Type-Based Structural Analysis for Modular Systems of Equations

Henrik Nilsson

School of Computer Science

University of Nottingham

The Problem (1)

- A core aspect of equation-based modelling: modular description of models through composition of equation system fragments.
- Naturally, we are interested in ensurig composition makes sense, catching any mistakes as early as possible.
- Central question: do the equations have a solution?
- Cannot be answered comprehensively before we have a complete model.
  Not very modular!

The Problem (2)

- However, it might be possible to check violations of certain ***necessary*** conditions for solvability in a ***modular way***!
- One necssary condition for solvability is that a system must not be ***structurally singular***.
- The paper investigates the extent to which the structural singularity of a system of equations can be checked modularly.

Modular Systems of Equations (1)

We need a notation for modular systems of equations. Note:

- a system of equations specifies a ***relation*** among a set of variables
- specifically, our interest is relations on time-varying values or ***signals***
- an equation system fragment needs an ***interface*** to distinguish between local variables and variables used for composition with other fragments.

Modular Systems of Equations (2)

These ideas can be captured through a notion of ***typed signal relations***:

$$
\begin{aligned}
&foo :: SR\,(Real, Real, Real)\\
&foo = \mathbf{sigrel}\;(x_1, x_2, x_3)\;\mathbf{where}\\
&\quad f_1\;x_1\;x_2\;x_3 = 0\\
&\quad f_2\;x_2\;x_3 \qquad = 0
\end{aligned}
$$

Modular Systems of Equations (3)

Composition can by expressed through ***signal relation application***:

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

yields

$$\begin{aligned} f_1\ u\ v\ w &= 0 \\ f_2\ v\ w &= 0 \\ f_1\ w\ (u + x)\ (v + y) &= 0 \\ f_2\ (u + x)\ (v + y) &= 0 \end{aligned}$$

Modular Systems of Equations (4)

Treating signal relations as ***first class entities*** in a functional setting is a simple way to add essential functionality, such as a way to parameterize the relations:

$$
\begin{aligned}
&\mathit{foo2} :: \mathit{Int} \rightarrow \mathit{Real} \rightarrow \mathit{SR}\,(\mathit{Real}, \mathit{Real}, \mathit{Real})\\
&\mathit{foo2}\ n\ k = \textbf{sigrel}\ (x_1, x_2, x_3)\ \textbf{where}\\
&\qquad f_1\ n\ x_1\ x_2\ x_3 = 0\\
&\qquad f_2\ x_2\ x_3 \qquad\quad\; = k
\end{aligned}
$$

Example: Resistor Model

$$twoPin :: SR\,(Pin, Pin, Voltage)$$

$$twoPin = \mathbf{sigrel}\,(p, n, u)\ \mathbf{where}$$

$$\quad u = p.v - n.v$$

$$\quad p.i + n.i = 0$$

$$resistor :: Resistance \rightarrow SR\,(Pin, Pin)$$

$$resistor\ r = \mathbf{sigrel}\,(p, n)\ \mathbf{where}$$

$$\quad twoPin \diamond (p, n, u)$$

$$\quad r * p.i = u$$

Tracking Variable/Equation Balance?

Equal number of equations and variables is a necessary condition for solvability. For a modular analysis, one might keep track of the ***balance*** in the signal relation ***type***:

$$SR\,(\ldots)\ \boldsymbol{n}$$

Tracking Variable/Equation Balance?

Equal number of equations and variables is a necessary condition for solvability. For a modular analysis, one might keep track of the ***balance*** in the signal relation ***type***:

$$SR\ (\ldots)\ \boldsymbol{n}$$

But very weak assurances:

$$\begin{aligned} f(x, y, z) &= 0 \\ g(z) &= 0 \\ h(z) &= 0 \end{aligned}$$

A Possible Refinement (1)

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

A Possible Refinement (2)

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

Equations	Incidence Matrix
$$\begin{aligned} f_1(x, y, z) &= 0 \\ f_2(z) &= 0 \\ f_3(z) &= 0 \end{aligned}$$	$$\begin{array}{ccc} x & y & z \end{array}$$ $$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

A Possible Refinement (3)

So maybe we can index signal relations with incidence matrices?

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

$$\begin{aligned} & foo = \mathbf{sigrel}\ (x_1, x_2, x_3)\ \mathbf{where} \\ & \quad f_1\ x_1\ x_2\ x_3 = 0 \\ & \quad f_2\ x_2\ x_3 = 0 \end{aligned}$$

Structural Type (1)

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

Structural Type (2)

- The structural type of a system of equations is obtained by ***composition*** of the structural types of constituent signal relations. ***Straightforward***.
- The structural type of a signal relation is obtained by ***abstraction*** over the structural type of a system of equations. ***Less straightforward***.

Composition of Structural Types (1)

Recall

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

Consider

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

in a context with five variables u, v, w, x, y.

Composition of Structural Types (2)

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

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix} \begin{array}{c} \begin{array}{ccccc} u & v & w & x & y \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array} =$$

$$\begin{array}{c} \begin{array}{ccccc} u & v & w & x & y \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Composition of Structural Types (3)

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

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix} \begin{array}{c} \begin{array}{ccccc} u & v & w & x & y \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{pmatrix} \end{array} =$$

$$\begin{array}{c} \begin{array}{ccccc} u & v & w & x & y \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{pmatrix} \end{array}$$

Composition of Structural Types (4)

Complete incidence matrix and corresponding equations:

$$\begin{array}{c} \begin{matrix} u & v & w & x & y \end{matrix} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{pmatrix} \end{array}$$

$$\begin{aligned} f_1\ u\ v\ w &= 0 \\ f_2\ v\ w &= 0 \\ f_1\ w\ (u+x)\ (v+y) &= 0 \\ f_2\ (u+x)\ (v+y) &= 0 \end{aligned}$$

Abstraction over Structural Types (1)

Now consider encapsulating the equations:

$$
\begin{aligned}
&bar = \textbf{sigrel}\ (u, y)\ \textbf{where} \\
&\quad foo \diamond (u, v, w) \\
&\quad foo \diamond (w, u + x, v + y)
\end{aligned}
$$

The equations of the body of bar needs to be partitioned into

- ***Local Equations***: equations used to solve for the local variables
- ***Interface Equations***: equations contributed to the outside

Abstraction over Structural Types (2)

How to partition?

Abstraction over Structural Types (2)

How to partition?

- ***A priori local equations***: equations over local variables only.

Abstraction over Structural 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 Structural 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.

Abstraction over Structural 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.

Note: too few or too many local equations gives an opportunity to catch locally underdetermined or overdeteremined systems of equations.

Abstraction over Structural Types (3)

In our case:

Abstraction over Structural Types (3)

In our case:

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

Abstraction over Structural Types (3)

In our case:

- We have 1 a priori local equation, 3 mixed equations
- We need to choose 3 local equations and 1 interface equation

Abstraction over Structural Types (3)

In our case:

- We have 1 a priori local equation, 3 mixed equations
- We need to choose 3 local equations and 1 interface equation
- Consequently, ***3*** possibilities, yielding the following possible structural types for bar:

$$\begin{matrix} u & y \\ \end{matrix} \quad \begin{matrix} u & y \\ \end{matrix} \quad \begin{matrix} u & y \\ \end{matrix}$$
$$\begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \end{pmatrix}$$

Abstraction over Structural Types (4)

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

Abstraction over Structural Types (4)

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

- Assume the choice is free

Abstraction over Structural 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.

Abstraction over Structural 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 Structural Types (4)

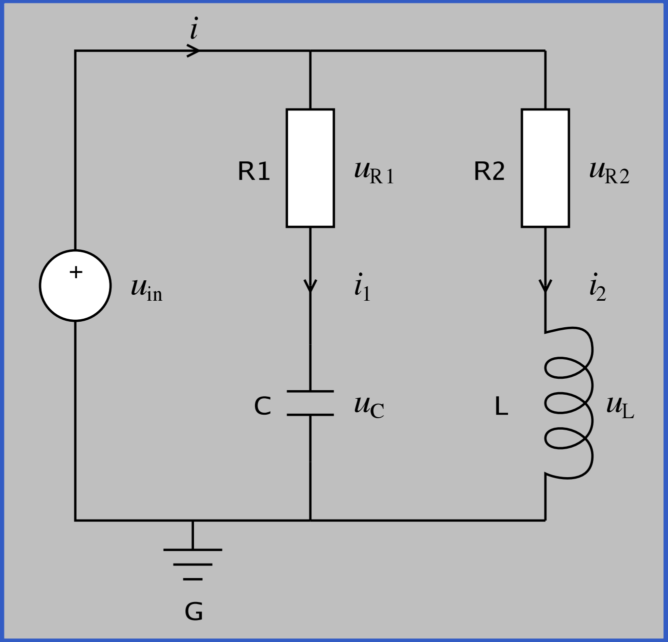
The two last possibilities are equivalent. But still leaves two distinct possibilities. How to choose?

- 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 the paper.

Also in the Paper

- A more realistic modelling example:



- Structural types for components of this model
- Example of error in this model that is caught by the proposed method, but would not have been found by just counting equations and variables.