VOLUME 13 NUMBER 2 JUNE 1996

TRANSACTIONS

of the Society for Computer Simulation International

Bernard P. Zeigler
Editor-In-Chief

SCS The Society for Computer Simulation International

Publication and subscription information:
Volume 13, Number 2, June 1996, ISSN 0740-6797 / 96 $3.00+.10 P.O. Box 17900, San Diego, California 92177. Telephone (619) 277-3888. Subscriptions: $145 per year, U.S. Special rates to members of The Society for Computer Simulation. Change of address: Send both old and new address to (include old label, if possible): Circulation Office, P.O. Box 17900, San Diego, California 92177. Allow six weeks for delivery. Use subscription number on all correspondence. Undelivered copies: Please return to address above.

TRANSACTIONS

of the Society for Computer Simulation International

VOLUME 13 NO 2 JUNE, 1996

Tearing algebraic loops in bond graphs

W. Borutzky and F. E. Cellier

Department of Computer Science, Cologne Polytechnic, D-51643 Gummersbach, Germany
FAX: 49.2261.8196.15 *E-mail: wolfgang.borutzky@uni-koeln.de*
Department of Electrical and Computer Engineering, University of Arizona, Tucson, Arizona 85721-0104
Tel: 602-621-6192 *FAX:* 602-621-8076 *E-mail: cellier@ece.arizona.edu*

For bond graphs with causal paths between resistive ports the mathematical model is a set of Differential Algebraic Equations. Depending on the purpose of the model, and the available software, there are different options to process algebraic constraints. A bond-graph-based approach is proposed that exploits tearing. By inserting special sinks into causal paths, indication is provided to the model processor as to which variables could be used as tearing variables, and which equations for their determination. An algorithm is proposed that identifies suitable locations for those sinks in the bond graph such that the number of tearing variables is small. It is shown how the approach can be applied to bond graphs with causal paths between stores as well.

Keywords: Bond graphs, algebraic loops, combined symbolic/numerical solution of index-one DAEs, tearing, algebraic companion model of a storage element

1. Introduction

It is well known that causal paths between resistive ports in a bond graph (class two ZCPs [1]) mean that there exist in the model algebraic constraints in addition to the state equations. If all storage elements are independent of each other, the mathematical model is of the semi-state-space form

$$\mathbf{x} = \mathbf{f}(\mathbf{x}, \mathbf{h}, \mathbf{u}, t) \tag{1.a}$$

$$0 = \mathbf{g}(\mathbf{x}, \mathbf{h}, \mathbf{u}, t) \ , \tag{1.b}$$

in which $\mathbf{h}$ is a vector of auxiliary variables.

Depending on the purpose of the model, its structural properties, and the available software, a model of the above form can be treated in different ways. One possible way that does not require any changes to the model is to pass the equations (automatically) derived from the bond graph without any investigation directly to a DAE solver such as the DASSL code [2] that approximates the derivative $\dot{\mathbf{x}}$ in terms of $\mathbf{x}$ by using a Backward Differentiation Formula (BDF), and then solves the resulting non-linear algebraic system by a modified Newton iteration. Structural properties of the linearized algebraic systems can be exploited by using an appropriate linear algebra package that accounts for the structure of the system matrix.

The Differential Algebraic Equation (DAE) solution, however, may not always be the most efficient way to deal with the problem. For example, if the model is non-stiff, a DAE approach forces the use of an iterative solution, which, in this situation, is numerically inefficient.

An alternative approach that has been known for a long time is to add small storage elements to the graph that have a negligible impact on the overall dynamic behavior of the system (e.g., see [3]). As a consequence, the resulting explicit Ordinary Differential Equations (ODEs) will be stiff, and an implicit stiffly-stable integration method is needed for their solution. A crucial point that requires some experience is the proper choice of parameter values for the added storage elements and for resistors, which ensure that the introduced high frequency oscillations quickly fade away. A disadvantage of this approach is that the complexity of the model is increased, while, on the other hand, the resulting high frequency transients are unwanted.

In order to avoid an (unnecessary) introduction of stiffness into a model, Barreto and Lefèvre proposed in an earlier paper [4] replacing the implicit algebraic part of a bond graph characterized by uncompleted causalities by an R-field, to solve for the output variables of that field, and to use an explicit integration method. As they reported, the procedure can be carried out by means of a rather simple program like TUTSIM [5], if the R-field is further replaced by a block diagram, since TUTSIM does not support multiport elements. The approach is mainly applicable to rather small models that can be handled manually.

The objective in this paper is to process efficiently large-scale systems that may comprise a considerable number of implicit algebraic equations. To achieve that goal, it is proposed to exploit symbolic formulae manipulation capabilities, as they are available, for instance, in the program Dymola, before numerical integration takes place. Certainly, Dymola supports the generation of code for a direct numerical solution

Received September 25, 1995
Revised September 12, 1996
Accepted October 1, 1996

TRANSACTIONS of The Society for Computer Simulation International
ISSN 0740-6797/96 $2.00+.10

Volume 13, Number 2, pp. 102-115

of DAE systems as a compiler option; however, a symbolic transformation of DAE systems to explicit ODE form may often lead to more efficient simulation code, especially if appropriate tearing information is provided to the compiler by the user. In this paper, it will be shown how the bond graph helps to determine appropriate tearing information.

The approach that is proposed exploits the structure of the set of algebraic equations and is based on tearing, a method introduced by Kron [6] in the early 60's, which splits up a large system of algebraically coupled equations into a number of smaller systems. Tearing is not limited to linear algebraic equations. The method can just as easily and profitably be employed in the case of nonlinear equation systems, as discussed, for instance, in [7]. In order to be able to apply this technique, tearing variables must be determined somehow. Since the task of determining a *minimal* set of tearing variables is an NP-complete problem, as has been shown by Mah [8], it is impractical to determine the minimal set in an exhaustive search. There are a number of heuristic methods, however, for finding a small (but not necessarily the smallest) set of tearing variables. In [8], Mah presents 12 different algorithms.

In this paper, it will be shown that, by inserting controlled sinks in causal paths of a bond graph between resistive ports, information can be passed from a bond graph to a model processor that helps it identify possible tearing variables and their associated residual equations used to compute them. Of course, it is not satisfactory to have the modeler inspect the causally augmented bond graph to look for algebraic loops and to manually add sinks in appropriate places. Rather, these steps should be implemented in a bond graph preprocessor.

If the algebraic equations are linear with respect to the tearing variables, then, by solving them symbolically, an initial index-one DAE system can be reduced efficiently to an explicit ODE problem such that there is no need for a DAE solver. Certainly, as indicated above, it is a comfortable and rather safe way to pass a model containing algebraic constraints directly to a DAE solver. Nevertheless, the approach we shall present here may be a more efficient alternative in some cases. For instance, if the ODEs are non-stiff, there is no need for an implicit integration algorithm, and a costly Newton iteration at each time point can be completely avoided. Of course, for nonlinear algebraic constraints, a symbolic solution is not possible. Due to tearing, however, iteration is only required on a *small number* of variables (the tearing variables), which reduces the costs of calculating, updating, and LU-decomposing the Jacobian and is an attractive feature. With regard to large systems of linearized algebraic equations, tearing can be viewed as an alternative to sparse matrix approaches applied to the overall non-partitioned system matrix [9]. Finally, since the solution of the algebraic equations can be interpreted as the substitution of a bond graph part by a multiport R-element with causalities according to the rest of the bond graph, the objective of our approach can be viewed as being similar to that of Barreto and Lefèvre [4]. Like those authors, we also suggest not excluding explicit methods from consideration, if there are implicit algebraic constraints. In contrast to Barreto and Lefèvre, however, we advocate a combined symbolic/numerical approach. Moreover, by employing tearing, we address large-scale systems. The key point is to identify a *small* but sufficient number of tearing variables.

2. Tearing algebraic loops

In this section, a heuristic approach to determining a small set of tearing variables is explained by means of a fairly small linear bond graph model. As pointed out above, the technique is, however, not limited to the linear case.

Let us consider a bond graph with a tree structure with five resistors and one storage element as depicted in Figure 1. In that graph the resistive ports are labeled by encircled numbers. Application of the Sequential Causality Assignment Procedure (SCAP) [10] immediately reveals that there is some freedom in assigning causality to the resistors. This indicates that algebraic loops will result. Since the standard SCAP does not provide any help in case of an incomplete causality, Lorenz and Wolper [11] investigated different choices of causality on unassigned bonds in some small examples, and initiated a search for the missing rules that help to minimize the computational costs for solving the algebraic loops. Moreover, in [12], Gawthrop and Smith propose a modification to the SCAP in order to make the algebraic variables associated with algebraic loops explicit on the bond graph by adding a special effort source to a causally incomplete zero junction, or a special flow source to a causally incomplete one-junction, respectively. In bond graphs, they label those sources SS "to emphasize that there is an implicit sensor associated with the additional sources" [12]. In our approach we shall exploit similar sources. The objective of this paper, however, is different than that of Gawthrop and Smith in [12]. As stated earlier, our aim is to introduce tearing information into a bond graph that can be exploited by the formulae manipulation capability of a model processor prior to the subsequent numerical solution of the model. The aim is not that "the choice of such variables is made by the bond grapher at the bond graph level and not left until the equation formulation and solution stages" [12]. Instead, as already pointed out, the proposed heuristic approach of adding sources to a bond graph indicating possible tearing variables can be implemented into a bond graph preprocessor such that the bond grapher does not need to be concerned with the choice of variables in algebraic loops. Similar to a choice among different integration algorithms, the modeler should have the option to choose between using a symbolic/numerical approach based on tearing, or passing the model directly to a DAE solver. Although rather straightforward, however, an automated implementation of this feature has not yet been attempted.

The approach will be demonstrated by means of the example shown in Figure 1. In order to link our bond-graph-based approach to a classical tearing approach at the level of equations, we first provide the equations for the bond graph in Figure 1 and verify that the variables resulting from an inspection of the causal bond graph indeed represent a possible choice

of tearing variables. In order to pass tearing variables and corresponding equations that determine those variables to a model processor, we add sinks to the bond graph that are similar to those introduced by Gawthrop and Smith in [12]. Since the mechanism is based on the features of the equation processing capabilities of Dymola, we need to explain the implementation of those sinks that we term *residual sinks*. They are not motivated by physical reasons. They are mathematical objects that support a symbolic/numerical approach based on tearing. As pointed out above, such an approach may be computationally attractive in some cases. The sinks can be interpreted as Lagrange multipliers.

Let us return to the example under consideration. For instance, assigning resistive causality to the R-elements attached to the one-junctions results in algebraic loops between the resistors R_1-R_2, R_3-R_4, R_1-R_4, and R_3-R_5, coupled by another algebraic loop between R_1 and R_5. Bond graphers will see immediately that the flow variables f_1, f_3, and f_5 break all algebraic loops. We shall verify this in a conventional manner at the level of equations. By looking at the bond graph in Figure 1, the following equations can be derived:

$$e_1 = R_1 \cdot f_1 \tag{2.a}$$

$$e_3 = r_3 \cdot f_3 \tag{2.b}$$

$$f_2 = f_1 - f_3 \tag{2.c}$$

$$f_4 = f_3 - f_5 \tag{2.d}$$

$$R_5 \cdot f_5 = E - e_1 - e_3 - e_C \tag{2.e}$$

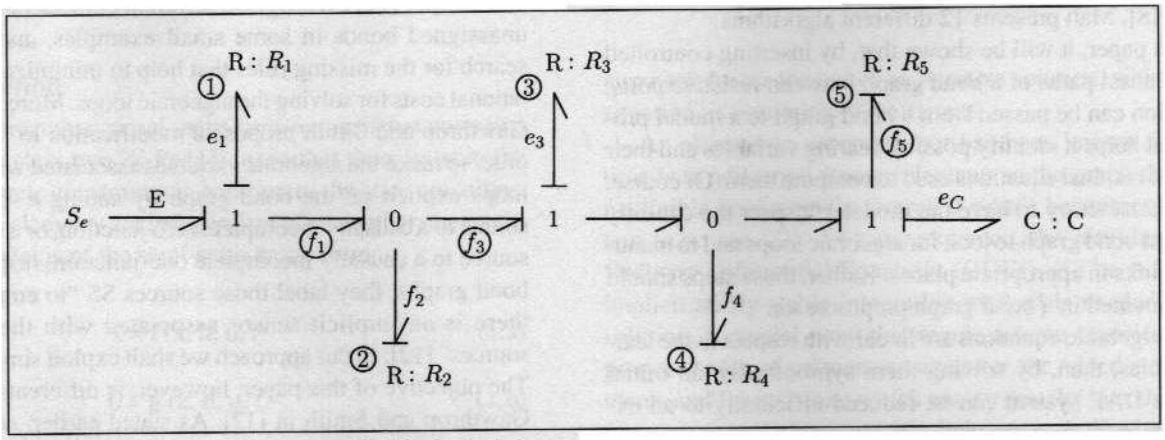

Figure 1. Tree-structured bond graph with algebraic loops

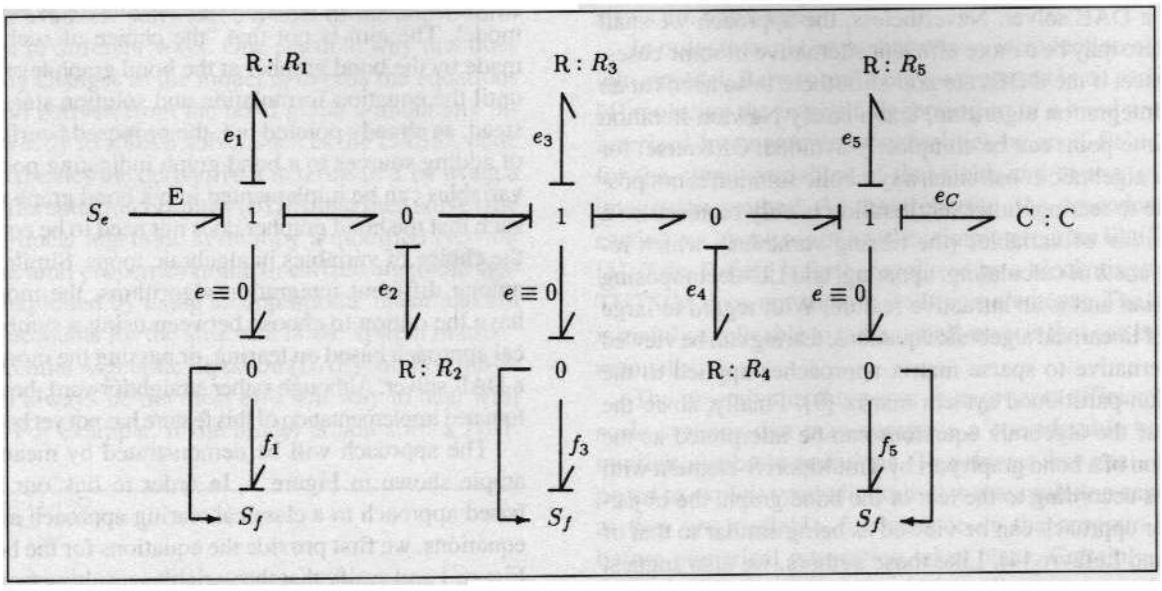

Figure 2. Bond graph with residual flow sinks

$$R_2 \cdot f_2 = E - e_1 \tag{2.f}$$

$$R_4 \cdot f_4 = E - e_1 - e_3 \tag{2.g}$$

$$\dot{e}_C = \frac{1}{C} f_5 \ . \tag{2.h}$$

By writing the algebraic equations in the form of a matrix equation:

$$\left(\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 0 & -R_1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -R_3 \\ 0 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & -1 \\ \hline 1 & 1 & 0 & 0 & R_5 & 0 & 0 \\ 1 & 0 & R_2 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & R_4 & 0 & 0 & 0 \end{array}\right) \cdot \left(\begin{array}{c} e_1 \\ e_3 \\ f_2 \\ f_4 \\ \hline f_5 \\ f_1 \\ f_3 \end{array}\right) = \left(\begin{array}{c} 0 \\ 0 \\ 0 \\ 0 \\ \hline E - e_C \\ E \\ E \end{array}\right), \tag{3}$$

it can be seen that, indeed, since the upper left submatrix is in lower triangular form, all variables e_1, e_3, f_2, f_4, can be computed by Gaussian forward substitution once the flows f_1, f_3, and f_5 are known; cf., also equations (5), (7), (8), and (10). Consequently, if *I*-elements providing the unknown flows are attached to the three one-junctions, then all algebraic loops apparently disappear. The flows f_1, f_3, and f_5 become state variables that determine all other non-state variables. Of course, the inertance parameters associated with the added *I*-elements must be made small, in order to keep the impact on the overall dynamic system behavior negligible. Consequently, the resulting state equations may be stiff. If the parameter values of the inertances tend to zero, their state equations turn into a set of coupled algebraic equations, which, in general, must be solved simultaneously. The variables of that subsystem (along with the state variable), however, still determine all other variables. Hence, they may be chosen as tearing variables. Since the flows f_1, f_3, and f_5 are determined such that their corresponding efforts vanish, they may be considered Lagrange multipliers. In a bond graph, those non-state variables can be represented by controlled flow sinks. In order to avoid the introduction of another bond graph element and to make the sensing of the effort explicit, we adopt a representation used by Bos [13, page 124] and van Dijk [14]; cf., Figure 2.

Observing that the effort into the flow sinks vanishes, the following equations can be derived from the bond graph in Figure 2:

$$e_1 = R_1 \cdot f_1 \tag{4.a}$$

$$e_2 = R_2 (f_1 - f_3) \tag{4.b}$$

$$e_3 = R_3 \cdot f_3 \tag{4.c}$$

$$e_4 = R_4 (f_3 - f_5) \tag{4.d}$$

$$e_5 = R_5 \cdot f_5 \tag{4.e}$$

$$0 = E - e_1 - e_2 \tag{4.f}$$

$$0 = e_2 - e_3 - e_4 \tag{4.g}$$

$$0 = e_4 - e_5 - e_C \tag{4.h}$$

$$\dot{e}_C = \frac{1}{C} f_5 \tag{4.i}$$

Note that since the flows f_1, f_3, and f_5 are not state variables, we need five equations for the effort variables of the resistors and another three equations (4.f)-(4.h) that determine the flows of the controlled sinks.

The algebraic equations can be written in the following form:

$$\left(\begin{array}{cc} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{array}\right) \left(\begin{array}{c} \mathbf{e} \\ \mathbf{f} \end{array}\right) = \left(\begin{array}{c} \mathbf{b}_1 \\ \mathbf{b}_2 \end{array}\right) \tag{5}$$

with $\mathbf{e} = (e_1\ e_2\ e_3\ e_4\ e_5)^T$, $\mathbf{f} = (f_1\ f_3\ f_5)^T$:

$$\left(\begin{array}{ccccc|ccc} 1 & 0 & 0 & 0 & 0 & -R_1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -R_2 & R_2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -R_3 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -R_4 & R_4 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & -R_5 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \end{array}\right) \cdot \left(\begin{array}{c} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ \hline f_1 \\ f_3 \\ f_5 \end{array}\right) = \left(\begin{array}{c} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \hline E \\ 0 \\ e_C \end{array}\right) \tag{6}$$

Since in this simple example the matrix $\mathbf{A}_{11}$ is an identity matrix of size 5 x 5, the equation

$$\mathbf{A}_{11}\,\mathbf{e} + \mathbf{A}_{12}\,\mathbf{f} = \mathbf{b}_1 \tag{7}$$

can be solved symbolically for **e** and inserted into

$$\mathbf{A}_{21}\,\mathbf{e} + \mathbf{A}_{22}\,\mathbf{f} = \mathbf{b}_2\,, \tag{8}$$

which gives an equation determining the tearing variables f_1, f_3, and f_5; cf., also [7]. In the example, the submatrix $\mathbf{A}_{22}$ is a zero matrix of size 3 x 3, and we obtain for the vector **f** the equation

$$(\mathbf{A}_{21}\ \mathbf{A}_{12})\,\mathbf{f} = \mathbf{A}_{21}\,\mathbf{b}_1 - \mathbf{b}_2\ . \tag{9}$$

That is, the controlled flow sources introduced into the bond graph indeed represent possible tearing variables.

In the general case, permutation matrices **P** and **Q** are needed to transform the algebraic equations into a matrix equation (5) with a submatrix $\mathbf{A}_{11}$ that is lower block-triangular with non-vanishing diagonal elements. Consequently, if $\mathbf{A}_{11}$ is non-singular, equation (5) can be solved symbolically for **e** in the general case as well.

If the tearing variables and the equations that determine them are known, transformation of a system of linearized equations to a system with a bordered block-triangular matrix is always possible, and will be performed by Dymola if a compiler switch requesting tearing to be used is set *on*. The transformation of a structural non-singular matrix **PA** into a block triangular matrix $\mathbf{Q}^T(\mathbf{PA})\mathbf{Q}$ is based on Tarjan's algorithm [15], finding the strong components in a directed graph.

Finally, we shall give an idea of the savings with respect to computational time due to tearing, if the algebraic equations are solved numerically. Suppose that the equations are linear and that $\mathbf{A}_{11}$ is a non-singular lower-triangular matrix. Inserting equation (7) into equation (8) gives

$$\mathbf{A}_{22} - \mathbf{A}_{21}\,\mathbf{A}_{11}^{-1}\,\mathbf{A}_{12}\,\mathbf{f} = \mathbf{b}_2 - \mathbf{A}_{21}\,\mathbf{A}_{11}^{-1}\,\mathbf{b}_1\ . \tag{10}$$

Suppose there are n algebraic unknowns and p tearing variables. Since the matrix $\mathbf{A}_{11}$ is lower-triangular, the computational cost for the transformation into equation (10) is $O((n-p)^2)$. Solving a linear system with a dense matrix of dimension p requires $O(p^3)$ long operations. Consequently, comparing the computational cost for solving the system with and without the use of tearing roughly reveals the ratio

$$ratio = \frac{n^3}{(n-p)^2 + p^3}\ .$$

For the above small example (cf., Figure 2) with $n = 8, p = 3$, it is about 15.5 times more costly to solve the overall system by Gaussian elimination than to employ a solution based on tearing.

3. Residual sinks in Dymola

One reason for using the language Dymola is that it provides an operator, *residue*(), allowing the model processor for identifying a tearing variable and an equation that determines the tearing variable. The purpose of this section is to explain that mechanism.

An inertance with integral causality provides a flow such that

$$I \cdot \frac{df}{dt} = e\ . \tag{11}$$

For $I \to 0$, zero on the left-hand side is replaced by *residue*(*f*):

$$residue(f) = e\ . \tag{12}$$

(Obviously, the limit only exists if the parameter I is not in the denominator.) For the Dymola model processor, the meaning of equation (12) is that f is a tearing variable, and *residue*(*f*) a variable that must be kept zero. Consequently, with these two variables, the equations (4.f), (4.g), and (4.h) derived from the bond graph depicted in Figure 2, can be written equivalently in the form:

$$residue(f_1) = E - e_1 - e_2 \tag{13.a}$$

$$residue(f_3) = e_2 - e_3 - e_4 \tag{13.b}$$

$$residue(f_5) = e_4 - e_5 - e_C \tag{13.c}$$

Adding a vanishing term to equations (4.f), (4.g), and (4.h) enables the Dymola model processor to identify them as the residual equations for the tearing variables f_1, f_3, and f_5.

Equation (12) can be interpreted as the constitutive equation of a flow sink that provides a flow such that the input effort vanishes. Its output can be viewed as a Lagrange multiplier.

In the object-oriented language Dymola, such a sink can be described by means of a *model class*, which is a generic model for that type of sink and which can be instantiated to represent an actual sink of that type in the bond graph.

```
model class ResSf
  main cut Port1 (e/f)
    residue(f) = e
end
```

In the second line, a port with the name *Port*1 is declared by means of the keyword *cut*. Cuts are connection points for across variables. They are associated by a list of across variables and a list of through variables specified in parentheses. The two lists are separated by a slash. Since the essential equation in the body of that model class employs the *residue*() operator, we call the sink a *residual* (*flow*) *sink*.

Obviously, the way in which causalities and flow sinks have been added to the bond graph in Figure 1 is not the only possible one. Instead of flow sinks, dual controlled effort sinks could be used as well (cf., Figure 3). They are specified by the model class:

model class *ResSe*
  main cut *Port* 1 (*e*/*f*)
    residue(*e*) = *f*
end

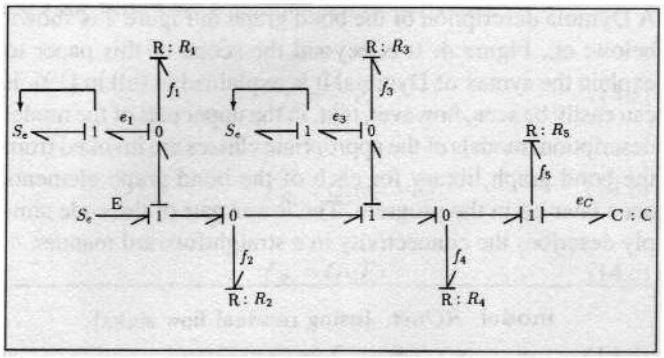

Figure 3. Bond graph with controlled effort sinks added

(In Bond Graphs of rigid multibody systems, that type of sink represents a constraint force in a joint [16].) Apparently, as has been discussed by Gawthrop and Smith in [12], this is another way of breaking algebraic loops; cf., Figure 3. Notice that, in this example, the dual approach leads to a smaller number of tearing variables.

4. An algorithm for adding tearing information to a bond graph

Guided by the consideration of bond graphs with class-2 ZCPs, we propose a simple, heuristic algorithm for adding tearing information to a bond graph.

The first step is to start from a resistive port and to track all causal one-way paths from that port to other resistive ports; that is, to follow all class-2 ZCPs starting from that port. This tracking must be done for all resistive ports in the bond graph. Fortunately, since normally a resistive element is not connected to all others, even if it is a multiport element, the number of class-2 ZCPs is usually much smaller than $n \cdot (n\text{-}1)/2$, in case there are n resistors. (If there are n vertices in a graph, and each node is connected to all others, then there are $n \cdot (n\text{-}1)/2$ edges.) From electronic circuits, it is well known that an element normally is not connected to all others but only to a small number of them, which leads to a sparse nodal admittance matrix for large circuits. If there is a multiport R-element with mixed causalities, then there are causal one-way paths through that element that may link resistive ports of other elements.

If resistive ports are numbered (cf., the bond graph in Figure 1), a bookkeeping of the algebraic loops in a bond graph can be done by means of an algebraic loop matrix of which the rows as well as the columns represent resistive ports. An entry (x) in position (i, j) means that there is an algebraic loop from port i to port j and, since each bond represents two opposite signals, there is also an algebraic loop from port j to port i. One loop relates the effort variables, while the other establishes a relation between the corresponding flows. Hence, the algebraic loop matrix we introduce is square symmetric. It is sufficient to consider only the effort, or the flow relations, and to store only the upper, or the lower, triangular part of the matrix. For instance, for the bond graph in Figure 1, we obtain the algebraic loop matrix given in Table 1. Indeed, by looking at the bond graph in Figure 1, we see that there are, for instance, causal one-way paths from port 1 to port 2, as well as from port 1 to port 4 and from port 1 to port 5. Therefore, the first row of the matrix in Table 1 (representing port 1) has entries in columns 2, 4, and 5.

Table 1. Algebraic loop matrix corresponding to bond graph 1

port	1	2	3	4	5
1		x		x	x
2	x				
3				x	x
4	x		x		
5	x		x		

As can be seen from the bond graph in Figure 1, it is likely that long causal one-way paths between resistive ports have joint internal bonds with other algebraic loops. Since the aim is to break as many algebraic loops as possible by introducing one tearing variable, the row (and the column) containing the most entries is chosen and eliminated from the matrix because the entries in one row indicate algebraic loops that have joint bonds. For instance, if there is a causal one-way path from port 1 to port 2 and another disjoint algebraic loop from port 4 to 5, then there are entries in position (1,2) as well as in position (2,1), but not in positions (1,4) and (1,5). If there are several rows (columns) having the same number of entries, the first one is selected. The procedure is repeated until all entries in the array are eliminated. In the above example, removing the first row and first column, and subsequently the third row and third column, indeed leads to an empty array. If instead the first row and first column and the fourth row and fourth column were removed deliberately, entries in position (3,5) and (5,3) would remain. In that case, port three would have to be chosen additionally.

As a result, we obtain a small, not necessarily minimum number of resistive ports (in the bond graph of Figure 1 the resistors with parameters R_1, respectively, R_3). If we cut the incident bond at each of those ports and insert a residual sink (cf., Figure 3), causality is reversed at that port, so that other resistive ports in the bond graph cannot be reached any longer from that port via causal one-way paths, i.e., all algebraic loops in the bond graph vanish. If all resistors in Figure 1 have a

nonlinear, non-invertible characteristic, causality at their ports is predetermined. In that case, controlled flow sinks attached to one-junctions along with controlled effort sinks added to the zero-junctions break the algebraic loops. This means that the algebraic loops are cut at the internal bond with the associated pair of power variables (f_1, e_2) and at the internal bond with the variables (f_3, e_4).

In the above matrix, a causal one-way path between two resistive ports is characterized only by the two bonds incident to the terminal ports. None of the internal bonds that contribute to a causal one-way path are stored. As has been shown by Lorenz and Wolper in [11], in some cases there may be a smaller number of variables that break all algebraic loops, if variables of internal bonds are chosen. In particular, in case a closed causal path (cycle) in the junction structure touches a causal one-way path between resistive ports, the algebraic loops involved are torn if a residual sink is inserted in a joint internal bond of both causal paths. In regard to an automatic, generally applicable approach, however, further investigation is needed and will be the subject of further research. There is no unique choice for good tearing variables. In our search for a simple algorithm that automatically provides possible tearing variables as well as their residual equations in a form that is understood by the model processor Dymola, variables at internal bonds have been excluded from consideration in a first approach. As a consequence, causal cycles and causal meshes [1] in a junction structure are not torn, and in some cases a more sophisticated algorithm may reveal a smaller number of tearing variables.

The algorithm presented here has been tested on a number of bond graphs with different structural properties. Some of them have been considered by other authors as well; see [11], [12], [14], and [17].

5. Bond graphs in Dymola

Dymola is a language that was not designed for bond graphs. It is based on the concept of generalized networks using across and through variables. Nevertheless, as shown in [18], bond graph elements can be described in Dymola. Although feasible, descriptions of bond graphs in Dymola are not completely satisfactory up to now, since there is no equivalent element to a bond graph 1-junction in the language. To overcome this deficiency, one may use gyroscopic bond graphs only (as done in [18]), or define a model class for that bond graph element (the approach advocated here), which is not convenient either. For instance, since the number of *cuts* cannot vary in a Dymola model, and since power flow directions must be incorporated into the definition of the cuts, several model classes are needed, differing in the number and orientation of bonds attached to the 1-junction. Once appropriate model classes have been defined (in a hierarchical manner), however, they can be stored in a library, and the user does not need to be concerned with those details any longer. A graphical description can be automatically translated into Dymola by referencing models from that library. For instance, for each one-junction the number of ports is known. After power reference directions have been added automatically to a bond graph, for each one-junction in the bond graph the proper model class from the library can be instantiated automatically.

Such a library has been set up and used for the above example, as well as for bond graphs of multibody systems (MBS). A Dymola description of the bond graph in Figure 2 is shown below; cf., Figure 4. It is beyond the scope of this paper to explain the syntax of Dymola. It is explained in full in [19]. It can easily be seen, however, that, in the upper part of the model description, models of the appropriate classes are invoked from the bond graph library for each of the bond graph elements used later on in the program. The lower part of the code simply describes the connectivity in a straightforward manner.

```
model RCnet {using residual flow sinks}
submodel (Se)     Se
submodel (ResSf)  Sf1, Sf2, Sf3
submodel (R)      R1, R2, R3, R4, R5
submodel (C)      C
submodel (one4P)  one1, one2, one3
node              zero1, zero2

input E

connect Se        at one1 : Port1
connect zero1     at one1 : Port2
connect R1        at one1 : Port3
connect Sf1       at one1 : Port4

connect R2        at zero1
connect zero1     at one2 : Port1
connect zero2     at one2 : Port2
connect R3        at one2 : Port3
connect Sf2       at one2 : Port4

connect R4        at zero2
connect zero2     at one3 : Port1
connect R5        at one3 : Port2
connect C         at one3 : Port3
connect Sf3       at one3 : Port4

Se.E0 = E

end
```

Figure 4. Dymola description of bond graph with residual flow sinks

6. Causal violations at junctions

If resistive ports have a non-invertible, or preferred, characteristic, causal violations may occur at junctions instead of algebraic loops; in particular, if the method of relaxed causality of Joseph and Martens [20] is used. This type of problem can be solved as well by adding controlled sinks representing a possible tearing variable.

Consider a simple circuit with two nonlinear resistors in series for which the constitutive law is assumed to be non-invertible. The bond graph is depicted in Figure 5. $G_1()$ and $G_2()$ symbolize that these are nonlinear resistors

with conductance causality. Inserting a (small) capacitor as indicated in Figure 6 obviously removes the causal violation at the 1-junction. Instead of the small C-element, a controlled effort sink can be introduced that imposes an effort E such that the corresponding flow vanishes. From the bond graph in Figure 7 the following equations can be derived:

$$f_R = \frac{1}{R} e_C \tag{14.a}$$

$$f_{R_1} = G_1(e_C - E) \tag{14.b}$$

$$f_{R_2} = G_2(E) \tag{14.c}$$

$$residue(E) = f_{R_1} - f_{R_2} \tag{14.d}$$

$$\dot{e}_C = \frac{1}{C}\left(F - f_R - f_{R_1}\right) \tag{14.e}$$

If the tearing variable E is known, all other non-state variables can be computed. The residual equation, in this case, is nonlinear:

$$G_1(e_C - E) = G_2(E) \quad , \tag{15}$$

and must be solved by iteration.

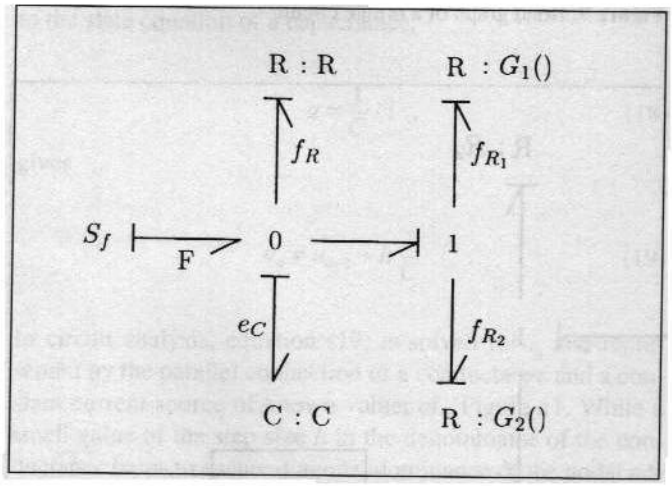

Figure 5. Bond graph with a causal violation at a 1-junction

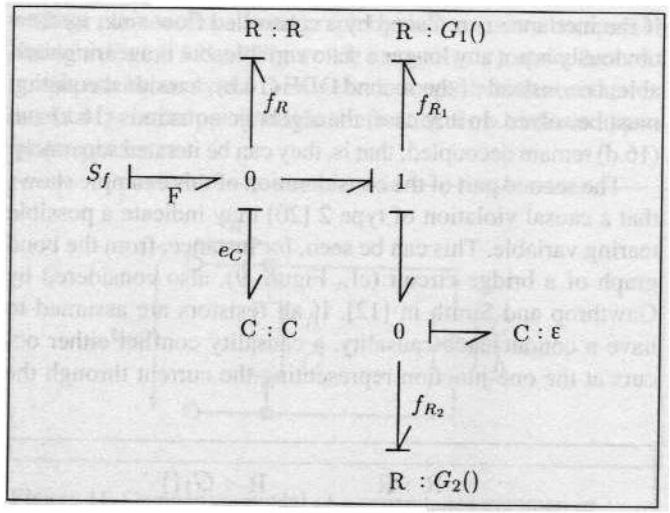

Figure 6. Bond graph with C element added

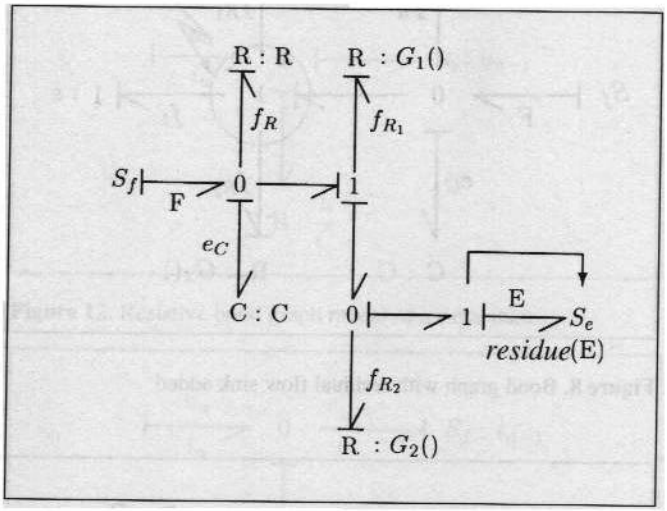

Figure 7. BG with residual effort sink added

If an inductance with a small parameter ε is attached to the one-junction instead of the small capacitance (cf., Figure 8), the causal violation is not removed. However, it is interesting to note that the algebraic equations that need to be solved in order to evaluate the right-hand side of the ODE, are decoupled by means of the additional state variable.

$$\dot{e}_C = \frac{1}{C}\left(F - \frac{1}{R} \cdot e_C - f_I\right) \tag{16.a}$$

$$\varepsilon \cdot \dot{f}_I = \left(e_C - e_{R_1} - e_{R_2}\right) \tag{16.b}$$

$$G_1\left(e_{R_1}\right) = f_I \tag{16.c}$$

$$G_2\left(e_{R_2}\right) = f_I \tag{16.d}$$

If the inertance is replaced by a controlled flow sink, its flow obviously is not any longer a state variable, but is a tearing variable, i.e., instead of the second ODE (16.b), a residual equation must be solved. In any case, the algebraic equations (16.c) and (16.d) remain decoupled; that is, they can be iterated separately.

The second part of the consideration of this example shows that a causal violation of type 2 [20] may indicate a possible tearing variable. This can be seen, for instance, from the bond graph of a bridge circuit (cf., Figure 9), also considered by Gawthrop and Smith in [12]. If all resistors are assumed to have a conductance causality, a causality conflict either occurs at the one-junction representing the current through the load or at both zero-junctions representing the voltages at the terminals of the load. Following the rule that a causal violation at a junction may indicate a tearing variable, we attach a controlled effort sink at both zero-junctions and end up with the same result as given by Gawthrop and Smith.

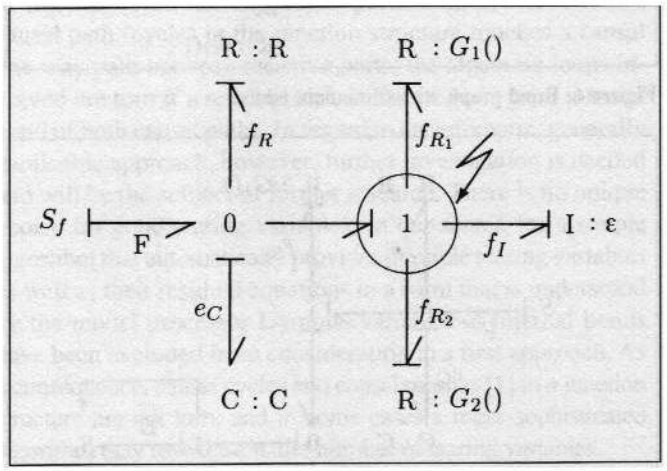

Figure 8. Bond graph with residual flow sink added

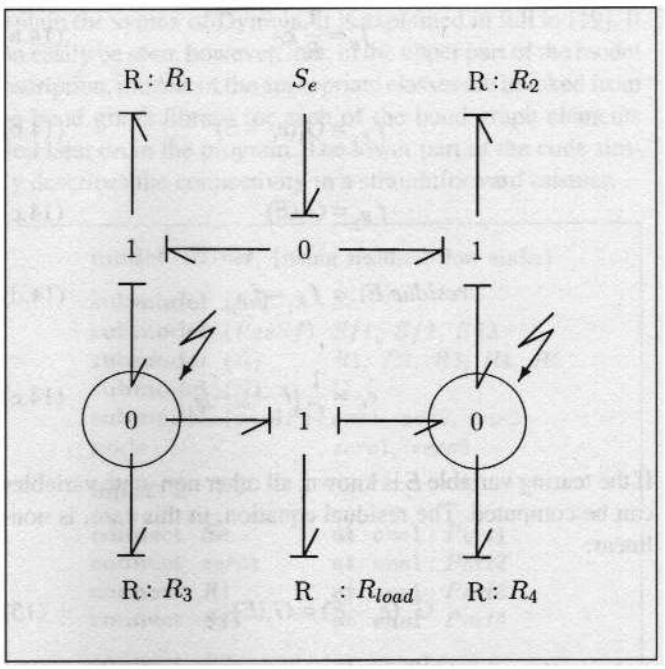

Figure 9. Bond graph of a bridge circuit

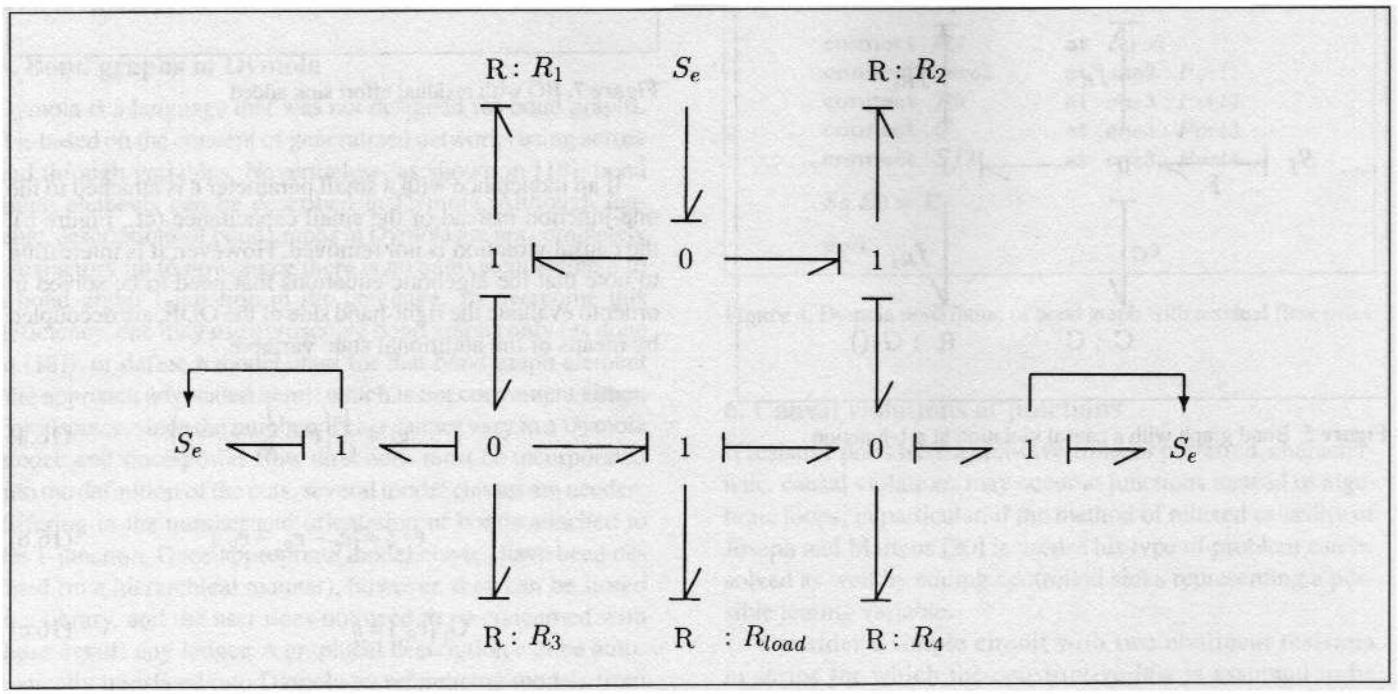

Figure 10. Bond graph of a bridge circuit with controlled effort sinks added

7. Causal paths between stores of the same type

If there exists a causal path between two storage elements of the same type (class-1 ZCP, [14, page 45]), this means that the state variables of these stores are related to each other by an algebraic equation. The approach explained so far can be extended such that it is applicable to this type of problem as well. To this end, we adopt an approach that is well known in circuit analysis, and by which the initial DAE system is transformed into an algebraic problem. Once we have implicit algebraic equations, we can employ the proposed approach again. In order to perform the transformation from a DAE system to a pure algebraic one, we need a modification of the initial bond graph. That is, the method presented so far is not directly applicable to bond graphs with class-1 ZCPs. For that reason, we shall reduce the problem of causal paths between storage elements to that of causal paths between resistive ports. In [21], we propose an alternative approach that does not require such a transformation.

In circuit analysis, model equations and the numerical integration formula are interleaved; that is, the integration formula (most commonly a BDF) is applied to the constitutive equations of the storage elements leading to a resistive companion model [9]. Let $x\,(t_n)$ be the analytical solution of x at time instance t_n, and let x_n denote an approximation obtained by numerical integration. If $x_n \approx x\,(t_n)$ and $f_n := f\,(x_n, t_n)$, application of, for instance, the Backward-Euler formula,

$$x_n = x_{n-1} + h \cdot f_n \ , \tag{17}$$

to the state equation of a capacitance,

$$\dot{u} = \frac{1}{C} \cdot i \ , \tag{18}$$

gives

$$u_n = u_{n-1} + h \frac{1}{C} \cdot i_n \ . \tag{19}$$

In circuit analysis, equation (19) is solved for i_n and represented by the parallel connection of a conductance and a constant current source of known value; cf., Figure 11. While a small value of the step size h in the denominator of the conductance helps to ensure diagonal dominance of the nodal admittance matrix allowing for the application of relaxation methods, it was shown in [22] that the approach leads to difficulties in solving general DAE systems, when the derivative $\dot{x}_n = f(x_n, t_n)$ is expressed in terms of x_n and values of x at past time points. In [22], it is recommended to insert equation (19) into the state-space model eliminating x_n instead of $\dot{x}_n$. In terms of bond graph elements, this means that a one-port C-element has to be replaced by a linear resistor of resistance h/C and a known constant effort sink; cf., Figure 12. Correspondingly, an inertance is replaced by a parallel connection of a conductance and a constant known flow sink; cf., Figure 13. Having replaced all storage elements in a bond graph by their resistive companion models (Figure 12 and Figure 13, respectively), again the bond graph can be analyzed for algebraic loops, which can then be broken by introducing residual effort or flow sinks at appropriate places.

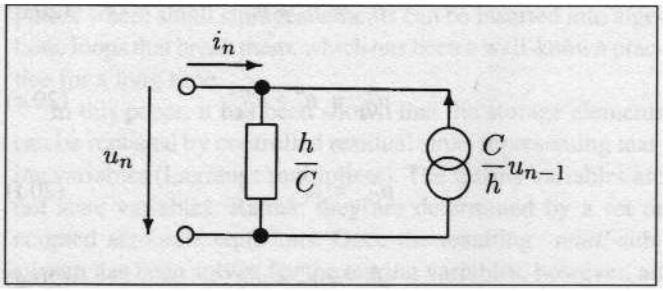

Figure 11. Companion model of a capacitance common in circuit analysis

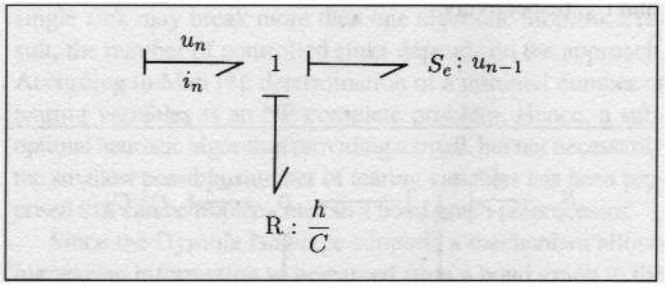

Figure 12. Resistive bond graph model of a capacitance

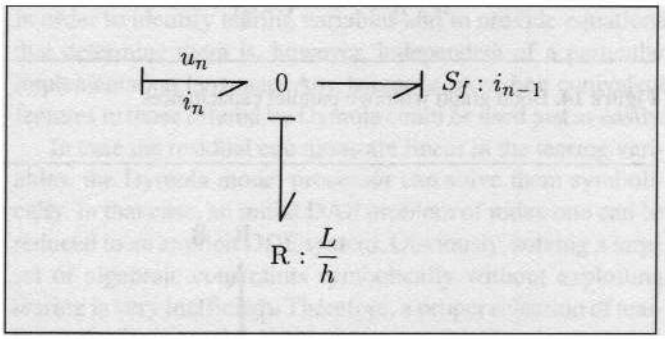

Figure 13. Resistive bond graph model of an inertance

In the sequel, we shall consider the simple example of two parallel capacitances. Replacing the storage elements by their resistive companion models, a bond graph with algebraic loops is obtained; cf., Figure 15. The algebraic loops can be solved by introducing two controlled sinks; cf., Figure 16. From the bond graph in Figure 16, the following equations can be derived directly:

$$u_R^n = R\left(i_1^n + i_2^n\right) \tag{20.a}$$

$$u_1^n = \frac{h}{C_1} i_1^n \tag{20.b}$$

$$u_2^n = \frac{h}{C_2} i_2^n \qquad (20.c)$$

$$E^n = u_R^n + u_{C_1}^n \qquad (20.d)$$

$$u_{C_1}^n = u_1^n + u_{C_1}^{n-1} \qquad (20.e)$$

$$u_{C_1}^n = u_{C_2}^n \qquad (20.f)$$

$$u_{C_2}^n = u_2^n + u_{C_2}^{n-1} \qquad (20.g)$$

In the above equations, superscripts indicate time points t_n and t_{n-1}, respectively.

The above equations can be written in matrix form:

$$\left(\begin{array}{ccccc|cc} 1 & 0 & 0 & 0 & 0 & -R & -R \\ 0 & 1 & 0 & 0 & 0 & -h/C_1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -h/C_2 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ \hline 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 \end{array}\right) \cdot \left(\begin{array}{c} u_R^n \\ u_1^n \\ u_2^n \\ u_{C_1}^n \\ u_{C_2}^n \\ \hline i_1^n \\ i_2^n \end{array}\right) = \left(\begin{array}{c} 0 \\ 0 \\ 0 \\ E^n \\ 0 \\ \hline u_{C_1}^{n-1} \\ u_{C_2}^{n-1} \end{array}\right) \qquad (21)$$

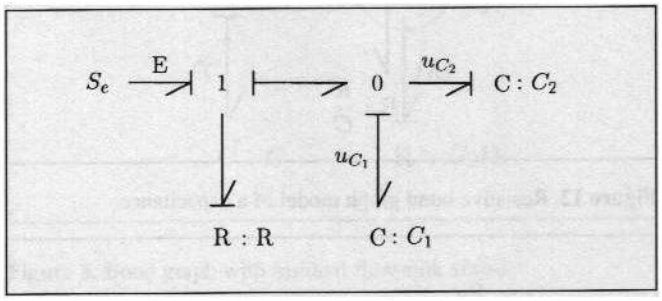

Figure 14. Bond graph with two parallel capacitances

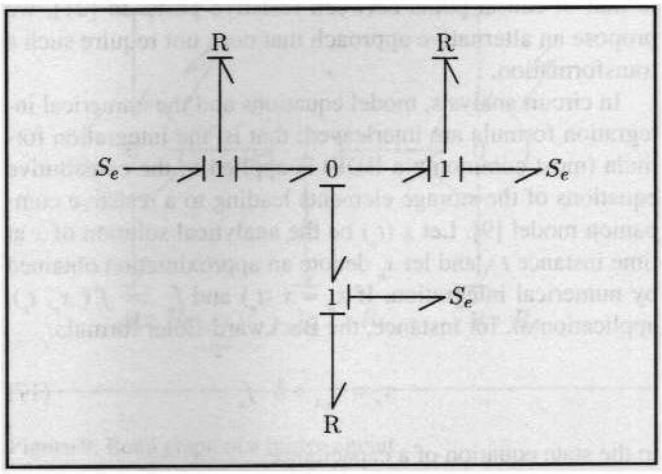

Figure 15. Resistive equivalent of parallel capacitors

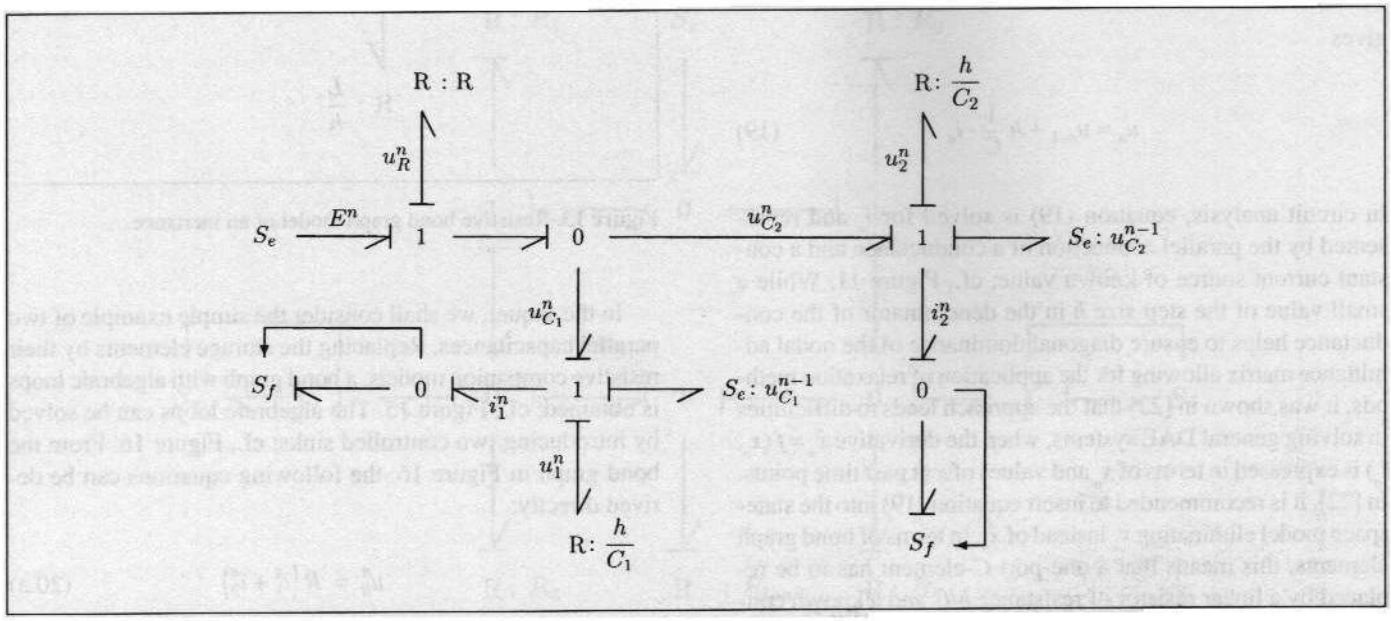

Figure 16. Resistive equivalent with controlled flow sinks

As was to be expected, the set of equations is in the form of equation (5). Since the $\mathbf{A}_{11}$ matrix is in lower-triangular form, it is trivial to eliminate the non-tearing variables from equation (8) by successive substitution. This results in the matrix equation of reduced size:

$$\begin{pmatrix} \left(R+\frac{h}{C_1}\right) & R \\ R & \left(R+\frac{h}{C_2}\right) \end{pmatrix} \begin{pmatrix} i_1^n \\ i_2^n \end{pmatrix} = \begin{pmatrix} E^n - u_{C_1}^{n-1} \\ E^n - u_{C_2}^{n-1} \end{pmatrix} \quad (22)$$

which can be solved easily for i_1^n and i_2^n. The following equation is found for i_1^n :

$$i_1^n = \frac{RC_1C_2\left(u_{C_2}^{n-1} - u_{C_1}^{n-1}\right) + C_1 h\left(E^n - u_{C_1}^{n-1}\right)}{R\left(C_1+C_2\right)h + h^2} . \quad (23)$$

Recognizing that $u_{C_2}^{n-1} = u_{C_1}^{n-1}$, equation (23) can be simplified to:

$$i_1^n = \frac{C_1\left(E^n - u_{C_1}^{n-1}\right)}{R\left(C_1+C_2\right)+h} . \quad (24)$$

Then, the remaining (non-tearing) variables can be solved for by back-substitution:

$$u_{C_1}^n = \frac{R\left(C_1+C_2\right)u_{C_1}^{n-1} + hE^n}{R\left(C_1+C_2\right)+h} . \quad (25)$$

In this simple example, both capacitors can alternatively be combined into one. If the resulting capacitor is replaced by its resistive companion model, the same equation (25) is obtained. Whereas dependent storage elements may be conveniently combined only in simple cases, as in the above example of two parallel capacitors, the tearing approach presented is general.

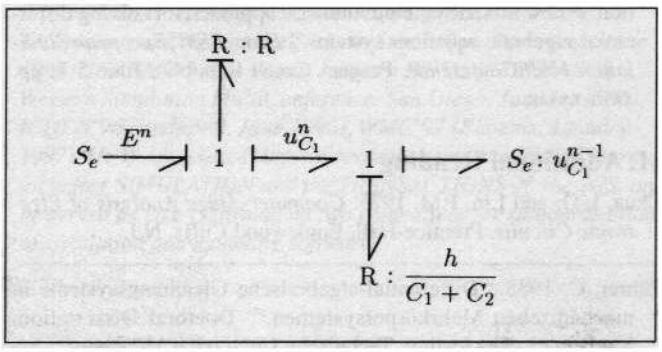

Figure 17. Resistive bond graph of combined capacitors

8. Conclusions

One of the advantages of causally augmented bond graphs is that algebraic loops can be immediately identified as causal paths between ports of resistors. In case the corresponding DAE system is unwanted, a bond graph representation suggests places where small storage elements can be inserted into algebraic loops that break them, which has been a well-known practice for a long time.

In this paper, it has been shown that the storage elements can be replaced by controlled residual sinks representing tearing variables (Lagrange multipliers). The tearing variables are not state variables. Rather, they are determined by a set of coupled algebraic equations. Once the resulting *small* subsystem has been solved for the tearing variables, however, all other algebraic variables can be expressed through the tearing variables and the state variables. There are different possibilities for adding controlled residual sinks, and, in some cases, a single sink may break more than one algebraic loop. As a result, the number of controlled sinks depends on the approach. According to Mah [8], determination of a minimal number of tearing variables is an NP complete problem. Hence, a suboptimal heuristic algorithm providing a small, but not necessarily the smallest possible, number of tearing variables has been proposed that can be implemented in a bond graph preprocessor.

Since the Dymola language supports a mechanism allowing tearing information to be passed from a bond graph to the model processor, it has been chosen for the description of bond graphs. The approach of adding controlled sinks to a bond graph in order to identify tearing variables and to provide equations that determine them is, however, independent of a particular implementation language. Any language providing equivalent features to those offered by Dymola could be used just as easily.

In case the residual equations are linear in the tearing variables, the Dymola model processor can solve them symbolically. In that case, an initial DAE problem of index one can be reduced to an explicit ODE system. Obviously, solving a large set of algebraic constraints symbolically without exploiting tearing is very inefficient. Therefore, a proper selection of tearing variables is crucial. In this paper, a rather simple algorithm for bond graphs with class two zero-order causal paths is proposed that has been tested on a number of examples.

If the algebraic equations are nonlinear, tearing is still advantageous. The sets of nonlinear algebraic equations that must be solved by Newton iteration are reduced. Consequently, calculating, updating and LU-decomposing the Jacobians is less costly. Hence, a combined symbolic and numerical approach is advocated that avoids the disadvantages of small additional storage elements, and that provides a computationally attractive alternative to passing the original model directly to a numerical DAE solver.

Finally, it has been shown that, by using a resistive companion model for the storage elements, the approach can easily be extended to bond graphs with causal paths between storage ports (class one ZCPs) as well. The resistive companion model corresponds to a BDF applied to the constitutive equation of the store.

For an actual solution of the resistive equivalent of an initial bond graph, a prerequisite is that the model equations be interleaved with a numerical integration formula. This technique has been termed *in-line integration* in [22]. In another article, [21], an alternative approach is investigated that attempts to tackle the problem of causal one-way paths between ports of storage elements directly. Tearing in bond graphs with causal cycles that touch causal paths of class two, respectively, class one, will be the subject of future research.

9. Acknowledgment

This paper is based on research work carried out during the first author's sabbatical leave at the University of Arizona in Tucson, in 1995. The first author gratefully acknowledges a grant from the German Research Council "Deutsche Forschungsgemeinschaft" (DFG) in Bonn, Germany.

10. References

[1] van Dijk, J. and Breedveld, P.C. 1991. "Simulation of system models containing zero-order causal paths —I. Classification of zero-order causal paths." *J. of the Franklin Institute* vol. 328, no. 5/6, pp. 959-980.

[2] Brenan, K.E., Campbell, S.L., and Petzold, L.R. 1989. *Numerical Solution of Initial-Value Problems in Differential Algebraic Equations*. North-Holland, New York.

[3] Karnopp, D.C. and Margolis, D. 1979. "Analysis and simulation of planar mechanism systems using bond graphs." *J. of Mechanical Design* vol. 101, pp. 187-191.

[4] Barreto, J. and Lefèvre, J. 1985. "R-fields in the solution of implicit equations." *J. of the Franklin Institute* vol. 319, no. 1/2, pp. 227-236.

[5] Meerman, J.W. 1988. *TUTSIM on IBM PC Computer — User's Manual*. Meerman Automation, Postbus 154, 7160 AC Neede, The Netherlands.

[6] Kron, G. 1962. *Diakoptics — The Piecewise Solution of Large-Scale Systems*. MacDonald & Co., London.

[7] Elmqvist, H. and Otter, M. 1994. "Methods for tearing systems of equations in object-oriented modeling." *Proc. ESM, European Simulation MultiConference*. Barcelona, Spain, June 1-3, pp. 326-332.

[8] Mah, R.S.H. 1990. *Chemical Process Structures and Information Flows*. Butterworths.

[9] Nagel, L.W. 1975. *SPICE2: A Computer Program to Simulate Semiconductor Circuits*. University of California, Electronic Research Laboratory, ERL-M 520.

10] Karnopp, D.C. and Rosenberg, R.C. 1968. *Analysis and Simulation of Multiport Systems*. M.I.T. Press, Cambridge, MA.

[11] Lorenz, F. and Wolper, J. 1985. "Assigning causality in the case of algebraic loops," *J. of the Franklin Institute* vol. 319, no. 1/2, pp. 237-241.

[12] Gawthrop, P.J. and Smith, L.P.S. 1992. "Causal augmentation of bond graphs." *J. of the Franklin Institute* vol. 329, no. 2, pp. 291-303.

[13] Bos, A.M. 1986. *Modelling Multibody Systems in Terms of Multibond Graphs with Application to a Motorcycle*. Doctoral Dissertation, Twente University, Enschede, The Netherlands.

[14] van Dijk, J. 1994. *On the Role of Bond Graph Causality in Modelling Mechatronic Systems*. Doctoral Dissertation, Twente University, Enschede, The Netherlands.

[15] Tarjan, R.E. 1972. "Depth first search and linear graph algorithms." *SIAM J. of Computing* vol. 1, pp. 146-160.

[16] Félez, J., Vera, C., San José, I., and Cacho, R. 1990. "BONDYN: A bond graph based simulation program for multibody systems." *J. of Dynamic Systems, Measurement, and Control* vol. 112, pp. 717-727.

[17] Hood, S.J., Rosenberg, R.C., Withers, D.H., and Zhou, T. 1987. "An algorithm for automatic identification of R-fields in bond graphs." *IBM J. Res. Develop* vol. 31, no. 3, pp. 382-390.

[18] Cellier, F.E. 1992. "Hierarchical nonlinear bond graphs: A unified methodology for modeling complex physical systems." *SIMULATION* vol. 58, no. 4, pp. 230-248.

[19] Elmqvist, H. 1995. *Dymola — User's Manual*. Dynasim AB, Park Ideon, Lund, Sweden.

[20] Joseph, B.J. and Martens, H.R. 1974. "The method of relaxed causality in the bond graph analysis of nonlinear systems." *J. of Dynamic Systems, Measurement, and Control* pp. 95-99.

[21] Borutzky, W. and Cellier, F.E. 1996. "Tearing in bond graphs with dependent storage elements." *Proc. Symp. on Modelling, Analysis, and Simulation. part of CESA'96: IMACS Multiconference on Computational Engineering in Systems Applications*. Lille, France, July 9-12, vol. 2, pp. 1113-1119.

[22] Elmqvist, H., Otter, M., and Cellier, F.E. 1995. "In-line integration: A new mixed symbolic/numeric approach for solving differential-algebraic equations systems." *Proc. ESM, European Simulation MultiConference*. Prague, Czech Republic, June 5-7, pp. xxiii-xxxiv.

11. Additional Reading

Chua, L.O. and Lin, P.M. 1975. *Computer-Aided Analysis of Electronic Circuits*. Prentice-Hall, Englewood Cliffs, N.J.

Führer, C. 1988. "Differential-algebraische Gleichungssysteme in mechanischen Mehrkörpersystemen." Doctoral Dissertation, Mathematisches Institut, Technische Universität München.

Steward, D.V. 1965. "Partitioning and tearing systems of equations." *SIAM J. Numer. Anal. Ser. B*. vol. 2, no. 2, pp. 345-365.

van Dijk, J. and Breedveld, P.C. 1991. "Simulation of system models containing zero-order causal paths —II. Numerical implications of class 1 zero-order causal paths." *J. Franklin Institute* vol. 328, no. 5/6, pp. 981-1004.

WOLFGANG BORUTZKY is a Professor of Computer Science at Cologne Polytechnic, Germany. He received his diploma degree in Mathematics in 1979, and his doctoral degree in Mechanical Engineering in 1985, both from the Technical University of Braunschweig, Germany. In 1993 he was a visiting professor at Twente University in Enschede, The Netherlands for two months, and in 1995 he spent his sabbatical leave at the University of Arizona in Tucson, Arizona, USA. Since 1979 his main scientific interests have been physical system modeling and simulation methodologies for multidisciplinary engineering systems, as well as integrated circuits, and the design of advanced simulation environments. More recently his interests also covered algorithms and the design of software for parallel continuous system simulation on distributed memory machines. Dr. Borutzky has authored more than thirty scientific publications and has been a reviewer for several scientific journals. He is a member of SCS, of ASIM, the Association for Simulation of Germany, Austria and Switzerland, and of the IMACS Technical Committee 16 on Bond Graph Modeling and Simulation. Since 1990 he has served in several international scientific multiconferences on Modeling and Simulation as IPC member, as Chairman or Program Chairman of a subconference, as session organizer, and as panelist.

FRANÇOIS E. CELLIER received his BS degree in Electrical Engineering from the Swiss Federal Institute of Technology (ETH), Zurich in 1972, his MS degree in Automatic Control in 1973, and his PhD degree in Technical Sciences in 1979, all from the same university. Dr. Cellier joined the University of Arizona in 1984 as Associate Professor. His main scientific interests concern modeling and simulation methodologies, and the design of advanced software systems for simulation, computer-aided modeling, and computer-aided design. Dr. Cellier has authored or co-authored more than 100 technical publications, and he has edited four books. He recently published his first textbook on Continuous System Modeling (Springer-Verlag New York, 1991). He served as General Chairman or Program Chairman of many international conferences, most recently ICBGM'93 (SCS International Conference on Bond Graph Modeling, San Diego, January 1993), CACSD'94 (IEEE/IFAC Symposium on Computer-Aided Control System Design, Tucson, March 1994), ICQFN'94 (SCS Internationd Conference on Qualitative Information, Fuzzy Techniques, and Neural Networks in Simulation, Barcelona, June 1994), ICBGM'95 (Las Vegas, January 1995), WMC'96 (SCS Western Simulation MultiConference, San Diego, January 1996), ICQFN'96 (Budapest, June 1996), WMC'97 (Phoenix, January 1997). He is Associate Editor of several simulation related journals including SIMULATION and the TRANSACTIONS of the SCS, and he served as vice-chairman on two committees for standardization of simulation and modeling software.