
S.E. Mattsson, H. Olsson, H. Elmqvist:
**Dynamic Selection of States in Dymola.**
Modelica Workshop 2000 Proceedings, pp. 61-67.

Paper presented at the Modelica Workshop 2000, Oct. 23.-24., 2000, Lund, Sweden.

All papers of this workshop can be downloaded from
*http://www.Modelica.org/modelica2000/proceedings.html*

# Dynamic Selection of States in Dymola

**Sven Erik Mattsson, Hans Olsson and Hilding Elmqvist**
Dynasim AB, Lund, Sweden

## ABSTRACT

The object-oriented modelling language Modelica supports differential-algebraic equations (DAE) to describe physical phenomena and behaviour. A basic idea is to support flexible and safe reuse of model components. Model components are made general. When connecting them to form a system model their behaviours are constrained. It typically means that there are more variables appearing differentiated than variables needed for the continuous-time state. DAE problems of this kind are said to have a high DAE index and the solution procedure involves differentiation. There are no reliable general-purpose numerical DAE solvers for high-index problems. The paper illustrates how Dymola solves them in a reliable and efficient way by combining symbolic and numerical methods.

## Introduction

The object-oriented modeling language Modelica [5, 3] is designed for modeling of large and heterogeneous physical systems. General equations are used for modeling of the physical phenomena. Differential-algebraic equations (DAE) support modelling of continuous-time behaviour.

Modelica has been designed to allow tools to generate efficient code automatically. No particular variable needs to be solved manually. A Modelica tool will have enough information to do that automatically. The modelling effort is reduced considerably since model components can be reused. Tedious and error-prone manual manipulations are avoided.

Consider a DAE system

$$F(t, x, \dot{x}, y) = 0\,,$$

where $t$ is time, $x$ and $y$ are vectors of unknown variables. The elements of $x$ are called dynamic variables since their time derivatives, $\dot{x}$, appear in the equations and the elements of $y$ are called algebraic variables since none of its derivatives appear in the equations.

Solving a DAE problem involves more than solving algebraic loops for $y$ and $\dot{x}$ and integrating to obtain $x$. The solution procedure involves also differentiation if the Jacobian with respect to $\dot{x}$ and $y$ is structurally singular. It means that there are algebraic relations between the elements of $x$. In order to be able to solve for $y$ and $\dot{x}$ it is then necessary to differentiate some equations. The state to integrate is only a subset of the elements of $x$ and the number of initial values to set is not always equal to the number of dynamic variables. The first word *dynamic* in the title of this paper indicates that there are problems where there is no fixed set of state variables that works everywhere along the solution trajectory, i.e., the set of state variables change dynamically.

A DAE problem for which there is a need for differentiation in the solution procedure is said to have a high DAE index. It is well known that there are no reliable general-purpose numerical DAE solvers for high-index problems.

This paper illustrates how the Dynamic Modeling Laboratory, Dymola [2] solves high-index problems by combining symbolic and numeric methods.

## Background and Motivation

Before discussing how to solve the problems some more thorough explanations and motivations will be given to the statements of the introduction.

### High index DAE problems are natural

As a simple example consider Newton's second law

$$m\ddot{x} = F$$

It is a model for the motion, $x$, of a particle having constant mass, $m$, being influenced by a force $F$.

Newton's second law can be used to solve the dynamics problem, i.e., to calculate the motion when $m = m_0$ is a given constant and $F = F_R(t, x, \dot{x})$ is given as a function of time, speed and position. The problem is solved by integrating the force, $F_R$, twice. Position and velocity can be taken as states.

In, for example, robotics, it is of interest to calculate the force needed to have the body to follow a prescribed trajectory, $x = x_R(t)$. This problem has different mathematical characteristics. To calculate the force, $F$, the trajectory, $x_R$, needs to be differentiated twice. The model contains dynamic variables, but there are no continuous-time states

and thus no initial values to set. It is a high-index DAE problem.

The two problems discussed above are in some sense extremes. There are also mixed problems. To support reuse it is natural to develop models of bodies moving freely in 3 dimensions. When such components are used, connections to other components constrain their motions. A connection implies that two positions are equal, which means that there are reaction forces. To calculate the reaction forces, it is necessary to differentiate the position constraint twice to get a constraint in terms of accelerations from which the reaction forces can be calculated. Moreover, to support reuse, it should of course also be possible to use the 3D components to model planar mechanics. Then there is a constraint $x_3 = b$ or more generally $kx = b$, with $k$ being a constant vector of size three and $b$ a scalar constant.

Consequently, high index DAE problems are natural in object-oriented modelling, because the idea is to support reuse of model components.

### The DAE index

Consider the DAE problem

$$F(t, x, \dot{x}) = 0$$

We assume that it is *solvable* [1], with a unique, smooth solution when supplied with an appropriate number of consistent initial solutions. The *index* [1] of the problem equals the minimum number times that all or part of it must be differentiated with respect to time $t$ in order to determine $\dot{x}$ as a continuous function of $x$ and $t$.

For example, an ODE on explicit state space form,

$$\dot{x} = f(t, x),$$

is index 0, and the problem

$$\dot{x} = f(t, x, y)$$
$$0 = g(t, x, y)$$

is index 1 if the Jacobian of g with respect to y, $\partial g / \partial y$, is non-singular and it is only necessary to differentiate if calculation of $\dot{y}$ is wanted.

By default Dymola assumes that a user is interested in calculating all variables and appearing derivatives. It is thus more convenient to allow high order derivatives as well as purely algebraic variables. There is then no need for differentiation, if it is possible to determine the highest order derivatives as continuous functions of time and lower derivatives. The index is at most 1. The index is zero if there are no algebraic variables. To avoid this technical problem, we will just say that a problem has high index if it is necessary to differentiate when solving for the highest-order derivatives.

### Why do numerical DAE solvers fail?

When solving an ODE the solution procedure involves only integration, i.e., to calculate $x$ from $\dot{x}$. The solution of a DAE system may also involve differentiation, i.e., to calculate $\dot{x}$ from $x$. The error in the differentiation will increase with decreasing step-sizes, due to round-off errors and inaccurate solutions of algebraic equations in previous steps. This will introduce large unphysical transients if e.g. some other part of the system cause a drastic reduction in the step size. Furthermore it will invalidate the assumptions of the error control and a mild increase in the error might a stop the simulation since it is not possible to reduce the step size to get an acceptable error estimate. Given the index of variables, as provided by the index-reduction procedure, some solvers try to patch the error control by ignoring errors in high variables or scaling their errors with a suitable power of the step size. An additional problem is that all solvers have restrictions on how high the index of the DAE can be without reducing the order of the method.

## Dummy Derivatives

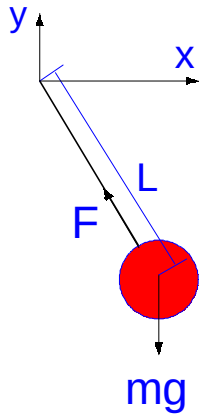



**Figure** 1: A planar pendulum.

As an introduction to basic ideas for reducing the DAE index and selecting state variables, consider a model in Cartesian coordinates for a planar pendulum of length $L$ and mass $m$. Newton's second law gives

$$m\ddot{x} = -\frac{x}{L}F \qquad (1a)$$

$$m\ddot{y} = -\frac{y}{L}F - mg \qquad (1b)$$

where $F$ is the force in the rod and $g$ is the gravitational acceleration. The length constraint is

$$x^2 + y^2 = L^2 \qquad (1c)$$

It is easy to see that it is not possible to use these three equations to solve for the highest order

derivatives $\ddot{x}$, $\ddot{y}$ and $F$, since the length constraint does not contain any of them. But they will appear if the length constraint is differentiated twice

$$x\dot{x} + y\dot{y} = 0 \qquad (1c')$$

$$x\ddot{x} + \dot{x}^2 + y\ddot{y} + \dot{y}^2 = 0 \qquad (1c'')$$

Equations (1a), (1b) and (1c'') constitute an index 1 problem, since it is possible to solve for the highest order derivatives $\ddot{x}$, $\ddot{y}$ and $F$. Thus the original problem (1a), (1b) and (1c) is index 3. The initial values of $x$, $\dot{x}$, $y$ and $\dot{y}$ cannot be chosen independently. They must fulfil (1c) and (1c').

A model including Equations (1a), (1b) and (1c'') and start values, which fulfil Equations (1c) and (1c'), is mathematically correct. However, the algebraic relations of the original DAE problem are now only implicit. Unless linear, these are generally not preserved under discretization. As a result the numerical solution drifts off the algebraic constraints. For the pendulum, it means that its length will not be constant.

To avoid such difficulties, one may try to obtain a low-index formulation, with a solution set identical to that of the original problem. Let us illustrate the idea on the pendulum problem. For small oscillations around the equilibrium point $x = 0$ and $y = -L$, it is possible use to (1c'') to eliminate $\ddot{y}$. You can eliminate $\ddot{y}$ by first solving $\ddot{y}$ from (1c'') to get an expression, which is then used to eliminate $\ddot{y}$ by substitution. However, since it is not possible to solve general non-linear equations analytically, this is not a general method. Since Dymola anyhow must handle and solve non-linear equations, $\ddot{y}$ can instead be eliminated by adding (1c'') as an implicit definition of $\ddot{y}$. To avoid an overdetermined system, a new algebraic variable, say $y_{dd}$, is introduced to represent $\ddot{y}$ wherever it appears in Equations (1). Similarly, use (1c') to replace $\dot{y}$ by $y_d$. This yields an augmented but determined system, which is index 1. It is mathematically equivalent to (1a), (1b) and (1c):

$$m\ddot{x} = -\frac{x}{L}F \qquad (2a)$$

$$my_{dd} = -\frac{y}{L}F - mg \qquad (2b)$$

$$x^2 + y^2 = L^2 \qquad (2c)$$

$$x\dot{x} + yy_d = 0 \qquad (2c')$$

$$x\ddot{x} + \dot{x}^2 + yy_{dd} + y_d{}^2 = 0 \qquad (2c'')$$

Problem (2) has five equations and five unknowns: $x$, $y$, $y_d$, $y_{dd}$ and $F$, which all are algebraic except for which appears differentiated twice; the problem has a single degree of freedom.

The original problem (1) has been augmented with (1c') and (1c''). For each differentiated equation appended to the original problem, one "new" dependent variable was introduced. The introduced variables represent derivatives and are called *dummy derivatives*. Dummy derivatives are purely algebraic variables and not subject to discretization. We know that $y_{dd} \equiv \ddot{y}$ and $y_d \equiv \dot{y}$, but this is not explicit in the transformed problem.

The selection of dummy derivatives was done above for small oscillations. Consider (2c'). It is linear in $y_d$ with the coefficient $y$. It means that the problem (2) becomes singular when $y = 0$. There are similar problems with (2c''). When y is small then instead dummy derivatives have to be introduced for $\ddot{x}$ and $\dot{x}$. Thus there is a need for dynamic selection of dummy derivatives or states.

A dynamic variable, *z*, is a potential candidate for being a state. When its derivative $\dot{z}$ is selected to be a dummy derivative, you may also say that *z* is deselected as state.

Recall Equation (1c), $x^2 + y^2 = L^2$. For a given $x$ such that $|x| \le |L|$, (1c) may seem to give two solutions $y = \pm\sqrt{L^2 - x^2}$. However, Equation (1b) implicitly assumes that $y$ is differentiable at least twice. It means that the solution for y cannot jump from a down position to an up position. Neither can $\dot{y}$ jump from ascending motion to a descending motion. Fortunately, also numerical solvers have the property to give a solution that is close to previous value. However, when the solution for $y$ crosses 0 things become complicated. The numerical solver may step over zero and give an acceptable solution. It may also happen that it gets stuck at zero and signals a singular problem. The solver may be able to produce the first period, while getting stuck in the second period. Small changes in error tolerance settings may give drastic changes in whether and when the solver gets stuck. The probability of getting stuck increases with tightened error tolerance.

## The Index Reduction Algorithm

The index reduction procedure consists of two major steps. First, the differentiated index 1 problem is derived and then it is used for selection of dummy derivatives.

To find the differentiated index 1 problem, the algorithm developed by Pantelides [7] is used. It establishes the minimum number each equation has to be differentiated to make the differentiated problem structurally non-singular with respect to

highest-order derivatives. The algorithm can be viewed as an extension of the algorithms for assigning equations to variables in sorting procedures.

The idea when selecting dummy derivatives is to start from the differentiated index 1 problem and work backwards in the chain of differentiated equations and variables and for each step find variables that can be selected as dummy derivatives. The dummy derivative method is formally described in [4]. It will be illustrated in next section. An outline of the procedure is given below. It can be skipped at a first reading.

1. Let the equations of the differentiated index 1 problem be the current equations.
2. Let the highest derivatives of the differentiated index one problem be the current candidates.
3. Consider all of the current equations that are differentiated versions of the original ones. Collect their predecessors and let them be the current equations.
4. If there is no current equation, go to Step 8.
5. Consider all current unknowns that are at least of order one. Collect their predecessors of one order less and let them be the current candidates for elimination.
6. Exploit the current equations for elimination of candidates.
7. Repeat from Step 3.
8. Collect all original and differentiated equations. In all equation introduce a unique dummy derivative for each derivative selected in Step 6.

## Application: The pendulum

Let us discuss in more detail the application of the index reduction method to the pendulum model.

First, the model in pure Cartesian coordinates will be discussed and then an angular coordinate will be introduced.

### Pure Cartesian coordinates

The model (1) for the pendulum includes second order derivatives. Modelica supports only first order derivatives. The equations are easily rewritten by introducing $v_x$ and $v_y$ for the velocity components to give the equations

$$m\dot{v}_x = -\frac{x}{L}F \tag{3a}$$

$$m\dot{v}_y = -\frac{y}{L}F - mg \tag{3b}$$

$$x^2 + y^2 = L^2 \tag{3c}$$

$$\dot{x} = v_x \tag{3d}$$

$$\dot{y} = v_y \tag{3e}$$

which directly can be used in a Modelica model.

Consider state selection for model (3). Pantelides's algorithm first differentiates (3c)

$$x\dot{x} + y\dot{y} = 0 \tag{3c'}$$

Then it finds out that (3c') needs to be differentiated as well as (3d) and (3e) to obtain the differentiated index 1 problem as

$$m\dot{v}_x = -\frac{x}{L}F \tag{3a}$$

$$m\dot{v}_y = -\frac{y}{L}F - mg \tag{3b}$$

$$x\ddot{x} + \dot{x}^2 + y\ddot{y} + \dot{y}^2 = 0 \tag{3c''}$$

$$\ddot{x} = \dot{v}_x \tag{3d'}$$

$$\ddot{y} = \dot{v}_y \tag{3e'}$$

The highest order derivatives are $\ddot{x}$, $\ddot{y}$, $\dot{v}_x$, $\dot{v}_y$ and $F$. To obtain the differentiated index 1 problem, (3c) was differentiated twice giving (3c'') and Equations (3d) and (3e) were differentiated once giving (3d') and (3e'). A solution must fulfil (3c'), (3d) and (3e). They can be used to eliminate variables.

Let us apply the dummy derivative procedure outlined in previous section to the pendulum model:

1. The first set of current equations is (3a), (3b), (3c''), (3d') and (3e').
2. The first set of current candidates are $\ddot{x}$, $\ddot{y}$, $\dot{v}_x$, $\dot{v}_y$ and $F$.
3. In the set of current equations the equations (3c''), (3d') and (3e') are differentiated versions of the original ones. Their predecessors are (3c'), (3d) and (3e). Let them be the new current set of equations.
4. Go on to Step 5, because there are current equations.
5. In the set of current candidates the following are derivatives: $\ddot{x}$, $\ddot{y}$, $\dot{v}_x$ and $\dot{v}_y$. Collecting their predecessors of one order less is giving: $\dot{x}$, $\dot{y}$, $v_x$ and $v_y$. Let them be the new set of current candidates for elimination.
6. When exploiting the current equations for elimination, Dymola first tries to eliminate derivatives. Equation (3d), $\dot{x} = v_x$, can always be used to calculate $\dot{x}$ which means that $\ddot{x}$ is selected as a dummy derivative and thus $\dot{x}$ is

deselected as state. Similarly (3e), $\dot{y} = v_y$, is used to select $\ddot{y}$ as a dummy derivative. Equation (3c'), $x\dot{x} + y\dot{y} = 0$, has to be used dynamically to deselect either $v_x$ or $v_y$ as a state.

7. Repeat from Step 3.
8. Step 3 gives the current equations (3c)
9. Step 4 implies continue.
10. Step 5 gives the new candidates: *x* and *y*.
11. Step 6: Equation (3c) has to be used dynamically to deselect either *x* or *y* as a state.
12. Step 7 and Step 3-4 lead to Step 8 and collecting the resulting index 1 problem. In the resulting problem, $\ddot{x}$ and $\ddot{y}$ are selected as dummy derivatives. Equation (3c'), $x\dot{x} + y\dot{y} = 0$, is used dynamically to deselect either $v_x$ or $v_y$ as a state and (3c) is used dynamically to deselect either *x* or *y* as a state.

Unfortunately, the procedure discussed above introduces some extra derivatives, namely $\ddot{x}$ and $\ddot{y}$. Dymola avoids this by identifying simple differential equations such as (3d) and (3e). You may view them as introducing alias names for $\dot{x}$ and $\dot{y}$. The chain of derivatives, $x \to \dot{x} \to \ddot{x}$, is viewed as, $x \to v_x \to \dot{v}_x$.

Dymola proceeds as follows, when (3d) and (3e) have been identified as simple differential equations they are temporarily removed from the problem and all appearances of $\dot{x}$ and $\dot{y}$ in the remaining equations are substituted. Pantelides's algorithm is then applied on the Equations (3a), (3b) and (3c) to find the differentiated index 1 system for the unknowns $v_x$, $v_y$ and $F$. When differentiating (3c'), Dymola substitutes $\dot{x}$ and $\dot{y}$ by $v_x$ and $v_y$ giving

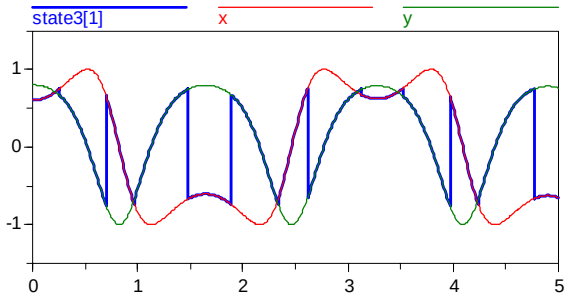
$$xv_x + yv_y = 0 \qquad (3c')$$

In the next step Pantelides's algorithm only calls for differentiation of (5c') giving

$$x\dot{v}_x + v_x^2 + y\dot{v}_y + v_y^2 = 0 \qquad (3c'')$$

The highest order derivatives are $\dot{v}_x$, $\dot{v}_y$ and $F$. Going back one step in the differentiation chain gives Equation (3c'). The candidates for elimination are $v_x$ and $v_y$. Going back one step further gives Equation (3c) and the candidates for elimination are *x* and *y*. The result is similar to Equation (1), but the variables are named a bit differently. Finally, Equation (3d) and (3e) are put back.

The plot below shows the result of a simulation in Dymola. It illustrates the use of Equation (3c) to select either *x* or *y* as state. The thick line shows the value of the state. Initially *x* = 0.6 and *y* = 0.8. Thus *y* > *x* and (3c) is used to eliminate *y* and *x* is then the state. At around *t*=0.2, *x* becomes greater than *y* and *y* is selected as state.




## Introducing an angular quantity

Unfortunately, the pendulum model in Cartesian coordinates requires dynamic selection of states, because the model has no variable that can function as a state for all positions. It is well known that a polar angle works fine as state at any position. Let us introduce such an angle, $\varphi$, that is zero, when the pendulum is hanging downward in its rest position; $x = L\sin(\varphi)$ and $y = -L\cos(\varphi)$. Introduce also $\omega$ for the angular velocity and $\alpha$ for the acceleration.

It is convenient to write Newton's equation of motion in Cartesian coordinates. However, it may be clearer to use $\varphi$ for the reaction forces from the rod and write $m\dot{v}_x = -F\sin(\varphi)$ instead of $m\dot{v}_x = -\frac{x}{L}F$.

The model can be written as

$$m\dot{v}_x = -F\sin(\varphi) \qquad (4a)$$

$$m\dot{v}_y = F\cos(\varphi) - mg \qquad (4b)$$

$$x = L\sin(\varphi) \qquad (4c)$$

$$y = -L\cos(\varphi) \qquad (4d)$$

$$\alpha = \dot{\omega} \qquad (4e)$$

$$\dot{x} = v_x \qquad (4f)$$

$$\dot{y} = v_y \qquad (4g)$$

$$\dot{\varphi} = \omega \qquad (4h)$$

To make $\omega$ eligible for state selection, it must be a dynamic variable. Equation (4e) makes $\omega$ dynamic. Dymola exploits (4e) at an early phase of the translation to eliminate $\alpha$ as an alias variable for $\dot{\omega}$.

Equation (4f), (4g) and (4h) are identified as simple differential equations just introducing alias names for the derivatives $\dot{x}$, $\dot{y}$ and $\dot{\varphi}$. Pantelides's algorithm is applied to Equations (4a-4d) with the

unknowns $v_x$, $v_y$, $\omega$ and $F$. First, Equation (4c) and (4d) are differentiated

$$v_x = L\omega\cos(\varphi) \qquad (4c')$$

$$v_y = L\omega\sin(\varphi) \qquad (4d')$$

Then they are differentiated once more

$$\dot{v}_x = L\dot{\omega}\cos(\varphi) - L\omega\sin(\varphi) \qquad (4c'')$$

$$\dot{v}_y = L\dot{\omega}\sin(\varphi) + L\omega\cos(\varphi) \qquad (4d'')$$

Equation (4c') and (4d') are used to eliminate $v_x$ and $v_y$, and (4c) and (4d) are used to eliminate *x* and *y*. Thus Dymola makes a fixed state selection with $\varphi$ and $\omega$ as states.

## Application: Quarternions

Consider modelling of the motion of a free, rigid body in a 3D space. It is a major issue to describe orientation in a 3D space. Unfortunately, there is no set of three variables to describe orientation that works for all positions.

Historically, the most popular approach to describing orientation has been in terms of Euler angles, where a general rotation is described as a sequence of rotations about three mutually orthogonal coordinate axes fixed in space. However, for a given set of Euler angles there are rotations that cannot be described by [8]. The quaternion, which was invented by the great mathematician Sir William Hamilton in 1843, is a solution to the problem.

A quaternion, $q$, is a vector of four elements. A 3-dimensional rotation can be described as

$$q = \begin{bmatrix} n \cdot \sin(\varphi/2) \\ \cos(\varphi/2) \end{bmatrix}$$

where *n* is the unique axis of rotation to transform a coordinate system 1 into a coordinate system 2 by a planar rotation via a rotation angle $\varphi$. The four elements are not independent from each other, but have to fulfil the normalization equation

$$q_1^{\,2} + q_2^{\,2} + q_3^{\,2} + q_4^{\,2} = 1 \qquad (5)$$

In coordinate system 2, the angular velocity, $\omega$ is

$$\omega = 2L\dot{q}$$

with

$$L = \begin{bmatrix} q_4 & q_3 & -q_2 & -q_1 \\ -q_3 & q_4 & q_1 & -q_2 \\ q_2 & -q_1 & q_4 & -q_3 \end{bmatrix}$$

The complete model of a body includes also the usual force and torque balances. For more information of use of quaternions in mechanics, see any good textbook in mechanics, e.g., [6].

When simulating a free body, one of the four components of $q$ has to be deselected as state. Dymola finds that Equation (5) have to be differentiated and that it is necessary to have a dynamic selection of states where three of the four elements of $q$ are selected as state and the remaining element is calculated from (5).

The structure of Equation (5) is similar to the length constraint of the pendulum. Dymola treats it in a similar way as when the Cartesian coordinates *x* and *y* were used. However, for the free body there is no need for dynamic selection of states at velocity level. The variable $\omega$ can be used as a state all the time. It was possible to eliminate the need of dynamic selection of states for the pendulum by introducing an angular coordinate. Unfortunately, this is not possible when modelling motion of bodies in a 3D space. There is no set of three variables to describe orientation that works for all positions. This is one reason why multbody mechanics in 3D is much more difficult than planar 2D mechanics.

## Application: Kinematic Loops

The Dymola distribution includes a demonstration example "Two coupled kinematic loops". It is a mechanical mechanism with one degree of freedom. Consider the model diagram in Figure 2 and a 3D-view of it in Figure 3.

The component b5 models the longest rod and to that is b4 welded orthogonally. The prismatic joint j3 allows the body b3 to slide along b5. Body b1 is mounted via a joint j1 to the back end of b5. There is also a "motor" to drive j1. The other end of b1 is mounted to j2, which is a rod with spherical joints at each end. The mass of j2 is given by b2. The element j2 is also connected to b3. The components j1, b1, j2, b3, j3 and b5 form a kinematic loop.

There is another kinematic loop j3, j4, b6, j5, b7, j6, j7 and b4, which are in the front part of the 3D view. The joint j3 makes the two loops coupled.

When the joint j1 has turned half a revolution, then b3 will be back in the same position. It means that the position of b3 cannot be used as a state, because there is two possible configurations for the same position. The positions of the components in the second kinematic loop j3, j4, b6, j5, b7, j6, j7 and b4 are completely determined by the position of b3. Thus their positional variables cannot be used as states. Also the angle relative angle between b1 and b2 is determined by the position of j3. It is possible to use the angle and angular velocity of j1 as the states all the time. If this had not been the case, it had not been possible to run the mechanism by driving j1.

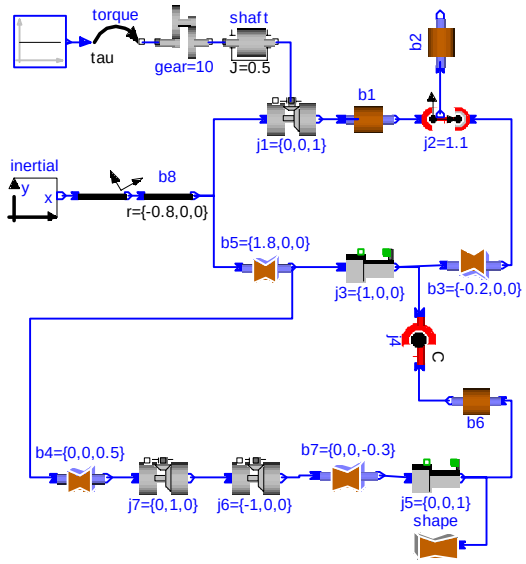



**Figure** 2: A Modelica model of the two coupled kinematics loops.

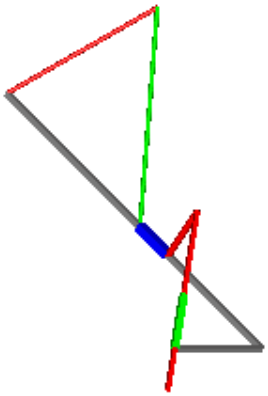


**Figure** 3: Initial configuration of mechanism.

When Dymola translates the model, it identifies two sets for dynamic selection of states. One is on position level where the relative angle of the joints j1, j5, j6, and j7 and the relative distances of j3 and j5 are potential candidates to be the positional state for this one degree-of-freedom system. The other set contains the corresponding velocities. Dymola is not able to deduce that the angle of j1 can be used as the state all the time because the Jacobian for the system to eliminate states is complex including sine and cosine functions of the joint angles.

At the start of a simulation run, Dymola selects the angle and angular velocity of j1 as states and keeps them for the complete simulation run. The overhead to check the selection of states increases the simulation time by 10%, which is a marginal increase compared to manual static selection of the angle and angular velocity of j1 as state variables.

## Conclusions

This paper has illustrated how Dymola solves the important problem of handling high-index DAE problems and selection of states in an efficient and reliable way.


### Acknowledgements

This work was in parts supported by the European Commission under contract IST-199-11979 with Dynasim AB under the Information Societies Technology as the project entitled "Real-time simulation for design of multi-physics systems".


## References


[1] K. Brenan, S Campbell and L. Petzold, *Numerical Solution of Initial Value Problems in Differential-Algebraic Equations*, North-Holland, Amsterdam, 1989

[2] Dymola — Dynamic Modeling Laboratory, Dynasim AB, Lund, Sweden. Homepage: http://www.Dynasim.se

[3] H. Elmqvist, S. E. Mattsson, and M. Otter, Modelica — *A Language for Physical System Modeling, Visualization and Interaction*, Proceedings of the 1999 IEEE Symposium on Computer-Aided Control System Design, CACSD'99, Hawaii, USA, 1999

[4] S. E. Mattsson and G. Söderlind, *Index Reduction in Differential Algebraic Equations Using Dummy Derivatives*, SIAM Journal on Scientific Computing, Vol. 14, No. 3, pp. 677-692, 1993

[5] Modelica, Modelica Association, Homepage: http://www.Modelica.org/.

[6] P. E. Nikravesh, *Computer-Aided Analysis of Mechanical Systems*, Prentice Hall, 1988

[7] C. C. Pantelides, The consistent initialization of differential-algebraic systems, SIAM Journal on Scientific and Statistical Computing, Vol. 9, pp. 213-231, 1988

[8] A. Watt and M. Watt, *Advanced Animation and Rendering Techniques: Theory and Practice*, Addison-Wesley, 1992