BOND GRAPH MODELLING OF PHYSICAL SYSTEMS

A Dissertation

submitted to the Faculty of Engineering

of Glasgow University

in partial fulfilment of the requirements

for the degree of

Doctor of Philosophy

By

Lorcan Stuart Peter Stillwell Smith

May 1993

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

Abstract

This thesis describes new methods for creating and analysing bond graph models of continuous physical systems.

The concept of a core model representation is central to this research, since it is shown that the need to generate and maintain a range of models discourages the widespread use of modelling. Mathematical models appropriate to specific applications are not, in general, sufficiently comprehensive to be used as the core model representation, whereas all the models of interest for analysis and simulation may be derived from a bond graph model. Hierarchical model representations are shown to be an aid to reducing complexity, and thus the bond graph methodologies, which are developed, fully support hierarchical models.

A new bond graph algorithm for identifying and solving algebraic loops is described, and extended to provide a steady-state model of the system. The new algorithm is shown to systematically create a differential algebraic equation (DAE) model of the system.

Bond graph causality is shown to be a powerful analytical concept, but classical causal propagation algorithms have limitations which are discussed. These limitations are overcome by a novel computable causality approach, and its bicausal bond graph representation. The computable causality algorithm is used for resolving algebraic loops, and handling of modulations. The new concepts of unilateral bonds and bicausal bond graphs generalise the classical causality notation to permit physically unrealisable (but computationally useful) bond graph causalities. The computable causality algorithm provides a systematic method for deriving generalised state equation (or DAE) mathematical models from bicausal bond graphs.

Practical applications of the new bond graph techniques are demonstrated through the analysis of four real physical systems as case studies. The implementation and operation of a DOS-based tool which uses bond graphs as the core model representation is described.

Contents

Acknowledgements

The author would like to express his gratitude to his supervisor, Professor P.J. Gawthrop, for his continued help, advice and encouragement during the course of the work reported in this thesis, and to the EDRC and the Department of Mechanical Engineering of Glasgow University for the facilities provided.

The author is indebted to the staff of Eurotherm Controls Ltd, Fisons Instruments, and BICC plc for their co-operation and help in supporting this work. Particular thanks is given to Dr. G. Turnbull, Dr. L. Fenney, Mr. D. Garner, Mr. R. Rahmani-Torkaman, and Mrs S. Spearing for their help and encouragement.

CHAPTER 1 INTRODUCTION

1.1. Introduction

This chapter gives a general overview of the thesis and the concepts of modelling dynamic systems on which the research is based. The chapter breaks down into the four following sections:

- Scope and objectives
- Motivation
- Contribution of this thesis
- Overview

1.2. Scope and objectives

The main aim of this thesis is to support the use of modelling as a useful and knowledge-enhancing exercise, and to propose improved modelling methodologies. As a result, the thesis is concerned with separating out the model development process from the functions for which the model is developed. A secondary aim of the thesis is to produce a modelling tool which can systematically produce a wide variety of derived mathematical models from a given core model description. The major emphasis is on modelling of continuous physical systems, but it is recognised that there are few `real world' systems which can be modelled exclusively in this manner, and thus the integration with discrete event models is also discussed.

Bond graphs are evaluated and, because of their unique properties, used thereafter as the notation for the core model description. Hierarchical model representations are shown to be an aid to reducing complexity, and thus the bond graph methodologies, which are developed, fully support hierarchical models.

1.3. Motivation

System models are normally constructed in order to solve a problem or, at least to test a proposed solution to a problem. A systems analysis view of modelling has been proposed by Schmidt[1], in which modelling is shown to be a significant part of the systems analysis process:

a) problem identification,

b) specification of objectives,

c) definition of the system,

d) model formulation,

e) model verification and validation,

f) model implementation,

g) model use,

h) solution identification,

i) solution implementation,

j) model revalidation.

In his paper[1], Schmidt acknowledges that not all problems warrant all these steps, whereas others may require several iterations between steps. For some problems a simple mental model of the system is sufficient to resolve the problem, while other more difficult problems may best be solved by more detailed modelling, but the time or skills may not be available for this.

This paper also categorises models into two types - those whose purpose is *descriptive*, and those which are *prescriptive*. Descriptive models have the function of aiding understanding, or are developed for communication of concepts. Common formats for descriptive models are engineering documentation, including drawings, and scale models.

Prescriptive models are used to recommend a course of action, since they permit predictions of the real system behaviour to be made. Typical model formats to achieve this end are simulation models, and those used for experimentation and parameter optimisation.

Simulation models themselves have a variety of uses, not least of which are education and training. Mathematical models suited to specialised analysis tools may also be included in this category. An important function for mathematical models is control design, for which a large variety of tools are available - frequency domain analysis, stability and eigenvalue analysis all depend on different formulations of the system model.

Paynter and Shoureshi[2] make a similar distinction between simple *exploratory, strategic* models and detailed *predictive, tactical* models. In this case, however, the strategic models may be simplified mathematical models. It is evident, therefore, that not only do models vary in format, according to the application, but also in the required complexity.

In the field of system modelling, it is generally accepted that one must define the application of the model before its required form, and level of detail, can be determined. This approach discourages re-use of models and can result in inconsistencies, when different models of the same system are developed for, say, analysis or simulation. This thesis proposes a different view of system modelling; as a sequence of transformations from the physical system through a sequence of representations to obtain an *appropriate system model*[3] as shown in figure 1.1.

- Physical system
- $Transformation_1$ => $Representation_1$
- $Transformation_2$ => $Representation_2$
- ...
- $Transformation_n$ => *Model*

Figure 1.1 Transformation view of system modelling

The fundamental difference in the approach described in this thesis, is that the same *core model representation* is used for deriving different representations appropriate to a variety of different applications. The range of uses envisaged covers control design, process design, simulation and system understanding. The *derived representations* must clearly be *appropriate* to the use of the model, and are considered as different views of the physical

system. Some possible representations are: a state space equation, a frequency response of a linear transfer function, an inverse system transfer function, a human readable equation or machine readable (possibly non-linear) simulation code.

The first transformation is thus to the core model representation, $Representation_1$, and will always require some degree of skilled input, and should not be automated. In order to simplify this transformation, it is important that the core model be `close' in some sense to the physical system, and map directly onto the structure of that system. Equally, $Representation_1$ should contain enough information to generate all the required models. For these reasons, and others which will be discussed, energy bond graphs have been chosen as $Representation_1$, in the context of continuous system modelling.

The intermediate transformations probably can, and certainly should, be completely automated. An aim of this research is to provide tools for accomplishing such transformations from the core (bond graph) representation.

1.3.1. A motivational example

This example is included to show how a modelling tool must offer a range of functions in order to meet a variety of application requirements. The example used in this discussion is an industrial process for extruding polymer sheathing onto wire for manufacturing electrical cables (figure 1.2). This process is analysed in greater detail in Chapter 6 of this thesis, but it is useful to consider here to understand the problems in modelling such a process.

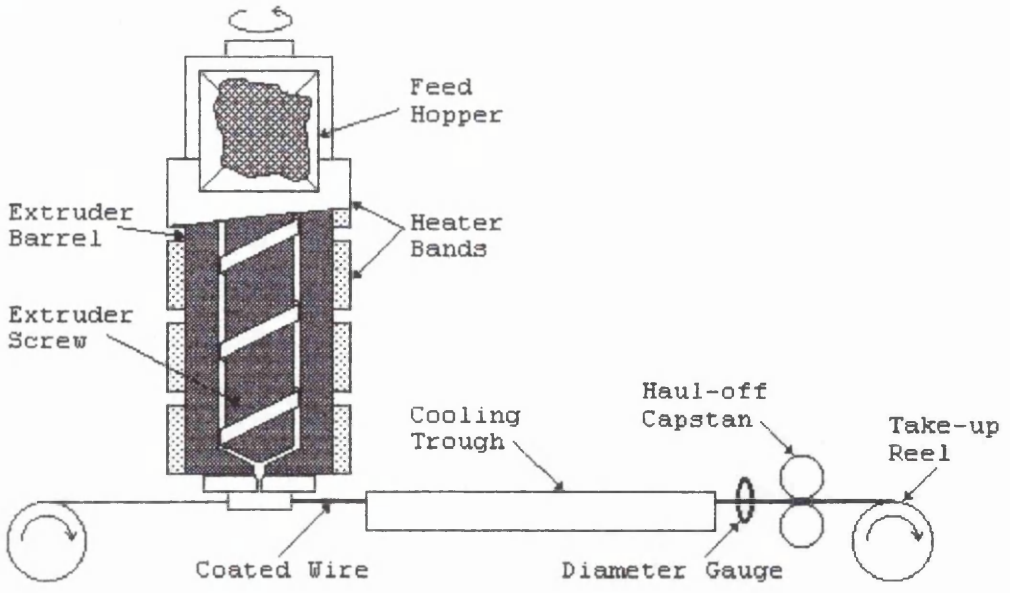


Figure 1.2 Extrusion system with section through extruder

For the moment, it is sufficient to know that a plasticating extruder is merely a large metal barrel in which a screw rotates in order to meter out quantities of molten polymer through a die. The screw is typically driven by an electric (D.C.) motor which provides the mechanical energy necessary to overcome the shear friction against the polymer and generate sufficient hydraulic pressure to force the polymer through a die. The polymer is initially heated by electrical heater bands round the barrel, but when it is being extruded at normal production rates, sufficient work heat is generated by the shear friction of the screw forcing the melt down the barrel and out of the die. Finally there are measurement systems on the extruder - measuring temperature and pressure - and also on the final product - measuring the outer diameter of the cable after it has been hauled through a cooling trough. This last measurement system is of greatest interest as it gives the main measure of product quality, although the measurement is subject to a long transport delay due to the cooling process.

Figure 1.2 is, in fact, our first model of the process and is well suited to the purpose of describing the process at an overview level. It is graphical and encapsulates the description in a very concise and understandable manner, but it also has some major disadvantages. In the first place, the drawing does not explicitly show all the sub-systems - the mechanical translation of the polymer through the barrel, and the associated hydraulics are

assumed. The model is not complete and had to be supplemented by the written description in the above paragraph. Most important to the engineer, however, is the fact that even the combination of the figure and the written description is insufficient for any analysis or prediction of the performance of the process. The engineer needs some form of mathematical model to achieve these ends.

If our process engineer's purpose for modelling is just to achieve a relationship between the outer diameter of the coated cable and the screw speed or the haul-off speed, then he must find the steady-state gain of the process. This is achieved by deriving a mass balance equation for the polymer flow into and out of the die. Intuitively one is not surprised to find that this transfer function shows that the diameter depends on the internal dimensions of the barrel and the screw, and on the ratio (screw speed)/(haul-off speed).

This transfer function is very useful if the engineer wants to judge the rate at which he can produce a given diameter of cable, but it has limited use if he wishes to design an automatic control system for this parameter. The problem is that this mathematical model only gives the steady state gain of the process, whereas the dynamic transfer function is a more useful model for control design. In practice, some of the variables are often ignored at this stage in order to simplify the modelling exercise, but at the expense of reducing its usefulness in achieving an overall understanding of the process. A typical simplification is based on the fact that the temperature of the barrel wall is closely controlled by a multi-zone automatic control system. It is assumed that the melt temperature is approximately constant, or, at least, varies slowly with respect to the achievable changes in screw speed or line speed. An important feature lost by this assumption is the ability to predict the response of the diameter to large scale changes in screw speed when the process ramps up to full speed and the generation of work heat changes rapidly.

It is important to be able to model the process behaviour during ramp-up to production speed (and ramp-down), because the diameter variation caused by this disturbance can mean that significant amounts of cable have to be scrapped. For this analysis, a

simulation proves an invaluable tool, and, since the entire process forms rather a large model it is desirable to neglect some of the faster dynamics in order to run the simulation faster. In this case we require a mixed model which includes the dynamics of the slower sub-systems, and static models of the fast sub-systems.

The above discussion has shown that three different modelling requirements have resulted in three different mathematical models to provide each specific functionality. Modellers are not unused to this sort of problem, but it may explain why the benefits of process modelling are not as widely exploited in industry as they might be. The problem in industry is that the processes are subject to continuous change as market demands, financial constraints, and technology all change. The process engineer often cannot afford the time to generate more than the static model let alone keep several models up to date.

There is, therefore, a very strong incentive to provide one core model representation from which the variety of mathematical models described in the preceding paragraphs can automatically be generated.

It has been pointed out[4,5] that the dominance of simulation tools as a means of predicting system behaviour, has led to models being too tightly bound in to this one particular type of experiment. Modern simulation tool design methodologies reflect structuring trends in software engineering, by segregating the functionalities of model building and experiment building. Breitenecker has used the term `method' to describe generic experiments, while retaining `experiment' to describe the performance of a specific method on a specific model. It may then be possible to achieve the desirable goal of separate and orthogonal databases of models, methods and experiments, which would then permit models to be developed without knowledge of the experiments and vice versa.

It can be seen that this goal can best be achieved by adopting the core model approach proposed in the previous section, with appropriate transformations as the front end for each analysis/simulation tool. A library of derived model variants appropriate to each tool is not only inefficient, but also

ultimately unmaintainable. Since individual analysis/simulation tools are unlikely to have front ends to cope with an arbitrary core model description format, the modelling tool must be extendible to provide derived models in existing formats. Similarly, the modelling tool must be able to generate all derived sub-models, in a format acceptable to the user.

In general we can see that, in the non-academic world at least, modelling is only performed if the risk and cost of failure (of the real design) outweighs the cost of building models and running appropriate experiments. The way forward is to provide tools which support and accelerate the model building and experimenting processes.

1.4. Contribution of this thesis

The contribution of new work to the body of bond graph theory is described in detail in chapters 4 and 5 of this thesis, while new practice is detailed in Chapters 6 and 7.

The major theoretical advances are:

- a new algorithm for completing causal assignment of models with algebraic loops (chapter 4)
- an extended bond graph notation for deriving non-standard mathematical models (chapter 5)

Some applications are given in case studies of modelling physical systems, using these techniques (chapter 6):

- a plasticating extruder
- a drum boiler - turbine
- a telephone anti-sidetone circuit
- a production carpet cutter

The implementation of a bond graph modelling tool which utilises some of the new concepts is described in chapter 7

1.5. Overview

The remainder of this thesis is divided into three main parts, covering first a literature survey (chapters 2 and 3), followed by details of the novel contribution of this research (chapters 4 and 5), and finally use and implementation of tools resulting from this work (chapters 6 and 7). These three parts are subdivided as follows:

Chapter 2 Representation of elementary systems

The decomposition of a system into a structure of elements representing its static and dynamic behaviour is reviewed, first using classical dynamical analysis and then using the energy bond graph notation. Bond graphs are shown to provide a unified model representation for physical systems covering all energy domains. The classic bond graph causality algorithm is shown to provide a systematic means for deriving mathematical models of the system. Finally, several bond graph modelling tools are described.

Chapter 3 Hierarchical modelling using bond graphs

Bond graphs are shown to be well suited as a core model representation, using different causal initiations to achieve the different derived representations. Multi-port and multi-bond representations are discussed as candidates for hierarchical core model representations. The acausal word bond graph is shown to be most flexible for representing hierarchically structured systems.

Chapter 4 Causal augmentation of bond graphs with algebraic loops

The causes of algebraic loops are discussed together with limitations of existing methods for solving such loops. This chapter describes a new algorithm for identifying and solving algebraic loops, and also extends the use of this algorithm for steady-state analysis. The new algorithm is shown to systematically create a differential algebraic equation (DAE) model of the system.

Chapter 5 Bicausal bond graphs and unilateral bonds

This chapter identifies limitations of conventional causality algorithms, and describes a novel computable causality approach, and its bicausal bond graph representation. The computable causality algorithm is used for resolving algebraic loops, and handling of modulations. The new concepts of unilateral bonds and bicausal bond graphs generalise the classical causality notation to permit physically unrealisable (but computationally useful) bond graph causalities; deriving inverse system models, for example. These concepts are also expressed in terms of generalised state equation (or DAE) mathematical models

Chapter 6 Case studies using bond graph models

Bond graph models of four real physical systems are developed using the concepts and methodologies outlined in the previous chapters. The four physical systems modelled are:

An industrial process - a plasticating extruder

A process engineering system - a drum boiler-turbine

An electrical network - a telephone anti-sidetone circuit

A mechanical process - a production carpet cutter

Chapter 7 Implementation of a bond graph modelling tool

The implementation and operation of a DOS-based tool using bond graphs as the core model representation is discussed. This tool has been used to automatically generate mathematical models for some of the case studies described in chapter 6.

The applicability of object-oriented techniques, used to implement the modelling tool, is compared to the use of bond graphs in the context of hierarchical modelling.

Bond graph modelling and causality algorithms introduced in this thesis are detailed as they have been implemented in the modelling tool.

Chapter 8 Conclusion

This chapter concludes the thesis, and suggests areas where related research could be productive.

The main conclusion drawn is that the classical bond graph causality notation is too concise to provide all the information to systematically derive all system models. The new computable causality algorithm, together with its graphical notation (the unilateral bond) resolves this limitation and thereby extends the scope of bond graph modelling techniques. The new algorithm has been shown to be useful in the analysis of inverse system models, and also permits the new graphical method for resolving algebraic loops to generate the minimum number of algebraic loops.

This thesis has limited the evaluation of bicausal bond graphs to those graphs where the unilateral bonds only appear in the junction structure. Further useful work can be done by evaluating the use of unilateral bonds to describe changes in constitutive equations of dissipators and energy storage elements, perhaps providing the basis of a systematic bond graph approach to fault detection.

CHAPTER 2 REPRESENTATION OF ELEMENTARY SYSTEMS

2.1. Introduction

The aim in this chapter is to describe the background to bond graph theory, and the present state of research in this area. This is viewed from the context of a generalised approach to modelling, which unifies physical systems of all energy domains. A structured approach is to analyse the system in terms of its constituent parts, within a defined system boundary (a frame). This process requires the modeller to abstract the model to a structure of interacting sub-models in a hierarchical manner until at the lowest level each sub-model consists of a structure of elementary component behaviours (expressed as constitutive relations). Before discussing methodologies for handling hierarchical systems, it is useful to understand the problems of modelling at the lowest sub-model level.

In this chapter, section 2.2 describes a suitable set of structural and constitutive relations for the primitive elements, while section 2.3 describes how energy bond graphs provide a powerful notation for representing models using these concepts. Section 2.5 gives examples of bond graphs covering a variety of energy domains. Having captured this representation of the system, it is then necessary to transform this to a derived mathematical model suitable for analysis or simulation. Section 2.5 shows how various causal augmentations of bond graphs permit this to be systematically achieved, whilst providing deeper insights into the model and system. Section 2.6 describes the use of multi-port components and hierarchical models, and section 2.7 applies pseudo bond graphs to solve modelling problems for non-energy systems. Section 2.8 reviews some bond graph tools, and the chapter is summarised in section 2.9.

2.2. Structure and constitutive relations

The previous chapter has indicated that the core model representation should include both the static and dynamic

characteristics of the process. It should not be a set of mathematical equations, but should instead have a close mapping to the physical process, permitting the model to be extended to track modifications to this process. A natural way to achieve this aim is to subdivide the model into a set of standard elements and interconnect them in a structure appropriate to the process. This separation of structure and component behaviour is essential in order to permit the model to be interpreted easily by both humans and computers, thus facilitating modification in step with that of the process.

A popular method of modelling is to construct an electrical analogue of the actual process. A brief analysis of why this is the case may prove useful. Electrical schematics are quite concise, and unambiguously describe the structure (wiring) relating a set of idealised components - analysts of this energy domain are fortunate in having components that are close to ideal over a wide operating range. The schematic has the advantage of being easily understood by (trained) humans, and also, more recently, by CAD software. Unfortunately, the mapping between the electrical analogue and the process is not always one to one, so occasionally some confusion may arise. A more direct mapping also permits the modeller to evolve the model more easily to achieve a closer match to the real process. Another disadvantage of this circuit-based modelling approach is that it does not offer any direct insights into the workings of the real process, since it is purely an analogue.

Modelling using electrical analogues also tends to obscure the fact that, for processes covering multiple energy domains, the unifying variable is in fact energy. Much has been written by previous researchers[8,9,10] in this field, exploiting this unification, which can only be summarised here. However, modelling energy transfers does provide a very useful focus for this discussion of system representations. In practice, this turns out not to be a significant limitation, as most of the processes we are interested in modelling - general physical systems, mechanics and industrial processes - involve energy transfers. Section 2.6 will show how the same techniques can be applied to developing models of processes where energy is not the exchange variable.

2.2.1. Energy transfer models

At this point it is necessary to give an overview of the basic concepts of system modelling based on energy as the variable manipulated by the system. For a more detailed exposition, the reader is referred to several excellent texts[9,10,11] on this specific subject.

Choosing energy as the exchange variable for a model, leads naturally to the use of two co-variables in each energy domain, which are conventionally called effort (e) and flow (f), where

$$\text{energy } E = \int e.f \, dt \tag{2.1}$$

It is worth commenting that some authors feel this nomenclature is unfortunate, in that the concept of across and through variables, instead of effort and flow, is more consistent when dealing with mixed energy domains including the mechanical domain. Across variables (transvariables) are spatially-extensive and are often described[8] as those requiring a 2-point measurement. Through variables (pervariables) are spatially-intensive and imply that the variable passes through the measurement instrument. This way of classifying variables results in voltage, pressure and velocity being grouped as across variables, while current, flow rate and force are the corresponding through variables.

In the effort-flow classification, voltage, pressure and force are effort variables, while current, flow rate and velocity are the corresponding flow variables. The consequence of this difference is that mechanical systems described using the effort-flow notation are duals of those using across-through notation. Each approach shows some inconsistencies, but since the effort-flow classification is most widely used in bond graph theory, this is adopted henceforth in this thesis.

Energy is exchanged through so-called ports on each element, where each port represents a single distinct energy interface. The energy model has four basic types of elements:

a) **Energy sources.** The system inputs which are a convenient way of defining a boundary on the modelled system, for determining its reaction to effort or flow stimuli.

b) **Energy stores.** These elements accumulate either the effort or flow variable and are described as effort or flow stores, respectively. This accumulation (integration) of either effort or flow gives the system a state, and thus endows the system with dynamics.

c) **Energy dissipators.** Elements which dump energy out of the system into its environment, and which, for non-thermodynamic models, provide a convenient termination boundary to the model. This irreversible conversion of energy to the thermal domain results in non-dynamic elements.

d) **Energy transfer elements.** These elements conserve energy, merely routing it through the model, between any other model elements. In some energy domains these elements are well-defined (e.g. parallel connections in electrical systems), while in others they are more abstract (common force points in mechanical systems). Included in this group of elements are couplers which neither store, nor dissipate energy, but transform the effort and flow variables without energy loss.

It is recognised that it is also important to have system outputs (via sensors), but for analysis purposes outputs are signals and do not exchange energy. A sensor output may also exhibit dynamics, which may be either inherent or due to its location or relationship to the measured variable. Outputs will be dealt with in detail in subsequent discussion of hierarchical systems and inverse system models.

The behaviour of a specific element is described by a physical law which is called its *constitutive relation*, and the form of this relationship determines which of the above groupings is appropriate to a given element. Specific constitutive relations will be discussed further in section 2.1.3, after energy transfer elements have been discussed in more detail.

2.2.2. Model structure

The energy transfer elements actually represent the model structure, and are called multi-ports, indicating that they have

two or more ports for transferring energy. The constitutive relation which is common to these elements is that the sum of all the energy flows into the junction is zero,

$$\text{i.e. } e_1.f_1 + e_2.f_2 + \ldots + e_n.f_n = 0 \qquad (2.2)$$

where subscripts 1, 2 .. n indicate the ports through which energy is flowing into the element. Note that a sign convention must be chosen which is consistent; for example, all energy flows being measured **into** the element.

There are four basic elements within this category, two of which maintain one of the variables constant through the element, and two which perform a transformation.

a) Junctions

The first type is termed a junction element where either effort or flow is fixed and the co-variables must sum to zero. Electrical engineers will recognise this as a more general formulation of Kirchoff's Laws. There are two such laws for each energy domain, since either the effort or the flow may be fixed at a specific junction. Thus at an effort junction (also termed a parallel junction from its electrical domain equivalent) the following relations must hold:

$$e_1 = e_2 = \ldots = e_n \qquad (2.3)$$

$$\text{and } f_1 + f_2 + \ldots + f_n = 0 \qquad (2.4)$$

Conversely, for a flow (series) junction the flow is fixed for each path into or out of the junction while the efforts must sum to zero, i.e.

$$f_1 = f_2 = \ldots = f_n \qquad (2.5)$$

$$\text{and } e_1 + e_2 + \ldots + e_n = 0 \qquad (2.6)$$

The direction of energy flow is generally assumed to be from input sources and into stores and dissipators. With a complex junction

structure it is sometimes not obvious which way energy may be flowing, so structural conventions must be able to unambiguously represent the chosen sign of the energy flows.

b) Transformers and gyrators

If the energy transfer element also transforms one of the effort or flow variables then the co-variable must also be transformed such that the energy conservation relationship (2.2) is still valid. The most widely used elements of this type have just two ports, so these will be described here, although the description can be applied more generally to `n' ports.

There are two elements of this type - the transformer and the gyrator. A 2-port transformer has a relationship where the efforts on the two ports are constrained by the relationship:

$$e_2 = ke_1 \tag{2.7}$$

where the transformer ratio, k, is either a constant or may be dependant on some other system variable, resulting in a modulated transformer. For energy conservation to hold at any instant

$$e_1 f_1 = -e_2 f_2$$

so $$f_1 = -kf_2 \tag{2.8}$$

The direction of power flows are normally defined such that one port is an input and the other an output, resulting in the transformer ratio being positive for both effort and flow relations.

Typical physical examples of transformer elements are a frictionless lever in the mechanical domain, or a two port transformer in the electrical domain. The reason that the latter example only transforms a.c. signals will be used to show how energy bond graphs can provide deeper insight into system behaviour. This restriction on the electrical transformer highlights one area where using an electrical analogue for a mechanical system is inexact.

The gyrator constitutive relation occurs when the relation is constrained by:

$$f_2 = ge_1 \tag{2.9}$$

where g is sometimes referred to as the mutual conductance.

Substituting (2.9) back into the energy conservative relation (2.2) for a 2-port, gives the complementary form of the gyrator constitutive relation:

$$f_1 = -ge_2 \tag{2.10}$$

As for the transformer ratio, the mutual conductance, g, may be either a constant or dependant on some other system variable as long as both relations are simultaneously true.

Physical instances of gyrators are less easily recognised than transformers, as they occur most often when transformation from one energy domain to another is modelled. A typical example is the fixed field d.c. motor where the back e.m.f. generated by the armature rotation is proportionally related to the shaft speed, by the motor gyrator constant, and the input current is related to the load torque by the same constant. If the field current is derived by placing the field winding in series with the armature winding, then the mutual conductance becomes a function of this current resulting in a modulated gyrator.

2.2.3. Constitutive relationships of energy nodes

Energy sources, stores and dissipators have been identified as the basic elements which may be used to emulate the range of system behaviours required for a comprehensive energy model. A fuller understanding of these elements can be gained by studying their constitutive relations. These constitutive properties of an element will generally be expressed as an equation relating the effort and flow variables, although they could equally be described by graphs. It is important that any modelling technique adopted must be able to handle constitutive relations which are non-linear or time-dependent.

a) Energy sources

The system inputs can be either effort sources or flow sources, where the type of source defines the variable controlled by the source, which, for an ideal source, is independent of the co-variable (figure 2.1).

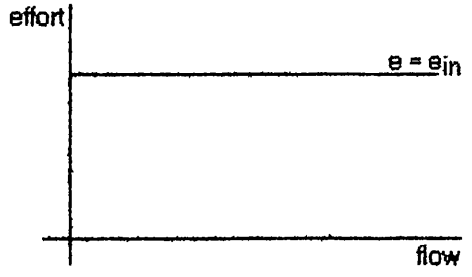


Figure 2.1 Constitutive relation for ideal effort source

The value of the co-variable is defined by the system which the source supplies. Thus using an electrical example once again, a battery is an effort source and if the system consists of a resistor across the battery terminals, then this resistor determines the current (flow) from the battery. Sources can also be modulated by another system variable, as is often the case with control systems, and in the electrical domain, an amplifier providing a low impedance voltage output may be modelled as a modulated effort source.

b) Energy stores

Energy stores are a little more complicated, but again there are two basic types - those that accumulate effort and those that accumulate flow* .

Dealing first with effort accumulating stores, the constitutive relation has the form:

$$f = \phi(p) \qquad (2.11)$$

* There is the possibility of confusion here. Some authors use `effort store' to refer to a store with effort output, that is a flow accumulating store, and `flow store' to refer to a store with flow output, that is an effort accumulating store. In this thesis, the opposite convention is used whenever the abbreviated form is used, that is an `effort store' refers to an effort accumulating store.

where $\phi(p)$ is a (possibly non-linear) function of the *integrated effort* or *generalised momentum variable* p, given by:

$$p = \int e \, dt \tag{2.12}$$

In the linear case, equation 2.11 can be rewritten as:

$$f = \frac{p}{I} \tag{2.13}$$

where the proportional constant I is called the inertance.

Equation (2.12) is shown in integral form as this best indicates the storage mechanism and is physically realisable. However, equations 2.12 and 2.11 can be rewritten in derivative form as:

$$e = \frac{dp}{dt} \tag{2.14}$$

$$p = \phi^{-1}(f) \tag{2.15}$$

where $\phi^{-1}(p)$ is the inverse of ϕ.

In the linear case, the derivative form can be used to evaluate the total stored energy,

$$\text{since } E = \int e.f \, dt = I \int f \, df = I \frac{f^2}{2} \tag{2.16}$$

Example An example of an effort store from the mechanical domain occurs when the effort variable, force, is applied for a time to a mass, resulting in a change in the flow variable, velocity.

$$\text{i.e. velocity} = \frac{1}{\text{mass}} \int \text{force} \, dt$$

The energy imparted to the mass has been stored as kinetic energy and from equation (2.12) accumulated energy is given by:

$$E = \frac{\text{mass}}{2} \text{velocity}^2$$

In a similar way, the flow accumulating store has a general constitutive relation of the form:

$$e = \phi(q) \tag{2.17}$$

where $\phi(q)$ is a (possibly non-linear) function of the *integrated flow* or *generalised displacement variable* q, given by:

$$q = \int f \, dt \tag{2.18}$$

In the linear case, equation 2.17 can be rewritten as:

$$e = \frac{q}{C} \tag{2.19}$$

where the proportional constant C is known as the capacitance.

Example An easily visualised example of a flow store is a uniform tank filled with incompressible fluid from an independent flow source. The flow variable in this case is the volume flow rate of fluid into the tank, and the effort variable is the resulting pressure at the bottom of the tank. Simple hydraulics indicate that this pressure is given by:

$$\text{Pressure} = \frac{\text{volume.density.g}}{\text{Area}}$$

$$= \frac{\text{density.g}}{\text{Area}} \int \text{VolumeFlowRate} \, dt$$

Hence the capacitance, C, is Area/(density.g) and in this case, the energy is stored as potential energy in the head of water in the tank.

c) Energy dissipators

Energy dissipators are not divided into effort or flow types because their constitutive relations can generally be expressed in either form,

$$e = \phi(f); \; f = \phi^{-1}(e) \tag{2.20}$$

or, in the linear case,

$$e = R.f \quad \text{or} \quad f = e/R \tag{2.21}$$

These equations (2.22) are seen to be general forms of Ohms law in the electrical engineering domain, where R represents an electrical resistance, and the energy dissipated in the linear case may be expressed as:

$$E = \int f^2.R \, dt = \int e^2/R \, dt \tag{2.22}$$

Mechanical and hydraulic dissipators are not necessarily linear, however, and thus their constitutive relations may be more easily calculated when expressed in one particular form. Dissipators in these domains exert forces which always oppose the direction of motion imposed upon them and vary according to a variety of laws. The effort (pressure drop) generated by incompressible flow through an orifice is typically given by:

$$e = R.f|f|$$

thus giving two possible values of flow if this expressed as a function of the effort variable.

As a final comment on dissipators, it should be realised that when modelling thermodynamic systems one is often specifically interested in calculating the dissipation of thermal energy into the environment, and so the environment itself contributes to the system variables. Thermodynamic systems will be dealt with in more detail in section 2.2.1

Due to the conflicting variable names used in each energy domain, and since the point of using energy as the manipulated variable is to unify the approach to all these domains, the designations used in this section will be used throughout the remainder of the text. The correspondence of these variables to individual energy domains is shown in table 2.1.

Domain	effort	e	flow	f	momentum	p	displacement	q
electric	e.m.f.	e	current	i	lines	λ	charge	q
magnetic	m.m.f.	M	flux rate		-	-	flux	ϕ
hydraulic	pressure	P	volume flow rate		pressure momentum	p	volume	V
mechanics translation	force	F	velocity	V	momentum	P	displacement	x
mechanics rotation	torque	T	angular velocity	ω	angular momentum		angle	α
thermo-dynamics	temperature	T	entropy flow rate		-	-	entropy	S

Table 2.1 Effort and flow variables for each energy domain

To summarise this brief overview of modelling systems as energy manipulators, we have identified four basic element types which can be differentiated by the form of their constitutive relations. Elements which conserve energy and distribute it between other elements are seen to define the structure of the system. The remaining elements have constitutive relations which either put energy into the system (sources), remove energy from the system (dissipators), or store either potential or kinetic energy (stores). These energy stores accumulate all the history of the system and thus can be used to derive state variables for dynamical models.

2.3. Energy bond graph models

The *bond graph* notation is a graphical language designed specifically for the description of processes which manipulate energy. In consequence, the language includes elements which model all the requirements analysed in the preceding discussion on structure and constitutive relations. A graphical notation is necessary in order to provide a concise description of the entire process at a higher level of abstraction than the equations describing the energy transfers between elements. In addition, bond graphs also highlight the structure of the model, making the mapping between the model and the system more intuitive. This application of bond graphs to show the system structure is utilised to describe systems at a higher level of abstraction using the *word bond graph*, where the elements are (potentially hierarchical) sub-

models.

If it were just the case that bond graphs provide `the acceptable face of energy equations' to improve their palatability to engineers, the notation would have less value than it actually provides. It is hoped that the following discussion will show how bond graphs not only represent the process in a form with which the user can easily interact, but also help to improve understanding of process fundamentals and yet permit unambiguous interpretation of the graph by software for transformation to a variety of derived models.

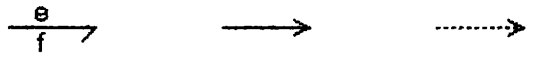
The remainder of this section describes bond graph syntax, with special emphasis on the interpretation of computational causality. Finally, the use of multi-port elements is described with, hopefully, a fresh view on their application.

2.3.1. Energy bonds

Bond graphs have the effect of shifting the users attention away from the element which manipulates energy and towards its interaction with the rest of the system in which it exists. The energy bond carries all the information about this interaction, which notionally occurs through a `port' on the element.

The bond is represented as a half arrow (figure 2.2a) indicating the (supposed) direction of energy flow, between the ports to which it is attached. In practice, the direction of the half arrow cannot be arbitrarily assigned, and thus a convention has been developed[6] to ensure this assignment is consistent with the sign convention.

Another convention has been established[7] (although it is not exclusively employed in bond graph literature) whereby horizontal bonds are drawn with the half arrow downwards and vertical bonds are drawn with the half arrow on the right hand side. The bond may be annotated by symbols representing the flow (on the side of the half arrow) and the effort (on the other side of the bond) subscripted with the bond identification, which is typically the same as the identification of the attached energy node.



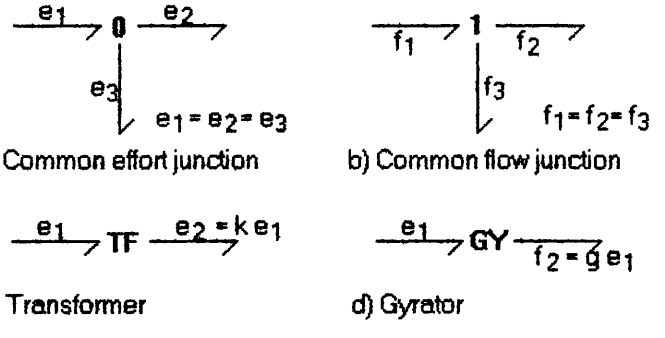
a) Energy bond b) Activated bond c) Modulation

Figure 2.2 Representation of bonds and signals

An energy transfer is implicit in every bond, so an equivalent symbol is required to indicate the transfer of zero energy signals (or information). The symbol for a signal is the full arrow (figure 2.2b) borrowed from block diagram notation. The signal may convey either an effort or a flow, or alternatively, the value of a state variable. By convention, a signal pointing towards an energy node implies that the constitutive relation of that element is modulated by the value conveyed by the signal. A shorthand notation has arisen where a signal directed at a junction implies a combination of a signal modulating the appropriate energy source on that junction, without having any effect on the source junction i.e. a buffered signal. For this reason, signals are also called activated bonds, although these are distinguished from modulating signals (figure 2.2c) in bond graphs given in this text.

2.3.2. Junction structure

The need for the four structural elements provided by bond graphs has been outlined in section 2.1, and these are illustrated in figure 2.3.



a) Common effort junction b) Common flow junction

c) Transformer d) Gyrator

Figure 2.3 Junction structure elements

The (common) effort junction is conventionally called a `0' junction, and has at least two ports, but typically three or more. The constitutive relation of the 0-junction ensures that the effort is identical at each port and that the algebraic sum of the flows

on each port is zero. The (common) flow junction is called a `1' junction and conserves energy by defining the flows on each port to be identical while the efforts sum to zero. Since the 0- and 1-junctions are generalisations of parallel and series electrical junctions, a convention has arisen labelling these as `p' and `s' junctions respectively. Since this is meaningless for mechanical systems and the `0' and `1' convention is most widely used, this thesis uses the latter henceforth.

Transformers are designated by `TF' nodes in bond graphs and are again power conserving although the effort on the output port is scaled by the transformer ratio to the effort on the input port. In section 2.1.2 we noted that the transformer ratio can be modulated by another system variable, which is indicated graphically by directing a signal toward the `TF' node from a node carrying the relevant system variable. An example of a modulated mechanical transformer is shown in figure 2.4 where a rigid bar pivoted at its end converts the translational force F to a torque T with a transformer ratio (l.cos(α)) dependant on the angle of the bar.

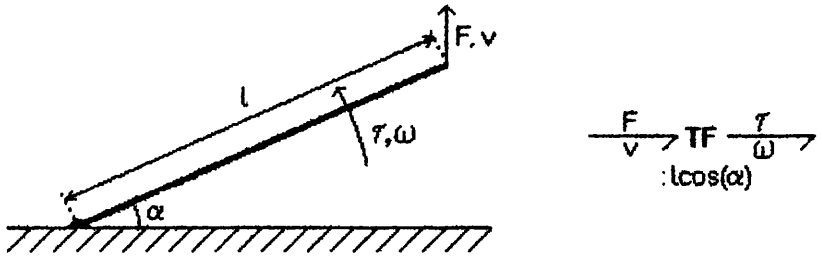


Figure 2.4 Transformation between mechanical domains

$$T = (l.\cos(\alpha)).F$$

and $$(l.\cos(\alpha)).\omega = v$$

Figure 2.4 is also an example of the use of a transformer to convert between energy domains - in this case, between the translational and rotational mechanical domains.

Gyrators (designated `GY') are also energy conserving, but directly relate the input effort to the output flow - they most frequently occur when representing transducers between energy domains. A

further example of this is shown in figure 2.5 where an electrical coil wound on a magnetic core is modelled as a gyrator between the electrical and magnetic energy domains.

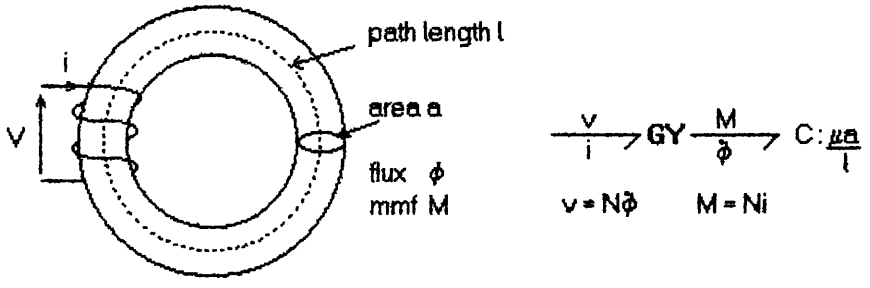


Figure 2.5 Gyration between electrical and magnetic domains

In this case, the coil gyrates electrical effort (e.m.f.) to magnetic flow (rate of change of flux), with a gyrator ratio equal to 1/N, where N is the number of turns in the coil (Faraday's Law). Since the gyrator is energy conserving, the electrical flow is related to the magnetic effort variable (m.m.f.) by the same ratio. Whereas the coil appears to the electrical system to which it is connected to be an effort store, this model indicates that the energy is stored in a flux store in the magnetic domain. The capacitance of the magnetic circuit (normally called the permeance) can be shown to be given by:

$$C = \frac{\mu A}{l} \tag{2.23}$$

Hence the magnetic effort generated by the flow into this capacitance is given by:

$$M = \frac{1}{C} \int \frac{d\phi}{dt}\, dt$$

$$= \frac{1}{C} \int \frac{e}{N}\, dt$$

$$= \frac{1}{\mu AN} \int e\, dt \tag{2.24}$$

Hence:

$$i = \frac{M}{N} = \frac{1}{\mu AN^2} \int e \, dt \qquad (2.25)$$

i.e. the electrical inductance is $\mu AN^2/l$

2.3.3. Energy nodes

In section 2.1.2 we divided energy nodes into three categories: energy sources, stores and dissipators. Table 2.2 shows the bond graph representations for each of these elements and standard (linear) forms for their associated constitutive relations.

Symbol	Element Type	Constitutive Relation
SF⟶	Flow Source	$f = f_{in}$
SE⟶	Effort Source	$e = e_{in}$
⟶C	Flow Store	$e = 1/C \int f \, dt = q/C$
⟶I	Effort Store	$f = 1/I \int e \, dt = p/I$
⟶R	Dissipator	$e = R.f$ or $f = e/R$

Table 2.2 Bond graph elements

Non-linear constitutive relations are of course possible, and may be represented within a bond graph model. Each node is illustrated with one associated energy bond, indicating that these are representations of single port elements. The effort and flow sources are shown supplying energy while for the remaining elements the nominal direction of the energy flow is toward each element.

At this point, it is useful to consider what elements or behaviours these symbols represent in the context of specific energy domains.

a) Electrical elements.

Since this domain has relatively ideal components, their behaviours can be mapped exactly onto those listed in table 2.2. Voltage and current sources are represented by `SE' and `SF, respectively, and these can be modulated by some other system variable to model perfect amplifiers. `C' and `I' energy stores represent capacitors and inductors which store energy either as electric charge or

magnetic flux. Finally, electrical resistors dissipate energy from the system and can be represented by `R' nodes in the bond graph.

b) Magnetic elements.

Magnetomotive force (m.m.f.) can occur as a fixed effort source, when modelling the remanent magnetism in a permanent magnet, or as an effort source when produced by an electric current in a wire. In our discussion of gyrators, in section 2.2.2, it was noted that the magnetic flow (rate of change of flux) is proportional to the voltage across the coil, so a magnetic flow source is created whenever a voltage is applied to an electrical path.

It was also seen that the energy is stored in the magnetic path, due to the accumulation of flux resulting in a magnetic `C' element. There is no equivalent magnetic element to the effort store - `I' node - this will be discussed further at the end of this section. Magnetic circuits can only dissipate energy when the m.m.f. is changing - this is due to the hysteresis loss of a magnetic core, and can be modelled by an `R' node. Eddy current losses can also occur in metal cores, but these are due to a gyration back to the electrical domain.

c) Hydraulic elements.

When dealing with incompressible hydraulics, pumps can be represented by `SE' (pressure sources) or `SF' nodes depending on the type of pump. A tank capacity is readily seen to be an accumulator of flow, and is represented by a `C' node. An ideal pressure source is a large reservoir - effectively an infinite capacitance. The kinetic energy associated with mass flowing through a pipe is the result of accumulated effort and represented by an `I' node.

Energy may be dissipated in two basic ways in hydraulic systems, either due to viscous forces between the fluid and static objects or viscous forces between fluid particles. Both are represented by an `R' node. Laminar flow results in a linear constitutive relationship, but whenever turbulent flow exists this becomes highly non-linear.

d) Mechanical elements.

Although translational and rotational mechanics are deemed to be separate domains, they are dealt with together here as so much terminology is common. Imposed forces and torques are effort sources, the most common constant `SE' node being gravity. Imposed (linear or angular) velocities are also possible, represented by the `SF' node.

Mechanical engineers make the distinction between potential and kinetic energy, according to whether it is stored in a `C' or `I' element respectively. Springs are flow stores (`C' nodes), while mass accumulates effort and is represented by an `I' node. Friction dissipates energy from the mechanical system and is represented by an `R' node (often with a non-linear constitutive relation).

e) Thermodynamic elements.

Thermodynamic systems are often analysed using the variables temperature and heat flow rate (dQ/dt), but the latter cannot be used as the flow variable in an energy bond graph, as it is an energy rate variable. One can use heat flow rate in some bond graph representations (called pseudo bond graphs), but then care has to be exercised in interfacing with other energy domains.

Energy bond graphs for thermodynamic systems use entropy flow rate (dS/dt) as the flow variable and absolute temperature (T) as the effort variable, thus satisfying the requirement that the product of effort and flow is instantaneous power. Effort sources are therefore models of elements which can force the temperature at one point in the system - a standard `SE' input to thermodynamic systems is the ambient temperature.

Although entropy flow sources do exist they rely on inputs from other energy domains - cf. flow sources in the magnetic domain. In this case, energy lost through a dissipator in the other energy domain is conserved in the thermodynamic domain and emerges as a defined entropy flow rate. Since this is such a common mechanism for sourcing entropy flows, bond graphers have added the `RS' node to the terminology. The constitutive relation of the `RS' node is

energy conservative as illustrated in figure 2.6

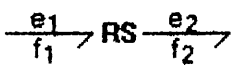


Figure 2.6 An RS element coupling two domains

$$e_2 f_2 = e_1 f_1$$

$$\text{i.e. } dS_2/dt = \frac{e_1 f_1}{T_2} \tag{2.26}$$

It can be seen from this constitutive relation that this is a modulated `SF' node in that the flow is dependent on the effort variable (temperature), as well as the energy imparted from the other domain.

The argument applied to dissipators from other energy domains conserving energy by passing it into the thermal domain, implies that a thermodynamic dissipator cannot exist. Thermal resistance is not a dissipator but rather a dual entropy flow source which is also represented by an `RS' node. The constitutive relation of a thermal resistance is given by:

$$\frac{dE}{dt} = T_1 \frac{dS_1}{dt} = T_2 \frac{dS_2}{dt} = H.(T_1 - T_2) \tag{2.27}$$

where H is the heat transfer coefficient.

Thermodynamic systems have flow stores in the form of thermal capacity, represented by a `C' element. The constitutive relation of a thermal capacity is:

$$T = T_0 \exp(S/C) \tag{2.28}$$

where T_0 is the initial (absolute) temperature.

Equation (2.28) approximates to

$$T = T_0.(1 + S/C) \tag{2.29}$$

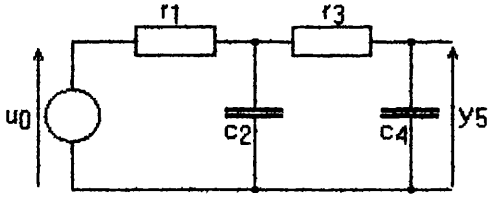
for small differences between T and T_0.

Like magnetic systems, there is no effort store ('I') in thermodynamic systems, which has lead researchers[12] to the conclusion that such stores are not fundamental. It was shown, in the discussion of gyrators that the electrical effort store (inductance in a coil) is fundamentally a gyrated version of a magnetic flow store. Thus, it is always possible to use the gyrator's ability to make duals of elements to remove the need for the effort store. They are, however conceptually convenient, and in the mechanical domain, at least, neither the 'I' nor 'C' element appears more fundamental. Breedveld has proposed a generalised bond graph theory, where inertances only exist when gyrated from 'C' elements, thus requiring dual (potential and kinetic) mechanical domains.

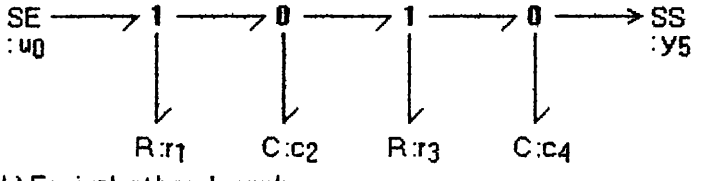
2.4. Bond graph examples

2.4.1. An electrical second order lag

The electrical schematic for a second order lag is given in figure 2.7a, while the bond graph equivalent is shown in figure 2.7b.



a) Electrical schematic of second order lag



b) Equivalent bond graph

Figure 2.7 An electrical second order lag

Since electrical schematics provide an unambiguous representation of the real system, it is possible to give precise rules[13] for transforming such schematics to bond graph notation:

i) Draw a `0' junction for each point in the schematic where parallel paths coincide.

ii) Draw a `1' junction for each component on a series path, and attach the appropriate bond graph component by a bond to that junction. The arrowhead on each bond indicates the assumed direction of power flow, i.e. from sources and towards stores and dissipators.

iii) Draw bonds between adjacent junctions, again indicating notional direction of power flow

iv) Remove the `0' junction representing the reference point (typically the 0 Volt rail) and remove all bonds attached to this junction.

v) Remove any remaining 2-port junctions and move attached nodes to the adjacent junction.

This procedure converts even the most complex electrical schematics to bond graph form, for further analysis using bond graph techniques. The `SS' element at the end of the graph shown in figure 2.7b has been added to indicate an ideal sensor is needed and that, for this model we are interested in monitoring the output voltage across capacitor C4. General bond graph notation does not include sensor elements, but they are included in this text to explicitly identify outputs from the model, and, as will be described in chapter 4, to provide systematic analysis of inverse system models.

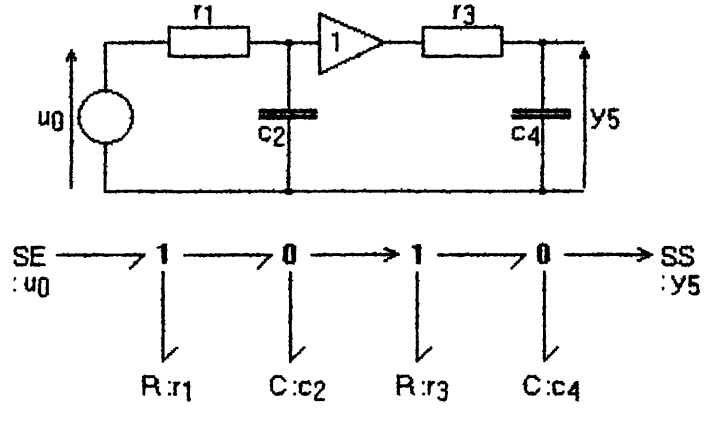


Figure 2.8 An electrical second order lag with buffer

Figure 2.8 shows a modified version of the circuit of 2.7 containing a buffer amplifier (of unit gain) connecting the two

halves of the circuit. In this case there is no current flow from the 0-junction to the 1-junction and so the corresponding bond is replaced by an activated bond (signal).

2.4.2. A hydraulic brake system

Figure 2.9a shows a simplified schematic of an automobile braking system with a hydraulic system connecting the foot pedal to two brake pads, pressing against the brake disc. The system is shown first as a word bond graph (figure 2.9b) to better illustrate the components of the system, while figure 2.9c shows the complete bond graph of this system.

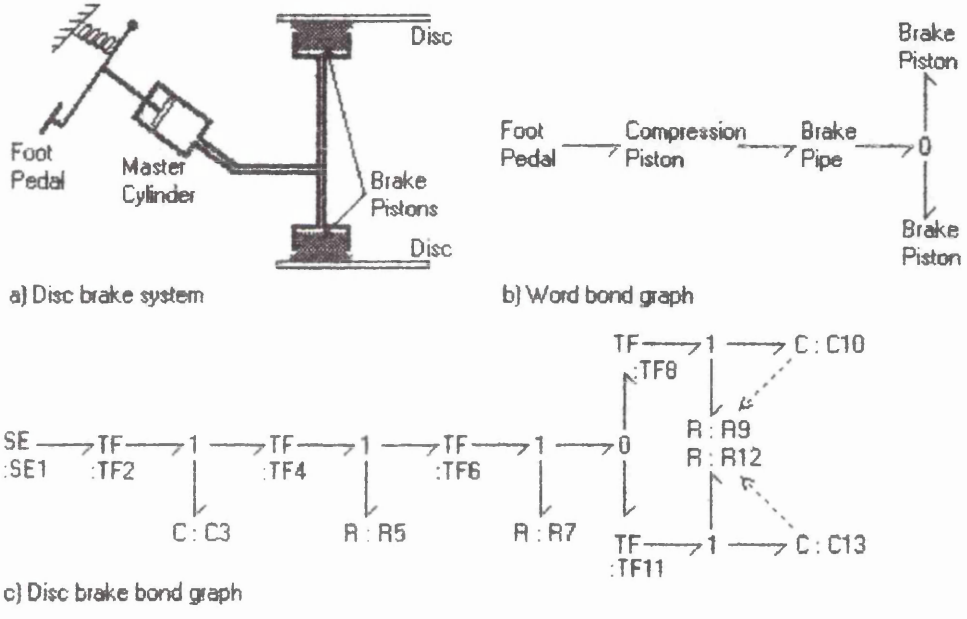


Figure 2.9 A disc brake system

A force is applied (by the effort source SE1) to the brake pedal which is coupled by an end-pivoted lever, represented by element TF2, to a return spring with compliance C3. Since the piston rod is connected to a third point on the lever a further transformer (TF4) is required to couple the resultant of the applied and spring forces to the piston rod. Frictional force imposed on the piston rod is represented by R5 which is attached to the 1-junction representing the velocity of the piston rod.

The master cylinder (TF6) transforms the force on the piston to a hydraulic pressure applied to the brake pipe. This pressure is measured at the outlet of the master cylinder into the brake pipe, which is assumed to have a small resistance (R7) to fluid flow. The

brake fluid is assumed to be incompressible, as is the case for normal safe operation of the system; although in a faulty system, air in the fluid can make it appear compressible.

The split into pipes for each brake is modelled by a 0-junction where the common pressure is applied to each brake piston in its calliper cylinder. These cylinders transform (TF7, TF11) the hydraulic pressure to forces on the brake pads which firstly overcome the frictional forces and the compliance due to the pad retainers. The reaction force from the brake disc may be modelled in several ways, but here it has been chosen to model this by modulating the dissipator parameters (R9 and R12) according to the position of the pads (i.e. the states of C10 and C11). The modulation causes the `friction' to become infinite when the pads meet the disc thus giving zero (pad) velocity. A more detailed model of this system could employ an `RS' element (see section 2.3.3e) to indicate that the force of the pad on the disc results in conversion of mechanical (friction) energy into heat which can effect the pad friction parameters, and cause the brake fluid to expand.

2.4.3. A DC motor

The bond graph model of a DC motor is developed from first principles, by considering the force (F) on a current carrying wire perpendicular to a uniform magnetic field (B). If the length of the wire is l and the current is i, then Faraday's law gives:

$$F = Bli \tag{2.30}$$

Assuming the wire is free to move across the magnetic field with velocity (u) the emf generated in the wire is:

$$e = Blu \tag{2.31}$$

Since we have defined voltage and translational force as effort variables, and current and velocity as flow variables, we can see that equations (2.30) and (2.31) represent gyrator action between the electrical and mechanical energy domains. The power passed through the gyrator is Blui, and the gyrator ratio is Bl.

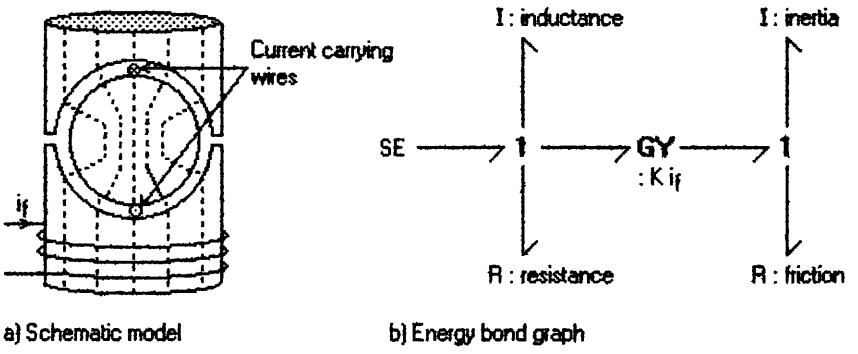


Figure 2.10 Models of a DC motor

Figure 2.10a schematically shows how this is implemented in a DC motor, where each turn of the armature wiring experiences a force 2*F due to the two lengths per turn. In practice, the armature winding has significant resistance and inductance since many turns are required. The armature mass also results in rotational inertia, while friction losses occur in the bearings. The bond graph model is shown in figure 2.10b, indicates that the electrical resistance and inductance are in series with the e.m.f. required to drive the motor, and the armature inertia and friction losses are on a common velocity junction.

It can be seen that the gyrator ratio is proportional to both the number of active turns on the armature (n), and to the magnetic flux density, B. Since the magnetic field is often generated by a separate field winding, the gyrator ratio is then dependant on the field current, since:

$$B = \frac{\phi}{A} = \frac{\mu N i_f}{l_e} \qquad (2.32)$$

where μ is the permeability of the field core, N is the number of turns on the field winding, l_e is the effective magnetic path length, and i_f is the field current.

Thus for a given motor, the gyrator ratio is Ki_f,

$$\text{where } K = \frac{2lN\mu N}{l_e} \qquad (2.33)$$

Hence the motor gyrator ratio is actually modulated by the field

current, if this is not constant.

2.4.4. An electric heater

For this example we will develop an energy bond graph model of an electrical heater rather than the more common model which uses heat flow rate as the flow variable. Figure 2.11 models the electro-thermal conversion as an energy conservative RS element which sources an entropy flow to the thermal capacitance (C3) of the heater, and to a thermal resistance (RS4) representing heat loss to the ambient (SE5).

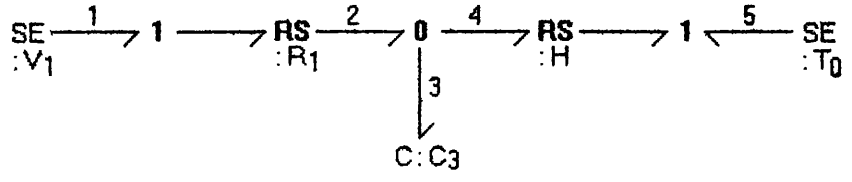


Figure 2.11 Bond graph of an electrical heater

The input power from the electrical source is $V_1^2/R1$, where R1 is the electrical resistance of the heater. The thermal power generated is therefore:

$$e_2 f_2 = V_1^2/R1 \tag{2.34}$$

where e_2 is the absolute temperature and f_2 is the entropy flow generated. This entropy flow splits at the 0-junction between that into the thermal capacitance, causing the rise in temperature, and that passing through RS4 to ambient. The (linearised) rise in temperature is approximated $T_0 S/C3$ where T_0 is the initial (ambient) temperature and S the integrated entropy flow into C3, giving:

$$e_3 = T_0(1 + S/C3) \tag{2.35}$$

The heat flow to ambient is

$$Q_4 = e_4 f_4 = H(e_4 - T_0) \tag{2.36}$$

where H is the thermal conductance between heater and ambient.

Since the efforts are common at a 0-junction

$$e_4 = e_2 = e_3 = T_3 \qquad (2.37)$$

and the flows split at this junction

$$f_2 = f_3 + f_4 \qquad (2.38)$$

Therefore

$$f_3 = [V_1^2/R1 - H(T_3 - T_0)\ ]/T_3 \qquad (2.39)$$

$$= [V_1^2/R1 - H(T_0(1 + S/C3) - T_0)\ ]/T_0(1 + S/C3)$$

For $S/C3 < 1$ we can approximate $(1 - S/C3) = 1/(1 + S/C3)$ giving the state equation:

$$f_3 = [V_1^2/R1 - HT_0S/C3\ ](1 - S/C3)/T_0$$

i.e. $$S' = [V_1^2/R1 - HT_0S/C3 - V_1^2S/(C3R1)]/T_0 \qquad (2.40)$$

ignoring terms in $(S/C3)^2$.

2.5. Causal augmentation of bond graphs

The concept of (computational) causality is central to the systematic resolution of bond graphs into the mathematical form chosen by the modeller. Due to the importance of this concept Chapter 3 is devoted to exploring this in more depth. This section explains causality in the context of bond graph analysis.

Assigning the causal orientation of a given bond in the graph implies that specifically either the effort or flow variable on that bond is known, and this known value (or expression) may then be propagated through the graph to arrive at a complete mathematical model. The rules for causally augmenting the bond graph permit the system equations to be ordered automatically for solution either by hand or by computer software.

In keeping with the concise graphical approach, causality is indicated on the bond graph by a causal stroke at one end of a bond joining two nodes on the graph. This stroke is drawn at the end of the bond nearest the node to which the effort is directed - the flow by implication is directed toward the node at the other end. The only elements that can force causality are effort or flow sources, and the structural elements - figure 2.12a shows this notation applied to sources and to dissipators - the indicated direction of the energy flow is seen to be irrelevant to causality.

Figure 2.12b shows how causality is propagated through the bond graph by the structural elements; `0', `1', `TF' and `GY'. Since the effort at a 0-junction is common to all the bonds on that junction, only one bond can define the effort on that junction, the remaining bonds impose flows on the junction, while propagating the known effort to attached nodes. In contrast, only one bond determines the flow at a 1-junction, while the remaining bonds impose efforts on the junction. The transformer ('TF' node) passes causality on directly (thus a bond can be considered as a transformer with ratio 1), while gyrators have the effect of inverting causality - hence the application of gyrators to achieve the dual of an element.

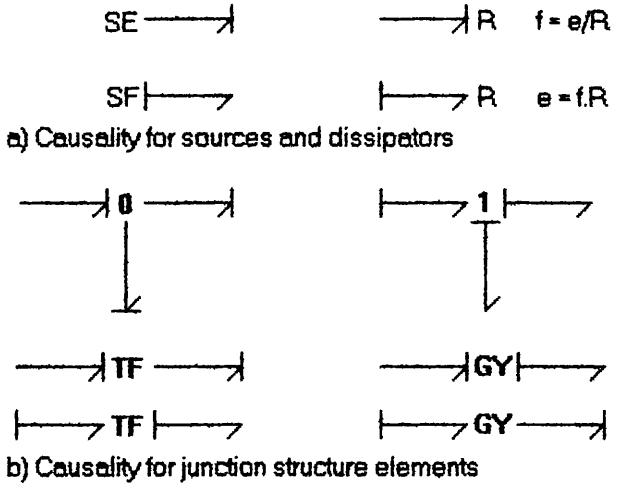


Figure 2.12 Permissible causalities.

Elements which are energy stores or dissipators do not impose causality on the system, although they may have preferred causality for computational reasons. In general, therefore, *the causality*

assignment of a given bond graph is not unique, being dependant on the modeller's choice of mathematical model. In particular, systems having a large proportion of dissipators could be described as 'under-causal' since the modeller may have to make one or more arbitrary choices of causality in order to complete causal assignment on the bond graph. The consequence of under-causal systems is that some intermediate variable has to be eliminated by the solution of an algebraic loop, before the complete mathematical model can be derived.

In such cases, the modeller's choice of causality assignment may not be entirely arbitrary but, as preferred to improve ease of computation, and minimise the number of algebraic loops[14]. For example, it is more convenient to calculate the effort variable from the flow for a dissipator (R) representing the turbulent flow through a pipe where pressure drop (e) is given by the following function of flow rate (f):

$$e = Rf|f|^{3/4} \tag{2.41}$$

It has been noted that activated bonds can be used to represent sources modulated by a signal. In such cases, the activated bond has the causality of the modulated source and is drawn with a causal stroke in this text, to distinguish it from a pure modulating signal.

Model reduction.

The modeller may also wish to investigate the effect on the process when a component is removed. This can be done by removing the element from the graph, or, more conveniently, changing its parametric value to zero. This can have fundamental effects on the system states, due to changes in causality. For example, if the element is a dissipator whose causality was initially defined such that the constitutive relation was evaluated as:

$$f = e / R$$

then defining R to be zero gives a computational problem, unless the opposite causality is forced by the modeller, with consequent

changes in the causal augmentation of the model. This may have the effect of turning a `stiff' model of the complete system into a reduced order model with interdependent energy stores.

2.5.1. Integral or derivative causality?

Table 2.2 shows that the constitutive relations of the energy stores contain information about the system inputs and state variables p and q, thus permitting the system dynamics to be fully represented. The emphasis in bond graph literature has been on the transformation of graphs to state equations - choosing alternative causality assignment rules results in different forms of mathematical model. When one is transforming the bond graph into its state equation form, the causality of interest for energy stores is termed integral causality, where the constitutive relations of the energy stores are in the form given in Table 2.2. The ability to assign integral causality also implies that the system is physically realistic, thus providing a deeper level of analysis of system constraints than would be possible without the concept of causality. A mixture of integral and derivative causality may then be forced by the causality propagation in real physical systems, but it implies that at least two of the energy stores are not dynamically independent - only those exhibiting integral causality result in state variables. This causal conflict can be considered as `over-causal' by comparison with largely dissipative systems, since the consequence is also an algebraic loop - this time relating the interdependent energy stores.

Applying derivative causality to the energy stores in a bond graph results in the derivative form of mathematical model for the system[15]. The resulting mathematical model is then in the most general form - a set of differential and algebraic equations (DAE's), although in some cases ordinary differential equations (ODE's) may result.

Derivative causality may also be applied to energy stores in order to facilitate static analysis of systems, without modifying the fundamental structure of the bond graph model. Since the derivative forms of the constitutive relations for energy store are:

$$e = I \frac{df}{dt} \text{ and } f = C \frac{de}{dt} \qquad (2.42)$$

it can be seen that the static model is given when either the constitutive parameters (I and C) are zero, or when the effort and flow derivatives are all zero, i.e. the stationary state.

Thus, assigning derivative causality to stores and propagating this through the bond graph permits a derivative based model to be generated, and substituting zero for all energy store components then results in the steady-state mathematical model. Similar bond graph solutions to the problem of deriving the steady-state model have been proposed[16], but the method described here has the advantage of retaining an invariable bond graph core model regardless of the transformation required to obtain the desired mathematical model.

2.5.2. Rules for assigning causality to a bond graph

Bond graph causality rules define the causality augmentation of any given bond graph in an entirely systematic manner, permitting the automatic derivation of the appropriate mathematical model for the system. The Sequential Causality Assignment Procedure (SCAP) due to Karnopp and Rosenberg[17], is the basis of most such algorithms. The following procedure is based on the SCAP, but with an additional rule to integrate activated bonds into the causality assignment. The rules listed here give a systematic method for causally augmenting a bond graph such that a state equation model may be derived. Section c) should be amended appropriately if derivative causality is needed for a DAE model.

a) Assign causality to any known effort or flow, such as activated bonds (signals) derived from junctions. For example, a signal from a 0-junction transmits the effort from that junction while the flow from this junction into the signal is, by definition, zero.

b) Assign causality to bonds linking directly to each source and propagate these causalities as far as possible through the junction structure by applying the causality constraints for structure elements (0, 1, TF, GY).

c) Assign integral causality to each energy store in turn and propagate this throughout the junction structure. Any conflict between the causality due to the store must be resolved by reassigning derivative causality on that store and propagating the new causality through the bond graph.

d) If any unassigned bonds remain, then assign a causality either arbitrarily or for computational convenience and again propagate this through the junction structure. Assigning causality to an unassigned interjunction bond (internal bond), such that this bond forces causality on both attached nodes, minimises the number of algebraic loops. Repeat for any remaining unassigned bonds.

2.5.3. Examples of causally augmented bond graphs

a) An electrical second order lag.

In this example, the second order lags modelled in section 2.4.1 have causality applied according to the rules given in section 2.5.3. Figures 2.13a and b indicate that the causality pattern is the same for the model without and with the buffer amplifier (represented by the activated bond 7), respectively. The interjunction bonds have been allocated identification numbers, while the remaining bonds are identified by the subscript of the terminating constitutive element.

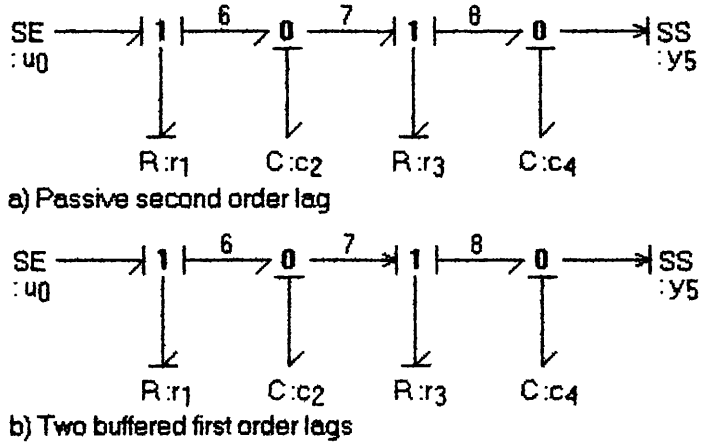


Figure 2.13 Causalities for electrical second order lags

Table 2.3 is derived by following causality in the order which it propagates through the graphs, following each propagation path as far as possible i.e. while each right-hand-side variable is known.

Since there are no algebraic loops there are 18 bond equations, corresponding to the effort and flow on each of the nine bonds. The ordering of the equations is different for the two models since f7 = 0 is one of the known variables for the buffered circuit. The state equation may be evaluated by selecting the derivatives (f2 and f4) of the state variables (q2 and q4) and working backwards through the table of equations.

#	Passive lags	Comment	With buffer	Comment
1	f5 = 0	ideal sensor	f5 = 0	ideal sensor
2	e0 = u	input source	f7 = 0	activated bond
3	e2 = q2/c2	integral causality	e0 = u	input source
4	e6 = e2		e2 = q2/c2	integral causality
5	e1 = e2 - e6		e6 = e2	
6	f1 = e1/r1		e1 = e2 - e6	
7	f0 = f1		f1 = e1/r1	
8	f6 = f1		f0 = f1	
9	e7 = e2		f6 = f1	
10	e4 = q4/c4	integral causality	f2 = f6 - f7	
11	e5 = e4	output	e7 = e2	
12	e8 = e4		e4 = q4/c4	integral causality
13	e3 = e7 - e8		e5 = e4	output
14	f3 = e3/r3		e8 = e4	
15	f7 = f3		e3 = e7 - e8	
16	f2 = f6 - f7		f3 = e3/r3	
17	f8 = f3		f8 = f3	
18	f4 = f8 - f5		f4 = f8 - f5	

Table 2.3 Causally ordered equations

b) A fixed field DC motor.

The DC motor modelled in section 2.4.3 with a voltage source applied to the armature, indicates the potential of bond graphs for unambiguously representing a mixed energy domain system (figure 2.14a).

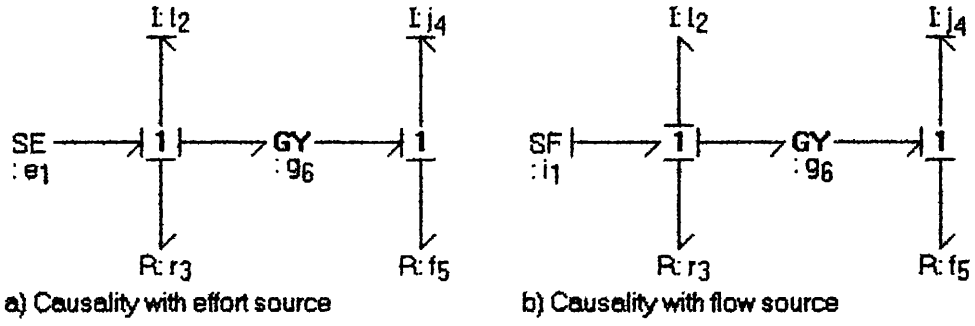


Figure 2.14 Causality variations on a DC motor

Applying the causality rules, with integral causality on stores, results in a model with two state variables p2 and p4 and a single input e1. A bond graph model of the same motor, driven by an electrical current source is shown in figure 2.14b. Applying the causality rules to this bond graph indicate that I2 now has derivative causality imposed on it, and the system reduces to a first order model since p4 is the only state variable and the input is f1. The physical implication of derivative causality on the inductance I2 is that the current source, SF1, must be able to supply the very high voltages which will occur for step changes in motor loading.

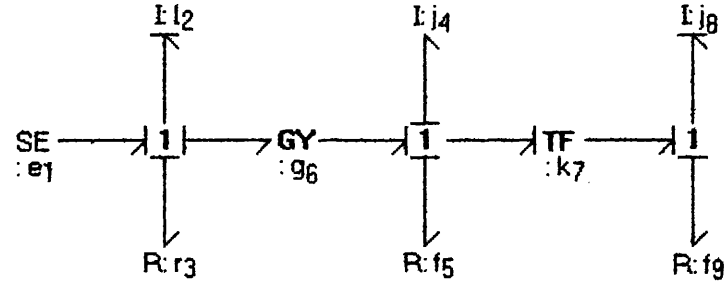


Figure 2.15 Causal conflict due to interdependent inertias

Figure 2.15 shows the effect of adding a gearbox to the voltage driven DC motor. In this case, derivative causality is forced on the motor shaft inertia or the load inertia, since these are not independent, being linked by the transformer ratio of the (non-compliant) gearbox. In practice, the shaft linking the motor to the gearbox will have some compliance resulting in a 'C' element between the motor inertia and the gearbox, which solves the causality problem and introduces another state variable. The likelihood is, however, that this compliance is very small

resulting in a `stiff' system where the time constant due to the compliance is significantly smaller than those due to the inertias. This may give numerical resolution problems when simulating the system using this mathematical model.

2.6. Multi-port energy nodes

In the preceding descriptions of bond graph elements, all those representing component behaviour, i.e. sources, stores and dissipators, have had only one port through which energy is exchanged with the rest of the system. In general, these elements can be multi-ports (alternatively called N-Ports or fields) in the same way that the structure elements, discussed in section 2.1.2, have more than one interface to the rest of the model.

It is important to note that all the structure elements of a given model may be considered as a single multi-port element called the *junction structure*, and this concept is the basis of much bond graph theory. This section gives examples of multi-port elements in a variety of energy domains, and their application in bond graph models.

2.6.1. R-fields

In the electrical domain it is often convenient to group a network of resistors together into one multi-port resistor (or R-field) represented by a matrix of resistive (or conductive) elements. Figure 2.16a shows a simple electrical circuit where the dissipators may be grouped together as a 2-port `R-field', as represented in the augmented bond graph of figure 2.16d. The circuit is also shown represented by one-port `R' elements in the partially-augmented bond graph of figure 2.16c. This last figure indicates that there are several options for completing causality on this bond graph; choosing to assign f7 as `known' permits causality to be completed, resulting in only one algebraic loop and the shortest computation.

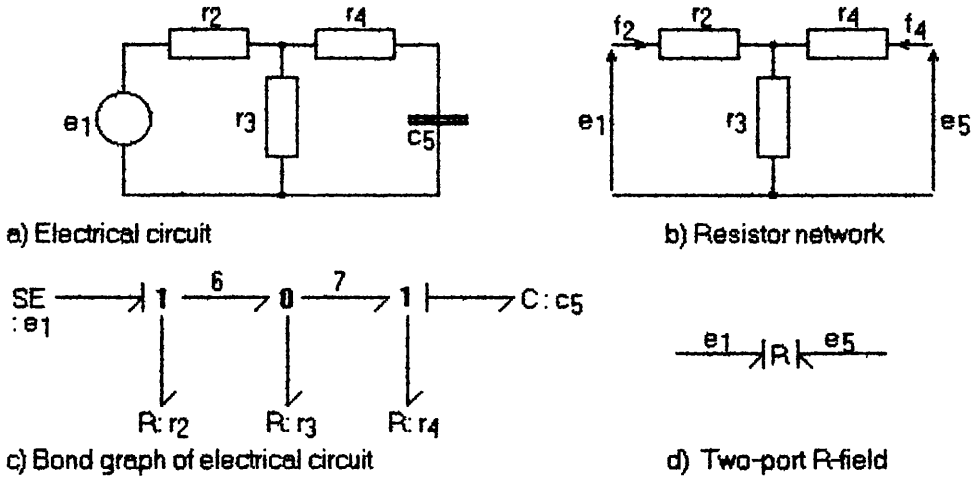


Figure 2.16 Applications of R-fields

Alternatively, simple circuit analysis of the resistors as a separate network (figure 2.16b) gives the constitutive relations of the R-field in the resistive form which may be inverted to give the conductance matrix form required by the given causality:

$$\begin{bmatrix} f_2 \\ f_4 \end{bmatrix} = \frac{1}{d} \begin{bmatrix} (R_3+R_4) & -R_3 \\ -R_3 & (R_2+R_3) \end{bmatrix} \begin{bmatrix} e_1 \\ e_5 \end{bmatrix} \qquad (2.43)$$

where denominator $d = (R_2R_3 + R_2R_4 + R_3R_4)$.

It is then a trivial substitution, $f_5 = -f_4$ and $q_5/C_5 = e_5$, to obtain the state equation:

$$f5 = \frac{1}{d} \left[R_3 e_1 \; - \; (R_2+R_3) \frac{q_5}{C_5} \right] \qquad (2.44)$$

The same result is obtained for similar computational effort, including the algebraic loop, by following the completed causal assignments on the one-port bond graph. Hence, it can be seen that the R-field has been used to solve the algebraic loop while calculating the matrix coefficients - in such cases, the choice of one-port or multi-port representation is purely the modeller's preference. It can be seen that R-fields can also be defined as having mixed causality, i.e. the dependent vector may be a mixture of efforts and flows.

2.6.2. I-fields

A more useful example of an electrical multi-port is that of an N-

port inductance (I-field) representing an electrical transformer with multiple secondaries. For integral causality, the constitutive relations of this I-field are given by a symmetric matrix with self-inductances on the diagonal and mutual-inductances between windings as the off-diagonal elements.

Many mechanical components are also best represented by multi-ports - the dimensional constraints on the mass elements of rigid bodies implies that all such bodies are I-fields, and conceptually it is most appropriate to model such bodies using a single constitutive relation.

2.6.3. C-fields

Multi-port `C' elements also have significant use in the analysis of mechanical systems - a common example is that of an elastic beam deformed by forces applied to two points along the beam. For such cases the elastic displacement of the beam at the two points is related to both the applied forces and to their relative positions along the beam.

C-fields can also be used to represent the behaviour of energy stores which span energy domains - some transducers operate by storing energy in one domain and later converting it (ideally without loss) into the other domain. An example of such a transducer is the condenser microphone, where a velocity (due to acoustic pressure) is imposed on a springy diaphragm (mechanical capacitance), which is also a plate on a pre-charged electrical capacitor. Movement of the diaphragm causes the electrical capacitance to vary (ideally as the inverse of the distance between the diaphragm and the fixed plate) thus resulting in a change of the voltage on the capacitor.

2.6.4. Multi-bonds

Multi-bonds[18] (originally known as *vector bonds*) are a generalisation of the single bond used up to this point, and indicate multiple energy transfers between (multi-port) nodes on the bond graph. The multi-bond is drawn as a large arrow (figure 2.17) to distinguish it from a single bond, and is treated as a vector of individual bonds.

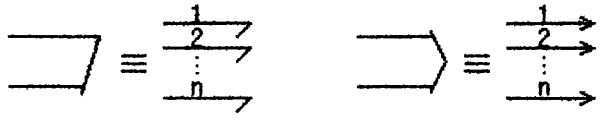


Figure 2.17 Multi-bond notation

Multi-bonds extend the advantage of conciseness and clarity when graphing systems with many multi-port components. A restriction is that all the bonds represented by the multi-bond must have the same causality i.e. the vector of dependent variables must consist entirely of efforts or entirely of flows. Similarly, an activated multi-bond must consist only of signals having the same causality.

2.7. Pseudo bond graphs

Throughout this chapter we have restricted our modelling to systems where energy is the exchange variable, accepting that this may limit the application of the resulting modelling technique. This restriction is overcome by the use of *pseudo bond graphs*, which provide a means of modelling systems in which the integrated product of the effort and flow variables is not energy. Two examples of the use of pseudo bond graphs are given in the remainder of this section, firstly for analysing manufacturing system dynamics, and then for a heated tank using heat flow as the flow variable (rather than entropy).

2.7.1. A manufacturing system model

Significant work has been done in the field of macro economic modelling using pseudo bond graphs[19], where the effort variable is price/unit and the flow variable is the flow rate of a given commodity. The resulting exchange variable is the accumulated price of goods exchanged, i.e. the rate of movement of capital (value rate) is analogous to power in an energy bond graph. In economic systems, the analogy to energy conservation laws is Walras' law, which states that the sum of the value rates into a port is zero.

Since, we are attempting to achieve a continuous model of the system, it is necessary for the flow rate of the commodity to be large enough for aggregation of this flow to be statistically

valid. This must be born in mind when modelling manufacturing systems, where the flow variable is typically the flow of produced items throughout the factory.

For this example, we consider a single manufacturing production line for electronic instrumentation consisting of a mechanical package, one basic printed circuit board (PCB), up to `n' option PCBs, and the associated documentation and packaging. We will assume that demand for the instrumentation is very variable, but delivery times must be low, resulting in the manufacturer building for stock. The pseudo bond graph for this system is shown in figure 2.18, which combines both elemental components, and hierarchical sub-systems.

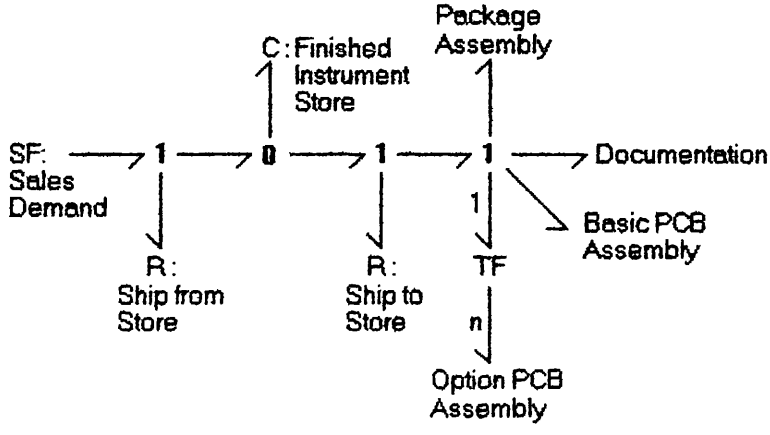


Figure 2.18 Pseudo bond graph model of a manufacturing system

The sub-systems are represented as word bond graph nodes, which have specific dynamics associated with the underlying processes. The system input is a flow source representing the demand for the instruments from sales, which is supplied from the finished instrument stores, represented by a capacitance.

In economic bond graphs 1-junctions are used to describe points at which several incremental costs are added to give the overall cost of the item, but the flow of items on each attached bond is identical (by application of Walras' law). Thus we can see that the overall cost of the instrument before it passes to the stores is the sum of the costs of all its sub-assemblies, and the process of locating it in the store dissipates additional handling costs.

The store itself is linked to a 0-junction, at which the cost remains constant, but the flows into the junction must all add to zero, i.e. the store accumulates the difference between the supply from the production line and the output to sales. Again, the process of handling the instruments between stores and sales incurs a cost represented by the effort across a dissipator. Dissipators in such systems (representing value**less** added activities) are typically highly non-linear, the constitutive `resistance' having high values for small flows. The final unit cost to sales varies according to the demand, being dependent on both the finished instrument cost and the additional handling costs.

The addition of multiple option boards to the instrument is modelled using a transformer which scales the cost on the finished instrument side by `n', and scales the flow rate demand on the option board PCB assembly sub-system by the same factor.

This model has not explicitly included an `I' element, but these occur in macro-economic models, representing investment in capital equipment used to produce higher volumes of equipment more cheaply. The constitutive relation of this inertance results in rapid unit cost increases when the flow suddenly decreases, and vice versa, although the relationship is typically non-linear. Care should be exercised when modelling capital investment in individual manufacturing systems using inertance, since the low level of aggregation may invalidate the model. However, applying integral causality to manufacturing models using inertance to represent investment, does produce interesting qualitative insights whenever causal conflicts occur.

2.7.2. Thermal energy transport model

This example[20] has been chosen to illustrate that it is quite reasonable to model energy transfer systems using pseudo bond graphs. Further, it is possible to mix these with energy bond graphs, as long as the interface between these forms is consistent. Figure 2.19a illustrates the single tank system diagramatically, while figure 2.19b shows the pseudo bond graph model of the system.

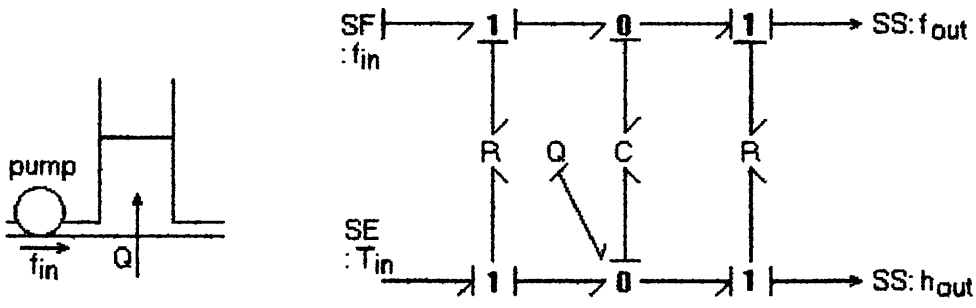


Figure 2.19 Heated tank system

The pseudo bond graph model is in two parts; the upper half corresponds to the hydraulic properties (pressure and mass flow), and the lower half corresponds to the thermal properties (temperature and enthalpy flow). These two parts are interfaced via 2-port elements between the two sub-systems - the constitutive relation of the `R' elements is:

$$\frac{dh}{dt} = (c_p \frac{dm}{dt}) T_{in} \qquad (2.45)$$

That is, the enthalpy flow dh/dt into the system is the product of the temperature (effort) T_{in} and the `conductance' ($c_p dm/dt$). The mass flow dm/dt, derived from the hydraulic model, modulates this relation, and the modulation coefficient c_p is then the specific heat of the liquid.

The integral of this relation also gives the constitutive relation of the thermal capacitance:

$$h = (c_p\ m).T \qquad (2.46)$$

The modulation for this thermal capacitance is shown on the bond graph by the 2-port `C' element, where the hydraulic state variable m is the modulating variable. The hydraulic capacitance is expressed in the constitutive relation between the effort variable (pressure at base of tank), and the state variable (mass in the tank):

$$p = \frac{gm}{a} \qquad (2.47)$$

where g is gravity.

Although the bond graph is equally valid for non-linear relations, for simplicity it is assumed that:

a) the fluid is incompressible,

b) the tank has constant cross-section (a), and

c) the pipes to and from the tank have constant flow resistance (r).

The hydraulic state equation can be derived, using the causality shown, as:

$$\frac{dm}{dt} = f_{in} - f_{out}$$

$$= f_{in} - \frac{gm}{ar} \qquad (2.48)$$

The thermal state equation is:

$$\frac{dh}{dt} = c_p f_{in} T_{in} + Q - \frac{h}{m} f_{out} \qquad (2.49)$$

2.8. Bond graph tools

It has been shown that mathematical models may be systematically derived from a bond graph, and thus the bond graph is well suited to act as a `front-end' to a computer-aided modelling tool.

Several such tools exist:

- ENPORT
- CAMP
- CAMAS
- BONDYN
- BondTool_M

Although this is not an exhaustive list, it includes the most widely used implementations.

ENPORT[21] is probably the most widely used modelling program based on bond graph theory, and is capable of simulating a wide range of

systems. The product is fairly old, so the user interface is driven by a dialogue of questions with user responses, resulting in a text file description of the bond graph. This, and the flexibility the system offers in describing constitutive laws, result in complexity in entering model data. A model cannot be derived directly from graphical input of a bond graph, so modifying or extending models is also complicated. ENPORT supports modulated and multi-port components and non-linear constitutive laws for bond graph components, and also for a small selection of block diagram components, which can interface to the bond graph.

The program uses causality to sort equations for simulation and to identify algebraic loops, so that the simulator included in the package can solve these by iteration. The simulator output is restricted to Tektronix graphics terminals or character based hard-copy plots. The main limitation is that no other output option than a simulation run is available.

CAMP[22] (Computer-Aided Modelling Program) is a (relatively primitive) bond graph pre-processor for several simulation languages, such as ACSL[23] and IBM/DSL[24]. Bond graphs are entered textually using a line code, listing each element on the graph, and the bond number to which it is attached. CAMP parses this textual representation of the graph and assigns causality using the SCAP algorithm. Causal conflicts and algebraic loops are *detected*, but only as a guide to the user, whose responsibility it is to resolve the problem. The resulting output is an *unsorted* set of system bond equations, which have been converted to a FORTRAN format suitable for the specific simulation back-end.

CAMAS[25] was developed to overcome the main limitations of ENPORT, and offers graphical model creation, and high quality simulation output (including visualisation). It is intended to offer comprehensive analysis capabilities, in the longer term. Its main weakness, is that in handling complex hierarchical systems, it uses the multi-bond notation, which restricts the causal augmentation options on the sub-models. In addition, CAMAS enforces rigorous model checking, which removes some of the flexibility offered by ENPORT's flexible use of modulations, restricting CAMAS from modelling some pseudo bond graphs.

BONDYN[26] is a bond graph based pre-processor similar in nature to CAMP, except that it is designed specifically for simulation of non-linear dynamic systems resulting from multibody systems. BONDYN utilises multi-bond concepts for dealing with large multibody systems, and bond graph models are created by selecting sub-models from a library. The interesting feature of this package is that it uses two new bond graph elements (a `Rigid Spring' and a `Zero Inertia') to represent constraint equations. Causality is assigned by the program, but derivative causality results in an error which must be manually resolved. The output is a set of FORTRAN77 formatted files defining the parameters and the system equations, which may be used as the mathematical model for simulation using DASSL[27].

BondTool$_M$[28] is an interesting implementation of a bond graph front-end for the Matlab[29] System Identification Toolbox. The modeller enters the bond graph model as a graph, and the tool produces a semi-symbolic M-file as input to the identification toolbox, which is used for parameter estimation and running simulations. The main limitation of the tool is that it can only handle linear systems, but it can identify and solve algebraic loops and loops due to derivative causality. The resulting causal augmentation is not shown on the bond graph, which, in this case is just an input mechanism for the model.

2.9. Conclusion

This chapter has highlighted the requirements for modelling elementary systems based on their energy transfer characteristics. A review of bond graph theory has shown that this notation meets all these requirements, while pseudo bond graphs may be used to model non-energy systems. Choosing energy as the unifying variable permits physical systems covering several energy domains to be modelled in a consistent manner, with pre-defined interfaces.

Separating the model structure from the elemental behaviours permits the model to be easily modified, due to its close mapping onto the actual system structure. This also allows non-linear and time-dependent behaviours to be handled separately in the constitutive relationships of the bond graph elements.

The application of a small set of causality rules permits bond graphs to be analysed systematically, either by hand or using a computer. Causality analysis has been shown to be a powerful tool not only for deriving different forms of the mathematical model, but also for revealing conflicting system constraints.

CHAPTER 3 HIERARCHICAL MODELLING USING BOND GRAPHS

3.1. Introduction

In Chapter 2, the literature relating to modelling of elementary physical systems was reviewed with particular reference to the application of bond graph notation and theory. This chapter extends this review to the modelling of more complex systems which are best described by a hierarchical model. The main emphasis is on the application of bond graph notation to hierarchical modelling.

Having reviewed modelling concepts at the elemental level we can now extend these concepts into hierarchical models where the hierarchy reflects the structure of the system being modelled. It has been shown that reticulation - dividing a system into a network of components - is fundamental to bond graph modelling. Reticulation is also a natural way of analysing large systems, since engineers are used to considering such systems as composed of a structure of interacting sub-systems.

In developing a model of a system, an analyst naturally attempts to break it down into smaller, better understood, components. Thus the ability to model the system as composed of hierarchical sub-models is essential to reducing complexity. The use of sub-models permits the modeller to verify that an individual component functions as specified, before testing the entire system. In addition the model structure is clarified, permitting models to be easily modified and documented.

When analysing large systems, the complexity of the problem may be reduced by decomposing the system into a structure of smaller sub-systems. The sub-systems themselves may also be sufficiently complex that they in turn require further decomposition in a hierarchical manner. It is desirable that a model reflects this hierarchical structure of the real system, for two reasons. Firstly, the modeller can use the same abstractions as the system engineer, thus reducing the scope for interpretation errors while producing the initial model. Secondly, the developed model is to a

large extent self-documenting, and may therefore be modified with a higher degree of confidence than an unstructured model.

Hierarchical decomposition of models has three further advantages for the modeller. Firstly, the modeller is encouraged to focus on a well-defined function within the overall system, thus minimising and defining interactions with other parts of the system. Secondly, this facilitates verification of the model, as the modeller should have a more precise mental model (or even a specification) of the sub-model behaviour, than of the entire model. Finally, once this sub-model has been developed, verified and validated, it can be saved for re-use, as many components are used in a wide variety of applications. Modelling can only be considered to be successful, when easily maintained libraries of sub-models are available, where the sub-models are regularly re-used, in the same way that the real system components are re-used.

It was indicated, in Chapter 1, that the core model representation is a central concept in this thesis, in order to separate the modelling process from the application of a specific derived model. This approach was shown to have other advantages, in maintainability of the model, and in achieving consistency between different types of derived models. The following chapters will illustrate the point that different models may be derived from a bond graph, by the application of specific causal initiation rules before propagating causality through the model.

Section 3.2 of this chapter justifies the choice of bond graphs as the core model representation for a hierarchical modelling tool. The following sections evaluate different approaches to hierarchical modelling using bond graphs; section 3.3 discusses the multi-port representation, and section 3.4 reviews the multi-bond graph notation. The hierarchical word bond graph model is proposed (section 3.5) as the best solution for the requirements outlined in section 3.2. Finally, additional features necessary for the implementation of a hierarchical bond graph modelling tool are highlighted in section 3.6, while section 3.7 concludes the chapter.

3.2. Bond graphs as a core model representation

In this section we give further justification for the choice of energy bond graphs as the core model representation for a modelling tool. By comparison, most control system design environments use a control-oriented data representation[30,31], such as transfer functions or state space equations. One motivation of this thesis is that such representations are not sufficiently fundamental to permit all other representations to be derived from them.

The fundamental difference in the approach described in this thesis, is that here, the same core model representation is used for deriving different representations applicable to a variety of different applications. The range of uses envisaged covers control design, process design, simulation and system understanding. The derived representations must clearly be appropriate to the use of the model, and are considered as different views of the physical system. Some possible representations are: a state space equation, a frequency response of a linear transfer function, an inverse system transfer function, a human readable equation, or machine readable (possibly non-linear) simulation code.

Bond graphs[10,11,17] provide a clear concise notation for describing a wide variety of systems, and have a number of advantages compared to block diagrams.

- They have the important property that they relate closely to the structure of the physical system[7], at a level below the hierarchical (word bond graph) structure. This has the advantage of making the model amenable to modification for the purpose of process development, and 'what-if?' simulations.
- The bond graph can be drawn before causality is considered, whereas causality has to be considered before a block diagram can be drawn. Thus different block diagrams may be appropriate for the same system depending on the exogenous inputs to the system. The great power of the bond graph notation is that *the acausal graph is a declarative representation of the system,* independent of the system environment.

- The bond graph provides a *symbolic* description of the system, and is therefore ideally suited to symbolic, rather than numeric analysis. In particular, the bond graph contains enough information to derive various other system representations, where functions and parameters may remain in a completely symbolic form. Such a symbolic model can then be readily converted into numerical models for producing, for example, simulations and frequency responses of the modelled system.
- A sign convention can be systematically assigned by the bond graph method. In particular, directions of power flow are assigned rather than having to assign direction individually for effort and flow variables.
- In addition, bond graphs provide a unified method for describing systems comprised of mixed energy domains[9], since energy is the unifying quantity in this notation.

Example: Dependence of derived model on exogenous inputs

Perhaps to the detriment of bond graphs, the standard texts on the subject[9,32] have concentrated on the application of integral causality to bond graph models. This leads naturally to the systematic derivation of a state equation model of the system from a set of ordered bond equations, as illustrated in Chapter 2. Either of these models is suitable as input to a continuous system simulation tool, such as ACSL, TSIM, etc. It was also noted, using the example of causal augmentation of the bond graph of various configurations of a d.c. motor (section 2.5.3b), that causal conflicts when applying integral causality imply that the model may not be physically realisable. This additional check, provides the bond graph modeller with a useful design aid, and is often cited as one of the most useful features of bond graphs.

The d.c. motor examples also highlighted other important points to consider when assigning causality to hierarchical models. Firstly, the inputs to the sub-models do not, in general, have a pre-defined causality - in the case of the d.c. motor, the armature coil may be either voltage or current driven. Taking this point to the extreme, it is evident from this example that the electrical interface of

the motor cannot be uniquely defined as the input, since the motor may be driven from a mechanical input, i.e. as a generator.

The more important consideration is that the causality internal to the sub-model is entirely dependent on that of the exogenous input(s) - the resulting state equation model of the d.c. motor has either one or two states dependent on the input causality. The final d.c. motor example (coupling it to a gearbox sub-model) indicates that the internal causality is also dependent on other connected sub-models. Again the d.c. motor has one state variable out of the possible two, but, in this case, the armature momentum becomes the non-state variable.

Example: Dependence of derived model on causal initiation

The example considered here is a hydraulic system of two tanks coupled through a flow restricting pipe, as shown in figure 3.1.

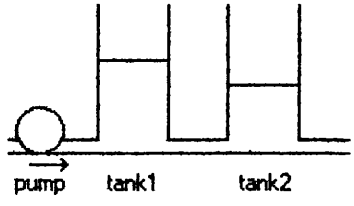


Figure 3.1 Coupled tank system.

An incompressible fluid is pumped into the first tank by a constant pressure pump, and drains out of the second through another pipe.

The *acausal* bond graph representing the hydraulics of this system is shown in figure 3.2a, including labels to emphasise the correspondence to the physical system.

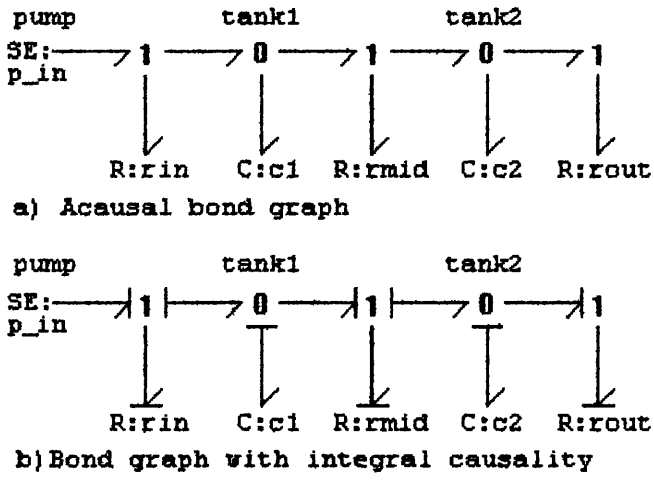


Figure 3.2 Bond graphs of two-tank system

If the modeller wishes to derive a state equation model, he must follow the causal initiation rules appropriate to this model. These require propagation of causality from each exogenous input (p_in), using a standard causal propagation algorithm. Then integral causality may be propagated from each energy store (c1 and c2) in turn (figure 3.2b), assuming no dependent states exist. The automatic application of causality rules also permits rapid derivation of other system models, by applying different rules for *initiating* causal propagation. This is equivalent to defining which are the 'known' parameters in the system equations.

The application to reduced order modelling is typical of bond graph analysis in that it relates much more closely to the physical system compared to other approaches. Model reduction may be approached on a physical basis by zeroing parameter values, if it appears that individual component effects may be insignificant. This has the advantage that the core bond graph is unaffected by the model reduction experiments.

In this example, the modeller might know that one tank in the physical system is much larger than the other, and choose to ignore one capacity (set it to zero) to obtain a first order model. Alternatively, it may be more realistic to ignore the restriction between the two tanks by setting r_{mid} equal to zero. The effect of this is to define the causality on the bond attached to this dissipator, such that the constitutive relation is expressed in the form:

$e_{mid} = f_{mid}r_{mid}$ i.e. $e_{mid} = 0$

rather than $f_{mid} = e_{mid}/r_{mid}$, which is indeterminate.

The design tool described in Chapter 7 includes an algorithm which, prior to initiating causal propagation, checks for zero value parameters, and forces the appropriate causal initiation from such nodes. In this case, r_{mid} imposes an effort on the system, resulting causal propagation completing as shown in figure 3.3. This becomes a first order system with a state variable due to c1, while the causality imposed on c2 results in a dependent non-state.

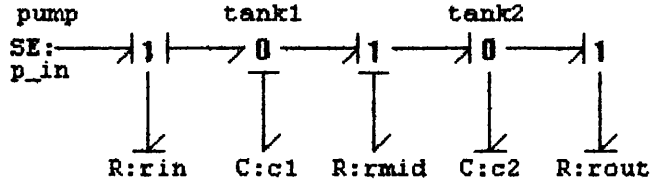


Figure 3.3 Reduced-order model

Note that if the modeller had chosen $c_2 = 0$ (i.e. $c_2 << c_1$), rather than $r_{mid} = 0$, then the constitutive relation of this capacity is only valid in the derivative form:

$$f = c_2\frac{de}{dt}$$

resulting in the same causal augmentation pattern (figure 3.3) and a first order model. However, the ordering of the causal propagation is different, as are the derived reduced order models.

This example has shown that the same acausal bond graph may be used to derive either the complete state equation model of the system or a reduced order model. It can be seen that if this example had been considered as a hierarchical model, by separating out the tanks and feed pipes as sub-models, then the reduced order model would have different mathematical models for tanks 1 and 2. However, using acausal bond graphs to represent the sub-models permits the appropriate sub-model to be derived by applying causality at the `flat' bond graph level.

The proposed approach is therefore, to generate acausal bond graphs for each word bond graph node, and use computing power to reduce the hierarchical system to a `flat' bond graph model. The flat bond graph model can then be systematically causally augmented with the exogenous inputs specific to the test, using a causal initiation appropriate to the derived model required.

3.3. Multi-port representation

One of the strengths of bond graphs is that physically related constitutive laws can be conveniently combined in a multi-port element (section 2.6). The bond graph multi-port has proven to be a useful notation mainly for multi-port storage elements where energy is transferred between ports via the storage mechanism. In many cases, the multi-port store represents a transducer element, so the ports are in different energy domains.

Much theory has been devoted to analysis of multi-ports in bond graphs[12], and the modelling program ENPORT[21] has been implemented to take advantage of this (n-port) notation. A proposed advantage of this approach is that `causal conflicts or algebraic loops can be eliminated once and for all'[33], with the implication that the multi-port may be re-used in an ad hoc manner as a blackbox sub-model within a larger system graph.

The weakness of this approach is that constitutive relations implicit in an n-port element (an n x n field) must be expressed with a given causality. Thus, the field equations will need to be inverted (partially or completely) if the causality imposed by the surrounding bond graph does not match that of the given field equations. In general, a linear field matrix may not be invertible and, for non-linear constitutive relations, the inversion may be non-trivial.

3.3.1. Discussion

Considering a general multi-port element (MP in figure 3.4) with a given causality on each of the n input ports, there are,

potentially, 2^n possible causal patterns. The n outputs of this multi-port are the co-energy variables; i.e. each flow (effort) input has a collocated effort (flow) output. The input and output vectors u and y respectively are then:

$$u = \begin{bmatrix} e_1 \\ \cdot \\ e_m \\ f_{m+1} \\ \cdot \\ f_n \end{bmatrix} \quad \text{and} \quad y = \begin{bmatrix} f_1 \\ \cdot \\ f_m \\ e_{m+1} \\ \cdot \\ e_n \end{bmatrix} \tag{3.1}$$

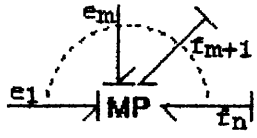


Figure 3.4 Bond graph representation of multi-port

Since the inputs and outputs are collocated, the field representation corresponds to an impedance/admittance matrix rather than a general transfer matrix.

i.e. $y = \phi u$ (3.2)

In the linear case, the field can be inverted to give

$$u = \phi^{-1} y \tag{3.3}$$

as long as $\det\phi \neq 0$, in which case valid multi-port constitutive relations exist with each input causality reversed.

Example: Inversion of an R-field

Considering again the example R-field representation of an electrical resistor network analysed in section 2.6.1, figure 3.5b shows the bond graph for the 2-port resistor network with each port driven by a voltage source.

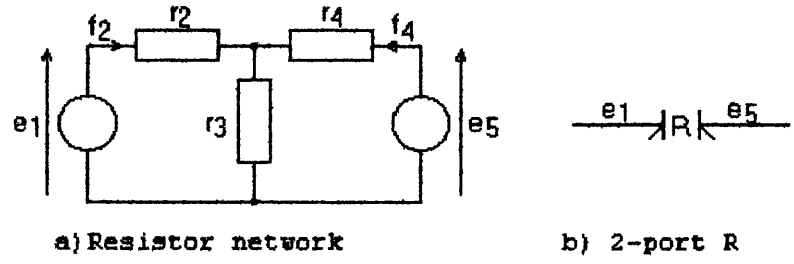


Figure 3.5 Resistor network driven by voltage sources

The system equations for the R-field give a conductance matrix:

$$\begin{bmatrix} f_2 \\ f_4 \end{bmatrix} = \frac{1}{d} \begin{bmatrix} (R_3+R_4) & -R_3 \\ -R_3 & (R_2+R_3) \end{bmatrix} \begin{bmatrix} e_1 \\ e_5 \end{bmatrix} \qquad (3.4)$$

where denominator $d = (R_2R_3 + R_2R_4 + R_3R_4)$.

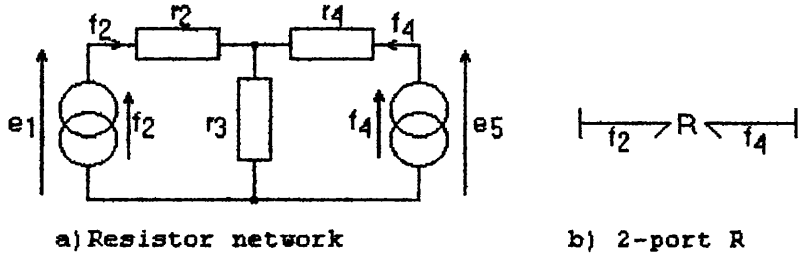


Figure 3.6 Resistor network driven by current sources

Reversing the causality of both ports as shown in figure 3.6 results in a resistance matrix which may be obtained by inverting the conductance matrix, since $\det\phi = d \neq 0$:

$$\begin{bmatrix} e_1 \\ e_5 \end{bmatrix} = \begin{bmatrix} (R_2+R_3) & R_3 \\ R_3 & (R_3+R_4) \end{bmatrix} \begin{bmatrix} f_2 \\ f_4 \end{bmatrix} \qquad (3.5)$$

However, although the field has proved to be invertible, some care should be exercised in the use of this algorithm as special cases can still make the system unrealisable. For this example, removing R3 leaves R2 and R4 in series between two independent current sources, which only results in a realisable system if f4 = - f2. This condition is of course a special case of equation 3.5, where R3 is infinite, demonstrating that invertibility of the field is a necessary condition, which may not be sufficient to guarantee the reversed causality is realisable.

3.3.2. A general algorithm to test for invertibility[34]

More generally, it is useful to be able to assess whether a valid constitutive relationship (CR field) exists for *any* given combination of input causalities.

Given the CR for one causal pattern, we may wish to test whether the CR for another causal pattern exists, and, if so, derive it. In general, m of the outputs (y) will be the same as before and we will put these into an m-vector y_1, and the rest of the outputs into the (n-m)-vector y_2. The complementary decomposition of u is also constructed.

The new output and input vectors can, after re-arrangement, be written as:

$$Y = \begin{bmatrix} y_1 \\ u_2 \end{bmatrix} \text{ and } U = \begin{bmatrix} u_1 \\ y_2 \end{bmatrix} \tag{3.6}$$

We require the n x n matrix Φ such that

$$Y = \Phi U \tag{3.7}$$

3.3.2.1. Algorithm

1. Choose the vector Y of n outputs, and U of n inputs
2. Re-arrange the original CR to be of the form
$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{3.8}$$
where ϕ_{ij} is the ijth sub-matrix of ϕ, appropriately partitioned.
3. If ϕ_{22} is singular, then the desired causal form cannot exist.
4. If ϕ_{22} is not singular, then the desired causal form has a CR:
$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \phi_{11}-\phi_{12}\phi_{22}^{-1}\phi_{21} & \phi_{12}\phi_{22}^{-1} \\ -\phi_{22}^{-1}\phi_{21} & \phi_{22}^{-1} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

That is

$$\Phi = \begin{bmatrix} \phi_{11}-\phi_{12}\phi_{22}^{-1}\phi_{21} & \phi_{12}\phi_{22}^{-1} \\ -\phi_{22}^{-1}\phi_{21} & \phi_{22}^{-1} \end{bmatrix} \tag{3.9}$$

Example: Transformer

A transformer is a two-port component with potentially 4 CRs, one of which is:

$$\begin{bmatrix} e_2 \\ f_1 \end{bmatrix} = \begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix} \begin{bmatrix} e_1 \\ f_2 \end{bmatrix} \tag{3.10}$$

Choosing the entire output vector for the sub-vector u_2

$$u = u_2 = \begin{bmatrix} e_2 \\ f_1 \end{bmatrix} \tag{3.11}$$

$$y = y_2 = \begin{bmatrix} e_2 \\ f_1 \end{bmatrix} \tag{3.12}$$

then $\phi_{22} = \begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix}$ (3.13)

and $y = \phi_{22}^{-1}u = \begin{bmatrix} 1/k & 0 \\ 0 & 1/k \end{bmatrix} u$ (3.14)

Thus this CR exists and so does the corresponding causal form. However, choosing

$$y1 = e2,\ y2 = e1,\ u1 = f2,\ u2 = f1 \tag{3.15}$$

gives

$$\phi = \begin{bmatrix} 0 & k \\ k & 0 \end{bmatrix} \tag{3.16}$$

where ϕ_{22} is zero, so the corresponding causal form does not exist for a two-port transformer.

3.3.3. Decomposition of multi-ports

Despite the elegance of collecting related constitutive laws into a multi-port, it may, in some cases, be more informative to decompose the multi-port into a collection of interacting one-port elements. Multi-bond theory (section 3.4) provides systematic methods for decomposing multi-ports into arrays of interconnected one-port elements.

The following examples show cases where advantage may be gained from decomposing multi-ports into a collection of one-ports, to reveal the under-lying mechanisms.

Example: A piezo-electric transducer

Recent research[35] has developed the bond graph model for a piezo-electric transducer, shown in figure 3.7. The piezo-electric transducer is represented by a 2-port C with inertance I=m. The effort source represents the force applied to the transducer, while the flow source represents an electrical current source.

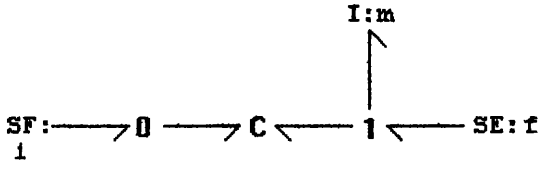


Figure 3.7 Multi-port representation of a piezo-electric transducer

The same authors have, however, developed this model further by decomposing the 2-port capacitance into its electrical and mechanical sub-components, as shown in figure 3.8.

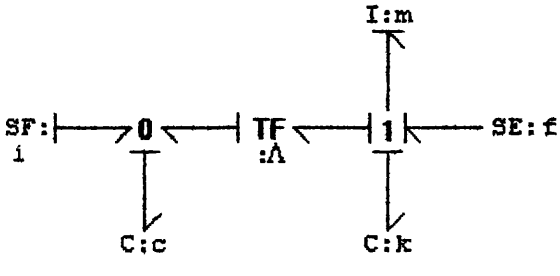


Figure 3.8 Decomposition of a piezo-electric transducer

The separation into the electrical domain on the left, and the mechanical domain on the right of the transformer helps to clarify the model, by separating the electrical capacitance from the mechanical stiffness, and highlighting the interaction mechanism.

Example: Heated tank

In section 2.7.2 (chapter 2) 2-port R and C components were used to model a heated tank system. These 2-port abstractions were used to describe the interaction between the hydraulic and thermal domains of this model. This results in an elegant encapsulation of the mechanisms involved, but in so doing, hides details which may be of interest to the researcher.

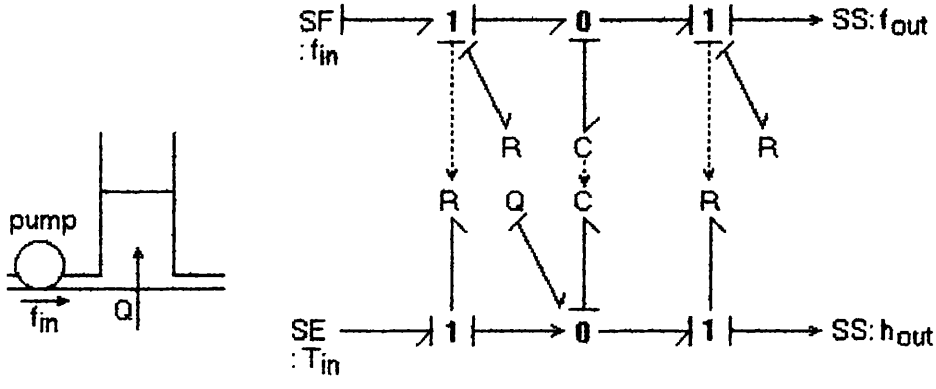


Figure 3.9 Decomposed bond graph model of heated tank

Figure 3.9, by comparison, makes all this detail explicit by emphasising the separation between the hydraulic and thermal domains.

In particular, the model highlights the fact that the interactions are unidirectional, as represented by the modulations. The thermal R elements are modulated by the hydraulic flow rates:

$$\frac{dh}{dt} = (c_p \frac{dm}{dt}) T_{in} \qquad (3.17)$$

The thermal capacitance is modulated by the state (mass) of the hydraulic capacitance:

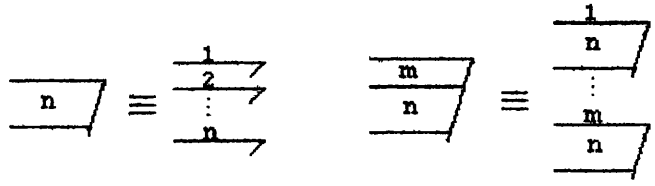
$$h = (c_p.m).T \qquad (3.18)$$

In real physical systems, the constitutive law of an energy store cannot be modulated, since this would imply an instantaneous change in stored energy when the modulation changed, and hence for an energy bond graph, a two-port capacitance would be an appropriate model for linking the hydraulic and thermal parts of this model. However, for this pseudo bond graph model it is more informative to show the unidirectional influence of the hydraulic state variable on the thermal capacitance.

3.4. Multi-bond representation

Multi-bond theory has been thoroughly developed[12,36] as a generalised extension of standard (single) bond graph theory. The multi-bond notation and its application are reviewed in this section, since the concepts complement those of the multi-port representation, discussed above.

The graphical notation is shown in figure 3.10a, where it can be seen that one `n-bond' is equivalent to an array of n `single bonds'. Similarly, the multi-bond notation has been extended to allow the possibility of nesting multi-bonds together in a multi-bond array, as shown in figure 3.10b



a) Multi-bond b) Multi-bond array

Figure 3.10 Multi-bond notation

The notation does not permit multi-bonds to have mixed causality, and thus a single causal stroke at the end of the multi-bond informs the reader that each included bond has the same causality. Figure 3.11 shows propagation of causality through an array of n 0-junctions, using a multi-bond array equivalent to two multi-bonds of dimension n. This figure also shows two ways of representing multi-port elements in a multi-bond graph. The most convenient notation for arrays of junctions is to underline the junction type;

e.g. 0 or 1 indicate arrays of 0- or 1-junctions respectively. For other multi-ports, the prefix MP is used; e.g. MP C, MP R etc.

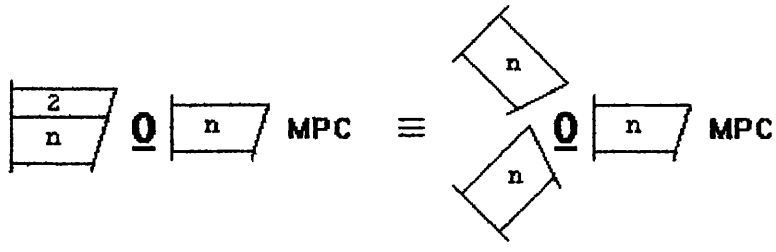


Figure 3.11 Integral causality on a multi-port C

Similarly, individual multi-bonds in nested multi-bond arrays must all have the same causality. The notation has been fully documented[18], including references to its usage with multi-port elements and, in particular, junction structures.

3.4.1. Decomposition of multi-bonds

It has been shown in Chapter 2 that a *simple junction structure* (SJS) consisting of 0- and 1-junctions may be considered as a multi-port element. Similarly, a *weighted junction structure* (WJS) - an SJS which also contains transformer elements (weighted bonds) - may also be manipulated as a single multi-port.

The theory of multi-bonds is based on the ability to decompose such multi-port elements into a so-called *analytic junction structure* (AJS), connected to basic 1-port elements. The derived AJS, composed of 1- and 2-port elements, is equivalent to the original multi-port, in the same way that a Thevenin equivalent `looks' the same to the electrical network from which it has been extracted.

Decomposition of general multi-ports into an AJS uses the decomposition of multi-port transformers as the basis of each equivalent AJS, and so this is considered here briefly. Figure 3.12 shows the decomposition of a multi-port transformer into the standard (single) bond representation.

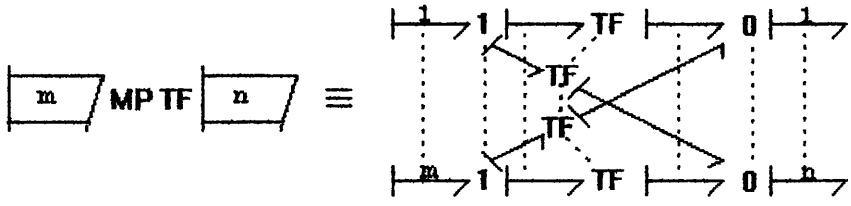
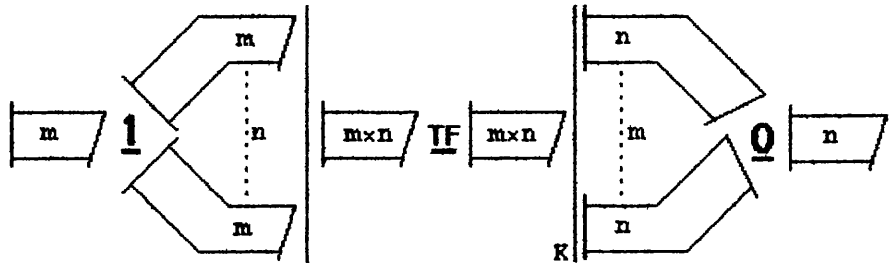
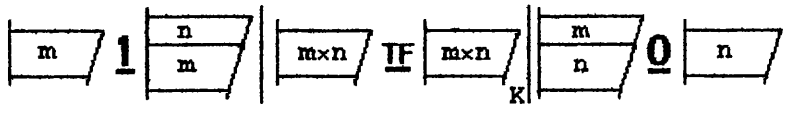


Figure 3.12 Multi-bond and single bond representations of a multi-port transformer

The single bond decomposition shown above may be represented using multi-bond notation as indicated in figure 3.13a, while this diagram is compacted further by use of multi-bond arrays, as shown in figure 3.13b. The vertical lines represent direct sums which relate the 2-port transformer array to the multi-bond arrays.



a) Multi-bond decomposition



b) Multi-bond array decomposition

Figure 3.13 Decomposition of multi-port transformer.

This technique may then be used to permit the decomposition of any other multi-port elements i.e. multi-port R's, C's, I's or GY's. A typical decomposition of a multi-port R is shown in figure 3.14. This decomposition may also be shown using multi-bond arrays, as demonstrated for the multi-port transformer (figure 3.13b).

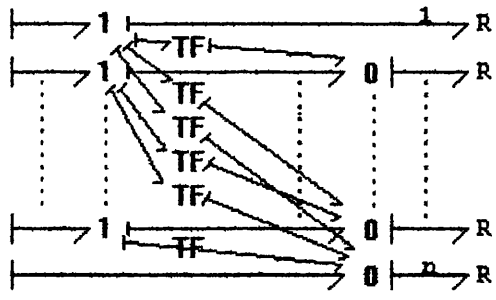


Figure 3.14 Decomposition of multi-port R

Such standard techniques are useful for systematic decomposition of multi-ports, but serve to highlight the fundamental weakness of the multi-bond notation. The problem is that decomposition is often necessary to gain better insight into the physical mechanisms, and also to obtain the required physical models.

It has been shown that the causality of interconnecting bonds (as well as that of the bonds within a sub-model) is entirely dependent on both the exogenous inputs and the required derived models. Since the notation requires that each bond represented within a given multi-bond has the same causal orientation, it is generally necessary to decompose the multi-bonds before analysis can progress.

The benefit of the multi-bond notation is, therefore, limited to providing a concise acausal representation of large systems. Causal augmentation will generally require decomposition to a `flat' (i.e. non-hierarchical) bond graph.

3.5. Hierarchical word bond graphs.

Bond graph notation provides a means for describing hierarchical systems by aggregation into word bond graphs, which are, in general, multi-ports. Word bond graphs are analogous to block diagrams, but since interconnections are generally by energy bonds rather than signals, these well-defined interfaces between the nodes on the word bond graph facilitate modelling the *interactions* between sub-systems.

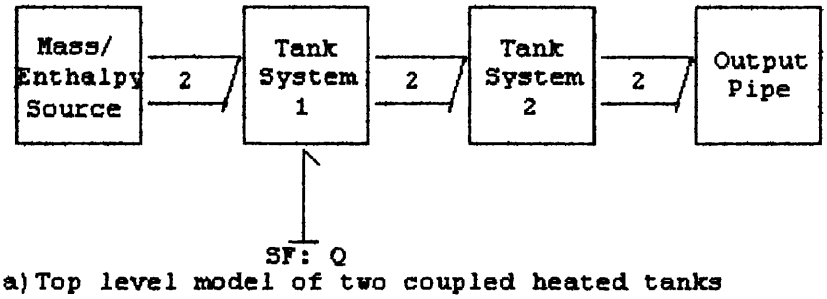
It has been shown that the causality of a system is not determined until the exogenous inputs are fully defined, so it can be seen that saving sub-models as acausal bond graphs makes the sub-models more general. Acausal bond graphs meet the requirements for the core model representation that the model is uncommitted until the application has been defined. Modelling a system graphically permits the model to be interpreted declaratively, i.e. without pre-defining an execution order. For the purposes of hierarchical system modelling we specify that a node in a word bond graph is acausal, thus retaining the declarative form, until the full model is causally augmented. The acausal sub-model implies that all the constitutive relations within that sub-model must be in a symbolic declarative (equation based) form. Similarly, where multi-bonds prove useful to describe interconnections between word bond graph nodes, these are also acausal.

It has been noted that bond graphs provide a natural means for describing the reticulation of a system into its structural and behavioural components. This also applies in hierarchical models where individual bonds and multi-bonds precisely describe the interfaces between word bond nodes (sub-models). Subsequent aggregation of each functional area into an acausal word bond graph, permits nesting of sub-models, ad infinitum. When the inputs to the top level sub-model are defined, and the form of the derived model is specified, these constraints may be propagated through the bond graph by causal augmentation rules. *A bond graph model differs from the equivalent sub-model only in that its causality has been fully defined, by exogenous inputs.*

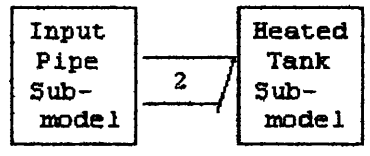
Example: Heated tank system

Figure 3.15 shows the use of word bond graphs for the hierarchical decomposition of a system consisting of two interacting tanks of heated liquid. The model is built from two sub-models described in section 3.3.3 (figure 3.9), and is fully analysed by Gawthrop[37]. The multi-bonds in this case represent both the hydraulic flow and the thermal flow between the multi-port nodes. The single `tank system' sub-model is decomposed into two further sub-models - `hydro-thermal pipe' and `basic tank' in figure 3.15b. The final

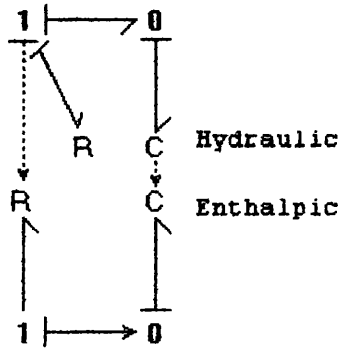
decomposition of `tank system' to a `flat' bond graph is that shown in figure 3.15c.



a) Top level model of two coupled heated tanks



b) Decomposition of Tank System



c) Bond graph of Tank System

Figure 3.15 Word bond graph of heated tank system

This example illustrates the requirement for nesting of sub-models and the re-use of each of the pipe, basic tank and tank system sub-models. Obviously, these properties are not intrinsic to bond graphs, although bond graphs have features which facilitate their implementation.

3.5.1. Parameters and symbolic representation

For hierarchical models, parameters are vital to the re-use of sub-models, since the same sub-model structure may be instantiated several times with different physical parameters. This leads naturally to the use of symbolic models in preference to numeric models which are, at present more common for simulation and analysis tools. Symbolic models have the additional advantage that they are easier for humans to analyse, and can give a greater insight into the operation of the modelled system than a pure numerical representation. The numeric model can be considered as a transformation of the symbolic core model representation. It might,

for example, be useful to convert all the parameters except one to numeric form so that the significance of the remaining parameter is highlighted in the resulting derived model.

Symbolic names for parameters are helpful in making models self-documenting, so that future users of the model can easily understand the concepts used to develop the model. For this reason, modelling tools should provide an extensible library of templates for the constitutive relations used in all models. If we use the example of a uniform tank containing an incompressible liquid, the constitutive relation relating the pressure at the bottom of the tank to the volume of liquid can be given as:

$$\text{pressure} = \frac{1}{C} \int \text{flow}\ dt \quad = \frac{\text{volume}}{C}$$

where C is the capacity.

However, it is much more informative to express C in terms of the physical parameters, which constitute it,

i.e. C = area/(density * gravity)

and, hence

pressure = volume * (density * gravity)/area

Providing a library of such relationships eases model development, and simultaneously encourages the modeller to document his work, making re-use of models more likely.

Parameterised models are also essential for describing non-linear constitutive relations and the relations of modulated components such as gyrators and transformers. If it is necessary to model combined continuous and discrete-event systems, then it must also be possible to express certain constitutive relations as (possibly non-linear) functions of time.

3.5.2. Rules for building hierarchical word bond graphs

The above points can be summarised into a set of rules which permit bond graph sub-models to be re-used in a hierarchical model. The rules are based on a `top-down' analysis of a large system, although real modelling often iterates between `top-down' and` bottom-up' approaches.

1. Generate the word bond graph representing the major components of the system to be modelled.
2. Decompose each complex node (sub-model) in the word bond graph into a further word bond graph, or `flat' bond graph, as appropriate.
3. Repeat step 2 until the largest node in each word bond graph can be easily modelled using a bond graph.
4. For each sub-model so produced, formulate the *acausal* bond graph (or re-use sub-models from a library).
5. Define all constitutive relations in each sub-model using symbolic parameters.
6. Test each sub-model individually, to verify its behaviour.
7. Repeat steps 4 to 6 until all acausal sub-models have been generated and tested.
8. Aggregate all sub-models into a `flat' bond graph of the complete hierarchical word bond graph.
9. Apply required inputs to the complete bond graph.
10. Apply the causal initiations appropriate to the required derived model.
11. Follow causal propagation rules to obtain the ordered equations for the derived model.

3.6. Defined and typed Input/Output terminals

Information hiding is a software concept which helps to make software more secure, essentially by hiding irrelevant information from the user. This is also applicable to model building, and modellers should try to minimise the number of data items in a

model which are accessible from its external environment. Even then it may be possible to apply the wrong type of source to an input, or to wrongly interpret an output type. An electronic example of such an error would be applying a current generator to a high input impedance amplifier - at best, the amplifier input would saturate, at worst it would be destroyed. Obviously, giving terminals a specific (effort or flow) type declaration can prevent such errors, but it may also be desirable to provide a higher level of data typing in addition to this.

By contrast, it is often quite reasonable to apply either an effort or flow source to an input terminal as shown in some previous bond graph examples. In hydraulic systems, a tank might be filled either from a flow source into the open top of the tank or a pressure source applied at the base of the tank. It is then important that the models are represented in a declarative fashion (rather than assignment form), so that they may be evaluated according to the required causality.

A higher level of data-typing has been proposed by Mattsson[38], using *terminal attributes*, which would then be checked for consistency between interconnected terminals. Terminal attributes give specialising information about the terminals of a model, such as the domain type (e.g. electrical), units, valid range and additional documentation such as pipe diameters etc. This information may be omitted for generic models which may be used in a variety of energy domains.

It is the author's belief that, although providing terminal attributes for sub-models makes the re-use of models less error-prone, the information hiding concept may result in other problems. In particular, the user may be interested in different variables within the model than those provided by the original modeller. It would obviously detract from the re-use of models if internal variables could not be accessed after the original encapsulation. This highlights another advantage of the word bond graph approach to hierarchical modelling, in that each model is ultimately reduced to a flat bond graph model. The user can then allocate further outputs (or inputs) to the flat bond graph to meet his particular

modelling needs. This approach offers a compromise between minimising the interface at the word bond graph level and allowing access to the model for specific analyses.

3.7. Conclusion

Bond graphs have been proposed as a suitable notation for a core model representation, since they offer several advantages:

- Close correspondence between the bond graph and the physical system.
- The bond graph can be created before causality is considered, and so the core model is not constrained by the application of the model, or the chosen inputs to the system.
- Acausal bond graphs provide a declarative, symbolic representation of the system.
- Bond graphs are concise and may be systematically interpreted.
- Bond graphs provide a unified description of systems which include multiple energy domains.

Three different approaches to using bond graphs to represent hierarchical systems have been examined:

The multi-port representation was shown to be unsuitable for modelling hierarchical systems where the causality is not fixed. This loses the main advantage of bond graphs over block diagrams for representing hierarchical systems, and requires constitutive relations to be (partially) inverted. An algorithm providing a generalised test for invertibility of multi-port fields was examined. It was shown, by example, that this algorithm was not sufficient to guarantee invertibility for all parameter values.

The multi-bond notation can be used to model systems where causality is not fixed, but only where the modeller is prepared to perform significant decompositions on the multi-bond representation to achieve this. The complexity of the notation also sacrifices the close mapping of `flat' bond graphs to the physical system.

The word bond graph was originally defined merely as a means of aggregating bond graph fragments into functional blocks within the complete system. This chapter has proposed extending this notation to permit hierarchical nesting of nodes (sub-models) of a word bond graph. The declarative, symbolic nature of bond graphs may then be preserved in the word bond graph model, by ensuring that word bond sub-models are acausal. The constitutive equations of each element comprising a word bond sub-model must be expressed in declarative (equation-based) form with symbolic parameters. Arbitrary causalities may then be applied to the `flat' bond graph model resulting from decomposition of the acausal word bond graph.

Application of strict data-typing to sub-model interface bonds, and reducing the number of interface ports was proposed, in order to minimise errors when re-using sub-models. The automatic flattening of the word bond graph to a conventional bond graph for causal augmentation ensures that all bonds are accessible for non-standard analyses at one point in the modelling process.

CHAPTER 4 CAUSAL AUGMENTATION OF BOND GRAPHS WITH ALGEBRAIC LOOPS

4.1. Introduction

Previous research[39,40,41] has laid down the theory for bond graphs which are causally complete, and, in particular, Auslander[42] has pointed out that 'the selection of a tree in a system graph is the exact counterpart of causal assignment in a bond graph'. Since the systematic assignment of causality to bonds in a graph results in an ordered set of system equations, algorithms for completing causal augmentation of bond graphs are of particular interest. Such algorithms may be viewed at the highest level as composed of two distinct functions: the first of which propagates causal constraints through the bond graph according to the rules described in the previous chapter, and the second which handles exceptions to these rules.

These functions are outlined as:

1. Using the causal constraints imposed by the bond graph structure, propagate the causality implied by sources and storage elements through the bond graph.

There are three possibilities resulting from this.

a) The constraints imposed imply that the causalities assigned to one or more of the stores lead to causal conflict - this we term an over-causal model.

b) There is no causal conflict and all bonds have causality assigned - the model is causally complete.

c) There is no causal conflict but some bonds do not have assigned causality - this we term an under-causal model.

In the first case, the bond graph model and/or the desired causalities may need to be rethought. In the second case the algorithm terminates, and a complete set of system equations may be obtained. In the third case, the second part of the algorithm is executed.

2. Identify an intermediate variable, with an assigned causality, and propagate this through the junction structure. Repeat until all bonds have been causally assigned.

This chapter introduces a new approach[43] to the second part of this algorithm, which complements the strengths of bond graphs by revealing the intermediate variable(s) chosen to complete causal propagation, and permits the incorporation of rules to minimise the number of resulting algebraic loops.

It is stated, and illustrated by example, in the texts[9,14,32] that each iteration of part two of the algorithm corresponds to an algebraic loop in the system equations. In other words, if part two has to be executed the system is not described by an ordinary differential equation (ODE), but rather by a differential algebraic equation (DAE)[45-47]. In the solution of such systems, an intermediate variable must be chosen and solved either symbolically by an algebraic method or numerically by iteration. The difficulty of this solution depends on the *index* of the DAE[45-46], but a fuller discussion is beyond the scope of this thesis. Suffice it to say that simple algebraic loops which can, in principle, be solved for the intermediate variables without differentiation correspond to index one DAEs and have well-established solution techniques. All DAEs arising from the dynamic examples in this chapter are index one DAEs.

The second part of the algorithm can be avoided altogether if the system model is suitably modified by the addition of small dynamic elements to break the algebraic loops[32,48], but a different system is now being analysed. The addition of such elements, we believe, should be for physical modelling reasons rather than merely for expediting equation solution. In some cases moreover, algebraic loops arise from deliberate system approximation based on removing small dynamic elements.

The following section first identifies situations in which under-causal models arise, and then reviews standard implementations of part two of the above algorithm. A new algorithm for completing causal propagation of under-causal models is introduced, which

explicitly identifies the intermediate variable arising associated with the algebraic loop at the bond graph level. Section 4.3 shows how this new algorithm may be specialised to formulate the steady-state equations from a bond graph model. The applicability of this approach to systematic derivation of a set of DAEs from an under-causal bond graph is illustrated in section 4.4, while section 4.5 summarises the chapter.

4.2. Assigning causality to under-causal models

Under-causal models arise either when there are insufficient constraints imposed on the bond graph junction structure, or due to the topology of the junction structure itself.

Bond graphs with closely coupled dissipators (R-fields) lack the causal constraints necessary to complete causal propagation through the graph. This may occur in a dynamic model, as a valid representation of the physical system, or, typically, when dynamic elements are `removed' to derive the steady state model.

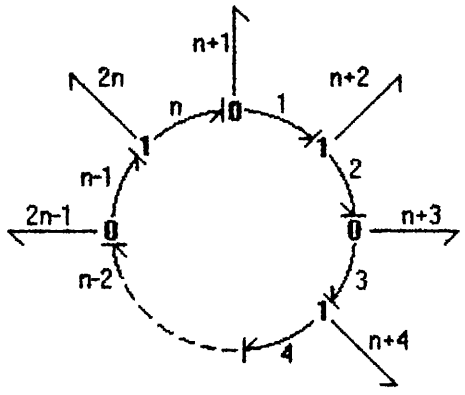


Figure 4.1 A generalised causal loop

Under-causal models can also occur as a result of topological loops in the junction structure[40,41], which may also include transformer elements within the loop. Ort & Martens have named such a loop a *causal loop* (figure 4.1) - a closed loop in a graph composed of internal bonds, where every junction in the loop has a strong causal determination given to it by an internal bond in the loop. Figure 4.1 shows a non-redundant junction structure where 0- and 1-

junctions alternate, hence causal loops must include an even number **n** of (non-redundant) junctions. There may, of course, be more than one external bond attached to each junction.

Similar topological loops can occur if modulated elements are included in the bond graph. Such loops are possible with all types of modulated element, making these hard to identify in complex bond graphs. A deficiency of the standard SCAP algorithm is that it does not identify loops due to modulations and, therefore, cannot be guaranteed to produce correct ordering of the system equations in this case. This will be discussed further in the following chapter.

The following sections describe various techniques available for deriving the system equations for examples of bond graphs which are under-causal due to either R-fields or causal loops.

4.2.1. Standard solutions

The standard algorithm for causal augmentation of bond graphs is the Sequential Causality Assignment Procedure (SCAP) due to Karnopp and Rosenberg[17], and described in Chapter 2 of this thesis. Part 1 of this algorithm is common (with occasional specialisations) to all bond graph causality assignment procedures.

Part 2 of the SCAP algorithm is as follows:

Choose an *arbitrary* causality on any unassigned bond and propagate causality from this through the junction structure. Repeat until all bonds have causality assigned.

This algorithm, and those that follow, are illustrated by an example taken from standard bond graph textbooks[9]. This example is the electrical circuit illustrated in figure 4.2a

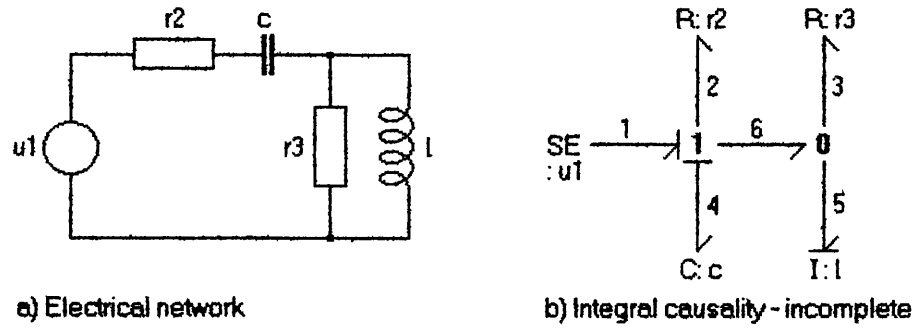


Figure 4.2 Electrical circuit resulting in algebraic loop

Figure 4.2b shows the causal augmentation after integral causality has been propagated from the storage elements, i.e. part 1 of the algorithm has terminated with three bonds still unassigned. Part 2 of the algorithm then permits the modeller to assign either causality to any of the unassigned bonds. In this case, either of the two causal augmentations shown in figure 4.3 would be possible and valid, resulting in two equivalent sets of ordered equations.

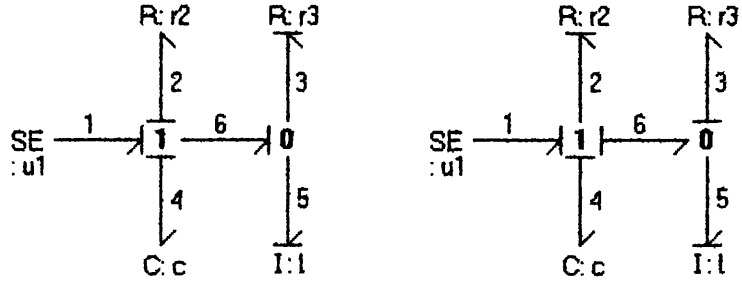


Figure 4.3 Causal completion options

It can be seen that the SCAP algorithm for handling under-causal bond graphs, is unsystematic, and, furthermore, does not give any indication of the choice of intermediate variable made to complete the set of equations. Lastly, it will be shown in the following section that arbitrary choice of the intermediate variable, may result in more than the minimal number of algebraic loops.

4.2.1.1. Minimising the number of algebraic loops.

It is stated in the texts[9,14] that one algebraic loop occurs for each repetition of part 2 of the causality algorithm, and since each loop may be computationally intensive, it is desirable to minimise the number which occur.

Lorenz and Wolper[14], have demonstrated, by comparison with equivalent signal flow graphs, that the arbitrary assignment of causality in part 2 of the SCAP algorithm may result in more intermediate variables than necessary to complete the set of system equations. They have developed rules to minimise the number of algebraic loops, and these rules are repeated here:

Rule 2 If there exist some causal uncertainties on internal bonds, choose one of those bonds that allows a causal assignment which is strong at both ends of the bond (eventually through a TF or GY element), then break the computational loop on any of the two variables associated with that bond. Otherwise use Rule 1.

Rule 1 Break the causal uncertainty on any bond but give the junction a strong causal determination, then break the computational loop on any of the two variables associated with that bond.

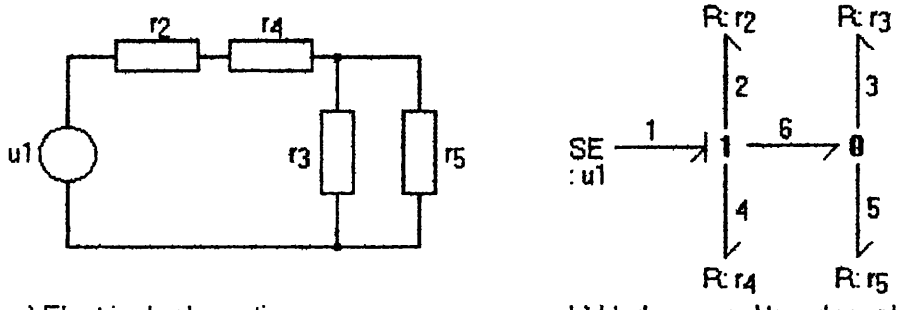
Rule 1 is fairly trivial, in that choosing causality on an unassigned bond, such that the causal determination of the attached junction is weak, could immediately result in causality failing to propagate further. An additional algebraic equation will then be required to describe the system, as shown in the following example.

A key observation for the application of rule 2 is that internal bonds which connect two junctions of the same type are redundant. For example, an internal bond connecting two 1-junctions may be removed, and the two junctions merged to give a single 1-junction with the same incident bonds. Thus, it can be seen that all minimal junction structures (excluding TF and GY elements) consist of an alternate sequence of 1- and 0-junctions. Rule 2 therefore states that causality should be propagated from a non-redundant internal

bond, by either defining the effort on the attached 0-junction, or the flow on the attached 1-junction.

Example: An electrical resistor network

The example used in the previous section, to illustrate the SCAP approach to handling under-causal bond graphs can only result in one algebraic loop. A similar electrical network, having the same bond graph junction structure but having both storage elements replaced by dissipators, is shown in figure 4.4. Figure 4.4b shows the bond graph after part 1 of the causal augmentation procedure has terminated - five bonds are still unassigned.



a) Electrical schematic b) Under-causal bond graph

Figure 4.4 Resistor network resulting in bond graph R-field

Using part 2 of the SCAP algorithm, can result in two intermediate variables being required to complete causality, as indicated in column 2 of Table 4.1, where the chosen variables are f2 and e3.

By comparison, using Lorenz and Wolper's rule results in only one algebraic loop, by choosing f6 as the intermediate variable - the equations derived following this causal propagation path are listed in column 3 of table 4.1.

The tables are used to illustrate the order in which the bond equations are generated whilst following the causality propagation rules. The required system equation is derived algorithmically by selecting the bond equation with the required variable on the left hand side, and then progressing back, sequentially, through the bond equations in the table. The computer implementation of this is thus analogous to pushing the equations onto a stack whilst

causality is being propagated, and popping them off to solve for a specific variable.

#	Column 2 - SCAP	Column 3 -Lorenz & Wolper
1	e1 = u	e1 = u
2	f2 = i (intermediate var.)	f6 = i (intermediate var.)
3	f1 = f2	f1 = f6
4	f4 = f2	f2 = f6
5	e4 = f4r4	e2 = f2r2
6	f6 = f2	f4 = f6
7	e3 = v (intermediate var.)	e4 = f4r4
8	e5 = e3	e6 = e1-e2-e4
9	f5 = e5/r5	e3 = e6
10	f3 = f6-f5	f3 = e3/r3
11	e3 = f3r3	e5 =e6
12	e6 = e3	f5 = e5/r5
13	e2 = e1-e4-e6	f6 = f3+f5
14	f2 = e2/r2	

Table 4.1 Ordered equations showing algebraic loops

Example: A sun and planet gear system

The models considered thus far were examples of under-causal bond graphs resulting from R-fields. In this example, the algebraic loop results from a causal loop in the bond graph junction structure; in this case including a transformer element within the loop.

This rotational mechanics example is taken from the bond graph literature[9,32], and represents a sun and planet gear system with compliance. The bond graph is illustrated in figure 4.5a, with integral causality propagated as far as possible using part 1 of the SCAP algorithm - four bonds have unassigned causality. Figure 4.5b indicates the complete causal augmentation which results from applying the Lorenz and Wolper algorithm; in this case, internal bond 6 defines the flow on the 1-junction.

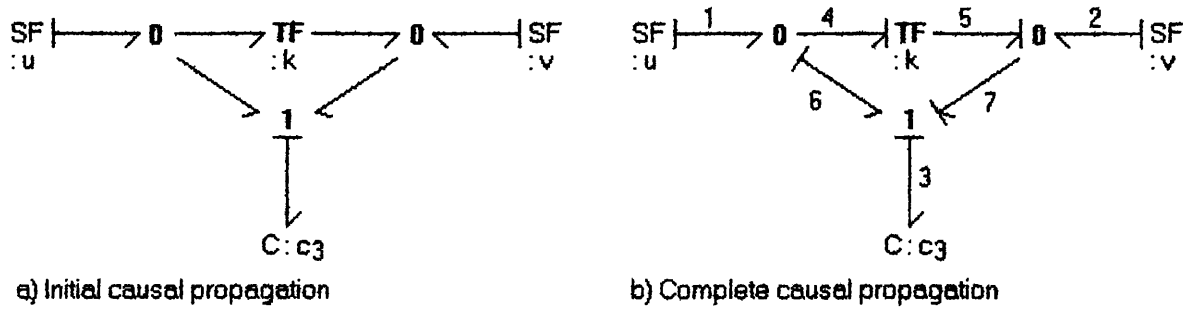


Figure 4.5 Bond graph of sun & planet gear system

This example is interesting in that neither the SCAP nor the Lorenz and Wolper algorithm reveal that the model requires *two* intermediate variables, in order to obtain the complete set of system equations. This highlights a limitation of the standard graphical representation of causality on bond graphs, in that the causal direction is defined by the causal stoke, but *this does not imply that both the effort and flow equations on the bond are known*. This is an important limitation when evaluating algorithms for automatic assignment of bond graph causality.

#	Effort/flow equations
1	f1 = u
2	f2 = v
3	e3 = q/c
4	f6 = w (intermediate variable)
5	f3 = f6
6	f7 = f6
7	f5 = f7-f2
8	f4 = kf5
9	f6 =f1-f4
10	e6 = t (intermediate variable)
11	e1 = e6
12	e4 = e6
13	e5 = ke4
14	e2 = e5
15	e7 = e5
16	e6 = e3-e7

Table 4.2 Ordered equations for sun and planet gears

Table 4.2 shows that selecting f6 as the intermediate variable results in all the flow equations (Table 4.2, rows 1 to 9) being defined for the system, but the only effort equation is that due to integral causality on c3. At this stage the graphical causality appears complete, despite the fact that seven system equations have yet to be derived. The remaining equations can easily be derived, however, by propagating effort causality from bond 6 in the opposite direction round the causal loop. This example shows that neither the SCAP, nor the Lorenz and Wolper algorithm gives a complete graphical description of the causal augmentation of systems containing causal loops.

4.2.2. A new algorithm

As pointed out by Karnopp[15], an important aspect of bond graph modelling is that the equation structure is clearly defined by the bond graph *before* the equations are explicitly formulated. The author believes that the second part of the classic causality algorithm detracts from this feature: the intermediate variables arising from the algebraic loop are not explicitly shown on the bond graph, and the choice of such variables is made during equation formulation rather than at the bond graph level.

In the previous examples, one or more intermediate variables have been selected as an implied input to the bond graph, defining a constraint which is then propagated through the graph. For this new algorithm, we choose to make this input explicit by adding an appropriate source, and a constraint equation which ensures that the additional source does not disturb the system model.

Assuming that the bond graph is proper (all bonds impinge on a junction) then at least one junction in an under-causal graph does not have causality imposed on it. That is, a causally incomplete 0-junction does not have an effort imposed on it, or a causally incomplete 1-junction does not have a flow imposed on it. An appropriate source (an effort source for a 0-junction; a flow source for a 1-junction) can then be attached to the junction, and the causalities propagated throughout the graph. This procedure can be repeated until the bond graph is causally complete. It can be

seen that this approach to part 2 of the causality algorithm is very similar to Lorenz and Wolper's rule 1, but has the advantage of making the choice of intermediate variable explicit on the bond graph.

We now have a causal bond graph, corresponding to the original system, but with **n** new input sources. However, an effort source connected to a 0-junction has no effect on the system if the source effort is such that the flow into the source is zero; the junction is at its natural effort. A more general way of expressing this is that the effort sources are constrained such that the effort imposed is equal to that resulting from the unmodified system. A similar statement may be made about flow sources added on 1-junctions.

The result of the algorithm is to add **n** additional sources to the system, with source output u_i and source input y_i. The system thus has **n** additional inputs u_i which have no effect on the system if they are chosen such that the n implicit algebraic equations

$$y_i = 0 \tag{4.1}$$

are satisfied.

Thus the **n** additional inputs u_i lead to **n** new unknown variables which can be found by solving the **n** additional equations (4.1).

In models where the bond graph is under-causal due to a causal loop in the junction structure, the constraint used to solve the algebraic loop is the more general one that the new effort (flow) source must equal the natural effort (flow) imposed on that 0- (1-) junction. Thus, the criterion expressed in equation (4.1) is still valid, but becomes one of the bond equations produced while propagating causality through the loop.

4.2.2.1. Example: Electrical circuit resulting in algebraic loop

Considering, firstly, the example given in section 4.2.1, we can show how the new algorithm highlights the choice of intermediate variable made by the modeller.

Figure 4.6a shows the bond graph after part 1 of the SCAP algorithm has terminated (for integral causality); causality is not complete and, in particular, neither junction is causally complete.

In part 2 of the new algorithm, we can either add an effort (voltage) source to the 0-junction or a flow (current) source to the 1-junction. Either completes causality.

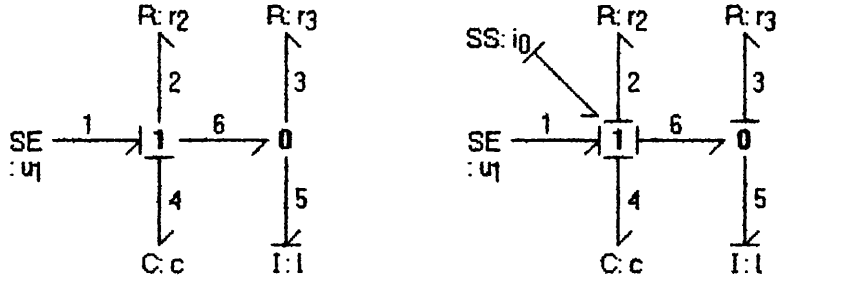


Figure 4.6 Bond graph of electrical network

Addition of a flow source i_0 to the 1-junction gives rise to the causally complete bond graph of figure 4.6b; the input (e_0) to the additional source is the additional system output, which is to be set to zero. The additional source has been represented by an `SS' element to emphasise that there is an implicit sensor associated with the additional source.

The system variables can now be systematically assigned from the bond graph:

states $$x = \begin{bmatrix} q \\ p \end{bmatrix} \qquad (4.2)$$

system inputs $u = \begin{bmatrix} u_1 \\ i_0 \end{bmatrix}$ (4.3)

system outputs $y = \begin{bmatrix} f_1 \\ e_0 \end{bmatrix}$ (4.4)

the corresponding state equations are derived from the ordered bond equations:

$$dq/dt = f_4 = i_0$$

$$dp/dt = e_5 = (i_0 - p/I)r3$$

$$y_1 = i_0$$

$$y_2 = q/C - pr3/I + i_0(r2+r3) - u_1 \quad (4.5)$$

To these must be added the additional constraint that

$$y_2 \ (= e_0) = 0 \quad (4.6)$$

In this particular case, equations 4.5 and 4.6 are linear, and may be explicitly solved to give the state equations:

$$x = \begin{bmatrix} q \\ p \end{bmatrix}$$

$$dx_1/dt = \frac{1}{r2+r3}(u_1 - x_1/C + x_2 r3/I)$$

$$dx_2/dt = \frac{r3}{r2+r3}(u_1 - x_1/C - x_2 r2/I)$$

$$y_1 = \frac{1}{r2+r3}(u_1 - x_1/C + x_2 r3/I) \quad (4.7)$$

In the general non-linear case, however, an algebraic solution may not be possible and a numerical solution would be required.

As has been shown in section 2.1, there are two valid ways in which causality can be completed for this bond graph. The second

alternative is shown in figure 4.7, which uses an additional effort source v_7 applied to the 0-junction to complete causality[43].

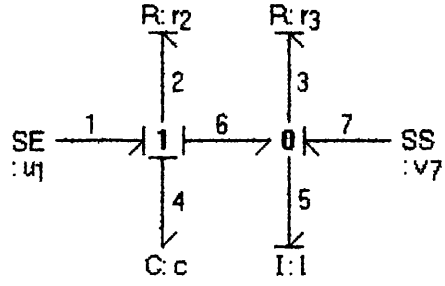


Figure 4.7 Causal completion with additional effort source

4.2.2.2. Example: An electrical resistor network

The electrical resistor network is shown in figure 4.4, with incomplete causality after terminating part 1 of the causality algorithm. Applying the new algorithm for part 2 of the causal augmentation, requires the modeller to choose one of the causally incomplete junctions, and add an appropriate source to this junction.

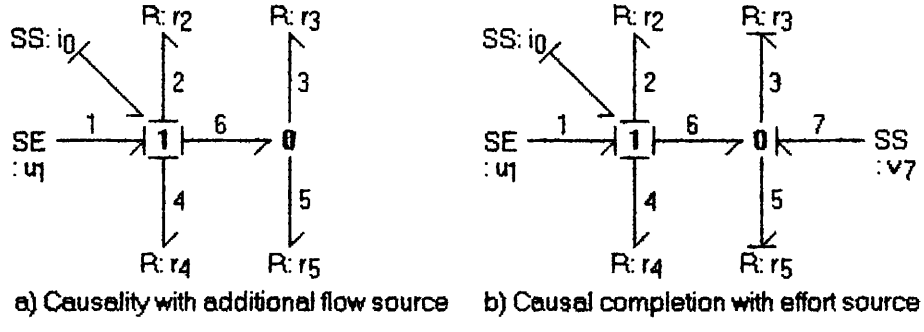


Figure 4.8 Causal completion of resistor network model by adding sources

Figure 4.8a indicates that the result of adding a flow source (i_0) to the 1-junction is that causality is still incomplete. An effort source (v_7) must be added to the 0-junction, in order to complete causality, as shown in figure 4.8b.

From the causally complete bond graph, the system variables may be assigned as:

states $x = ()$ i.e. none

system inputs $u = \begin{bmatrix} u_1 \\ i_0 \\ v_7 \end{bmatrix}$ (4.8)

system outputs $y = \begin{bmatrix} f_1 \\ e_0 \\ f_7 \end{bmatrix}$ (4.9)

the corresponding system equations are:

$$y_1 = f_1 = +u_2$$

$$y_2 = e_0 = -u_1 + u_2(r2+r4) + u_3 \quad (4.10)$$

$$y_3 = f_7 = -u_2 rp + u_3 \quad (4.11)$$

where $rp = r3r5/(r3+r5)$

The additional constraints, $y_2 = 0$ and $y_3 = 0$ give:

$$u_3 = u_2 rp$$

and $$u_2 = -u_1/(r2+r4+rp) \quad (4.12)$$

In section 4.2.1.1 it was shown that, by applying both rules of the Lorenz and Wolper algorithm, the system equations for this resistor network have a minimum of one algebraic loop. This example indicates that the new algorithm is equivalent to applying only rule 1 of the Lorenz and Wolper algorithm, and does not, in general, minimise the number of algebraic equations. In the following chapter it will be shown that this new algorithm can be used to give the minimum number of algebraic loops, by using another new extension to bond graph causality theory.

4.2.2.3. Example: A sun and planet gear system

Applying the new algorithm to the sun and planet system model, the modeller may choose to add a flow source to the 1-junction in order to complete causality, as shown in figure 4.9a.

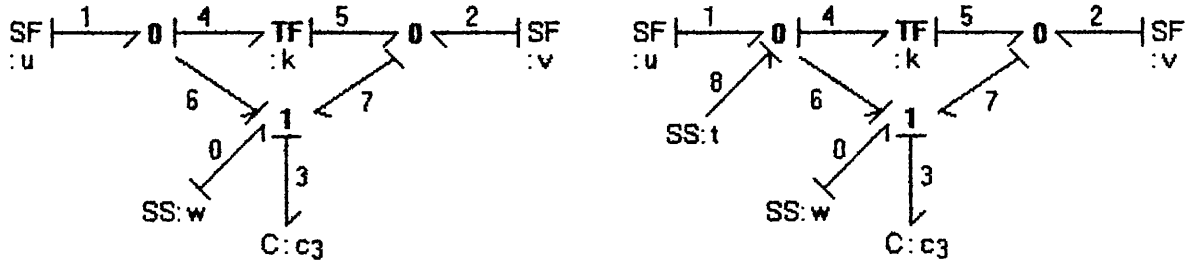


Figure 4.9 Causal completion of gear model by adding sources

#	Effort/flow equations
1	f1 = u
2	f2 = v
3	e3 = q/c
4	f0 = w (intermediate variable)
5	f3 = f0
6	f6 = f0
7	f4 = f1-f6
8	f5 = f4/k
9	f7 = f2+f5
10	f0 = f7 (constraint equation)
11	e0 = 0 (constraint equation)
12	e8 = t (intermediate variable)
13	e6 = e8
14	e7 = e3-e0-e6
15	e2 = e7
16	e5 = e7
17	e4 = e5/k
18	e1 = e4
19	e8 = e4 (constraint equation)
20	f8 = 0 (constraint equation)

Table 4.3 Ordered equations for sun and planet gears

The consequence of propagating causality through the system is that the equations appear in the order given in table 4.3, and a causal conflict appears to arise on the 1-junction, due to the added flow source.

This causal conflict is resolved by the constraint that the input flow defined by the added source must be equal to the natural flow at the 1-junction in order that the system is not disturbed by adding the source. This appears as the constraint equation (10) in table 4.3. The second constraint equation associated with source is equation (11); e0 = 0.

As for the previous methods for completing causality, this new algorithm does not generate all the system equations from a single intermediate variable. Figure 4.9b illustrates the effect of applying an effort source to the 0-junction at the opposite end of bond 6, in addition to the source-sensor at the 1-junction. Again, graphical causality is complete, thus deriving all the remaining bond equations, with the apparent causal conflict between the added effort source and causality propagated back through bond 4. This is resolved, as before, by the constraint equation (19) which requires that the source effort be equal to the natural effort at that 0-junction, with the result that the corresponding flow constraint equation (20) is also valid.

The new algorithm has again been shown to highlight each of the chosen intermediate variables on the bond graph, so that completing causality becomes explicit as a bond graph manipulation rather than an implicit algebraic manipulation. In addition, for bond graphs with causal loops, the constraint equation is made explicit on the bond graph, by appearing as a causal conflict at the junction where the intermediate variable source is added.

4.3. Steady-state analysis

Breedveld[16] defines the equilibrium state of a system as that state where all time-varying derivatives of the state variables are zero in the absence of time-varying disturbances. In that paper, an algorithm to determine the equilibrium state of a system was

proposed; replacing the energy stores in the system by zero value sources with causality the same as derivative causality on the respective stores. In this section, an equivalent algorithm is examined which retains the bond graph in its original form, but assumes zero value energy stores which must also force derivative causality on the system.

A further alternative algorithm[43] is proposed which uses the constraint propagation technique discussed in the preceding sections. The advantage of this latter algorithm is that the causal propagation and resulting bond equations applicable to the dynamic system are re-used, with additional equations to specify the constraint that all time-varying derivatives of the state variables are zero.

4.3.1. A new algorithm for steady-state analysis

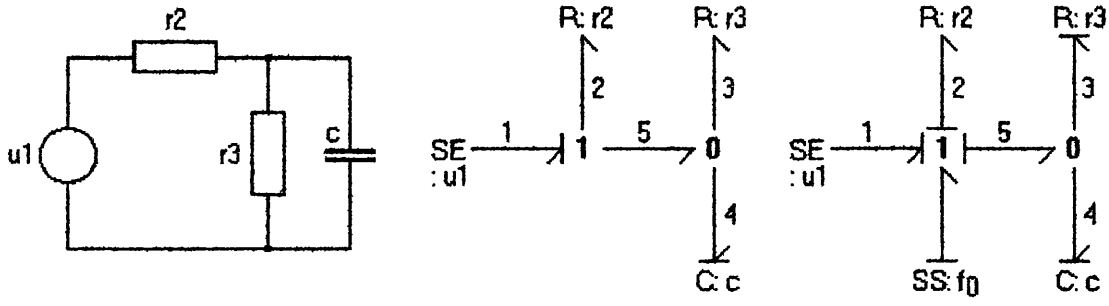
1. Complete causal augmentation for the dynamic system, for integral causality, using the algorithm proposed in section 2.2.
2. Replace all store components by sources with the same causality.
3. Sources corresponding to stores with derivative causality have zero output.
4. The **m** sources corresponding to stores with integral causality have zero input (*to* each source), and lead to **m** additional algebraic equations expressing this constraint.

By construction, the resultant bond graph has complete causality, and this causality is identical to that of the dynamic system. There are now no state equations, but the additional sources lead to **m** new algebraic equations to be solved.

4.3.2. Example: Simple electrical circuit

In this example the two new techniques for deriving the steady state model are contrasted for a simple electrical circuit (figure 4.10a) with a single storage element (capacitance).

For the first technique, the capacitance is set to zero (c = 0), and the resulting bond graph of this system is shown in figure 4.10b, augmented for derivative causality.



a) Electrical schematic b) Under-causal graph c) Causal completion

Figure 4.10 Simple electrical system

It can be seen that the bond graph is under-causal with derivative causality applied to the capacitance, resulting in an algebraic loop. In this example, causality is completed (figure 4.10c), using the new causal completion algorithm, by adding an auxiliary input, SS = f_0. In order that the system is unchanged, this requires the additional constraint equation

$$y = e_0 = 0 \tag{4.13}$$

i.e. $e_0 = e_2 + e_5 - e_1 = 0$

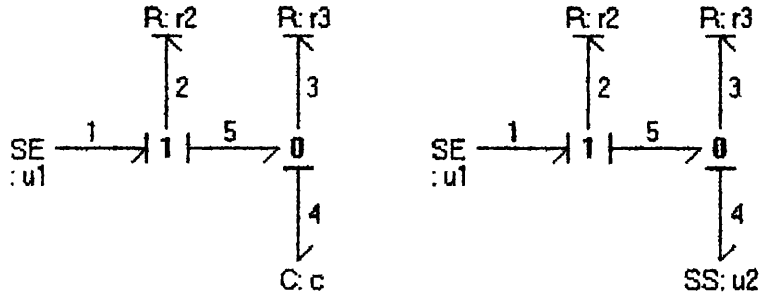
which gives

$$e_1 = f_0(r_2 + r_3)$$

and hence, the steady state voltage across c is:

$$e_4 = f_0 r_3 = u_1 \frac{r_3}{(r_2 + r_3)} \tag{4.14}$$

The alternative algorithm for deriving the steady state equations of this system is illustrated in the bond graphs of figure 4.11. The first part of the algorithm (section 4.3.1) requires causal augmentation to be completed after applying integral causality to the energy stores. In this case, this results in a causally complete bond graph as shown in figure 4.11a.



a) Integral causality b) Steady state model

Figure 4.11 Steady state solution using the new algorithm

The energy store, c, is then replaced by a source-sensor element having the same causality which results in the steady state model of figure 4.11b. The system variables are then assigned as:

states $x = ()$ i.e. none

system inputs $u = \begin{bmatrix} e_1 \\ e_4 \end{bmatrix}$ (4.15)

system outputs $y = \begin{bmatrix} f_1 \\ f_4 \end{bmatrix}$ (4.16)

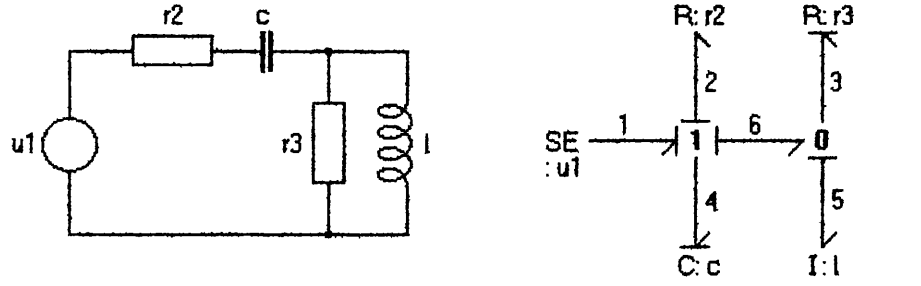
The steady state equations for specific variables may be obtained from the ordered equations derived for the dynamic model with integral causality, together with the constraint equation

$$y_2 = f_4 = 0 \tag{4.17}$$

This example has shown the potential advantage of the new algorithm for deriving steady state models from a bond graph where the dynamic model is known to be causally complete with integral causality applied to the energy stores. In this case, there is no need to apply the second part of the causal augmentation algorithm, and there is no algebraic loop.

4.3.3. Example: Electrical circuit

The system chosen for this example (figure 4.12a) is the electrical circuit which was shown in section 4.2.1 to be under-causal when analysed with integral causality applied to the energy stores.



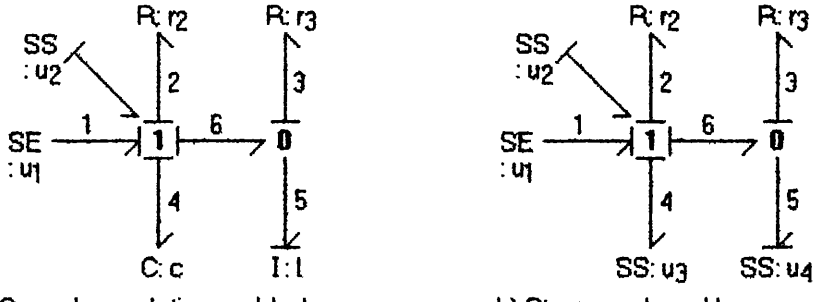
a) Electrical network b) Derivative causality - complete

Figure 4.12 Electrical circuit with derivative causality

Applying derivative causality to each of the (zero value) stores gives the causally complete bond graph shown in figure 4.12b. The bond equations, ordered according to this causal propagation sequence, are listed in Table 4.4 column 1. As before, the system equations can be automatically derived by choosing the required variable and working back through the list of bond equations.

e.g. e4 = e1 - e2 - e6 eventually gives: e4 = u

The new method for steady-state analysis (algorithm 4.3.1) follows the same causal propagation as used for generating a dynamic model (figure 4.13a). Once this causal propagation is complete, the energy stores are replaced by sources with the same causality, as indicated in figure 4.13b.



a) Causal completion - added source b) Stores replaced by sources

Figure 4.11 Electrical circuit with integral causality

A comparison of the causal propagation sequences for the two methods is shown in Table 4.4.

#	Derivative causality	Sources replacing stores
1	e1 = u	e1 = u1
2	f4 = 0	e4 = u3
3	f1 = f4	f5 = u4
4	f2 = f4	f0 = u2
5	e2 = f2r2	f1 = f0
6	f6 = f4	f2 = f0
7	e5 = 0	e2 = f2r2
8	e3 = e5	f4 = f0
9	f3 = e3/r3	f6 = f0
10	f5 = f6-f3	f3 = f6-f5
11	e6 = e5	e3 = f3r3
12	e4 = e1-e2-e6	e5 = e3
13		e6 = e3
14		e0 = e1-e2-e4-e6

Table 4.4 Ordered equations for steady-state models

Using the new algorithm and the notation adopted when analysing the equivalent dynamic system, the system variables can be systematically assigned from the bond graph:

states $x = ()$

system inputs $u = \begin{bmatrix} e_1 \\ f_0 \\ e_4 \\ f_5 \end{bmatrix}$ (4.18)

system outputs $y = \begin{bmatrix} f_1 \\ e_0 \\ f_4 \\ e_5 \end{bmatrix}$ (4.19)

There is one effort/flow constraint equation for each of the flow/effort sources which replace the energy stores:

$$y_3\ (= f_4) = 0 \quad \text{and} \quad y_4\ (= e_5) = 0$$

in addition to the constraint equation

$$y_2 \ (= e_0) = 0$$

required to ensure that the input added to complete causality does not disturb the system.

The corresponding system output equations are derived from the bond equations:

$$y_1 = u_2$$

$$y_2 = e_1 - e_2 - e_4 - e_6 = u_1 - u_3$$

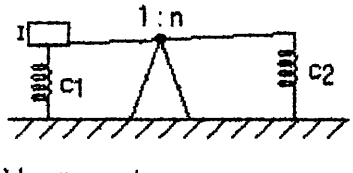
$$y_3 = u_2$$

$$y_4 = e_3 = (u_2 - u_4)r3 \qquad (4.20)$$

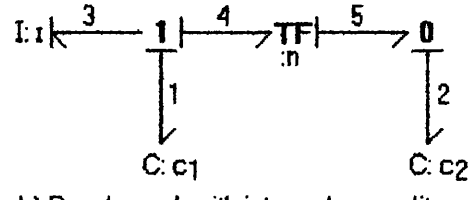
Thus, an equation can be derived for any required system variable by applying the constraints to equations (4.20).

4.3.4. Example: Equilibrium state of a lever system

This example has been used[16] to demonstrate a bond graph method due to Breedveld for determining the equilibrium state of a system. The mechanical system is illustrated in figure 4.14a, with the bond graph shown in figure 4.14b, fully augmented for integral causality.



a) Lever system

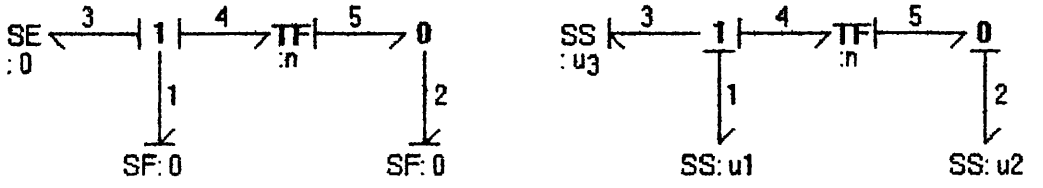


b) Bond graph with integral causality

Figure 4.14 Lever system

Using Breedveld's method, the capacitances and inertances are replaced by zero-valued auxiliary flow- and effort-sources, respectively (equivalent to derivative causality on the original stores). The resulting bond graph, representing the equivalent

steady-state system, is shown in figure 4.15a and has a causal conflict, but since both flows on the 0-junction are zero there is no actual conflict. This indicates that "the junction structure is indeterminate, i.e. the constitutive relations are dependent. In case the constitutive equations are inconsistent, there is no equilibrium state, but in case these equations are consistent there are infinitely many equilibrium states." Thus Breedveld uses the causal conflict to identify that one of these situations has occurred, and thereafter checks that the implied constraints are consistent, in order to evaluate if an equilibrium state exists.



a) Breedveld steady-state model b) Constraint-based method

Figure 4.15 Steady-state bond graphs for lever system

Figure 4.15b shows the bond graph resulting from applying the new constraint-based algorithm to the same system. The energy stores have been replaced by source-sensor elements having the same causality as was given the stores for the dynamic system analysis. The resulting system variables are:

states $x = ()$

system inputs $u = \begin{bmatrix} e_1 \\ e_2 \\ f_3 \end{bmatrix}$ (4.21)

system outputs $y = \begin{bmatrix} f_1 \\ f_2 \\ e_3 \end{bmatrix}$ (4.22)

The bond equations reduce to:

$$y_1 = u_3$$

$$y_2 = nu_3$$

$$y_3 = -(u_1 + u_2)$$

where the equilibrium state is defined by the additional constraints:

$$y_1 = y_2 = y_3 = 0$$

hence $f_3 = 0$ and $e_1 = -ne_2$, giving infinitely many equilibrium values.

Considering the physical implications of this result can lead to some confusion, until it is made clear that the infinite number of equilibrium values result from the infinite number of possible initial conditions. This is clarified by decomposing e_1 and e_2 to show the initial conditions in the linear case:

$$\frac{q_1}{c_1} + e_{01} = -n\frac{q_2}{c_2} - ne_{02}$$

where

$$e_{01} = - ne_{02}$$

This example is now extended to illustrate the case (figure 4.16a) where no equilibrium state can exist, although the dynamic system bond graph (figure 4.16b) is causally complete.

Again Breedveld's method indicates a causal conflict for the equivalent steady-state bond graph (figure 4.16c), but, in this case, some algebraic analysis is needed to show that the flows imposed on the 0-junction can only be consistent if $f_0 = 0$.

By contrast, the new method must give a causally complete steady-state bond graph (figure 4.16d), since the causal pattern is identical to that for the dynamic system bond graph. The system variables now include the inputs and outputs to the forcing velocity source f_0. The constraints

$$y_1 = y_2 = y_3 = 0$$

result in a steady-state solution only in the condition that the input $f_0 = 0$.

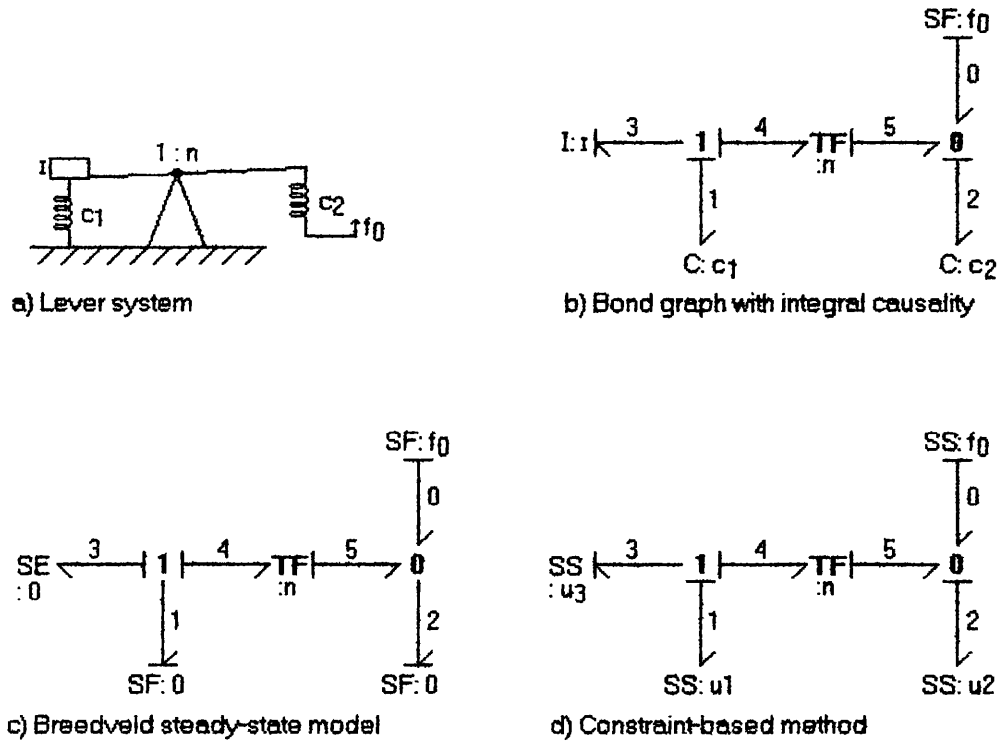


Figure 4.16 Lever system with forcing velocity source.

4.3.5. Example: Current driven d.c. motor

The d.c. motor example is most often analysed in voltage driven mode, resulting in a model with two state variables due to the armature inductance and the motor shaft inertia, respectively. This example is designed to show the application of the new algorithm to a system where the dynamic model includes an energy store with derivative causality.

The bond graph shown in figure 4.17a, illustrates that current driving the motor results in the armature inductance having derivative causality, and thus the dynamic model has one non-state variable ($z = \lambda$). There is one state variable ($x = p$) associated with the shaft inertia. The input ($u = i$) is the armature current, and the output ($y = v$) is the shaft velocity.

The state equation is derived from the ordered bond equations:

$$dx/dt = x/\tau + ku$$

where $\tau = j/f$.

The non-state equation is:

$$0 = -z + lu$$

These equations may be manipulated to give the transfer function

$$\frac{Y}{u} = G(s) = \frac{k}{f(s\tau + 1)} \qquad (4.23)$$

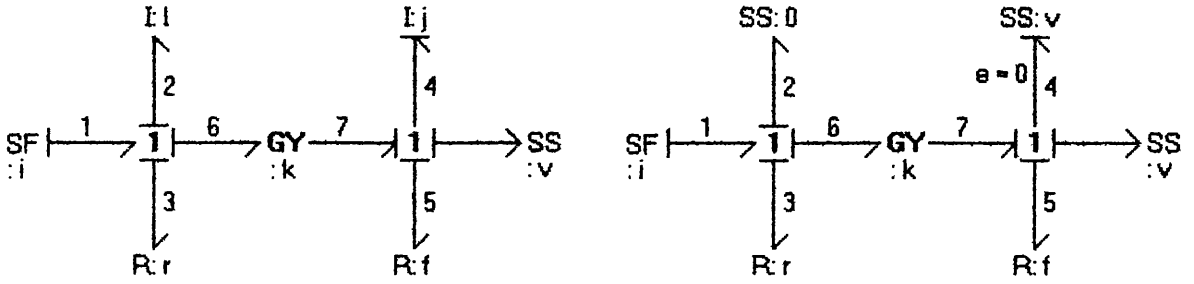


Figure 4.17 Bond graphs for current-driven d.c. motor

Figure 4.17b shows the steady state model of the same system, resulting from the application of the new algorithm. The I-store representing the armature inductance has been replaced by a source with zero output, since in the dynamic model this had derivative causality. The I-store representing the shaft inertia has been replaced by a source with zero input.

The resulting constraint equations are now :

$$e_2 = 0 = l\ df_2/dt$$

and $e_4 = 0 = ku - vf$

giving the steady-state gain as

$$\frac{Y}{u} = G(0) = \frac{k}{f}$$

4.4. D.A.E and generalised state equations

As has been shown, the causal bond graph of a system provides a systematic way of writing down a set of system equations. In the elementary case[9], for example , the system state derivative vector dx/dt and the system output y can be written down in terms of the system state x and the system input:

$$dx/dt = f(x,u)$$

$$y = g(x,u) \qquad (4.24)$$

where f(x,u) and g(x,u) represent, possibly non-linear, functions of their arguments.

In the linear case, equations (4.24) can be rewritten[9] as:

$$dx/dt = ax + bu$$

$$y = cx + du$$

where a, b, c and d are matrices of appropriate dimension.

The corresponding system transfer function is:

$$g(s) = c(sI - a)^{-1}b + d \qquad (4.25)$$

Such a transfer function cannot be improper: it cannot have a denominator of higher order than the numerator.

Equations (4.24) form an ordinary differential equation (ODE). As inverse systems may be improper, such equations cannot, in general, describe the inverse systems considered in this thesis. Hence a more general system representation is required.

In place of the state vector x, the descriptor vector X is used, where

$$X = \begin{bmatrix} x \\ z \\ z' \\ v \end{bmatrix} \tag{4.26}$$

and

a) x is the N_x x 1 vector of state variables associated with C and I energy stores with integral causality.

b) z is the N_z x 1 vector of non-state variables associated with C and I elements with derivative causality.

c) z' is the N_z x 1 vector containing the corresponding derivatives.

d) v is the N_v x 1 vector of additional inputs, where $N_v = N_n$, the number of non-collocated sensors.

As will be shown, the system equations can be represented in terms of X as:

$$E\,dX/dt = F(X,u)$$

$$y = G(X,u) \tag{4.27}$$

where

$$E = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \tag{4.28}$$

Where E is a square matrix of dimension $N_x + 2N_z + N_v$ and I is the unit matrix of dimension $N_x + N_z$. For simplicity, we will denote this particular form of E by:

$$E = I_0(N_x + N_z,\ N_z + N_v) \tag{4.29}$$

where $I_0(N_1,N_2)$ is a square $(N_1 + N_2)$ x $(N_1 + N_2)$ matrix with unit elements on the first N_1 diagonal elements and zeros elsewhere.

In general, E is singular (unless $N_z = N_v = 0$), and so equations (4.27) cannot be written as an ordinary differential equation (4.24). Such equations are variously called differential-algebraic equations (DAE's)[44,45,47] descriptor equations[46], singular

equations[48,49], or generalised state-space equations[48]. In the linear case, equations (4.27) become

$$E\dot{X} = AX + Bu$$

$$y = CX + Du \tag{4.30}$$

where A, B, C and D are matrices of appropriate dimension.

The corresponding transfer function is:

$$G(s) = C(sE - A)^{-1}B + D \tag{4.31}$$

Such a transfer function can, unlike that arising from an ODE, be improper: it can have a numerator of higher order than the denominator.

The functions F and G in (4.27) can be derived in a systematic way from the causal system bond graph by expanding (4.27) into

$$\dot{x} = F_x(x, z', v, u)$$

$$\dot{z} = z'$$

$$0 = -z + F_z(x, z', v, u)$$

$$0 = w = F_w(x, z', v, u)$$

$$y = G_y(x, z', v, u) \tag{4.32}$$

Thus each element of $\dot{x}$, $\dot{z}$, w and y is obtained in terms of x, z', v and u by following the bond graph causal strokes.

In the linear case, equations (4.32) become

$$\dot{x} = A_{xx}x + A_{xz}z' + A_{xw}v + B_x u$$

$$\dot{z} = z'$$

$$
\begin{aligned}
0 &= A_{zx}x - z + A_{zz}z' + A_{zw}v + B_z u \\
0 = w &= A_{wx}x + A_{wz}z' + A_{ww}v + B_w u \\
y &= A_{yx}x + A_{yz}z' + A_{yy}v + B_y u
\end{aligned}
\tag{4.33}
$$

Thus A, B , C and D are given by:

$$
A = \begin{bmatrix} A_{xx} & 0 & A_{xz} & A_{xw} \\ 0 & 0 & I & 0 \\ A_{zx} & -I & A_{zz} & A_{zw} \\ A_{wx} & 0 & A_{wz} & A_{ww} \end{bmatrix}
$$

$$
B = \begin{bmatrix} B_x \\ 0 \\ B_z \\ B_w \end{bmatrix}
$$

$$
C = (A_{yx} \quad 0 \quad A_{yz} \quad A_{yv})
$$

$$
D = (B_y)
$$

4.4.1. Example: Electrical circuit resulting in algebraic loop

Here we consider again the example analysed in section 4.2.2.1, which is described (equations 4.5) by two state equations and a single algebraic equation. This example illustrates the systematic transformation from a bond graph which has been causally augmented using the new method to D.A.E. representation.

Equation 4.26 illustrates that the descriptor vector is comprised of the state vector, the vector for the non-states and their derivatives and, lastly, a vector of the auxiliary inputs required to complete causality. Thus the descriptor vector for this system is derived from the system variable descriptions of equations 4.2 and 4.4.

$$
\text{i.e. } X = \begin{bmatrix} q \\ p \\ i_0 \end{bmatrix} \tag{4.34}
$$

where i_0 is the auxiliary input due to the algebraic loop, and there are no non-state variables.

The differential-algebraic equations (4.5) for this system are rewritten as:

$$dX_1/dt = i_0$$

$$dX_2/dt = (i_0 - p/l)r3$$

$$e_0 = 0 = q/c - pr3/l + i_0(r2+r3) - u_1 \qquad (4.35)$$

and the output equation is:

$$f_1 = i_0$$

These linear DAE's can be rewritten in generalised state equation form, where

$$A = \begin{bmatrix} 0 & 0 & 1 \\ 0 & -r3/l & r3 \\ 1/c & -r3/l & (r2+r3) \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}$$

$$C = (0\ 0\ 1)$$

$$D = 0$$

and,

$$E = I_0(2,1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Thus the transfer equation may be derived using (4.31) as

$$\frac{f_1}{u_1} = G(s) = (0\ 0\ 1) \begin{bmatrix} s & 0 & -1 \\ 0 & s+r3/l & -r3 \\ -1/c & r3/l & -(r2+r3) \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} + 0$$

which gives

$$G(s) = \frac{s\tau 2(1+s\tau)}{r2(s^2\tau(\tau 2+\tau 3)+s(\tau+\tau 2)+1)}$$

where $\tau = 1/r3$

$\tau 2 = cr2$

and $\tau 3 = cr3$

4.5. Conclusions

This chapter has described a new method for completing causality of under-causal models, and compared this with existing standard solutions to this problem. Using the new method, the modeller completes causality by bond graph manipulation rather than by algebraic manipulation. There are three significant advantages of this approach:

a) the choice of intermediate variables is made by the bond grapher at the bond graph level, and not left until equation formulation and solution stages,

b) the chosen intermediate variables become auxiliary inputs, explicitly shown on the bond graph, thus building on the graphical strengths of this technique for documenting the model,

c) the auxiliary inputs used to complete causality can be mapped systematically into the descriptor vector for solution using a range of methods developed for solving DAEs.

It has been shown, by example, that the new method does not minimise the number of intermediate variables, in comparison with that due to Lorenz and Wolper which therefore offers a computationally more efficient solution. The following chapter will show how another new technique can resolve this short-coming.

The new method has been extended to provide the equilibrium state of the model, by further bond graph manipulation. The advantage of this solution is that the causality of the steady-state bond graph

for the system is the same as that for the dynamic model of the system. This obviates the possibility of a non-invertible constitutive equation being encountered if a different causal propagation is required for the steady-state model.

CHAPTER 5 BICAUSAL BOND GRAPHS AND UNILATERAL BONDS

5.1. Introduction

In chapter 2, it was shown that a bond graph is a system representation which can provide a close mapping to the structure and component behaviours which occur in the real physical system. Bond graph causality augmentation permits automatic generation of mathematical models such as state equations, whilst providing additional insights into the realisability of that model. It will be shown in this chapter that specific mathematical models may be derived from a bond graph by *systematic selection of the appropriate causal initiation.* In this way, bond graphs successfully decouple the modelling process from the analysis/test process, and thus are well suited to be the core model representation from which a variety of formulations of system equations may be derived[15].

This chapter extends the concept of computational causality[50] to automate the derivation of equations for inverse system models. The inverse model of a dynamic system is that model which, given the system output at its input, will reproduce the system input at its output. Such models are useful for control system synthesis, using either feedforward or feedback techniques. One example of this is the computed-torque manipulator control technique[51] where the joint torques required to give a pre-specified manipulator trajectory are computed. The extended causality notation described in this chapter permits the additional system constraints implied by inverse models to be directly imposed on the bond graph, resulting in an augmented bond graph which explicitly identifies the states and non-states of the model.

The secondary purpose of this chapter is to relate the derived models to the differential-algebraic equation (or *generalised* state equation) representation, which permits the description of systems with improper transfer functions, and algebraic loops.

This chapter is illustrated by considering simple examples appropriate to the analysis of manipulators - a mass, spring and

damper system, and a manipulator arm. Section 5.2 of the chapter describes two extensions to the bond graph notation which permit the automatic derivation of inverse system models, whilst section 5.3 provides the causal augmentation procedure necessary to achieve this. An electrical example highlights the manner in which the new notation explicitly identifies the states of the inverse system model. The systematic conversion from bond graph representation to DAE model is shown in each case. Section 5.4 integrates the new concepts with the new algorithm for completing causality of under-causal bond graphs, and section 5.5 concludes the chapter.

5.2. New bond graph concepts

This section contrasts a purely computational view of causality of a system model with the conventional bond graph view. Whereas *computational* causality assigns effort or flow variables to the left- or right-hand-side of a constitutive relation (in assignment statement form), *computable causality* also identifies whether the variable on the right-hand-side of this relation is known.

This results in two related, but distinct, concepts:

- A compatible extension to the `causal stroke' notation is introduced to clarify the process of computable causality propagation.
- A novel concept is that of the *unilateral bond*, which is described using the new notation, and is shown to be essential for systematic causal analysis of inverse systems.

A notation for collocated source-sensor elements (SS) is introduced. `SS' elements unambiguously identify the system outputs, and are also required for systematic derivation of transfer functions and inverse system transfer functions.

5.2.1. Two views of causal propagation

A common element of systematic modelling methodologies for physical systems, is the concept of *pairs* of variables[8,10] which together define the energy flows throughout the system. In the bond graph

methodology, these variables are generalised as effort and flow variable pairs, and a notation has been developed[9,11] which describes the fundamental element behaviours.

When building a mathematical model of the system which the bond graph represents, causal augmentation is used to systematically derive the equations required for the chosen form of model. The first step must therefore be to decide on the form of the mathematical model required, i.e. which are the independent variables appearing on the right-hand-side (RHS) of a mathematical model expressed in assignment form. The augmentation process starts with known independent variables and propagates these known variables through the graph using the defined causality rules, until the effort and flow variables on every bond are known.

The bond graph convention uses a causal stroke at one end of each interconnecting power bond to indicate which node is imposing the effort/flow across the bond. The causal stroke is placed at the end of the bond closest to the node which has the effort imposed upon it. Figure 5.1 illustrates this notation, where Node J has an effort e imposed upon it and, as a corollary, imposes a flow back on Node K.

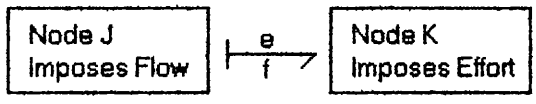


Figure 5.1 Graphical causal notation

This results in a very concise graphical notation for detecting causal conflicts within the model, *but at the expense of lost information*. The bond graph displays the causal direction on each augmented bond, but gives no information regarding which of the effort or flow variables is known, at any point in the augmentation process. (The direction of the half arrow assigns the positive direction of power flow and is irrelevant to causality.)

Thus, a more complete view of causal augmentation is required in order to use the technique to automatically derive mathematical

models from a bond graph model. In addition, it is useful to extend the graphical notation, in order to give the modeller a clearer view of the state of progress while propagating computable causality. Thus it is proposed that, in order to describe computable causal augmentation, the meaning of the causal stroke becomes: *`the effort on this bond is known, and this effort is being imposed on the node nearest the stroke'*. A further addition is a dot at the end of the bond, which is added when *`the flow on this bond is known, and this flow is being imposed on the node nearest the dot'*. The extended notation is illustrated for a causally complete bond graph, in figure 5.2, indicating the *bilateral* causal effects.

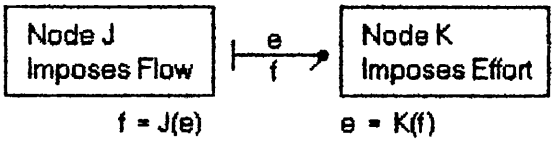


Figure 5.2 Extended graphical causal notation

Using the Pascal assignment operator instead of an `equals' sign to emphasise that each RHS is known, the notation shown in figure 5.2 explicitly states:

$$e_j = e := K(f), \text{ and}$$

$$f_k = f := J(e).$$

The new causal dot notation makes the distinction between flow-driven causality and the (conventional) effort-driven causality. It will be shown in the following sections that this notation is not tautology, but proves to be essential in deriving mathematical models of inverse systems.

5.2.2. Collocated sources and sensors

In order to systematically derive transfer functions, we must now introduce a general source-sensor element `SS', to identify the inputs and outputs of the system model, and *explicitly* define the causality of the bond connecting the source-sensor to the system.

The `SS' element is used in two ways - first as a pure sensor with causality defined by an activated bond, and secondly as a source with reversible causality.

No standard notation exists for defining the outputs of the system, although this has, in some cases, been indicated by an activated bond (signal) to some undetermined sensor. Another approach is to identify the sensor as a flow/effort source where the flow/effort variable is set to zero, and the resulting effort/flow becomes the system output (figure 5.3a) and b)). This is a better defined sensor, but requires the additional f=0 (or e=0) written on the graph to explicitly destinguish it from an input.

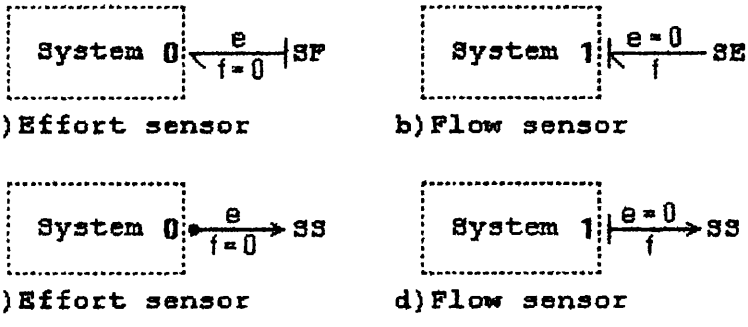


Figure 5.3 Ideal output measurements.

The `SS' element is proposed to explicitly indicate the location of system outputs on the bond graph. An activated bond attached from the appropriate system junction element to the `SS' element defines this as a system output since the appropriate covariable *must* be zero. Figure 5.3c shows an effort measurement which must be taken from a `0' junction in the system. The output e is measured using an `SS' element, which imposes zero flow back on the system due to the signal connection, i.e. an ideal (zero power) measurement. Here the additional variable definitions are made explicit by the signal, and are added just to emphasise the argument.

Measurement of the flow f at a `1' junction is shown in figure 5.3d. Experienced bond graphers will note that, so far, this has merely formalised the use of signals to identify system outputs, for systematic evaluation. This is the first application of the `SS' element.

The second use of the `SS' element is to identify where, in some models, the outputs of the system may be *collocated* with the system inputs, i.e. the sensor measures the other member of the effort-flow pair associated with the corresponding source or system input. By definition, such systems have the same number of inputs and outputs. An example of such a system appears in section 5.2.2.2.

In bond graph terms, a collocated source-sensor pair can be regarded as a source element with the corresponding measurement being an input to the source element. To emphasise the twofold role of the source, a collocated source-sensor will be denoted `SS' on the bond graph as in Figure 5.4. The left-hand SS component of Figure 5.4 represents an *effort source* e_1 with a collocated *flow measurement* f_1; the right-hand SS component represents a flow source f_2 with a collocated effort measurement e_2.

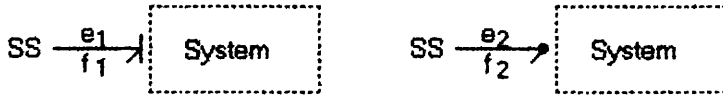


Figure 5.4 Source-sensor component

In such a system, then, the system inputs are the source outputs (as indicated by the causal stroke/dot), and the system outputs are inputs to that source. The inverse system, that is the system where the inputs and outputs are interchanged, is simply obtained by reversing the causality on each of the source-sensors.

The bond graph notation provides elements to represent system inputs in the form of effort/flow sources (`SE' or `SF' elements, respectively), which have fixed causality, e.g. an SE element can only impose an effort on the system, and the system determines the flow on the effort source. For inverse systems, however, we wish to know the effort that an effort source is constrained to provide for a given system output. Consequently, we now need a source-sensor element which can have reversible causality, such that a computer algorithm can propagate causality without the resultant causal conflict on the source (now sensor).

5.2.2.1. Algorithm for inverting systems with collocated source-sensors

The algorithm for deriving the bond graph of the inverse system of a system with N collocated sensors and sources is then:

a) Represent the N system input/output pairs by source-sensor (SS) elements.

b) The bond graph of inverse system is obtained by reversing the causality of the N source-sensor (SS) elements.

5.2.2.2. Example: Mass, spring, and damper system

Figure 5.5 shows a simple mechanical system with an external force F(t) imposed horizontally on a sliding mass attached to a spring and damper.

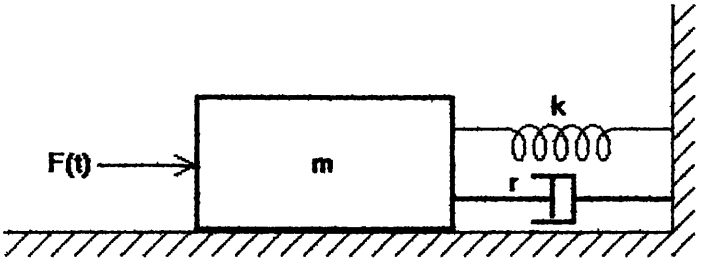


Figure 5.5 Mass, spring and damper system

If one wishes to derive the state equation of this system, then the known variables on the RHS of the derived system equations are the system input, and the states of the energy stores. Causal propagation then starts from the fixed causality effort source, and then from each of the energy stores, with integral causality applied to them. The derived mathematical model is then the state equation:

$$\frac{d}{dt}\begin{bmatrix} q \\ p \end{bmatrix} = \begin{bmatrix} 0 & 1/m \\ -k & -r/m \end{bmatrix} \begin{bmatrix} q \\ p \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} F \qquad (5.1)$$

where q is the state variable for the spring, and p is the state variable of the inertia.

The corresponding causal system bond graph is given in Figure 5.6a, where integral causality is applied to the energy stores.

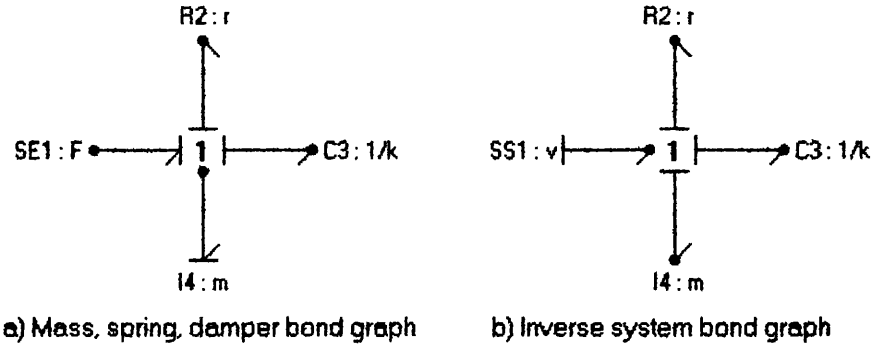


Figure 5.6 System and inverse system bond graphs

Using the algorithm of section 2.1.1 gives the bond graph of the inverse system in figure 5.6b. This causal augmentation pattern provides the solution to the problem: what is the force F(t) required to achieve the velocity v(t)?

The eight bond equations derived by propagating computable causality from the inverse system input, f_1, are listed to illustrate the propagation path:

1 $f_1 = v(t)$

2 $f_2 = f_1$

3 $e_2 = rf_2$

4 $f_3 = f_1$

5 $e_3 = kq$

6 $f_4 = f_1$

7 $e_4 = m\, df_4/dt$

8 $e_1 = e_2 + e_3 + e_4$

In the linear case, this gives:

$$F(t) = kq + rv + m\, dv/dt \qquad (5.2)$$

where $q = \int v \, dt$

It can be seen that since the inertia is no longer an independent state, equation (5.2) is not exactly in ODE form, nor is it a state equation. It will be shown that this is best represented by a generalised state equation[48].

This example also illustrates the application of bond graph analysis to deriving the constitutive law at the system input port, as is often required in electrical circuit analysis. Considering the equivalent electrical system - a series R,C,L network - this inverse bond graph provides the solution to the problem: what is the voltage generated for a given source current. Thus inverse system analysis of bond graphs with collocated source and sensor can be seen to be more generally applied to the derivation of impedances or admittances at the input port (bond).

The differential-algebraic equations for this example are extracted from the ordered bond equations. The state- and non-state variables are obtained directly from the bond graph (figure 5.6b), and we may re-identify the variables as follows:

state $x = q$

non-state $z = p = mf_4$

input $u = f_1$

output $y = e_1$

The state equation is

$$dx/dt = u$$

the non-state equation is

$$z = mu \quad \text{or} \quad 0 = -z + mu$$

the output equation is

$$y = kx + dz/dt + ru$$

and since the source and sensor are collocated there is no constraint equation $w = F_w(x, z', v, u)$.

These linear DAE's can be rewritten in generalised state equation form where

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 \\ 0 \\ m \end{bmatrix}$$

$$C = (k\ 0\ 1)$$

$$D = (r)$$

and, $E = I_0(2,1)$

Thus the transfer equation may be derived using (3.26) as

$$\frac{y}{u} = G(s) = (k\ 0\ 1) \begin{bmatrix} s & 0 & 0 \\ 0 & s & -1 \\ 0 & 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 0 \\ m \end{bmatrix} + r$$

which gives

$$G(s) = ks^{-1} + ms + r$$

Two points are evident from this example. Firstly, the sole use of the source-sensor element `SS1' in figure 5.6 was to permit the causality of the input source to be reversed. Secondly, the new causal (dot) notation is unnecessary to describe the standard causal propagation for the inverse system, although it is entirely compatible. The following sections show examples where the new causal notation must be used to identify the causal propagation path.

5.2.3. Unilateral bonds

For inverse system analysis we wish to take a more abstract view of the system compared with the normal bond graph view, which is closely related to the realisability of the physical system. Dropping this insistence on realisability, and considering system causality from the point of view: `*computationally*, what input is required to achieve a given output?', causes us to take a novel approach to causality on any given bond.

Figure 5.7 Multi-Input Multi-Output system

Considering the MIMO system of figure 5.7, with input vector U and output vector Y, the output is expressed in transfer function form as:

$$Y = H(s)U \tag{5.3}$$

where $Y = [y_1 \; .. \; y_n]^T$ and y_i etc. may be any combination of effort or flow variables,

$U = [u_1 \; .. \; u_n]^T$ and u_i etc. may be any combination of effort or flow variables.

H(s) is assumed to be a square (n x n) matrix.

The signals in figure 5.7 state explicitly that for all outputs in Y, if y_i is an effort variable then the corresponding flow variable (w_i) is zero, and vice versa. Consequently, if one wishes to obtain the inverse system, both the effort and flow variables for the output must be included on the RHS of the inverse equation. A further consequence is that both the effort and flow variables of the system inputs are of interest, resulting in an expression in transfer matrix (chain matrix) form:

$$\begin{bmatrix} U \\ V \end{bmatrix} = \begin{bmatrix} P & Q \\ R & S \end{bmatrix} \begin{bmatrix} Y \\ W \end{bmatrix} \qquad (5.4)$$

where $V = [v_1 \; .. \; v_n]^T$, and v_i are the flow/effort variables corresponding to the effort/flow inputs u_i. For the signal output vector W, all $w_i = 0$.

The significance of equation (5.4) to applying computable causality, is that the RHS now consists of *both* the effort and flow variables for each output. To indicate this on the bond graph, it must be possible to show that both the effort and flow are imposed by the same node, as in figure 5.8.

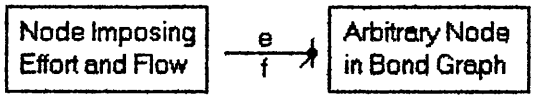


Figure 5.8 Bond with effort and flow imposed from same node

To emphasise that the causal effects on such a bond act in one direction only, we describe this as a *unilateral bond*. The effort and flow variables on the unilateral bonds are independent.

When analysing systems with pre-determined constitutive relations, a bond cannot impose both effort and flow on any type of node* . This unilateral causality is confined to bonds which make up the graph structure (interjunction bonds and those attached to transformers and gyrators) and input/output bonds. It is normal for bond graphs of inverse systems to have both unilateral and bilateral bonds, so we term such augmented graphs *bicausal bond graphs*.

---

* This is not true if the model is being used for fault detection. In this case, it is equally valid to propagate both known effort and flow onto a storage or dissipator node using a unilateral bond. This shows the potential for extending this technique to other applications, not covered in this thesis.

With this addition to the standard convention, it is now necessary to analyse the effect on the standard causality propagation rules. The following section shows that the existing propagation algorithm for computational causality can be used unchanged.

5.3. A procedure for causal augmentation

A bond graph model may be systematically causally augmented just by following the simple rules for initiating causality and propagating each initialisation through the graph. The rules for causal augmentation are derived from the Sequential Causality Assignment Procedure (SCAP)[17], and are given in this section, with the added implications for inverse systems.

5.3.1. Rules for initiating causal propagation

When the form of the required mathematical model is an inverse model, then both effort and flow variables on the system outputs are defined as known i.e. the signal to the `SS' element becomes a unilateral bond with fixed causality (figure 5.9). All input causalities are redefined as reversible, using `SS' elements to replace `SE' and `SF' sources, but energy stores retain their preferred integral causality. Causalities due to zero or infinite parameter values remain fixed.

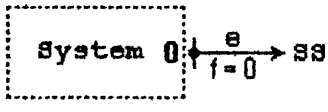


Figure 5.9 Effort sensor redefined as inverse system input

The order for initiating causal propagation is described by the following four steps:

a) Scan the entire bond list and initiate causal augmentation from fixed causality input/outputs. For each fixed causality found, propagate the known effort/flow through the graph using defined causality rules for the structural elements (`0', `1', `TF', `GY'). Causal conflicts are likely to arise, as derivative causality is imposed on energy stores or resulting from conflicting constraints due to incompatible parameter values. Highlight causal conflicts, as the modeller may have to reconsider the constraints.

b) Scan the entire bond list and initiate causal augmentation from the remaining fixed causalities as in (a).

c) Scan the entire bond list and initiate causal propagation from unaugmented nodes with preferred causalities. For each preferred causality found, propagate the known effort/flow through the graph using defined causality rules for the structural elements. Highlight causal conflicts (in this case, non-states), as before.

d) If the graph is causally incomplete at this stage the model is *under-causal*, and causality may be completed either by arbitrary assignment of the effort/flow on one or more bonds, or by employing one of the better defined techniques[14,43] for under-causal systems. Under-causal systems result in algebraic loops (implicit equations) which must be solved before the full mathematical model can be derived. The effort or flow assigned by the modeller in this procedure, becomes the intermediate variable in the algebraic loop.

5.3.2. Rules for causal propagation

Graphical causal propagation is entirely defined by the energy conserving constitutive relations of the structural nodes. These are the `0', `1', `TF', `GY' nodes.

The 0- and 1-junction nodes each exhibit so-called `strong' or `weak' causality, depending on the causality imposed by the known incident bond. When augmenting causality of inverse systems the same causality rules apply, but the incident bond can now impose both effort and flow on the junction as in figure 5.10a. At some stage in the augmentation process, the `weak' causality law (figure

5.10b) will also have to be applied, although the strong causality case always applies when the incident bond is unilateral.

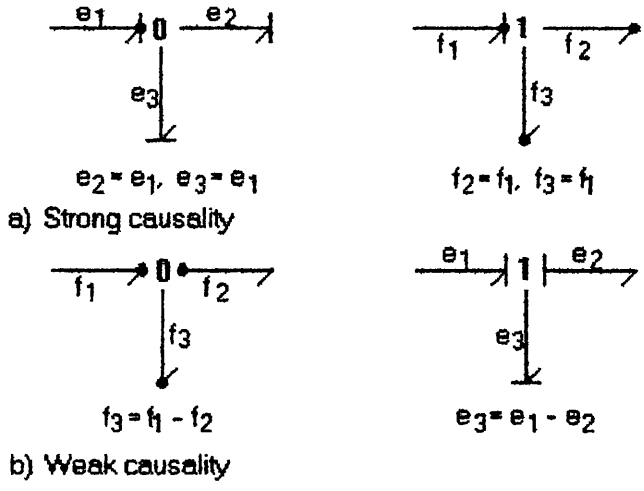


Figure 5.10 Junction causality rules

Whereas 0- and 1-junctions may have an arbitrary number of ports (bonds), transformer and gyrator (`TF' and `GY') nodes typically have only two ports. Transformers propagate the causality imposed by the incident bond to the second transformer port. Thus, if both effort and flow are imposed on one port these are both propagated from the second port (figure 5.11a). In this figure, the numbers above the bonds are used to identify the respective effort/flow variables.

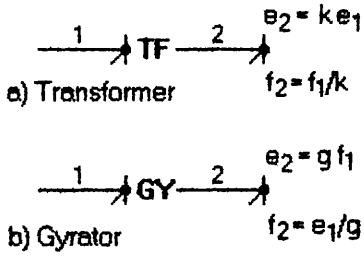


Figure 5.11 Transformer and gyrator causal propagation for unilateral bonds

Gyrators propagate the reverse causality imposed on the first port through to the second port, and vice versa. Imposing both effort and flow on one port cause flow and effort, respectively, to be

propagated from the second port. Using the `dot' notation described in this chapter graphically distinguishes the causal augmentation of inverse systems from conventional analysis, as shown in figure 5.11. The graphical causality pattern for the gyrator in a bicausal bond graph is noticeably different from that for a conventional bond graph, since, in this case, the effort and flow are both imposed on the same gyrator port. However, it should be noted that following computable causality propagation rules, the augmentation procedure is unchanged i.e. the gyrator constitutive relation is unchanged.

5.3.2.1. Modulated transformers and gyrators

A major advantage of using a bond graph as the core representation from which any given mathematical model may be derived, is that bond graph causal augmentation uniquely identifies algebraic loops (section 5.3.1d)). Modulated components, such as transformers and gyrators, will cause algebraic loops if the modulated parameter is a function of a variable which is dependent on the `output' of that component. In such cases, as shown in the following example, standard graphical causality does not identify the algebraic loop, thus preventing the automatic derivation of the mathematical model.

The computable causality algorithm does, however, inhibit causal propagation through a modulated component where the modulating variable is unknown. The result is an identified algebraic loop, which can be solved using a standard algorithm.

In the bond graph shown in figure 5.12, the hypothetical system consists of an effort source imposed on a dissipator through a modulated transformer[50], where the transformer modulus is a function of the flow variable at the dissipator.

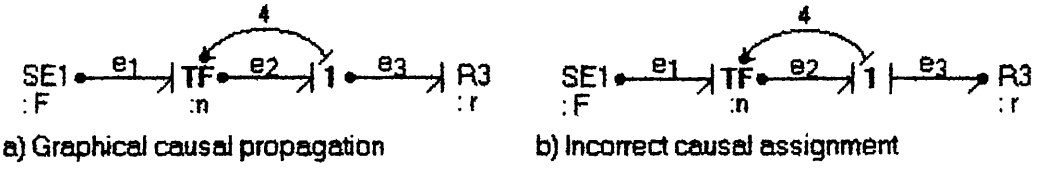


Figure 5.12 Causal propagation through a modulated transformer

Figure 5.12a shows that graphical causality completes without any problem, but the bond equations are no longer ordered, and include an algebraic loop with the independent variable n, as shown in column 2 equations in Table 5.1.

#	2: Graphical causality	3: Computable causality
1	$e_1 = F$	$e_1 = F$
2	$e_2 = ne_1$	$e_4 = 0$
3	$e_4 = 0$	$f_3 = e_3/r$
4	$e_3 = e_2 - e_4$	$f4 = f_3$
5	$f_3 = e_3/r$	$n = kf_4$
6	$f4 = f_3$	$e_2 = ne_1$
7	$n = kf_4$	$e_3 = e_2 - e_4$
8	$f_2 = f_3$	$f_2 = f_3$
9	$f_1 = nf_3$	$f_1 = nf_3$

Table 5.1 Causally ordered equations with modulated transformer

This algebraic loop occurs `accidentally' in equation 2, as it is not known whether n has been derived at this time. Column 3 of Table 5.1 shows the result of computable causal propagation, where the propagation terminates due to the unknown modulation, and e_3 is specifically chosen as the intermediate variable in equation 3. Thus, for the computable causal propagation the algebraic loop is explicitly identified, and may be solved for the intermediate variable before the rest of the equations are solved.

Equation 6 in the computable causality propagation, highlights the difference between the conventional propagation rule for activated bonds and the equivalent computable causal propagation rule. Conventionally, activated bonds are ignored at the source node, giving $e_3 = e_2$, but for the computable causality algorithm the effort/flow variables at each junction are handled consistently for both energy bonds and activated bonds. The constitutive law for activated bonds and modulating signals defines one of its effort/flow variables as zero (in this case $e_4 = 0$).

Figure 5.12b indicates that the computable causality algorithm could fail completely if f_3 were arbitrarily chosen as the intermediate variable rather than e_3. In this case, the algorithm

fails since it tries to force a causal conflict on the transformer. The solution provided in the algorithm implemented by the author was a two-pass causal analysis - where graphical causal propagation indicates preferred causal directions, and thereafter computable causality sorts the equations and identifies algebraic loops.

5.3.3. Non-collocated sources and sensors

In many systems, for example that discussed in section 5.3.3.1, the sensors and sources are not collocated, but a necessary condition for a system to be invertible is that it has the same number N of inputs and outputs. This is automatically satisfied for systems with only collocated sensors and sources, but must be included as an assumption in the case of systems with some non-collocated sensors and sources.

The method for deriving the inverse system bond graph for systems containing non-collocated sensors and sources has two stages:

1) Convert the system bond graph into that of an equivalent system with collocated sensors and sources.

2) Apply Algorithm 5.2.2.1

Part 1 of this algorithm is

a) Replace all system *inputs* without corresponding collocated sensors by source-sensor elements. This introduces additional system outputs internal to the system model. The ith such internal system output is denoted v_i. This does not necessarily imply changing all SE and SF elements to SS elements. Example 3.3.1 indicates one case where gravity is represented by an effort source; it would be meaningless to treat this as a possible time-varying output of the inverse system.

b) Replace all sensors without corresponding collocated sources by source-sensor elements. This introduces N_n additional system inputs internal to the system model. The ith such internal system input is denoted w_i

c) Impose the additional N_n constraint equations:

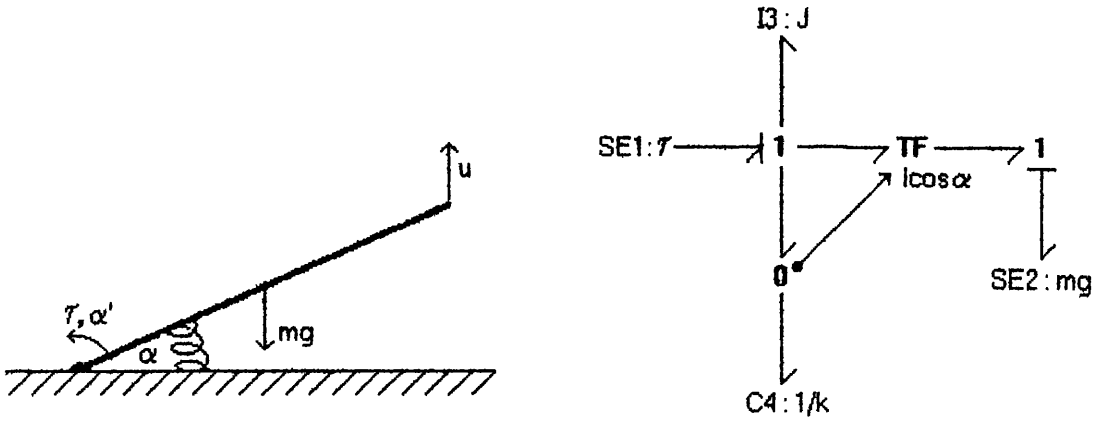
$$w_i = 0, \text{ where } i = 1 \ldots N_n \qquad (5.5)$$

The additional inputs w_i, together with the additional outputs v_i play a crucial role in determining the system inverse. The N_n constraint equations (5.5) applied to the inverse system outputs w_i ensure that the N_n additional inputs to the system have no effect on the system, and implicitly define the values of the inverse system inputs v_i.

Indicating system outputs by *signals* to `SS' elements, makes these constraint equations explicit on the bond graph, while unilateral bonds permit these constraints to be automatically propagated into the graph, using computable causality.

5.3.3.1. Example: Manipulator arm

The manipulator arm (figure 5.13) is attached by a linear spring (equivalent to a proportional controller) to the reference angle. This state is used in the bond graph to derive the angle, α, of rotation of the arm, as a modulation ($l\cos\alpha$) on transformer TF, where l is the distance of the centre of mass from the pivot. The aim of the inverse system analysis is to calculate the required torque $\tau(t)$ required to obtain a given velocity trajectory u(t) at the end of the arm. Only the vertical velocity component is considered, in order to clarify the example.



a) Hinged manipulator arm with spring b) Acausal bond graph

Figure 5.13 Manipulator arm

In this example the sensor and source are no longer collocated. The first step of the algorithm, conversion to a collocated system, is accomplished in figure 5.14, where an `SS' element (labelled `u') has been added to the right hand end of the diagram, indicating the output which is to be defined. The transformation n scales the velocity u at the end of the rod to that at the centre of mass.

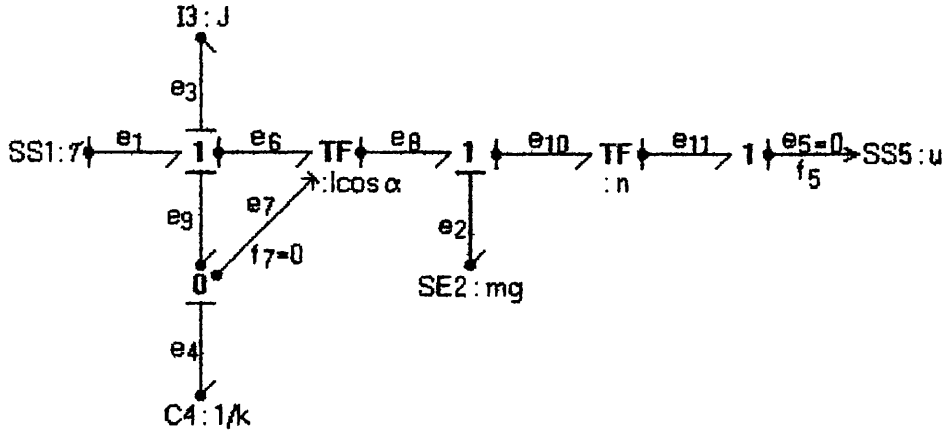


Figure 5.14 Bond graph of manipulator arm inverse system

In addition, the following constraint is imposed:

$$w_1 = e_5 = 0$$

where e_5 is the force associated with the `SS' element labelled `u'.

The effort source of figure 5.13 has also been replaced by a source-sensor element (SS1) in figure 5.14 to permit this to have any causality. The equations for the inverse system are listed below, ordered according to the computable causal propagation path:

1 $e_2 = m\,g$ (mg is the independent effort variable)
2 $e_5 = 0$ (flow sensor imposes zero effort on system)
3 $e_{11} = e_5$
4 $e_{10} = n\,e_{11}$
5 $e_8 = e_{10} + e_2$
6 $f_5 = u$ (flow defined by input to inverse system)
7 $f_{11} = f_5$
8 $f_{10} = f_{11}/n$
9 $f_2 = f_{10}$

10 $f_8 = f_{10}$

11 $f_7 = 0$ (modulation imposes zero flow on junction)

12 $e_4 = k\alpha$ (initiate integral causality from energy store)

13 $e_7 = e_4$

14 $e_6 = l\cos\alpha\ e_8$ (transformer modulation now known)

15 $f_6 = f_8/l\cos\alpha$

16 $f_1 = f_6$

17 $f_3 = f_6$

18 $e_3 = J\ df_3/dt$ (derivative causality imposed on I element)

19 $f_9 = f_6$

20 $f_4 = f_9 - f_7$

21 $e_9 = e_4$

22 $e_1 = e_3 + e_6 + e_9$

where $\alpha = \int f_4\ dt$ (5.6)

Since these equations do not result in a linear transfer function, a more general formalisation must be used - differential-algebraic equations may be used to describe such systems.

Again, we can extract the differential-algebraic equations for this example from the ordered bond equations, using figure 5.14, if we re-identify the variables from the bicausal bond graph:

state $x = \alpha$

non-state $z = p_3 = Jf_3$

inputs $u = f_5$

$w = e_5 = 0$

outputs $y = e_1$

$v = f_1$

The state equation is

$$dx/dt\ (= f_4) = u/nl\cos(x)$$

the non-state equation is

$$z = Ju/nl\cos(x)$$

the output equations are

$$y = kx + dz/dt + wnlcos(x) + mglcos(x)$$

and $v = u/nlcos(x)$

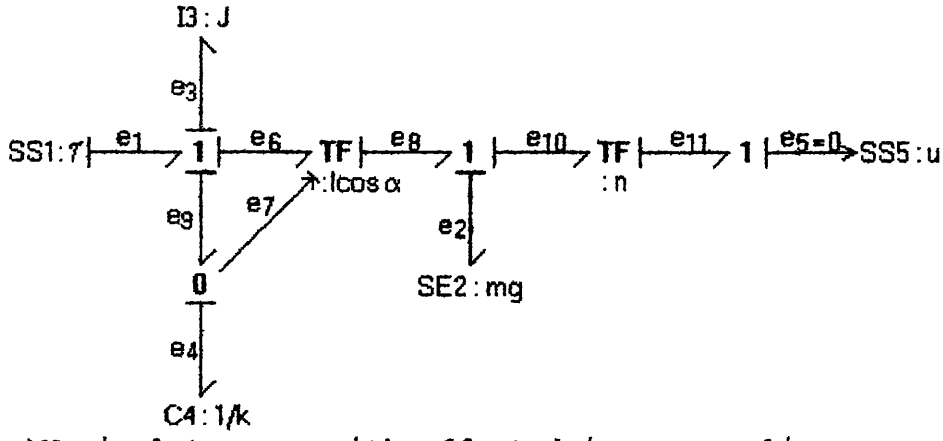
These equations are all non-linear functions of x, and thus cannot be expressed in the form of equations (4.27), hence the DAE form is used for further analysis.

5.3.4. Application of bicausal bond graphs

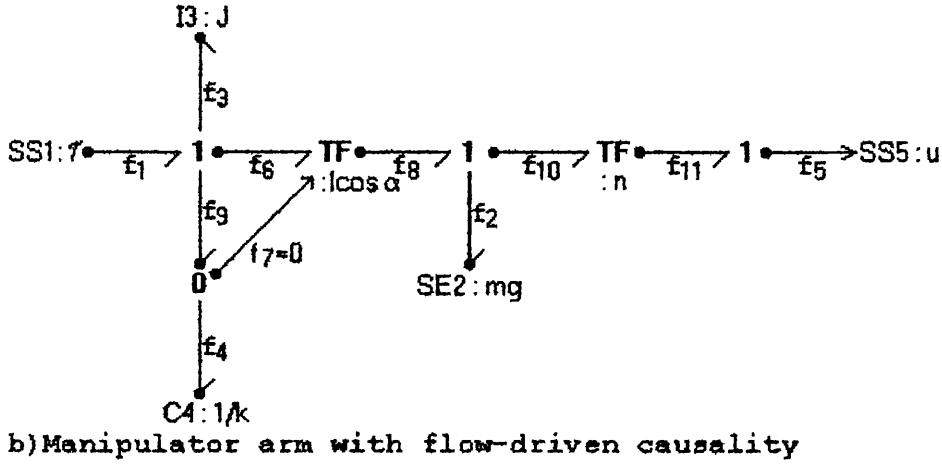
The manipulator example analysed in section 5.3.3.1 demonstrated that bicausal bond graphs can be used to derive the inverse system transfer function of a 2-input, 2-output system. However, this method for obtaining the inverse system transfer function, directly from the bicausal bond graph model, is generally applicable to any N-input, N-output system.

Furthermore, in this specific example, one of the `inputs' of the inverse system was constrained to be zero, since this represented an ideal sensor SS5. This constraint need not, in general, apply, with the result that the bicausal bond graph permits the derivation of the inverse transmission matrix of the system.

The bicausal bond graph may be considered to be the superposition of two causally augmented bond graphs - one having conventional effort-driven causality, and one having flow driven causality. This is illustrated for the manipulator arm example in figure 5.15. Figure 5.15a) is causally augmented in the conventional manner, using the causal stroke. The graph is causally complete, but I3 has derivative causality imposed on it, due to the reversal of causality of SS1, following rule 5.2.2.1b). Figure 5.15b) is causally augmented using the dot notation to illustrate flow-driven causality, resulting in f1 being imposed on SS1. This decomposition indicates clearly that the conventional causality rules apply at each of the junction elements.



a)Manipulator arm with effort-driven causality



b)Manipulator arm with flow-driven causality

Figure 5.15 Decomposition of a manipulator arm bicausal bond graph

The advantage of the bicausal bond graph, in this case is restricted to conciseness, and the ability to order the inverse system equations by following the bicausal propagation path.

The following example illustrates another case where the bicausal bond graph specifically identifies all the states and non-states of the inverse system, whereas conventional causal augmentation fails to do this.

5.3.4.1. Example: An electrical RC circuit

Figure 5.16 shows a two stage RC circuit with a voltage source; the output being the voltage measured across the second capacitor c2.

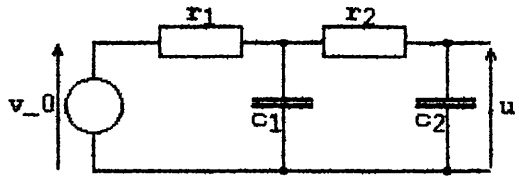
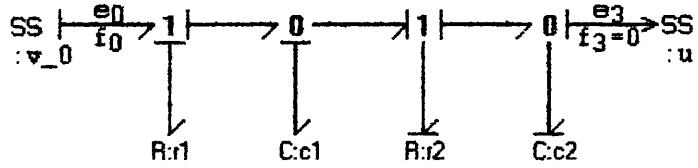
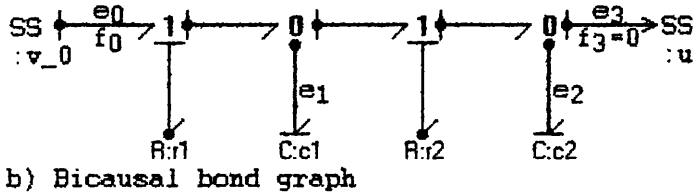


Figure 5.16 RC circuit

Gawthrop[52] has illustrated the application of conventional bond graph causality to deriving the inverse transfer function of this system. Figure 5.17a) shows the resulting causal augmentation, using the rules given in section 5.3.3. The result is that the capacitor c_1 is given (preferred) integral causality, implying that the inverse system has one state variable.



a) Conventional bond graph



b) Bicausal bond graph

Figure 5.17 Bond graphs for inverse model of RC circuit

The bicausal bond graph (figure 5.17b) for the same inverse system, indicates that propagating causality with unilateral bonds using the computable causality algorithm, forces derivative causality on both capacitors, which shows that the inverse system actually has *no* state variables.

The state- and non-state variables are obtained from the bicausal bond graph, and we may re-identify the variables as follows:

state x = None, thus there is no state equation.

non-state $z_1 = q_1 = c_1 e_1$

$z_2 = q_2 = c_2 e_2$

input $e_3 = u$

The (inverse system) output equations are:

$$y = e_0 = z_1'r_1 + z_2'(r_1 + r_2) + w(r_1 + r_2) + u$$

$$v = f_0 = z_1' + z_2' + w$$

The equation for the second output, v, is combined with the input constraint

$$w = f_3 = 0$$

to give the constraint equation

$$w = F_w(x, z', v, u).$$

i.e. $w = 0 = z_1' + z_2' - v$

The non-state equations are derived by calculating the effort variables e_1 and e_2 corresponding, to the energy stores c_1 and c_2 having derivative causality:

$$0 = -z_1 + c_1r_2z_2' + c_1u = -z_1 + \tau z_2' + c_1u$$

where $\tau = c_1r_2$

and

$$0 = -z_2 + c_2u$$

Using the descriptor vector

$$X = \begin{bmatrix} x \\ z_1 \\ z_2 \\ z_1' \\ z_2' \\ v \end{bmatrix}$$

these linear DAE's can be rewritten in generalised state equation form where

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & \tau & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & -1 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ c_1 \\ c_2 \\ 0 \end{bmatrix}$$

$$C = (0\ 0\ 0\ r_1\ (r_1+r_2)\ 0)$$

$$D = (1)$$

and,

$$E = I_0(3,3)$$

Since there are no states, column and row 1 of A are all zero, so this may be reduced to a 5x5 matrix, with corresponding changes to B, C, D and E.

Thus the transfer equation may be derived using (5.31), as

$$G(s) = (0\ 0\ r_1\ (r_1+r_2)\ 0) \begin{bmatrix} s & 0 & -1 & 0 & 0 \\ 0 & s & 0 & -1 & 0 \\ 1 & 0 & 0 & -\tau & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ c_1 \\ c_2 \\ 0 \end{bmatrix} + 1$$

i.e. $G(s) = c_1c_2r_1r_2s^2 + (c_1r_1 + c_2(r_1+r_2))s + 1$

5.4. Bicausal bond graphs for under-causal models

In chapter 4, a new algorithm was described, which completed causal augmentation of under-causal bond graphs by the addition of auxiliary sources. It was noted that there is a constraint equation associated with each auxiliary source, which results in an algebraic equation. The constraint equation takes one of two forms;

either that the input variable to the source is zero, or that the output from the source must be equal to the `natural' effort/flow on the chosen 0/1 junction.

In this section, it will be shown, using the bicausal bond graph notation, that these two forms of constraint equation are directly equivalent. It will also be shown that bicausal bond graphs can be used with the new algorithm to give the minimal number of algebraic equations, as for the Lorenz and Wolper method, but with the added advantage of the explicit graphical algorithm.

5.4.1. Example: Electrical circuit resulting in algebraic loop

Considering the example given in chapter 4, section 4.2.1, causality is completed, as before, by adding an auxiliary source with output i_0. In order that the system is not disturbed by this source, the input to the source must be zero, i.e. $e_0 = 0$.

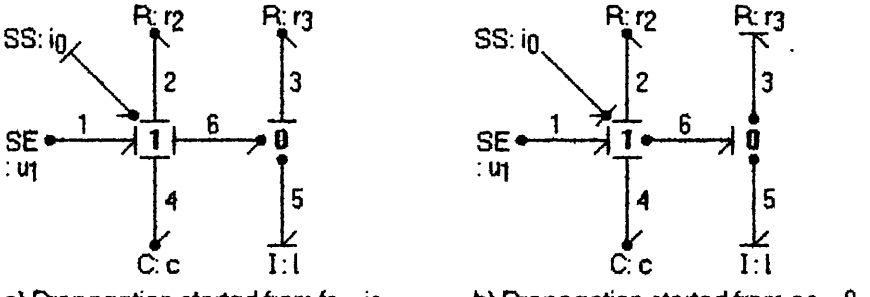


Figure 5.18 Bond graph of electrical network

Figure 5.18a) indicates the bond graph for the system with the auxiliary input i_0 using the new notation. The resulting causal pattern is identical to that obtained using the standard causal notation (chapter 4, section 4.2.2.1). The ordered equations are listed in Table 5.2, column 2, with the additional constraint equation (15), giving e_0 as the repeated LHS variable, from which the algebraic equation is derived.

#	Column 2	Column 3
1	e1 = u	e1 = u
2	e4 = q/c	e4 = q/c
3	f5 = p/1	f5 = p/1
4	f0 = i0	e0 = 0 (constraint equation)
5	f1 = f0	f0 = i0
6	f2 = f0	f1 = f0
7	e2 = f2r2	f2 = f0
8	f4 = f0	e2 = f2r2
9	f6 = f0	e6 = e0+e1-e2-e4
10	f3 = f6-f5	e3 = e6
11	e3 = f3r3	f3 = e3/r3
12	e5 = e3	f6 = f3+f5
13	e6 = e3	e5 = e6
14	e0 = e2+e4+e6-e1	f4 = f0
15	e0 = 0 (constraint equation)	f6 = f0 (causal conflict)

Table 5.2 Ordered equations for electrical circuit

If, however, we choose to define this constraint ($e_0 = 0$) as an input to the system and propagate from this input *before* propagating from the flow i_0, the order of the bond equations is changed (Table 5.2, column 3), and the resulting bond graph is shown in Figure 5.18b). Two points are clear on this *bicausal* bond graph:

- both e0 and i0 are inputs to the model, as indicated by the unilateral bond, and
- there is an apparent causal conflict as both bonds 0 and 6 define the flow on the 1-junction.

This second point is expressed in Table 5.2, column 3, as equations (12) and (15) both have f_6 as the LHS variable. Solving for f6 gives the same algebraic equation, as solving Table 5.2, column 2 for e_0.

The use of the new notation in this example has indicated that the constraint that the auxiliary input to the system is equal to the 'natural' effort/flow on the chosen 0/1 junction, is equivalent to the constraint that the corresponding system output is zero. Either

the choice of propagation order, or the system topology (chapter 4, section 4.2.2.3), can, therefore, result in an apparent causal conflict, expressing the constraint.

5.4.2. Example: An electrical resistor network

This electrical resistor network has been analysed in chapter 4 (figure 4.4), to illustrate the ability of the Lorenz and Wolper algorithm to minimise the number of algebraic loops. In that chapter it was noted that the new algorithm for completing causality of under-causal bond graphs did not, in general, result in the minimum number of algebraic equations. This example will show that the new algorithm can utilise the concept of the unilateral bond to achieve this minimisation.

The resulting bicausal bond graphs are shown in figure 5.19, where causal completion is achieved using a single auxiliary source, with both the effort and flow constraints applied to the model. In this case, the input causality due to the auxiliary source is propagated through the bond graph as far as possible, and, if causality is still not complete, the second constraint causality is also propagated through the graph.

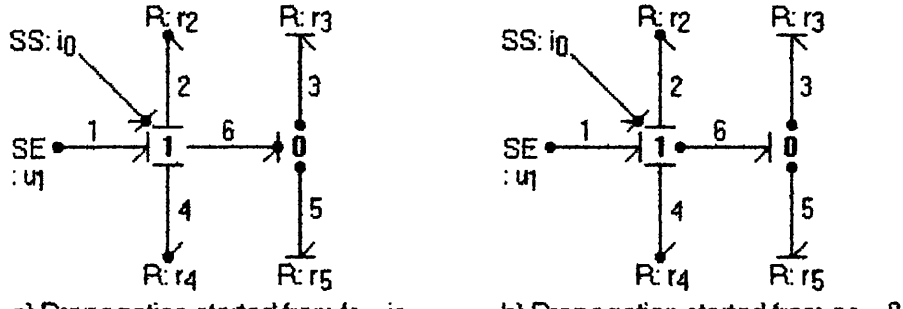


Figure 5.19 Causal completion with additional flow source

Figure 5.19a) shows the completed causal augmentation for the system, where the new causal completion algorithm started by propagating the flow input i_0. The corresponding set of ordered equations is listed in Table 5.3 column 2, which indicates that this causality could propagate no further than equation (8).

#	Column 2	Column 3
1	e1 = u	e1 = u
2	f0 = i0 (intermediate var)	e0 = 0 (constraint equ'n)
3	f1 = f0	f0 = i0 (intermediate var)
4	f2 = f0	f1 = f0
5	e2 = f2r2	f2 = f0
6	f4 = f0	e2 = f2r2
7	e4 = f4r4	f4 = f0
8	f6 = f0	e4 = f4r4
9	e0 = 0 (constraint equ'n)	e6 = e0+e1-e2-e4
10	e6 = e0+e1-e2-e4	e3 = e6
11	e3 = e6	f3 = e3/r3
12	f3 = e3/r3	e5 = e6
13	e5 = e6	f5 = e5/r5
14	f5 = e5/r5	f6 = f3+f5
15	f6 = f3+f5 (causal conflict)	f6 = f0 (causal conflict)

Table 5.3 Alternative equation ordering for resistor network

At this point the constraint $e_0 = 0$ was then propagated into the model, using the unilateral bond notation for the bond attached to the auxiliary input. This results in an apparent causal conflict where all flow inputs to the 0-junction are defined, since there are two equations for f6.

The alternative causal completion shown in figure 5.19b) results from propagating the constraint equation $e_0 = 0$ before propagating the input flow i_0. The resulting ordered equations are listed in table 5.3, column 3, where the two equations for f6 again indicate the apparent causal conflict. It can be seen that the differences between the two solutions are trivial as reflected in the limited extent of the graphical differences in the causality.

5.4.3. A new causal completion algorithm resulting in the minimal number of algebraic equations

This algorithm fulfils part 2 of the causal augmentation requirement, where the bond graph has proven to be under-causal:

1. Assuming that the bond graph is proper (all bonds impinge on a junction) then at least one junction in an under-causal graph does not have causality imposed on it. That is, a causally incomplete 0-junction does not have an effort imposed on it, or a causally incomplete 1-junction does not have a flow imposed on it. An appropriate auxiliary source (an effort source for a 0-junction; a flow source for a 1-junction) can then be attached to the junction, and the causalities propagated throughout the graph.
2. If the model is still not causally complete, apply the causality due to the corresponding constraint from the auxiliary source ($f = 0$ from an effort source; $e = 0$ from a flow source) using the unilateral bond notation, and propagate this through the graph.
3. Repeat steps 1 and 2 until the bond graph is causally complete.

5.5. Conclusions

This chapter has highlighted the unique characteristic of acausal bond graphs as a core model representation from which a variety of mathematical models may be derived, by applying model-specific causal initiation rules. In particular, we have shown that the computable causality propagation algorithm can automatically generate mathematical models of inverse systems. An extended `dot' notation permits computable causality to be described graphically, but is a compatible superset of the standard bond graph causal stroke notation.

The major extension to existing causal analysis is the concept of the unilateral bond - a bond which can impose both effort and flow variables on a node in the bond graph, whereas for conventional causality a bond imposes only one of the variables on each node. The dot notation permits the path of the computable causality algorithm to be visualised, thus extending the strong graphical analysis properties of bond graphs to inverse system models. For inverse systems with non-collocated sources and sensors, the dot notation graphically identifies the state- and non-state-variables, which may not be indicated by conventional causal propagation. Rules for propagating this causality through the junction structure

have been listed and shown to obey the constitutive relations of these bond graph elements.

The bicausal bond graph has also been shown to be useful in extending the capabilities of the graphical algorithm for completing causality of under-causal bond graphs. In this case, the unilateral bond is used to simultaneously impose both the auxiliary input variable and the constraint variable onto the model, resulting in the minimal number of algebraic equations. Since bicausal bond graphs permit constraint propagation to be handled by the established causal propagation rules, this indicates a promising area for further work; specifically in the field of fault detection.

It has been demonstrated that an inverse system model can be represented by a set of differential and algebraic equations which are systematically derived from the bicausal bond graph. For a linear system these equations may be expressed in the generalised state equation form, while a further transformation provides the transfer function, offering a transformation-based method of obtaining this, rather than Mason's Rule.

The generalised state equation has also been shown to be an appropriate formalisation for describing either inverse systems, or systems which include non-state variables and/or algebraic loops.

CHAPTER 6 CASE STUDIES USING BOND GRAPH MODELS

6.1. Introduction

The previous chapters have discussed the application of bond graphs to modelling of physical systems, and introduced new techniques for deriving specific models. In this chapter, these new techniques are used to model real physical systems, to better illustrate their application in realistic scenarios.

Each of the systems modelled may be represented hierarchically, although in most cases the hierarchical nesting is only appropriate to one level. The hierarchical modelling rules described in chapter 3 are used to build the models, and thence aggregate the complete model to a `flat' bond graph.

The four systems chosen as case studies are all modelled using bond graphs as the core model representation. The depth of analysis of the resulting models is then dependent on the derived model chosen to illustrate the techniques described in the previous chapters.

Section 6.2 shows the development of a hierarchical model for the extruder system described in section 1.2, and a steady state model is derived from this bond graph. In section 6.3, an example simulated by Elmquist[53] is redeveloped as a bond graph model from the original block diagram implementation. A detailed transfer function model of a passive electronic telephone network is derived from the equivalent bond graph representation, in section 6.4. Finally, section 6.5 describes the development and analysis of a bond graph model of a flying blade used for production carpet cutting. The last two examples were used to generate symbolic transfer functions, which were parameterised for further analysis using Matlab. Section 6.6 concludes the chapter.

6.2. A plasticating extruder

A descriptive model of this industrial application was developed in section 1.2 (chapter 1), to illustrate the need for a core model

representation. In this case study, a hierarchical word bond graph is developed for this system following the development process described in section 3.5.2. Due to the complexity of the extrusion process, significant effort could be expended developing a detailed model[54,55,56], but in this example, the bond graph developed is a simple exploratory model suitable for understanding the basic processes. In particular, the model represents the final *metering* section of the extruder where all the polymer is molten, although similar models may be cascaded to represent the feed and transition sections of the extruder. The resulting `flat' bond graph is then analysed, using the algorithm detailed in chapter 4 to model the steady state performance, as would be required to predict the thickness of polymer extruded onto electrical wire.

6.2.1. Developing the hierarchical word bond graph

At the highest level of abstraction, a plasticating extruder consists of the following sub-systems:

- d.c. motor
- heated extrusion barrel
- extrusion screw
- extruded polymer
- die

These sub-systems are inter-connected to give the word bond graph shown in figure 6.1, where the `SS' elements are drawn to indicate the major inputs and outputs of the system without imposing causal constraints.

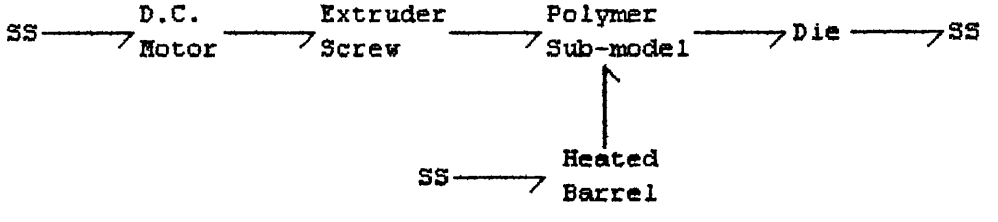


Figure 6.1 Word bond graph of plasticating extruder

Following the method of section 3.5.2, shows that the only word bond node which can usefully be further decomposed into another word bond graph is the polymer sub-model. The polymer node may be modelled as two separate but interactive processes in the expanded word bond graph (figure 6.2) for the full model.

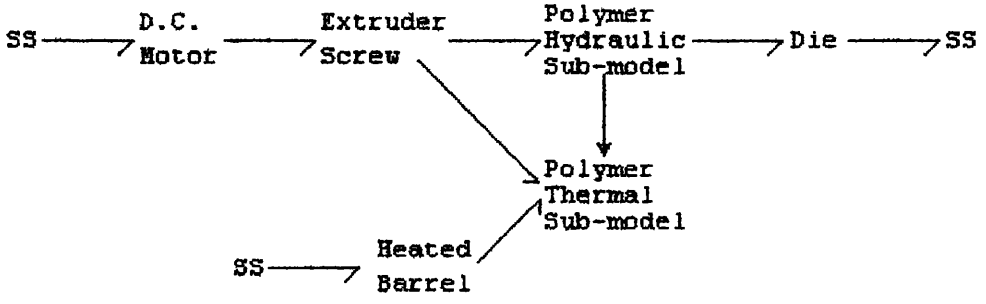


Figure 6.2 Expanded word bond graph of plasticating extruder

6.2.2. Combined energy and pseudo bond graph model

As for any engineering design, there are decisions to be made on trade-offs between alternative approaches, and in this case, the main decision is whether to use an energy bond graph or a pseudo bond graph.

For the d.c. motor, the energy bond graph model described in section 4.3, is appropriate, since it describes the transduction between electrical and rotational energy domains most effectively. Since we are particularly interested in polymer *mass flow rates* through the extruder, and can linearly control the *power* input to the barrel heater, the pseudo bond graph appears most appropriate when modelling the polymer and heater sub-systems. Thus possible variables for the hydraulic sub-model are pressure (effort) and mass flow rate (flow), while those for the thermal sub-model are temperature and enthalpy flow rate respectively. The mass of polymer in the modelled section of the extruder is then the hydraulic state variable. The enthalpy state variable of the polymer results from enthalpy flows from the heated barrel sub-system, and from the viscous shearing action of the screw in the polymer, together with the nett enthalpy flows as polymer passes through the extruder. It can be seen that the hydraulic-enthalpic (heated tank) sub-model described in section 3.5 is suitable for

modelling the polymer sub-system, where the hydraulic capacitance is replaced by a capacitance representing the compressibility of the polymer. However, this would result in a stiff system model, and since most extruder models assume the polymer is incompressible, the chosen model drops this capacitance and assumes constant mass of polymer in the barrel control volume. Using this assumption the hydraulic model can revert to an energy bond graph where the bond variables are *pressure* and *volume flow rate*.

Since the extrusion screw sub-model interfaces between the d.c. motor energy bond graph and the polymer pseudo bond graph, this sub-model must contain the transformations between the variables on the input bond whose product is power, and those chosen for convenience in the pseudo bond graph. Figure 6.3 shows the acausal bond graph of the extruder screw.

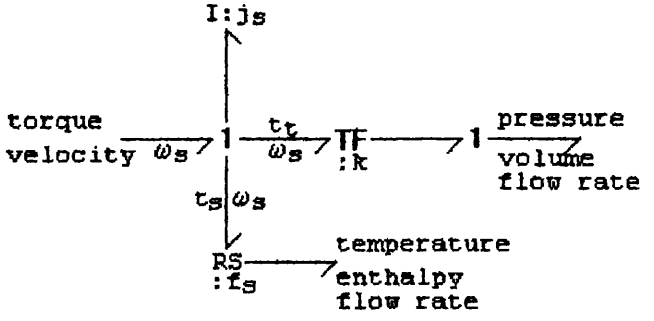


Figure 6.3 Acausal sub-model of extruder screw

It can be seen that the bond variables on the bonds connected to the left-hand 1-junction are torque (effort) and angular velocity (flow). The energy conserving conversion to pressure and volume flow rate is implemented using the transformer TF:k, giving

$$\text{pressure} = \frac{t_t}{k}$$

and volume flow rate = $\omega_s k$

where the transformer ratio k is calculated from the internal barrel radius (R), the inner screw radius (r) and the screw pitch (P) as

$$k = (R^2 - r^2)P/2 \qquad (6.1)$$

and ω_s is given in radians/sec.

Polymer inertia is assumed insignificant at this point due to the low translational velocity of polymer through the extruder.

The second transformation to temperature and enthalpy flow rate is achieved by an unconventional application of a 2-port 'RS' element, where the enthalpy flow rate is given by:

$$\frac{dh_s}{dt} = t_s\omega_s$$

and the temperature is determined by the sub-system into which the enthalpy flows.

The other external contribution of enthalpy flow to the polymer is from the electrical heaters around the barrel, as illustrated by the bond graph sub-model shown in figure 6.4

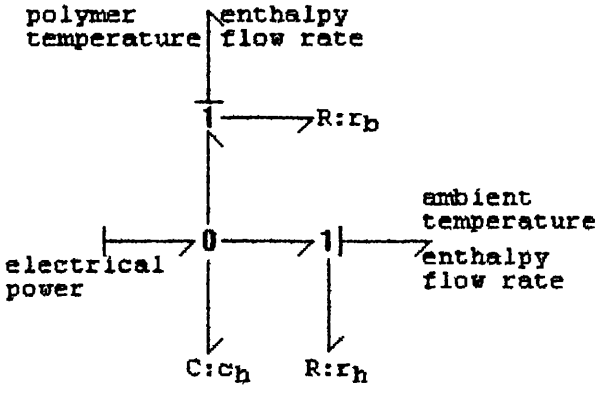


Figure 6.4 Sub-model of barrel heater sub-system

The input and output bonds on this bond graph have known causalities, as illustrated. The input flow source supplies electrical power - this is applied as a constant a.c. voltage to a resistive heater, with power controlled by linearly pulse-width modulating the on/off switching. The electrical power is sourced directly into the thermal capacitance c_h of the heater. There are two effort inputs, the polymer temperature and the ambient temperature, which are needed to calculate the enthalpy flows into the polymer and into the extruder environment, respectively. r_b models the thermal resistance of the barrel between the heater and

the polymer, while r_h models the thermal resistance between the heater and its environment.

The last sub-system to be modelled is the polymer flow through the extruder die, and is shown in figure 6.5.

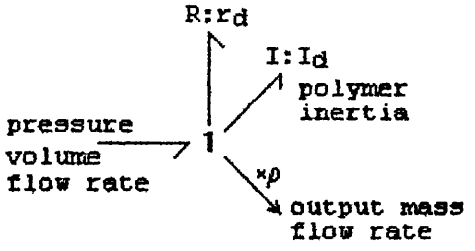


Figure 6.5 Acausal bond graph of polymer extrusion through die

At the end of the barrel the molten polymer is forced through a screen filter and thence through the die resulting in a very high shear friction loss r_d.

Taking the control volume approach to calculating flow inertia[9], the inertia of the polymer extruded from the die is given by:

$$I_d = \rho l/a$$

where ρ is the density of the polymer,

l is the length of the die channel

and a is the cross-sectional area of the die.

The large reduction in cross-sectional area of the polymer flow as it passes through the die results in a rapid increase in linear velocity, such that polymer inertia I_d becomes a significant element. The output mass flow rate is a signal which is measured (implicitly) so that the cross-sectional area of the cooled polymer can be automatically controlled.

These five sub-systems have been aggregated in the bond graph illustrated in figure 6.6, where causality has been completed as shown. The complete bond graph has been slightly simplified by including the motor armature friction and moment of inertia with the corresponding parameters r_s and j_s for the screw, since the

latter are the dominant effects. The bonds on the graph have been numbered for reference purposes, e.g. $e_1 = e_m$.

The polymer (melt) temperature is shown as an additional output, as this variable is normally measured and automatically controlled by varying the electrical power into the barrel heaters.

In the bond graph of figure 6.6, the dissipators r_s and r_d represent irreversible energy dissipation due to shearing of the polymer. For r_d, this energy passes out of the extruder and is dissipated in the environment, so a conventional R element is used. For r_v (the viscous friction of polymer moving through the barrel) and r_s, the energy dissipated becomes an enthalpy flow into the thermal capacitance of the polymer, via RS elements. The constitutive equations for r_s and r_d are given[57] by:

$$e_7 = r_s(T_m) f_7 = r_s(T_m) \omega_s$$

$$\text{and} \quad e_{12} = r_d(T_m) f_{12} = r_d(T_m) k\omega_s \tag{6.2}$$

since the polymer volume flow rate $v = k\omega_s$.

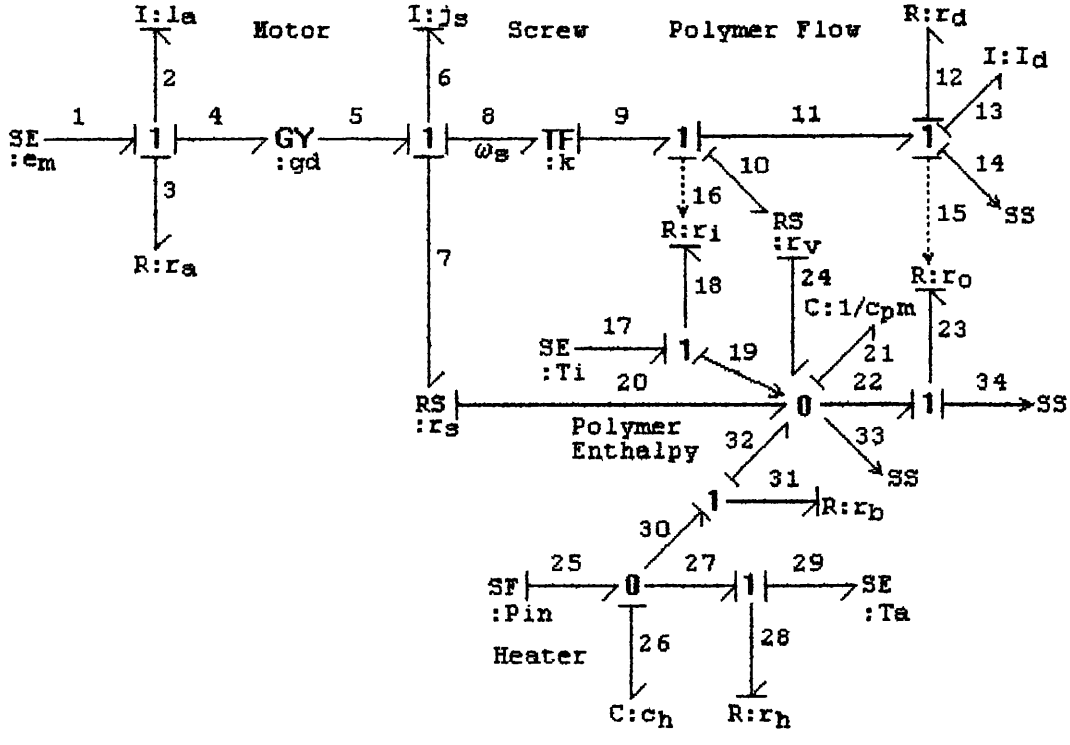


Figure 6.6 Bond graph of extruder metering section

The viscous friction has the constitutive equation

$$e_{10} = r_v(T_m) f_{10} = r_v(T_m) k\omega_s \tag{6.3}$$

where $r(T_m)$ indicates the dependence of the viscosity and shear dissipation on polymer temperature, and

$$T_m = T_{21} = h_{21}/(c_p m) \tag{6.4}$$

where m is the mass of polymer melt in the barrel section, and c_p is the specific heat of the polymer.

The electrical heater sub-model gives equations relating enthalpy flow and temperature.

$$f_{25} = P_{in} \tag{6.5}$$

$$T_{26} = T_h = h_{26}/(c_{ph} m_h) \tag{6.6}$$

$$f_{27} = (T_h - T_a)/r_h \tag{6.7}$$

$$f_{30} = (T_h - T_m)/r_b \tag{6.8}$$

$$f_{26} = f_{25} - f_{27} - f_{30} \tag{6.9}$$

The enthalpy equations for the melt polymer are

$$f_{19} = e_{18} f_{16}/r_i = T_i v \rho c_p \tag{6.10}$$

$$f_{20} = e_7 f_7 = r_s(T_m) \omega_s^2 \tag{6.11}$$

$$f_{22} = e_{23} f_{15}/r_o = T_m v \rho c_p \tag{6.12}$$

$$f_{24} = e_{10} f_{10} = r_v(T_m) v^2 \tag{6.13}$$

$$f_{32} = f_{30} \tag{6.14}$$

$$f_{21} = f_{19} + f_{20} - f_{22} + f24 + f_{32} \tag{6.15}$$

Hence the state equations can be obtained from the bond graph in the usual manner, and the dynamic response of the system (to changes in screw speed, for example) may be obtained.

6.2.3. Deriving the steady state model

For this example, we wish to derive the steady state equations for the system, in order to predict the output mass flow rate and the

melt temperature for a given set of input conditions. This is achieved using the new algorithm described in section 4.3.1., where the first step - generating the dynamic model with integral causality -has been performed in the preceding section. Replacing energy stores by source-sensors u1 - u5, we get the steady state bond graph shown in figure 6.7, where $u_5 = 0$ since I_d has derivative causality, and the outputs to source-sensors $w_1 = w_2 = w_3 = w_4 = 0$.

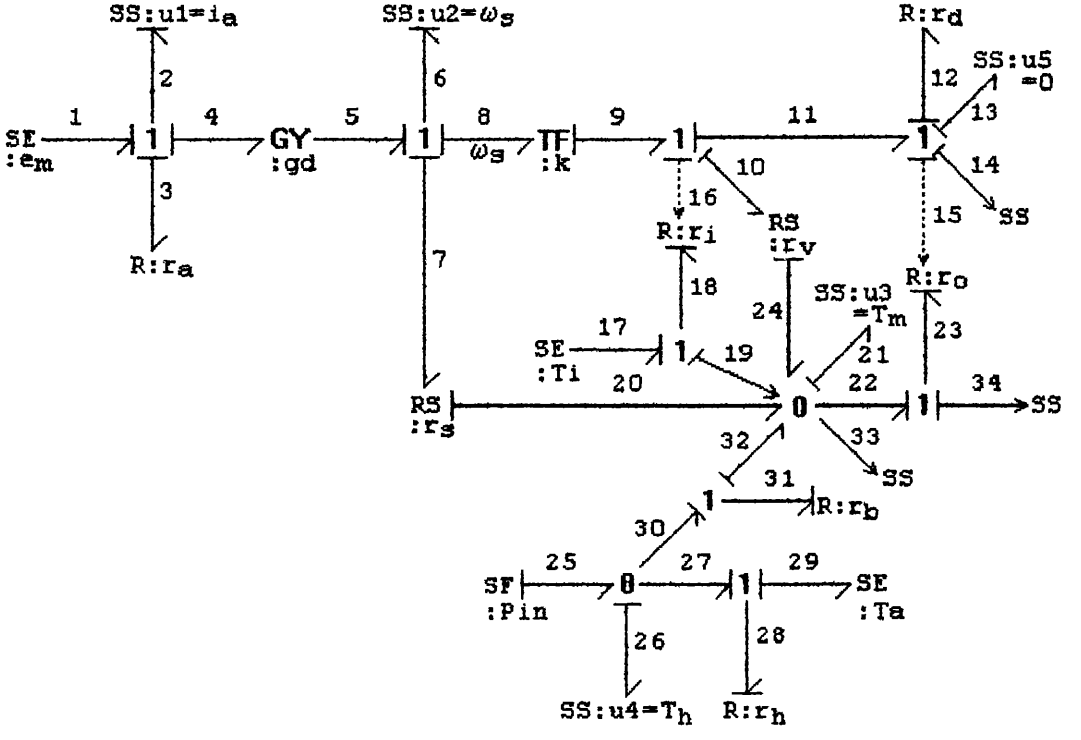


Figure 6.7 Steady state bond graph for extruder

Propagating causality for the steady state bond graph gives:

$$e_2 = w_1 = e_m - i_a r_a - \omega_s/gd \tag{6.16}$$

and $$e_6 = w_2 = i_a/gd - \omega_s(r_s + k^2(r_v + r_d)) \tag{6.17}$$

Hence $$\omega_s = \frac{e_m gd}{1 + r_a gd^2 (r_s + k^2(r_v + r_d))} \tag{6.18}$$

From equations (6.5) to (6.9)

$$f_{26} = w_4 = P_{in} - (T_h - T_a)/r_h - (T_h - T_m)/r_b \tag{6.19}$$

and from equations (6.10) to (6.15)

$$f_{21} = w_3$$

$$= k\omega_s\rho c_p(T_i - T_m) + r_s\omega_s^2 + r_v k^2\omega_s^2 + (T_h - T_m)/r_b \qquad (6.20)$$

Hence,

$$T_m = \frac{T_a + P_{in}r_h + (k\omega_s\rho c_p T_i + r_s\omega_s^2 + r_v k^2\omega_s^2)(r_h + r_b)}{1 + k\omega_s\rho c_p(r_h + r_b)} \qquad (6.21)$$

Note that since r_s, r_v and r_d relate to the polymer viscosity, they are all temperature dependent, and thus equations (6.18) and (6.21) must be solved iteratively. In practice, the temperature T_m is maintained approximately constant by automatic control loops, and thus the polymer viscosity and, therefore, r_s, r_v and r_d are also approximately constant. Equation 6.21 is therefore used to optimise the extruder design for a given maximum throughput, such that the melt temperature is maintained by work heat, and the electrical input P_{in} is minimised.

The variable of interest for calculating the mass output of the extruder is the volume flow rate through the die:

$$f_{14} = f_{11} = f_9 = kf_8$$

i.e. output mass flow rate $= (R^2 - r^2)P\omega_s\rho/2$

where ω_s is given by (6.18).

Thus the extruded diameter of plastic on the wire may be calculated as a function of the output mass flow rate[58], and adjusted by controlling the angular velocity of the extruder screw.

6.3. A drum boiler - turbine model

This system is the thermal unit of an electric power generation plant and has been modelled by Eklund[59] and Lindahl[60] in order to design a co-ordinated control system, with the aid of a simulation

model based on physical laws. Elmquist[53] subsequently converted Lindahl's FORTRAN simulation into a Simnon model. Lindahl takes the standard approach to analysing large systems by deriving the mathematical equations for each sub-system independently and aggregating all these in the final (simulation) application. The aim of the modelling exercise in this section is to investigate the use of bond graphs for deriving the same model, in order to highlight the strengths and weaknesses of bond graphs for modelling large hierarchical systems. In addition, the new techniques for handling under-causal systems and the use of the computable causality algorithm will be demonstrated.

Since bond graph modelling is based on understanding the physical laws of the real system, this model relates most closely to the Lindahl model rather than that produced by Elmquist which consists of simulation equations. The model has over 500 variables and thus a formal naming convention was defined and is re-used in this section. The derived equations used are those from the original work, although in some instances the simplifications used result in unconventional bond graphs.

6.3.1. Functional description

The hierarchical view of the system is shown in figure 6.8, which is replicated from Lindahl's thesis, and a brief overview follows.

Feedwater is pumped at high pressure through the preheaters 1 to 7 into the economiser where it is heated from 230°C to 290°C. The water output from the economiser feeds the drum system, where heat from the combustion chamber generates steam at 300-340°C, which passes to the superheaters. Attemperators between the first and second, and second and third superheaters cool the steam by about 30°C by spraying in water derived from the final feedwater preheater. Steam at 540°C passes through a valve which controls the flow rate into the high pressure turbine, from which it passes into a reheater and then into intermediate and low pressure turbines. Steam is extracted from each of the turbines to heat the feedwater in the preheaters, whereas the combustion chamber provides the heat input to the economiser, drum system and each of the superheaters.

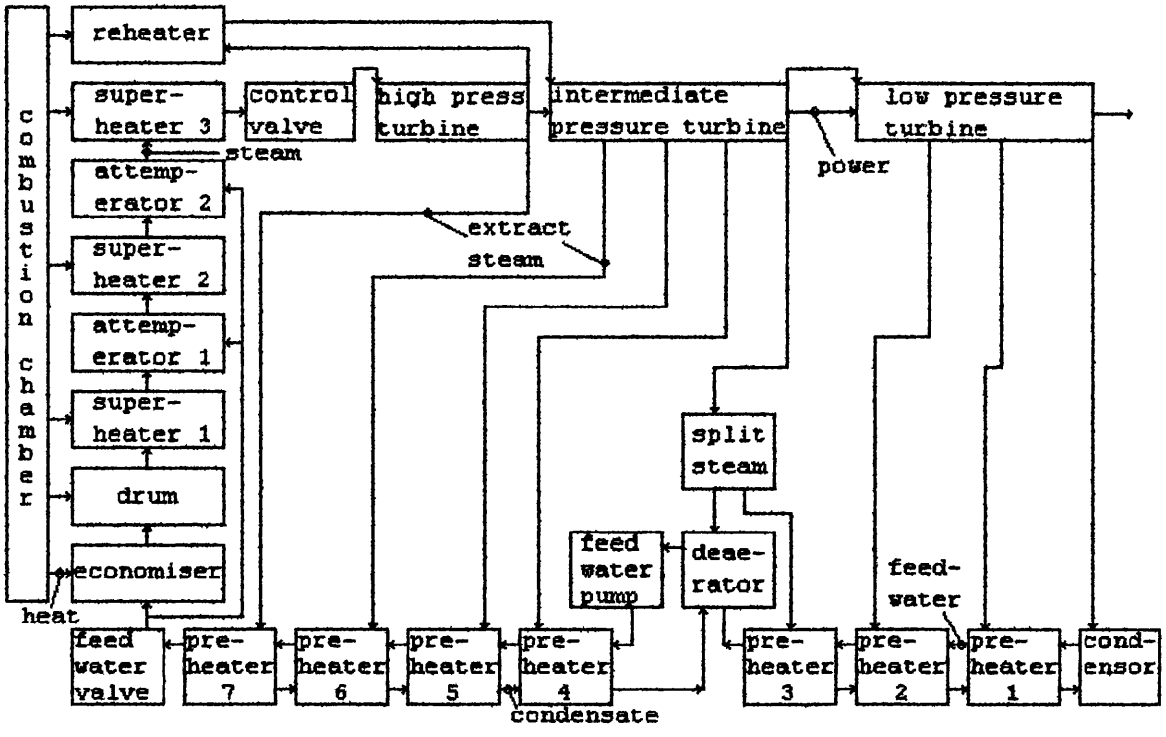


Figure 6.8 Schematic diagram of the thermal unit

Finally, steam leaving the last section of the low pressure turbine is taken to the condenser so that the condensate can be recycled into the low-pressure feedwater preheaters, before passing back through the de-aerator and feedwater pump.

6.3.2. Developing a bond graph model

This model is decomposed into the sub-models described in figure 6.8, and the physical equations and assumptions made for each sub-model are reproduced from the original model. In several cases, assumptions were made by Lindahl to simplify the overall model, and in all cases the equivalent bond graph model is designed to be consistent with these simplifications.

6.3.2.1. High pressure feedwater sub-model

Bond graph modelling of the system is started at the feedwater pump as this is the input source for the hydraulic flows in the system. The original high pressure feedwater sub-model was simplified by ignoring the water flows to the attemperator valves, and translates to the pseudo bond graph shown in figure 6.9, where pressure is the

effort variable and mass flow is the flow variable. It is assumed that the back-pressure p_{ds2} from the drum system is defined as an input to this sub-model.

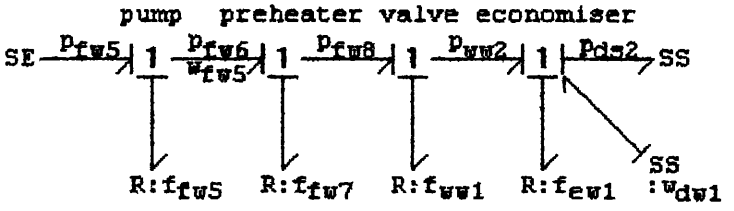


Figure 6.9 High pressure feedwater sub-model

This bond graph is under-causal, so the algebraic loop is resolved, using the method described in section 4.2.2, by adding an auxiliary flow source $f_0 = w_{wd1}$, with the constraint $e_0 = 0$. This results in the algebraic equation:

$$w_{fw5} = \sqrt{\frac{|\Delta p|}{f_{fw5}+f_{fw7}+f_{ww1}/a_{ww1}^2+f_{ew1}}} \qquad (6.22)$$

where $\Delta p = p_{fw5} - p_{ds2}$

The following section shows that the original simplification of ignoring the water flows to the attemperator valves results in inconsistencies which are highlighted by the bond graph model.

6.3.2.2. Steam flow sub-model

This models the steam flow from the drum, where it is generated, through each of the three superheaters and the two spray attemperators and the steam control valve, before it passes into the high pressure turbine. Here, the original model merged the hydraulic equations relating to pressure drops through each sub-system, with the ideal gas law to obtain a composite law, as shown below. The pressure drop $(p_1 - p_2)$ through an orifice of area A, is given by

$$p_1 - p_2 = \frac{0.5}{\rho} \left(\frac{w}{A}\right)^2 \qquad (6.23)$$

where w is the mass flow rate.

The ideal gas law is then applied using the *average* pressure for the gas upstream and downstream of the orifice, assuming that the steam temperature T, and density ρ remain constant:

$$0.5(p_1 + p_2)/\rho = RT \tag{6.24}$$

Equations (6.23) and (6.24) are then combined to give

$$p_1^2 - p_2^2 = \frac{RT}{A_{max}^2}\left(\frac{A_{max}w}{A}\right)^2 = f\left(\frac{w}{a}\right)^2 \tag{6.25}$$

where a is the normalised area of the orifice and f is an equivalent friction coefficient.

This combination of constitutive relations is inappropriate for a bond graph model, since a major advantage of bond graphs is that they expose the underlying behaviours. However, the equations can be modelled using a pseudo bond graph (figure 6.11) where p^2 is the effort variable and w is the flow variable.

Alternatively, the complete behaviour for a single superheater could be modelled as an energy bond graph as shown in figure 6.10, thus highlighting the conversion of energy between domains.

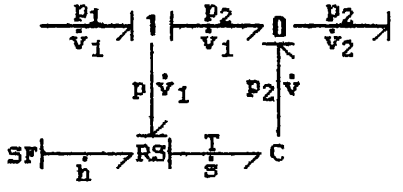


Figure 6.10 Energy bond graph for gas flow through an orifice

Here, the pressure drop across the orifice is modelled by the 1-junction, and the pressure difference (p) is due to dissipation in an RS element, where the energy loss (pv_1) is converted irreversibly to the thermal domain. Some of the thermal energy (Ts) is converted by the two-port C energy store back into the hydraulic domain, and the volume flow rate v' is determined by the difference ($v_1' - v_2'$). This model has the advantage that enthalpy flows from the combustion chamber can easily be added, as shown by the flow source h'. Since the intention of this case study is to replicate

the original model as closely as possible, the pseudo bond graph representation will be used for this sub-model instead.

After the primary and secondary superheaters, the steam is sprayed with high pressure feedwater in the attemperators, resulting in additional mass flow rates into the secondary and tertiary superheaters respectively:

$$w_{sw1} = a_{sw1}\sqrt{\frac{|p_{ww2} - p_{ps2}|}{f_{sw1}}}$$

$$w_{tw1} = a_{tw1}\sqrt{\frac{|p_{ww2} - p_{ss2}|}{f_{tw1}}} \qquad (6.26)$$

where p_{ww2} is the pressure of water leaving the feedwater valve

p_{ps2} is the pressure of steam leaving the primary superheater

p_{ss2} is the pressure of steam leaving the secondary superheater

a_{sw1} and a_{tw1} are the areas of the areas of the spray flow valves

f_{sw1} and f_{tw1} are the pressure drop coefficients of the spray flow valves

In this case, the feedwater sub-model has an additional 0-junction, which indicates the pressure p_{ww2} causing the water flow through the attemperator spray valves. Since the attemperator spray is ignored in the original model, signals must be used instead of bonds in the bond graph linking the steam flow model with the feedwater system (figure 6.11).

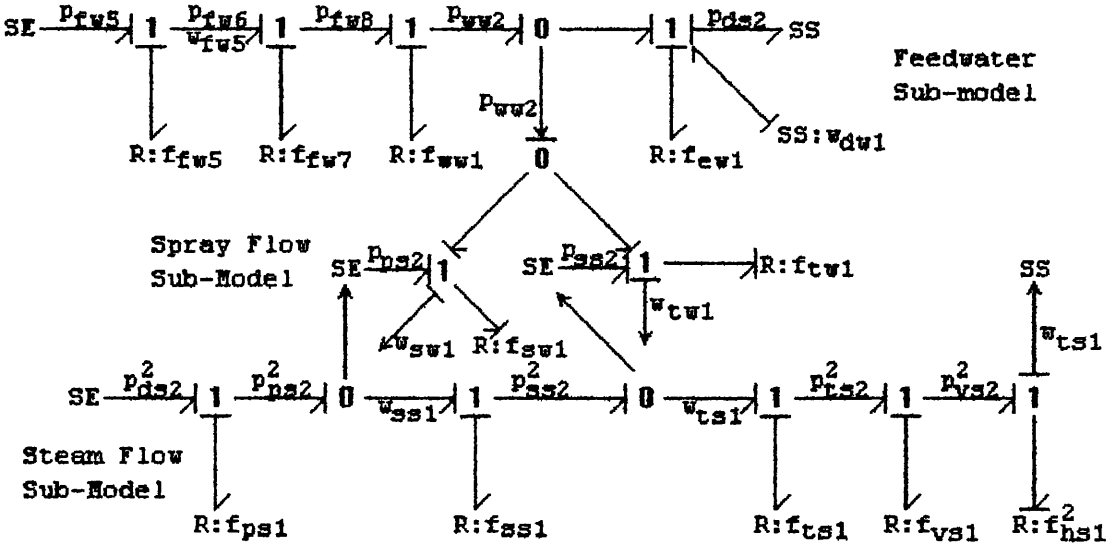


Figure 6.11 Pseudo bond graph of feedwater and steam flow systems

In this pseudo bond graph, modulated effort sources have been used to convert from the steam flow sub-model, where the effort variable is p^2, to the rest of the graph where the effort variable is p. The steam pressure outputs from the primary, secondary and tertiary superheaters are p_{ps2}, p_{ss2} and p_{ts2} respectively, while p_{vs2} indicates the pressure output from the steam pressure at the control valve output. The corresponding dissipators f_{ps1}, f_{ss1} and f_{ts1} all have the same constitutive relation

e.g. $p_{ss2}^2 - p_{ts2}^2 = f_{ts1} w_{ts1}^2$

while that for the control valve is

$$p_{ts2}^2 - p_{vs2}^2 = f_{vs1} \left(\frac{w_{ts1}}{a_{vs1}}\right) 2$$

where the pressure after the control valve is determined by the pressure coefficient of the high pressure turbine:

$$p_{vs2} = f_{hs1} w_{ts1}$$

The simplification that the additional mass flows (due to the attemperator sprays) are ignored, is carried over from the original model.

i.e. $w_{ts1} = w_{ss1} = w_{ps1}$ = steam output from drum.

Again, signals from the 0-junctions (the superheater pressure outputs) are used to model this simplification.

6.3.2.3. Economiser heat flow sub-model

Feedwater entering the economiser is heated by the combustion gases from the combustion chamber such that the water temperature T_{ew2} is raised from 230 to 290°C. The input from the combustion chamber is Q_{em2} (kJ/sec) which heats the metal economiser skin to a temperature T_{em2}, resulting in an enthalpy flow to the water, which is determined by the thermal resistance T_{ew3}. The equation relating the combustion heat input to the temperature difference has been simplified in the original model by ignoring the resulting time constant:

$$T_{em2} - T_{ew2} = T_{ew3}Q_{em2} \tag{6.27}$$

The equations for this sub-system are similar to the heated tank example given in section 2.7.2, and the bond graph is shown in figure 6.12.

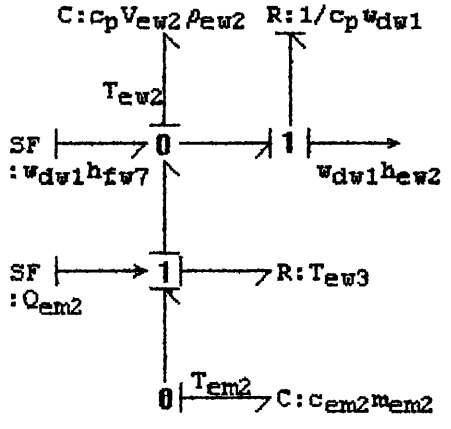


Figure 6.12 Pseudo bond graph model of economiser heat flows

All enthalpies in the original model are expressed in KJ/kg and thus the enthalpy flows with the water entering and leaving the economiser are respectively $w_{dw1}h_{fw7}$ and $w_{dw1}h_{ew2}$. The resulting balance of enthalpy flows gives:

$$\frac{d}{dt}(c_{em2}m_{em2}T_{em2} + V_{ew2}\rho_{ew2}h_{ew2}) = Q_{em2} + w_{dw1}hfw7 - w_{dw1}h_{ew2} \tag{6.28}$$

This simple enthalpy model is repeated identically for each of the three superheaters, but with steam (rather than water) being heated to a higher temperature in each sub-system.

6.3.2.4. The drum system sub-model

The drum system consists of the drum, risers where the water is converted to steam, and downcomers which feed water from the drum into the risers. The risers and downcomers are the same length, and water from the drum circulates via the downcomers into the risers, due to the force difference between the gravitational pressure on the water in the risers and that due to the lower density steam-water mixture in the risers. The drum supplies steam to the primary superheater, with this mass loss being replaced by high pressure water from the Feedwater sub-system.

The flow of water and steam-water mix round the downcomer and risers is given by:

$$gL_{dl5}\rho_{dl4} = f_d w_{dl5}^2 + f_r w_{dl7}^2 + gL_{dl5}\rho_{dl8}$$

and assuming the mass flow rate in the risers is the same as that in the downcomers ($w_{dl7} = w_{dl5}$) then

$$w_{dl5} = \sqrt{gL_{dl5}\frac{|\rho_{dl4} - \rho_{dl8}|}{f_d + f_r}} \qquad (6.29)$$

This is modelled by the simple hydraulic bond graph shown in figure 6.13a, where m_d and m_r represent the mass of the water in the downcomer and the mix in the riser respectively.

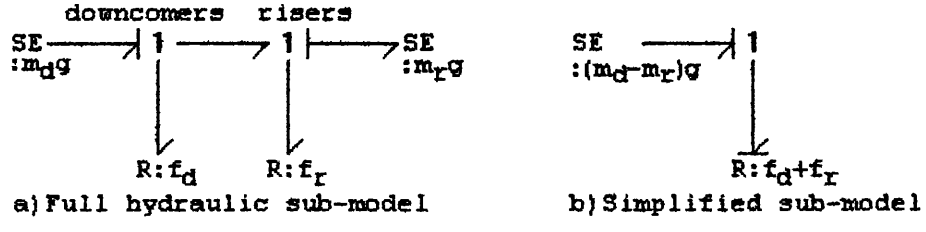


Figure 6.13 Pseudo bond graph of downcomer-riser system hydraulics

This sub-model is under-causal, although the simplified sub-model shown in figure 6.13b is causally complete and represents equation (6.29). This bond graph is simplified in order to clarify the overall drum system bond graph, shown in figure 6.14.

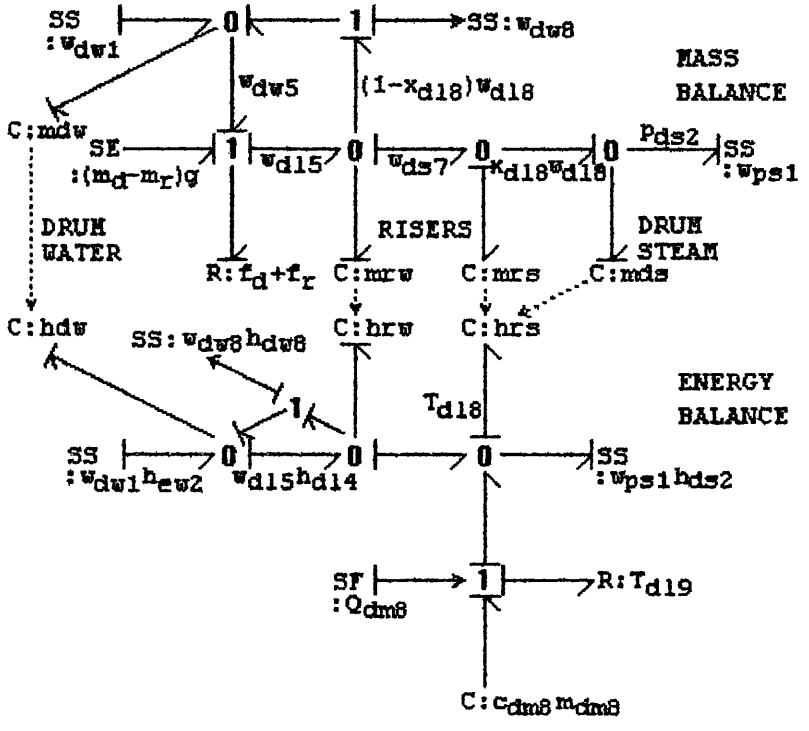


Figure 6.14 Pseudo bond graph of mass and enthalpy flow for steam generation

The full drum system sub-model (figure 6.14) can be considered as composed of a mass flow sub-model, from which the mass balance equations are derived, and an enthalpy flow sub-model, resulting in the energy balance equations. The mass flow passes between for hydraulic capacitances which have been labelled as follows:

mdw water in the drum

mrw water in the risers

mrs steam in the risers

mds steam in the drum

The mass flow rate of water into the drum from the economiser is represented by an `SS' element (w_{dw1}), and this water circulates to the risers via the downcomers with the flow rate w_{d15} (= w_{dw5}) as indicated by the sub-model of figure 6.13b. Some of this water is

converted to steam at the steam production rate w_{ds7}, while the remainder ($w_{dw8} = (1-x_{dl8})w_{dl8}$) recirculates into the drum. x_{dl8} is the steam quality of the steam-water mix leaving the risers and is defined as:

$$0.5x_{dl8} = \frac{\text{mass steam in risers}}{\text{total mass in risers}} = \frac{V_{ds8}\rho_{ds8}}{V_{dl8}\rho_{dl8}} \qquad (6.30)$$

Thus, the mass flow rate of steam passing to the drum is $x_{dl8}w_{dl8}$, some which is stored in the drum capacitance, mds, while the remaining mass flow (w_{ps1}) passes to the superheater.

Each of these capacitances results in a mass balance equation, which is identified in Lindahl's original work. For example, the mass balance for the steam-water mix in the risers and the steam in the drum is given as:

change of (water mass + steam mass) = water input - water output - steam output

i.e.

$$\frac{d}{dt}((V_{dl8}-V_{ds8})\rho_{dw8} + (V_{ds2}+V_{ds8})\rho_{ds2}) = w_{dl5} - w_{dw8} - w_{ps1} \qquad (6.31)$$

where V_{dl8} is the volume of the risers and V_{ds8} is the volume of steam in the risers.

The corresponding energy balance of the riser metal, the steam-water mix in the risers and the steam in the drum is given as:

change of enthalpy of (metal + water + steam) =

combustion input + water input - water output - steam output

i.e. $$\frac{d}{dt}(c_{dm8}m_{dm8}T_{dm8} + (V_{dl8}-V_{ds8})\rho_{dw8}h_{dw8} + (V_{ds2}+V_{ds8})\rho_{ds2}h_{ds2})$$

$$= Q_{dm8} + w_{dl5}h_{dl4} - w_{dw8}h_{dw8} - w_{ps1}h_{ds2} \qquad (6.32)$$

It can be seen that these equations (6.32) consider the enthalpy of the steam in the risers and the drum together, and so these are represented by a single enthalpic capacitance, hrs, in figure 6.14. However since separate equations were derived for the mass of steam in the risers and in the drum, these are modelled as separate

capacitances in the bond graph. This model is a derivative[61] of the two tank model, formalised in section 3.5, where the hydraulic mass states modulate the enthalpic C parameters. In figure 6.14, there is the additional complication of the bond graph cycle as enthalpy ($w_{dw8}h_{dw8}$) flows with the recirculated water back into the drum.

The heat flow into the risers is modelled as an enthalpy source Q_{dm8} through a thermal resistance T_{dl9}, where the bond graph models the original temperature equation:

$$T_{dm8} = T_{dl8} + T_{dl9}\sqrt[3]{Q_{dm8}} \tag{6.33}$$

In the original model, the differential equations are generated for a set of overlapping sub-systems, and thus, a similar mass and energy balance is performed for the steam and water in the drum system. For the complete drum system, the *steam* mass balance is:

change of (steam mass)

= total steam production - total out to superheater

i.e. $$\frac{d}{dt}((V_{ds2}+V_{ds8})\rho_{ds2}) = w_{ds7} - w_{ps1} \tag{6.34}$$

where V_{ds2} is the volume steam in the drum and V_{ds8} is that in the risers.

The corresponding *water* mass balance is:

-(change of steam volume)water density

= feedwater input - steam production

i.e. $$\frac{d}{dt}(((V_{tot} - (V_{ds2}+V_{ds8}))\rho_{dw8}) = w_{dw1} - w_{ds7} \tag{6.35}$$

where V_{tot} is the total volume of the drum system.

These two equations are part of the overall model described by the bond graph of figure 6.14. It can be seen from the causal augmentation of the bond graph model that this highlights some of the assumptions made. In particular the lack of dissipators in the mass flow model, indicates that the pressure is assumed to be the same for each capacitance (= steam pressure, p_{ds2}), and thus mdw, mrw and mds each have derivative causality. This choice of V_{ds2} as

a state thus conflicts with choosing the level of feedwater in the drum (z_{dl4}), as an independent state, in the original model. (In this model, the pressure differential due to the head of water is neglected.) Similarly derivative causality is forced on the enthalpic capacitances hdw and hrw, with hrs and $c_{dm8}m_{dm8}$ having integral causality. Thus the water temperature is assumed to be equal to that of the steam T_{dl8}.

In the original model the steam quality (x_{dl8}) and pressure (p_{ds2}) were chosen as additional states for this sub-model, but these cannot be automatically obtained as states from the bond graph.

6.3.2.5. The superheater sub-model

Saturated steam from the drum enters the primary superheater, where it is heated such that, on exit to the spray attemperator the temperature has risen to 450-480°C. This sub-system can be represented by the familiar mass flow and enthalpy flow pseudo bond graph, as shown in figure 6.15.

The mass flow equation is:

$$\frac{d}{dt}(V_{ps2}\rho_{ps2}) = w_{ps1} - w_{ps2} \qquad (6.36)$$

and the corresponding enthalpy equation is:

$$\frac{d}{dt}(c_{pm2}m_{pm2}T_{pm2} + V_{ps2}\rho_{ps2}h_{ps2}) = Q_{pm2} + w_{ps1}h_{ps1} - w_{ps2}h_{ps2} \qquad (6.37)$$

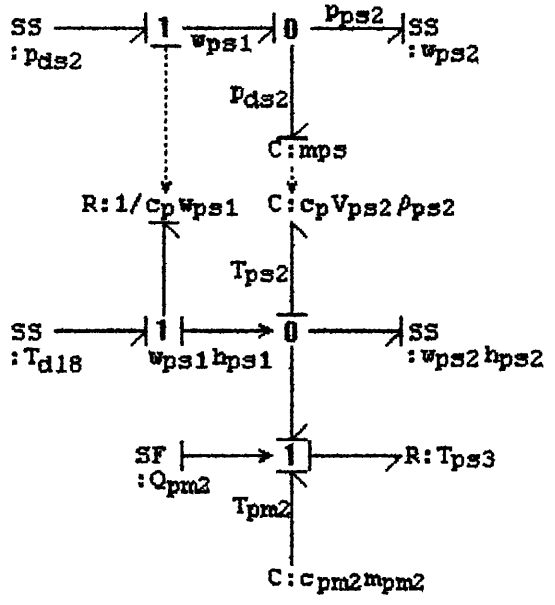


Figure 6.15 Pseudo bond graph of mass and enthalpy flow for superheaters

As before, the mass of the gas in capacitance, mps, modulates the enthalpic capacitance, while the output mass flow modulates a conceptual resistor R to give the output enthalpy flow. The hydraulic capacitance is drawn with derivative causality to reflect the ordering of the original equations and the assumption that $w_{ps1} = w_{ps2}$.

Identical bond graphs may be drawn for each of the remaining superheaters, and the reheater which follows the high pressure turbine. In contrast, the mass of steam in the attemperators is assumed to be insignificant, and thus, the energy balance for each attemperator reduces to a simple algebraic equation:

$$wss1hss1 = wps1hps2 + wsw1hfw7$$

i.e. energy into secondary superheater =

steam energy from primary superheater + attemperator water energy

6.3.2.6. The high pressure turbine sub-model

The superheated steam from the tertiary superheater passes through the control valve into the high pressure turbine, where some of the steam enthalpy is converted to mechanical energy. The steam leaves the high pressure turbine at a reduced pressure and temperature,

and thus most of this exit steam passes through a reheater before entering the intermediate turbine. The remainder is extracted through a valve to the high pressure feedwater heater.

Previous work[62] has indicated that an energy bond graph is not well suited to describing the behaviour of a turbine, and so a pseudo bond graph model (figure 6.16) is used, giving consistency with the other sub-models for this system.

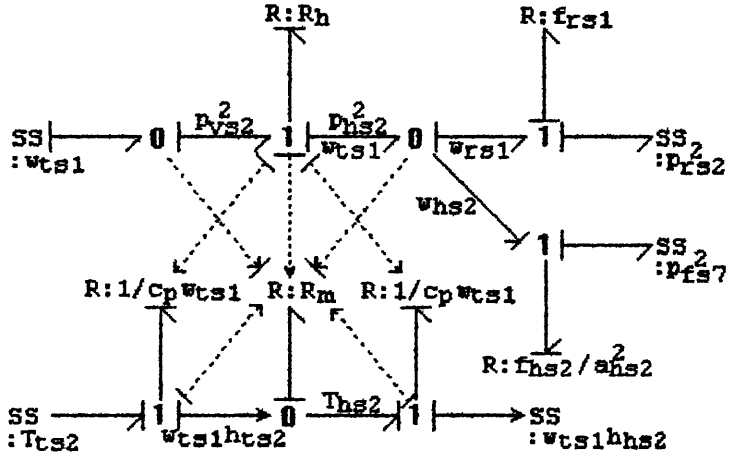


Figure 6.16 Pseudo bond graph model of turbine and reheater

In this model the mechanical output power (N_{hs2}) from the turbine is represented by the dissipation R_m:

$$N_{hs2} = w_{ts1}(h_{ts2} - h_{hs2}) \qquad (6.38)$$

where R_m is modulated by p_{vs2}, p_{hs2}, h_{ts2} and h_{hs2}, according to functions defined in the original thesis:

$$h_{hs2} = h_{hs2}^{*} + (1-\eta_{hs2})(h_{ts2}-h_{hs2}^{*})$$

$$h_{hs2}^{*} = \text{ISENX}(h_{ts2}, p_{vs2}, p_{hs2})$$

and $$T_{hs2} = \text{THP}(h_{hs2}, p_{hs2}) \qquad (6.39)$$

ISENX is a function defining the expansion in the turbine, characterised by a constant isentropic efficiency.

It can be seen that this sub-model emphasises the difficulties in deriving a model which is close to the physical system structure, from a set of pre-defined mathematical behaviours. The result is a modulated component, R_m, which hides many of the physical behaviours.

The mass flow wrs1 in the reheater results in a pressure drop across the dissipator frs1:

$$p_{hs2}^2 - p_{rs2}^2 = f_{rs1} w_{rs1}^2 \qquad (6.40)$$

The mass flow whs2 to the feedwater preheater results in a pressure drop across the extraction valve, represented by the dissipator fhs2/ahs22:

$$p_{hs2}^2 - p_{fs7}^2 = f_{hs2} w_{hs2}^2 / a_{hs2}^2 \qquad (6.41)$$

Similar pseudo bond graph sub-models represent the intermediate and low pressure turbine sub-systems, but with slight differences due to the extraction of steam (to the preheaters) at intermediate points in these turbines.

6.3.2.7. Feedwater enthalpy sub-model

Steam is extracted from the intermediate sections of the turbines to the feedwater preheaters where it condenses, passing heat to the feedwater. Condensate also passes out of the condenser into the low-pressure feedwater preheater and some condensate from the feedwater preheater also enters the steam side of the condenser. In consequence there is a flow of condensed feedwater from the condenser (due to the feedwater pump), while there is a smaller flow of condensed steam back from each preheater stage to the previous stage. This is represented by the pseudo bond graph shown in figure 6.17

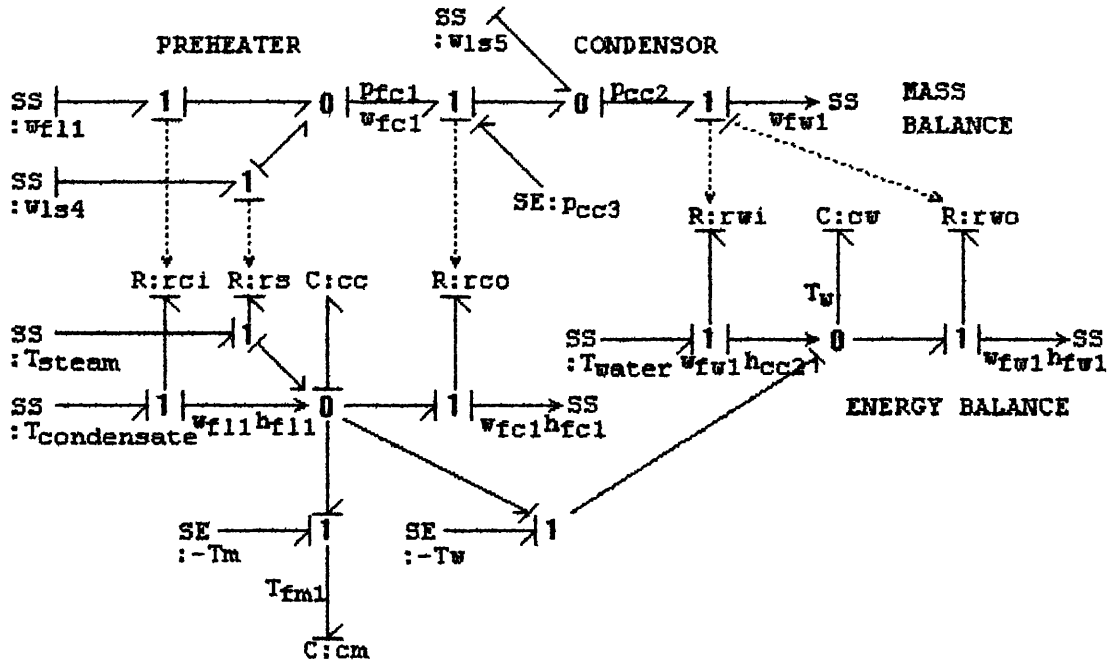


Figure 6.17 Pseudo bond graph of mass and enthalpy flows in preheater 1

The top part of the bond graph represents the mass balance equations for the steam and condensate flows in the preheater, resulting in condensate flow, w_{fc1}, back into the condenser:

$$w_{fc1} = w_{fl1} + w_{ls4} \tag{6.42}$$

The condensate entering the condenser plus the steam flow, w_{ls5}, from the final stage of the turbine balance the mass flow of feedwater, w_{fw1}, out from the condenser:

$$w_{fw1} = w_{fc1} + w_{ls5} \tag{6.43}$$

It should be noted that the bond graph arrows do not indicate the direction of theses hydraulic flows. These flows are used to modulate the dissipators, rci, rs, rco, rwi and rwo, in the usual manner, to give the appropriate enthalpy flows associated with each mass flow. The overall energy balance equation is then:

$$\frac{d}{dt}(V_{fc1}\rho_{fl1}h_{fl1} + c_{fm1}m_{fm1}T_{fm1} + V_{fw1}\rho_{fw1}h_{fw1})$$

$$= w_{ls4}h_{ls4} + w_{fl1}h_{fl1} + w_{fw1}h_{cc2} - w_{fw1}h_{fw1} - w_{fc1}h_{fc1} \tag{6.44}$$

The assumption that there is constant temperature difference between the temperature of the metal and that of the condensate is represented by the effort source -Tm. Similarly, the assumption that there is a constant difference between the output feedwater *specific* enthalpy and that of the condensate is represented by a second effort source -Tw. It can be seen that these assumptions result in derivative causality being imposed on the capacitances, cm and cw, representing the enthalpy stored in the metalwork and the feedwater, respectively. Finally, the assumption that there is a constant difference between the saturation pressure of the steam in the condenser and that of the condensate in the first preheater is represented by the effort source p_{cc3}.

Similar bond graph models can be created for each of the feedwater preheaters and for the condenser, which is represented in part in this model.

This section has shown that the complete system can be modelled from the original equations, using pseudo bond graphs, with some knowledge of the underlying physical laws. However, this 'reverse engineering' of the model is liable to produce some omissions, since the significance of some of the original simplifications may not be fully understood.

6.4. A telephone anti-sidetone circuit

A passive electronic telephone circuit may be considered as comprised of the sub-systems shown in the word bond graph illustrated in figure 6.8, although, in practice, the efficient circuit design uses individual components for more than one function. The circuit is designed to compensate transmission and receive gains for different line lengths and to reduce the feedback of transmitted signals to the user's ear (sidetone) to a level appropriate for normal speech* .

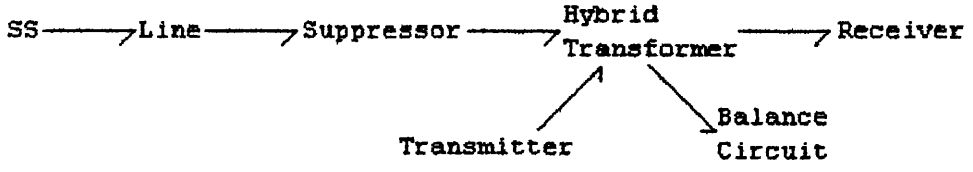


Figure 6.18 Word bond graph of anti-sidetone circuit

This case study will show the use of bond graph modelling for circuit analysis, specifically to obtain the transmission gain and the sidetone gain of the circuit. This has been done previously[63] by mesh techniques appropriate to electrical analysis, and the circuit is analysed here to demonstrate the use of the bond graph technique, described in chapter 4, for solving algebraic loops. This example also provides a comparison between a multi-port representation of the 3-port hybrid transformer, and a flat bond graph, when used to obtain the required transfer functions.

---

* Sidetone refers to that component of the transmitted signal fed back to the speaker via the receiver. This circuit uses a hybrid transformer to reduce this sidetone signal, which would otherwise cause the telephone user to speak more quietly, due to the enhanced level of the acoustic feedback. This effect is used to advantage in this circuit, which is designed to give a greater reduction in sidetone when line lengths are long in order to encourage the speaker to compensate for the greater line losses.

6.4.1. Detailed description

The electrical schematic of this circuit is shown in figure 6.19, including parameter names. The three windings N_1, N_2, N_3 of the hybrid transformer are shown connected in series-aiding fashion, indicated by the dot near one end of the winding.

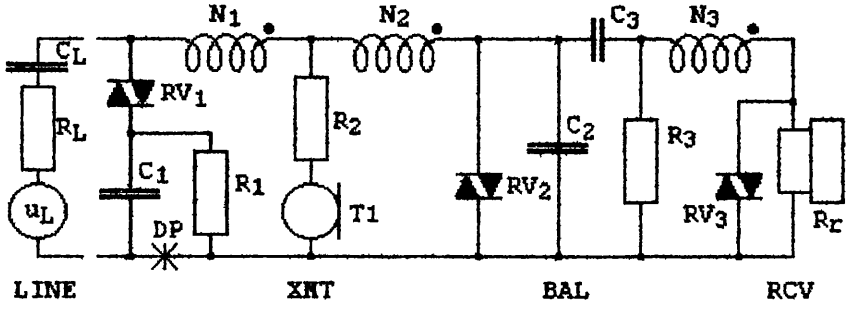


Figure 6.19 Electrical schematic of anti-sidetone circuit.

This arrangement, together with the impedance branches designated LINE, XMT, BAL and RCV results in a `conjugate circuit'. This means that an e.m.f. impressed on branch LINE will result in equal currents in branches XMT and RCV with no current induced in the branch BAL, if the impedances of XMT and RCV are perfectly matched and due account is taken of the turns ratios of the transformer. Conversely, an e.m.f. impressed on branch XMT results in equal currents in branches LINE and BAL and no current in branch RCV, if LINE and BAL are matched. In practice, perfect balance is not achieved, nor desirable, so that a current induced in RCV results in sidetone.

The non-linear varistors RV_1 and RV_2 provide some loop current regulation of the transmit, receive and sidetone characteristics of the telephone set. The magnetic characteristics of the hybrid transformer result in additional non-linear and frequency-dependent elements which affect the regulation and frequency characteristics of the network under the influence of varying loop currents. Because these elements and their effects are not apparent in the schematic of figure 6.19, a further development into an equivalent circuit will be described in the following paragraphs.

The network formed by C_1 and R_1, provides arc suppression across the contacts of the dial DP. R1 is also used to limit the current through RV_1 in the talking mode, while C_1 aids transient suppression. Resistor R_2 stabilises the current in the transmit branch XMT, as the resistance R_t of the transmitter has a very large tolerance. RV_2, C_2, R_3 and C_3 are the components of the balance network BAL, where C3's main function is to block DC from the receiver to prevent demagnetisation, and from R_3 to reduce dissipation. For the purposes of this analysis, the line source impedance is chosen to be the nominal 900Ω in series with 2.16μF.

6.4.2. Developing a bond graph model

The schematic diagram of figure 6.19 may be transformed to its equivalent bond graph representation by using the rules given in section 2.4.1, and treating the hybrid transformer as a multi-port I element. The resulting bond graph is shown in figure 6.20, augmented for integral causality.

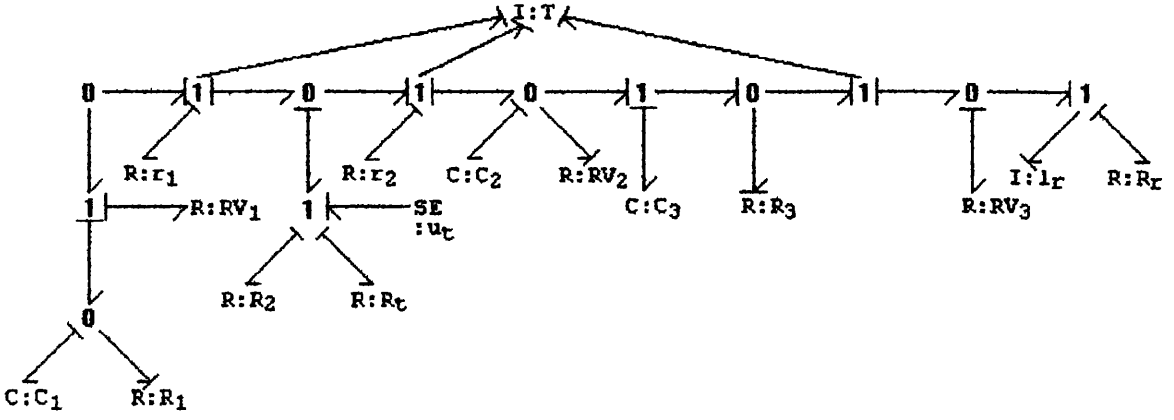


Figure 6.20 Bond graph of the anti-sidetone circuit

Analysis of the circuit using electrical network transformations[63] is achieved by combining individual components in each branch into equivalent impedances, and then deriving the mesh current equations. In addition, the click suppressor RV_3 is ignored for signal analysis due to its very high impedance to small signals. Equivalent bond graph transformations are also documented in the literature[64], but are not used in this thesis - the only equivalent circuit used is that of the 3-port transformer (figure 6.21), which is used to derive the equivalent I-field. Since it is viewed to be

important to retain the system structure in the model, the 3-port I has been chosen to include the dissipator elements which are included in the transformer. Thus r_1, r_2 and r_3 (the winding resistances) are included with the leakage inductances l_1, l_2, l_3, and the core loss R is included with the magnetising inductance L.

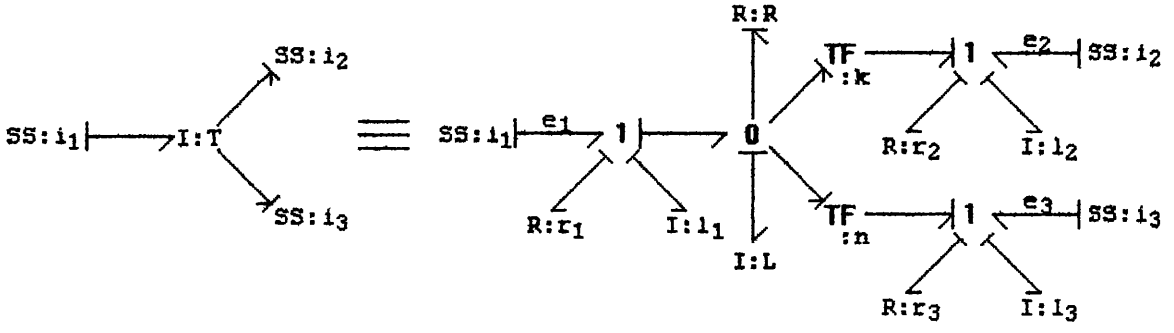


Figure 6.21 Bond graph representation of 3-port transformer

The constitutive equations for this 3-port are most conveniently derived by applying derivative causality to the equivalent model of the transformer, but the resulting I-field (equation 6.45) needs inverting to obtain the integral causal form required for the model of figure 6.20.

$$\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} (r_1+sl_1+Z) & -kZ & -nZ \\ -kZ & (r_2+sl_2+k^2Z) & knZ \\ -nZ & knZ & (r_3+sl_3+n^2Z) \end{bmatrix} \begin{bmatrix} i_1 \\ i_2 \\ i_3 \end{bmatrix} \qquad (6.45)$$

where s is the derivative operator and $Z = \frac{sLR}{R+sL}$.

The problem that arises with this representation is that the 3-port field now includes dissipator elements, and thus the transfer function form cannot be arrived at using the systematic transformations described in chapter 4. The alternative solution method, using the bond graph equivalent of Mason's Rule[64], is cumbersome, and not conducive to implementation as an automated algorithm.

However, replacing the I-field by its equivalent bond graph (figure 6.21) reduces the bond graph to its flat form as shown in figure 6.22, which again is augmented for integral causality. This bond graph also includes the test source u_L and the nominal line source impedance of C_L and R_L in series. The removal of RV_3 now forces

derivative causality on the receiver inductance l_r, resulting in a non-state variable p_{31}.

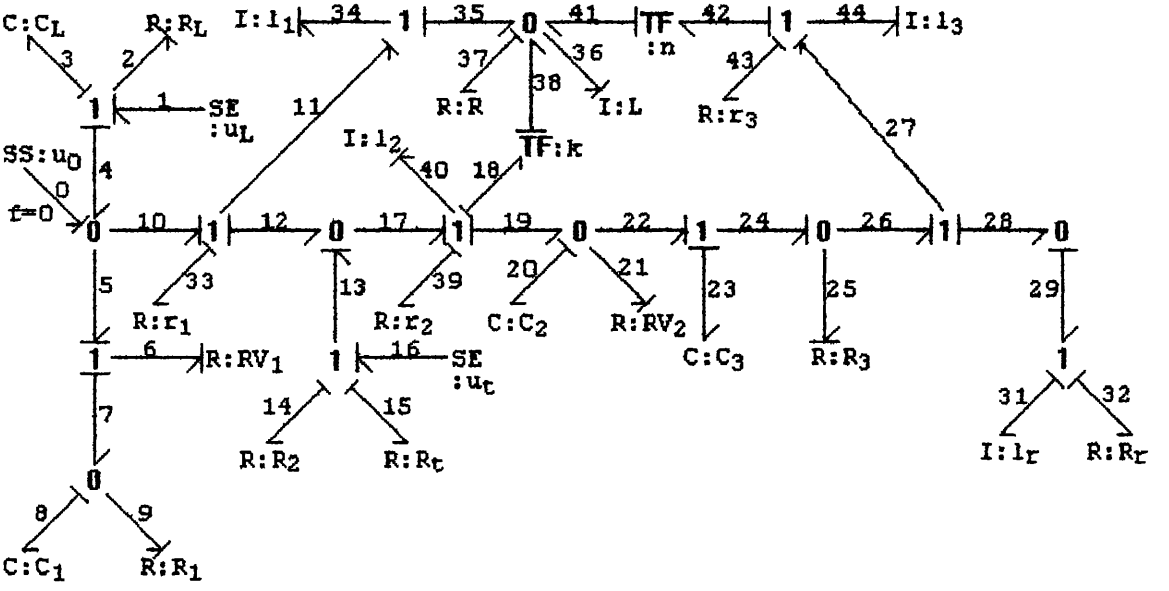


Figure 6.22 Flat bond graph representation of the anti-sidetone circuit

This bond graph is *also under-causal*, so the auxiliary source $SS:u_0$ is added on the input 0-junction, using the method for resolving algebraic loops described in section 4.2.2. The auxiliary source has no effect on the system if the constraint $f_0 = 0$ is applied to the flow equations of the 0-junction:

$$f_0 = f_5 + f_{10} - f_4 = 0 \tag{6.46}$$

which gives

$$0 = -\frac{q_3}{C_L R_L} + \frac{q_8}{C_1 RV_1} - \frac{p_{34}}{l_1} - u_0 \frac{R_L + RV_1}{R_L RV_1} + \frac{u_L}{R_L} \tag{6.47}$$

Since this bond graph includes derivative causality and an algebraic loop, it is appropriate to represent it as a set of DAEs. Solving the bond equations produced during causal propagation gives the following generalised state equations:

$$f_3 = -\frac{q_3}{C_L R_L} - \frac{u_0}{R_L} + \frac{u_L}{R_L}$$

$$f_8 = -\frac{q_8}{C_1}\left(\frac{R_1 + RV_1}{R_1 RV_1}\right) + \frac{u_0}{RV_1}$$

$$e_{34} = -\frac{p_{34}}{I_1}(R_s+r_1+R) + \frac{p_{36}}{L}R - \frac{p_{40}}{I_2}(kR-R_s) - \frac{p_{44}}{I_3}nR + u_0 - u_t$$

$$e36 = \left(\frac{p_{34}}{I_1} - \frac{p_{36}}{L} + \frac{p_{40}}{I_2}k + \frac{p_{44}}{I_3}n\right)R$$

$$e40 = -\frac{p_{34}}{I_1}(kR-R_s) + \frac{p_{36}}{L}kR - \frac{p_{40}}{I_2}(R_s+r_2+k^2R) - \frac{p_{44}}{I_3}knR - \frac{q_{20}}{C_2} + u_t$$

$$e44 = -\frac{p_{34}}{I_1}nR + \frac{p_{36}}{L}nR - \frac{p_{40}}{I_2}(knR) - \frac{p_{44}}{I_3}(R_r+r_3+n^2R) + \frac{q_{20}}{C_2} - \frac{q_{23}}{C_3} - z'$$

$$f20 = \frac{p_{40}}{I_2} - \frac{p_{44}}{I_3} - \frac{q_{20}}{C_2}\left(\frac{R_3+RV_2}{R_3RV_2}\right) + \frac{q_{23}}{C_3R_3}$$

$$f23 = \frac{p_{44}}{I_3} + \frac{q_{20}}{C_2R_3} - \frac{q_{23}}{C_3R_3} \qquad (6.48)$$

where the non-state variable z is given by:

$$z = p31 = I_r f_{31} = p_{44}\frac{I_r}{I_3} \qquad (6.49)$$

z' represents dz/dt

and $R_s = R_2 + R_t$

From these equations, one can build a parameter matrix for the DAE representation, where the descriptor vector (section 4.4) is comprised of the state variables, the non-state variable and its derivative, and the auxiliary input:

$$X = [q_3,\ q_8,\ p_{34},\ p_{36},\ p_{40},\ p_{44},\ q_{20},\ q_{23},\ z,\ z',\ u_0]^T \qquad (6.50)$$

Then

$$E\,dX/dt = AX + Bu$$

$$y = CX + Du \qquad (6.51)$$

where

$E = I_0(9,2)$

$$A = \begin{bmatrix} F & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & G \\ 0 & H & 0 & 0 & 0 & 0 & 0 & 0 & 0 & J \\ 0 & 0 & K & L & M & N & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & P & Q & R & S & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & T & U & V & W & Z & 0 & 0 & 0 & 0 \\ 0 & 0 & a & b & c & d & e & f & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & g & h & j & k & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & l & m & n & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & r & 0 & 0 & -1 & 0 & 0 \\ t & u & v & 0 & 0 & 0 & 0 & 0 & 0 & 0 & w \end{bmatrix}$$

$$B = \begin{bmatrix} \frac{1}{R_L} & 0 \\ 0 & 0 \\ 0 & -1 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{R_L} & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & \frac{R_r}{l_3} & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$D = ()$$

The corresponding transfer function is:

$$G(s) = C(sE - A)^{-1}B + D \qquad (6.52)$$

At this point, it was intended to derive the required transfer functions as a set of symbolic equations using Reduce[64], a symbolic manipulation package, but a bug in this package prevented this. Consequently, since the aim is just to validate the bond graph model against the previous results, a numerical model is now derived. The original analysis[63] used the following measured values as constant parameters:

N1 = 760.5, N2 = 250, N3 = 280 (turns);

giving k = 0.3287 and n = 0.3682

R1 = 175.4, R2 = 22.4, R3 = 65.5, RL = 900 (Ohms)

C1 = 1.48e-7, C2 = 4.67e-7, C3 = 2.28e-6, CL = 2.16e-6 (Farads)

r1 = 29.27, r2 = 12.68, r3 = 13.04 (Ohms)

Some parameter values are frequency and dc (loop) current dependent and a typical set for 20 mA loop current and 1000 Hz is given below:

R = 20000, Rr = 83.5 (Ohms)

L = 250, lr = 17, l1 = 0.94, l2 = 0.1, l3 = 0.13 (mH)

The remaining parameter values are dependent on loop current alone:

RV1 = 9000, RV2 = 884, Rt = 61 (Ohms)

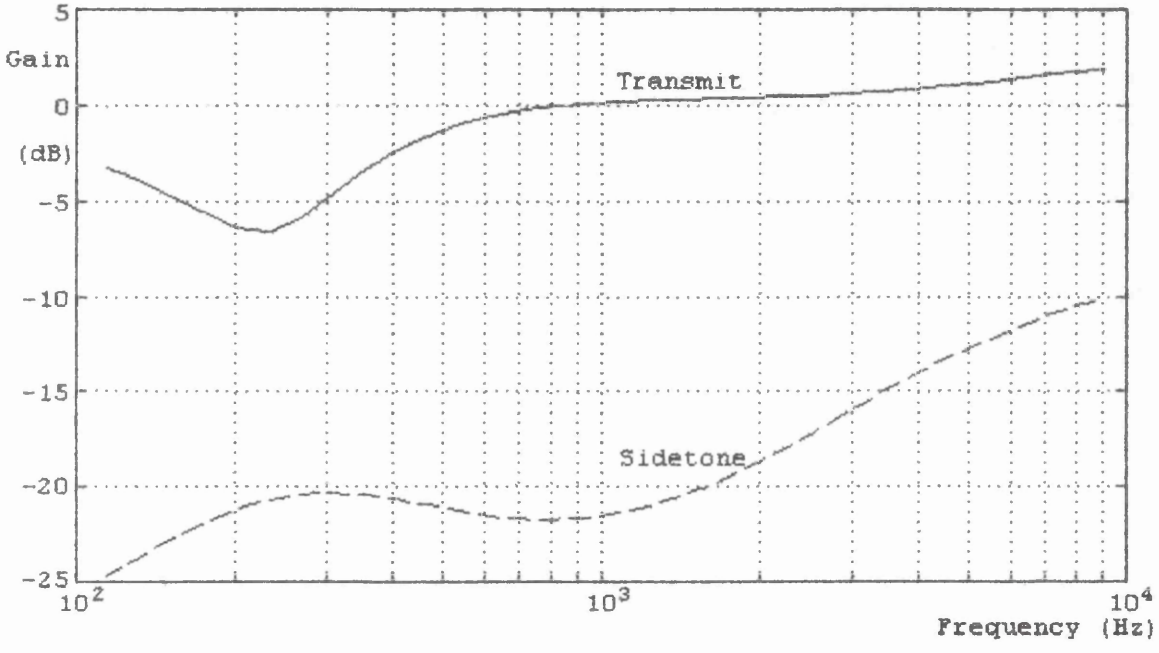


Figure 6.23 Transmit and sidetone frequency response plots

These results approximate those obtained in the obtained in the original analysis, as would be expected due to the frequency dependence of several parameters.

6.5. A high speed carpet cutter

In this case study, a mechanical system used for cutting carpets is modelled, in order to design an appropriate control system. The system is designed to cut carpets to length after they have been woven on an automatic loom. Since the manufacturing process is continuous, the progress of the carpet through the loom does not stop during the cutting process, and hence this operation must be performed at the highest speed possible. The specific requirement is to cut a 5.5 metre width of carpet in less than 0.5 second, with consequent constraints on system design. In particular, the knife blade is a circular design so that it can be rotated to ensure that the cutting friction does not heat the blade to the extent that it will burn the carpet. The speed of the blade across the carpet must also be constant so that the friction heating effect is approximately constant across the width of the carpet. Finally, the whole system must have minimal inertia, so that the high blade speed can be achieved with minimum power expenditure.

6.5.1. Detailed description

The concept used in this design is centred on a circular knife blade which turns through 360° during one stroke across the carpet width, thus minimising local heating on the blade. The blade is carried on a dolly which is pulled across the carpet by a steel reenforced belt, which is in turn driven via a pulley from an electric motor. A similar sub-system is used to rotate the blade. In this case, a second belt drives a worm gear which rotates the blade whenever there is a speed differential between the two belts.

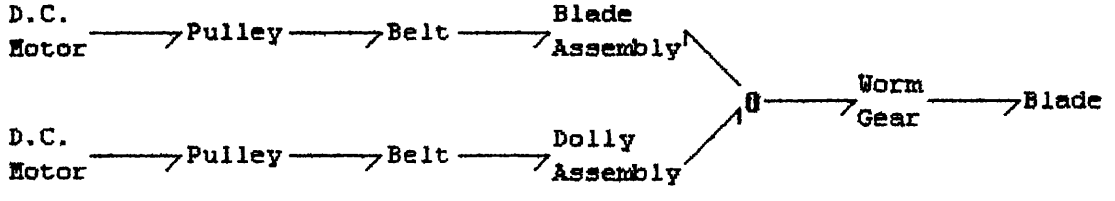


Figure 6.24 Word bond graph of carpet cutter

The basic concept is illustrated well by the word bond graph of figure 6.24, although it is unusual to start with this as the first

view of the system. The 0-junction represents the common effort point at which the difference between the velocities of the blade and dolly assemblies is derived to drive the worm gear.

A more conventional starting point is given in figure 6.25, which shows plan and elevation views, including the motor and pulley systems used to drive the two belts. The entire system is given rigidity by mounting the sub-systems on an I-beam. Each belt is supported between an idler pulley and its drive pulley, and each encloses one half of the top part of the `I' section. The two belts are independently driven by two d.c. motors.

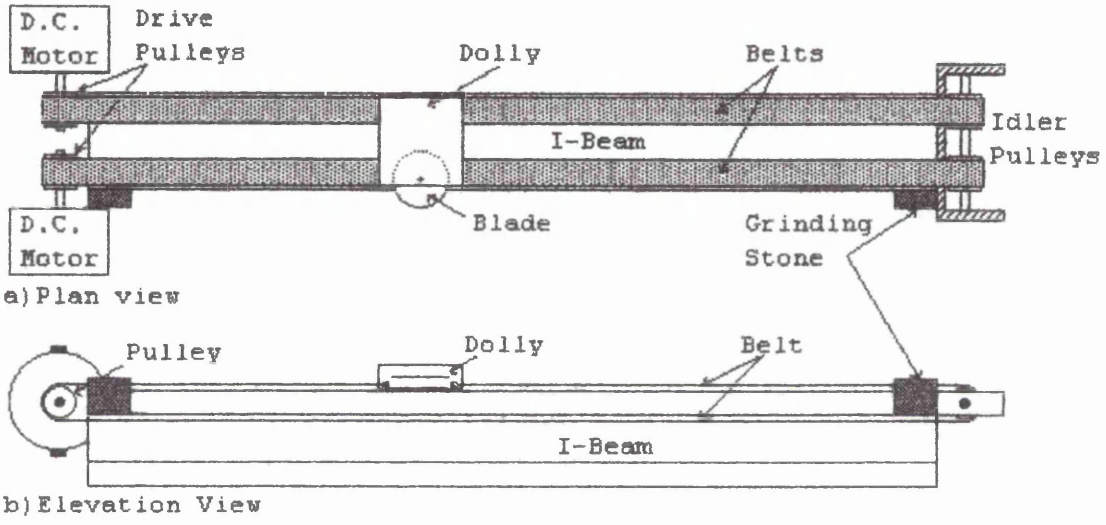


Figure 6.25 Schematic views of carpet cutter

The dolly, which carries the blade, is attached to one of the belts and is supported by a linear bearing on top of the I-beam. The second belt passes through the dolly and links to the blade drive gears in the dolly.

The total length of the stroke is 6.5 metre, with the central 5.5 metre being traversed by the blade and dolly assembly at constant, maximum speed. The 0.5 metre remaining at each end is used for acceleration and deceleration of this assembly, and for sharpening the knife on fixed grinding stones.

From the word bond graph (figure 6.24) it can be seen that the drive sub-systems for the blade assembly and the dolly have the same structure. This structure may be modelled in detail using bond

graph notation as illustrated in figure 6.26 for the dolly sub-system. All parameter labels include the suffix `d' to distinguish from their equivalents in the blade assembly sub-system (suffix `b'). The bond graph of figure 6.26 is shown with integral causality applied to the energy stores. It can be seen that the model is causally complete, and thus the sub-system is physically realisable.

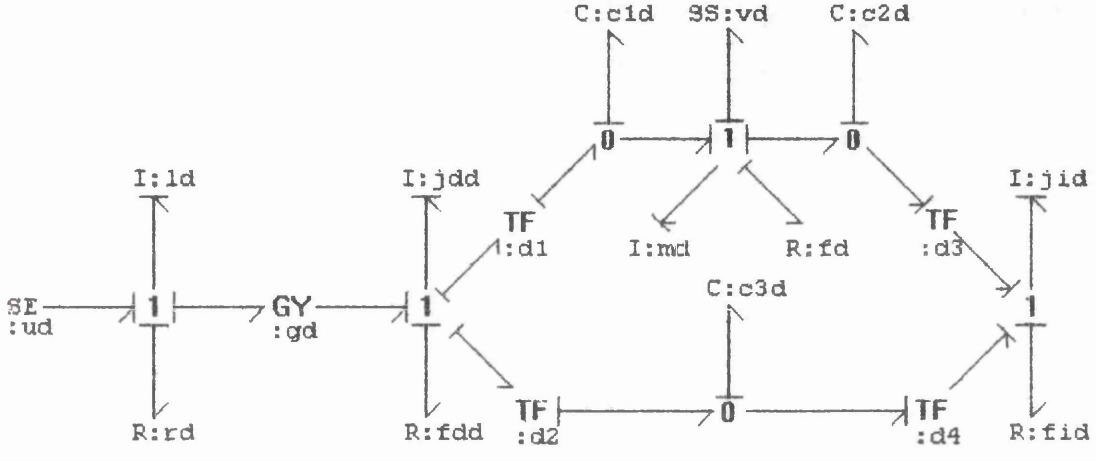


Figure 6.26 Bond graph of dolly drive sub-system

The model has been slightly simplified by combining the inertance jdd and friction fdd on the drive pulley with those of the motor armature, since the drive pulley is mounted directly on the armature shaft. The d.c. motor model is that discussed in chapter 2, section 2.5.3. Rotation of the drive pulley on the armature shaft is transformed to the translational mechanics domain by the two tranformers d1 and d2. The transformation ratios d1 = d2 = dd (the radius of the dolly drive pulley, where angular velocities are given in radians/sec and linear velocities are in m/sec). The capacitances c1d and c2d represent the elasticity of the belt on either side of the dolly, while c3d represents the elasticity of the belt section below the I-beam. Translational forces from the upper and lower belts are transformed back to the rotational forces on the idler pulley by transformers d3 and d4. In practice, the idler pulleys have the same dimensions as the drive pulleys, and thus d3 = d4 = dd. The inertia and friction due to the idler pulley and its bearings are jid and fid respectively.

In practice, the drive belts themselves have mass, which should ideally be modelled as distributed with the belt elasticity, but since the latter is relatively small, the model is simplified by lumping this mass with that of the dolly. The mass of the dolly and the friction between the dolly and the I-beam due the linear bearing are represented by md and fd respectively. Only in this respect does the blade drive sub-system differs from the dolly drive sub-system, since the equivalent inertia and friction are those due to rotating the blade. The forces on the dolly are summed at the common flow 1-junction, from which the velocity vd is monitored using a sensor element (SS). In the same way, a velocity vb is derived from the blade assembly, and the difference between these velocities drives the blade via the worm gear as illustrated in the bond graph fragment shown in figure 6.27.

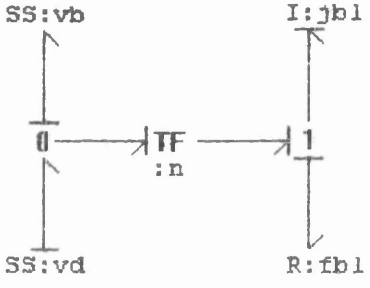


Figure 6.27 Bond graph of blade drive sub-system

In figure 6.27 the reduction ratio of the worm gear is modelled by the transformer n, and the inertia and friction on the blade itself are jbl and fbl respectively. The friction on the blade is a non-linear function of the position of the dolly, but this dependency cannot be modelled by a modulation on the existing bond graph since the position of the dolly is not explicitly modelled. This may be achieved by attaching a spring with unit compliance to the drive pulley via a signal bond. However this remains rather clumsy as the state of capacitance c1d is also required to calculate the exact position of the dolly.

6.5.1.1. Parametric values

Relating this model to actual parametric values (given in Table 6.1) permits the modeller to achieve a better understanding of the constraints and possible simplifications of the model.

Parameter	Dolly sub-system		Blade sub-system	
Max motor current (A)		150		20
Motor gain (A/Nm)	g_d	1	g_b	3.6
Inductance (H)	l_d	$4x10^{-3}$	l_b	$2.4x10^{-3}$
Resistance (Ω)	r_d	0.1	r_b	1.3
Rotor inertia ($kg.m^2$)		$9.4x10^{-3}$		$8.3x10^{-5}$
Pulley inertia ($kg.m^2$)	j_{id}	$57.3x10^{-3}$	j_{ib}	$4x10^{-3}$
Pulley radius (m)	dd	$55x10^{-3}$	db	$23x10^{-3}$
Belt compliance (m/N)	c_{3d}	$0.8x10^{-3}$	c_{3b}	$2.1x10^{-3}$
Belt mass (kg)		2.62		0.53
Dolly mass (kg)		0.5		
Blade inertia ($kg.m^2$)			j_{bl}	$1x10^{-3}$ *

Table 6.1 Parameter values for dolly and blade sub-systems

As might be expected, none of the friction losses in the model are known, but reasonable estimates may be made based on the values shown in the above table. Firstly, we will assume that the belts are not compliant and lump all the inertias and the mass of the dolly together, in order to check the torque required to accelerate the dolly to 11 metre/sec. The equivalent total inertia J_t is, therefore

$$\text{rotor inertia} + 2j_{id} + m_d dd^2 = 0.0094 + 0.114.6 + (2.62+0.5)0.055^2$$

i.e. $J_t = 0.133\ kg.m^2$

The maximum toque is obtained for maximum armature current and is 150 Nm, so the maximum acceleration is

$$\omega' = T/J_t = 150/0.133 = 1128\ rad/sec^2$$

which gives a translational acceleration of 62 m/sec^2. Assuming constant acceleration and a maximum angular velocity of 11/0.055 = 200 rad/sec, the time to reach this velocity is 200/1128 = 0.177 sec, and the distance to reach this from zero velocity is

$$s = 0 + 0.5 \times 62 \times 0.177^2 = 0.97 \text{ metre}$$

Thus the dolly cannot reach full speed by the time it reaches the carpet (after 0.5m), and the friction losses on the pulleys and the motor shaft must be negligible, even to achieve this. The friction due to the carpet can be estimated by simple reasoning about the heat generated by cutting the carpet, and checked from the maximum continuous torque (25Nm) given in the motor specification. It would appear that the highest possible dissipation before burning of the carpet is likely to occur, would be approximately 1000 Watt, giving an estimated value of the friction resistance:

$$f_d = \frac{\text{power dissipated in blade}}{(\text{translational velocity})^2} = \frac{1000}{11^2} = 8.3 \text{ N/(m.sec}^{-1}\text{)}$$

The force required to overcome friction and maintain the velocity of 11 m/sec is 91N, equivalent to a torque of 5 Nm which is one fifth of the maximum continuous torque, so the friction estimate appears reasonable.

Assuming only 10 Watt can be generated in the pulley bearings without excessive heat generation, the rotational frictional resistance is estimated as:

$$f_{id} = \frac{\text{torque lost to bearing friction}}{\text{rotational velocity}} = \frac{0.05}{200} = 2.5\text{x}10^{-4} \text{ Nm/(sec}^{-1}\text{)}$$

Similarly, the blade rotates once every sweep (0.77 seconds), i.e. at 8.17 rad/sec, and assuming a further 10 Watt is generated in the blade due to rotation alone, while cutting the carpet, the rotational frictional resistance is estimated as:

$$f_{bl} = \frac{\text{power dissipated in blade}}{(\text{rotational velocity})^2} = \frac{10}{8.17^2} = 0.15 \text{ Nm/(sec}^{-1}\text{)}$$

These estimates may now be used to calculate the system time constants, in order to analyse the system so that an appropriate control system may be designed.

6.5.2. Causal analysis

Up to this point the bond graph models of the carpet cutter system have been used only to describe the dynamics of the system, with

integral causality preferred for the energy stores, to check for system realisability. This analysis by inspecting the causal augmentation of the model can be used to determine how many independent states the derived model has, and may be extended to an evaluation of possible reduced-order models.

Inspection of the causally complete bond graphs of the dolly and blade assembly sub-systems (figure 6.26), has shown that each sub-system is realisable, and has seven independent state variables. The bond graph fragment shown in figure 6.27 indicates that, with integral causality is imposed on the blade inertia jbl, the blade drive sub-system is driven by the velocity source vd, and a force source from the blade drive belt. In this case, both md and jbl determine the velocity on their respective drive belts.

Since the blade friction, fbl, is modulated by the position of the dolly, it is useful to understand whether changes of this parameter will change the causality of the complete model. The bond graph fragment shown in figure 6.27 indicates that the flow (rotational velocity) of the blade is known, and therefore the constitutional relation has the form

```
e = f*fbl
```

This form is valid for all values of fbl including zero, but excluding infinity. Thus causality of the complete model will only be affected if f_{bl} is infinite; for example if the blade has jammed. A similar argument applies to the translational friction f_d, as the dolly carries the blade across the carpet resulting in a value for fd which varies from approximately zero up to $8.3N/(m.sec^{-1})$ when the blade is in the carpet.

6.5.3. Reduced order model

The first assumption made by the system designers is that the belt has zero compliance. This assumption is worth investigating, and in this case the inverse system analysis technique described in section 4.3.3, is applied to the dolly drive sub-system in

isolation, in order to calculate the motor current f_1 required to achieve a given speed trajectory for the dolly.

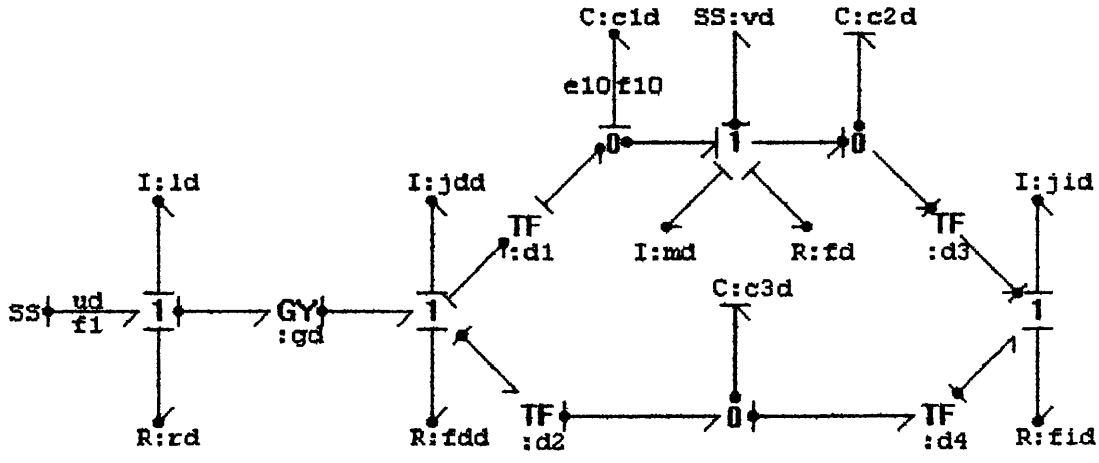


Figure 6.28 Inverse system bond graph for dolly drive sub-system

Propagating effort and flow variables from the SS input v_d, requires only one state variable input definition ($e_{10} = q_{10}/c_{1d}$) for causality to complete resulting in derivative causality on all the remaining energy stores. Solving the algebraic loop which arises for f_{10}, the resulting transfer function is:

$$\frac{f_1}{v_d} = (dd^2(Z_d+Z_i+dd^2Z_m) + sc(Z_dZ_i+dd^2Z_m(Z_d+2Z_i)) + scZ_dscZ_iZ_m$$

$$- (dd^2sc(Z_d+Z_i)+scZ_d(1+scZ_i)) \frac{Z_i(1+scZ_m)+2dd^2Z_d}{(scZ_i+3dd^2)} \frac{q_d}{dd^3} \quad (6.53)$$

where s is the derivative operator, and

$$Z_d = (f_{dd} + sj_{dd})$$

$$Z_i = (f_{id} + sj_{id})$$

$$Z_m = (f_d + sm_d)$$

$$c = c_{1d} = c_{2d} = c_{3d} \quad (6.54)$$

The three compliances for the three sections of belt are assumed to be equal to simplify the transfer function and, for the purposes of estimating relative time constants, c is assumed equal to the largest value. Thus the largest time constant due to the belt compliance is cf_d, arising from the term

$$scZ_m = sc(f_d + sm_d)$$

Comparing these time constants for the parameter values given in section 6.5.1.1:

$cf_d = 0.0066$ seconds

$\frac{j_{dd}}{f_{dd}} = 266$ seconds

$\frac{j_{id}}{f_{id}} = 229$ seconds

$\frac{m_d}{f_d} = 0.376$ seconds

Hence even with the largest likely values of c and fd the time constant due to the belt compliance is over fifty times faster than the next smallest time constant, and it is reasonable to assume that c = 0.

This implies that the constitutive laws for c1d, c2d and c3d *cannot* have the form

$$e = \frac{1}{c} \int f \, dt$$

The alternative form

$$f = c \frac{de}{dt}$$

is expressed in the bond graph shown in figure 6.29, where this causality is propagated from c1d, c2d, c3d and integral causality is propagated from 1d and md. There is an apparent causal conflict at the idler pulley since both transformers d3 and d4 impose a velocity on this pulley. However, since the belt is non-compliant and the transformer ratios are equal, both velocities are equal.

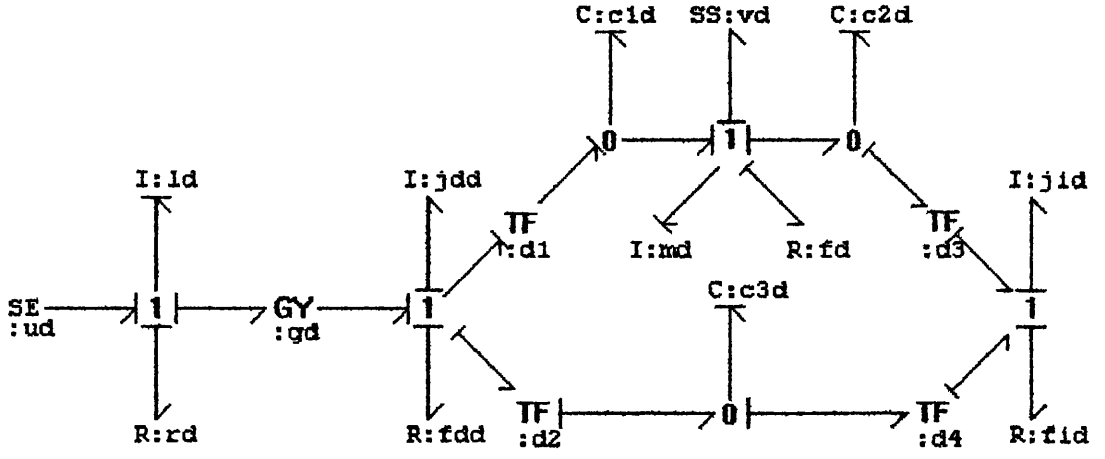


Figure 6.29 Reduced order model assuming non-compliant belt

It can be seen from this bond graph that the reduced order model for each sub-system has only two independent state variables, and two dependent non-state variables - the inertias of the pulleys. For clarity, the drive sub-systems can be simplified to that shown in chapter 2, section 2.5.3, and redrawn in figure 6.30. This simplification reduces the two dependent non-state variables to one equivalent non-state variable.

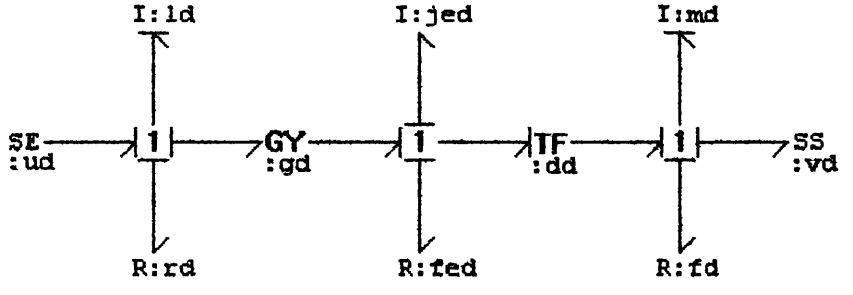


Figure 6.30 Simplified dolly drive bond graph

In the simplified reduced order model, the equivalent inertia j_{ed} replaces the inertias j_{dd} and j_{id} for the drive and idler pulleys. Similarly, the friction force f_{ed} replaces those due to the two friction forces f_{dd} and f_{id}.

Using this approach, the simplified model for the complete system is shown in figure 6.31. The causal augmentation shown is that for the inverse system model, with inputs v_d (dolly velocity) and w (angular velocity of the blade).

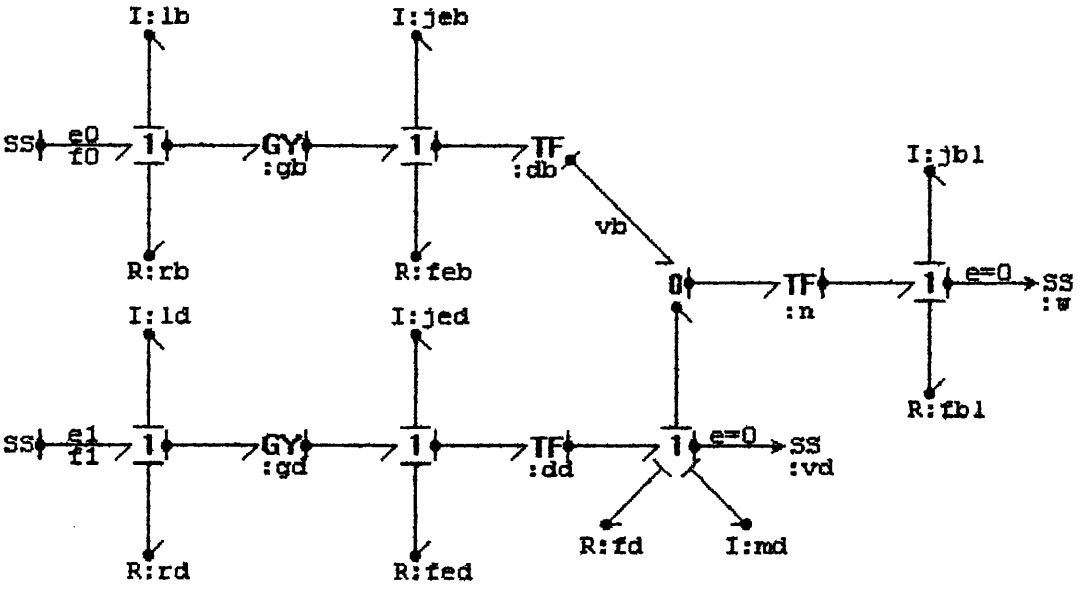


Figure 6.31 Inverse bond graph model of complete carpet cutter

The transfer functions derived from this inverse model are

$$f_1 = \left(\frac{Z_d}{dd} + Z_m dd\right) g_d v_d + Z_{bl} \frac{g_d dd}{n} w \tag{6.54}$$

$$e_1 = \left(Z_{ad}\left(\frac{Z_d}{dd} + Z_m dd\right) g_d + \frac{1}{g_d dd}\right) v_d + Z_{ad} Z_{bl} \frac{g_d dd}{n} w \tag{6.55}$$

$$f_0 = -Z_b \frac{g_b}{db} v_d + \left(\frac{Z_b n}{db} + \frac{Z_{bl} db}{n}\right) g_b w \tag{6.56}$$

$$e_0 = -\left(Z_{ab} Z_b \frac{g_b}{db} + \frac{1}{g_b db}\right) v_d + \left(Z_{ab}\left(\frac{Z_b n}{db} + \frac{Z_{bl} db}{n}\right) g_b + \frac{n}{g_b db}\right) w \tag{6.57}$$

where $Z_d = (f_{ed} + sj_{ed}) = (0.5 + s12.4) \times 10^{-3}$

$Z_b = (f_{eb} + sj_{eb}) = (0.5 + s8.08) \times 10^{-3}$

$Z_m = (f_d + sm_d) = (8.3 + s53.12)$

$Z_{bl} = (f_{bl} + sj_{bl}) = (150 + s) \times 10^{-3}$

$Z_{ad} = (r_d + sl_d) = (100 + s4) \times 10^{-3}$

and $Z_{ab} = (r_b + sl_b) = (1300 + s2.4) \times 10^{-3}$

Using the numerical values provided and estimated in section 6.5.1.1:

$$f_1 = (0.465 + s0.397)v_d + (8.3 + s0.055)x10^{-3}w \quad (6.58)$$

$$e_1 = (18.2 + s0.042 + s^2 1.6x10^{-3})v_d + (0.83 + s0.039 + s^2 0.22x10^{-3})x10^{-3}w \quad (6.59)$$

$$f_0 = -(0.078 + s1.26)v_d + (0.09 + s1.26)w \quad (6.60)$$

$$e_0 = -(12.2 + s1.64 + s^2 3x10^{-3})v_d + (12.2 + s1.64 + s^2 3x10^{-3})w \quad (6.61)$$

The transfer functions e_1/v_d and f_1/v_d (for w=0) are shown in figure 6.32, and the former indicates that voltage driving the motor gives a significant resonance at 16.8Hz. This is confirmed in practice, as the real system exhibits a high frequency vibration when under proportional control of velocity, formerly thought to be due to compliance of the motor drive shaft. By comparison, current driving the motor gives the potential for good control using a simple control scheme, but then additional controller inputs are required to control the shaft velocity. For the case where the blade is not in the carpet, the drive system friction losses become negligible ($f_{ed} = f_d = 0$), but the transfer function for e_1 is virtually unchanged due to the dominance of the s^0 term.

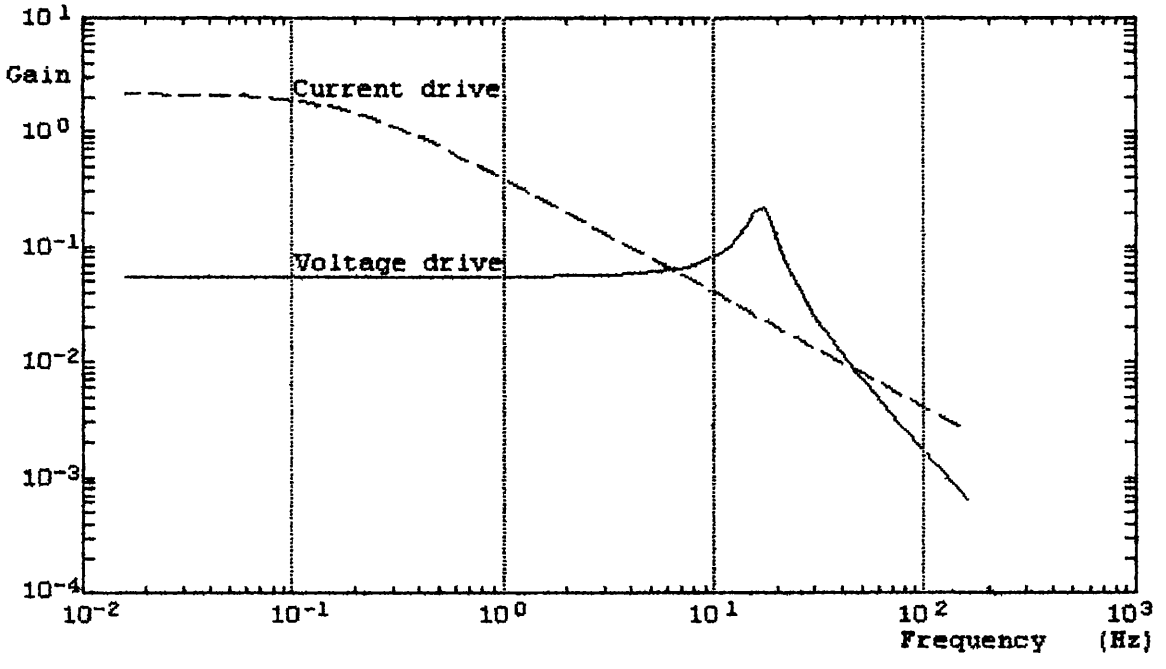


Figure 6.32 Bode plots of dolly drive transfer functions for current and voltage inputs

The polynomials for all the remaining transfer functions have real negative roots resulting in systems which may be easily controlled by simple proportional or PID regulators. The control of the dolly drive sub-system may be improved with the voltage driven motor, by using PI or PID control, since the integral provides higher gain at the lower operating frequencies appropriate to control of the dolly speed (0 to 5.6Hz). The phase margin can be increased by using derivative action in the controller to compensate the rapid change to 180° phase lag in the dolly subsystem after the resonance.

In this case, the controller used includes derivative gain roll-off at Td/4, resulting in the transfer function:

$$G(s) = P\left(1 + \frac{1}{sTi} + \frac{sTd}{(1+sTd/4)}\right) = P\left(\frac{s^2TiTd(1+1/4)+s(Ti+Td/4)+1}{sTi(1+sTd/4)}\right) \quad (6.62)$$

where P is the proportional gain,

Ti is the integral time constant

and Td is the derivative time constant

For simplicity, and in order to avoid interaction between the integral and derivative gains, the integral time constant is set at six times the derivative term. The controller is designed to provide minimum gain at the resonance of the process and maximum phase advance just above the resonant frequency, where the phase lag of the plant tends to 180°. These criteria result in controller parameters:

Td = 5.8 mSec and Ti = 35 mSec

Using these parameter values and a proportional gain of 1, gives a worthwhile improvement in the gain and phase margins, as indicated by comparing the Nyquist plots (figure 6.33) for the open loop gain of the dolly sub-system against that of the system plus controller.

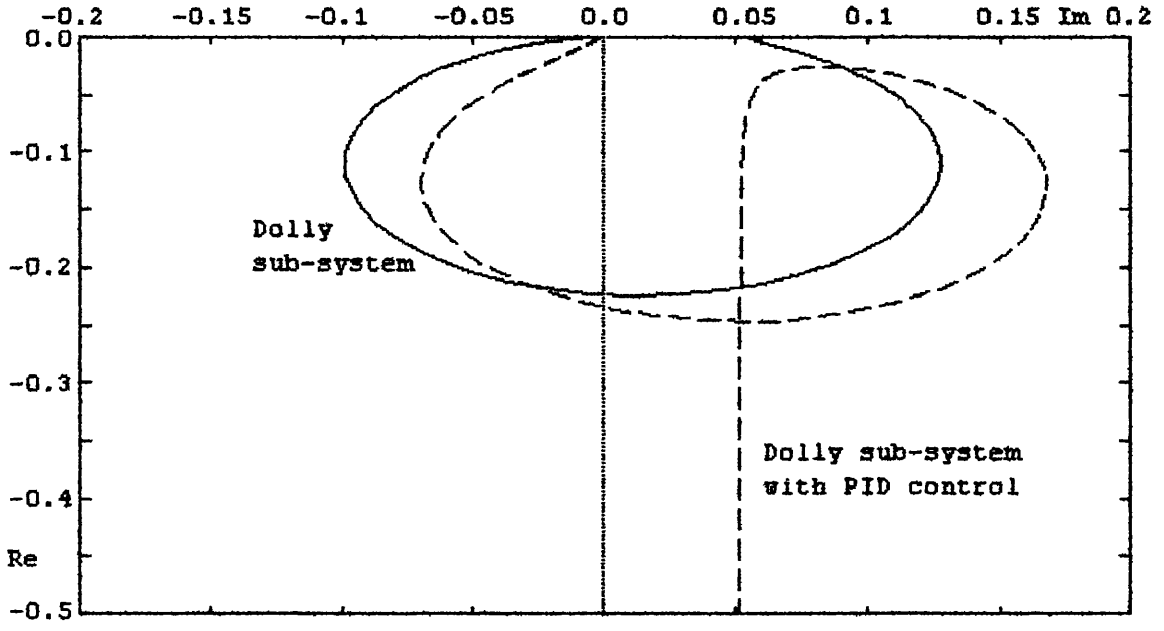


Figure 6.33 Nyquist plots for open loop gain of dolly sub-system with and without PID

6.6. Conclusions

The case studies have indicated how hierarchical bond graph models can be generated and how a variety of mathematical models can be derived using the procedures developed in this thesis. In the case of the extruder model, the method proposed in chapter 4 was successfully used for deriving a steady state model.

Reverse engineering the boiler turbine model from the equations developed by Lindahl, indicated that it was possible to develop pseudo bond graph sub-models of the system, but that these were complicated by some of the original assumptions. It is probable a better model would have resulted from developing a pseudo bond graph as the core model and including the assumptions as specific parameter sets. The exercise did, however, reveal that bond graphs can adequately represent a general system of (potentially non-linear) differential and algebraic equations.

The bond graph model of the telephone hybrid network was unusual in that the model was simultaneously over-causal (containing a dependent state variable) and under-causal resulting in an algebraic equation. The methods described in chapter 4 for deriving

the system equations as a set of DAEs were used, but the resulting DAEs were too complex for symbolic solution using a standard symbolic maths package, and a numerical solution was used to derive the system transfer functions. This highlights a potential problem with the analysis of large systems from flat bond graph models, in that a robust symbolic maths package must be available.

The final case study of the carpet cutter utilised bond graph techniques to simplify the system to a reduced order model. In addition, the new bicausal bond graph notation, described in chapter 5, was used to generate an inverse system model for analysis, resulting in a practical control system design.

In each case, the model generation and analysis was performed entirely systematically, ensuring that the procedures are well suited to encoding as computer algorithms. However, the method of reduction to a flat bond graph representation, before deriving a given mathematical model, can mean that the approach is not as well suited to manual derivation as some techniques appropriate to manipulating mathematical sub-models.

CHAPTER 7 IMPLEMENTATION OF A BOND GRAPH MODELLING TOOL

7.1. Introduction

The previous chapter illustrated the creation of bond graphs for some real physical systems, and the subsequent derivation of a selection of mathematical models for different applications. From these examples, it was evident that manual derivation of a mathematical model from a bond graph of even the smaller systems can be a tedious and error-prone process, despite the systematic procedures involved. This chapter describes the implementation of software tools which permit the modeller to create a bond graph (core model), and use readily available computing power to derive the required mathematical model.

Section 7.2 of this chapter outlines the requirements for the modelling tool, and the discussion of the complete design environment highlights the central role of the database within the environment. Section 7.3 discusses the appropriateness of symbolic, declarative programming, and the advantages of applying object-oriented techniques in the context of a hierarchical approach to modelling. The implementation detail is described in section 7.4, specifically relating to the procedural causality algorithms, and section 7.5 concludes the paper.

7.2. Summary of requirements

The need to define and develop an integrated tool set for the specification, design and implementation of control systems for manufacturing processes, has been expressed both in academia[65] and industry. Initial work[3] on this project identified that a tool to facilitate rapid modelling of industrial processes is the cornerstone of this tool set. A structured model library is essential, requiring full database facilities in order to provide easy access, consistency and version control.

The increasing technical demands of industrial process control applications have focussed equipment manufacturers on the need for

improved design tools to cope with frequent changes to process plants. This need has been recognised in the academic world for some time, as indicated by a survey[66] by Maciejowski, and it was a prime motivation for the ECSTASY research initiative[30]. MacFarlane et al.[65] have indicated some possible future directions for developments in this area.

The design environment is targetted at a wide range of industrial processes such as distillation columns, furnaces, plasticating extruders, robots and physical systems in general. In addition, it is intended to research its applicability to modelling of economic and manufacturing systems. This mix of systems demonstrates both the variety and range of complexity of the systems which may be usefully modelled.

Although many tools exist for either analysis or simulation of such processes, generic tools for modelling hierarchical systems are not widely available, resulting in negligible re-use of models. The motivational example, described in section 1.3.1, highlighted the need for such a tool, and indicated that the core model representation provides a path to achieving this.

7.2.1. Target and research environments

This project emphasised the use of standard hardware and software tools to minimise development effort, and concentrate research in the areas of process modelling, model validation, performance analysis and fault diagnosis[37]. Since, however there were three separate sites for research and development, independent work proceeded in environments best suited to each site, with the intention of merging the research onto the industrial toolset.

The target environment for the industrial tools utilises a standard (80x86-based) work-station, running System V UNIX and a portable graphics tool based on the X-11 interface. In addition, a proprietary relational database is employed, which provides the SQL standard database query language.

In this industrial toolset, tools are interfaced using a combination of three Programmable Logic Control (PLC) application languages, specified by the IEC65 standardisation committee. The first is a function block definition language, incorporating some object-oriented features, while the second (Structured Text) facilitates wiring between function blocks. Lastly, a Sequential Function Chart (a Grafcet derivative) permits sequential and parallel tasks to be defined graphically. These tools constitute the environment within which specific application models for process analysis and simulation may be exercised, before running the real applications via integrated input/output hardware.

The environment at Glasgow University is also UNIX-based, but utilises an object-oriented database for the research work in that area, while the bond graph research was performed using the Prolog language and Reduce mathematical tools.

The remaining bond graph tool research work was based on a PC environment, with algorithms coded using Pascal, due to the availability of these tools. The descriptions in this chapter mainly refer to *this* work.

7.2.2. The database

The choice of architecture described above is based on the premise that the database is central to any design environment concerned with the design of complex systems, as discussed by Maciejowski and Bruer[31]. The core model representation was chosen as a fundamental concept for the modelling tool due to the perceived needs to minimise model development effort and to maintain consistency and integrity across the range of derived models. The core model representation can only provide part of this requirement, and a model database is vital for retaining consistency and tracability of models and data, as a number of model variants are developed, analysed and tested.

It was noted in chapter 1 that many different forms of model can exist; this implementation treats all the different types of model

as differing *views* of the physical system, each relevant to a specific need. The database must be able to keep all the views itemised below in track:

- Technical documentation
- Engineering drawing
- Engineering change notes
- Schematic (iconic) view
- Hierarchical (word bond graph) model
- Core (bond graph) model
- Derived mathematical models
- Parameter values
- Predicted results from derived models
- Experimental results for the physical system.

The descriptive models, such as engineering drawings, change notes and technical documentation, cannot be automatically related to the core model by any means other than a database. Similarly, global parameter values, e.g. gravity, are best held in the database, with the result that it is more consistent to hold all parameter values in the database, and retain responsibility for unit conversion in one place. Derived models and test results could be re-evaluated from the latest core model, whenever needed, but in this case a database provides rapid access, with the required level of integrity.

Whereas the research work is based on an object-oriented (OO) database (Ontos), the industrial environment uses a fast relational database (Sybase). This split was due to the need to minimise the commercial risk for the industrial collaborators, since OO databases are a relatively new and unproven technology. However the OO database has advantages in handling more complex data such as engineering drawings, documentation, process models, and the results of analysis/simulation experiments. OO databases also provide better facilities for handling multiple versions of

models/designs and their associated test data, which occur during a development cycle.

Sybase uses the more conventional relational database technology, with a client-server architecture. This meets the industrial requirement for concurrent access to data from multiple work-stations, whilst enforcing data integrity and security. Sybase is able to store a wide range of data types within the database, although the data is essentially `flattened' to fit the relational technology. This does permit the database to hold SQL code instructions, which give automatic responses from the database to data as it is entered - resulting in a dynamic response to changing process conditions. This, and the provision of an interface to the `C' programming language, facilitates integration with other work-station tools.

7.2.3. The bond graph tool

The previous chapters have shown that bond graphs are well-suited to the role of core model representation, by demonstrating that all the mathematical models required for dynamic systems analysis, design and simulation may be derived from the bond graph. In addition, chapter 3 demonstrated that word bond graphs can be used to represent hierarchically structured systems, as long as causality is not predefined for bonds connecting the sub-models. The limitation on the usefulness of bond graphs has been shown to be that the resulting `flat' bond graph for complex systems, results in a large number of bond equations which must be solved automatically in order to guarantee an error-free mathematical model.

The bond graph tool must `understand' the distinctions between bonds, activated bonds (signals) and modulations, since these distinctions are fundamental to the implementation of the computable causality algorithm. The computable causality algorithms then gives access to the derivations attainable using bicausal bond graphs, which should also be available as a *view* of the model, distinct from the standard causal view.

7.3. Implementation issues

Following the adoption of the bond graph notation for the core model representation, different approaches to implementing the bond graph tool were evaluated at the two research sites. The first approaches focussed on deriving symbolic and numeric models from *textual* representations of the bond graph structure, and these implementations are contrasted in the following section. A subsequent version was designed so that the bond graph could be drawn, and then automatically reduced to either a symbolic or numerical model.

7.3.1. Symbolic, declarative implementation

The bond graph is ideally suited to providing symbolic rather than numerical solutions, since, at its highest level, it is a symbolic description of the system. When the *derived* model is also entirely symbolic, this offers significant advantage in terms of improving understanding the system behaviour. Alternatively, the modeller can assess the effect of a particular parameter on the overall system behaviour, by converting all the remaining parameters to numeric form. The derived symbolic model can readily be converted to either partially or fully numeric models for analysis using numerically-based tools. Since the symbolic solution has such significant advantages, the implementation of the purely numeric tool was rapidly abandoned.

In computing environments, not only is there the distinction between symbolic and numeric approaches, but also that between imperative (procedural) and declarative approaches. This latter distinction is between telling the computer how to find the solution, and declaring what we mean by the problem solution.

The first approach to causal analysis was to use Prolog (a declarative language) to discover *all* valid causal augmentations given the propagation rules for the bond graph junction structure, and then select the appropriate solution for the chosen derived

representation. For example, to obtain a state equation representation, one chooses the solution which maximises the number of states.

For the PC based tool the imperative approach was evaluated, using Pascal as the development language. In this case, the causal augmentation algorithm follows a defined procedure for recursively propagating causality through the junction structure from the pre-defined system inputs. The derived representation is pre-determined by choosing the appropriate `known' inputs to the system, i.e. those parameters which are on the right-hand-side of an assignment statement. This imperative approach was chosen for the implementation of the industrial product due to its greater speed, and ease of integration with the industrial system.

For either approach, the acausal bond graph provides a symbolic declarative core representation from which any specific derived model representation may be obtained. This is achieved by applying specific causal initiation rules, before automatically propagating causality, and applying one or more transformations on the resulting ordered equations. The Prolog implementation is discussed in more detail in[3,37], whereas this thesis continues with a discussion of the implementation of the procedural tool, and the relevance of object-oriented techniques to this implementation.

7.3.2. Object-orientation

An important advantage of hierarchical decomposition is that existing sub-systems may be re-used throughout the system. In this compositional type of hierarchy, re-use implies using identical sub-models in different environments, usually with different parameter values - the sub-model is `A Part Of' the overall model. It is also most important that the modeller be able to work in `bottom-up' manner, by re-using existing sub-models from the library and *specialising* them to achieve a new function. In the resulting hierarchy, the new sub-model inherits the original attributes and functions and is described as `A Kind Of' the original sub-model class. This is analogous to extending the class

hierarchy in an object-oriented design. The contrast between the two hierarchies is equivalent to that between the techniques of functional decomposition and object-oriented (data-centred) analysis for software systems analysis.

Re-use is a major topic in the software engineering field, since so much effort is dissipated in redesigning systems which are modifications of systems which already exist. To date, attempts at modularising software to overcome this problem have not proven overwhelmingly successful, although the object-oriented (OO) paradigm[67,68] shows potential in this area.

Objects in the real world can be used repeatedly without having to be redesigned. Real world objects have, both, attributes (e.g. size and state) and behaviours reflecting the functionality of that object. Three important concepts are key to the re-use of software objects built using the OO methodology - encapsulation, inheritance and polymorphism.

An OO approach to modelling maximises re-use by abstracting all sub-models to a small set of primitive behaviours and structural components, with precise definition of the interfaces through which sub-models interact. This is achieved by data encapsulation, where data which is relevant only to the internal workings of the sub-model is hidden, while that which must be accessible is accessed using methods defined in the abstract sub-model class.

An example of an OO Pascal implementation of an abstract sub-model is the WordBondNode class:

```
WordBondNode = object(BondGraph)
   Identifier : DescriptorStrings;
   OtherViews : FileList;
   Structure  : LinkList;
   constructor  Init(Identifier : DescriptorStrings);
   procedure GetRelations;
   procedure GetIOBonds;
   procedure AugmentGraph;
   procedure BuildStateEquations;
   procedure DeriveTransferFunction;
end;
```

The bond graph structure is saved as a list (type LinkList) of bonds and nodes, similar to the Prolog list description of the bond graph. A node, in this implementation, may then represent either an elementary constitutive law, or another WordBondNode (sub-model). In the case of an elementary node, the constitutive relations of each node are saved with each bond attached to its ports, while for WordBondNodes the attached bond holds a reference to that sub-model.

Defining a sub-model as an object class then permits multiple instantiations of that sub-model within a hierarchical model, each with its independent parameter set. This concept is also supported by using acausal sub-models, as the model can then automatically execute in a manner appropriate to its inputs for any particular instantiation.

Inheritance also encourages re-use, since it permits objects to acquire the attributes and behaviour of other related objects, thus permitting models to be evolved as specialisations of existing sub-models. Multiple inheritance is also a common requirement since it may be appropriate to consider an object being derived from more than one class. Many OO languages do not offer multiple inheritance, partly due to the difficulties in implementing this securely. Object-oriented Pascal was used to code the procedural

implementation, utilising two single inheritance paths from the most basic drawable objects, as illustrated in figure 7.1. This enabled bonds and nodes to be drawn to give a graphical (bond graph) system representation, and also to have specialised properties such as effort and flow variables and causal orientation.

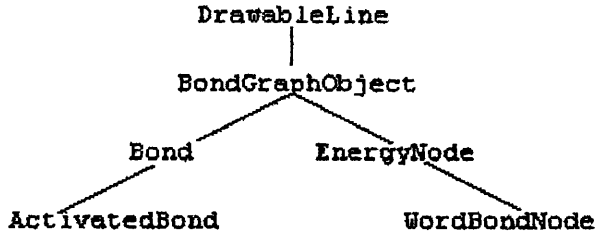


Figure 7.1 Class inheritance for OO bond graph tool

Polymorphism is the ability of different objects within a given class hierarchy to respond appropriately to common method names. For example, each specific object will 'know' how to respond to a message to, say, draw itself on a screen, regardless of whether that object was created before or after compilation. If the object was created at run time, a virtual method is used for this function, overwriting the inherited methods using a technique called late-binding.

Although these attributes facilitate the implementation of a modelling tool, they are not always relevant to modelling of physical processes. For example, good data encapsulation requires that interactions between objects should be minimised, while the trend in industrial processes is toward greater interaction e.g. when waste heat is recuperated into the process which generated it. The model should, in this case, reflect the interactions intrinsic to the physical system. Furthermore, the improved integrity of an encapsulated sub-model, must be balanced against the reduced flexibility in accessing the internal data from the sub-model. However, this can be overcome by the addition of appropriate diagnostic tools to the modelling tool, to inspect information hidden within the model.

The modelling tool is based on the same concepts as an OO drawing tool; it provides a palette of primitive bond graph modelling objects, which can be combined to represent real continuous processes. Figure 7.2 shows a screen dump of the tool, with an acausal bond graph displayed in the workspace window. The second horizontal bar from the top is the palette of bond graph elements, such as sources `SE' and `SF', signals and bonds, 0- and 1-junctions, and nodes representing the remaining constitutive laws.

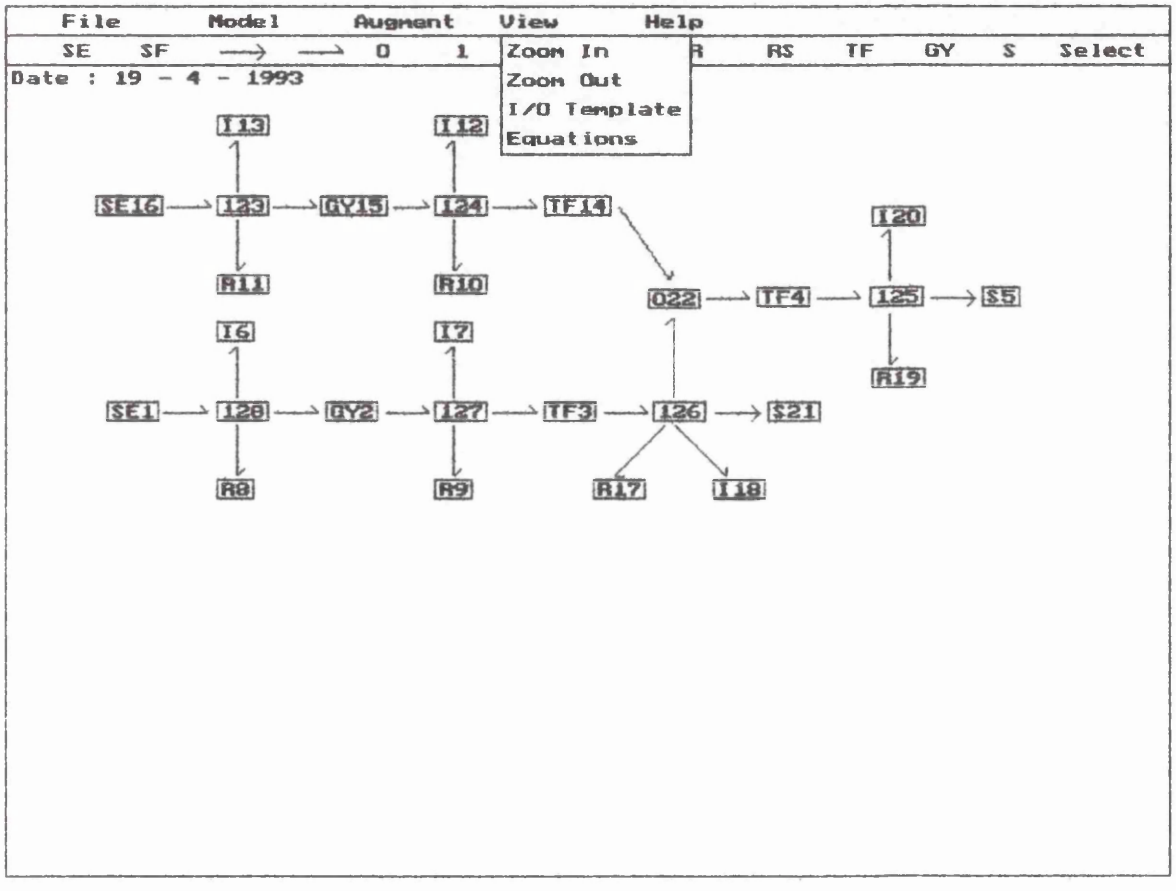


Figure 7.2 Sample bond graph screen for carpet cutter model

Each node on the graph is built from this palette, and is identified by its type followed by a unique identification number, which is used as a reference for identifying which nodes each bond interconnects. This point illustrates where this specific implementation was complicated by failing to fully utilise the benefits of object-oriented design. Making a hierarchical model into a flat bond graph requires that each node has a unique identifier, which may be achieved in one of three ways:

1. Renumber all the nodes and show them on the flat bond graph, as above
2. Keep the original element identifiers but prefix them by a sub-model identifier
3. Reference each node by its pointer held in the LinkList representing the structure

This implementation used the first, and arguably the worst, approach which requires the modeller to manually renumber all the nodes if he wishes to cross reference between model variants - the automatic renumbering algorithm could not do this. The second approach takes up significant screen area. The third scheme is best, since the pointer reference is always unique, although it does lose the advantage of having identifiers immediately visible on the graph. However, in this implementation, each bond or node can be selected and all its data (including name and constitutive relations) can be accessed via a dialogue box.

Within this OO implementation, symbolic descriptions of the constitutive relations of the bond graph elements permit these relations to be non-linear and time-dependent. Providing input `terminals' to modify these parameters, as well as for system inputs, gives good encapsulation, and also permits interfacing to discrete event system models.

These encapsulated sub-models can be saved and re-used either individually, or with other bond graph sub-models to produce hierarchically structured models to an arbitrary level of nesting. The advantage of an OO implementation in this case, is that the class includes references which permits each sub-model instantiation to be viewed in various ways (as a bond graph, an icon, or as textual documentation), and duplicated, specialised or deleted, as needed.

7.4. Specific implementation

The prototype procedural tool presents the modeller with four objects, of which three are visible on entry to the tool. The four objects are:

1. a menu bar
2. a palette of bond graph objects
3. a workspace for creating `flat' bond graphs
4. a model window for viewing word bond graph models or interconnected sub-models

The model window is not visible until the modeller `Zooms Out' from a sub-model created in the bond graph workspace. Figure 7.2 shows the options on the `View' menu, which permit the user to `Zoom In/Out' on sub-models in hierarchically structured models, or to highlight the input/output nodes on sub-models. Finally, an `Equations' view results in a window listing all the bond equations sorted and ordered according to the causal propagation.

Selecting a sub-model in the model window highlights both that sub-model and its components in the bond graph workspace. The extruder screw sub-model has been selected in the extruder bond graph shown in figure 7.3, and as a result nodes 44 (1-junction), 8 (TF) and 20 (RS) are highlighted in the bond graph.

The `File' menu permits the user to `Load' or `Save' models, or to select and load sub-models from specific energy domains. The alternative (partially implemented) sub-menu option is to browse through model categories so that models appropriate to given applications can be easily selected. Finally the user can close the bond graph tool by selecting `Quit' from the File menu.

The `Model' menu offers functions relating to housekeeping of models, permitting the bond graph workspace to be encapsulated as a sub-model or cleared completely. Interconnections can be updated and model documentation can viewed or editted.

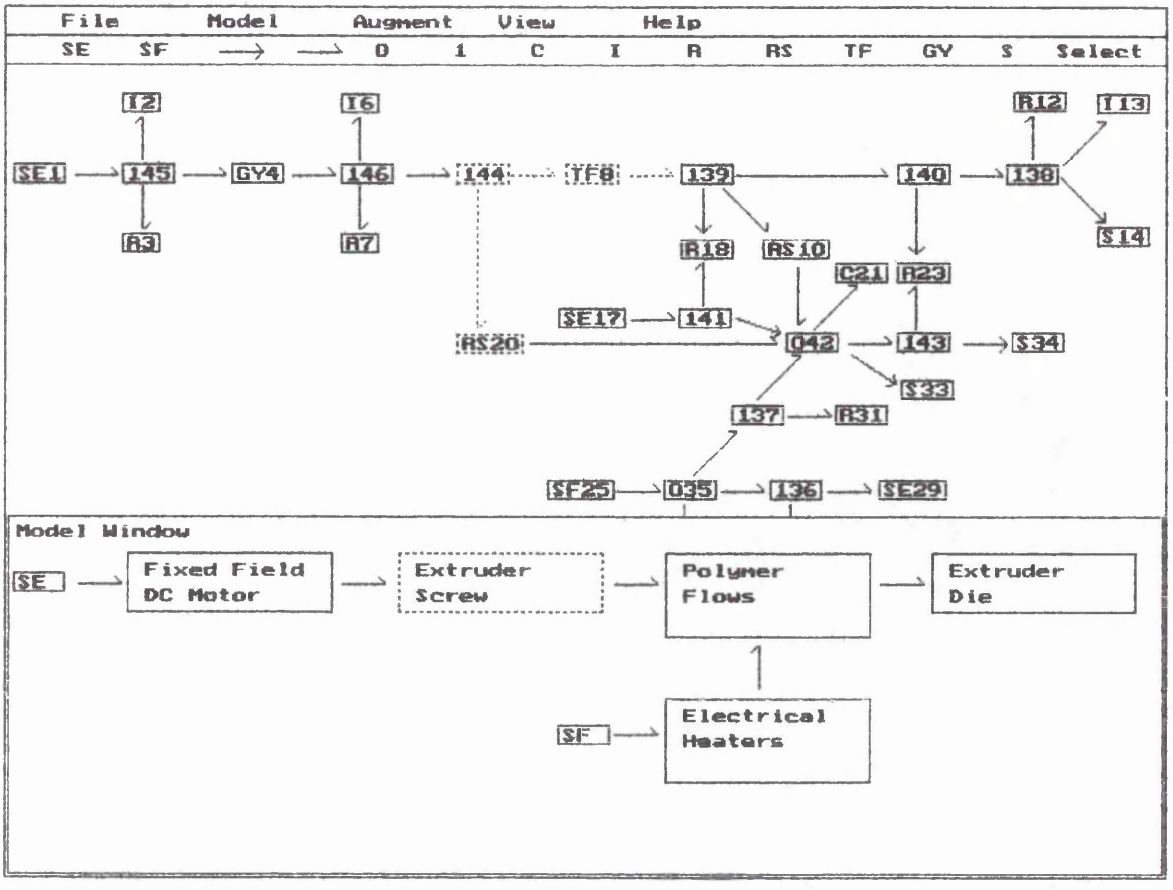


Figure 7.3 Extruder bond graph with hierarchical model view

The `Augment' menu permits automatic renumbering of bond graph nodes, and editting of the node information and constitutive relations. Constitutive relations are accessed by selecting the bond attached to the node which identifies the behaviours of interest, and selecting the `Relations...' option from the `Augment menu. A dialogue box pops up, in the form of a table showing all the information relating to the selected bond. At this point causal behaviour may be manually defined as fixed or preferred (the default is reversible), so that selecting `Causality' results in automatic causal assignment appropriate to the fixed causality inputs.

7.4.1. Algorithms

The causality algorithm for bond graphs automatically re-arranges the symbolic, declarative constitutive relations to the form for the required mathematical model and the specific exogenous inputs to the model. This permits sub-models to be interconnected while

bond graph causality rules take care of sorting the equations to account of interactions between sub-models.

The operation of this algorithm is best explained with the aid of a software structure chart (figure 7.4),which shows the calling hierarchy of the relevant functions and procedures.

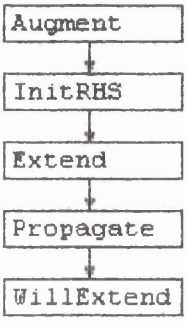


Figure 7.4 Fragment of structure chart for causal augmentation algorithm

The procedure **Augment** has three main functions:

1. Clear all previous causal information from bond graph
2. Run causal augmentation algorithm (calling InitRHS)
3. Order the bond equations using the propagation sequence generated by Propagate

InitRHS initiates causal propagation from known inputs using the rules listed in section 5.3.1, in the chapter on bicausal bond graphs. (This implementation does not show bicausal bond graphs, but the underlying computable causality algorithm is identical.) The order for initiating causal propagation is repeated here, for convenience:

a) Scan the entire bond list and initiate causal augmentation from fixed causality input/outputs. For each fixed causality found, propagate the known effort/flow through the graph using defined causality rules for the structural elements (`0', `1', `TF', `GY'). Causal conflicts are likely to arise, as derivative causality is imposed on energy stores or resulting from conflicting constraints due to incompatible parameter values. Highlight causal conflicts, as the modeller may have to reconsider the constraints.

b) Scan the entire bond list and initiate causal augmentation from the remaining fixed causalities as in (a).

c) Scan the entire bond list and initiate causal propagation from unaugmented nodes with preferred causalities. For each preferred causality found, propagate the known effort/flow through the graph using defined causality rules for the structural elements. Highlight causal conflicts (in this case, non-states), as before.

d) If the graph is causally incomplete at this stage the model is *under-causal*, and causality may be completed either by arbitrary assignment of the effort/flow on one or more bonds, or by employing one of the better defined techniques for under-causal systems. Under-causal systems result in algebraic loops (implicit equations) which must be solved before the full mathematical model can be derived. The effort or flow assigned by the modeller in this procedure, becomes the intermediate variable in the algebraic loop.

By default, the causal augmentation algorithm initiates propagation with energy stores having integral causality, resulting in the bond equations being ordered for automatic generation of state equations. The ordered equations are saved in a text file which can be used as the source for an ACSL simulation model. Alternative mathematical models can be derived by *manually* setting the preferred causalities on individual bonds according to the rules listed in the previous chapters. This process may be easily automated for a subsequent revision of the bond graph software.

For each causal initiation InitRHS calls the procedure **Extend** which indicates whether effort or flow causality is being propagated according to the initiating node and determines the junction onto which this causality is being imposed. Extend passes these parameters to the main propagation procedure.

The causal propagation algorithm, **Propagate**, works *recursively* through the bonds in the structure list until it encounters a bond whose causality has already been assigned, or until there is insufficient causal information at the node of interest for propagation to proceed further. The body of the Propagate procedure has the form:

```
BEGIN {Propagate}
IF (NOT ((Stroke = EFFORT) AND (ThisBond^.EKnown)) AND
    NOT ((Stroke = FLOW) AND (ThisBond^.FKnown)) ) THEN
  BEGIN  {NOT already augmented}
  ThisBond^.SetEquate(RHS, Stroke, EquationNo);
      {Set the RHS of this bond equation equal to the known
variable}
  NextNode:= ThisBond^.GetOtherEnd(ThisNodeID);
      {Find the node at the other end of ThisBond}
  Propagation:= WillExtend(Stroke, ThisBond, NextNode, Eqns);
      {Check if propagation will extend beyond this node and find
the Bond(s)OnNode}
      {WillExtend finds the NoOfBonds attached to NextNode}

  IF Propagation = SUCCESS THEN
    BEGIN  {Propagation will extend further}
    FOR Index := 1 TO NoOfBonds DO
      Propagate(Stroke, BondOnNode[Index], NextNode, RHS, FALSE,
Eqns);
    END  {IF Propagation will extend further}
  END; {IF NOT already augmented}
END; {Propagate}
```

The function **WillExtend** checks through the complete structure list to find which bonds are attached to the NextNode (at the other end of the bond), and interrogates each attached bond for its causal status, in order to judge whether causality can propagate beyond NextNode. For example, if NextNode is a 1-junction and FLOW causality is being propagated, and there is no other bond defining the flow at that junction, then WillExtend returns SUCCESS. A causal conflict, on the other hand, returns FAILURE and the message

`Causal Conflict' is displayed, highlighting the bond where this has occurred. If there is insufficient causal information from all the bonds on NextNode, e.g. when EFFORT causality is imposed on a 1-junction having three or more ports, and no other efforts are known, then WillExtend returns NO_EFFECT.

The procedure Propagate only calls itself recursively if Propagation = SUCCESS, otherwise the existing recursive propagation path terminates, and a new one is begun in the procedure InitRHS. By default, InitRHS implements the Lorenz and Wolper procedure for resolving algebraic loops, but this may be over-ridden by adding an auxiliary source for bond graphs which are known to under-causal.

7.4.2. Results

The example used to demonstrate the use of the causality algorithms is the case study of the telephone anti-sidetone network which was analysed in detail in section 6.4. The bond graph of the three-port hybrid transformer was entered first, and this was made into a sub-model for later use. This sub-model was included with the remaining components to create the bond graph of the complete network, which was renumbered so that the node identifiers correspond to the identifiers of the attached bonds in figure 6.22. The complete model was saved and `Causality' was selected from the `Augment' menu. The procedural causality algorithm is effectively instantaneous, but halts when a causal conflict occurs, as happens when flow causality is propagated from the leakage inductance represented by node I44 to the node I31, representing the inductance of the receiver. At this point the bond attached to I31 is highlighted and the message `Causal Conflict' informs the modeller that this has occurred, as shown on the screen dump illustrated in figure 7.5.

The modeller can then continue the causal augmentation by pressing the return key, whereupon the routine is halted again since the model is also under-causal. Using the Lorenz and Wolper rule, the algorithm selects the bond connecting junctions 1_45 and 0_4 and highlights this to the modeller before completing the causal

augmentation, as shown in figure 7.5. The ordered bond equations may be viewed by selecting `Equations' from the `View' menu, and they are simultaneously saved to an ASCII file.

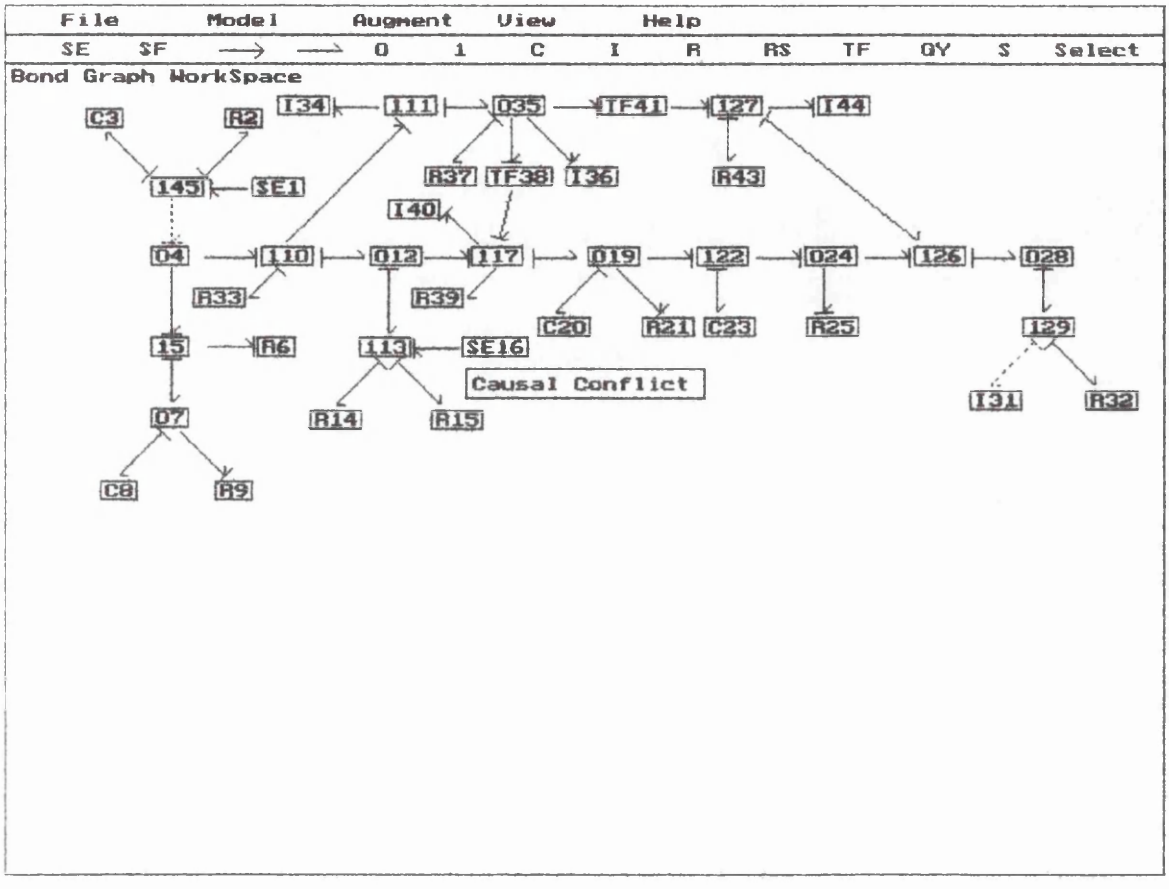


Figure 7.5 Causally augmented bond graph of anti-sidetone network

Once the algebraic loop has been automatically identified, the modeller may prefer to use the method described in chapter 4 to complete causality. In this case, an effort source, u_0, is added to junction 0_4 of the existing model, with its constitutive relation defined as $f_0 = 0$. When the causality algorithm is run again, the causal propagation halts only once, due to the causal conflict on the bond attached to I31.

Selecting `Equations' from the `Augment' menu causes the equation window to pop up, showing the ordered bond equations with constitutive relations sorted according to the imposed causalities. Thus I31 is shown having derivative causality,

i.e. e31 = l_rD.f31, where D. represents the derivative operator,

and the variables:

q3, q8, p34, p36, p40, p44, q20 and q23 are identified as the state variables.

These equations were used as the basis of the work described in section 6.4.

7.5. Conclusions

The previous sections have indicated that the bond graph tool has implemented each of the following requirements for model generation:

- Bond graph core model representation
- Symbolic, declarative representation using parameterised models
- Graphical model creation and editting
- Hierarchical model structure
- Automatic causal augmentation allowing user intervention
- Automatic ordering and assignment statement arranging of symbolic bond equations for simulation models
- Semi-automatic ordering and arrangement of symbolic bond equations for other derived models

The bond graph tool does not, at this time, provide:

- Tight integration with a model database
- Nesting levels greater than two for hierarchical models
- Automatic causal initiation for modulating signals
- Automatic causal initiation for derived models, other than state space models
- Automatic generation of derived models from the ordered bond equations - this can be provided by passing the output to a symbolic manipulation tool

The present version of the tool requires manual intervention to achieve some of the causal initiations, but these are not difficult

to fully automate. Even with these limitations the tool is useful for deriving simple hierarchical models, and as a basis for evaluating the usefulness of object-oriented techniques as an aid to implementation.

The combination of object-oriented techniques with bond graph notation has resulted in a graphical modelling tool where hierarchical models of complex systems can be rapidly developed, and easily extended. The OO paradigm not only has advantages in providing an extensible graphical interface, but inheritance gives a natural mechanism for spawning hierarchically structured models, while encapsulation permits safe rapid prototyping.

Using bond graphs as the underlying core representation permits the modeller to work at a high level of abstraction whilst the required mathematical model may be derived automatically. The causality algorithm for bond graphs automatically re-arranges the declarative constitutive relations to the form for the required mathematical model and the specific exogenous inputs to the model. This permits sub-models to be interconnected while bond graph causality rules take care of sorting the equations to account of interactions between sub-models.

The core model representation permits the derivation of all the required mathematical models so that any changes to the structure or the component behaviours of this model are automatically reflected in the derived models. Consistency between model versions and test results is of prime importance, and should be re-enforced by tight coupling between the models and a relational database. This feature has not been implemented with the present tool, but it does highlight a potentially productive area for further implementation. In particular, modern database tools are available which offer comprehensive drawing tools and an object-oriented database language, giving the potential for implementing the bond graph tool as a specialised view of the database.

CHAPTER 8 CONCLUSION

8.1. Conclusions

This thesis has described research into techniques for modelling continuous physical systems, such that the model development process is separated from the application of the model. This led to the concept of a core model representation which should contain sufficient information for the derivation of any desired mathematical model, by a set of systematic transformations. Since models should provide a way of understanding processes and systems, significant attention has been paid to reducing complexity of large models by decomposing these into a hierarchical structure of sub-models. The thesis has focussed on modelling and model manipulation techniques which permit both these aims to be achieved.

Chapter 2 reviewed the literature on existing modelling techniques, and highlighted the benefits of separating model structure from the behaviours of its component parts. Energy bond graphs have this characteristic, and were shown to provide a consistent notation for modelling systems which include any energy domain, by using energy as the unifying variable. Pseudo bond graphs were also shown to have application in modelling non-physical systems and thermal transport systems. It was demonstrated that causality is a central concept in bond graph theory, which permits the model to be viewed in several ways, according to the exogenous inputs to the system.

Chapter 3 justified the choice of bond graphs as the core model representation, and evaluated various hierarchical modelling approaches proposed in the bond graph literature. The restrictions on causal re-assignment of the multi-port representation and the multi-bond graph notation were highlighted, and shown to limit their usefulness as hierarchical representations. The *acausal* word bond graph was proposed as offering a solution to the hierarchical modelling problem since it can be represented, and causally augmented, in a manner consistent with the underlying energy bond graph representation.

Chapter 4 detailed existing methods for handling under-causal bond graphs which result in algebraic loops, and proposed a new approach to solving this problem. The classic Sequential Causality Assignment Procedure was shown in the literature not to guarantee the minimal number of algebraic loops. This is due to its initial causality assignment to an arbitrary acausal bond, in the event that the model was under-causal. The Lorenz and Wolper algorithm solves this problem, by selecting only interjunction bonds linking a pair of 0- and 1-junctions, and initiating causality from one of these. This algorithm has three disadvantages:

1. The resulting bond graph does not specifically indicate that the model was initially under-causal
2. No indication is given on the bond graph as to which bond was chosen for causal assignment in order to complete causality
3. Causal completion using this method does not guarantee that all the system equations can be automatically obtained, as shown by an example where the effort and flow equations are independent.

The new approach to solving this problem extends the declarative nature of bond graphs, by graphically identifying the intermediate variable chosen to complete causality as an auxiliary input to the model. This retains the advantage of the bond graph model, that the equation structure is clearly defined by the bond graph *before* the system equations are explicitly formulated. In particular, it was demonstrated how this approach permits an entirely systematic derivation of a set of Differential Algebraic Equations from the resulting bond graph. The constraint that the auxiliary input must have no effect on the original system, implies that the covariable on the auxiliary input must be zero. This constraint then defines the algebraic equation associated with each auxiliary input. Using this new approach with conventional bond graphs can result in more than the minimal number of algebraic loops, but the application of bicausal bond graph methods resolves this.

A variation of this algorithm was used to define a new method for deriving the steady state model of a system from the bond graph. The bond graph of the dynamic system is causally augmented for

integral causality, and then each energy storage element is replaced by an auxiliary source having the same causality. The steady state model is thus obtained by manipulating the resulting constraint equations. This method has the advantage that causality propagation used to derive the state equation model is unchanged for this steady state model, whereas existing methods require an entirely new causal augmentation.

Chapter 5 introduced the two new concepts of unilateral bonds and bicausal bond graphs. The classic view of causality, associated with physical realisability of models and ordering of the bond equations, is extended to cover the *computability* of any particular system variable. The unilateral bond extends the causal stroke notation, which effectively defines effort causality, to include a causal dot notation defining flow causality. In conventional bond graphs, this additional notation is unnecessary, since the bond is assumed to imply a bilateral interaction between the two nodes at either end of the bond i.e. if the first node defines the effort then the second must define the effort. In a bicausal bond graph, we assume that both the effort and the flow variable on a bond are computable, and may be inputs to the model, thus requiring the use of the unilateral bond notation. The extensions to the causality propagation rules, required to augment bicausal bond graphs, were laid down in this chapter.

Two specific applications of these concepts were investigated:

1. Inverse system models
2. Constraint propagation in under-causal models, using the auxiliary source method for solution.

For the inverse system model it is assumed that both the effort and flow variable for each model output is known; for example, in the case of an effort sensor output, the effort is assumed to be computable and the flow is constrained to be zero. Both effort and flow causality are propagated from each output, using unilateral bonds as needed. In the resulting bicausal bond graph, it is normal for the interjunction bonds to have unilateral causality while the `external' bonds, attached to nodes outside the junction structure,

have bilateral causality. If causality is not completed after propagating from each of the system outputs, then integral causality is propagated from any energy stores attached to acausal bonds, until the causality on each of the original inputs is defined. The bond equations are ordered according to the causal propagation path with causalities as indicated on the graph, and these equations may be systematically organised to give the DAE for the inverse system. More importantly, the bond graph exactly defines which energy stores result in states and which result in non-states.

The second application of bicausal bond graphs discussed in this chapter complemented the new method for completing causality on under-causal bond graphs, discussed in chapter 4. It was pointed out in chapter 4 that the new method does not necessarily minimise the number of algebraic loops in the system equations. However, the computable causality algorithm may be used to complete causality with the minimum number of algebraic loops, by propagating not only the effort/flow variable imposed by the auxiliary source, but also the flow/effort covariable which comprises the constraint equation. This results in a bicausal bond graph, where the bond attached to the auxiliary source must have unilateral causality, and the remaining interjunction bonds may also be unilateral bonds according to the order that the causal propagation is initiated from this source.

Chapter 6 demonstrated the application of these new methods to the generation of a variety of system models of real physical systems. The first case study followed the procedure defined in chapter 3 for generating a bond graph model from an initial word bond graph which defined the major sub-systems of a plasticating extruder. The bond graph model was then augmented with preferred integral causality on each energy store, such that a state equation model could be generated. Finally the steady state model of the extruder was generated, using the method proposed in chapter 4.

The second case study took an existing model used for simulation of a drum boiler-turbine system, and reverse-engineered a set of bond

graphs from these system equations. The assumptions made in the original model, and the fact that the system involved mass and energy transports resulted in the system being better suited for modelling by pseudo bond graphs rather than energy bond graphs. It was, however, possible to model this system using pseudo bond graphs, although the most significant result of the exercise was to show that pseudo bond graphs can model *any* system of differential and algebraic equations.

A passive electronic network used in the telephone was modelled for the third case study. The system uses a multi-port transformer and it was demonstrated that even with such a `natural' multi-port component the bond graph is most useful in its fully decomposed `flat' form. The model is unusual in that it is both over-causal (having two dependent energy stores) and under-causal (since there are insufficient constraints on another part of the model). An auxiliary source was added to resolve the algebraic loop, using the method proposed in chapter 4, and the resulting equations used to generate a set of DAEs. The intention was to convert these DAEs to a symbolic transfer function model, but the symbolic manipulation tool was inadequate for the task so the work had to be completed numerically. The inability of the maths tool to solve this system of symbolic equations is perhaps an indication of why the potential of bond graphs has so far remained unexploited, as modellers find it easier to analyse complex systems by cascading `non-interacting' sub-systems.

Finally, a bond graph model was developed of a flying-blade carpet cutter, and using bond graph model reduction techniques (chapter 3) a reduced order model was developed and analysed. Since it was desired to find the inputs required for a specified blade trajectory, an inverse system model was developed resulting in a bicausal bond graph. The resulting symbolic transfer functions were analysed numerically, so that a stable control system design was achieved.

Chapter 7 summarised the requirements for a bond graph modelling tool and the resulting implementation. It was shown that a

symbolic, declarative model format is needed to achieve the core model concept, and object-oriented techniques were identified as offering implementation benefits. Results of hierarchical model development for the case studies were demonstrated, highlighting the strengths and weaknesses of the implementation. It was emphasised that the tool implementation does not constitute a full modelling environment, since it is not yet integrated with a model database.

8.2. Further work

This thesis has highlighted two significant areas where further work should produce useful and novel results:

- Integrating the bond graph tool with an object-oriented database
- Applying bicausal bond graphs to constraint propagation and fault detection

It has been pointed out that bond graphs have all the significant characteristics of an object-oriented language, and consequently bond graph models are best represented as objects. These objects can hold all the information to provide whichever view of the model is most appropriate to the user's application. It has been shown that all the prescriptive models can be derived from the core bond graph by appropriate transformations, and similarly each descriptive model should be bound to the core object. An object-oriented database provides the natural mechanism for achieving this end, although the same results could probably be obtained with more difficulty using a relational database.

In the chapter formulating the concepts of bicausal bond graphs, it was shown that, for the applications discussed, unilateral bonds can only occur in the junction structure. It was noted, however, that if more constraints are imposed on the model, then a given external node could have both effort and flow imposed on it, thus defining its constitutive law. In a fault detection algorithm, for example, all inputs and outputs to the system could be fully defined, and similarly, the constitutive laws of all but one of the

components assumed to be defined. The resulting bicausal bond graph would then have a unilateral bond attached to the component under test. The bicausal bond graph becomes a way in which constraint propagation can be both viewed and calculated.

REFERENCES

1 Schmidt J.W., Introduction to Systems Analysis, Modelling & Simulation; *Proc. 1985 Winter Simulation Conference;* Eds D.Gantz,G.Blais,S.Solomon.

2 Paynter H.M. & Shoureshi R., An Introduction to the Dynamics and Control of Thermofluid Processes & Systems, *ASME J.Dynamic Systems, Measurement & Control, Vol 107,* December 1985, pp.230-232

3 Gawthrop P.J. & Smith L.S., An Environment for Specification, Design, Operation, Maintenance, and Revision of Manufacturing Control Systems, in *Proceedings of IEE Conference UKIT 90,* 1990.

4 Breitenecker F. & Solar D., Models, Methods & Experiments - Modern Aspects of Simulation Languages; *Proc. 1985 Winter Simulation Conference;* Eds D.Gantz,G.Blais,S.Solomon.

5 Zeigler B.P., *Theory of Modelling & Simulation;* Wiley & Sons, New York, 1976.

6 Perelson A.S., Bond Graph Sign Conventions; *ASME J.Dynamic Systems, Measurement & Control, Vol* , pp.184-187, June 1975

7 Breedveld, P.C., Fundamentals of Bond Graphs, *IMACS Transactions on Scientific Computing Computing, Vol 3: Modelling and Simulation of Systems,* pp.7-14, 1989

8 MacFarlane, A.G.J., *Engineering Systems Analysis;* G.G.Harrap, 1964

9 Karnopp D., Rosenberg R., *System Dynamics: A Unified Approach;* Wiley & Sons, New York 1975

10 Wellstead P.E., *Introduction to Physical System Modelling;* Academic Press, London, 1979

11 Paynter H.M., *Analysis & Design of Engineering Systems;* MIT Press, Cambridge, Mass., 1961

12 Breedveld P.C., *Physical Systems Theory in Terms of Bond Graphs;* PhD Thesis, University of Twente, Netherlands, 1984

13 Karnopp D., Rosenberg R., *Analysis and Simulation of Multiport Systems;* M.I.T. Press, 1968

14 Lorenz F., Wolper J., Assigning Causality in the Case of Algebraic Loops; *Journal of Franklin Institute, #319,* 1985

15 Karnopp D.C., Alternative Bond Graph Causal Patterns and Equation Formulation for Dynamic Systems, *ASME J.Dynamic Systems, Measurement & Control, Vol 105,* pp.58-63, March 1983

16 Breedveld P.C., A Bond Graph Algorithm to Determine the Equilibrium State of a System; *Journal of Franklin Institute, #318,* 1984

17 Rosenberg R., Karnopp D., *Introduction to Physical System Dynamics;* McGraw-Hill, 1983

18 Breedveld P.C., A Definition of the Multibond Graph Language, in *IMACS Complex and Distributed Systems: Analysis, Simulation and Control;* Elsevier, 1986.

19 Brewer J., Craig P., Hubbard M., Watt K., The Bond Graph Methodology for Technological Forecasting & Resource Policy Analysis; *Energy, Vol 7,6,* 1982

20 Karnopp D.C., Pseudo Bond Graphs for Thermal Energy Transport, *ASME J.Dynamic Systems, Measurement & Control, Vol.100,* pp.165-169, September 1978

21 Rosenberg, R.C., *A Users Guide to ENPORT-4;* Wiley, New York, 1974

22 Granda, J.J., Computer Generation of Physical System Differential Equations Using Bond Graphs; *Journal of Franklin Institute, #319,* 1985

23 Mitchell E.E.L., Gauthier J.S., Advanced Continuous Simulation Language (ACSL), *Simulation, Vol 26, #5,* pp 72-78, 1976

24 Dost, M.H., Syn, W.M., DSL/DP, A High-Level Language for the Simulation of Continuous Systems, *Motion Control Symposium;* Boulder Colorado, Sept 1976

25 Broenink, J.F., *Computer-Aided Physical Systems Modelling and Simulation: a bond-graph approach;* PhD Thesis, University of Twente, Enschede, Netherlands, 1990

26 Felez, J., Vera, C., et al., BONDYN: A Bond Graph Based Simulation Program for Multibody Systems; *ASME J.Dynamic Systems, Measurement & Control, Vol.112,* pp.717-727, December 1990

27 Petzold, L.R., A Description of DASSL: A Differential/Algebraic Equation Solver, SAND82-8637, Sandia National Laboratories, Livermore CA, 1982

28 Nagy, P.A.J., *BondTool$_M$: An interactive interface to the System Indentification Toolbox;* Report LTH-1SY-1-1264, Linkoping University, Sweden, 1991

29 Moler C., *Matlab Users Guide;* The MathWorks Inc., 1987

30 Munro N., ECSTASY - a Control System CAD Environment, in *Proceedings of IEE Conference "Control '88";* Oxford, U.K., pp 76-81, 1988.

31 Maciejowski J.M., Bruer P.T. & Phaal P., Definition & Implementation of a Computer Model for Computer-Aided Control Engineering, in *IFAC World Congress,* Munich FRG, 1987.

32 Thoma J.U., *Simulation by Bond Graphs;* Springer Verlag, Berlin, 1990

33 Birkett S.H., Explicit Multi-port Representation of General Subsystems; submitted to *ASME J.Dynamic Systems, Measurement & Control,* 1992.

34 Gawthrop P.J., Smith, L.S., *Modelling of Dynamic Systems Using Bond Graphs;* Prentice-Hall, to be published 1993

35 Lee F., Moon T.J. & Masada G.Y., Extended Bond Graph Reticulation of Piezoelectric Continua; submitted to *ASME J.Dynamic Systems, Measurement & Control*, 1992.

36 Breedveld P.C., Multibond Graph Elements in Physical Systems Theory; *Journal of Franklin Institute, Vol 319 #1/2,* pp 1-36, 1985.

37 Gawthrop P.J., Marrison N.A., Smith L.S., MIT: A Bond Graph Toolbox, in *Proceedings of 5th IFAC/IMACS Symposium on Computer-Aided Design of Control Systems; CADCS '91,* Swansea, U.K., 1991.

38 Mattsson S.E., Modeling of Interactions between Submodels, *European Simulation Multiconference,* ESM 89, Rome, Italy, June 1989.

39 Karnopp D.C., Power-Conserving Transformations: Physical Interpretations & Applications Using Bond Graphs; *Journal of Franklin Institute, #288,* 1969

40 Ort J.R., Martens H.R., Properties of Bond Graph Junction Structure Matrices; *ASME J.Dynamic Systems, Measurement & Control, Vol* , pp.362-367, December 1973

41 Perelson A.S., Bond Graph Junction Structures; *ASME J.Dynamic Systems, Measurement & Control, Vol* , pp.189-195, June 1975

42 Auslander d., Oster G., Perelson A., Clifford G., On Systems with Coupled Chemical Reaction and Diffusion; *ASME J.Dynamic Systems, Measurement & Control, Vol* , pp.239-248, September 1972

43 Gawthrop P.J., Smith L.S., Causal Augmentation of Bond Graphs with Algebraic Loops; *Journal of Franklin Institute, #329,* 1991

44 Gear C.W., Petzold L.R., ODE Methods for the Solution of Differential-algebraic Systems, *SIAM J. Numerical Analysis,* 21 pp.716-728,1984

45 Campbell S.L., Brenan K.E., Petzold L.R., *Numerical Solution of Initial Value Problems in Differential-algebraic Equations;* North-Holland, NY 1989

46 Luenberger D.G., Dynamic Equations in Descriptor Form, *IEEE Trans. Automatic Control,* AC-22:312, 1977

47 Mattsson S.E., On Modelling and Differential/algebraic Systems, *Simulation,* pp.24-32, January 1989

48 Levy B.C., Verghese G.C., Kailath T, A Generalised State-space or Singular Systems, *IEEE Trans. Automatic Control,* AC-26, pp811-831, 1981

49 Campbell S.L., *Singular Systems of Differential Equations II;* Pitman, London, 1982

50 Cornet A., Lorenz F., Equation Ordering Using Bond Graph Causality Analysis, *IMACS Transactions on Scientific Computing Computing, Vol 3: Modelling and Simulation of Systems;* pp.55-58, 1989

51 Craig J.J., *Introduction to Robotics: Mechanics & Control (2nd Ed.);* Addison-Wesley, 1989

52 Gawthrop P.J., Inverse Systems: Bond Graph, Feedback and Descriptor Representations; *Control Group Research Report R-91/11, Dept. of Mechanical Engineering, University of Glasgow;* June 1991

53 Elmquist, H., *A Structured Model Language for Large Continuous Systems;* Ph.D Dissertation, Report CODEN: LUTFD2/(TRFT-1015), Dept. of Automatic Control, Lund Institute of Technology, LUND, Sweden, 1978

54 Tadmor, Z., Lipshitz, S. & Lavie, R., Dynamic Model of a Plasticating Extruder, *Polymer Engineering & Science, Vol 14;* pp 112-119, February 1974

55 Reber, D., Lynn, R.E. & Freeh, E., A Mathematical Model for Predicting Dynamic Behaviour of a Plasticating Extruder, *Polymer Engineering & Science, Vol 13;* pp 346-356, September 1973

56 Parnaby, J., Kochhar, A., Chow C. & Wood B., Identification of Process Models for Plastics Extrusion Systems - A Preliminary Study, *Proceedings of 3rd IFAC Conference, Delft;* pp 299-306, June 1973

57 Kurihara, F. & Kimura, S., An Experimental Study of the Viscosity of Polymer Melts at High Shear Rate, *Polymer Journal, Vol 17, N0 7;* pp 863-868, 1985

58 Smith L.S., *Modelling the diameter control process;* Technical Report, Eurotherm Ltd, January 1989

59 Eklund, K., *Linear Drum Boiler-Turbine Models;* Ph.D Thesis, Report 7117, Department of Automatic Control, Lund Institute of Technology, 1971

60 Lindahl, S., *A Non-Linear Drum Boiler-Turbine Model;* Report 7620(C), TFRT-3132, Department of Automatic Control, Lund Institute of Technology, 1976

61 Mackenzie, S.A., Gawthrop, P.J., Jones, R.W., Modelling Chemical Processes with Pseudo Bond Graphs, *International Conference on Bond Graph Modelling, ICBGM'93, Simulation Series, Vol25, #5;* pp 327-332, editted Granda, J.J., Cellier, F.E., 1993

62 Stephanian, A., *Development of a Bond Graph Model of an Automotive Gas Turbine;* M.Sc. Thesis, University of Manchester Institute of Science and Technology, 1977

63 Reedyk, C.W., Smith, L.S, *NES 1087-J Miniature Coil and Computational Model for the Anti-sidetone Circuit;* Technical Report TR 7E31-1-77, Bell-Northern Research, 1977

64 White, B.A., *Development, Computer Implementation, and Application of Bond Graph Theory;* Ph.D Thesis, University of Manchester Institute of Science and Technology, 1975

64 Rayna, G., *REDUCE: Software for Algebraic Computation;* Springer-Verlag, 1987

65 MacFarlane A.G.J., Gruebel G. & Ackermann J., Future Design Environments for Control Engineering, *Automatica*, 25(2), 1989.

66 Maciejowski J.M., Data Structures and Software Tools for the Computer Aided Design of Control Systems: A Survey, *IFAC workshop on CACSD*, Beijing PRC, 1989.

67 Meyer B., *Object-Oriented Software Construction;* Prentice-Hall, 1988.

68 Rumbaugh, J., Blaha, M., et al., *Object-oriented Modeling and Design;* Prentice Hall International Editions, 1991