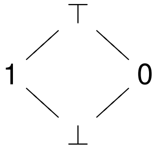
- Mealy machines = **Deterministic Mealy machines**
- Mealy machine = set of states $S$ + transition function $f$

$$\begin{array}{lcl} f & : & S \rightarrow (B \times S)^A \\ f(s)(a) & = & < b, s' > \end{array}$$

$$s \xrightarrow{a|b} s'$$

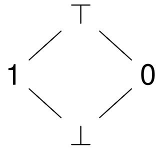
  $A$ is the input alphabet and $B$ is the output alphabet

- Binary Mealy machines : $A = 2$ and

$B = \mathcal{B} =$ 

# Information order

$\mathcal{B} =$




- $\top$ – *abstraction* (under-specification)
- $\bot$ – *inconsistency* (over-specification)
- 0 and 1 – concrete output values.

# Mealy automata are coalgebras

**Observation:**

A Mealy machine is a coalgebra of the functor

$$\begin{aligned} M &: Set \to Set \\ M(X) &= (B \times X)^A \end{aligned}$$

**(Almost) for free:**

- Notion of bisimulation : equivalence between states, minimization
- Specification logic for Mealy machines

# Mealy automata are coalgebras

## Observation:

A Mealy machine is a coalgebra of the functor

$$\begin{aligned} M &: Set \to Set \\ M(X) &= (B \times X)^A \end{aligned}$$

## (Almost) for free:

- Notion of bisimulation : equivalence between states, minimization
- Specification logic for Mealy machines

# A coalgebraic logic for Mealy machines

Based in the work by Marcello and Alexander Kurz, we derive a logic for $M(X) = (B \times X)^A$:

# A coalgebraic logic for Mealy machines

Based in the work by Marcello and Alexander Kurz, we derive a logic for $M(X) = (B \times X)^A$:

$$\phi ::= tt \quad | \quad \phi \wedge \phi \quad | \quad a(\phi) \quad | \quad a \downarrow b \quad | \quad x \quad | \quad \nu x.\psi$$

$\psi$ guarded

# A coalgebraic logic for Mealy machines

Based in the work by Marcello and Alexander Kurz, we derive a logic for $M(X) = (B \times X)^A$:

$$\phi ::= tt \quad | \quad \phi \wedge \phi \quad | \quad a(\phi) \quad | \quad a \downarrow b \quad | \quad x \quad | \quad \nu x.\psi$$

$\psi$ guarded

# A coalgebraic logic for Mealy machines

Based in the work by Marcello and Alexander Kurz, we derive a logic for $M(X) = (B \times X)^A$:

$$\phi ::= tt \quad | \quad \phi \wedge \phi \quad | \quad a(\phi) \quad | \quad a \downarrow b \quad | \quad x \quad | \quad \nu x.\psi$$

$\psi$ guarded

**Remark** Simple but expressive logic: Every finite Mealy machine corresponds to a finite formula in the logic.

# Examples

- Output 0 at each input of 1

$$1 \downarrow 0$$

- Output 0 at each input of two consecutive 1's

$$\nu x.(1(1 \downarrow 0 \wedge 1(x) \wedge 0(x)) \wedge 0(x))$$

- Output 0 at each second input of 1

$$\nu x.(0(x) \wedge 1(\nu y.0(y) \wedge 1 \downarrow 0 \wedge 1(x))$$

# Examples

- Output 0 at each input of 1

$$1 \downarrow 0$$

- Output 0 at each input of two consecutive 1's

$$\nu x.(1(1 \downarrow 0 \wedge 1(x) \wedge 0(x)) \wedge 0(x))$$

- Output 0 at each second input of 1

$$\nu x.(0(x) \wedge 1(\nu y.0(y) \wedge 1 \downarrow 0 \wedge 1(x))$$

# Examples

- Output 0 at each input of 1

$$\nu x.(1 \downarrow 0 \wedge 1(x) \wedge 0(x))$$

- Output 0 at each input of two consecutive 1's

$$\nu x.(1(1 \downarrow 0 \wedge 1(x) \wedge 0(x)) \wedge 0(x))$$

- Output 0 at each second input of 1

$$\nu x.(0(x) \wedge 1(\nu y.0(y) \wedge 1 \downarrow 0 \wedge 1(x))$$

# Examples

- Output 0 at each input of 1

$$\nu x.(1 \downarrow 0 \wedge 1(x) \wedge 0(x))$$

- Output 0 at each input of two consecutive 1's

$$\nu x.(1(1 \downarrow 0 \wedge 1(x) \wedge 0(x)) \wedge 0(x))$$

- Output 0 at each second input of 1

$$\nu x.(0(x) \wedge 1(\nu y.0(y) \wedge 1 \downarrow 0 \wedge 1(x))$$

# Examples

- Output 0 at each input of 1

$$\nu x.(1 \downarrow 0 \wedge 1(x) \wedge 0(x))$$

- Output 0 at each input of two consecutive 1's

$$\nu x.(1(1 \downarrow 0 \wedge 1(x) \wedge 0(x)) \wedge 0(x))$$

- Output 0 at each second input of 1

$$\nu x.(0(x) \wedge 1(\nu y.0(y) \wedge 1 \downarrow 0 \wedge 1(x))$$

# Examples

- Output 0 at each input of 1

$$\nu x.(1 \downarrow 0 \wedge 1(x) \wedge 0(x))$$

- Output 0 at each input of two consecutive 1's

$$\nu x.(1(1 \downarrow 0 \wedge 1(x) \wedge 0(x)) \wedge 0(x))$$

- Output 0 at each second input of 1

$$\nu x.(0(x) \wedge 1(\nu y.0(y) \wedge 1 \downarrow 0 \wedge 1(x))$$

# Examples

- Output 0 at each input of 1

$$\nu x.(1 \downarrow 0 \wedge 1(x) \wedge 0(x))$$

- Output 0 at each input of two consecutive 1's

$$\nu x.(1(1 \downarrow 0 \wedge 1(x) \wedge 0(x)) \wedge 0(x))$$

- Output 0 at each second input of 1

$$\nu x.(0(x) \wedge 1(\nu y.0(y) \wedge 1 \downarrow 0 \wedge 1(x))$$

# Satisfaction relation

For a Mealy machine $(S, f)$ and $s \in S$:

$$
\begin{array}{lll}
s \models_\eta tt & & \text{for all } s \\
s \models_\eta a(\phi) & \textit{iff} & f(s)(a)_2 \models_\eta \phi \\
s \models_\eta a \downarrow b & \textit{iff} & f(s)(a)_1 \leq_B b \\
s \models_\eta \phi_1 \wedge \phi_2 & \textit{iff} & s \models_\eta \phi_1 \textit{ and } s \models_\eta \phi_2 \\
s \models_\eta x & \textit{iff} & s \in \eta(x) \\
s \models_\eta \nu x.\phi & \textit{iff} & \exists T \subseteq S. s \in T \textit{ and } \forall t \in T. t \models_{\eta[T/x]} \phi
\end{array}
$$

# Satisfaction relation

For a Mealy machine $(S, f)$ and $s \in S$:

$$\begin{array}{lcl} & \vdots & \\ s \models_\eta a(\phi) & \textit{iff} & f(s)(a)_2 \models_\eta \phi \\ s \models_\eta a \downarrow b & \textit{iff} & f(s)(a)_1 = b \\ & \vdots & \end{array}$$

# So far...

- We can specify properties – $\phi$
- Given a mealy machine $(S, f)$ and $s \in S$ we can check if $s \models \phi$
- Next: Given $\phi$ generate $(S, f)$ such that $\exists_{s \in S}\ s \models \phi$.

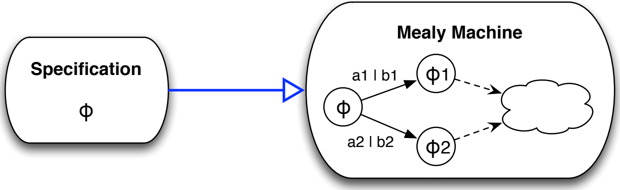
# So far...

- We can specify properties – $\phi$
- Given a mealy machine $(S, f)$ and $s \in S$ we can check if $s \models \phi$
- Next: Given $\phi$ generate $(S, f)$ such that $\exists_{s \in S}\ s \models \phi$.

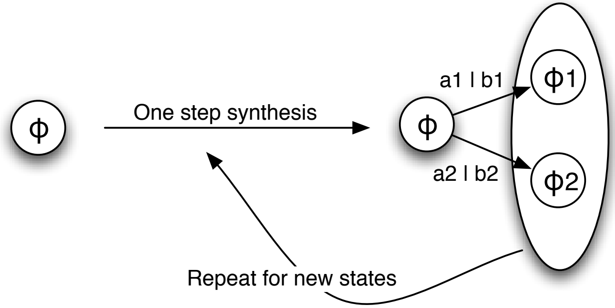
# So far. . .

- We can specify properties – $\phi$
- Given a mealy machine $(S, f)$ and $s \in S$ we can check if $s \models \phi$
- Next: Given $\phi$ generate $(S, f)$ such that $\exists_{s \in S}\ s \models \phi$.

# Synthesis



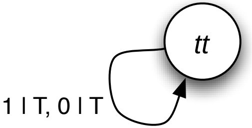
# Synthesis



ϕ
One step synthesis
ϕ
a1 l b1
a2 l b2
ϕ1
ϕ2
Repeat for new states

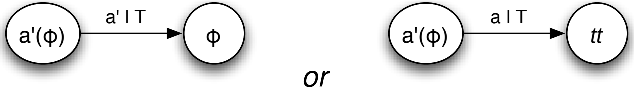
# One-step synthesis

$$\delta(tt)(a) = < \top, tt >$$



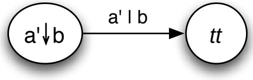
# One-step synthesis

$$\delta(a'(\phi))(a) = \begin{cases} <\top, \phi> & a = a' \\ <\top, tt> & \textit{otherwise} \end{cases}$$

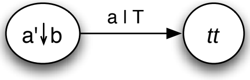


# One-step synthesis

$$\delta(a' \downarrow b)(a) = \begin{cases} < b, tt > & a = a' \\ < \top, tt > & otherwise \end{cases}$$




*or*

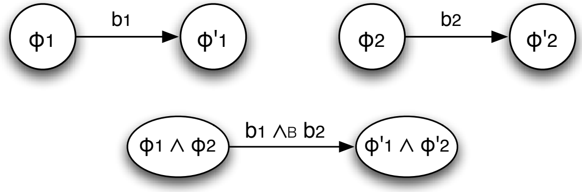


# One-step synthesis

$$\delta(\phi_1 \wedge \phi_2)(a) = \delta(\phi_1)(a) \sqcap \delta(\phi_2)(a)$$

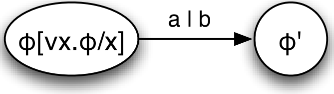
# One-step synthesis

$$\delta(\phi_1 \wedge \phi_2)(a) = \delta(\phi_1)(a) \sqcap \delta(\phi_2)(a)$$



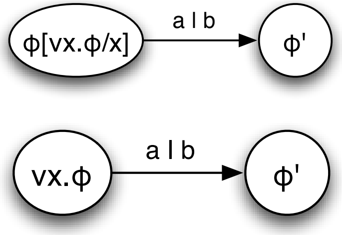
# One-step synthesis

$$\delta(\nu x.\phi)(a) = \delta(\phi[\nu x.\phi/x])(a)$$



# One-step synthesis

$$\delta(\nu x.\phi)(a) = \delta(\phi[\nu x.\phi/x])(a)$$



# Examples

- $\phi = 1 \downarrow 0 \wedge (\nu x.1(x))$

$$\begin{aligned} \delta(\phi)(0) &= <\top, tt> \\ \delta(\phi)(1) &= \delta(1 \downarrow 0)(1) \sqcap \delta(\nu x.1(x))(1) \\ &= <0, tt> \sqcap <\top_B, \nu x.1(x)> \\ &= <0, \nu x.1(x)> \end{aligned}$$

# Examples

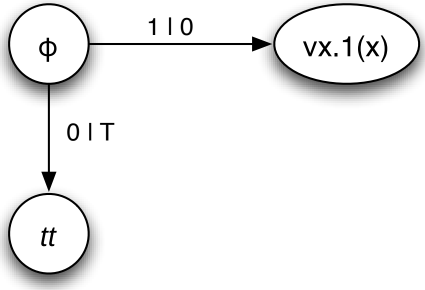
- $\phi = 1 \downarrow 0 \wedge (\nu x.1(x))$

$$\begin{array}{rcl} \delta(\phi)(0) & = & < \top, tt > \\ \delta(\phi)(1) & = & \delta(1 \downarrow 0)(1) \sqcap \delta(\nu x.1(x))(1) \\ & = & < 0, tt > \sqcap < \top_B, \nu x.1(x) > \\ & = & < 0, \nu x.1(x) > \end{array}$$

# Examples

- $\phi = 1 \downarrow 0 \wedge (\nu x.1(x))$

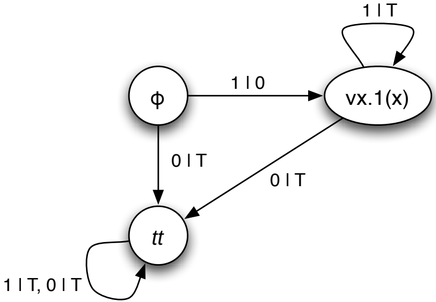
$$\begin{aligned}
\delta(\phi)(0) &= <\top, tt> \\
\delta(\phi)(1) &= \delta(1 \downarrow 0)(1) \sqcap \delta(\nu x.1(x))(1) \\
&= <0, tt> \sqcap <\top_B, \nu x.1(x)> \\
&= <0, \nu x.1(x)>
\end{aligned}$$



# Examples

- $\phi = 1 \downarrow 0 \wedge (\nu x.1(x))$

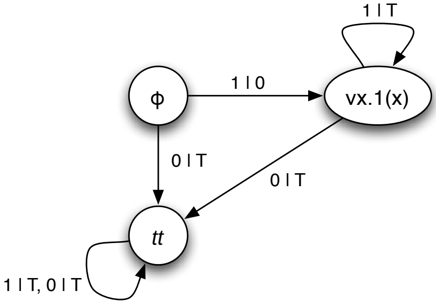
$$\begin{aligned}\delta(\phi)(0) &= < \top, tt > \\ \delta(\phi)(1) &= \delta(1 \downarrow 0)(1) \sqcap \delta(\nu x.1(x))(1) \\ &= < 0, tt > \sqcap < \top_B, \nu x.1(x) > \\ &= < 0, \nu x.1(x) >\end{aligned}$$



# Examples

- $\phi = 1 \downarrow 0 \wedge (\nu x.1(x))$

$$\begin{aligned}
\delta(\phi)(0) &= <\top, tt> \\
\delta(\phi)(1) &= \delta(1 \downarrow 0)(1) \sqcap \delta(\nu x.1(x))(1) \\
&= <0, tt> \sqcap <\top_B, \nu x.1(x)> \\
&= <0, \nu x.1(x)>
\end{aligned}$$




**Remark:** Not minimal.

# Examples

- $\phi_1 = \nu x.(1(1 \downarrow 0) \wedge 1(x))$

$$
\begin{array}{rcl}
\delta(\phi_1)(0) & = & < \top, tt > \\
\delta(\phi_1)(1) & = & \delta(1(1 \downarrow 0))(1) \sqcap \delta(\nu x.1(x))(1) \\
 & = & < \top, 1 \downarrow 0 > \sqcap < \top, \nu x.1(x) > \\
 & = & < \top, 1 \downarrow 0 \wedge (\nu x.1(x)) >
\end{array}
$$

# Examples

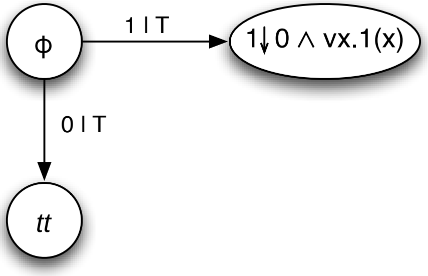
- $\phi_1 = \nu x.(1(1 \downarrow 0) \wedge 1(x))$

$$\begin{array}{rcl} \delta(\phi_1)(0) & = & < \top, tt > \\ \delta(\phi_1)(1) & = & \delta(1(1 \downarrow 0))(1) \sqcap \delta(\nu x.1(x))(1) \\ & = & < \top, 1 \downarrow 0 > \sqcap < \top, \nu x.1(x) > \\ & = & < \top, 1 \downarrow 0 \wedge (\nu x.1(x)) > \end{array}$$

# Examples

- $\phi_1 = \nu x.(1(1 \downarrow 0) \wedge 1(x))$

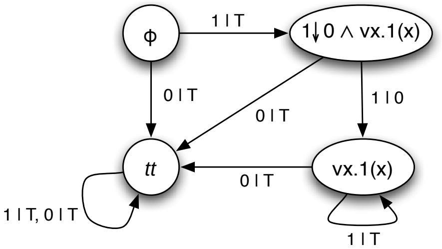
$$\begin{array}{rcl}\delta(\phi_1)(0) & = & <\top, tt> \\ \delta(\phi_1)(1) & = & \delta(1(1\downarrow 0))(1) \sqcap \delta(\nu x.1(x))(1) \\ & = & <\top, 1\downarrow 0> \sqcap <\top, \nu x.1(x)> \\ & = & <\top, 1\downarrow 0 \wedge (\nu x.1(x))>\end{array}$$



# Examples

- $\phi_1 = \nu x.(1(1 \downarrow 0) \wedge 1(x))$

$$\begin{array}{rcl} \delta(\phi_1)(0) & = & < \top, tt > \\ \delta(\phi_1)(1) & = & \delta(1(1 \downarrow 0))(1) \sqcap \delta(\nu x.1(x))(1) \\ & = & < \top, 1 \downarrow 0 > \sqcap < \top, \nu x.1(x) > \\ & = & < \top, 1 \downarrow 0 \wedge (\nu x.1(x)) > \end{array}$$



# Conclusions

- Coalgebraic approach : bisimulation and logics
- New logic for Mealy machines
- Synthesis algorithm (implemented in HASKELL)

# Future work

## We did not show...

- Semantics in terms of causal functions
- Proof system (sound and complete)

## Future work

- Generalize this approach to other type of automata: Moore automata, weighted automata, ...
- Make the logic more *user-friendly* (syntactic sugar)
- Calculate complexity of the synthesis algorithm

# Future work

## We did not show. . .

- Semantics in terms of causal functions
- Proof system (sound and complete)

## Future work

- Generalize this approach to other type of automata: Moore automata, weighted automata, . . .
- Make the logic more *user-friendly* (syntactic sugar)
- Calculate complexity of the synthesis algorithm