Coequational Logic for Finitary Functors

Daniel Schwencke

Department of Theoretical Computer Science
Institute of Technology, Braunschweig

5th April 2008
Motivation

Goal

Deduce properties of (state-based) systems from known properties of these systems.
Motivation

Goal

Deduce properties of (state-based) systems from known properties of these systems.

<table>
<thead>
<tr>
<th>concept</th>
<th>model as</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>system</td>
<td>coalgebra</td>
<td>deterministic automaton</td>
</tr>
<tr>
<td>system property</td>
<td>coequation</td>
<td>each state accepts the empty word</td>
</tr>
<tr>
<td>deduction of system properties</td>
<td>coequational logic</td>
<td>each state accepts the empty word $\Rightarrow$ each state accepts every word</td>
</tr>
</tbody>
</table>
Motivation

Goal

Deduce properties of (state-based) systems from known properties of these systems.

<table>
<thead>
<tr>
<th>concept</th>
<th>model as</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>system</td>
<td>coalgebra</td>
<td>deterministic automaton</td>
</tr>
<tr>
<td>system property</td>
<td>coequation</td>
<td>each state accepts the empty word</td>
</tr>
<tr>
<td>deduction of system properties</td>
<td>coequational logic</td>
<td>each state accepts the empty word ⇒ each state accepts every word</td>
</tr>
</tbody>
</table>
Outline

1. Coequations

2. Coequational logic
   - Polynomial functors
   - Accessible functors

3. A logic for finitary functors
   - Labelled transition systems
   - Finitary functors preserving preimages

4. Further results & summary
Outline

1 Coequations

2 Coequational logic
   • Polynomial functors
   • Accessible functors

3 A logic for finitary functors
   • Labelled transition systems
   • Finitary functors preserving preimages

4 Further results & summary
Definitions

Definition (P. Gumm)

Given a cofree coalgebra $Q$, a coequation is a coatomic set $Q \setminus q$ for some $q \in Q$.

- completely determined by $q$
- notation $\Box q$ (read: avoid $q$)
Definitions

Definition (P. Gumm)
Given a cofree coalgebra $Q$, a coequation is a coatomic set $Q \backslash q$ for some $q \in Q$.

- completely determined by $q$
- notation $\boxdot q$ (read: avoid $q$)

Definition
Given a cofree $H$-coalgebra $(Q, \alpha_Q, \gamma)$ on $C$, an $H$-coalgebra $(S, \alpha_S)$ satisfies $\boxdot q$ if $c^*(s) \neq q$ for all colourings $c : S \rightarrow C$ and for all $s \in S$.

- $c^* : S \rightarrow Q$ denotes the unique $H$-coalgebra homomorphism with $c = \gamma \circ c^*$
Coequations for deterministic automata

Example

Deterministic automata with binary input alphabet \( \{0, 1\} \)

- coalgebras of the polynomial functor \( \{tt, ff\} \times (-)\{0,1\} \)
- \( Q = \) all coloured ordered binary trees without leaves and with node labels from \( \{tt, ff\} \)
- given \( c: S \rightarrow C, \ c^*: S \rightarrow Q \) maps \( s \in S \) to the tree of its outgoing transitions
Coequations for deterministic automata

Example

Deterministic automata with binary input alphabet \( \{0, 1\} \)

- coalgebras of the polynomial functor \( \{tt, ff\} \times (-)^{\{0,1\}} \)
- \( Q = \) all coloured ordered binary trees without leaves and with node labels from \( \{tt, ff\} \)
- given \( c : S \rightarrow C, c^* : S \rightarrow Q \) maps \( s \in S \) to the tree of its outgoing transitions

\[
\begin{align*}
  &s_0 \xrightarrow{0} s_1 \\
  &s_1 \xrightarrow{0, 1} s_2 \\
  &s_2 \xrightarrow{1} \text{ (satisfies)}
\end{align*}
\]

satisfies

\[
\begin{align*}
  &\cdots \xrightarrow{\cdots} \\
  &\cdots \xrightarrow{\cdots}
\end{align*}
\]
Coequations for deterministic automata

Example

Deterministic automata with binary input alphabet \( \{0, 1\} \)

- coalgebras of the polynomial functor \( \{tt, ff\} \times (-)^{\{0,1\}} \)
- \( Q = \) all coloured ordered binary trees without leaves and with node labels from \( \{tt, ff\} \)
- given \( c : S \rightarrow C \), \( c^* : S \rightarrow Q \) maps \( s \in S \) to the tree of its outgoing transitions

\[
\begin{align*}
S_0 & \rightarrow 0 \quad S_1 \\
S_2 & \rightarrow 0, 1 \\
\end{align*}
\]

satisfies

\[
\begin{align*}
\cdots & \rightarrow \cdots \\
\end{align*}
\]
Coequations for deterministic automata

Example

Deterministic automata with binary input alphabet \( \{0, 1\} \)
- coalgebras of the polynomial functor \( \{tt, ff\} \times (-)^{0,1} \)
- \( Q = \) all coloured ordered binary trees without leaves and with node labels from \( \{tt, ff\} \)
- given \( c : S \to C, c^* : S \to Q \) maps \( s \in S \) to the tree of its outgoing transitions

satisfies

\( no \ state \ accepts \ \{w \in \{0, 1\}^* \mid w \ has \ even \ length\} \)
Semantical consequences

Definition

$q$ is a semantical consequence of $q'$ if every coalgebra satisfying $q'$ also satisfies $q$.

- notation $q' \models q$
Semantical consequences

Definition

\( \Box q \) is a semantical consequence of \( \Box q' \) if every coalgebra satisfying \( \Box q' \) also satisfies \( \Box q \).

- notation \( \Box q' \models \Box q \)

Example

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]

\[ \Box \]

\[ \vdash \]
Semantical consequences

Definition

$q$ is a semantical consequence of $q'$ if every coalgebra satisfying $q'$ also satisfies $q$.

- notation $q' \models q$

Example

![Diagram](image)

- no state accepts $\{w \in \{0,1\}^* | w \text{ has even length}\}$
- no state accepts $\{w \in \{0,1\}^* | w \text{ has odd length}\}$
Semantical consequences

**Definition**

$q$ is a semantical consequence of $q'$ if every coalgebra satisfying $q'$ also satisfies $q$.

- notation $q' \models q$

**Example**

- no state accepts $\{w \in \{0,1\}^* | w \text{ has even length}\}$
- no state accepts $\{w \in \{0,1\}^* | w \text{ has odd length}\}$

Is this true?
Outline

1. Coequations

2. Coequational logic
   - Polynomial functors
   - Accessible functors

3. A logic for finitary functors
   - Labelled transition systems
   - Finitary functors preserving preimages

4. Further results & summary
The relation $\sqsubseteq$

**Definition**

$q'$ is a *recolouring* of $q$ if there exists a homomorphism $h : Q \to Q$ with $h(q) = q'$.

- or equivalently: $q$ and $q'$ have the same shape and equivalent subtrees in $q$ are equivalent in $q'$ again
- write $q' \sqsubseteq q$ if $q'$ is a recolouring of a subtree of $q$
The relation $\sqsubseteq$

**Definition**

$q'$ is a recolouring of $q$ if there exists a homomorphism $h : Q \to Q$ with $h(q) = q'$.

- or equivalently: $q$ and $q'$ have the same shape and equivalent subtrees in $q$ are equivalent in $q'$ again
- write $q' \sqsubseteq q$ if $q'$ is a recolouring of a subtree of $q$

**Example**

Diagram showing examples of recolouring relationships.
Logic for polynomial functors

**Theorem (J. Adámek)**

\[
\{ q_i \mid i \in I \} \models q \iff \exists i \in I : q_i \sqsubseteq q
\]
Logic for polynomial functors

Theorem (J. Adámek)
\[
\{ \Diamond q_i \mid i \in I \} \models \Diamond q \iff \exists i \in I : q_i \sqsubseteq q
\]

Corollary

Deduction with the following rules is sound and complete.

Child rule:
\[
\frac{\Diamond q'}{\Diamond q}
\]
where \( q' \) is a child tree of \( q \)

Recolouring rule:
\[
\frac{\Diamond q'}{\Diamond q}
\]
where \( q' \) is recolouring of \( q \)
An answer to our question

Example

\[
\begin{array}{c}
\begin{array}{ccc}
\bullet & \rightarrow & \bullet \\
\vdots & & \vdots \\
\end{array}
\end{array}
\quad \models \quad
\begin{array}{c}
\begin{array}{ccc}
\circ & \rightarrow & \bullet \\
\vdots & & \vdots \\
\end{array}
\end{array}
\]

Yes, this is true!
An answer to our question

Example

\[
\begin{align*}
\ast & \quad \ast \\
\vdots & \quad \vdots \\
\vdots & \quad \vdots \\
\vdots & \quad \vdots \\
\vdots & \quad \vdots \\
\vdots & \quad \vdots \\
\end{align*}
\]

Yes, this is true!

Check the example automaton:

Example

\[
\begin{array}{c}
s_0 \quad 0 \quad 0,1 \quad 1 \quad 0 \quad 1 \\
\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\
s_1 \quad s_2 \quad s_0
\end{array}
\]
An answer to our question

Example

\( \neq \neq \neq \neq \)

Yes, this is true!

Check the example automaton:
Logic for accessible functors

Accessible functor $H$

- quotient of some polynomial functor $H_{\Sigma}$
  - via surjective natural transformation $\epsilon_S : H_{\Sigma}S \rightarrow HS$
- cofree coalgebra $Q$ quotient of $Q_{\Sigma}$
  - via surjective $H$-homomorphism $\gamma_{\Sigma}^* : (Q_{\Sigma}, \epsilon_{Q_{\Sigma}} \circ \alpha_{Q_{\Sigma}}) \rightarrow (Q, \alpha_Q)$
- notation $[t]$ for the equivalence class of $t \in Q_{\Sigma}$
Logic for accessible functors

Accessible functor $H$
- quotient of some polynomial functor $H\Sigma$
  - via surjective natural transformation $\epsilon_S : H\Sigma S \to HS$
- cofree coalgebra $Q$ quotient of $Q\Sigma$
  - via surjective $H$-homomorphism $\gamma^* : (Q\Sigma, \epsilon_Q \circ \alpha_{Q\Sigma}) \to (Q, \alpha_Q)$
- notation $[t]$ for the equivalence class of $t \in Q\Sigma$

Theorem (J. Adámek)
\[
\{ \Box q_i \mid i \in I \} \vdash \Box q \iff \forall t. [t] = q \exists i \in I \exists r. [r] = q_i : r \sqsubseteq t
\]
- no simple deduction rules
Outline

1. Coequations

2. Coequational logic
   - Polynomial functors
   - Accessible functors

3. A logic for finitary functors
   - Labelled transition systems
   - Finitary functors preserving preimages

4. Further results & summary
Labelled transition systems

Example

Finitely branching LTS with labels from $A$

- coalgebras of the functor $P_f(A \times -)$
- logic for accessible functors applies, but:
  - cofree coalgebra has direct description (J. Worrell)
    - $Q =$ coloured unordered strongly extensional trees with edge labels from $A$
    - notation: $t^q$ represents the equivalence class $q$
Labelled transition systems

Example

Finitely branching LTS with labels from $A$

- coalgebras of the functor $\mathcal{P}_f(A \times -)$
- logic for accessible functors applies, but:
- cofree coalgebra has direct description (J. Worrell)
  - $Q =$ coloured unordered strongly extensional trees with edge labels from $A$
  - notation: $t^q$ represents the equivalence class $q$

Dag representation:

- $s_0 \xrightarrow{a_0} s_2, s_1 \xrightarrow{a_0, a_1} s_1$
- satisfies $\Box a_1 a_0$
Labelled transition systems

Example

Finitely branching LTS with labels from $A$

- coalgebras of the functor $\mathcal{P}_f(A \times -)$
- logic for accessible functors applies, but:
  - cofree coalgebra has direct description (J. Worrell)
    - $Q =$ coloured unordered strongly extensional trees with edge labels from $A$
    - notation: $t^q$ represents the equivalence class $q$

Each state with only $a_0$-/$a_1$-transitions to halting states has a common $a_0$-/$a_1$-successor.
Logic for LTS

Theorem

\[ \{ \Box q_i | i \in I \} \models \Box q \iff \exists i \in I \exists r. [r] = q_i : r \sqsubseteq t^q \]
Logic for LTS

**Theorem**

\[\{\lozenge q_i | i \in I\} \models \lozenge q \iff \exists i \in I \exists r. [r] = q_i : r \sqsubseteq t^q\]

**Corollary**

*Deduction with the following rules is sound and complete.*

*Child rule:*

\[\lozenge q' \quad \text{where } t^{q'} \text{ is a child tree of } t^q\]

*Recolouring rule:*

\[\lozenge q' \quad \text{where } [r] = q' \text{ for a recolouring } r \text{ of } t^q\]
Finitary functors preserving preimages

Finitary functor $H$

- quotient of some polynomial functor $H_\Sigma$ with finitary signature $\Sigma$
  - $Q$ is quotient of $Q_\Sigma$ via $H$-homomorphism $\gamma^*_\Sigma$
- example: the LTS-functor $P_f(A \times -)$
Finitary functors preserving preimages

Finitary functor $H$
- quotient of some polynomial functor $H_\Sigma$ with finitary signature $\Sigma$
  - $Q$ is quotient of $Q_\Sigma$ via $H$-homomorphism $\gamma_\Sigma^*$
- example: the LTS-functor $\mathcal{P}_f(A \times -)$

Preimages
- pullbacks along monos
- if preserved by $H$, $Q$ is a “well-behaved” quotient of $Q_\Sigma$
- preserved by $\mathcal{P}_f(A \times -)$
Finitary functors preserving preimages

Finitary functor $H$
- quotient of some polynomial functor $H_\Sigma$ with finitary signature $\Sigma$
  - $Q$ is quotient of $Q_\Sigma$ via $H$-homomorphism $\gamma^*_\Sigma$
- example: the LTS-functor $\mathcal{P}_f(A \times -)$

Preimages
- pullbacks along monos
- if preserved by $H$, $Q$ is a “well-behaved” quotient of $Q_\Sigma$
- preserved by $\mathcal{P}_f(A \times -)$

Logic as for LTS
- find a set of representative trees $t^q$ (closed under taking subtrees)
- prove compatibility of equivalence classes with subtrees and recolourings
Equivalence classes and subtrees

Proposition

Let $H$ preserve preimages, then $[t] = [t'] \implies \forall s \exists s' : [s] = [s']$

(where $s, s'$ are subtrees of $t$ and $t'$ respectively).
Equivalence classes and subtrees

Proposition

Let $H$ preserve preimages, then $[t] = [t'] \implies \forall s \exists s' : [s] = [s']$ (where $s, s'$ are subtrees of $t$ and $t'$ respectively).

Proof.

It suffices to show this for child trees $s, s'$ of $t, t'$.
Equivalence classes and subtrees

Proposition

Let $H$ preserve preimages, then $[t] = [t'] \implies \forall s \exists s' : [s] = [s']$ (where $s, s'$ are subtrees of $t$ and $t'$ respectively).

Proof.

It suffices to show this for child trees $s, s'$ of $t, t'$.

\[ Q_\Sigma \xrightarrow{\gamma^*_\Sigma} Q \]
\[ \alpha_{Q_\Sigma} \downarrow \quad \quad \quad \quad \downarrow \alpha_Q \]
\[ H_\Sigma Q_\Sigma \quad \quad \quad \quad \quad \quad \quad HQ \]
\[ \epsilon_{Q_\Sigma} \downarrow \quad \quad \quad \quad \downarrow \quad \quad \downarrow \]
\[ HQ_\Sigma \xrightarrow{H\gamma^*_\Sigma} HQ \]

- $\alpha_{Q_\Sigma}$ and $\alpha_Q$ are isomorphisms
Equivalence classes and subtrees

**Proposition**

Let $H$ preserve preimages, then $[t] = [t'] \implies \forall s \exists s' : [s] = [s']$ (where $s, s'$ are subtrees of $t$ and $t'$ respectively).

**Proof.**

It suffices to show this for child trees $s, s'$ of $t, t'$.

- $\alpha_{Q\Sigma}$ and $\alpha_Q$ are isomorphisms
- $\alpha_{Q\Sigma}$ presents $t \in Q\Sigma$ by its root labels and child trees $c/\sigma(t_1, \ldots, t_n)$
Equivalence classes and subtrees

**Proposition**

Let $H$ preserve preimages, then

$$[t] = [t'] \implies \forall s \exists s' : [s] = [s']$$

(where $s, s'$ are subtrees of $t$ and $t'$ respectively).

**Proof.**

It suffices to show this for child trees $s, s'$ of $t, t'$.

- $\alpha_{Q\Sigma}$ and $\alpha_Q$ are isomorphisms
- $\alpha_{Q\Sigma}$ presents $t \in Q\Sigma$ by its root labels and child trees $c/\sigma(t_1, \ldots, t_n)$
- $\ker(H\gamma^*_\Sigma \circ \epsilon_{Q\Sigma} \circ \alpha_{Q\Sigma}) = (\ker(\epsilon_{Q\Sigma} \circ \alpha_{Q\Sigma}) \cup M)^*$
  where $M := \{(t, t') | \alpha_{Q\Sigma}(t) = c/\sigma(t_1, \ldots, t_n), \alpha_{Q\Sigma}(t') = c/\sigma(t'_1, \ldots, t'_n), [t_i] = [t'_i]\}$
Equivalence classes and subtrees

**Proposition**

Let $H$ preserve preimages, then $[t] = [t'] \implies \forall s \exists s' : [s] = [s']$ (where $s, s'$ are subtrees of $t$ and $t'$ respectively).

**Proof.**

It suffices to show this for child trees $s, s'$ of $t, t'$.

- $\alpha_{Q\Sigma}$ and $\alpha_Q$ are isomorphisms
- $\alpha_{Q\Sigma}$ presents $t \in Q\Sigma$ by its root labels and child trees $c/\sigma(t_1, \ldots, t_n)$
- $\text{ker}(H\gamma_{\Sigma}^* \circ \epsilon_{Q\Sigma} \circ \alpha_{Q\Sigma}) = (\text{ker}(\epsilon_{Q\Sigma} \circ \alpha_{Q\Sigma}) \cup M)^*$ where $M := \{(t, t') \mid \alpha_{Q\Sigma}(t) = c/\sigma(t_1, \ldots, t_n), \alpha_{Q\Sigma}(t') = c/\sigma(t'_1, \ldots, t'_n), [t_i] = [t'_i]\}$
- pairs in $\text{ker}(\epsilon_{Q\Sigma} \circ \alpha_{Q\Sigma})$ and $M$ have same child tree equivalence classes, preserved by $\cup$ and $*$
Equivalence classes and recolourings

We try: "Let $H$ preserve preimages, then $[t] = [t'] \implies \forall r \exists r' : [r] = [r']$ (where $r, r'$ are recolourings of $t$ and $t'$ respectively)."
Equivalence classes and recolourings

We try: “Let $H$ preserve preimages, then $[t] = [t'] \implies \forall r \exists r' : [r] = [r']$ (where $r, r'$ are recolourings of $t$ and $t'$ respectively).” But:

Example

Let $H$ be the quotient of $H_\Sigma = (-)^2 + 1$ modulo the order of the binary operation, then $H$ preserves preimages.
Representative trees

Given $H$ as a quotient of $H_\Sigma$ ($\Sigma$ finitary), it holds that

- $T_\Sigma$ consists of finitely branching $\Sigma$-trees
- a linear order on $\Sigma$ induces a linear order on $T_\Sigma$
  - lexicographic order: compare trees by breadth-first search
- similarly, an order on $\Sigma \times C$ induces one on $Q_\Sigma$
Representative trees

Given $H$ as a quotient of $H_{\Sigma}$ ($\Sigma$ finitary), it holds that

- $T_{\Sigma}$ consists of finitely branching $\Sigma$-trees
- a linear order on $\Sigma$ induces a linear order on $T_{\Sigma}$
  - lexicographic order: compare trees by breadth-first search
- similarly, an order on $\Sigma \times C$ induces one on $Q_{\Sigma}$
- for each $q \in Q$ the first tree of that class in the order can be used as the representative tree $t^q$
- subtrees of these representative trees are representative trees again
  - suppose the opposite and derive a contradiction
Representative trees

Given $H$ as a quotient of $H_\Sigma$ ($\Sigma$ finitary), it holds that
- $T_\Sigma$ consists of finitely branching $\Sigma$-trees
- a linear order on $\Sigma$ induces a linear order on $T_\Sigma$
  - lexicographic order: compare trees by breadth-first search
- similarly, an order on $\Sigma \times C$ induces one on $Q_\Sigma$
- for each $q \in Q$ the first tree of that class in the order can be used as the representative tree $t^q$
- subtrees of these representative trees are representative trees again
  - suppose the opposite and derive a contradiction

Example

Let again $H_\Sigma = (-)^2 + 1$, i.e. $\Sigma = \{c, \sigma\}$ and $C = \{b, r\}$. Order $\Sigma \times C$ by $(c, r) < (c, b) < (\sigma, r) < (\sigma, b)$. Obtain

```
  <  <  <  <  <  ...
```

```plaintext
```
```
Equivalence classes and recolourings, continued

**Proposition**

Let $H$ preserve preimages, then $[t^q] = [t'] \implies \forall r \exists r': [r] = [r']$ (where $r, r'$ are recolourings of $t^q$ and $t'$ respectively).
Equivalence classes and recolourings, continued

Proposition

Let $H$ preserve preimages, then $[t^q] = [t'] \implies \forall r \exists r' : [r] = [r']$
(\textit{where $r$, $r'$ are recolourings of $t^q$ and $t'$ respectively}).

Proof.

- $r$ induces a map $f' : \text{subtrees}(t^q) \to C$
- extend $f'$ to $f : Q_\Sigma \to C$ with respect to equivalence classes
- obtain unique homomorphism $f^* : Q_\Sigma \to Q_\Sigma$ with $f = \gamma_\Sigma \circ f^*$
- obviously $r = f^*(t^q)$, choose $r' : = f^*(t')$ and prove $[r] = [r']$
Logic for preimage preserving finitary functors

Theorem

\[ \left\{ \square q_i \mid i \in I \right\} \models \square q \iff \exists i \in I \exists r. [r] = q_i : r \sqsubseteq t^q \]
Logic for preimage preserving finitary functors

Theorem

\[ \{ \mathbf{q}_i | i \in I \} \models \mathbf{q} \iff \exists i \in I \exists r. [r] = q_i : r \sqsubseteq t^q \]

Proof.

Show equivalence of the above rhs and

\[ \forall t'. [t'] = q \exists i \in I \exists r'. [r'] = q_i : r' \sqsubseteq t' \] (logic for accessible functors).
Logic for preimage preserving finitary functors

Theorem
\[ \{ \Box q_i \mid i \in I \} \models \Box q \iff \exists i \in I \exists r. [r] = q_i : r \sqsubseteq t^q \]

Proof.

Show equivalence of the above rhs and
\[ \forall t'. [t'] = q \exists i \in I \exists r'. [r'] = q_i : r' \sqsubseteq t' \] (logic for accessible functors).

\( \Rightarrow \) Suppose there is \( r \) with \([r] = q_i\) for some \( i \in I \) and \( r \sqsubseteq t^q \)

\( \Rightarrow \) i.e. \( r \) is recolouring of some subtree \( s \) of \( t^q \)
### Logic for preimage preserving finitary functors

**Theorem**

\[ \{ \Diamond q_i \mid i \in I \} \models \Diamond q \iff \exists i \in I \exists r. [r] = q_i : r \sqsubseteq t^q \]

**Proof.**

Show equivalence of the above rhs and

\[ \forall t'. [t'] = q \exists i \in I \exists r'. [r'] = q_i : r' \sqsubseteq t' \] (logic for accessible functors).

⇒ Suppose there is \( r \) with \( [r] = q_i \) for some \( i \in I \) and \( r \sqsubseteq t^q \)

- i.e. \( r \) is recolouring of some subtree \( s \) of \( t^q \)
- for every tree \( t' \) with \( [t'] = [t^q] \) we can find a subtree \( s' \) of \( t' \) with \( [s'] = [s] \)
- \( s \) is a representative tree because \( t^q \) is
Logic for preimage preserving finitary functors

Theorem

\[ \{ \boxdot q_i \mid i \in I \} \models \boxdot q \iff \exists i \in I \exists r. [r] = q_i : r \sqsubseteq t^q \]

Proof.

Show equivalence of the above rhs and

\[ \forall t'. [t'] = q \exists i \in I \exists r'. [r'] = q_i : r' \sqsubseteq t' \] (logic for accessible functors).

⇒ Suppose there is \( r \) with \([r] = q_i \) for some \( i \in I \) and \( r \sqsubseteq t^q \)

- i.e. \( r \) is recolouring of some subtree \( s \) of \( t^q \)
- for every tree \( t' \) with \([t'] = [t^q]\) we can find a subtree \( s' \) of \( t' \) with \([s'] = [s]\)
- \( s \) is a representantive tree because \( t^q \) is
- for \( s' \) with \([s'] = [s]\) we can find a recolouring \( r' \) of \( s' \) with \([r'] = [r]\)
- together: \( r' \sqsubseteq t' \) and \([r'] = [r] = q_i\)
Logic for preimage preserving finitary functors

Theorem
\[
\{ \Box q_i | i \in I \} \models \Box q \iff \exists i \in I \exists r.[r] = q_i : r \subseteq t^q
\]

Proof.
Show equivalence of the above rhs and
\[
\forall t'.[t'] = q \exists i \in I \exists r'.[r'] = q_i : r' \subseteq t' \quad \text{(logic for accessible functors)}.
\]

⇒ Suppose there is \( r \) with \([r] = q_i\) for some \( i \in I \) and \( r \subseteq t^q \)
  - i. e. \( r \) is recolouring of some subtree \( s \) of \( t^q \)
  - for every tree \( t' \) with \([t'] = [t^q]\) we can find a subtree \( s' \) of \( t' \) with \([s'] = [s]\)
  - \( s \) is a representative tree because \( t^q \) is
  - for \( s' \) with \([s'] = [s]\) we can find a recolouring \( r' \) of \( s' \) with \([r'] = [r]\)
  - together: \( r' \subseteq t' \) and \([r'] = [r] = q_i\)

⇐ Suppose the formula for accessible functors to be true
  - apply it to the representative tree \( t^q \) \([t^q] = q\)
Outline

1. Coequations

2. Coequational logic
   - Polynomial functors
   - Accessible functors

3. A logic for finitary functors
   - Labelled transition systems
   - Finitary functors preserving preimages

4. Further results & summary
Finitary functors not preserving preimages

**Theorem**

A finitary functor $H$ preserves preimages if and only if for every linear order of $\Sigma \times C$ we have

$$\left\{ \Box q_i \mid i \in I \right\} \models \Box q \iff \exists i \in I \exists r. [r] = q_i : r \sqsubseteq t^q.$$
Finitary functors not preserving preimages

**Theorem**

A finitary functor $H$ preserves preimages if and only if for every linear order of $\Sigma \times C$ we have

$$\{\square q_i | i \in I\} \models \square q \iff \exists i \in I \exists r. [r] = q_i : r \sqsubseteq t^q.$$  

Nevertheless, there are finitary functors not preserving preimages to which the simple logic still applies in case of a special order.

**Example**

Let $H$ be the quotient of $H_\Sigma = (-)^2 + 1$ ($\Sigma = \{c, \sigma\}$) modulo the equation $\sigma(x, x) = c$. Then $H$ does not preserve preimages, but given an order of $\Sigma \times C$ where $(c, k) < (\sigma, k)$ for all $k \in C$, the simple logic applies.
Summary

- coequations as system properties
- coequational logic for
  - polynomial functors
  - accessible functors
- finitary functors preserving preimages
  - example: LTS
  - how to find a simpler logic
Summary / Literature

- coequations as system properties
- coequational logic for
  - polynomial functors
  - accessible functors
- finitary functors preserving preimages
  - example: LTS
  - how to find a simpler logic

► J. Adámek.

_A Logic of Coequations._


► D. Schwencke.

_Coequational Logic for Finitary Functors._

To appear in ENTCS (CMCS’08 volume).
Thank you...  

...for your attention!

schwencke@iti.cs.tu-bs.de
Definition

A **covariety** is a class of coalgebras that is closed under sums, subcoalgebras and homomorphic images.
Properties presentable by coequations

**Definition**

A *covariety* is a class of coalgebras that is closed under sums, subcoalgebras and homomorphic images.

**Theorem (Co-Birkhoff-Theorem, J. Rutten)**

*Covarieties are exactly the classes of coalgebras defined by coequations (where the latter means that the class contains all coalgebras satisfying the coequations).*
Properties presentable by coequations

Definition
A **covariety** is a class of coalgebras that is closed under sums, subcoalgebras and homomorphic images.

Theorem (Co-Birkhoff-Theorem, J. Rutten)
 Covarieties are exactly the classes of coalgebras defined by coequations (where the latter means that the class contains all coalgebras satisfying the coequations).

Corollary
*The system properties coequations stand for are exactly the ones preserved by sums, subcoalgebras and homomorphic images of coalgebras.*
A presentation of finitary functors

**Definition**

Let $X$ be a set of variables. For two operations $\sigma, \tau \in \Sigma$ the expression

$$\sigma(x_1, \ldots, x_n) = \tau(y_1, \ldots, y_m) \quad x_i, y_j \in X$$

is called an $\epsilon$-equation if $\epsilon_X : H_\Sigma X \to HX$ merges the two sides.
A presentation of finitary functors

Definition
Let $X$ be a set of variables. For two operations $\sigma, \tau \in \Sigma$ the expression

$$\sigma(x_1, \ldots, x_n) = \tau(y_1, \ldots, y_m) \quad x_i, y_j \in X$$

is called an $\epsilon$-equation if $\epsilon_X : H_{\Sigma}X \rightarrow HX$ merges the two sides.

Application of an $\epsilon$-equation (from left to right) to $t \in T_{\Sigma}$ at node $v$

- possible if
  - $v$ is labelled by $\sigma$ and
  - two of its children at positions $k, l$ root the same tree whenever $x_k = x_l$
- it means
  - relabelling $v$ by $\tau$ and
  - rearranging the child trees of $v$ according to the rhs of the equation (for new variables arbitrary trees are chosen)
Application of $\epsilon$-equations

Example

Let $\Sigma = \{c, \sigma, \tau\}$ ($c$ constant, $\sigma$ binary and $\tau$ ternary). We apply the $\epsilon$-equation $\sigma(x, y) = \tau(y, x, x)$.

\[
\begin{align*}
\tau &
\Rightarrow \\
c &
\tau &
\Rightarrow
\end{align*}
\]

\[
\begin{align*}
\tau
\Rightarrow \\
c &
\tau &
\tau
\Rightarrow \\
c &
\tau &
\tau
\Rightarrow
\end{align*}
\]
A presentation of finitary functors (continued)

**Definition**

We say \( t' \) can be obtained from \( t \) by (possibly infinitely many) **applications of **\( \epsilon \)-equations if for every \( k \in \mathbb{N} \) the tree \( \partial_k t' \) can be obtained from \( \partial_k t \) by (finitely many) applications of \( \epsilon \)-equations.

- \( \partial_k t \) denotes the cutting of \( t \) at level \( k \)
A presentation of finitary functors (continued)

Definition

We say $t'$ can be obtained from $t$ by (possibly infinitely many) applications of $\epsilon$-equations if for every $k \in \mathbb{N}$ the tree $\partial_k t'$ can be obtained from $\partial_k t$ by (finitely many) applications of $\epsilon$-equations.

- $\partial_k t$ denotes the cutting of $t$ at level $k$

Theorem (J. Adámek, S. Milius)

*The terminal $H$-coalgebra $T$ is the quotient of $T_{\Sigma}$ modulo the congruence of application of $\epsilon$-equations.*

- similar for cofree coalgebras
ε-equations and preimages

Definition

An ε-equation is called regular if the sets of variables on both sides are the same.
$\epsilon$-equations and preimages

**Definition**

An $\epsilon$-equation is called **regular** if the sets of variables on both sides are the same.

**Theorem (J. Adámek, D. Lücke, S. Milius)**

A finitary functor $H$ preserves preimages if and only if there is a presentation of $H$ as a quotient of some $H_\Sigma$ via regular $\epsilon$-equations.