Handling Multimode Models and Mode Changes in Modelica

Albert Benveniste, Benoît Caillaud, Mathias Malandain

Handling Multimode Models and Mode Changes in Modelica

Albert Benveniste, Benoît Caillaud, Mathias Malandain[1]

[1]Inria Centre de Rennes Bretagne Atlantique, University of Rennes 1, France,
{albert.benveniste,benoit.caillaud,mathias.malandain}@inria.fr

Abstract

Since its version 3.3, the Modelica language offers the possibility to model multimode systems having different DAE-based dynamics in each mode, thanks to the introduction of state machines. When the differentiation index and structure varies with mode changes, compilers generate erroneous simulation code, often resulting in runtime exceptions. We propose in this paper a multimode structural analysis for both multiple modes and mode change events and we show how correct code for restarts can be generated. Our approach is illustrated on two simple but representative mechanical systems.
Keywords: multimode DAE, structural analysis

1 Introduction

Since version 3.3, the Modelica language offers the possibility of specifying *multimode dynamics,* by describing state machines with different DAE dynamics in each different state (Elmqvist et al., 2012). This feature enables describing large complex cyber-physical systems with different behaviors in different modes.

While being undoubtedly valuable, multimode modeling has been the source of serious difficulties for non-expert users of the current generation of Modelica tools. Indeed, while many large-scale Modelica models are properly handled, some physically meaningful models do not result in correct simulations with most Modelica tools. As such problematic models are actually easy to construct, the likelihood of such bad cases occurring in large models is significant.

It is unfortunately unclear which multimode Modelica models will be properly handled, and which ones will fail. As a consequence, quite often, end users have to ask Modelica experts, or even tool developers themselves, to tweak their models in order to make them work as expected. While it is accepted that physical modeling itself requires expertise, requiring expertise in how to get around tool idiosyncrasies is not desirable. This situation hinders a wider spreading of Modelica tools among a larger class of users, such as Simulink-trained engineers.

As our review of two examples will reveal, this problem is mainly due to an inadequate structural analysis, performed during compilation. As far as we know, no industrial-strength Modelica tool implements a mode-dependent structural analysis—a few academic prototypes address this difficulty in part, see Section 3. Worse, it is not even understood what kind of structural analysis should be associated with mode change events.

Some years ago, we started a project aiming at addressing all the above issues. In this paper we explain our approach, by illustrating it on two simple yet physically meaningful examples that current Modelica tools fail to properly simulate. The use of nonstandard analysis allows us to perform the analysis of both modes and mode changes in a unified framework, including the handling of transient modes and that of impulsive mode changes. Standardization techniques are then used in order to generate effective code for restarts at mode changes. As an efficient implementation of such methods in Modelica compilers would greatly expand the class of multimode models amenable to reliable numerical simulation, we hint at possible mechanizations towards the end of the paper; this aspect is developed in both the companion paper (Benveniste et al., 2021). and the previously published article (Caillaud et al., 2020).

2 Two problematic examples

We review two small examples of multimode DAE systems and analyse how they are handled by two state-of-the-art Modelica tools, OpenModelica and Dymola.

2.1 An ideal clutch

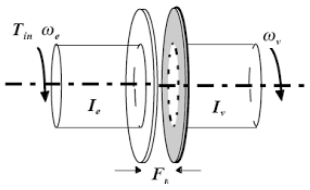

Figure 1. An ideal clutch with two shafts.

The clutch depicted in Figure 1 is an idealized clutch interconnecting two rotating shafts. It is assumed that this system is closed, meaning that the two shafts are not connected to anything else, whence the corresponding model:

$$\left\{\begin{array}{lll} & & \omega_1' = f_1(\omega_1,\tau_1) \quad (e_1)\\ & & \omega_2' = f_2(\omega_2,\tau_2) \quad (e_2)\\ \texttt{if } \gamma & \texttt{do} & \omega_1-\omega_2=0 \quad (e_3)\\ & \texttt{and} & \tau_1+\tau_2=0 \quad (e_4)\\ \texttt{if not } \gamma & \texttt{do} & \tau_1=0 \quad (e_5)\\ & \texttt{and} & \tau_2=0 \quad (e_6) \end{array}\right. \tag{1}$$

In model (1), the dynamics of each shaft i is described by ODE $\omega_i' = f_i(\omega_i, \tau_i)$ for some, yet unspecified, function f_i, where ω_i is the angular velocity and τ_i is the torque applied to shaft i. Depending on the value of the input Boolean variable γ, the clutch is either engaged ($\gamma = \text{T}$, the constant "true") or released ($\gamma = \text{F}$, the constant "false"). When the clutch is released, the two shafts rotate freely: no torque is applied to them ($\tau_i = 0$). When the clutch is engaged, it ensures a perfect join between the two shafts, forcing them to have the same angular velocity ($\omega_1 - \omega_2 = 0$) and opposite torques ($\tau_1 + \tau_2 = 0$). When $\gamma = \text{T}$, equations (e_3, e_4) are active and equations (e_5, e_6) are disabled, and vice-versa when $\gamma = \text{F}$. If the clutch is initially released, then, at the instant of contact, the relative speed of the two rotating shafts jumps to zero; as a consequence, an impulse is expected on the torques.

The model yields an ODE system when the clutch is released, and a DAE system of index 1 when the clutch is engaged (see Section 5.1).

The clutch in Modelica: Figure 2 details the Modelica model of the Ideal Clutch system. It is a faithful translation in the Modelica language of the two-mode DAE (1), except that the two differential equations have been linearized. Also, the trajectory of the input guard γ (here called `g`) has been fully specified: it takes the value T between t_1 and t_2 and F otherwise.

```
model ClutchBasic
parameter Real w01=1;
parameter Real w02=1.5;
parameter Real j1=1;
parameter Real j2=2;
parameter Real k1=0.01;
parameter Real k2=0.0125;
parameter Real t1=5;
parameter Real t2=7;
Real t(start=0, fixed=true);
Boolean g(start=false);
Real w1(start = w01, fixed=true);
Real w2(start = w02, fixed=true);
Real f1;  Real f2;
equation
  der(t) = 1;
  g = (t >= t1) and (t <= t2);
  j1*der(w1) = -k1*w1 + f1;
  j2*der(w2) = -k2*w2 + f2;
  0 = if g then w1-w2 else f1;
  f1 + f2 = 0;
end ClutchBasic;
```

Figure 2. Modelica code for the idealized clutch.

This model is deemed structurally nonsingular by the two Modelica tools we had the opportunity to test: OpenModelica 1.17.0 (Fritzson et al., 2020) and Dymola 2021 (Dassault Systèmes AB). However, none of these tools generates correct simulation code from this model. Indeed, simulations fail precisely at the instant when the clutch switches from the uncoupled mode (`g=false`) to the coupled one (`g=true`). This is evidenced by a division by zero exception, as shown in Figure 3.

```
Log-file of program ./dymosim
(generated: Tue Apr  6 08:10:09 2021)

dymosim started
... "dsin.txt" loading (dymosim input file)
... "ClutchBasic.mat" creating (simulation result file)

Integration started at T = 0 using integration method DASSL
(DAE multi-step solver (dassl/dasslrt of Petzold modified by Dassault Systemes))
Error: The following error was detected at time: 5
Error: Singular inconsistent scalar system for f1 = ((if g then w1-w2 else 0.0))/( -(if g then 0.0 else 1.0)) = -0.502623/-0
Integration terminated before reaching "StopTime" at T = 5
States and derivatives:
t=5 1
w1=0.951225 -0.00951225
w2=1.45385 -0.00908655
```

```
The initialization finished successfully without homotopy method.
division by zero at time 5.000000000600098, (a=0.5026218926585789) / (b=0), where divisor b expression is: if g then 0.0 else 1.0

Debug more
Simulation process failed. Exited with code 255.
```

Figure 3. Division by zero exceptions with Dymola 2021 (top) and OpenModelica 1.17.0 (bottom) occuring when simulating the Ideal Clutch Modelica model.

The cause of this exception is that none of these tools performs a multimode structural analysis. Instead, the structure of the model is assumed invariant, and a Dummy Derivatives method (Mattsson and Söderlind, 1993) is implemented, which is correct on single-mode DAE systems, whereas it may fail on multimode systems unless the model structure is independent of the mode. The structural analysis methods in these tools do not detect that the differentiation index jumps from 0 to 1 when the shafts are coupled, and that the structure is not invariant. The division by zero results from the pivoting of a linear system of equations that becomes singular when `g` becomes equal to `true`.

2.2 A Cup-and-Ball game

Figure 4. The Cup-and-Ball game.

We sketch here a multimode extension of the popular example of the pendulum in Cartesian coordinates (Pantelides, 1988), namely the Cup-and-Ball game illustrated by Figure 4. A ball, modeled by a point mass, is attached to one end of a rope, while the other end of the rope is fixed, to the origin of the plane in the model. The ball is subject to the unilateral constraint set by the rope, but moves freely while the distance between the ball and the origin is less than its length. The system is assumed closed. The model for a 2D-version of this example is:

$$\begin{cases} 0 = x'' + \lambda x & (e_1) \\ 0 = y'' + \lambda y + g & (e_2) \\ 0 \leq L^2 - (x^2 + y^2) & (\kappa_1) \\ 0 \leq \lambda & (\kappa_2) \\ 0 = \left[L^2 - (x^2 + y^2)\right] \times \lambda & (\kappa_3) \end{cases} \tag{2}$$

where the dependent variables are the position (x, y) of the ball in Cartesian coordinates and the rope tension λ.

The subsystem $(\kappa_1, \kappa_2, \kappa_3)$ expresses that the tension is nonnegative, the distance of the ball from the origin is less

than or equal to L, and one cannot have a nonzero tension and a distance less than L at the same time. Constraints κ_1 and κ_2 are unilateral, which is not supported by Modelica and related languages. Therefore, using the technique presented in (Mattsson et al., 1999), we redefine the graph of this complementarity condition as a parametric curve, represented by the following three equations:

$$\begin{array}{rcl} s & = & \texttt{if } \gamma \texttt{ then } -\lambda \texttt{ else } L^2-(x^2+y^2) \\ 0 & = & \texttt{if } \gamma \texttt{ then } L^2-(x^2+y^2) \texttt{ else } \lambda \\ \gamma & = & [s \leq 0] \end{array} \qquad (3)$$

Similarly to the clutch model, impulsive behavior is expected on the torques. However, an other possible difficulty is present: subsystem $(\kappa_1, \kappa_2, \kappa_3)$ of (2) leaves the impact law at mode change insufficiently specified; it could be fully elastic, fully inelastic, or in between. *Can both of these aspects be detected at compile time, using some kind of structural analysis?*

The Cup-and-Ball in Modelica: Figure 5 details the Modelica model of the Cup-and-Ball game. It is a faithful translation of the two-mode DAE (2) using rewriting (3). The point mass, modeling the ball, initially stands at the origin of the plane with zero velocity; the Boolean guard γ, named `gamma` in the model, is thus set to `false`.

```
model CupAndBall
constant Real g=9.81;
constant Real L=1.0;
Real x(start=0,fixed=true);
Real y(start=0,fixed=true);
Real u(start=0,fixed=true);
Real v(start=0,fixed=true);
Real lambda;
Real s;
Boolean gamma(start=false,fixed=true);
equation
   der(x) = u;
   der(y) = v;
   der(u) + lambda*x = 0;
   der(v) + lambda*y + g = 0;
   gamma = (s <= 0);
   0 = if gamma then L^2 - (x^2 + y^2)
               else lambda;
   s = if gamma then - lambda
               else L^2 - (x^2 + y^2);
end CupAndBall;
```

Figure 5. Modelica code for the Cup-and-Ball.

As is the case for the clutch model presented above, this model is deemed structurally nonsingular by both OpenModelica 1.17.0 and Dymola 2021, but the simulation fails at the instant of mode change. Figure 6 depicts the resulting trajectory of variables `y` and `gamma`; it ends when `gamma` switches from `false` to `true`, as the tool is unable to correctly reinitialize the model after the mode change. Replacing condition `s <= 0` with `last(s) <= 0` in order to break the fixpoint equation defining variable `gamma` (see the introduction of Section 6) leads to the same simulation results, but with a division by zero error similar to that shown in Figure 3 occurring at the moment of mode change.

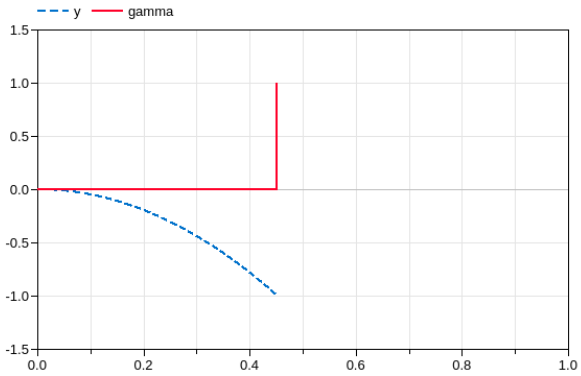

Figure 6. Trajectory of the Cup-and-Ball Modelica model: it stops around $t = 0.452s$, when the rope becomes straight.

.

It appears from both exemples that some fundamental study is needed to correctly simulate multimode models, and that the problem is twofold: the varying structure of the model has to be taken into account, and mode changes have to be handled in a specific fashion.

Smoothing '`if then else`' equations could help solve both issues by essentially turning multimode models into single-mode models, but this requires a delicate and definitely non-modular tuning, as it depends on the different time scales arising in the system. We believe that, as the tools reputedly support multimode DAE models, they should handle them correctly.

3 Related work

For the general literature on DAE and Modelica, we refer the reader to https://www.modelica.org/publications and (Benveniste et al., 2019, 2020).

(Elmqvist et al., 2014; Mattsson et al., 2015) propose a high-level description of multimode models as an extension to the synchronous Modelica 3.3 state machines, by using continuous-time state machines having continuous-time models as "states". State machines are transformed so that the resulting equations can be processed by standard symbolic algorithms supported by Modelica tools. Describing variable-structure systems with *causal* state machines is discussed in (Pepper et al., 2011). Dynamically changing the structural analysis at runtime is also proposed by (Höger, 2014, 2017), with (Höger, 2014) proposing a dynamic execution of the Σ-method (Pryce, 2001), and by (Nilsson and Giorgidze, 2010) in the context of their Functional Hybrid Modelling paradigm. Such approaches typically rely on the explicit declaration of reinitializations at mode changes.

As such, the computation of correct restarts at mode changes, while being a central issue in multi-mode DAE systems, is not being tackled in the already mentioned references. Some authors still address this issue.

(Benveniste et al., 2017) tackled this issue, as well as the problem of varying structure and index, from a fundamental point of view, by relying on nonstandard analysis to capture continuous-time dynamics and mode change events in a unified framework. A first structural analysis algorithm was presented in this paper, by significantly modifying the original Pantelides algorithm (Pantelides, 1988). This first attempt suffers from some deficiencies: the proposed structural analysis does not boil down to the Pantelides algorithm in the case of single-mode systems; it involves nondeterministic decisions, an unwanted feature for the mathematical foundation of compilers; and its mathematical study is incomplete.

(Trenn, 2009a,b) are important works as they point out the difficulty in defining piecewise smooth distributions. Liberzon and Trenn were able to define complete solutions for a class of switched DAE systems in which each mode is in *quasi-linear form* (Liberzon and Trenn, 2012): notably, switching conditions are time-based only.

In (Benveniste et al., 2019), an interesting subclass of multimode DAE systems was identified, which possibly exhibit impulsive variables at mode changes. They extend the "quasi-linear systems" proposed by Trenn et al.; in particular, switching conditions are no longer restricted to be time-based, but can be state-based. Nevertheless, the analysis and discretization schemes proposed in (Benveniste et al., 2019) are mathematically sound. Building on this work, Elmqvist and Otter have developed the ModiaSim[1] Julia packages for semi-linear multimode DAE systems. It turns out that the general approach of the present paper coincides with the schemes proposed in (Benveniste et al., 2019) when applied to the considered subclass. Our present contribution thus extends and significantly improves that work. An in-depth comparison can be found in (Benveniste et al., 2020).

4 Our contributions

Structural analysis of mode changes and code generation for restarts: *We develop a structural analysis that is valid at any time, that is, for both continuous dynamics and mode changes.* Impulsive behaviors may occur at mode changes for certain variables. Whereas the few current tools able to handle such situations discover them at runtime, our structural analysis covers these situations and handles them at compile time.

Rejecting or accepting programs on a clear basis, at compile time: Our structural analysis is precise enough to properly identify models that are structurally over- or under-specified at mode change events. In turn, mode-dependent index/state/dynamics are not reasons for rejection: our approach handles such cases.

We now move to developing our approach by discussing the two examples from Section 2.

[1] https://modiasim.github.io/docs/index.html

5 The ideal clutch

Its model was given in (1). We first analyze separately the model for each mode of the clutch. Then, we focus on mode changes and propose a comprehensive analysis.

5.1 Separate Analysis of Each Mode

In the released mode, i.e., when $\gamma = \text{F}$ in System (1), the two shafts are independent and one obtains the following two independent ODEs for ω_1 and ω_2:

$$\begin{array}{llll} \omega_1' = f_1(\omega_1, \tau_1) & (e_1) & \tau_1 = 0 & (e_5) \\ \omega_2' = f_2(\omega_2, \tau_2) & (e_2) & \tau_2 = 0 & (e_6) \end{array} \tag{4}$$

In the engaged mode, however ($\gamma = \text{T}$), the two velocities and torques are algebraically related:

$$\begin{array}{llll} \omega_1' = f_1(\omega_1, \tau_1) & (e_1) & \omega_1 - \omega_2 = 0 & (e_3) \\ \omega_2' = f_2(\omega_2, \tau_2) & (e_2) & \tau_1 + \tau_2 = 0 & (e_4) \end{array} \tag{5}$$

System (5) is a DAE. Its structural analysis tells that equation (e_3) must be differentiated and added to the model (it is highlighted in red):

$$\begin{array}{llll} \omega_1' = f_1(\omega_1, \tau_1) & (e_1) & \omega_1 - \omega_2 = 0 & (e_3) \\ \omega_2' = f_2(\omega_2, \tau_2) & (e_2) & \omega_1' - \omega_2' = 0 & (e_3') \\ & & \tau_1 + \tau_2 = 0 & (e_4) \end{array} \tag{6}$$

Although this change of differentiation index is the root cause of the runtime exceptions shown in Figure 3, solving this issue would not be enough for the correct simulation of the model, because of the need of handling mode changes.

As a matter of fact, while the cold initialization of the engaged mode yields 6 dependent variables for only 5 equations, thus leaving one degree of freedom (the common velocity of the two shafts), the mode change $\gamma : \text{F} \to \text{T}$, when the clutch gets engaged, is physically determinate, which makes the point that mode changes cannot be handled as "cold restarts".

Inferring by hand the reset values for rotation velocities when the clutch gets engaged is definitely non-trivial. Furthermore, these values depend on the whole system model, so that the task of determining them becomes complex if external components are added.

***It is therefore highly desirable, for this example, to let the compiler infer these reset values from model* (1).**

5.2 Mapping model to nonstandard analysis

If DAE dynamics is approximated in discrete time, then the whole model becomes discrete-time. To avoid the problem of approximation error, our idea is to use an "infinitesimal" time step in the discrete time approximation. This will yield an approximation up to an infinitesimal accuracy.

This can be made rigorous by relying on *nonstandard analysis* (Robinson, 1996; Lindstrøm, 1988; Benveniste et al., 2020), which extends the set $\mathbb{R}$ of real numbers to a superset ${}^\star\mathbb{R}$ of *hyperreals* that includes infinite sets of infinitely large numbers and infinitely small numbers.

For the understanding of this paper, it is enough to know the following about nonstandard analysis. There exist *infinitesimals*, defined as hyperreals that are smaller in absolute value than any real number. The arithmetic operations $+$, $\times$, etc., and usual relations, are lifted to ${}^\star\mathbb{R}$. For every finite hyperreal $x \in {}^\star\mathbb{R}$, there is a unique standard real number $\mathsf{st}(x) \in \mathbb{R}$ such that $\mathsf{st}(x) - x$ is infinitesimal, and $\mathsf{st}(x)$ is called the *standard part* (or *standardization*) of x. Standardizing functions or systems of equations, however, requires some care. One important issue is derivatives. For $t \mapsto x(t)$ an $\mathbb{R}$-valued (standard) signal ($t \in \mathbb{R}$),

$$\begin{array}{l} x \text{ is differentiable at instant } t{\in}\mathbb{R} \text{ if and only if there} \\ \text{exists } a{\in}\mathbb{R} \text{ such that, for any infinitesimal } \partial \in {}^\star\mathbb{R}, \\ \frac{x(t+\partial)-x(t)}{\partial} - a \text{ is infinitesimal; then, } a = x'(t). \end{array} \tag{7}$$

We can then consider the time index set $\mathbb{T} \subseteq {}^\star\mathbb{R}$:

$$\mathbb{T} = 0, \partial, 2\partial, 3\partial, \cdots = \{n\partial \mid n \in {}^\star\mathbb{N}\} \tag{8}$$

where ${}^\star\mathbb{N}$ denotes the set of *hyperintegers*, consisting of all integers augmented with additional infinite numbers called *nonstandard*, and ∂ is an arbitrary, but fixed, infinitesimal.[2] The following features of $\mathbb{T}$ are important: (1) any finite real time $t \in \mathbb{R}$ is infinitesimally close to some element of $\mathbb{T}$ (hence, $\mathbb{T}$ covers $\mathbb{R}$ and can be used to index continuous-time dynamics); and (2) $\mathbb{T}$ is "discrete": every instant $n\partial$ has a predecessor $(n-1)\partial$ (except for $n = 0$) and a successor $(n+1)\partial$.

Let x be a nonstandard signal indexed by $\mathbb{T}$. The *forward-* and *backward-shifted* signals $x^\bullet$ and ${}^\bullet x$ are defined by:

$$x^\bullet(n\partial) =_{\text{def}} x((n+1)\partial) \quad \text{and} \quad {}^\bullet x((n+1)\partial) =_{\text{def}} x(n\partial)\,,$$

implying that an initial value for ${}^\bullet x(0)$ must be provided. For $f(X)$ a function of the tuple X of signals, we set $(f(X))^\bullet =_{\text{def}} f(X^\bullet)$ where the forward shift $X \mapsto X^\bullet$ applies pointwise to all the components of the tuple. For example, $f^\bullet(x,y)(t) = f(x^\bullet, y^\bullet) = f(x(t+\partial), y(t+\partial))$.

Using (7), we represent, up to an infinitesimal, the derivative x' of a signal by its first-order explicit Euler approximation $\frac{1}{\partial}(x^\bullet - x)$. Solutions of multi-mode DAE systems may be non-differentiable or even non-continuous at events of mode change. To give a meaning to x' at any instant, *we **define** it everywhere as*

$$x' =_{\text{def}} \frac{1}{\partial}(x^\bullet - x)\,. \tag{9}$$

The nonstandard expansion of two-mode system (4,6) is:

$$\left\{\begin{array}{lll} & \frac{\omega_1^\bullet - \omega_1}{\partial} = f_1(\omega_1, \tau_1) & (e_1^\partial) \\ & \frac{\omega_2^\bullet - \omega_2}{\partial} = f_2(\omega_2, \tau_2) & (e_2^\partial) \\ \texttt{if } \gamma \texttt{ do} & \omega_1 - \omega_2 = 0 & (e_3) \\ \texttt{and} & \omega_1^\bullet - \omega_2^\bullet = 0 & (e_3^\bullet) \\ \texttt{and} & \tau_1 + \tau_2 = 0 & (e_4) \\ \texttt{if not } \gamma \texttt{ do} & \tau_1 = 0 & (e_5) \\ \texttt{and} & \tau_2 = 0 & (e_6) \end{array}\right. \tag{10}$$

[2] It is proved in (Benveniste et al., 2020) that the simulation code that is finally generated does not depend on the choice of this infinitesimal time step.

Note that the latent differentiated equation (e_3') of model (6) has been replaced by the forward shifted equation $(e_3^\bullet)$ (both are equivalent from a structural point of view). The state variables are ω_1, ω_2 whereas the leading variables are now τ_1, τ_2, $\omega_1^\bullet$, $\omega_2^\bullet$, in both modes $\gamma = \text{F}$ and $\gamma = \text{T}$. This yields a sort of explicit Euler scheme for model (1), which is exact up to infinitesimals within each mode. The structural analysis is correct in each mode.

5.3 Structural analysis of mode change γ:F→T

We focus on mode change $\gamma : \text{F} \to \text{T}$, when the clutch gets engaged. At the considered instant, we have ${}^\bullet\gamma = \text{F}$ and $\gamma = \text{T}$. We unfold System (10) at the two successive (previous and current) instants by taking the actual values for the guard at those instants into account:

$$\left\{\begin{array}{lll} \text{previous instant } {}^\bullet\gamma = \text{F} & & \\ \left\{\begin{array}{ll} \frac{\omega_1 - {}^\bullet\omega_1}{\partial} = f_1({}^\bullet\omega_1, {}^\bullet\tau_1) & ({}^\bullet e_1^\partial) \\ \frac{\omega_2 - {}^\bullet\omega_2}{\partial} = f_2({}^\bullet\omega_2, {}^\bullet\tau_2) & ({}^\bullet e_2^\partial) \\ {}^\bullet\tau_1 = 0 & \\ {}^\bullet\tau_2 = 0 & \end{array}\right. & \\ \text{current instant } \gamma = \text{T} & & \\ \left\{\begin{array}{ll} \frac{\omega_1^\bullet - \omega_1}{\partial} = f_1(\omega_1, \tau_1) & \\ \frac{\omega_2^\bullet - \omega_2}{\partial} = f_2(\omega_2, \tau_2) & \\ \omega_1 - \omega_2 = 0 & (e_3) \\ \omega_1^\bullet - \omega_2^\bullet = 0 & \\ \tau_1 + \tau_2 = 0 & \end{array}\right. & \end{array}\right. \tag{11}$$

We regard System (11) as an algebraic system of equations with dependent variables ${}^\bullet\tau_i, \omega_i; \tau_i, \omega_i^\bullet$ for $i = 1, 2$, i.e., the leading variables of System (10) at the previous and current instants. System (11) is structurally singular, as it includes the following subsystem[3] which has five equations and only four dependent variables $\omega_1, \omega_2, {}^\bullet\tau_1, {}^\bullet\tau_2$:

$$\left\{\begin{array}{ll} \frac{\omega_1 - {}^\bullet\omega_1}{\partial} = f_1({}^\bullet\omega_1, {}^\bullet\tau_1) & ({}^\bullet e_1^\partial) \\ \frac{\omega_2 - {}^\bullet\omega_2}{\partial} = f_2({}^\bullet\omega_2, {}^\bullet\tau_2) & ({}^\bullet e_2^\partial) \\ {}^\bullet\tau_1 = 0 & \\ {}^\bullet\tau_2 = 0 & \\ \omega_1 - \omega_2 = 0 & (e_3) \end{array}\right. \tag{12}$$

We resolve this conflict by applying the following principle:

Principle 1 (causality) *What was done at the previous instant cannot be undone at the current instant.*

Applying (1) leads to removing, from subsystem (12), the conflicting equation (e_3). This yields the following nonstandard code for the restart at mode change $\gamma : \text{F} \to \text{T}$:

$$\left\{\begin{array}{l} \omega_1, \omega_2, {}^\bullet\tau_1, {}^\bullet\tau_2 \text{ set by previous instant} \\ \omega_1^\bullet = \omega_1 + \partial \times f_1(\omega_1, \tau_1) \\ \omega_2^\bullet = \omega_2 + \partial \times f_2(\omega_2, \tau_2) \\ \omega_1^\bullet - \omega_2^\bullet = 0 \\ \tau_1 + \tau_2 = 0 \end{array}\right. \tag{13}$$

[3] Over- and underdetermined subsystems are structurally found by computing the Dulmage-Mendelsohn decomposition of the system (Dulmage and Mendelsohn, 1958).

The consistency equation $(e_3) : \omega_1 - \omega_2 = 0$ has been removed from System (13), thus modifying the original model. However, this removal occurs only at mode change events $\gamma : \text{F} \to \text{T}$. What we have done amounts to *delaying by one nonstandard instant the satisfaction of some of the constraints in force in the new mode* $\gamma = \text{T}$. Since our time step ∂ is infinitesimal, this takes zero standard time.

5.4 Generating effective code for restart at mode change $\gamma : \text{F} \to \text{T}$

We wish to use System (13) by identifying current values for the states ω_i with the *left-limits* ω_i^- i.e., the values of the velocities just before the mode change. From these values, we would then compute the restart values for the velocities $\omega_i^+ =_{\text{def}} \omega_i^\bullet$, together with the torques τ_i.

Unfortunately, hyperreals are unknown to computers, hence, System (13) cannot be used as such, but needs to be *standardized,* by "washing out" ∂. Since the time step ∂ is infinitesimal, it is tempting to get rid of of it in (13) by simply setting $\partial = 0$. Unfortunately, doing this leaves us with a structurally singular system, since the two torques are then involved in only one equation.

This problem of structural singularity is in fact due to the existence of impulsive variables. To discover them in a systematic way, we perform an *impulse analysis*.

Impulse analysis: Before engaging the clutch, we must generically assume $\omega_1 - \omega_2 \neq 0$. Since $\omega_1^\bullet - \omega_2^\bullet = 0$ holds, $\frac{(\omega_1^\bullet - \omega_2^\bullet) - (\omega_1 - \omega_2)}{\partial} = f_1(\omega_1, \tau_1) - f_2(\omega_2, \tau_2)$ cannot be finite because, if it was, then the function $\omega_1 - \omega_2$ would be continuous, contradicting the assumption that $\omega_1 - \omega_2 \neq 0$. Hence, the hyperreal $f_1(\omega_1, \tau_1) - f_2(\omega_2, \tau_2)$ is necessarily infinite. However, we assumed continuous functions f_i and finite state (ω_1, ω_2). Thus, one of the torques τ_i must be infinite at mode change, and because of equation $(e_4) : \tau_1 + \tau_2 = 0$, both torques are in fact infinite, i.e., are *impulsive*.

Eliminating impulsive variables: We now assume that the f_i's are linear in the torques, i.e., each f_i has the form

$$f_i(\omega_i, \tau_i) \quad = \quad a_i(\omega_i) + b_i(\omega_i)\tau_i\,, \tag{14}$$

where b_1 and b_2 are the inverse moments of inertia of the rotating masses and a_1 and a_2 are damping factors divided by the corresponding moments of inertia. This yields the following system of equations, to be solved for $\omega_1^\bullet, \omega_2^\bullet, \tau_1, \tau_2$ at the instant when γ switches from F to T:

$$\left\{\begin{array}{ll} \omega_1^\bullet = \omega_1 + \partial\,(a_1(\omega_1) + b_1(\omega_1)\tau_1) & (e_1^\partial) \\ \omega_2^\bullet = \omega_2 + \partial\,(a_2(\omega_2) + b_2(\omega_2)\tau_2) & (e_2^\partial) \\ \omega_1^\bullet - \omega_2^\bullet = 0 & (e_3^\bullet) \\ \tau_1 + \tau_2 = 0 & (e_4) \end{array}\right. \tag{15}$$

We now eliminate the impulsive variables from System (15), namely, the two torques. Using (e_4) yields $-\tau_2 = \tau_1 =_{\text{def}} \tau$. Premultiplying the system of equations

$$\left\{\begin{array}{ll} \omega_1^\bullet = \omega_1 + \partial\,(a_1(\omega_1) + b_1(\omega_1)\tau) & (e_1^\partial) \\ \omega_2^\bullet = \omega_2 + \partial\,(a_2(\omega_2) - b_2(\omega_2)\tau) & (e_2^\partial) \end{array}\right.$$

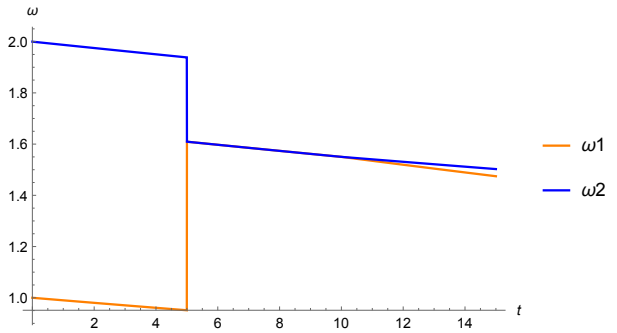

Figure 7. Simulation of the clutch model with resets. Mode change $\text{F} \to \text{T}$ occurs at $t = 5s$ and mode change $\text{T} \to \text{F}$ occurs at $t = 10s$.

by the row matrix $\left[\begin{array}{cc} b_2(\omega_2) & b_1(\omega_1) \end{array}\right]$ yields

$$\begin{array}{l} b_2(\omega_2)\,\omega_1^\bullet + b_1(\omega_1)\,\omega_2^\bullet = \\ b_2(\omega_2)\,(\omega_1 + \partial\,a_1(\omega_1)) + b_1(\omega_1)\,(\omega_2 + \partial\,a_2(\omega_2))\,. \end{array}$$

Using in addition $(e_3^\bullet)$ and setting $\omega^\bullet =_{\text{def}} \omega_1^\bullet = \omega_2^\bullet$ yields

$$\begin{array}{rcl} \omega^\bullet & = & \dfrac{b_2(\omega_2)\omega_1 + b_1(\omega_1)\omega_2}{b_1(\omega_1) + b_2(\omega_2)} \\ & & +\partial\,\dfrac{a_1(\omega_1)b_2(\omega_2) + a_2(\omega_2)b_1(\omega_1)}{b_1(\omega_1) + b_2(\omega_2)} \end{array} \tag{16}$$

It is now legitimate to set $\partial = 0$ in its right-hand side. This yields, by identifying $\mathsf{st}(\omega_i) = \omega_i^-$ and $\mathsf{st}(\omega_i^\bullet) = \omega_i^+$:

$$\omega_1^+ = \omega_2^+ = \frac{b_2(\omega_2^-)\,\omega_1^- + b_1(\omega_1^-)\,\omega_2^-}{b_1(\omega_1^-) + b_2(\omega_2^-)}, \tag{17}$$

where we recall that $\mathsf{st}(\omega)$ is the standard part of ω, see the beginning of Section 5.2. Eq. (17) provides us with the reset values for the positions in the engaged mode, which is enough to restart the simulation in this mode.

Figure 7 shows a simulation of the clutch where the resets are computed following this approach. As expected, the reset value sits between the two values of ω_1^- and ω_2^- when $\gamma : \text{F} \to \text{T}$ (at $t = 5$s), and the transition is continuous at the second reset (at $t = 10$s). An alternative approach for the computation of the reset values, which does not require the elimination of impulsive variables, is developed in (Benveniste et al., 2020), see also (Benveniste et al., 2021).

6 The Cup-and-Ball example

Using (3), the original model (2) is rewritten as

$$\left\{\begin{array}{rll} & 0 = x'' + \lambda x & (e_1) \\ & 0 = y'' + \lambda y + g & (e_2) \\ & \gamma = [s \leq 0] & (k_0) \\ \texttt{if } \gamma \texttt{ do} & 0 = L^2 - (x^2 + y^2) & (k_1) \\ \texttt{and} & 0 = \lambda + s & (k_2) \\ \texttt{if not } \gamma \texttt{ do} & 0 = \lambda & (k_3) \\ \texttt{and} & 0 = (L^2 - (x^2 + y^2)) - s & (k_4) \end{array}\right. \tag{18}$$

As stated in Section 2.2, two issues have to be addressed by our structural analysis: the expected impulsive behavior of the accelerations at mode changes, and the insufficient specification of the nature (elastic, inelastic or in between) of the impact.

We implicitly add to model (18) the following two equations, for each state variable v:

$$v' = \frac{v^\bullet - v}{\partial} \;;\; v'' = \frac{v^{\bullet 2} - 2v^\bullet + v}{\partial^2}, \tag{19}$$

where

$$\begin{array}{lll} v^\bullet(t) & =_{\text{def}} & v(t+\partial), \\ v^{\bullet 2}(t) & =_{\text{def}} & v(t+2\partial) \text{ and, more generally,} \\ v^{\bullet n}(t) & =_{\text{def}} & v(t+n\partial). \end{array}$$

Equation (19) means that the derivatives x', y', x'', y'' are interpreted using the explicit first-order Euler scheme with an *infinitesimal time step* ∂. Note that (19) implies

$$x'' \quad = \quad \frac{x'^\bullet - x'}{\partial}. \tag{20}$$

After performing the substitutions given by (19), we observe that the subsystem collecting equations (k_0)–(k_4) is a logico-numerical fixpoint equation, with dependent variables $x^{\bullet 2}, y^{\bullet 2}, \lambda, \gamma$. A possible solution would consist in performing a relaxation, by iteratively updating the numerical variables based on the previous value for the guards, and then re-evaluating the guard based on the updated values of the numerical variables, hoping for a fixpoint to occur. Such fixpoint equation, however, can have zero, one, several, or infinitely many solutions. No characterization exists that could serve as a basis for a (graph-based) structural analysis. *We thus decide to refuse solving such mixed logico-numerical systems.*

As a consequence, we are unable to evaluate guard γ, so that the mode the system is in cannot be determined: model (18) is rejected.

To break the fixpoint equation defining γ, we choose to systematically introduce infinitesimal delays to guards. For the Cup-and-Ball, the predicate $s \leq 0$ then defines the value of the guard *at the next nonstandard instant*.[4] This yields the corrected model (21), where the modification is highlighted in red.

$$\left\{\begin{array}{rll} & 0 = x'' + \lambda x & (e_1) \\ & 0 = y'' + \lambda y + g & (e_2) \\ & \gamma^\bullet = [s \leq 0]; \gamma(0) = \text{F} & (k_0) \\ \texttt{if } \gamma \texttt{ do} & 0 = L^2 - (x^2+y^2) & (k_1) \\ \texttt{and} & 0 = \lambda + s & (k_2) \\ \texttt{if not } \gamma \texttt{ do} & 0 = \lambda & (k_3) \\ \texttt{and} & 0 = (L^2 - (x^2+y^2)) - s & (k_4) \end{array}\right. \tag{21}$$

[4]The condition triggering the mode change is based on the positions, which remain continuous at mode changes, even though the velocities are discontinuous. As a result, the shifting of this guard by an infinitesimal time step only yields an infinitesimal change in the values of state variables, which will be erased by the standardization process, so that the numerical solution is not impacted by this change in the model.

This model is understood in the nonstandard setting, meaning that the derivatives are expanded using (19). The leading variables in all modes are $\lambda, s, x^{\bullet 2}, y^{\bullet 2}$.

6.1 Structural analysis of mode change γ:F→T

Due to equation (k_1), the mode $\gamma = \text{T}$ (where the rope is straight) requires index reduction. We thus augment model (21) with the two latent equations shown in red:

$$\left\{\begin{array}{rll} & 0 = x'' + \lambda x & (e_1) \\ & 0 = y'' + \lambda y + g & (e_2) \\ & \gamma^\bullet = [s \leq 0]; \gamma(0) = \text{F} & (k_0) \\ \texttt{if } \gamma \texttt{ do} & 0 = L^2 - (x^2+y^2) & (k_1) \\ \texttt{and} & 0 = L^2 - (x^2+y^2)^\bullet & (k_1^\bullet) \\ \texttt{and} & 0 = L^2 - (x^2+y^2)^{\bullet 2} & (k_1^{\bullet 2}) \\ \texttt{and} & 0 = \lambda + s & (k_2) \\ \texttt{if not } \gamma \texttt{ do} & 0 = \lambda & (k_3) \\ \texttt{and} & 0 = (L^2 - (x^2+y^2)) - s & (k_4) \end{array}\right. \tag{22}$$

Note that, as in System 10, the two latent equations $(k_1^\bullet)$ and $(k_1^{\bullet 2})$ were obtained by *shifting* (k_1) *forward*, which is equivalent to differentiating it for the structural analysis. To perform structural analysis at the considered mode change, we unfold model (22) at the successive instants

$${}^{\bullet 2}t =_{\text{def}} t - 2\partial, \quad {}^\bullet t =_{\text{def}} t - \partial, \quad \text{and} \quad t,$$

where t denotes the current instant. In the following, equation (e_1) at the instant $t - 2\partial$ (respectively, $t - \partial$) will be denoted by $({}^{\bullet 2}e_1)$ (resp., $({}^\bullet e_1)$).

In this unfolding, the two equations (k_1) and $(k_1^\bullet)$ are in conflict with selected equations from the previous two instants, shown in blue in the following subsystem, whose dependent variables are the leading variables at instants $t - 2\partial$ and $t - \partial$, namely $x, y, {}^{\bullet 2}\lambda; x^\bullet, y^\bullet, {}^\bullet\lambda$:

$$\left\{\begin{array}{ll} 0 = \frac{x - 2{}^\bullet x + {}^{\bullet 2}x}{\partial^2} + {}^{\bullet 2}\lambda\, {}^{\bullet 2}x & ({}^{\bullet 2}e_1) \\ 0 = \frac{y - 2{}^\bullet y + {}^{\bullet 2}y}{\partial^2} + {}^{\bullet 2}\lambda\, {}^{\bullet 2}y + g & ({}^{\bullet 2}e_2) \\ 0 = \frac{x^\bullet - 2x + {}^\bullet x}{\partial^2} + {}^\bullet\lambda\, {}^\bullet x & ({}^\bullet e_1) \\ 0 = \frac{y^\bullet - 2y + {}^\bullet y}{\partial^2} + {}^\bullet\lambda\, {}^\bullet y + g & ({}^\bullet e_2) \\ 0 = L^2 - (x^2+y^2) & (k_1) \\ 0 = L^2 - (x^2+y^2)^\bullet & (k_1^\bullet) \end{array}\right.$$

We resolve this conflict by applying causality Principle 1, which leads to erasing, in model (22), equations (k_1) and $(k_1^\bullet)$ at the instant of mode change ${}^\bullet\gamma{=}\text{F}, \gamma{=}\text{T}$. This yields:

$$\text{at} \left[\begin{array}{l} {}^\bullet\gamma{=}\text{F} \\ \gamma{=}\text{T} \end{array}\right] : \left\{\begin{array}{ll} 0 = x'' + \lambda x & (e_1) \\ 0 = y'' + \lambda y + g & (e_2) \\ 0 = L^2 - (x^2+y^2)^{\bullet 2} & (k_1^{\bullet 2}) \\ 0 = \lambda + s & (k_2) \end{array}\right. \tag{23}$$

System (23) uniquely determines all the leading variables from the state variables x, y and $x^\bullet, y^\bullet$. In turn, equations (k_1) and $(k_1^\bullet)$, which were erased from this model, are not satisfied. At the next instant, i.e., when ${}^{\bullet 2}\gamma{=}\text{F}, {}^\bullet\gamma{=}\text{T}, \gamma{=}\text{T}$,

the same argument is used. We thus erase, in model (22), the only equation (k_1) at the next instant. This yields:

$$\text{at } \begin{bmatrix} {}^{\bullet 2}\gamma{=}\mathrm{F} \\ {}^{\bullet}\gamma{=}\mathrm{T} \\ \gamma{=}\mathrm{T} \end{bmatrix} : \begin{cases} 0 = x'' + \lambda x & (e_1) \\ 0 = y'' + \lambda y + g & (e_2) \\ 0 = L^2 - (x^2+y^2)^{\bullet} & (k_1^{\bullet}) \\ 0 = L^2 - (x^2+y^2)^{\bullet 2} & (k_1^{\bullet 2}) \\ 0 = \lambda + s & (k_2) \end{cases} \tag{24}$$

Note that $(k_1^{\bullet})$ is a consistency equation that is satisfied by the state variables $x^{\bullet}, y^{\bullet}$. In turn, equation (k_1), which was erased from this model, is not satisfied. At subsequent instants, equation erasure is no longer needed.

This completes the nonstandard structural analysis of the mode change $\gamma : \mathrm{F} \to \mathrm{T}$, i.e., when the rope gets straight.

6.2 Getting effective code for restart

Code generation for restarts consists in standardizing nonstandard systems (23) and (24), in a way similar to Section 5.4. We focus on the standardization of the mode change $\gamma : \mathrm{F} \to \mathrm{T}$, i.e., when the rope gets straight. Our task is to standardize systems (23) and (24), by targeting discrete-time dynamics, for the two successive instants composing the restart phase. This will provide us with restart values for positions and velocities.

Due to the expansion of derivatives in equations $(e_1, e_2, e_1^{\bullet}, e_2^{\bullet})$, tensions λ and $\lambda^{\bullet}$ are both impulsive, hence so are s and $s^{\bullet}$ by $(k_2, k_2^{\bullet})$. We eliminate the impulsive variables by ignoring $(k_2, k_2^{\bullet})$, combining (e_1) and (e_2) to eliminate λ, and $(e_1^{\bullet})$ and $(e_2^{\bullet})$ to eliminate $\lambda^{\bullet}$. This yields:

$$\text{at } \begin{bmatrix} {}^{\bullet}\gamma{=}\mathrm{F} \\ \gamma{=}\mathrm{T} \end{bmatrix} : \begin{cases} 0 = y''x + gx - x''y \\ 0 = L^2 - (x^2+y^2)^{\bullet 2} \end{cases} \tag{25}$$

$$\text{at } \begin{bmatrix} {}^{\bullet 2}\gamma{=}\mathrm{F} \\ {}^{\bullet}\gamma{=}\mathrm{T} \\ \gamma{=}\mathrm{T} \end{bmatrix} : \begin{cases} 0 = y''x + gx - x''y \\ 0 = L^2 - (x^2+y^2)^{\bullet} \\ 0 = L^2 - (x^2+y^2)^{\bullet 2} \end{cases} \tag{26}$$

In System (25) we expand second derivatives using (19), whereas in System (26) we expand them using (20). Consequently, System (25) has dependent variables $x^{\bullet 2}, y^{\bullet 2}$, whereas System (26) has dependent variables $x'^{\bullet}, y'^{\bullet}$. We are now ready to standardize the two systems.

System (25) **to define restart positions:** We expand second derivatives using (19):

$$\begin{cases} 0 = (y^{\bullet 2} - 2y^{\bullet} + y)x - (x^{\bullet 2} - 2x^{\bullet} + x)y + \partial^2 gx \\ 0 = L^2 - (x^2+y^2)^{\bullet 2} \end{cases} \tag{27}$$

Setting $\partial = 0$ in this system yields a structurally regular system. Hence, by a theorem proved in (Benveniste et al., 2020), the so obtained system is the correct standardization of System (27). In contrast, had we set $\partial = 0$ in System (23) (without eliminating impulsive variable λ), we would get a structurally singular system, an incorrect standardization.

In System (27) with $\partial = 0$, we can interpret x and $x^{\bullet}$ as the left-limit x^- of state variable x in previous mode, and $x^{\bullet 2}$ as the restart value x^+ for the new mode. This yields

$$\begin{cases} 0 = (y^+ - y^-)x^- - (x^+ - x^-)y^- \\ 0 = L^2 - (x^2+y^2)^+ \end{cases} \tag{28}$$

which determines the restart values for positions. The constraint that the rope is straight is satisfied. Furthermore, as $0 = L^2 - (x^2+y^2)^-$ also holds (the rope is straight at the mode change), $x^+ = x^-, y^+ = y^-$ is the unique solution of (28): positions are continuous.

System (26) **to define restart velocities:** We expand second derivatives using (20):

$$\begin{cases} 0 = (y'^{\bullet} - y')x - (x'^{\bullet} - x')y + \partial . gx \\ 0 = L^2 - (x^2+y^2)^{\bullet} \\ 0 = L^2 - (x^2+y^2)^{\bullet 2} \end{cases} \tag{29}$$

By expanding $x^{\bullet 2} = x^{\bullet} + \partial x'^{\bullet}$, the right-hand side of the last equation rewrites

$$\begin{aligned} L^2 - (x^2+y^2)^{\bullet 2} = \quad & L^2 - (x^2+y^2)^{\bullet} \\ & + \; 2\partial (x^{\bullet}x'^{\bullet} + y^{\bullet}y'^{\bullet}) \\ & + \; \partial^2 \big((x'^{\bullet})^2 + (y'^{\bullet})^2 \big) \\ = \quad & 0 \;\; \text{(using (29))} \\ & + \; 2\partial (x^{\bullet}x'^{\bullet} + y^{\bullet}y'^{\bullet}) + O(\partial^2) \end{aligned} \tag{30}$$

Using this expansion, setting $\partial = 0$ in (29) yields

$$\begin{cases} 0 = (y'^{\bullet} - y')x - (x'^{\bullet} - x')y \\ 0 = x^{\bullet}x'^{\bullet} + y^{\bullet}y'^{\bullet} \end{cases} \tag{31}$$

where the dependent variables are now $x'^{\bullet}, y'^{\bullet}$—other variables are state variables whose values were set at previous time steps. System (31) is structurally regular, hence, it is the correct standardization of System (29).

To get effective code for restart, we perform, in (31), the following substitutions, where superscripts $^-$ and $^+$ denote left- and right-limits, and continuity of positions is used:

$$x = x^- \; ; \; x^{\bullet} = x^+ \quad \text{and} \quad x' = x'^- \; ; \; x'^+ = x'^{\bullet} \tag{32}$$

and similarly for y. This finally yields

$$\begin{cases} 0 = (y'^+ - y'^-)x^- - (x'^+ - x'^-)y^- \\ 0 = x^+x'^+ + y^+y'^+ \end{cases} \tag{33}$$

System (33) determines x'^+ and y'^+, which are the velocities for restart. The second equation guarantees that the velocity will be tangent to the constraint. With (28) and (33), we determine the restart conditions for positions and velocities. Invariants from the physics are satisfied.

Our reasoning so far produces a behavior in which the two modes (free motion and straight rope) gently alternate; the system always stays in one mode for some positive period of time before switching to the other mode.

This indeed amounts to assuming that the impact is totally inelastic at mode change, an assumption that was not explicit at all in (21)**.** So, what happened? In fact, ***the straight rope mode was implicitly assumed to last for at least three nonstandard successive instants, since we allowed ourselves to shift*** (k_1) ***twice*****.**

6.3 Handling transient modes

Let us instead assume elastic impact, represented by the cascade of mode changes γ: F → T → F, reflecting that the straight rope mode is *transient* (it is left immediately after being reached).

Consider again model (21). We regard the instant of the cascade when $\gamma = \mathrm{T}$ occurs as the current instant. We cannot add latent equations by simply shifting (k_1), since these shifted versions are not active in the mode $\gamma = \mathrm{F}$. Set

$$S(\mathrm{T}) = \{(e_1), (e_2), (k_1), (k_2)\}$$
$$S(\mathrm{F}) = \{(e_1), (e_2), (k_3), (k_4)\}$$

Systems $S^{\bullet}(\mathrm{T})$ and $S^{\bullet}(\mathrm{F})$ are obtained by shifting once the equations constituting $S(\mathrm{T})$ and $S(\mathrm{F})$; systems $S^{\bullet k}(\mathrm{T})$ and $S^{\bullet k}(\mathrm{F})$ are defined similarly for all $k \in \mathbb{N}$. Consider the *differentiation array* originally proposed by (Campbell and Gear, 1995), except that we take into account the trajectory $\mathrm{T}, \mathrm{F}, \mathrm{F}, \ldots$ for guard γ. Using shifting instead of differentiation yields the following *difference array*:

$$\mathscr{A}_n(S) =_{\text{def}} \left[\begin{array}{ccccc} S(\mathrm{T}) & S^{\bullet}(\mathrm{F}) & S^{\bullet 2}(\mathrm{F}) & \ldots & S^{\bullet n}(\mathrm{F}) \end{array} \right]^T \quad (34)$$

The dependent variables of System $\mathscr{A}_n = 0$ are $x^{\bullet 2}, y^{\bullet 2}, \lambda$, whereas $x^{\bullet(k+2)}, y^{\bullet(k+2)}, \lambda^{\bullet(k)}, k > 0$ must be eliminated. We look for the smallest n such that $\mathscr{A}_n = 0$ is structurally nonsingular in this sense. Unfortunately, although shifting (k_4) twice in System (21) produces one more equation involving the leading variables $x^{\bullet 2}, y^{\bullet 2}$, this equation also involves the new variable $s^{\bullet 2}$, which keeps the augmented system underdetermined; shifting other equations fails as well. Therefore, the structural analysis rejects this model as being underdetermined at transient mode $\gamma = \mathrm{T}$.

The user is then asked to provide one more equation. For example, they could specify an impact law for the velocity y' by providing the equation $(y')^+ = -(1-\alpha)(y')^-$, where $0 \le \alpha < 1$ is a fixed damping coefficient. This is reinterpreted in the nonstandard domain as $y'^{\bullet} = -(1-\alpha)y'$, yielding the following refined system for use at mode $\gamma = \mathrm{T}$ within the cascade γ:F→T→F:

$$\left\{ \begin{array}{ll} 0 = x'' + \lambda x & (e_1) \\ 0 = y'' + \lambda y + g & (e_2) \\ 0 = y'^{\bullet} + (1-\alpha)y' & (\tau_1) \\ 0 = L^2 - (x^2+y^2) & (k_1) \\ 0 = \lambda + s & (k_2) \end{array} \right. \quad (35)$$

The modified difference array is now structurally nonsingular. The so modified model is accepted and two-step restart code for the mode change is generated as before.

6.4 Consequences for the modeling language

Through the Cup-and-Ball example, *we demonstrated the need for the following user-given information: is the current mode long or transient?* Long / Transient is an information regarding modes, that cannot be found by an automatic inspection of the model. It must be inferred from understanding the system physics and must be manually specified. The natural way of performing this is to provide a different syntax for specifying long modes on the one hand, and events corresponding to transient modes on the other hand (mode changes separating two successive long modes need not be specified).

The '`if`' and '`when`' statements of the Modelica language are fit candidates for this purpose. We devote the '`if`' statement to long-lasting modes specified by a predicate, while the '`when`' statement, pointing to the event when a predicate switches from F to T, could be further restricted to be a zero-crossing condition, by which a $\mathbb{R}$-valued expression crosses zero from below (Bourke and Pouzet, 2013). Using this feature, the Cup-and-Ball example with elastic impact is specified as follows:

$$\left\{ \begin{array}{rll} & 0 = x'' + \lambda x & (e_1) \\ & 0 = y'' + \lambda y + g & (e_2) \\ & \gamma = [s^- \le 0]; \gamma(0) = \mathrm{F} & (k_0) \\ \texttt{when}\ \gamma\ \texttt{do} & y'^+ = -\alpha y'^- & (\tau_1) \\ \texttt{if not}\ \gamma\ \texttt{do} & 0 = \lambda & (k_3) \\ \texttt{and} & 0 = (L^2 - (x^2+y^2)) - s & (k_4) \end{array} \right.$$

7 Mechanization of the process

The approach developed in Sections 5.4 and 6.2 is a systematic way to define the solution of a multimode DAE system, than can be generalized to large-scale and/or multi-physics models. However, this reasoning:

- *Requires identifying impulsive variables.* We present in companion paper (Benveniste et al., 2021) a calculus for this, which is ready for automatization in a tool (this is under development in our IsamDAE tool[5]).
- *Requires eliminating impulsive variables.* This is easy if impulsive variables enter linearly in the model—this was the case for the clutch and the cup-and-ball examples. It is highly costly but still doable if impulsive variables enter polynomially in the model. It cannot be done practically in other cases.
- *Relies on a clever choice of how to map nonstandard variables to restart conditions.* This was straightforward for the clutch, but definitely not for the cup-and-ball (Section 6.2), where expansion (19) for the derivatives was used for resetting positions, whereas expansion (20) was used for resetting velocities.

This is not realistic for implementation in a tool, except for restricted classes of systems in which impulsive variables enter linearly in the model.

However, in companion paper (Benveniste et al., 2021) we propose an alternative, which is a good candidate for implementation, based on an *impulse analysis*. This post-processing of the structural analysis at mode changes is a simple and systematic calculus that identifies impulsive

[5] https://allgo18.inria.fr/apps/isamdae

variables at compile time and quantifies their (possibly infinite) magnitude order, called *impulse order*. When finite, the impulse order can be used to rescale impulsive variables, which allows for computing restart values for state variables as well as rescaled impulsive variables. When impulse orders are infinite, rescaling no longer applies. It is, however, still possible to compute restart conditions by using the nonstandard equations with a small positive (standard) time step. This provides converging approximations for the non-impulsive variables (the state variables in particular).

8 Conclusion

Through the case study of two examples of multimode DAEs that are currently not handled by the existing Modelica tools (with the notable exception of ModiaMath), we presented a mathematically sound and physics-agnostic compilation process for DAE-based physical systems modeling languages. This method relies, in particular, on an extension of structural analysis to multimode systems, that allows the handling of both modes and mode changes in a unified framework.

Both examples studied in this paper are multimode models with mode-dependent differentiation index and impulsive behaviour at mode changes, which is not well supported by existing Modelica compilers. This paper showed how our approach handles such models: not only is the structural analysis correctly performed in all continuous modes, but the computation of restart values at mode changes is also handled at compile time, unless an under/over-determination at a mode change event causes the model to be rejected with proper diagnostics.

Ongoing works include the effective mechanization of the process, which is detailed in the companion paper (Benveniste et al., 2021). An important bottleneck of this approach is that it needs to handle all modes and all possible mode changes at compile time: unfortunately, the number of modes tends to be roughly exponential in the size of the model, and the *a priori* number of mode changes is at least proportional to the square of the number of modes. This is a limitation of a model representation in which one characterizes the subset of equations and variables active in any given mode.

A possible way of alleviating this issue is by shifting to a dual representation, that provides predicates characterizing the set of modes in which each equation and each variable is active. In practice, not only does this approach lead to a much more compact representation, but it also allows for the design of efficient structural analysis methods for multimode DAE systems, working in an 'all-modes-at-once' fashion. Such a method was implemented in the IsamDAE tool, and first results are reported in (Caillaud et al., 2020). The examples coming with this tool already include thermodynamical, electrical and pneumatic models. Although only the structural analysis of long modes is currently performed, the implementation of the structural analysis of mode changes is in progress.

Acknowledgements

The authors are indebted to Hilding Elmqvist and Martin Otter, John Pryce, and Vincent Acary. Khalil Ghorbal participated to the first version of this approach.

This work was supported by the FUI ModeliScale DOS0066450/00 French national grant (2018-2021) and the Inria IPL ModeliScale large scale initiative (2017-2021, https://team.inria.fr/modeliscale/). Dymola licences were provided to the authors by Dassault Systèmes in the context of the FUI ModeliScale project.

References

Albert Benveniste, Benoît Caillaud, Hilding Elmqvist, Khalil Ghorbal, Martin Otter, and Marc Pouzet. Structural analysis of multi-mode DAE systems. In *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*, pages 253–263, 04 2017. ISBN 978-1-4503-4590-3. doi:10.1145/3049797.3049806.

Albert Benveniste, Benoît Caillaud, Hilding Elmqvist, Khalil Ghorbal, Martin Otter, and Marc Pouzet. Multi-mode DAE models - challenges, theory and implementation. In Bernhard Steffen and Gerhard J. Woeginger, editors, *Computing and Software Science - State of the Art and Perspectives*, volume 10000 of *Lecture Notes in Computer Science*, pages 283–310. Springer, 2019. ISBN 978-3-319-91907-2. doi:10.1007/978-3-319-91908-9_16.

Albert Benveniste, Benoít Caillaud, and Mathias Malandain. The mathematical foundations of physical systems modeling languages. *Annual Reviews in Control*, 50:72–118, 2020. ISSN 1367-5788. doi:10.1016/j.arcontrol.2020.08.001.

Albert Benveniste, Benoît Caillaud, and Mathias Malandain. Compile Time Impulse Analysis in Modelica. In *Proceedings of the 14th International Modelica Conference*. Linköping University Electronic Press, September 2021.

Timothy Bourke and Marc Pouzet. Zélus: A synchronous language with ODEs. In *Hybrid Systems: Computation and Control (HSCC)*, pages 113–118, Philadelphia, USA, April 2013. ACM.

Benoît Caillaud, Mathias Malandain, and Joan Thibault. Implicit structural analysis of multimode DAE systems. In *23rd ACM International Conference on Hybrid Systems: Computation and Control (HSCC 2020)*, Sydney, Australia, April 2020. doi:10.1145/3365365.3382201.

Stephen L. Campbell and C. William Gear. The index of general nonlinear DAEs. *Numer. Math.*, 72:173–196, 1995.

Dassault Systèmes AB. Dymola official webpage. URL https://www.3ds.com/products-services/catia/products/dymola/. Accessed: 2021-06-28.

Andrew L. Dulmage and Nathan S. Mendelsohn. Coverings of bipartite graphs. *Canadian Journal of Mathematics*, 10: 517–534, 1958. doi:10.4153/CJM-1958-052-0.

Hilding Elmqvist, Fabien Gaucher, Sven Erik Mattsson, and Dupont Fran cois. State machines in Modelica. In Martin Otter and Dirk Zimmer, editors, *Proc. of the Int. Modelica Conference*, pages 37–46, Munich, Germany, September 2012. Modelica Association.

Hilding Elmqvist, Sven Erik Mattsson, and Martin Otter. Modelica extensions for multi-mode DAE systems. In Hubertus Tummescheit and Karl-Erik Arzen, editors, *Proc. of the 10th Int. Modelica Conference*, Lund, Sweden, September 2014. Modelica Association.

Peter Fritzson, Adrian Pop, Karim Abdelhak, Adeel Ashgar, Bernhard Bachmann, Willi Braun, Daniel Bouskela, Robert Braun, Lena Buffoni, Francesco Casella, Rodrigo Castro, Rüdiger Franke, Dag Fritzson, Mahder Gebremedhin, Andreas Heuermann, Bernt Lie, Alachew Mengist, Lars Mikelsons, Kannan Moudgalya, Lennart Ochel, Arunkumar Palanisamy, Vitalij Ruge, Wladimir Schamai, Martin Sjölund, Bernhard Thiele, John Tinnerholm, and Per Östlund. The OpenModelica Integrated Environment for Modeling, Simulation, and Model-Based Development. *Modeling, Identification and Control*, 41 (4):241–295, 2020. doi:10.4173/mic.2020.4.1.

Christoph Höger. Dynamic structural analysis for DAEs. In *Proceedings of the 2014 Summer Simulation Multiconference, SummerSim 2014, Monterey, CA, USA, July 6-10, 2014*, page 12. SCS/ ACM, 2014.

Christoph Höger. Elaborate control: variable-structure modeling from an operational perspective. In Dirk Zimmer and Bernhard Bachmann, editors, *Proceedings of the 8th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools, EOOLT '17, Weßling, Germany, December 1, 2017*, pages 51–60. ACM, 2017. ISBN 978-1-4503-6373-0. doi:10.1145/3158191.3158198.

Daniel Liberzon and Stephan Trenn. Switched nonlinear differential algebraic equations: Solution theory, lyapunov functions, and stability. *Automatica*, 48(5):954–963, 2012. doi:10.1016/j.automatica.2012.02.041.

Tom Lindstrøm. An invitation to nonstandard analysis. In N.J. Cutland, editor, *Nonstandard Analysis and its Applications*, pages 1–105. Cambridge Univ. Press, 1988.

Sven Erik Mattsson and Gustaf Söderlind. Index reduction in differential-algebraic equations using dummy derivatives. *SIAM Journal on Scientific Computing*, 14(3):677–692, 1993. doi:10.1137/0914043.

Sven Erik Mattsson, Martin Otter, and Hilding Elmqvist. Modelica Hybrid Modeling and Efficient Simulation. In IEEE, editor, *38th IEEE Conference on Decision and Control*, pages 3502–3507, 1999.

Sven Erik Mattsson, Martin Otter, and Hilding Elmqvist. Multi-Mode DAE Systems with Varying Index. In Hilding Elmqvist and Peter Fritzson, editors, *Proc. of the 11th Int. Modelica Conference*, Versailles, France, September 2015. Modelica Association.

Henrik Nilsson and George Giorgidze. Exploiting structural dynamism in Functional Hybrid Modelling for simulation of ideal diodes. In *Czech Technical University Publishing House*, 2010.

Constantinos C. Pantelides. The consistent initialization of differential-algebraic systems. *SIAM J. Sci. Stat. Comput.*, 9(2):213–231, 1988.

Peter Pepper, Alexandra Mehlhase, Christoph Höger, and Lena Scholz. A compositional semantics for modelica-style variable-structure modeling. In *th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools*, 2011.

John D. Pryce. A simple structural analysis method for DAEs. *BIT*, 41(2):364–394, 2001.

Abraham Robinson. *Nonstandard Analysis*. Princeton Landmarks in Mathematics, 1996. ISBN 0-691-04490-2.

Stephan Trenn. Regularity of distributional differential algebraic equations. *MCSS*, 21(3):229–264, 2009a. doi:10.1007/s00498-009-0045-4.

Stephan Trenn. *Distributional Differential Algebraic Equations*. PhD thesis, Technischen Universität Ilmenau, 2009b.