An Index Theory for Hybrid DAE Systems

Albert Benveniste    Timothy Bourke    Benoît Caillaud    Marc Pouzet

INRIA (Rennes) and ENS (Paris)

10 December 2014
Compositionality and reuse: from ODE to DAE

from Simulink (ODE):
HS in state space form
\[
\begin{aligned}
\dot{x} &= f(x, u) \\
y &= g(x, u)
\end{aligned}
\]
the state space form depends on the context
reuse is difficult

to Modelica (DAE):
HS as physical balance equations
\[
\begin{aligned}
0 &= f(\dot{x}, x, u) \\
0 &= g(x, u)
\end{aligned}
\]
Ohm & Kirchhoff laws, bond graphs, multi-body mechanical systems
reuse is much easier
Compositionality and reuse: from ODE to DAE

- Modeling tools supporting DAE
  - Most modeling tools provide only a library of predefined models ready for assembly (Mathworks/Simscape, LMS/AmeSim)
  - Modelica comes with a full programming language that is a public standard https://www.modelica.org/

- Strange outcomes for the simulations were known to occur with Simulink/Stateflow (ask Tim Bourke and Marc Pouzet for nice ones);
- Exploration of Modelica is only starting…
Compositionality and reuse: from ODE to DAE

- Modeling tools supporting DAE
  - Most modeling tools provide only a library of predefined models ready for assembly (Mathworks/Simscape, LMS/AmeSim)
  - Modelica comes with a full programming language that is a public standard https://www.modelica.org/

- Strange outcomes for the simulations were known to occur with Simulink/Stateflow (ask Tim Bourke and Marc Pouzet for nice ones);
- Exploration of Modelica is only starting...
We do not claim that these tools are bad, as there are real difficulties:

- from ODE solvers to DAE solvers
- events of mode changes for Hybrid DAE systems
- and the (nonsmooth) physics itself:

semiconductor, circuit breaker, ...

multibody mechanics

sliding modes
A key notion in DAE Systems: the **index**

- **Differential** Algebraic Equation systems (continuous time) may involve more constraints than specified:

  \[
  \begin{cases}
    \dot{x} = f(x, u) \\
    0 = g(x)
  \end{cases}
  \]

  differentiating

  \[
  \begin{cases}
    \dot{x} = f(x, u) \\
    0 = g(x) \\
    0 = g'(x)\dot{x}
  \end{cases}
  \]

  substituting

  \[
  \begin{cases}
    \dot{x} = f(x, u) \\
    0 = g(x) \\
    0 = g'(x)f(x, u)
  \end{cases}
  \]

- **What is the effect on execution schemes?** (∼ constructive semantics)
A key notion in DAE Systems: the index

- difference Algebraic Equation systems (discrete time) may involve more constraints than specified:

\[
\begin{align*}
\begin{cases}
    x^\bullet &= f(x, u) \\
    0 &= g(x)
\end{cases}
\end{align*}
\]

shifting \[ \Rightarrow \]

\[
\begin{align*}
\begin{cases}
    x^\bullet &= f(x, u) \\
    0 &= g(x) \\
    0 &= g(x^\bullet)
\end{cases}
\end{align*}
\]

substituting \[ \Rightarrow \]

\[
\begin{align*}
\begin{cases}
    x^\bullet &= f(x, u) \\
    0 &= g(x) \\
    0 &= g(f(x, u))
\end{cases}
\end{align*}
\]

\[ (1) \quad (2) \quad (3) \]
A key notion in DAE Systems: the index

- **Difference** Algebraic Equation systems (discrete time) may involve more constraints than specified:

\[
\begin{align*}
\begin{cases}
x^\bullet &= f(x, u) \\
0 &= g(x)
\end{cases}
\end{align*}
\]

\[
\begin{align*}
\begin{cases}
x^\bullet &= f(x, u) \\
0 &= g(x) \\
0 &= g(x^\bullet)
\end{cases}
\end{align*}
\]\n
 Execution scheme (∼ constructive semantics):

1. Given \( x \) such that \( g(x) = 0 \)
2. Use (3) to evaluate \( u \) (constraint solver needed)
3. Use (1) to evaluate \( x^\bullet \), which satisfies \( g(x^\bullet) = 0 \), and repeat

Adding (3) essential in deriving the constructive semantics \( x \rightarrow x^\bullet \)
A key notion in DAE Systems: the index

- **difference** Algebraic Equation systems (discrete time) may involve more constraints than specified:

\[
\begin{align*}
\begin{cases}
x^\bullet &= f(x, u) \\
0 &= g(x)
\end{cases}
\end{align*}
\]

\[
\begin{align*}
\begin{cases}
x^\bullet &= f(x, u) \\
0 &= g(x) \\
0 &= g(x^\bullet)
\end{cases}
\end{align*}
\]

\[
\begin{align*}
\begin{cases}
x^\bullet &= f(x, u) \\
0 &= g(x) \\
0 &= g(x^\bullet) \\
0 &= g(x^{\bullet^2})
\end{cases}
\end{align*}
\]

- Shifting\(^2\) is useless since the second shifting introduces
  1. eqn (4) but also
  2. the fresh variable \(x^{\bullet^2}\)

Thus, adding (4) does not help getting the value of \(x^\bullet\)
A key notion in DAE Systems: the index

- difference Algebraic Equation systems (discrete time) may involve more constraints than specified:

\[
\begin{align*}
\begin{cases}
    x^{\bullet^2} & = f(x, u) \\
    0 & = g(x)
\end{cases}
\end{align*}
\]  

\[
\begin{align*}
\begin{cases}
    x^{\bullet^2} & = f(x, u) \\
    0 & = g(x)
\end{cases}
\end{align*}
\]

- Shifting$^2$ is useful since the second shifting introduces
  1. eqn (4) and
  2. the variable $x^{\bullet^2}$, which is not fresh but already there
A key notion in DAE Systems: the **index**

- Differential Algebraic Equations systems (continuous time) may involve more constraints than specified:

\[
\begin{align*}
\dot{x} &= f(x, u) \\
0 &= g(x)
\end{align*}
\]

\[
\begin{align*}
\dot{x} &= f(x, u) \\
0 &= g(x) \\
0 &= g'_x(x) \dot{x}
\end{align*}
\]

\[
\begin{align*}
\dot{x} &= f(x, u) \\
0 &= g(x) \\
0 &= g'_x(x) f(x, u)
\end{align*}
\]

\[
\begin{align*}
\dot{x} &= f(x, u) \\
0 &= G(x, u)
\end{align*}
\]

- ODE if we have a constraint solver getting \( x \mapsto u \) using (2,3)
A key notion in DAE Systems: the index

Define the index as being the minimal number of differentiations and shiftings needed until no further differentiation and no further shifting can reveal additional latent constraints.

The notion of differentiation index emerged in the late 1980’s in the applied mathematics community; other notions of index were proposed, see [Campbell & Gear 1995].
Research Agenda

- So-called index reduction is a front processing of models making DAE/dAE looking like known objects;
- The execution requires a constraint solver but no further deep forward exploration of runs is needed (warning: the index may be infinite)

- Unfortunately, no notion of index was mathematically defined for Hybrid DAE systems
  - it is informally claimed that “Hybrid DAE systems possess a mode-dependent index”
  - unfortunately this has no math basis and leads to problems at compilation: what to do at mode changes?
- This talk is about index for Hybrid DAE systems and it turns out that index for dAE systems is also interesting in itself
So-called index reduction is a front processing of models making DAE/dAE looking like known objects;

The execution requires a constraint solver but no further deep forward exploration of runs is needed (warning: the index may be infinite)

Unfortunately, no notion of index was mathematically defined for Hybrid DAE systems

it is informally claimed that “Hybrid DAE systems possess a mode-dependent index”

unfortunately this has no math basis and leads to problems at compilation: what to do at mode changes?

This talk is about index for Hybrid DAE systems and it turns out that index for dAE systems is also interesting in itself
Exact and Structural DAE index (linear algebra reasoning)

Structural dAE index and causality analysis

Through NonStandard semantics DAE become dAE

The index of a Hybrid DAE System is the dAE index of its NS-semantics

Conclusions
Exact Differentiation/Difference Index

\[ F(x, \dot{x}) = 0 \]
\[ F(x, x^\bullet) = 0 \]

\[
\begin{bmatrix}
  F(x, \dot{x}) \\
  \frac{d}{dt} F(x, \dot{x}) \\
  \vdots \\
  \frac{d^k}{dt^k} F(x, \dot{x})
\end{bmatrix} = \text{def}

\begin{bmatrix}
  F^0(x, \dot{x}, \omega) \\
  F^1(x, \dot{x}, \omega) \\
  \vdots \\
  F^k(x, \dot{x}, \omega)
\end{bmatrix}, \quad \left\{ \begin{array}{l}
  v = \text{def} \dot{x} \\
  w = \text{def} (x^{(2)}, \ldots, x^{(k+1)})
\end{array} \right.

\begin{bmatrix}
  F(x, x^\bullet) \\
  F^\bullet(x, x^\bullet) \\
  \vdots \\
  F^{\bullet k}(x, x^\bullet)
\end{bmatrix} = \text{def}

\begin{bmatrix}
  F^0(x, x^\bullet, \omega) \\
  F^1(x, x^\bullet, \omega) \\
  \vdots \\
  F^k(x, x^\bullet, \omega)
\end{bmatrix}, \quad \left\{ \begin{array}{l}
  v = \text{def} x^\bullet \\
  w = \text{def} (x^{\bullet 2}, \ldots, x^{\bullet k+1})
\end{array} \right.

\text{Index} = \text{def} \min k \text{ s.t. } x \mapsto v : \exists w. F_k(x, v, w) = 0 \text{ is a partial function}

solving \( F_k = 0 \) while eliminating \( w \) uniquely determines \( v \) as a partial function of \( x \)
The case of smooth systems ($F$ smooth)

Index $=_{\text{def}} \min k \text{ s.t. } x \mapsto v : \exists w. F_k(x, v, w) = 0$ is a partial function

Whence the following questions of interest if $F$ is smooth:

1. does $x \mapsto v : \exists w. F(x, v, w) = 0$ define a partial function?

   $\Leftrightarrow$ (by implicit function theorem)

2. does $x \mapsto v : \exists \delta w. A \delta v + C \delta w + E \delta x = 0$ define a partial function?

   where $A = F'_v, C = F'_w, E = F'_x$ are Jacobians at a solution $(v_o, w_o, x_o)$ and $\delta v = v - v_0, \delta w = w - w_0$ and $\delta x = x - x_0$
The case of smooth systems ($F$ smooth)

Index $= \text{def} \min k \text{ s.t. } x \mapsto v : \exists w. F_k(x, v, w) = 0$ is a partial function

Whence the following questions of interest if $F$ is smooth:

1. does $x \mapsto v : \exists w. F(x, v, w) = 0$ define a partial function?
   $\Leftrightarrow$ (by implicit function theorem)

2. does $x \mapsto v : \exists w. Av + Cw + Ex = 0$ define a partial function?
   where $A = F'_v$, $C = F'_w$, $E = F'_x$ are Jacobians at a solution $(v_o, w_o, x_o)$

We are interested in structural properties, i.e., properties that are valid outside exceptional values for the nonzero coefficients of the matrices
The case of smooth systems ($F$ smooth)

\[ \text{Subcase } \begin{cases} A^\text{\small structurally invertible} \iff \exists \ P \text{ permutation matrix such that } PA \text{ has a nonzero diagonal} \\ P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \ A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & 0 & a_{23} \\ a_{31} & a_{32} & 0 \end{bmatrix}, \ PA = \begin{bmatrix} a_{21} & 0 & a_{23} \\ a_{31} & a_{32} & 0 \\ a_{11} & a_{12} & a_{13} \end{bmatrix} \end{cases} \]

$A$ is structurally invertible. It may be singular for exceptional values of the nonzero coefficients of $A$, e.g., if $\det(A) = a_{31}a_{12}a_{23} - a_{32}(a_{11}a_{23} - a_{21}a_{13}) = 0$.

Finding $P$ amounts to pivoting, which is a graph based algorithm: reorder equations, and then use the $k$th equation to eliminate $v_k$ as a function of $x$ and $\{v_j | j > k\}$ does $x \mapsto v$:

$\exists w. Av + Cw + Ex = 0$ define a partial function, almost everywhere when the nonzero coefficients of $A, C, E$ vary over some neighborhood?
The case of smooth systems ($F$ smooth)

- Subcase $Av + x = 0$ with $A$ a square matrix: $A$ structurally invertible $\iff \exists P$ permutation matrix such that $PA$ has a nonzero diagonal

A is structurally invertible. It may be singular for exceptional values of the nonzero coefficients of $A$, e.g., if $\det(A) = a_{31}a_{12}a_{23} - a_{32}(a_{11}a_{23} - a_{21}a_{13}) = 0$.

does $x \mapsto v : \exists w. Av + Cw + Ex = 0$ define a partial function, almost everywhere when the nonzero coefficients of $A, C, E$ vary over some neighborhood?
The case of smooth systems ($F$ smooth)

- **Subcase** $Av + x = 0$ with $A$ a square matrix:  
  \[ A \text{ structurally invertible } \iff \exists P \text{ permutation matrix such that } PA \text{ has a nonzero diagonal} \]

\[
P = \begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
1 & 0 & 0
\end{bmatrix},
A = \begin{bmatrix}
a_{11} & a_{12} & a_{13} \\
a_{21} & 0 & a_{23} \\
a_{31} & a_{32} & 0
\end{bmatrix},
PA = \begin{bmatrix}
a_{21} & 0 & a_{23} \\
a_{31} & a_{32} & 0 \\
a_{11} & a_{12} & a_{13}
\end{bmatrix}
\]

$A$ is structurally invertible. It may be singular for exceptional values of the nonzero coefficients of $A$, e.g., if $\det(A) = a_{31}a_{12}a_{23} - a_{32}(a_{11}a_{23} - a_{21}a_{13}) = 0$.

- **Finding $P$ amounts to pivoting**, which is a graph based algorithm:
  1. reorder equations, and then
  2. use the $k$th equation to eliminate $v_k$ as a function of $x$ and \{ $v_j \mid j \succ k$ \}

\[
does \ x \mapsto v : \exists w. Av + Cw + Ex = 0 \ \text{define a partial function, almost everywhere when the nonzero coefficients of } A, \ C, \ E \ \text{vary over some neighborhood?}\]
The case of smooth systems ($F$ smooth)

▶ Subcase $Av + x = 0$ with $A$ a square matrix: $A$ structurally invertible
\[ \Leftrightarrow \exists P \text{ permutation matrix such that } PA \text{ has a nonzero diagonal} \]

\[
P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \quad A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & 0 & a_{23} \\ a_{31} & a_{32} & 0 \end{bmatrix}, \quad PA = \begin{bmatrix} a_{21} & 0 & a_{23} \\ a_{31} & a_{32} & 0 \\ a_{11} & a_{12} & a_{13} \end{bmatrix}
\]

$A$ is structurally invertible. It may be singular for exceptional values of the nonzero coefficients of $A$, e.g., if $\det(A) = a_{31}a_{12}a_{23} - a_{32}(a_{11}a_{23} - a_{21}a_{13}) = 0$.

▶ A similar result holds for structural properties of
\[
\begin{align*}
x & \mapsto v : \exists w. Av + Cw + Ex = 0
\end{align*}
\]

which also leads to a graph based algorithm

\[
\text{does } x \mapsto v : \exists w. Av + Cw + Ex = 0 \text{ define a partial function, almost everywhere when the nonzero coefficients of } A, C, E \text{ vary over some neighborhood?}
\]
The case of smooth systems ($F$ smooth)

\[
\begin{cases}
0 &= \dot{x} - f(x, u) \\
0 &= g(x)
\end{cases}
\]

set $S(x, v) = \begin{bmatrix} \dot{x} - f(x, u) \\ g(x) \end{bmatrix}$

where $x$ is the state and $v = (\dot{x}, u)$; consider the Jacobian

\[
J = dS/dv = \begin{bmatrix} 1 & -f'(x, u) \\ 0 & 0 \end{bmatrix}
\]

is structurally singular.

$v$ cannot be determined and $S$ has index $> 0$. Set $w = (\ddot{x}, \dot{u})$ and consider

\[
S_1(x, v, w) = \begin{bmatrix} S(x, v) \\ \frac{d}{dt} S(x, v, w) \end{bmatrix} = \begin{bmatrix} \dot{x} - f(x, u) \\ g(x) \\ g'(x)\dot{x} \\ \ddot{x} - \frac{d}{dt} f(x, u) \end{bmatrix} \Rightarrow \begin{bmatrix} \dot{x} - f(x, u) \\ g(x) \\ g'(x)\dot{x} \\ \ddot{x} - \frac{d}{dt} f(x, u) \end{bmatrix}
\]

\[
J_1 = dS_1/dv = \begin{bmatrix} 1 & -f'(x, u) \\ 0 & 0 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & -f'(x, u) \\ g'(x) & 0 \end{bmatrix}
\]

invertible.
The case of smooth systems ($F$ smooth)

\[
\begin{align*}
E_1 &: \quad 0 = \dot{x} - f(x, u) \\
E_2 &: \quad 0 = g(x) \\
E_3 &: \quad 0 = g'(x)\dot{x}
\end{align*}
\]

structural pivoting (Pantelides algorithm) \iff \text{searching for a consistent orientation of the incidence graph}

Pantelides algorithm \iff \text{causality analysis for constraint system}
Exact and Structural DAE index (linear algebra reasoning)

Structural dAE index and causality analysis

Through NonStandard semantics DAE become dAE

The index of a Hybrid DAE System is the dAE index of its NS-semantics

Conclusions
The need for guards

\[ E(x, u, v) : b = [x > 0] \land \text{if } b \text{ then } v = f(u) \text{ else } u = g(v) \]

Brute force application of the **Abstraction Principle** yields an incorrect abstraction
The need for guards

\[ E(x, u, v) : b = [x > 0] \land \text{if } b \text{ then } v = f(u) \text{ else } u = g(v) \]

\[ P_E(x, u, v) : x \rightarrow b \land \text{if } b \text{ then } u \rightarrow v \text{ else } u \rightarrow v \]

\[ \overrightarrow{P}_E(x, u, v) : x \rightarrow b \land \text{if } b \text{ then } u \rightarrow v \text{ else } v \rightarrow u \]
The need for guards

Refined formalism: guarded equations (we consider flat guards only)

\[ S = (\{b_1, \ldots, b_n\}, \{E_1, \ldots, E_n\}) \text{ where} \]

\[ b_j = \text{guard appearing in } E_j \]

\[ E_j = \text{if } b_j \text{ then } F_j(x, u, x^\bullet) \text{ and Abstraction Principle applies to } F \]

\[ \mathcal{P}_{E_j} = b_j \rightarrow E_j \land \text{if } b_j \text{ then } F_j \]

Compute a directed covering \( \vec{\mathcal{P}}_{E_j} \) of \( \mathcal{P}_{E_j} \) ensuring

- single assignment for each valuation of guards
- circuitfreeness for each valuation of guards

An extension of the Signal synchronous programming language:

- clock and causality calculus; yields constructive semantics
Summary on dAE causality analysis

Causality analysis (from which the index follows):
- Guarded equations with assertions on guards
- Guarded causality analysis

Constructive semantics and execution schemes
- Execution mode of synchronous languages, albeit
- Evaluating atoms requires dedicated constraint solvers
Exact and Structural DAE index (linear algebra reasoning)

Structural dAE index and causality analysis

Through NonStandard semantics DAE become dAE

The index of a Hybrid DAE System is the dAE index of its NS-semantics

Conclusions
Through NonStandard semantics DAE become dAE

\[
T = \{ t_n = n\partial | n \in *\mathbb{Z} \} \quad \text{where } \partial \text{ is an infinitesimal}
\]

\[
\forall t \in T : \dot{t} = \text{def} \max\{ s \mid s \in T, s < t \} = t - \partial
\]

\[
\dot{t}^\bullet = \text{def} \min\{ s \mid s \in T, s > t \} = t + \partial
\]

\[
\dot{x}_t = \text{def} \frac{x_{t^\bullet} - x_t}{\partial} \quad \text{(explicit scheme)} \quad \left( \frac{x_t - x_{t^\bullet}}{\partial} \quad \text{(implicit scheme)} \right)
\]

\[
\begin{bmatrix}
\frac{d}{dt}x \\
\frac{d^2}{dt^2}x \\
\vdots
\end{bmatrix}
= \mathcal{L}
\begin{bmatrix}
x \\
x^\bullet \\
x^{\bullet^2} \\
\vdots
\end{bmatrix}
, \quad \text{where } \mathcal{L} \text{ is an invertible lower triangular matrix}
\]

Theorem: structural index of a DAE = structural index of its NS-semantics
Through NonStandard semantics DAE become dAE

\[ T = \{ t_n = n\partial \mid n \in \mathbb{Z}^* \} \] where \( \partial \) is an infinitesimal

\[ \forall t \in T : \overset{.}{t} = \text{def} \max\{ s \mid s \in T, s < t \} = t - \partial \]

\[ t^{\bullet} = \text{def} \min\{ s \mid s \in T, s > t \} = t + \partial \]

\[ \dot{x}_t = \text{def} \frac{x_t^{\bullet} - x_t}{\partial} \text{ (explicit scheme)} \]

\[ \left[ \begin{array}{c}
 x \\
 \frac{d}{dt} x \\
 \frac{d^2}{dt^2} x \\
 \vdots
\end{array} \right] = \mathcal{L} \left[ \begin{array}{c}
 x \\
 x^{\bullet} \\
 x^{\bullet 2} \\
 \vdots
\end{array} \right], \text{ where } \mathcal{L} \text{ is an invertible lower triangular matrix} \]

**Theorem:** structural index of a DAE = structural index of its NS-semantics
Exact and Structural DAE index (linear algebra reasoning)

Structural dAE index and causality analysis

Through NonStandard semantics DAE become dAE

The index of a Hybrid DAE System is the dAE index of its NS-semantics

Conclusions
A conservative extension of the index

\[
\text{index of Hybrid DAE System} \overset{\text{def}}{=} \text{dAE index of its NS-semantics}
\]

- By the previous Theorem this yields a conservative extension of
  - the index of a DAE system
  - the index of a dAE system

- Warning: the above result requires considering the structural index
  (not the exact one)

- The computation of the index is a byproduct of the causality analysis
- The index is a global notion (the index may be finite or infinite)
- The causality analysis is guarded, i.e., mode dependent
A conservative extension of the index

\[
\text{index of Hybrid DAE System} =_{\text{def}} \text{dAE index of its NS-semantics}
\]

- By the previous Theorem this yields a conservative extension of
  - the index of a DAE system
  - the index of a dAE system
- Warning: the above result requires considering the structural index (not the exact one)
  - The computation of the index is a byproduct of the causality analysis
  - The index is a global notion (the index may be finite or infinite)
  - The causality analysis is guarded, i.e., mode dependent
A conservative extension of the index

We can perform causality analysis for the following kind of example:

\[ |j_i| < J_i; 0 = u_i \quad \text{ON}_i \quad \overset{|j_i| \geq J_i}{\longrightarrow} \quad \text{OFF}_i \quad 0 \leq U - |u_i| \perp |j_i| \geq 0 \]

A (not so) simple circuit with two fuses. Top: the circuit. Bottom: the mode automaton for each fuse \( i = 1, 2 \). For the ON mode, the current must stay below a threshold \( J_i \), while in the OFF mode, the complementarity condition shown holds.
Exact and Structural DAE index (linear algebra reasoning)

Structural dAE index and causality analysis

Through NonStandard semantics DAE become dAE

The index of a Hybrid DAE System is the dAE index of its NS-semantics

Conclusions
We have formally defined the index for Hybrid DAE

- Index of Hybrid DAE $=\text{def} \quad \text{index of its NonStandard semantics}$
  (Yet another evidence that NonStandard semantics helps...)

- Requires guarded causality analysis alike in synchronous languages
  (particularly Signal)

- Allows giving a mathematical semantics to more Hybrid DAE systems
  (of little help if the index is infinite)
dAE Systems are interesting

dAE Systems with general data types (e.g. Bool = numerics):

▶ Extend synchronous programming to Transition Systems where transition relations (constraints) are specified via systems of equations

▶ Guarded equations with atoms form an expressive syntax

▶ Index analysis for dAE is new (though an easy translation from DAE):
  ▶ relies on guarded causality analysis
  ▶ when the index is finite, index reduction identifies the look-ahead horizon that is sufficient to avoid future blocking

▶ Requires constraint solvers (no all purpose solver exists...)

What are dAE Systems useful for?

▶ model-guided testing?
▶ planification?
▶ ...?
dAE Systems are interesting

dAE Systems with general data types (e.g. Bool = numerics):

- Extend synchronous programming to Transition Systems where transition relations (constraints) are specified via systems of equations
- Guarded equations with atoms form an expressive syntax
- Index analysis for dAE is new (though an easy translation from DAE):
  - relies on guarded causality analysis
  - when the index is finite, index reduction identifies the look-ahead horizon that is sufficient to avoid future blocking
- Requires constraint solvers (no all purpose solver exists...)

What are dAE Systems useful for?

- model-guided testing?
- planification?
- ...?
Hybrid DAE Systems are more difficult than dAE Systems

- Index-and-causality analysis is a symbolic pre-processing. It does not address numerical difficulties.
- Making a step is more difficult than calling an S-function.

Global vs. Distributed solvers
- Global solver is normally used for simulation.
- Distributed solvers are used:
  - in code coupling (e.g. multi-physics)
  - in slow/fast dynamics
  - in FMI based simulation with several FMU

What to do with events?
- Handling every discontinuity as an event is not good.
- Handling no discontinuity as an event is not good either.
Hybrid DAE Systems are more difficult than dAE Systems

- Index-and-causality analysis is a symbolic pre-processing. It does not address numerical difficulties.

- Making a step is more difficult than calling an S-function.

- Global vs. Distributed solvers
  - Global solver is normally used for simulation.
  - Distributed solvers are used:
    - in code coupling (e.g. multi-physics)
    - in slow/fast dynamics
    - in FMI based simulation with several FMU

- What to do with events?
  - Handling every discontinuity as an event is not good.
  - Handling no discontinuity as an event is not good either.
Hybrid DAE Systems are more difficult than dAE Systems

- Index-and-causality analysis is a symbolic pre-processing.
  It does not address numerical difficulties.

- Making a step is more difficult than calling an S-function.

- Global vs. Distributed solvers
  - Global solver is normally used for simulation.
  - Distributed solvers are used:
    - in code coupling (e.g. multi-physics)
    - in slow/fast dynamics
    - in FMI based simulation with several FMU

- What to do with events?
  - Handling every discontinuity as an event is not good.
  - Handling no discontinuity as an event is not good either.
Hybrid DAE Systems are more difficult than dAE Systems

- Index-and-causality analysis is a symbolic pre-processing. It does not address numerical difficulties.

- Making a step is more difficult than calling an S-function.

- Global vs. Distributed solvers
  - Global solver is normally used for simulation.
  - Distributed solvers are used:
    - in code coupling (e.g. multi-physics)
    - in slow/fast dynamics
    - in FMI based simulation with several FMU.

- What to do with events?
  - Handling every discontinuity as an event is not good.
  - Handling no discontinuity as an event is not good either.
an exciting but difficult subject