Type-Based Structural Analysis for Modular Systems of Equations

Linköping University, 19 June 2014

Henrik Nilsson
Joint work with John Capper

School of Computer Science
University of Nottingham
Background (1)

- Large-scale systems of equations, such as models of physical systems, need to be expressed modularly.
Background (1)

- Large-scale systems of equations, such as models of physical systems, need to be expressed modularly.
- Languages like Modelica cater for that need.
Background (1)

- Large-scale systems of equations, such as models of physical systems, need to be expressed modularly.
- Languages like Modelica caters for that need.
- Ideally we would like to know, as early as possible:
Large-scale systems of equations, such as models of physical systems, need to be expressed modularly. Languages like Modelica cater for that need. Ideally we would like to know, as early as possible:
- Is a system of equations solvable?
Large-scale systems of equations, such as models of physical systems, need to be expressed modularly.

Languages like Modelica caters for that need.

Ideally we would like to know, as early as possible:
- Is a system of equations solvable?
- Does an individual equation system fragment “make sense”; i.e., could it be part of a solvable system?
Background (2)

Studying/enforcing aspects of the relation between equations and variables occurring in them can help identifying problems, e.g.:

- Bunus and Fritzon’s work on debugging Modelica models [1]
- Broman, Nyström, and Fritzon’s work on enforcing balance between variables and equations through the Modelica type system [2]
- Modelica’s notion of balanced models
Background (2)

Studying/enforcing aspects of the relation between equations and variables occurring in them can help identifying problems, e.g.:

- Bunus and Fritzon’s work on debugging Modelica models [1]
- Broman, Nyström, and Fritzon’s work on enforcing balance between variables and equations through the Modelica type system [2]
- Modelica’s notion of balanced models

However, (static) guarantees only possible under very limited circumstances.
Quick Review of the Problem (1)

- Consider an equation over $\mathbb{R}$:

\[ x + y + z = 0 \] (1)
- Consider an equation over $\mathbb{R}$:

$$x + y + z = 0 \quad (1)$$

- Does not have a (unique) solution.
Quick Review of the Problem (1)

- Consider an equation over $\mathbb{R}$:

\[
x + y + z = 0
\]  \hspace{1cm} (1)

- Does not have a (unique) solution.
- Could be part of a system that does have a (unique) solution.
Quick Review of the Problem (1)

- Consider an equation over $\mathbb{R}$:

$$x + y + z = 0 \quad (1)$$

- Does not have a (unique) solution.
- Could be part of a system that does have a (unique) solution.

- The same holds for:

$$x - y + z = 1 \quad z = 2 \quad (2)$$
Quick Review of the Problem (2)

- Composing (1) and (2):

\[
\begin{align*}
    x + y + z &= 0 \\
    x - y + z &= 1 \\
    z &= 2
\end{align*}
\]

Does have a solution.
Quick Review of the Problem (2)

- Composing (1) and (2):

\[
\begin{align*}
  x + y + z &= 0 \\
  x - y + z &= 1 \\
  z &= 2
\end{align*}
\]

Does have a solution.

- However, the following fragment is over-constrained:

\[
\begin{align*}
  x &= 1 \\
  x &= 2
\end{align*}
\]
Approaches (1)

- Broman et al. annotate the *type* of components with difference between number of equations and number of variables.
Approaches (1)

- Broman et al. annotate the *type* of components with difference between number of equations and number of variables.
- Type-based approach, so modular and early detection.
Approaches (1)

- Broman et al. annotate the type of components with difference between number of equations and number of variables.
- Type-based approach, so modular and early detection.
- Can confirm (3) OK while (4) has a problem.
• Broman et al. annotate the *type* of components with difference between number of equations and number of variables.

• Type-based approach, so modular and early detection.

• Can confirm (3) OK while (4) has a problem.

• However, would accept:

\[
\begin{align*}
x &= 1 \\
x &= 2 \\
x + y + z &= 0
\end{align*}
\]
Approaches (2)

- Modelica has adopted a simplified notion of balance:
Approaches (2)

- Modelica has adopted a simplified notion of balance:
  - Models are required to be locally balanced
Modelica has adopted a simplified notion of balance:
- Models are required to be locally balanced
- *Partial* models are allowed to be unbalanced
Approaches (2)

- Modelica has adopted a simplified notion of balance:
  - Models are required to be locally balanced
  - *Partial* models are allowed to be unbalanced
- Early detection, but type-level information limited to partial model or not.
Approaches (3)

- Bunus and Fritzon consider whether systems are *structurally non-singular* or not: can equations and variables be put in a one-to-one relationship?
Approaches (3)

- Bunus and Fritzon considers whether systems are *structurally non-singular* or not: can equations and variables be put in a one-to-one relationship?
- Finds more problems than simple balance checking.
Approaches (3)

- Bunus and Fritzon considers whether systems are *structurally non-singular* or not: can equations and variables be put in a one-to-one relationship?
- Finds more problems than simple balance checking.
- Would reject (5)
Approaches (3)

- Bunus and Fritzon considers whether systems are *structurally non-singular* or not: can equations and variables be put in a one-to-one relationship?
- Finds more problems than simple balance checking.
- Would reject (5)
- Based on analysing a complete system, so late detection. (Intended to be a debugging tool.)
This Talk

- Type-based approaches inherently modular.
This Talk

- Type-based approaches inherently modular.
- Sensible to look for type-based solutions if we wish to support
  - First-class equation system fragments
  - Equation systems that vary over time (structural dynamism).
This Talk

- Type-based approaches inherently modular.
- Sensible to look for type-based solutions if we wish to support
  - First-class equation system fragments
  - Equation systems that vary over time (structural dynamism).
- How far beyond basic balance checking can we go with a type-based approach?
This Talk

- Type-based approaches inherently modular.
- Sensible to look for type-based solutions if we wish to support
  - First-class equation system fragments
  - Equation systems that vary over time (structural dynamism).
- How far beyond basic balance checking can we go with a type-based approach?
- We’ll investigate two approaches:
  - Incidence Type
  - Structural Well-Formedness
We need some notation. Observations:
Functional Hybrid Modelling

We need some notation. Observations:

- a system of equations specifies a \textit{relation} among a set of variables
We need some notation. Observations:

- a system of equations specifies a *relation* among a set of variables
- an equation system fragment needs an *interface* to distinguish between local variables and variables used for composition with other fragments.
We need some notation. Observations:

- a system of equations specifies a \textit{relation} among a set of variables

- an equation system fragment needs an \textit{interface} to distinguish between local variables and variables used for composition with other fragments.

Our work has been carried out in the setting of Functional Hybrid Modelling (FHM), so let’s opt for that.

(But remember: the ideas are generally applicable.)
The FHM Setting (1)

- FHM: functional approach to modelling and simulation of (physical) systems that can be described by an *evolving* set of differential equations.
The FHM Setting (2)

- Undirected equations: *non-causal modelling*. (Differential Algebraic Equations, DAE; like Modelica)
The FHM Setting (2)

- Undirected equations: *non-causal modelling*. (Differential Algebraic Equations, DAE; like Modelica)

- Two-level design:
  - *equation level* for modelling components
  - *functional level* for spatial and temporal composition of components
The FHM Setting (2)

- Undirected equations: *non-causal modelling*. (Differential Algebraic Equations, DAE; like Modelica)

- Two-level design:
  - *equation level* for modelling components
  - *functional level* for spatial and temporal composition of components

- Equations system fragments are first-class entities at the functional level; viewed as relations on signals, or *signal relations*.
The FHM Setting (3)

- Spatial composition: signal relation application; enables modular, hierarchical, system description.
The FHM Setting (3)

- Spatial composition: signal relation *application*; enables modular, hierarchical, system description.
- Temporal composition: *switching* from one structural configuration into another.
\textit{resistor} :: \text{Resistance} \rightarrow \text{SR(Pin, Pin)}

\textit{resistor} \ r = \text{sigrel} \ (p, n) \ \text{where}

\begin{align*}
\text{local} \ u \\
\text{twoPin} \  \diamond \ (p, n, u) \\
r \cdot p.i &= u
\end{align*}
The FHM Setting (4)

A parametrisable model represented by a function mapping parameters to a model. Note: first class models!

\[
\text{resistor} :: \text{Resistance} \rightarrow \text{SR}((\text{Pin}, \text{Pin})
\]

\[
\text{resistor } r = \text{sigrel} (p, n) \text{ where local } u
\]

\[
\text{twoPin } \diamond (p, n, u)
\]

\[
r \cdot p.i = u
\]
The FHM Setting (4)

**Parametrised** model represented by function mapping parameters to a model. Note: first class models!

\[
\text{resistor :: Resistance} \rightarrow \text{SR(Pin, Pin)}
\]

\[
\text{resistor } r = \text{sigrel} \ (p, n) \quad \text{where}
\]

\[
\text{local } u \quad \text{twoPin} \otimes (p, n, u)
\]

\[
r \cdot p.i = u
\]

Encapsulated equation system fragment.
The FHM Setting (4)

Parametrised model represented by function mapping parameters to a model. Note: first class models!

\[
\text{resistor} :: \text{Resistance} \rightarrow \text{SR (Pin, Pin)}
\]

\[
\text{resistor } r = \text{sigrel } (p, n) \text{ where } \text{local } u
\]

\[
\text{twoPin } \odot (p, n, u) \quad r \cdot p.i = u
\]

Encapsulated equation system fragment.

Signal relation application allows modular construction of models from component models.
The FHM Setting (5)

Composition by signal relation application ($f_1$, $f_2$ are known function symbols):

\[
\begin{align*}
\text{foo} &:: SR (\text{Real, Real, Real}) \\
\text{foo} &= \text{sigrel} (x_1, x_2, x_3) \text{ where} \\
&\begin{align*}
&f_1 x_1 \ x_2 \ x_3 = 0 \\
&f_2 x_2 \ x_3 = 0
\end{align*}
\\
\text{foo} \odot (u, v, w) \\
\text{foo} \odot (w, u + x, v + y)
\end{align*}
\]

yields

\[
\begin{align*}
&f_1 u \ v \ w = 0 \\
&f_2 v \ w = 0 \\
&f_1 w (u + x) (v + y) = 0 \\
&f_2 (u + x) (v + y) = 0
\end{align*}
\]
The FHM Setting (7)

Evolving system of equations by switching blocks of equations in and out:

\[
\text{initially } [; \text{when } \text{condition}_1] \Rightarrow \text{equations}_1 \\
\text{when } \text{condition}_2 \Rightarrow \text{equations}_2 \\
\ldots \\
\text{when } \text{condition}_n \Rightarrow \text{equations}_n
\]
The FHM Setting (8)

FHM is thus characterised by *iterative staging*:

1. Compute model ("flat" system of equations)
2. Simulate (solve)
3. Repeat
A system of equations is *structurally singular* iff not possible to put variables and equations in a one-to-one correspondence such that each variable occurs in the equation it is related to.
Structural Non-singularity (1)

A system of equations is **structurally singular** iff not possible to put variables and equations in a one-to-one correspondence such that each variable occurs in the equation it is related to.

- Structural non-singularity is (generally) neither a necessary nor sufficient condition for solvability.
A system of equations is **structurally singular** iff not possible to put variables and equations in a one-to-one correspondence such that each variable occurs in the equation it is related to.

- Structural non-singularity is (generally) neither a necessary nor sufficient condition for solvability.
- However typical solvers are predicated on the system being structurally non-singular.
A system of equations is *structurally singular* iff not possible to put variables and equations in a one-to-one correspondence such that each variable occurs in the equation it is related to.

- Structural non-singularity is (generally) neither a necessary nor sufficient condition for solvability.
- However typical solvers are predicated on the system being structurally non-singular.
- Insisting on structural non-singularity thus makes sense and is not overly onerous.
Structural Non-singularity (2)

Structural singularities can be discovered by studying the *incidence matrix*:

\[
\begin{align*}
    f_1(x, y, z) &= 0 \\
    f_2(z) &= 0 \\
    f_3(z) &= 0
\end{align*}
\]

Equations | Incidence Matrix
---|---
\[
\begin{pmatrix}
1 & 1 & 1 \\
0 & 0 & 1 \\
0 & 0 & 1
\end{pmatrix}
\]
A Possible Refinement (3)

So maybe we can index signal relations by incidence matrices?

\[
\text{foo} :: \text{SR} (\text{Real, Real, Real}) \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}
\]

\[
\text{foo} = \text{sigrel} (x_1, x_2, x_3) \text{ where}
\]

\[
f_1 \ x_1 \ x_2 \ x_3 = 0
\]

\[
f_2 \ x_2 \ x_3 = 0
\]
The Incidence Type represents information about which variables occur in which equations.
Incidence Type

- The *Incidence Type* represents information about which variables occur in which equations.
- Denoted by an incidence matrix.
Incidence Type

- The **Incidence Type** represents information about which variables occur in which equations.
- Denoted by an incidence matrix.
- Two interrelated instances:
  - Incidence type of a *system of equations*
  - Incidence type of a *signal relation*
The Incidence Type represents information about which variables occur in which equations.

Denoted by an incidence matrix.

Two interrelated instances:
- Incidence type of a system of equations
- Incidence type of a signal relation

The incidence type of signal relation is obtained by abstraction over the incidence type of a system of equations as some variables are local.
Composition of Incidence Types (1)

Recall

\[ \text{foo} :: SR(\text{Real}, \text{Real}, \text{Real}) \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix} \]

Consider

\[ \text{foo} \odot (u, v, w) \]
\[ \text{foo} \odot (w, u + x, v + y) \]

in a context with five variables \( u, v, w, x, y \).
Composition of Incidence Types (2)

The incidence type for the equations obtained by instantiating $foo$ is simply obtained by Boolean matrix multiplication. For $foo \diamond (u, v, w)$:

\[
\begin{pmatrix}
1 & 1 & 1 \\
0 & 1 & 1 \\
0 & 1 & 1
\end{pmatrix}
\begin{pmatrix}
u & v & w & x & y \\
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0
\end{pmatrix}
= 
\begin{pmatrix}
u & v & w & x & y \\
1 & 1 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 & 0
\end{pmatrix}
\]
Composition of Incidence Types (3)

For $foo \diamond (w, u + x, v + y)$:

\[
\begin{pmatrix}
1 & 1 & 1 & 0 & 1 \\
0 & 1 & 1 & 1 & 1 \\
1 & 1 & 0 & 1 & 1
\end{pmatrix}
\begin{pmatrix}
u \\
v \\
w \\
x \\
y
\end{pmatrix}
\begin{pmatrix}
0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
u \\
v \\
w \\
x \\
y
\end{pmatrix}
= 
\begin{pmatrix}
u \\
v \\
w \\
x \\
y
\end{pmatrix}
\begin{pmatrix}
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 0 & 1 & 1
\end{pmatrix}
Composition of Incidence Types (4)

Complete incidence matrix and corresponding equations:

\[
\begin{pmatrix}
  u & v & w & x & y \\
  1 & 1 & 1 & 0 & 0 \\
  0 & 1 & 1 & 0 & 0 \\
  1 & 1 & 1 & 1 & 1 \\
  1 & 1 & 0 & 1 & 1 \\
\end{pmatrix}
\]

\[
f_1 \, u \, v \, w = 0
\]

\[
f_2 \, v \, w = 0
\]

\[
f_1 \, w \, (u + x) \, (v + y) = 0
\]

\[
f_2 \, (u + x) \, (v + y) = 0
\]
Abstraction over Incidence Types (1)

Now consider encapsulating the equations:

\[ \bar{r} = \text{sigrel} (u, y) \text{ where} \]
\[ \text{local } v, w, x \]
\[ \text{foo } \diamond (u, v, w) \]
\[ \text{foo } \diamond (w, u + x, v + y) \]

The equations of \( \bar{r} \) needs to be partitioned into:

- **Local Equations**: equations used to (notionally) solve for the local variables
- **Interface Equations**: equations contributed to the outside
Abstraction over Incidence Types (2)

How to partition?
Abstraction over Incidence Types (2)

How to partition?

- *A priori local equations*: equations over local variables only.
Abstraction over Incidence Types (2)

How to partition?

- *A priori local equations*: equations over local variables only.
- *A priori interface equations*: equations over interface variables only.
Abstraction over Incidence Types (2)

How to partition?

- **A priori local equations**: equations over local variables only.
- **A priori interface equations**: equations over interface variables only.
- **Mixed equations**: equations over local and interface variables.
How to partition?

- **A priori local equations**: equations over local variables only.
- **A priori interface equations**: equations over interface variables only.
- **Mixed equations**: equations over local and interface variables.

Note: too few or too many local equations, or too many interface equations, means *locally underdetermined* or *overdetermined* systems of equations.
Abstraction over Incidence Types (3)

In our case:
Abstraction over Incidence Types (3)

In our case:

- We have 1 a priori local equation, 3 mixed equations
Abstraction over Incidence Types (3)

In our case:

• We have 1 a priori local equation, 3 mixed equations

• We need to choose 3 local equations (as 3 local variables) and 1 interface equation
Abstraction over Incidence Types (3)

In our case:

- We have 1 a priori local equation, 3 mixed equations
- We need to choose 3 local equations (as 3 local variables) and 1 interface equation
- Consequently, 3 possibilities, yielding the following possible incidence types for $\bar{a}$:

\[
\begin{pmatrix}
u & y \\
1 & 0
\end{pmatrix}
\begin{pmatrix}
u & y \\
1 & 1
\end{pmatrix}
\begin{pmatrix}
u & y \\
1 & 1
\end{pmatrix}
\]
Abstraction over Incidence Types (4)

The two last possibilities are equivalent. But still leaves two distinct possibilities. How to choose?
Abstraction over Incidence Types (4)

The two last possibilities are equivalent. But still leaves two distinct possibilities. How to choose?

- Assume the choice is free
The two last possibilities are equivalent. But still leaves two distinct possibilities. How to choose?

- Assume the choice is free
- Note that a type with more variable occurrences is “better” as it gives more freedom when pairing equations and variables. Thus discard choices that are subsumed by better choices.
Abstraction over Incidence Types (4)

The two last possibilities are equivalent. But still leaves two distinct possibilities. How to choose?

- Assume the choice is free
- Note that a type with more variable occurrences is “better” as it gives more freedom when pairing equations and variables. Thus discard choices that are subsumed by better choices.
- As a last resort, approximate.
Abstraction over Incidence Types (4)

The two last possibilities are equivalent. But still leaves two distinct possibilities. How to choose?

- Assume the choice is free
- Note that a type with more variable occurrences is “better” as it gives more freedom when pairing equations and variables. Thus discard choices that are subsumed by better choices.
- As a last resort, approximate.

Details in [1].
Pros and Cons

Works for analysis in a first-order setting. However:

- Incidence types not intuitive.
- The matrix notation is cumbersome.
- Type annotations would often be needed.
- Type-checking seems expensive.
Pros and Cons

Works for analysis in a first-order setting. However:

- Incidence types not intuitive.
- The matrix notation is cumbersome.
- Type annotations would often be needed.
- Type-checking seems expensive.

Is there a middle ground between incidence types and basic equation-variable balance?
Structural Well-Formedness (1)

Structural well-formedness [2] is a notion that is:
Structural well-formedness [2] is a notion that is:

- a better approximation of structural non-singularity than equation-variable balance;
Structural well-formedness [2] is a notion that is:

- a better approximation of structural non-singularity than equation-variable balance;
- less precise, but much more practical (for FHM at least) than incidence types.
Structural Well-Formedness (1)

Structural well-formedness [2] is a notion that is:

- a better approximation of structural non-singularity than equation-variable balance;
- less precise, but much more practical (for FHM at least) than incidence types.

Signal relation types are indexed by \textit{balance}: henceforth meaning the \textit{number of contributed equations}. Further, \textit{constraints} on balance variables:

\[ 2 \leq m \leq 4, \; 3 \leq n \leq 5 \implies SR (\ldots) m \rightarrow SR (\ldots) n \]
The approach distinguishes two kinds of variables:

- **interface variables** \((i_Z)\)
- **local variables** \((l_Z)\)

and three kinds of equations:

- **interface equations** \((i_Q)\)
- **mixed equations** \((m_Q)\)
- **local equations** \((l_Q)\)

Total number of equations: \(a_Q = i_Q + m_Q + l_Q\).
Structural Well-Formedness (2)

The approach distinguishes two kinds of variables:

- **interface variables** ($i_Z$)
- **local variables** ($l_Z$)

and three kinds of equations:

- **interface equations** ($i_Q$)
- **mixed equations** ($m_Q$)
- **local equations** ($l_Q$)

Total number of equations: $a_Q = i_Q + m_Q + l_Q$.
Note: abstraction of earlier notion of local equation etc.
A signal relation is **structurally well-formed** (SWF) iff:

1. $l_Q + m_Q \geq l_Z$
2. $l_Q \leq l_Z$
3. $i_Q \leq i_Z$
4. $a_Q - l_Z \leq i_Z$
5. $i_Q \geq 0$, $m_Q \geq 0$, $l_Q \geq 0$

The balance (contribution) of a SWF relation is $n = a_Q - l_Z$. 
Recall that FHM allows for an evolving system of equations:

\[
\text{initially } [; \text{when } \text{condition}_1 ] \Rightarrow \text{equations}_1 \\
\text{when } \text{condition}_2 \Rightarrow \text{equations}_2 \\
\ldots \\
\text{when } \text{condition}_n \Rightarrow \text{equations}_n
\]

What about structural well-formedness?
Structural Dynamism and SWF (1)

Exactly one switch-branch active at any point. How should the number of equations in each branch be related?
Structural Dynamism and SWF (1)

Exactly one switch-branch active at any point. How should the number of equations in each branch be related?

- **Strong Approach**: exactly the same number of interface, mixed, and local equations in each branch.
Structural Dynamism and SWF (1)

Exactly one switch-branch active at any point. How should the number of equations in each branch be related?

- **Strong Approach**: exactly the same number of interface, mixed, and local equations in each branch.

*Very restrictive.*
Exactly one switch-branch active at any point. How should the number of equations in each branch be related?

- **Strong Approach**: exactly the same number of interface, mixed, and local equations in each branch. *Very restrictive.*
- **Weak Approach**: same number of equations in each branch.
Structural Dynamism and SWF (1)

Exactly one switch-branch active at any point. How should the number of equations in each branch be related?

- **Strong Approach**: exactly the same number of interface, mixed, and local equations in each branch. 
  *Very restrictive.*

- **Weak Approach**: same number of equations in each branch. 
  *Loses too much information.*
Structural Dynamism and SWF (1)

Exactly one switch-branch active at any point. How should the number of equations in each branch be related?

- **Strong Approach**: exactly the same number of interface, mixed, and local equations in each branch. 
  *Very restrictive.*

- **Weak Approach**: same number of equations in each branch. 
  *Loses too much information.*

- **Fair Approach**: branches are reconcilable.
A switch-block is *reconcilable*, contributing *i* interface equations, *m* mixed equations, *l* local equations, iff *i*, *m*, *l* satisfying the following constraints for each branch *k* can be found:

6. \( i \geq i_k \geq 0 \)
7. \( l \geq l_k \geq 0 \)
8. \( m \leq m_k - (i - i_k) - (l - l_k) \)
9. \( i + m + l = i_k + m_k + l_k \)

Note: Interestingly, *m* may be negative!
Consider:

\[ \text{foo} = \text{sigrel} (x, y) \text{ where} \]
\[ \text{local } z \]
\[ f (x, y, z) = 0 \]
\[ g (x) = 0 \]

\[ \text{fie} = \text{sigrel} (u) \text{ where} \]
\[ \text{local } v \]
\[ \text{foo} \diamond (u, v) \]
Preservation? (1)

Consider:

\[ \text{foo} = \text{sigrel} (x, y) \text{ where} \]
\[ \text{local } z \]
\[ f (x, y, z) = 0 \]
\[ g (x) = 0 \]

\[ \text{fie} = \text{sigrel} (u) \text{ where} \]
\[ \text{local } v \]
\[ \text{foo } \diamond (u, v) \]

Both \text{foo} and \text{fie} are structurally well-formed (why?) with balance 1 and 0, respectively.
Preservation? (2)

But if we carry out some “flattening”:

\[ f\text{ie} = \text{sigrel} \ (u) \ \text{where} \]
\[ \text{local} \ z, v \]
\[ f \ (u, v, z) = 0 \]
\[ g \ (u) = 0 \]
Preservation? (2)

But if we carry out some “flattening”:

\[ f(v) = \text{sigrel}(u) \text{ where} \]

local \( z, v \)

\[ f(u, v, z) = 0 \]

\[ g(u) = 0 \]

The equation that initially was classified as mixed turned out to be an interface equation; only one equation to solve for two local variables \( z \) and \( v \).
Preservation? (2)

But if we carry out some “flattening”:

\[
\text{flie} = \text{sigrel}(u) \text{ where}\n\]
\[
\begin{align*}
\text{local } & z, v \\
 f(u, v, z) &= 0 \\
 g(u) &= 0
\end{align*}
\]

The equation that initially was classified as mixed turned out to be an interface equation; only one equation to solve for two local variables \( z \) and \( v \).

Reduction turned a structurally well-formed relation into one that is ill-formed.
No Preservation = Big Problem?

- The FHM type system is a *refinement* of an underlying standard type for which progress and preservation does hold.
The FHM type system is a refinement of an underlying standard type for which progress and preservation does hold.

As to the refinement: we cannot say a (fragment of an) equation system is “definitely not flawed” . . .
No Preservation = Big Problem?

- The FHM type system is a *refinement* of an underlying standard type for which progress and preservation does hold.
- As to the refinement: we cannot say a (fragment of an) equation system is “definitely not flawed” . . . but we can say it is “not definitely flawed”.
No Preservation = Big Problem?

- The FHM type system is a *refinement* of an underlying standard type for which progress and preservation does hold.

- As to the refinement: we cannot say a (fragment of an) equation system is “definitely not flawed” . . . but we can say it is “not definitely flawed”.

- Good enough for finding problems.
No Preservation = Big Problem?

- The FHM type system is a refinement of an underlying standard type for which progress and preservation does hold.
- As to the refinement: we cannot say a (fragment of an) equation system is “definitely not flawed” . . . but we can say it is “not definitely flawed”.
- Good enough for finding problems.
- Similarities to e.g. Hybrid Types [Knowles and Flanagan].
References (1)


References (2)
