Introduction to Physical Systems Modelling with Bond Graphs

Jan F. Broenink
University of Twente, Dept EE, Control Laboratory
PO Box 217, NL-7500 AE Enschede Netherlands
e-mail: J.F.Broenink@el.utwente.nl

1 Introduction

Bond graphs are a domain-independent graphical description of dynamic behaviour of physical systems. This means that systems from different domains (cf. electrical, mechanical, hydraulic, acoustical, thermodynamic, material) are described in the same way. The basis is that bond graphs are based on energy and energy exchange. Analogies between domains are *more* than just equations being analogous: the used *physical concepts* are analogous.

Bond-graph modelling is a powerful tool for modelling engineering systems, especially when different physical domains are involved. Furthermore, bond-graph submodels can be re-used *elegantly*, because bond-graph models are *non-causal.* The submodels can be seen as *objects*; bond-graph modelling is a form of *object-oriented* physical systems modelling.

Bond graphs are labelled and directed graphs, in which the vertices represent submodels and the edges represent an ideal energy connection between power ports. The *vertices* are idealised descriptions of physical phenomena: they are *concepts*, denoting the relevant (i.e. dominant and interesting) aspects of the dynamic behaviour of the system. It can be bond graphs itself, thus allowing hierarchical models, or it can be a set of equations in the variables of the ports (two at each port). The *edges* are called *bonds*. They denote point-to-point connections between submodel ports. When preparing for simulation, the bonds are embodied as two-signal connections with opposite directions. Furthermore, a bond has a power direction and a computational causality direction. Proper assigning the power direction resolves the sign-placing problem when connecting submodels structures. The internals of the submodels give preferences to the computational direction of the bonds to be connected. The eventually assigned computational causality dictates which port variable will be computed as a result (output) and consequently, the other port variable will be the cause (input). Therefore, it is necessary to rewrite equations if another computational form is specified then is needed. Since bond graphs can be mixed with block-diagram parts, bond-graph submodels can have power ports, signal inputs and signal outputs as their interfacing elements. Furthermore, aspects like the physical domain of a bond (energy flow) can be used to support the modelling process.

The concept of bond graphs was originated by Paynter (1961). The idea was further developed by Karnopp and Rosenberg in their textbooks (1968, 1975, 1983, 1990), such that it could be used in practice (Thoma, 1975; Van Dixhoorn, 1982). By means of the formulation by Breedveld (1984, 1985) of a framework based on thermodynamics, bond-graph model description evolved to a systems theory.

In the next section, we will introduce the bond graph method by some examples, where we start from a given network composed of ideal physical models. Transformation to a bond graph leads to a domain independent model. In section 3, we will introduce the foundations of bond graphs, and present the basic bond graph elements in section 4. We will discuss a systematic method for deriving bond graphs from engineering systems in section 5. How to enhance bond–graph models to generate the model equations and for analysis is presented in section 6, and is called *Causal Analysis*. The equations generation and block diagram expansion of causal bond graphs is treated in sections 7 and 8. Section 9 discusses simulation issues. In section 10 we review this chapter, and include some hints for further reading.

2 Bond graph examples

To introduce bond graphs, we will discuss examples of two different physical domains, namely an RLC circuit (electrical domain) and a damped mass–spring system (mechanical domain, translation). The RLC circuit is given in Figure 1.

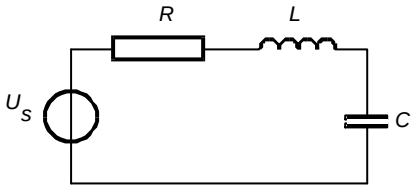

Figure 1: The RLC circuit

In electrical networks, the port variables of the bond graph elements are the *electrical voltage* over the element port and *electrical current* through the element port. Note that a *port* is an interface of an element to other elements; it is the connection point of the bonds. The *power* being exchanged by a port with the rest of the system is the product of voltage and current: $P = ui$. The equations of a resistor, capacitor and inductor are:

$$u_R = iR$$

$$u_C = \frac{1}{C}\int i\,dt$$

$$u_L = L\frac{di}{dt} \quad \text{or} \quad i_L = \frac{1}{L}\int u\,dt$$

In order to facilitate the conversion to bond graphs, we draw the different elements of the electric domain in such a way that their ports become visible (Figure 2). To this port, we connect a *power bond* or *bond* for short. This bond denotes the energy exchange between the elements. A bond is drawn as an edge with half an arrow. The direction of this half arrow denotes the positive direction of the energy flow. In principle, the voltage source delivers power and the other elements absorb power.

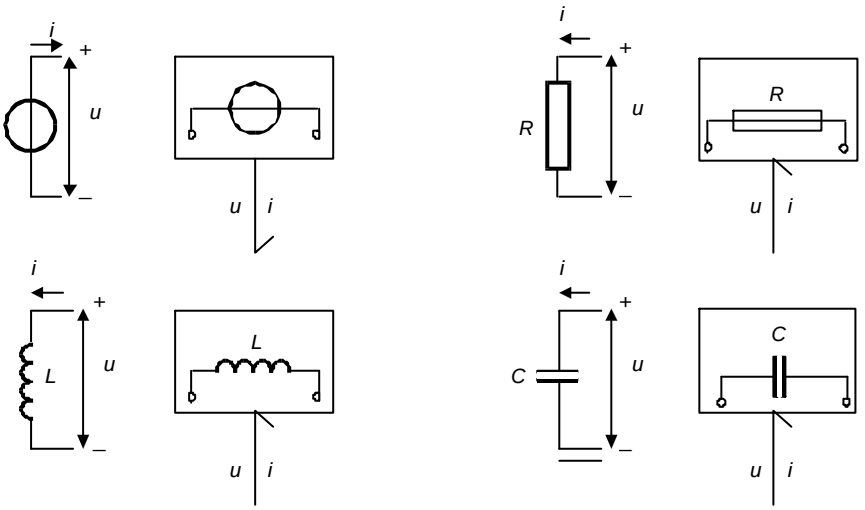

Figure 2: Electric elements with power ports

Considering the circuit of Figure 2, we see that the voltage over the elements are different and through all elements flows the same current. We indicate this current with *i* and connect the bonds of all elements with this current *I* (Figure 3). Changing the electric symbols into corresponding bond graph mnemonics, result in the bond graph of the electrical circuit. The common *i* is changed to a '1', a so-called *1-junction*. Writing the specific variables along the bonds makes the bond graph an electric bond graph. The voltage is mapped onto the domain–independent *effort* variable and the current maps onto the domain–independent *flow* variable (the current always on the side of the arrow). The 1-junction means that the current (flow) through all connected bonds is the same, and that the voltages (efforts) sum to zero, considering the sign. This sign is related to the power direction (i.e. direction of the half arrow) of the bond. This summing equation is the Kirchhoff voltage law.

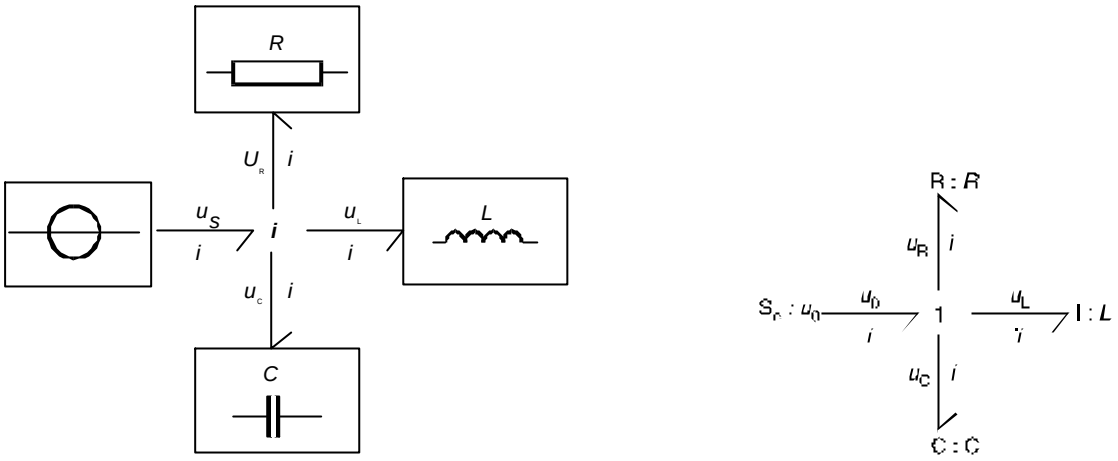

Figure 3: Bond graph with electrical symbols (left) and with standard symbols (right). The standard bond-graph symbols are defined in the section 4

Parallel connections, in which the voltage over all connected elements is the same, are denoted by a *u* in the port–symbol network. The bond–graph mnemonic is a *0*, the so–called *0–junction.* A 0–junction means that the voltage (effort) over all connected bonds is the same, and that the currents (flows) sum to zero, considering the sign. This summing equation is the Kirchhoff current law.

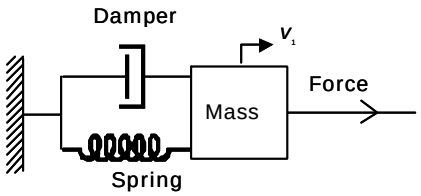

Figure 4: The damped mass spring system

The second example is the damped mass–spring system, a mechanical system shown in Figure 4. In mechanical diagrams, the port variables of the bond graph elements are the *force* on the element port and *velocity* of the element port. For the rotational mechanical domain, the port variables are the *torque* and *angular velocity*. Again, two variables are involved. The *power* being exchanged by a port with the rest of the system is the product of force and velocity: $P = Fv$ ($P = T\omega$ for the rotational case). The equations of a damper, spring and mass are (we use damping coefficient α, spring coefficient K_s, mass m and applied force F_a):

$$F_d = av$$

$$F_s = K_s \int v\,\mathrm{d}t = \frac{1}{C_s}\int v\,\mathrm{d}t$$

$$F_m = m\frac{dv}{dt} \quad \text{or} \quad v = \frac{1}{m}\int F_\mathrm{m}\,\mathrm{d}t$$

$$F_a = force$$

In the same way as with the electrical circuit, we can redraw the element such that their ports become visible (Figure 5). The loose ends of the example all have the same velocity, which is indicated by a *v*. This junction element also implies that the forces sum up to zero, considering the sign (related to the power direction). The force is mapped onto an *effort* and the velocity onto a *flow.* For the *rotational mechanical domain*, the *torque* is mapped onto an effort and the *angular velocity* onto a flow. This implies that force is related to electric voltage and that velocity is related to *electric current.*

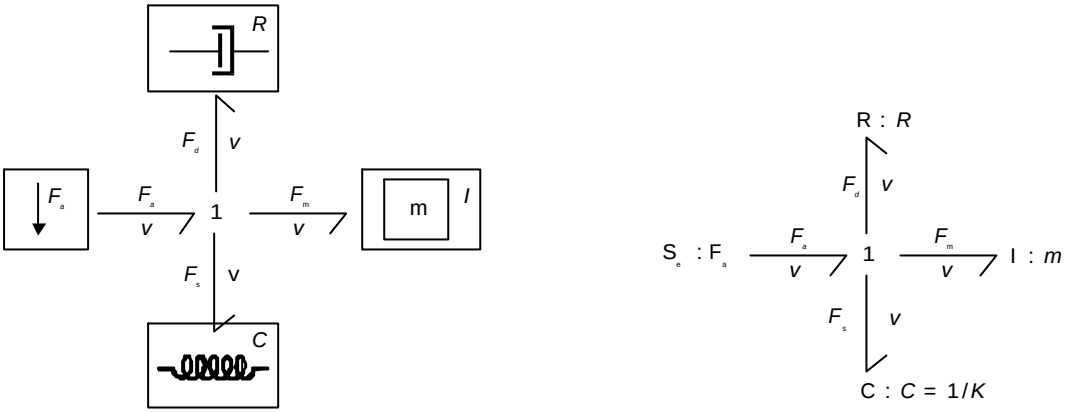

Figure 5: Bond graph with Mechanical symbols (left) and with standard symbols (right)

We see the following analogies between the mechanical and electrical elements:
- The damper is analogous to the resistor.
- The spring is analogous to the capacitor; the mechanical *compliance* corresponds with the electrical capacity.
- The mass is analogous to the inductor.
- The force source is analogous to the voltage source.
- The common velocity is analogous to the loop current.

Besides points with common velocity, also points with common force exist in mechanical systems. Then forces are all equal and velocities sum up to zero, considering the sign (related to the power direction). These common force points are denoted as *0-junctions* in a bond graph (an example is a concatenation of a mass, a spring and a damper: the three elements are connected in 'series'). A further elaboration on analogies can be found in the next section, where the foundations of bond graphs are discussed.

Through these two examples, we have introduced most bond graph symbols and indicated how in two physical domains the elements are transformed into bond graph mnemonics. One group of bond graph elements was not yet introduced: namely the *transducers.* Examples are the electric transformer, an electric motor and toothed wheels. In the next section, we will discuss the foundations of bond graphs.

3 Foundations of bond graphs

Analogies between different systems were shown in the previous section: Different systems can be represented by the same set of differential equations. These analogies have a *physical* foundation: the underlying physical concepts are analogous, and consequently, the resulting differential equations are analogous. The physical concepts are based on energy and energy exchange. Behaviour with respect to energy is domain *in*dependent. It is the same in all engineering disciplines, as can be concluded when comparing the RLC circuit with the damped mass spring system. This leads to identical bond graphs.

3.1 Starting points

Before discussing the specific properties of bond graphs and the elementary physical concepts, we first recall the assumptions general for network like descriptions of physical systems, like electrical networks, mechanical or hydraulic diagrams:
- The conservation law of energy is applicable.
- It is possible to use a *lumped approach*

This implies that it is possible to separate system properties from each other and to denote them distinctly, while the connections between these submodels are ideal. *Separate system properties* mean physical concepts and the *ideal connections* represent the energy flow, i.e. the bonds between the submodels. This *idealness* property of the connections means that in these connections no energy can be generated or dissipated. This is called *power continuity*. This structure of connections is a

conceptual structure, which does *not* necessary have a size. This concept is called *reticulation* (Paynter, 1961) or *tearing* (Kron, 1963).

The system's submodels are *concepts*, idealised descriptions of physical phenomena, which are recognised as the dominating behaviour in *components* (i.e. real–life, tangible system parts). This implies that a model of a concrete part is not necessary only *one* concept, but can consist of a set of interconnected concepts.

3.2 Bonds and Ports

The contact point of a submodel where an ideal connection will be connected to is called a *power port* or *port* for short. The connection between two submodels is called a *power bond* or *bond*; it is drawn as a single line (Figure 6). This bond denotes an ideal energy flow between the two connected submodels. The energy entering the bond on one side immediately leaves the bond at the other side (power continuity).

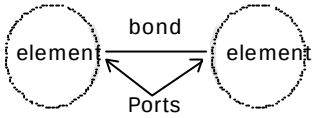

Figure 6: The energy flow between two submodels represented by a bond

The *energy flow* along a bond has the physical dimension of power, being the product of two variables. In each physical domain, there is such a combination of variables, for which a physical interpretation is useful. In electrical networks, the two variables are *voltage* and *current.* In mechanical systems, the variable pairs are *force* and *velocity* for translation and *torque* and *angular velocity* for rotation. In hydraulics, it is *pressure* and *volume flow.* For thermodynamic systems, *temperature* and *entropy flow* are used. These pairs of variables are called (*power–*) *conjugated variables*.

In order to understand the connection as established by a bond, this bond can be interpreted in two different ways, namely:

1. As an interaction of energy.
   The connected subsystems form a load to each other by their energy exchange. A power bond embodies a connection where a physical quantity is exchanged.
2. As a bilateral signal flow.
   The connection is interpreted as two signals, an effort and flow, flowing in opposite direction, thus determining the *computational direction* of the bond variables. With respect to one of the connected submodels, the effort is the input and the flow the output, while for the other submodel input and output are of course established by the flow and effort respectively.

These two ways of conceiving a bond is essential in bond graph modelling. Modelling is started by indicating the physical structure of the system. The bonds are first interpreted as interactions of energy, and then the bonds are endowed with the computational direction, interpreting the bonds as bilateral signal flows. During modelling, it need not be decided yet what the computational direction of the bond variables is. Not that, determining the computational direction during modelling restricts submodel reuse. It is however necessary to derive the mathematical model (set of differential equations) from the graph. The process of determining the computational direction of the bond variables is called *causal analysis*. The result is indicated in the graph by the so-called *causal stroke*, indicating the direction of the effort, and is called the *causality* of the bond (Figure 7).

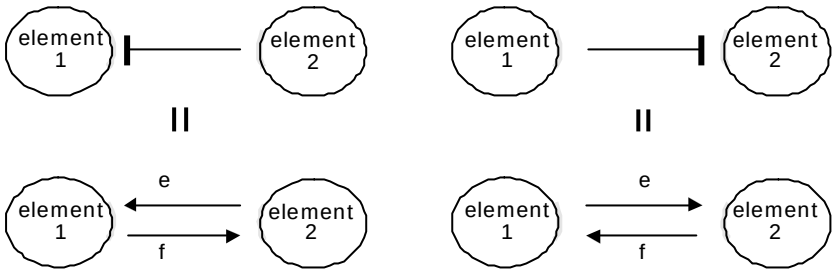

Figure 7: Determine the signal direction of the effort and flow (We do *not* use the power direction at the bonds, so it is not shown here).

In equation form, Figure 7 can be written as:

element1.e := element2.e element2.e := element1.e
element2.f := element1.f element1.f := element2.f

4 Bond graph elements

The constitutive equations of the bond graph elements are introduced via examples from the electrical and mechanical domains. The nature of the constitutive equations lay demands on the causality of the connected bonds. Bond graph elements are drawn as letter combinations (mnemonic codes) indicating the type of element. The bond graph elements are the following:

C storage element for a *q*-type variable, e.g. capacitor (stores charge), spring (stores displacement).
I storage element for a *p*-type variable, e.g. inductor (stores flux linkage), mass (stores momentum).
R resistor dissipating free energy, e.g. electric resistor, mechanical friction.
Se Sf sources, e.g. electric mains (voltage source), gravity (force source), pump (flow source).
TF transformer, e.g. an electric transformer, toothed wheels, lever.
GY gyrator, e.g. electromotor, centrifugal pump.
0 1 0– and 1–junctions, for ideal connecting two or more submodels.

4.1 Storage elements

Storage elements store all kinds of free energy. As indicated above, there are two types of storage elements: C–elements and I–elements. The *q*–type and *p*–type variables are *conserved quantities* and are the result of an accumulation (or integration) process. They are the *state variables* of the system.

In C–elements, like a capacitor or spring, the conserved quantity, q, is stored by accumulating the net flow, f, to the storage element. This results in the differential equation

$$\dot{q} = f$$

which is called a *balance equation*, and forms a part of the constitutive equations of the storage element. In the other part of the constitutive equations, the state variable, q, is related to the effort

$$e = e(q)$$

This relation depends on the specific shape of the particular storage element.

In Figure 8, examples of C–elements are given together with the equivalent block diagram. The equations for a linear capacitor and linear spring are:

$$\dot{q} = i, \qquad u = \frac{1}{C} q$$

$$\dot{x} = v, \qquad F = Kx = \frac{1}{C} x$$

For a capacitor, C [F] is the capacitance and for a spring, K [N/m] is the stiffness and C [m/N] the compliance. For all other domains, a C–element can be defined.

The effort variable is equal when two C–storage elements connected in parallel with a resistor in between are in equilibrium. Therefore, the domain–independent property of an effort is *determination of equilibrium*.

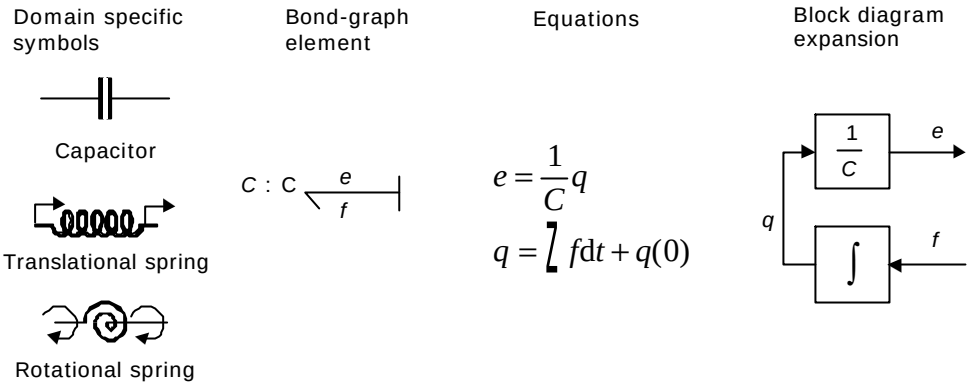

Figure 8: Examples of C elements

In I–elements, like an inductor or mass, the conserved quantity, p, is stored by accumulating the net effort, e, to the storage element. The resulting differential equation is

$$\dot{p} = f$$

which is the *balance equation*. The element–specific part of the constitutive equations is

$$f = f(p)$$

In Figure 9, examples of I–elements are given together with the equivalent block diagram. The equations for a linear inductor and linear mass are:

$$\dot{l} = u, \qquad i = \frac{1}{L} l$$

$$\dot{p} = F, \qquad v = \frac{1}{m} p$$

For an inductor, L [H] is the inductance and for a mass, m [kg] is the mass. For all other domains, an I–element can be defined.

The flow variable is equal when two I–storage elements connected in parallel with a resistor in between, are in equilibrium. Therefore, at I–elements, the domain–independent property of the flow is *determination of equilibrium*. For example, when two bodies, moving freely in space each having a different momentum, are being coupled (collide and stick together), the momentum will divide among the masses such that the velocity of both masses is the same (this is the conservation law of momentum).

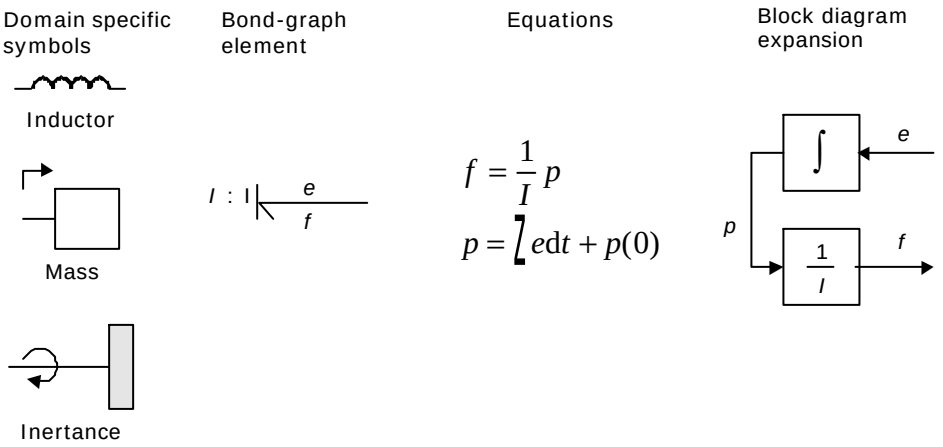

Figure 9: Examples of I elements

Note that when at the two types of storage elements, the role of effort and flow are exchanged: the C–element and the I–element are each other's *dual form.*

The block diagrams in Figure 8 and 9, and also in the next Figures 10 to 16, show the *computational direction* of the signals involved. They are indeed the expansion of the corresponding *causal* bond graph. The equations are given in computational form, consistent with the causal bond graph and the block diagram.

4.2 Resistors

Resistors, R–elements, dissipate free energy. Examples are dampers, frictions and electric resistors (Figure 10). In real-life mechanical components, friction is always present. Energy from an arbitrary domain flows irreversibly to the thermal domain (and heat is produced). This means that the energy flow towards the resistor is *always* positive. The constitutive equation is an algebraic relation between the effort and flow, and lies principally in the first or third quadrant.

$$e = r(f)$$

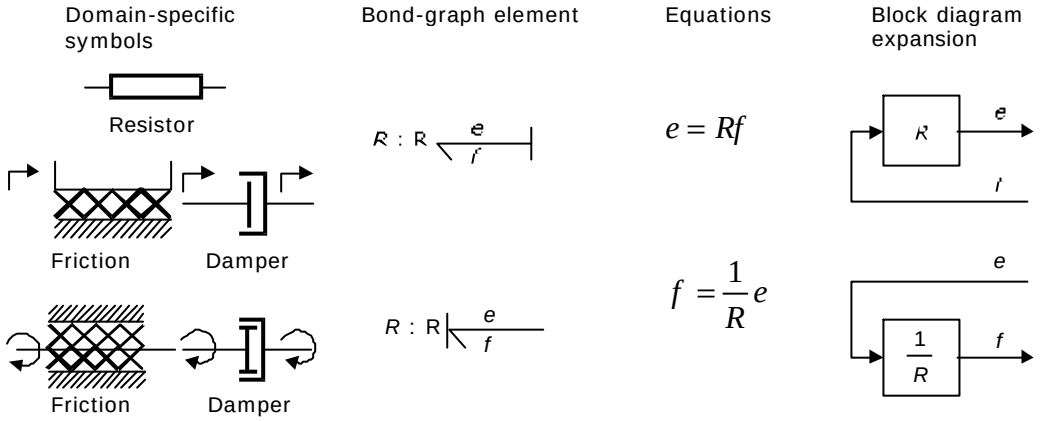

Figure 10: Examples of resistors

An electrical resistor is mostly linear (at constant temperature), namely Ohm's law. The electrical resistance value is in [Ω].

$$u = Ri$$

Mechanical friction mostly is non–linear. The resistance function is a combination of dry friction and viscous friction. Dry friction is a constant friction force and viscous friction is the linear term. Sometimes, also stiktion is involved, a tearing–loose force only applicable when starting a movement. All these forms of friction can be modelled with the R–element. The viscous friction has as formula (R in [Ns/m]:

$$F = Rv$$

If the resistance value can be controlled by an external signal, the resistor is a *modulated resistor*, with mnemonic MR. An example is a hydraulic tap: the position of the tap is controlled from the outside, and it determines the value of the resistance parameter.

If the thermal domain is modelled explicitly, the production of thermal energy should explicitly be indicated. Since the dissipator irreversibly produces thermal energy, the thermal port is drawn as a kind of source of thermal energy. The R becomes an RS.

4.3 Sources

Sources represent the interaction of a system with its environment. Examples are external forces, voltage and current sources, ideal motors, etc. (Figure 11). Depending on the type of the imposed variable, these elements are drawn as Se or Sf.

Besides as a 'real' source, source elements are used to give a variable a fixed value, for example, in case of a point in a mechanical system with a fixed position, a Sf with value 0 is used (fixed position means velocity zero).

When a system part needs to be excited, often a known signal form is needed, which can be modelled by a modulated source driven by some signal form. An example is shown in Figure 12.

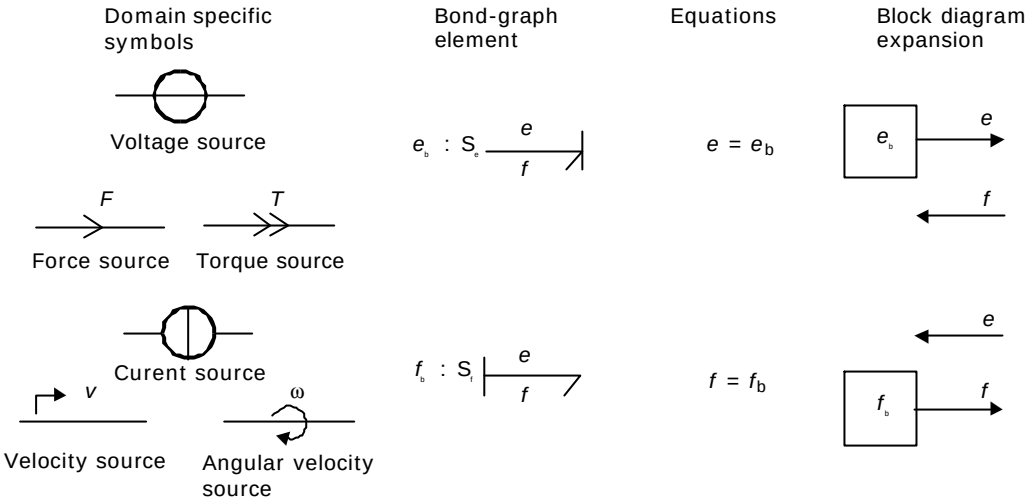

Figure 11: Examples of sources

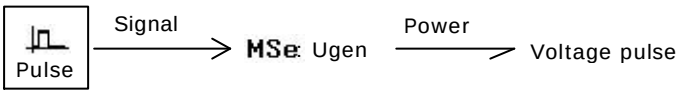

Figure 12 : Example of a modulated voltage source

4.4 Transformers and Gyrators

An ideal *transformer* is represented by TF and is power continuous (i.e. no power is stored or dissipated). The transformation can within the same domain (toothed wheel, lever) or between different domains (electromotor, winch), see Figure 13. The equations are:

$$e_1 = ne_2$$
$$f_2 = nf_1$$

Efforts are transduced to efforts and flows to flows. The parameter *n* is the *transformer ratio.* Due to the power continuity, only *one* dimensionless parameter, *n*, is needed to describe both the effort transduction and the flow transduction. The parameter *n* is *unambiguously* defined as follows: e_1 and f_1 belong to the bond pointing *towards* the TF. This way of defining the transformation ratio is standard in leading publications (Karnopp and Rosenberg, 1990; Thoma, 1989; Breedveld, 1985; Cellier, 1991). If *n* is not constant, the transformer is a *modulated transformer*, a MTF. The transformer ratio now becomes an input signal to the MTF.

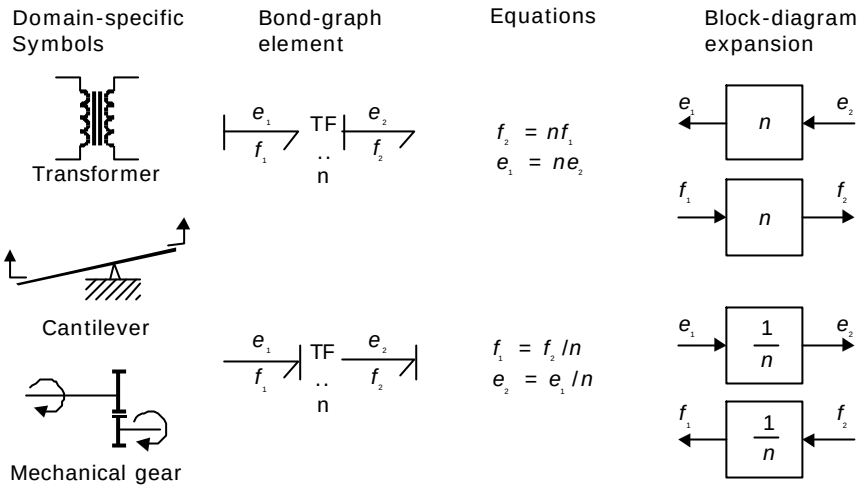

Figure 13: Examples of transformers

An ideal *gyrator* is represented by GY, and is also power continuous (i.e. no power is stored or is dissipated. Examples are an electromotor, a pump and a turbine. Real–life realisations of gyrators are mostly transducers representing a domain–transformation (Figure 14). The equations are:

$$e_1 = rf_2$$
$$e_2 = rf_1$$

The parameter *r* is the *gyrator ratio*, and due to the power continuity, only *one* parameter to describe both equations. No further definition is needed since the equations are symmetric (it does not matter which bond points inwards, only that one bond points towards and the other points form the gyrator). *r* has a physical dimension, since *r* is a relation between effort and flow (it has the same dimension as the parameter of the R element). If *r* is not constant, the gyrator is a *modulated gyrator*, a MGY.

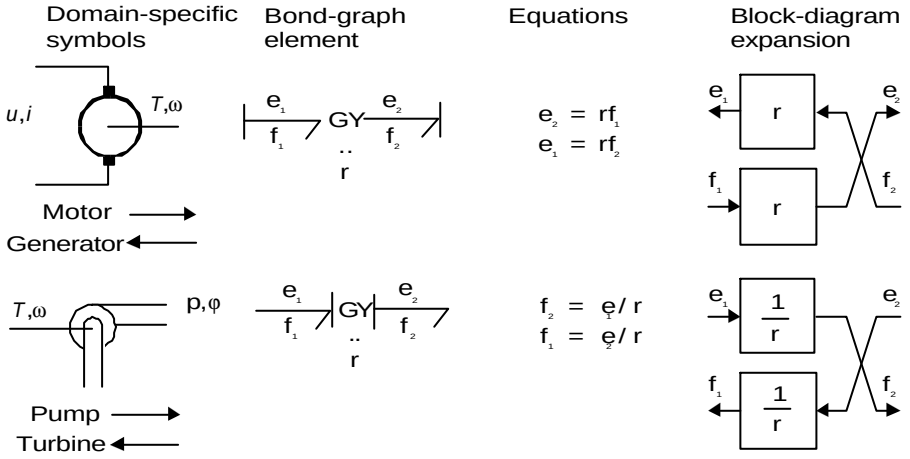

Figure 14: Examples of gyrators

4.5 Junctions

Junctions couple two or more elements in a power continuous way: there is *no* energy storage or dissipation in a junction. Examples are a series connection or a parallel connection in an electrical network, a fixed coupling between parts of a mechanical system. Junctions are *port–symmetric*: the ports can be exchanged in the constitutive equations. Following these properties, it can be proven that there exist only two pairs of junctions: the 1–junction and the 0–junction.

The *0–junction* represents a node at which all efforts of the connecting bonds are equal (Figure 15). An example is a parallel connection in an electrical circuit. Due to the power continuity, the sum of the flows of the connecting bonds is zero, considering the sign. The power direction (i.e. direction of the half arrow) determines the sign of the flows: all inward pointing bonds get a plus and all outward

pointing bonds get a minus. (Figure X). This summation is the Kirchhoff current law in electrical networks: all currents connecting to one node sum to zero, considering their signs: all inward currents are positive and all outward currents are negative.

We can depict the 0–junction as the representation of an effort variable, and often the 0–junction will be interpreted as such. The 0–junction is more than the (generalised) Kirchhoff current law, namely also the equality of the efforts (like electrical voltages being equal at a parallel connection).

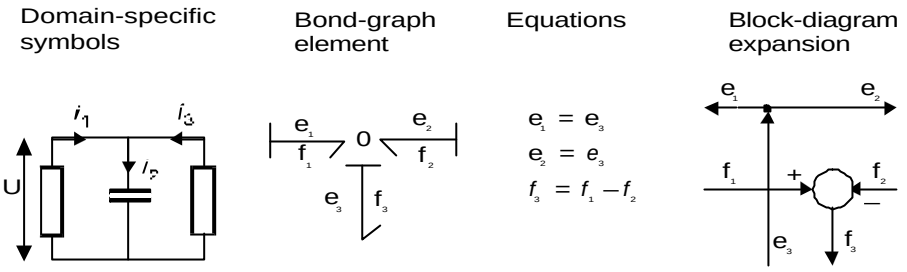

Figure 15: Example of a 0-junction

The *1–junction* (Figure 16) is the dual form of the 0–junction (roles of effort and flow are exchanged). The 1-junction represents a node at which all flows of the connecting bonds are equal. An example is a series connection in an electrical circuit. The efforts sum to zero, as a consequence of the power continuity. Again, the power direction (i.e. direction of the half arrow) determines the sign of the efforts: all inward pointing bonds get a plus and all outward pointing bonds get a minus. This summation is the Kirchhoff voltage law in electrical networks: the sum of all voltage differences along one closed loop (a mesh) is zero. In the mechanical domain, the 1–junction represents a *force balance* (also called the principle of d'Alembert), and is a generalisation of Newton's third law, action = –reaction). Just as with the 0–junction, the 1–junction is more than these summations, namely the equality of the flows. Therefore, we can depict the 1–junction as the representation of a flow variable, and often the 1-junction will be interpreted as such.

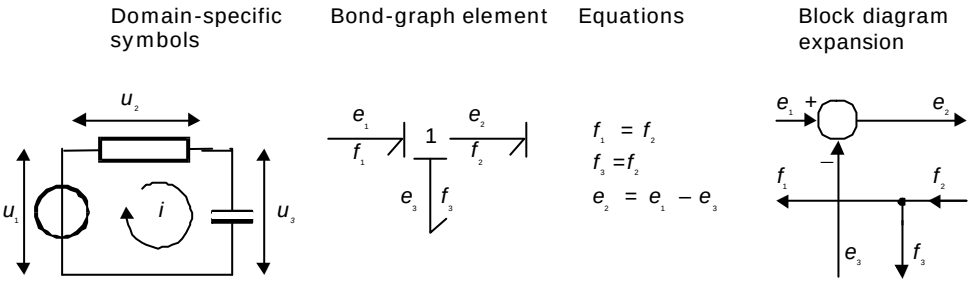

Figure 16: Example of a 1-junction

4.6 Positive orientation

By definition, the power is positive in the direction of the power bond (i.e. direction of the half arrow). A port that has an incoming bond connected to, consumes power if this power is positive (i.e. both effort and flow are either positive or negative, as the product of effort and flow is the power). In other words: the power flows in the direction of the half arrow if it is positive and the other way if it is negative.

R–, C– and I–elements have an incoming bond (half arrow towards the element) as standard, which results in positive parameters when modelling real–life components. For source elements, the standard is outgoing, as sources mostly deliver power to the rest of the system. A real–life source then has a positive parameter. For TF– and GY–elements (transformers and gyrators), the standard is to have one bond incoming and one bond outgoing, to show the 'natural' flow of energy. Furthermore, using the *standard* definition of the parameter at the transformer (incoming bond is connected to port 1 and the ratio n is e_1/e_2) positive parameters will be the result. Note that a gyrator does *not* need such a definition, since its equations are symmetric.

It is possible, however, that negative parameters occur. Namely, at transformers and sources in the mechanical domain when there is a reverse of velocity or the source acts in the negative direction.

Using the definitions discussed in this section, the bond–graph definition is *unambiguous*, implying that in principle there is no need for confusion. Furthermore, this systematic way will help resolving possible sign–placing problems often encountered in modelling.

4.7 Duality and dual domains

As indicated in section 4.1, the two storage elements are each other's dual form. The role of effort and flow in a C–element and I–element are exchanged. Leaving one of the storage elements (and also one of the sources) out of the list of bond graph elements, to make this list as small as possible, can be useful from a mathematical viewpoint, but does not enhance the insight in physics.

Decomposing an I–element into a GY and a C, though, gives more insight. The only storage element now is the C–element. The flow is only a time derivative of a conserved quantity, and the effort determines the equilibrium. This implies that the physical domains are actually pairs of two dual domains: in mechanics, we have *potential* and *kinetic* domains for both rotation and translation), in electrical networks, we have the *electrical* and *magnetic* domains. However, in the thermodynamic domain, no such dual form exists (Breedveld, 1982). This is consistent with the fact that no thermal I–type storage exists (as a consequence of the second law of thermodynamics: in a thermally isolated system, the entropy never decreases).

4.8 Overview

We have discussed the basic bond–graph elements and the bonds, so we can transform a domain–dependent *ideal–physical model*, written in domain–dependent symbols, into a bond graph. For this transformation, there is a systematic procedure, which will be presented in the next section.

5 Systematic procedure to derive a bond–graph model

To generate a bond–graph model starting from an *ideal–physical model*, a *systematic method* exist, which we will present here as a procedure. This procedure consists roughly of the identification of the domains and basic elements, the generation of the connection structure (called the *junction structure*), the placement of the elements, and possibly simplifying the graph. The procedure is different for the mechanical domain compared to the other domains. These differences are indicated between parenthesis. The reason is that elements need to be connected to *difference variables* or *across variables*. The efforts in the non–mechanical domains and the velocities (flows) in the mechanical domains are the across variables we need.

Step 1 and 2 concern the identification of the domains and elements.

1 Determine which physical domains exist in the system and identify all basic elements like C, I, R, Se, Sf, TF and GY.
Give every element a unique name to distinguish them from each other.
2 Indicate in the ideal–physical model per domain a reference effort (reference velocity with positive direction for the mechanical domains).
Note that only the references in the mechanical domains have a direction.

Steps 3 through 6 describe the generation of the connection structure (called the *junction structure*).

3 Identify all *other* efforts (mechanical domains: velocities) and give them unique names.
4 Draw these efforts (mechanical: velocities), and *not* the references, graphically by 0–junctions (mechanical: 1–junctions). Keep if possible, the same layout as the IPM.
5 Identify all effort differences (mechanical: velocity (= flow) differences) needed to connect the ports of all elements enumerated in step 1 to the junction structure.

Give these differences a unique name, preferably showing the difference nature. The difference between e_1 and e_2 can be indicated by e_{12}.

6 Construct the effort differences using a 1–junction (mechanical: flow differences with a 0–junction) according to Figure 17, and draw them as such in the graph.

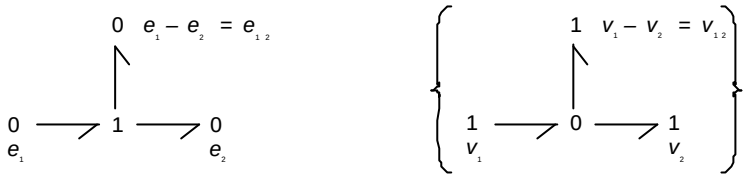

Figure 17: Construction of effort differences (velocity differences)

The junction structure is now ready and the elements can be connected.

7 Connect the port of all elements found at step 1 with the 0–junctions of the corresponding efforts or effort differences (mechanical: 1–junctions of the corresponding flows or flow differences).

8 Simplify the resulting graph by applying the following simplification rules (Figure 18):
- A junction between two bonds can be left out, if the bonds have a ‘through’ power direction (one bond incoming, the other outgoing).
- A bond between two the same junctions can be left out, and the junctions can join into one junction.
- Two separately constructed identical effort or flow differences can join into one effort or flow difference.

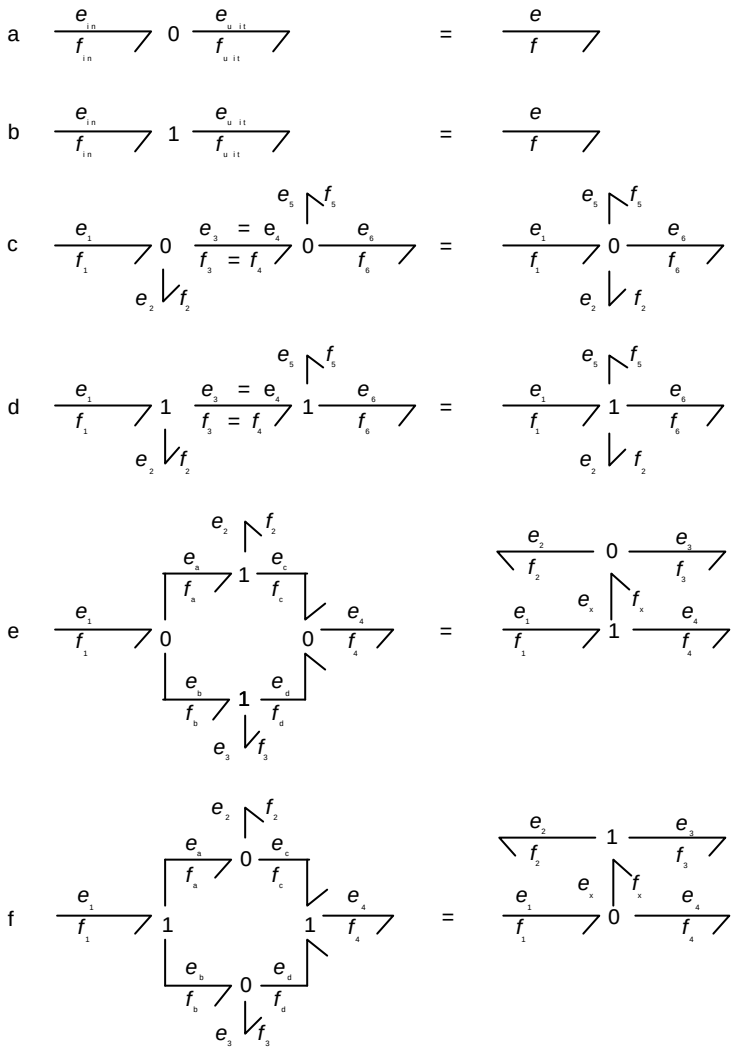

Figure 18: Simplification rules for the junction structure. (a, b) Elimination of a junction between bonds. (c, d) Contraction of two the same junctions. (e, f) Two separately constructed identical differences fuse to one difference.

We will illustrate these steps with a concrete example consisting of an electromotor fed by electric mains, a cable drum and a load (Figure 19).

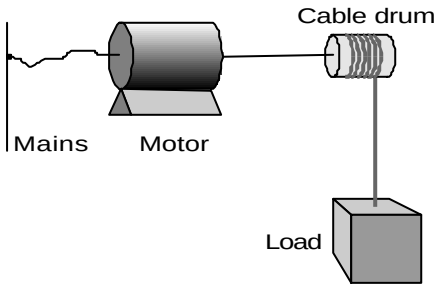

Figure 19: Sketch of the hoisting device.

A possible ideal–physical model (IPM) is given in Figure 20. The mains is modelled as an ideal voltage source. At the electromotor, the inductance, electric resistance of the coils, bearing friction and rotary inertia are taken into account. The cable drum is the transformation from rotation to translation, which we consider as ideal. The load consists of a mass and the gravity force. Starting from the IPM of Figure 20, we will construct a bond graph using the 8 steps mentioned above.

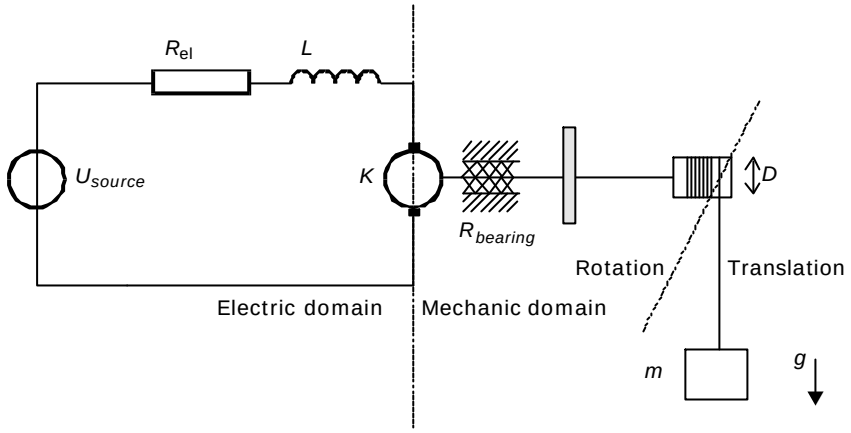

Figure 20: Possible ideal-physical model augmented with the domain information of step 1.

Step 1

This system contains:

- An *electric domain* part with a voltage source (Se), a resistor (R), an inductor (I) and the electric port of the electromotor (GY port).
- A *rotation mechanic domain* part with the rotation port of the electromotor (GY port), bearing friction (R), inertia (I), and the axis of the cable drum (TF port).
- A *translation mechanic domain* part with the cable of the cable drum (TF port), the mass of the load (I) and the gravity force acting on the mass (Se).

In Figure 20, the domains are indicated and all elements have a unique name.

Step 2

The references are indicated in the ideal physical model: the voltage u_0, the rotational velocity ω_0 and the linear velocity v_0. The two velocities also get a positive orientation (i.e. a direction in which the velocity is positive). This result is shown in Figure 21.

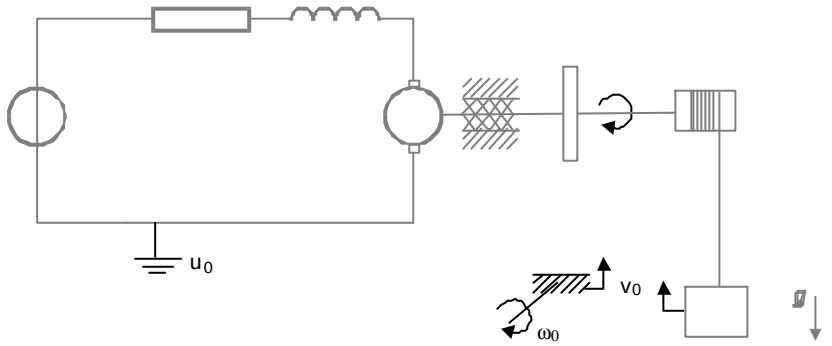

Figure 21: References added to the IPM.

Step3

The other voltages, angular velocities and linear velocities are sought for and are indicated in the IPM (Figure 22). These variables are respectively u_1, u_2, u_3, ω_1, v_1.

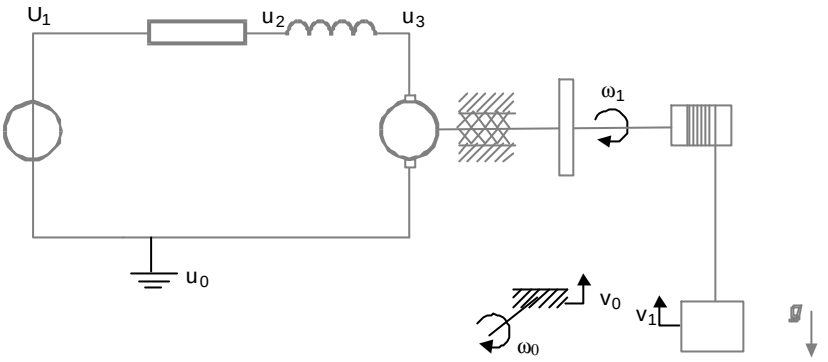

Figure 22: The IPM augmented with relevant voltages, velocities and angular velocities.

Step 4

The variables found in step 3 are depicted with 0– respectively 1–junctions in Figure 23, in a layout compatible to the IPM. The references are not drawn, because they are so to speak eliminated (references have the value 0 and do not contribute to the dynamic behaviour).

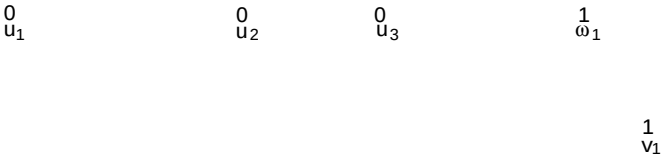

Figure 23: First skeleton of the bond graph: Voltages are shown as 0–junctions and velocities as 1–junctions.

Step 5

When checking all ports of the elements found in step 1 for voltage differences, angular velocity differences and linear velocity differences, only u_{12} and u_{23} are identified. No velocity differences are needed.

Step 6

The difference variables are drawn in the bond graph, see Figure 24. After this step, the junction structure is generated and the elements can be connected.

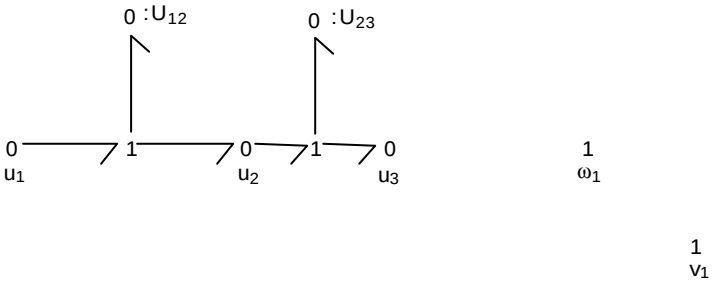

Figure 24: Difference variables (u_{12} and u_{23}) shown in the bond graph.

Step 7

All elements are connected to the appropriate junctions, as shown in Figure 25. Note that non-mechanical domain elements are always connected to 0-junctions (efforts or effort differences) and that mechanical domain elements are always connected to 1–junctions.

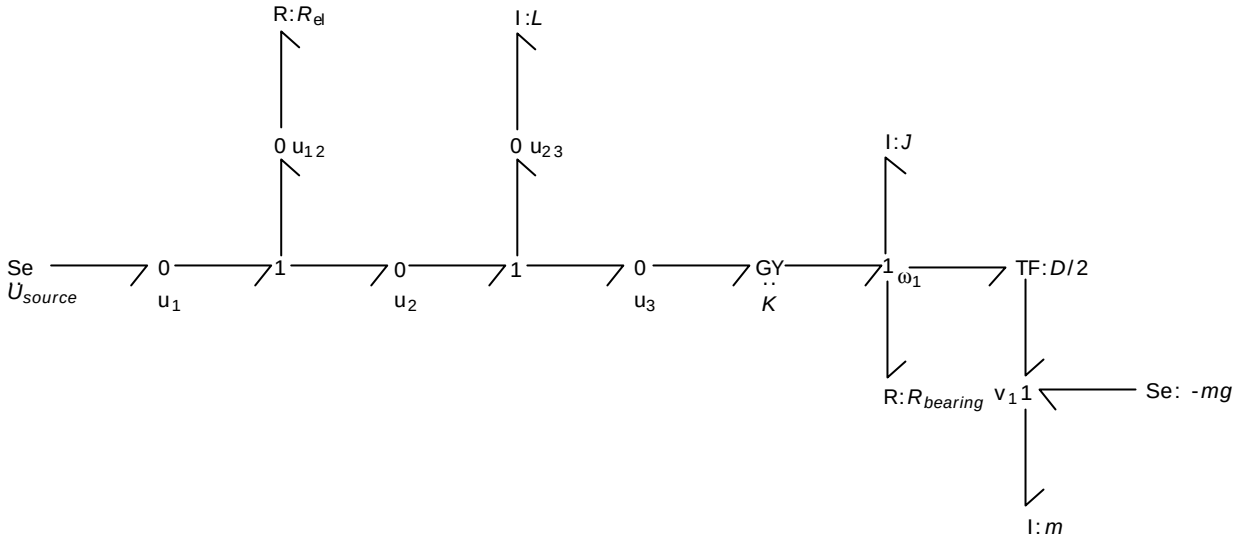

Figure 25: The complete bond graph.

Step 8

As last action, the bond graph needs to be simplified, to eliminate superfluous junctions (according to the rules given in Figure 18). The resulting bond graph is the outcome of the systematic method.

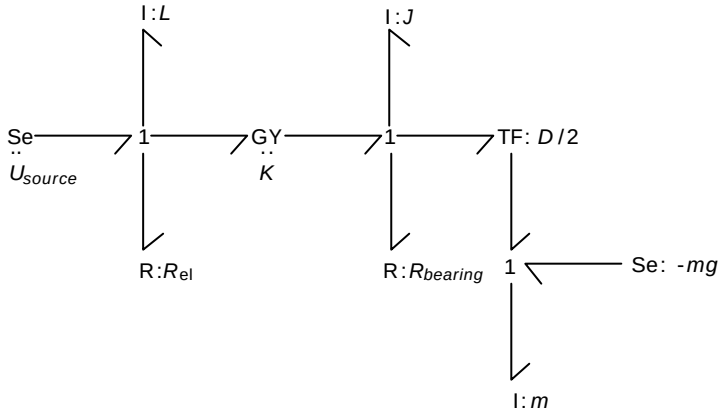

Figure 26: The simplified bond graph, the result of the systematic method.

Obviously, this systematic method is not the *only* method for deriving bond graphs from ideal physical models (IPMs). Another method is the so–called *inspection method*, where parts of the IPM are recognised that can be represented by one junction. An example is a series connection in an electrical network, which is drawn as one 1–junction. This is the case in the example above: The voltage source, inductor, electric resistor and electric port of the motor are directly connected to one 1–junction. Although the inspection method is shorter than the systematic method, it is rather error prone.

6 Causal analysis

Causal analysis is the determination of the signal direction of the bonds. The energetic connection (bond) is now interpreted as a bi-directional signal flow. The result is a *causal bond graph*, which can be seen as a compact block diagram. Causal analysis is in general completely covered by modelling and simulation software packages that support bond graphs like Enport (Rosenberg, 1974), MS1 (Lorenz, 1997), CAMP (Granda, 1985) and 20-SIM (Broenink, 1990, 1995, 1997, 1999b; Broenink and Kleijn, 1999). Therefore, in practice, causal analysis need not be done by hand. Besides derivation of equations, causal analysis can give insight in the correctness and competency of the model. This last reason especially motivates the discussion of causal analysis in this chapter.

Dependent on the kind of equations of the elements, the element ports can impose constraints on the connected bonds. There are four different constraints, which will be treated before a systematic procedure for causal analysis of bond graphs is discussed.

6.1 Causal constraints

Fixed causality

Fixed causality is the case, when the equations only allow one of the two port variables to be the outgoing variable. This occurs at sources: an effort source (Se) has by definition always its effort variable as signal output, and has the causal stroke outwards. This causality is called *effort-out causality* or *effort causality*. A flow source (Sf) clearly has a *flow-out causality* or *flow causality*.

Another situation where fixed causality occurs is at nonlinear elements, where the equations for that port cannot be inverted (for example, division by zero). This is possible at R, GY, TF, C and I elements. Thus, there are two reasons to impose a fixed causality:

1. There is no relation between the port variables.
2. The equations are not invertable ('singular').

Constrained causality

At TF, GY, 0– and 1–junction, relations exist between the causalities of the different ports of the element. These relations are *causal constraints*, since the causality of a particular port imposes the causality of the other ports. At a TF, one of the ports has effort-out causality and the other has flow-out causality. At a GY, both ports have either effort-out causality or flow-out causality. At a 0–junction, where all efforts are the same, *exactly one* bond must bring in the effort. This implies that 0–junctions always have exactly one causal stroke at the side of the junction. The causal condition at a 1–junction is the dual form of the 0-junction. All flows are equal, thus *exactly one* bond will bring in the flow, implying that *exactly one* bond has the causal stroke away from the 1–junction.

Preferred causality

At the storage elements, the causality determines whether an integration or differentiation with respect to time will be the case. Integration has preference above a differentiation. At the integrating form, an initial condition must be specified. Besides, integration with respect to time is a process, which can be realised physically. Numerical differentiation is not physically realisable, since information at future time points is needed. Another drawback of differentiation occurs when the input contains a step function: the output will then become infinite. Therefore, integrating causality is seen as the *preferred causality*. This implies that a C–element has effort-out causality and an I–element has flow-out causality at its preference. These preferences are also illustrated in Figure 8 and Figure 9, when looking at the block diagram expansion.

Example

When a voltage u is imposed on an electrical capacitor (a C–element), the current i is the result of the constitutive equation of the capacitor:

$$i = C\frac{\mathrm{d}u}{\mathrm{d}t}$$

A differentiation is thus happening. We have a problem when the voltage instantly steps to another value, since the current will be infinite (the derivative of a step is infinite). This is not the case when the current is imposed on a capacitor. Now, an integral is used:

$$u = u_0 + \int i\mathrm{d}t$$

The first case is flow-out causality (effort imposed, flow the result), and the second case is effort–out causality, which is the preferred causality. Furthermore, an effort–out causality also results in a state variable with initial condition u_0.

At an inductor, the dual form of the C–element is the case: flow-out causality will result in an integral causality, being the preference.

Indifferent causality

Indifferent causality is used, when there are no causal constraints! At a linear R, it does not matter which of the port variables is the output. Consider an electrical resistor. Imposing a current (flow) yields:

$u = iR$

It is also possible to impose a voltage (effort) on the linear resistor:

$$i = \frac{u}{R}$$

There is no difference choosing the current as incoming variable and the voltage as outgoing variable, or the other way around.

Overview

The Se and Sf have a fixed causality, the C and I have a preferred causality, the TF, GY, 0 and 1 have constrained causality, and the R has an indifferent causality (provided that the equations of these basic elements all are invertable). These causal forms have been shown in section 4. When the equations are *not* invertable, a *fixed causality* must be used.

6.2 Causal analysis procedure

The procedure for assigning causality on a bond graph starts with those elements that have the strongest causality constraint namely fixed causality (deviation of the causality condition cannot be granted by rewriting the equations, since rewriting is not possible). Via the bonds (i.e. connections) in the graph, one causality assignment can cause other causalities to be assigned. This effect is called *causality propagation*: after one assignment, the causality propagates through the bond graph due to the causal constraints.

The causality assignment algorithm is as follows:

1a. Chose a fixed causality of a *source* element, assign its causality, and propagate this assignment through the graph using the causal constraints. Go on until all sources have their causalities assigned.

1b. Chose a not yet causal port with *fixed* causality (non-invertable equations), assign its causality, and propagate this assignment through the graph using the causal constraints. Go on until all ports with fixed causality have their causalities assigned.

2. Chose a not yet causal port with *preferred* causality (storage elements), assign its causality, and propagate this assignment through the graph using the causal constraints. Go on until all ports with preferred causality have their causalities assigned.

3. Chose a not yet causal port with *indifferent* causality, assign its causality, and propagate this assignment through the graph using the causal constraints. Go on until all ports with indifferent causality have their causalities assigned.

Often, the bond graph is completely causal after step 2, without any causal conflict (all causal conditions are satisfied). If this is *not* the case, then the moment in the procedure where a conflict occurs or where the graph becomes completely causal, can give insight in the correctness and competence of the model. Before discussing these issues, first an example will be treated.

Example

To exemplify the causality algorithm, the same example as in section 5 is used. In Figure 27, the completely causal bond graph of the hoisting device is shown. Numbers at the causal strokes indicate the order in which the bonds were made causal.

At step 1a, we assign causality 1 and 2. It does not matter with which source we start. No causality can be propagated. Step 1b is not applicable, since there are no ports of that category. At step 2, we started with the inductor. Propagation is from stroke 3 until stroke 6. The next storage element (preferred causality) is inertia J. Propagation of this causality (number 7), completes the causality of the graph, implying that we do not need step 3. However, the mass of the load does not get his preference. What the consequences are, is subject of the next section.

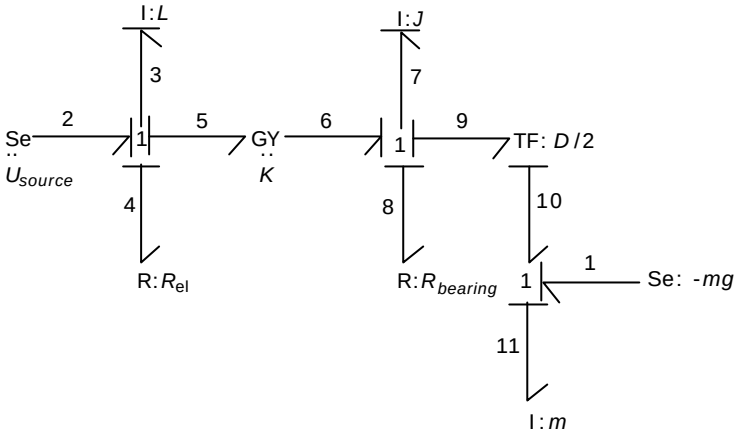

Figure 27: The causal bond graph of the hoisting device

6.3 Model insight via causal analysis

We discuss here those situations whereby conflicts occur in the causal analysis procedure or when step 3 of the algorithm appears to be necessary. The place in procedure where a conflict appears or the bond graph becomes completely causally augmented, can give insight in the correctness of the model.

Often, the bond graph is completely causal after *step 2*, *without* any causal conflict (all causal conditions are satisfied). Each storage element represents a state variable, and the set of equations is an explicit set of ordinary differential equations (not necessarily linear or time invariant).

When the bond graph is completely causal after *step 1a*, the model does *not* have any dynamics. The behaviour of all variables now is determined by the fixed causalities of the sources. Arises a causal conflict at *step 1a* or at *step 1b*, then the problem is ill posed. The model must be changed, by adding some elements. An example of a causal conflict at *step 1a* is two effort sources connected to one 0-junction. Both sources 'want' to determine the one effort variable.

At a conflict at *step 1b*, a possible adjustment is changing the equations of the fixed–causality element such that these equations become *invertable*, and thus the *fixed*ness of the constraint disappears. An example is a diode or a valve having zero current resp. flow while blocking. Allowing a small resistance during blocking, the equations become invertable.

When a conflict arises at *step 2*, a storage element receives a *non*-–preferred causality. This means that this storage element does *not* represent a *state variable*. The initial value of this storage element cannot be chosen freely. Such a storage element often is called a *dependent storage element*. This indicates that a storage element was *not* taken into account during modelling, which should be there from physical systems viewpoint. It can be deliberately omitted, or it might be forgotten. At the hoisting device example, the load of the hoist (I—element) is such a dependent storage element. Elasticity in the cable was *not* modelled. If it had been modelled, a C—storage element connected to a 0—junction between the cable drum and load would appear.

When *step 3* of the causality algorithm is necessary, a so—called *algebraic loop* is present in the graph. This loop causes the resulting set differential equations to be *implicit*. Often this is an indication that a storage element was not modelled, which should be there from a physical systems viewpoint.

In general, different ways to handle the causal conflicts arising at step 2 or step 3 are possible:

1. Add elements.
   For example, you can withdraw the decision to neglect certain elements. The added elements can be parasitic, for example, to add elasticity (C—element) in a mechanical connection, which was modelled as rigid. Additionally adding a damping element (R) reduces the simulation time considerably, which is being advised.
2. Change the bond graph such that the conflict disappears.
   For a step 2 conflict, the dependent storage element is taken together with an independent storage

element, having integral causality. For a step 3 conflict, sometimes resistive elements can be taken together to eliminate the conflict. This can be performed via transformations in the graph. The complexity of this operation depends on the size and kind of submodels along the route between the storage elements or resistors under concern.

3. The bond graph is *not* changed and during simulation, a special (implicit) integration routine is needed. The implicit equations are solved by means of the iteration schemes present in the implicit integration methods. In 20-SIM, only the BDF method can handle this situation.
   Also, the loop can be '*cut*' by adding a one-timestep delay, which is a rather pragmatic solution, but can be useful when no implicit integration methods are available. The accuracy might get too low. Note that the amplification of all elements in the loop must be smaller than one to obtain a stable solution.

Algebraic loops and loops between a dependent and an independent storage element are called *zero–order causal paths* (*ZCPs*). Besides these two kinds, there are three other kinds, having an increasing complexity and resulting in more complex equations. These occur for instance in rigid–body mechanical systems (van Dijk and Breedveld, 1991).

By interpreting the result of causal analysis, several properties of the model can be recognised, which could *otherwise* only be done after deriving equations.

6.4 Order of set of state equations

The causal analysis also gives information on the order of the set equations. The number of *initial conditions* equals the number of storage elements with integral causality, which was also the preference during causality assignment. This number is called the *order of the system.* In the example (Figure 28), the order of the system is 2.

The *order of the set state equations* is smaller than or equal to the order of the system, because storage elements can depend on each other. These kind of dependent storage elements each have their own initial value, but they together represent *one* state variable. Their input signals are equal, or have a factor in between (Figure 28).

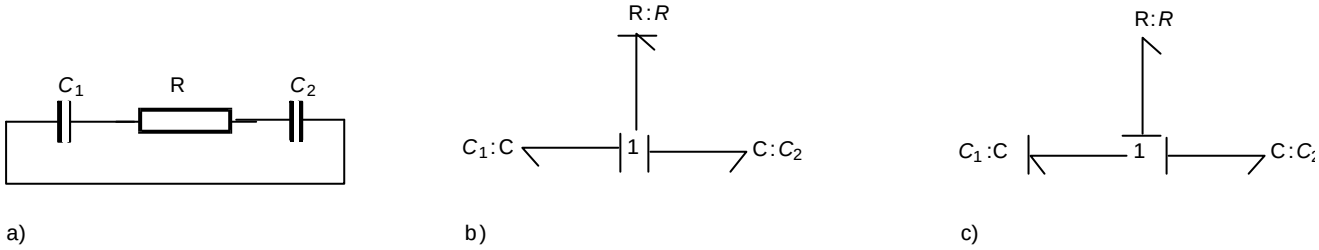

Figure 28: System with order of state equations smaller than order of the system. a) IPM; b) causality using integral preference; c) causality using differential preference.

A recipe exists to check whether this kind of dependent storage elements show up: Perform causal analysis again, but chose differential causality as preference. For the example, this is done in Figure 29.

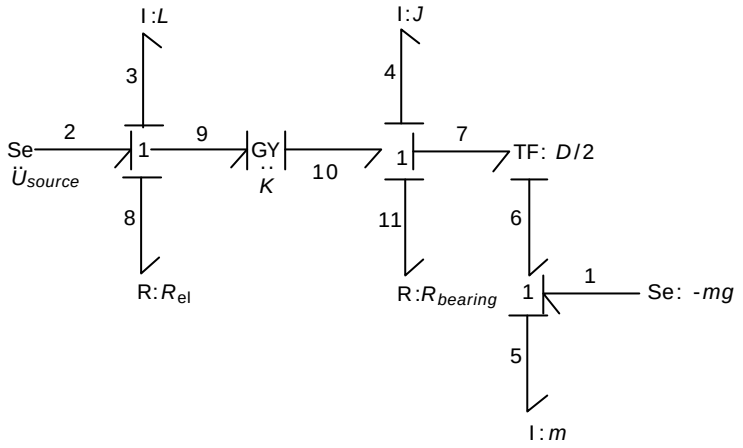

Figure 29: Differential preference causality for the hoisting device

Those storage elements that get both at differential preference and at integral preference their preferred causality are the *real* storage elements and contribute to the state of the system. The order of the set of state equations, is by definition the amount of storage elements that get in both cases their preferred causality. At the hoisting device, this is 2 (namely L and J). The storage elements that get in both cases *not* their preferred causality, are the dependent storage elements. Those storage elements that get *only* at the integral preferred causality case their preferred causality are called *semi-dependent storage elements*, to distinguish them from the dependent storage elements. This indicates that a storage element was not taken into account during modelling, which should be there from physical systems viewpoint.

6.5 Matrix form (linear systems)

If the system is linear, we can write the resulting set of state equations in the standard form, namely,

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$

where **A** is the *system matrix*, and **B** is the *input matrix*. The order of the system is the dimension of the square matrix **A** and the order of the set of state equations is the *rank* of **A**.

When dependent states or algebraic loops are present, the matrix description is as follows:

$$\mathbf{E}\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$

where **E** is a square matrix of size equal to the amount of storage elements plus the amount of algebraic loops. For each differential storage element and for each algebraic loop, **E** contains one row of zeros. The state vector **x** is extended with the algebraic loop variables, to enable implicit integration methods to solve these kinds of systems. The hoisting device gives such a set of equations.

7 Generation of equations

A causal bond graph contains all information to derive the set of state equations. It is either a set of *ordinary first–order differential equations, ODEs*, when the model is explicit (*no* causal conflicts), or a set of *differential and algebraic equations, DAE*s, when the model is implicit (a causal conflict in step 2 of the procedure or step 3 is necessary, cf. section 6.2).

The procedure to derive the equations is covered by bond–graph software like Enport (Rosenberg, 1974), MS1 (Lorenz, 1997), CAMP (Granda, 1985) and 20-SIM (Broenink, 1990, 1995, 1997, 1999b; Broenink and Kleijn, 1999). Therefore, in practice, generation of equations need not be done by hand. However, we *do* discuss the generation of equations on the one hand to be complete and on the other hand to indicate what exactly has to be done.

We use the following procedure to generate equations:

1. We first write the *set of mixed differential and algebraic equations*. These are the constitutive relations of all elements in computational form, or *causal form.* This comprises of 2*n* equations of

a bond graph having *n* bonds. *n* equations compute an effort and *n* equations compute a flow, or derivatives of them.

2. We then eliminate the algebraic equations. We can organise this elimination process by first eliminate the identities coming from the sources and junctions. Thereafter, we substitute the multiplications with a parameter, stemming from resistors and transducers (TF, GY). At last, we substitute the summation equations of the junctions into the differential equations of the storage elements. Within this process, it is efficient to *first* mark the state variables. In principle, the state variables are the contents of the storage elements (*p* or *q* type variables). However, if we write the constitutive relations of storage elements as one differential equation, we can also use the *efforts* at C-elements and *flows* at I-elements.

If we are going to generate the equations by hand, we can take the first elimination step into account while formulating the equations by, at the sources, directly use the signal function at the bond. Furthermore, we can write the variable determining the junction along all bonds connected to that junction. The *variable determining the junction* is that variable, which gets assigned to bond variables of all the other bonds connected to that junction via the identities of the junction equations. At a 0–junction, this is the effort of the only bond with its causal stroke *towards* the 0-junction. At a 1–junction, this is the flow of the only bond with its causal stroke *away from* the 1-junction.

In case of dependent storage elements, we have to take care that the accompanying state variable gets not eliminated. These are the so–called *semi state variables*. When we mark the state variables, including the semi state variables in this situation, on beforehand, we can prevent the wrong variable from being eliminated. In case of algebraic loops, implicit equations will be encountered. We choose one of the variables in these loops as *algebraic loop breaker* and that variable becomes a *semi state variable*. See also section 6.5. The equation consisting the semi state variable of a storage element gets eliminated at the second elimination step: it is a multiplication. The semi state variable itself must *not* be substituted.

7.1 Example

The *set of mixed differential and algebraic equations* of the hoisting device is shown below. The efforts and flows are numbered in the same order as their causality was assigned. We have 22 equations, of which 11 compute a flow and 11 compute an effort. Since we want to generate the set of equations as differential equations, we write at the storage elements in integral form the equations as differential equations and not as integral equations. This is called *deferred integration*.

$$
\begin{aligned}
e_2 &= u_{source} \\
\frac{\mathrm{d}f_3}{\mathrm{d}t} &= \frac{1}{L} e_3 \\
e_4 &= R_{el}\, f_4 \\
f_2 &= f_3 \\
f_4 &= f_3 \\
f_5 &= f_3 \\
e_3 &= e_2 - e_4 - e_5 \\
e_5 &= K f_6 \\
e6 &= K f_5 \\
\frac{\mathrm{d}f_7}{\mathrm{d}t} &= \frac{1}{J} e_7 \\
e_8 &= R_{bearing} f_8 \\
f_6 &= f_7 \\
f_8 &= f_7 \\
f_9 &= f_7
\end{aligned}
\qquad
\begin{aligned}
e_7 &= e_6 - e_8 - e_9 \\
e_9 &= -\frac{D}{2} e_{10} \\
f_{10} &= -\frac{D}{2} f_9 \\
f_1 &= f_{10} \\
f_{11} &= f_{10} \\
e_{10} &= e_{11} - e_1 \\
e_1 &= -mg \\
e_{11} &= m \frac{\mathrm{d}f_{11}}{\mathrm{d}t}
\end{aligned}
$$

As state variables, we have f_3 and f_7, whereas f_{11} is the semi state variable belonging to the mass having derivative causality (it is an I element). Note that the equation of the mass is written in causal form: e_{11} is the output. After eliminating the identities at the junctions and sources, we will get the following equations. Note that the flows of the two effort sources (f_1 and f_2) are not used elsewhere, so we leave them out.

$$\frac{\mathrm{d}f_3}{\mathrm{d}t} = \frac{1}{L}e_3 \qquad e_7 = e_6 - e_8 - e_9$$
$$e_4 = R_{el} f_3 \qquad e_9 = -\frac{D}{2}e_{10}$$
$$e_3 = u_{source} - e_4 - e_5 \qquad f_{11} = -\frac{D}{2} f_7$$
$$e_5 = Kf_7 \qquad e_{10} = e_{11} + mg$$
$$e_6 = Kf_3 \qquad e_{11} = m\frac{\mathrm{d}f_{11}}{\mathrm{d}t}$$
$$\frac{\mathrm{d}f_7}{\mathrm{d}t} = \frac{1}{J}e_7$$
$$e_8 = R_{bearing} f_7$$

After substitution of the multiplication with resistors and transducers, our set of equations reduces to the following 6 equations: 2 differential equations, 1 constraint equation, computing the semi state variable f_{11} and 4 junction equations. Below, only state variables, junction variables and input variables are used.

$$\frac{\mathrm{d}f_3}{\mathrm{d}t} = \frac{1}{L}e_3$$
$$e_3 = u_{source} - R_{el} f_3 - Kf_7 \qquad f_{11} = -\frac{D}{2} f_7$$
$$\frac{\mathrm{d}f_7}{\mathrm{d}t} = \frac{1}{J}e_7 \qquad e_{10} = m\frac{\mathrm{d}f_{11}}{\mathrm{d}t} + mg$$
$$e_7 = Kf_3 - R_{bearing} f_7 + \frac{D}{2}e_{10}$$

After substitution of the summations at the junctions into the differential equations and the constraint equations, the result looks like:

$$\frac{\mathrm{d}f_3}{\mathrm{d}t} = \frac{1}{L}u_{source} - \frac{R_{el}}{L} f_3 - \frac{K}{L} f_7$$
$$\frac{\mathrm{d}f_7}{\mathrm{d}t} = \frac{K}{J} f_3 - \frac{R_{bearing}}{J} f_7 + \frac{D}{2} m\frac{\mathrm{d}f_{11}}{\mathrm{d}t} + \frac{D}{2} mg$$
$$f_{11} = -\frac{D}{2} f_7$$

This system is a linear system, so we can write the equations in matrix form (according to the second form of section 6.5):

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -\frac{D}{2} \\ 0 & 0 & 0 \end{pmatrix} \frac{d}{dt} \begin{pmatrix} f_3 \\ f_7 \\ f_{11} \end{pmatrix} = \begin{pmatrix} -\frac{R_{el}}{L} & -\frac{K}{L} & 0 \\ \frac{K}{J} & -\frac{R_{bearing}}{J} & 0 \\ 0 & \frac{D}{2} & 1 \end{pmatrix} \begin{pmatrix} f_3 \\ f_7 \\ f_{11} \end{pmatrix} + \begin{pmatrix} \frac{1}{L} & 0 \\ 0 & \frac{D}{2} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} u_{source} \\ mg \end{pmatrix}$$

When the model first was made explicit by adding elements, according to alternative 1 of section 6.3, the causal bond graph and the equations are given below. We add the elasticity of the rope: A C-element connected to a 0-junction is inserted on the bond between the TF of the cable drum and the 1-junction of the payload (see Figure 30). The efforts and flows are numbered in the same order as their causality was assigned. We have 26 equations, of which 13 compute a flow and 13 compute an effort.

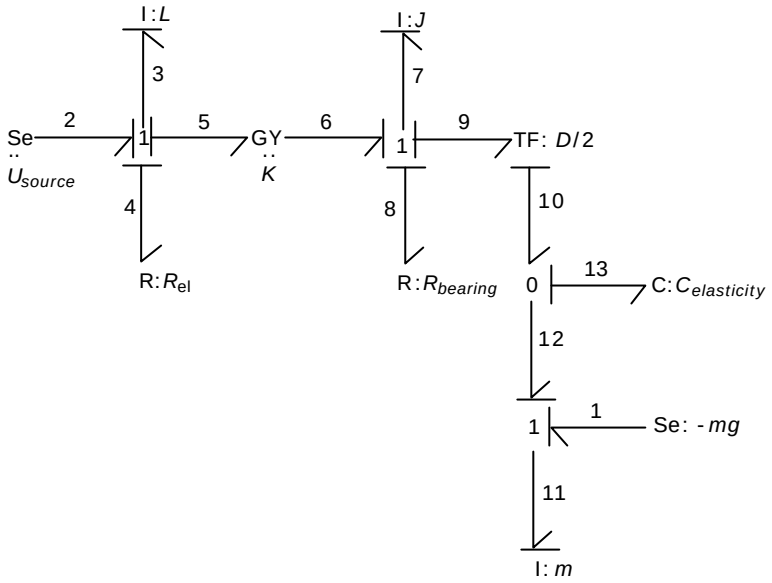

Figure 30 Causal bond graph of the Hoist with elasticity added

The mixed set of differential and algebraic equations is:

$$e_2 = u_{source} \qquad e_7 = e_6 - e_8 - e_9$$

$$\frac{df_3}{dt} = \frac{1}{L} e_3 \qquad e_9 = -\frac{D}{2} e_{10}$$

$$e_4 = R_{el}\, f_4 \qquad f_{10} = -\frac{D}{2} f_9$$

$$f_2 = f_3 \qquad e_{10} = e_{11}$$

$$f_4 = f_3 \qquad e_{12} = e_{11}$$

$$f_5 = f_3 \qquad f_{11} = f_{10} - f_{12}$$

$$e_3 = e_2 - e_4 - e_5 \qquad \frac{de_{11}}{dt} = \frac{1}{C_{elasticity}} f_{11}$$

$$e_5 = K f_6 \qquad f_{12} = f_{13}$$

$$e6 = K f_5 \qquad f_1 = f_{13}$$

$$\frac{df_7}{dt} = \frac{1}{J} e_7 \qquad e_{13} = e_1 + e_{12}$$

$$e_8 = R_{bearing} f_8 \qquad e_1 = -mg$$

$$f_6 = f_7 \qquad \frac{df_{13}}{dt} = \frac{1}{m} e_{13}$$

$$f_8 = f_7$$

$$f_9 = f_7$$

As state variables, we have f_3, f_7, e_{11} and f_{13}. Applying the procedure of deriving equations leads to the following set of state equations written in matrix form:

$$\frac{d}{dt}\begin{pmatrix} f_3 \\ f_7 \\ e_{11} \\ f_{13} \end{pmatrix} = \begin{pmatrix} -\frac{\mathrm{Re}\,l}{L} & -\frac{K}{L} & 0 & 0 \\ \frac{K}{J} & -\frac{R_{bearing}}{J} & -\frac{D}{2} & 0 \\ -\frac{D}{2C_{elasticity}} & -\frac{1}{C_{elasticity}} & 0 & 0 \\ 0 & 0 & \frac{1}{m} & 0 \end{pmatrix} \begin{pmatrix} f_3 \\ f_7 \\ e_{11} \\ f_{13} \end{pmatrix} + \begin{pmatrix} \frac{1}{L} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & -\frac{1}{m} \end{pmatrix} \begin{pmatrix} u_{source} \\ mg \end{pmatrix}$$

8 Expansion to block diagrams

To show that a causal bond graph is a compact block diagram, we treat in this section the expansion of a causal bond graph to a block diagram. Furthermore, a block diagram representation of a system might be more familiar than a bond graph representation. Thus this work might help understanding bond graphs.

The expansion of a causal bond graph into a block diagram consists of three steps:

1. Expand all bonds to *bilateral signal flows* (two signals with opposite directions). The *causal stroke* determines in which direction the effort flows. The bond graph elements can be encircled to connect the signals to.
2. Replace the bond–graph elements by their block–diagram representations (see section 4). Deduce the signs of the summations of the junctions from the directions of the bond arrows (half arrows, see section 4.5). Often, it is efficient to determine those signs after all bond–graph elements are written in block diagram form.
   The block diagram is ready in principle.
3. Redraw the block diagram in *standard form*: all integrators in an ongoing stream (form left to right) and all other operations as feedback loops. Of course, this is not always possible. Blocks might be taken together. Since block diagrams represent mathematical operations, for which *commutative* and *associative* properties apply, these properties can be used to manipulate the block diagram such that the result looks appealing enough.

As an example, we show the block diagram of the hoisting device, using the three steps to construct the block diagram.

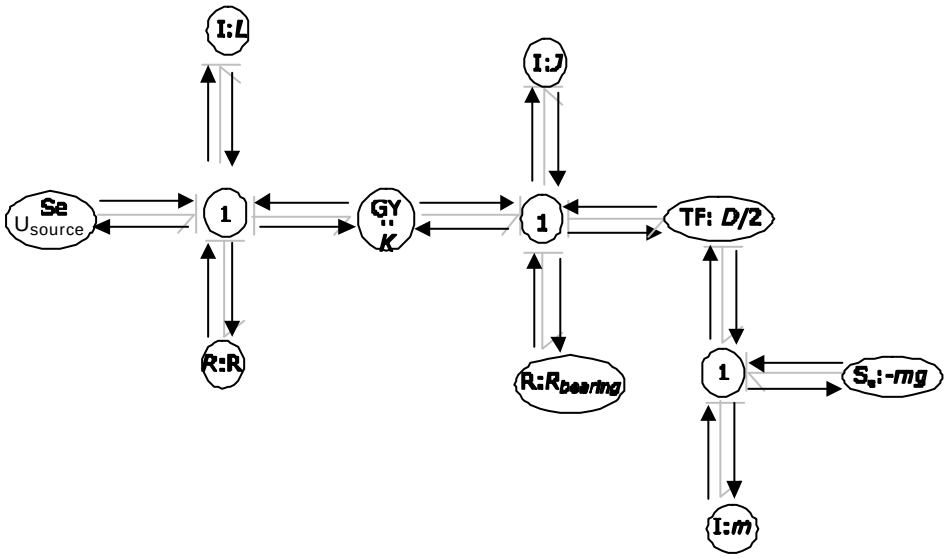

Figure 31 Expansion of bonds to bilateral signal flows of the hoisting device

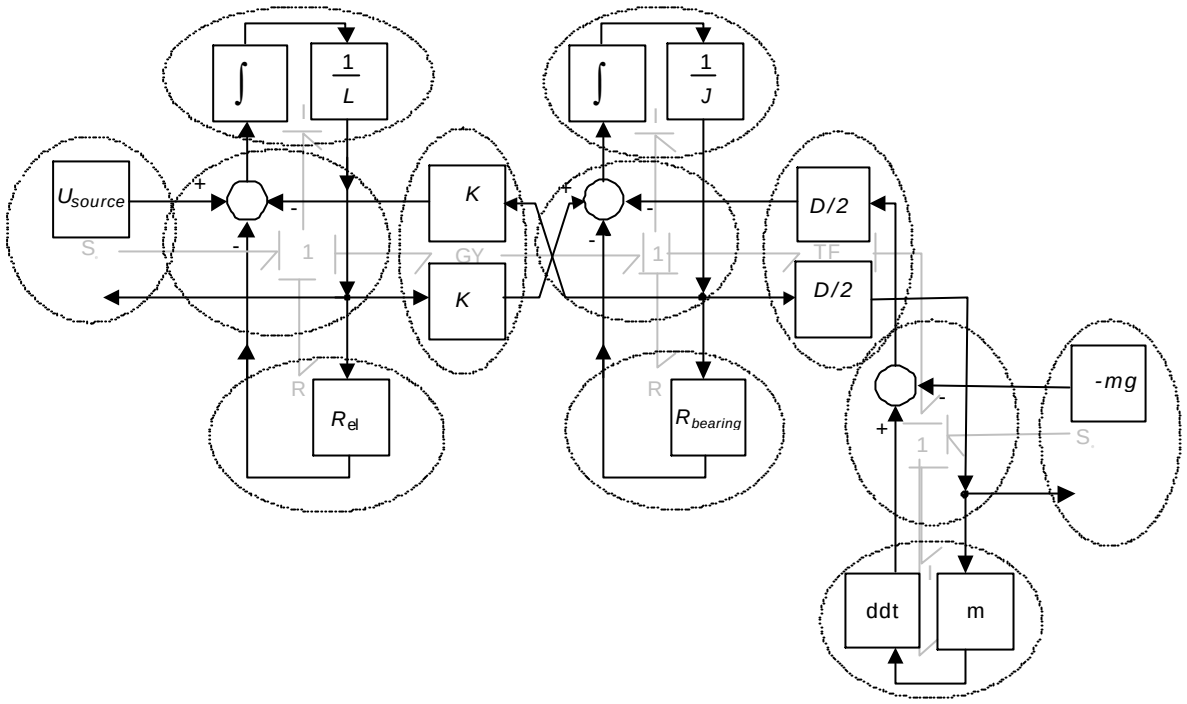

Figure 32 Bond graph expanded to a block diagram in the layout of the bond graph (the bond graph is shown in grey)

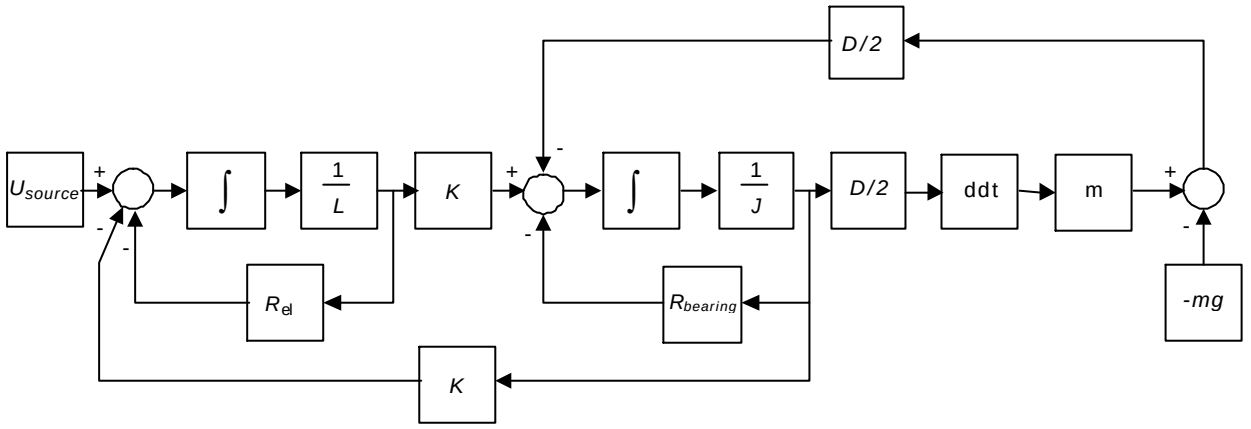

Figure 33 The block diagram redrawn in standard form

The block diagram from the causal bond graph of the Hoist with elasticity added (see Figure 30) is shown in Figure 34.

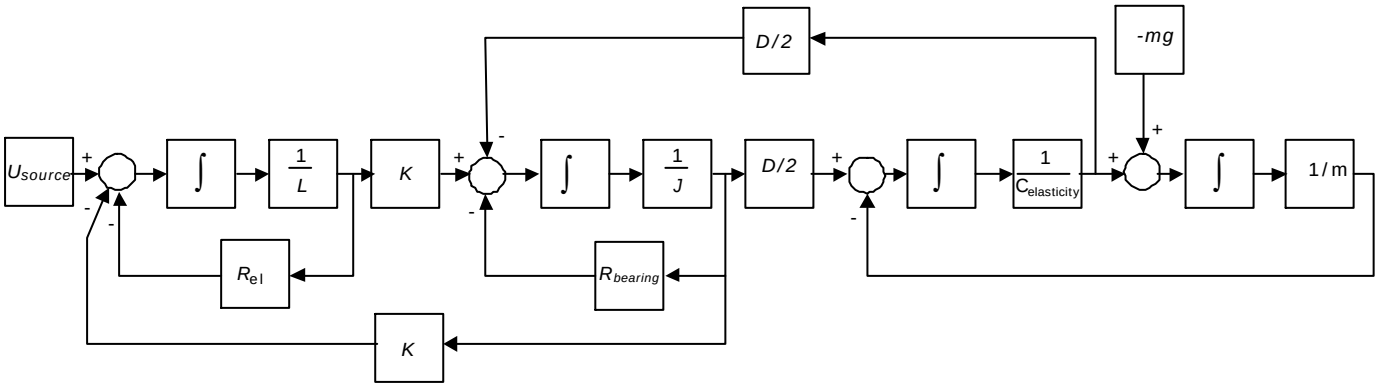

Figure 34 The block diagram of the Hoist with elasticity (cf Figure 30)

9 Simulation

The resulting set of equations coming from a bond–graph model is called the *simulation model*. It consists of first–order ordinary differential equations (ODEs), possibly extended with algebraic constraint equations (DAEs). Hence, it can be simulated using standard numerical integration methods. However, because numerical integration is an *approximation* of the actual integration process, it is useful to check the simulation model on aspects significant for simulation. As a result, an appropriate integration method can be chosen: the computational work is minimal and the results stay within a specified error margin.

Since at causal analysis, one can decide whether or not to change the bond–graph model to obtain an explicit simulation model, it is useful to know about the consequences for simulation of relevant characteristics of the simulation model. The following 4 aspects of simulation models are relevant for choosing an numerical integration method:

1. Presence of implicit equations.
   Implicit models (DAEs) can only be simulated with *implicit* integration methods. The iteration procedure of the implicit integration method is also used to calculate the implicit model. Explicit models (ODEs) can be simulated with both explicit as implicit integration methods. Sometimes, implicit integration methods need more computation time than explicit integration methods. During causal analysis, one can see whether a simulation model will be explicit or implicit, see section 6.3.
2. Presence of discontinuities.
   Integration methods with special provisions for events will perform best. If that is not available, variable step methods can be used. Multistep methods become less accurate, since they need information from the past, which is useless after a discontinuity.
   While constructing the model, presence of discontinuities can be marked.
3. Numerical stiffness.
   S(t), the *stiffness ratio*, is a measure for the distance between real parts of eigenvalues, λ, namely $S(t) = \max(|Re(\lambda(t)|) / \min(|Re(\lambda(t)|)$. Stiff models (large S) need stiff integration methods. The time step is now determined by the stability instead of the accuracy (namely, eigenvalues are now used to determine the step size). When the high frequency parts are faded out, they do not influence the step size anymore. Hence, the step size can grow to limits determined by the lower frequencies.
4. Oscillatory parts.
   When a model has *no* damping, it should not be simulated with a stiff method. Stiff methods perform badly for eigenvalues on the imaginary axis (i.e. no damping) of the complex eigenvalue plane.

Eigenvalues can be localised in a causal bond graph, especially when *all* elements are linear. There is a bond graph version of *Mason's Loop rule* to determine the transfer function from a bond graph (Brown, 1972). As a side effect, the eigenvalues can be calculated. We will not discuss the procedure to obtain eigenvalues from a causal bond graph by hand.

10 Conclusion

10.1 Review

In this chapter, we have introduced bond graphs to model physical systems in a *domain independent* way. Only *macroscopic* systems are treated, thus *quantum effects* do *not* play a significant role. Domain indepence has its basics in the fact that *physical* concepts are analogous for the different physical domains. 6 different elementary concepts exist: storage of energy, dissipation, transduction to other domains, distribution, transport, input or output of energy.

Another starting point is that it is possible to write models as *directed graphs*: parts are interconnected by bonds, along which exchange of energy occurs. A bond represents the energy flow between the two

connected submodels. This energy flow can be described as the product of 2 variables (effort and flow), letting a bond be conceived as a *bilateral signal* connection. During modelling, the first interpretation is used, while during analysis and equations generation the second interpretation is used.

Furthermore, we presented a method to systematically build a bond graph starting from an ideal physical model. Causal analysis gives, besides the computational direction of the signals at the bonds, also information about the correctness of the model. We presented methods to derive the causality of a bond graph. In addition, procedures to generate equations and block diagrams out of a causal bond graph are presented.

This chapter is only an introduction to bond graphs. Sometimes, procedures are just presented, without a deep motivation and possible alternatives. It was also *not* the incentive to elaborate on physical systems modelling. We did *not* discuss multiple connections (arrays of bonds written as one multibond) and multiport elements (to describe transducers), neither different causal analysis algorithms. Those different causality algorithms give slightly different sets of DAEs especially when applied to certain classes of models (for instance multibody systems with kinematic loops).

10.2 Object-oriented physical-systems modelling

Note that bond–graph modelling is in fact a form of *object–oriented physical-systems modelling*, a term which is currently often used (Andersson, 1994; Elmqvist et al, 1993; Mattson et al, 1997; Åström et al, 1998). This can be seen as follows: bond–graph models are declarative, can be hierarchically structured, and *fully* support encapsulation (due to the *non-causal* way of specifying equations, and the notion of *ports*). Furthermore, due to allowing hierarchy, the notion of definition and use of models are distinguished (i.e. the class concept and instantiation). Since bond graphs came into existence before the term *object oriented* was used in the field of physical systems modeling, bond graphs can be seen as an object-oriented physical–systems modeling paradigm *avant-la-lettre*.

A bond-graph library was written in Modelica (Mattson et al, 1997; Åström et al, 1998), the upcoming object-oriented modelling language (Broenink, 1997b). The basic bond–graph elements and block–diagram elements have been specified in Modelica, using the essential object–orientation features *inheritance* and *encapsulation.* Equations have been specified in an *acausal* format. Thus, it can be said that the Modelica modelling concepts are consistent with bond–graph concepts. Furthermore, automatic Modelica code generation from bond graphs appeared to be rather straightforward (Broenink, 1999a).

10.3 Further Reading

For a more thorough analysis of bond graphs, see Paynter (1961) and Breedveld (1984, 1985), while an extensive discussion on textbook level is given by Karnopp, Margolis and Rosenberg (1990). Cellier (1991) wrote a textbook on continuous system modelling in which besides bond graphs also other modelling methods are used. Current research on bond graphs is reported at the International Conference on Bond Graph Modeling, every two years (Granda and Cellier, 1993, 1995, 1997, 1999). Journals regularly publishing bond graph papers are the Journal of the Franklin Institute, which also had special issues on bond graphs (1991) and the Journal of Dynamic Systems, Measurement and Control.

11 Acknowledgements

Most of the inspiration for this chapter came from the Dutch course material of Breedveld and Van Amerongen (1994). I sincerely acknowledge Peter Breedveld and Job van Amerongen for their valuable suggestions and discussions.

References

Andersson, M., *Object–oriented modeling and simulation of hybrid systems*, PhD Thesis, Lund Institute of Technology, Sweden, (1994).

Åström K.J., Elmqvist, H, Mattson, S.E., (1998), Evolution of Continuous–time modeling and simulation, *Proceedings of the 12th European Simulaton Multiconference (ESM'98)*, SCS Publising, Manchester UK, pp 9-18.

Breedveld P.C. and Amerongen, J. van (1994), *Dynamische systemen: modelvorming en simulatie met bondgrafen*, (Dynamic systems: modelling and simulation with bond graphs), part 1-4, (in Dutch), Dutch Open University, Heerlen, Netherlands, ISBN 90 358 1302 2.

Breedveld P.C., (1982), Thermodynamic Bond Graphs and the problem of Thermal Inertance, *Journal of the Franklin Institute*, vol 314, no 1, pp. 15-40.

Breedveld P.C., (1984), *Physical systems theory in terms of bond graphs*, PhD thesis, University of Twente, Enschede, Netherlands.

Breedveld P.C., (1985), Multibond–graph elements in physical systems theory, *Journal of the Franklin Institute*, vol 319, no 1/2, pp. 1-36.

Breedveld, P.C. and Dauphin–Tanguy G. (1991), (eds.) *Current topics in bond graph related research*, Journal of The Franklin Institute, Special issue on bond–graph modeling, Volume 328 no 5/6, Pergamon Press.

Broenink J.F. (1990), *Computer–aided Physical–systems modeling and simulation: a bond–graph approach*, PhD thesis, University of Twente, Enschede, Netherlands.

Broenink, J.F. (1997b), Bond–graph modeling in Modelica, *Proceedings of 9th European Simulation Symposium*, W Hahn, A Lehmann (eds.), Passau Germany, Oct 19-22, pp 137-141.

Broenink, J.F., (1997a), Modelling, Simulation and Analysis with 20-Sim. *Journal A*, Special issue on CACSD, vol 38 no 3, pp. 22-25, ISSN 0771-1107.

Broenink, J.F. (1999a), Object-oriented modelling with bond graphs and modica. *International Conference on Bond Graph Modeling ICBGM'99*, Simulation Series Vol 31 no 1, SCS, pp. 163-168.

Broenink, J.F., (1999b), 20-SIM, software for hierarchical bond-graph / block-diagram models, Submitted to *Journal Simulation in Practice and Theory*.

Broenink, J.F., and C. Kleijn, (1999), Computer-aided design of mechatronic systems using 20SIM 3.0, Submitted to WESIC'99, Workshop on Scientific and Industrial Collaboration, Sep 1-3, Newport UK.

Broenink, J.F., Weustink, P.B.T., (1995) PC–version of the bond graph modeling analysis and simulation tool CAMAS, Proc. *International Conference on Bond Graph Modeling ICBGM'95*, Simulation Series Vol 27 no 1, SCS, pp. 203-208.

Brown, F.T., (1972), Direct application of the loop rule to bond graphs, *Journal of Dynamic Systems, Measurement and Control*, Vol 94, no 3, pp253-261.

Cellier F.E., Granda J.J., (1995), (eds.), *International Conference on Bond Graph Modeling ICBGM'95*, Simulation Series Vol 27 no 1, Society for Computer Simulation, San Diego, CA.

Cellier, F.E., (1991) *Continuous System Modeling*, Springer Verlag, ISBN 0 387 97502 0

Dijk, J. van, Breedveld P.C., (1991), *Simulation of System Models Containing Zero-order Causal Paths*, *J. Franklin Institute 328 (5/6)*, 959-980

Dixhoorn, J.J. van, (1982), Bond graphs and the challenge of a unified modelling theory of physical systems, *Progress in modelling and simulation*, Cellier F.E. (ed.), Academic Press, New York, pp 207-245.

Elmqvist, H., Cellier, F.E., and Otter, M. (1993) Object–oriented modeling of hybrid systems, *Proc. European Simulation Symposium*, Delft, Netherlands, 31-41, (1993).

Granda J.J., (1985), Computer generation of physical system differential equations using bond graphs, *Journal of the Franklin Institute*, vol 319, no 1/2, pp 243-256.

Granda J.J., Cellier F.E., (1993), (eds.), *International Conference on Bond Graph Modeling ICBGM'93*, Simulation Series Vol 25 no 2, Society for Computer Simulation, San Diego, CA.

Granda J.J., Cellier F.E., (1999), (eds.), *International Conference on Bond Graph Modeling ICBGM'99*, Simulation Series, Vol 31, Society for Computer Simulation, San Diego, CA.

Granda J.J., Dauphin-Tanguy G., (1997), (eds.), *International Conference on Bond Graph Modeling ICBGM'97*, Simulation Series Vol 29 no 1, Society for Computer Simulation, San Diego, CA.

Karnopp D.C., and Rosenberg R.C., (1968), *Analysis and Simulation of Multiport Systems – The bond graph approach to physical system dynamics*, MIT Press, Cambridge MA.

Karnopp D.C., and Rosenberg R.C., (1975), *System dynamics, a unified approach*, J Wiley, New York, NY.

Karnopp D.C., Margolis D.L., and Rosenberg R.C., (1990), *System dynamics, a unified approach*, (2nd edition), J Wiley, New York, NY, ISBN 0 471 45940 2. (Textbook, a teacher's guide is also available).

Kron G., (1963), *Diakoptics: The piecewise Solution of large–scale systems*, Macdonald & Co. Ltd., London.

Lorenz F., (1997), *Modelling System 1*, users manual, Lorenz Simulation, Liege, Belgium

Mattson, S.E., Elmqvist, H., and Broenink, J.F., (1997), Modelica: An international effort to design the next generation modeling language, *Journal A, Special Issue CACSD*, vol 38 (3), pp 22-25.

Paynter, H.M. (1961), *Analysis and design of engineering systems*, MIT Press, Cambridge, MA.

Rosenberg R.C., Karnopp D.C., (1983), *Introduction to physical system dynamics*, McGraw Hill, New York, NY.

Rosenberg, R.C. (1974), *A user's guide to ENPORT–4*, Wiley, New York.

Thoma, J.U. (1975), *Introduction to bond graphs and their applications*, Pergamon Press, Oxford.

Thoma, J.U. (1989) Simulation by bond graphs - Introduction to a graphical method, Springer Verlag.