MODELING INTERCONNECTED SYSTEMS

Jan C. Willems
ESAT-SISTA
K.U. Leuven
B-3001 Leuven, Belgium
Jan.Willems@esat.kuleuven.be
www.esat.kuleuven.be/∼jwillems

***Abstract*— A procedure for modeling interconnected systems is outlined, following the methodology of tearing, zooming, and linking. The interconnection architecture is a graph with leaves. The nodes are associated with the subsystems, the edges correspond to the interconnected terminals, and the leaves correspond to the terminals through which the interconnected system can interact with its environment. The subsystems are modeled as behavioral systems with terminals. The manifest variables, the variables at which the model aims, are specified by the manifest variable assignment. The system behavior is obtained by combining the module behavior, the interconnection constraints, and the manifest variable assignment.**

***Index Terms*— Interconnected systems, zooming, tearing, linking, manifest variables.**

I. INTRODUCTION

We outline a formal procedure for obtaining a model by viewing a system, a black box, as an interconnection of subsystems, smaller black boxes. This procedure is useable both in a pedagogical environment and as a blueprint for computer implementations [1]. This modeling methodology is in contrast to modeling using output-to-input assignment, which has limited applicability [3].

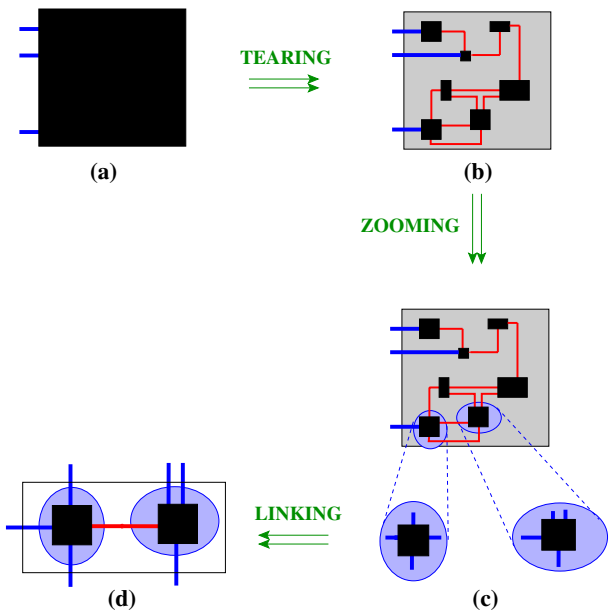

Fig. 1. Tearing, zooming, and linking

The problem addressed is to provide a mathematical language for obtaining a model for certain specified variables in an interconnected system from a model of the subsystems, the way in which the systems are interconnected, and the interconnection constraints, keeping in mind Figure 1. The formalism uses the notions of a behavior and of latent variables in an effective way [2], [3]. The basic ingredients are:

(i) terminals,
(ii) (parameterized) modules,
(iii) the interconnection architecture,
(iv) the module embedding, and
(v) the manifest variable assignment.

II. TERMINALS AND MODULES

A *terminal* is specified by its *type*. The terminal type may be of a physical nature, such as electrical, mechanical, hydraulic, or thermal type, or logical, such as input or output type. The terminal type implies the nature of the variables that live on this terminal. For example, a voltage and current for a terminal of electrical type, a force and position for a 1D mechanical terminal, a force, position, angle, and torque for a 2D or 3D mechanical terminal, a pressure and mass flow for a hydraulic terminal, a temperature and heat flow for a thermal terminal, an input for a terminal where a variable is imposed on the system, or an output for a terminal where a variable is imposed on the environment.

A *module* is a dynamical system with a finite number of terminals, and a specification of the behavior of the terminal variables as a dynamical system, a behavior. By specifying the *type* of the module, we provide a list of its terminals and their type, and therefore a list of the variables that live on the terminals of the module. Usually, the module specification involves, in addition to its type, a set of *parameters*, reflecting the material, geometric, and other properties of the physical device. We assume that, by providing the type of a module and its parameter values, we obtain a specification of the behavior, in the sense of the definition of a dynamical system [2], [3], of the variables on the terminals of the physical device.

In order to make concrete what we mean by modules and terminals, we give as an example a 3-ohm resistor. Its module type is *ohmic resistor*. This characterization means that the

978-1-4244-1688-2/08/$25.00

module has two terminals, both of electrical type, and that the module is parameterized by a nonnegative real number, the value of the resistor in ohms. Since the terminals are electrical, there are two variables, a voltage and a current, counted positive when the current flows into the resistor, on each terminal. In total there are thus four real variables associated with a resistor, namely, (V_1, I_1) and (V_2, I_2). From the fact that we have an ohmic resistor, we know that the relationship among these variables is

$$V_1 - V_2 = RI_1, \quad I_1 + I_2 = 0,$$

where R is the value of the resistor in ohms. Setting $R = 3$ yields the behavioral equations

$$V_1 - V_2 = 3I_1, \quad I_1 + I_2 = 0.$$

These equations completely specify the behavior of an ohmic resistor with parameter value 3.

III. The Interconnection Architecture and the Module Embedding

The layout of an interconnected system is visualized as a graph with modules in the vertices and connected terminals as edges (see Figure 2).

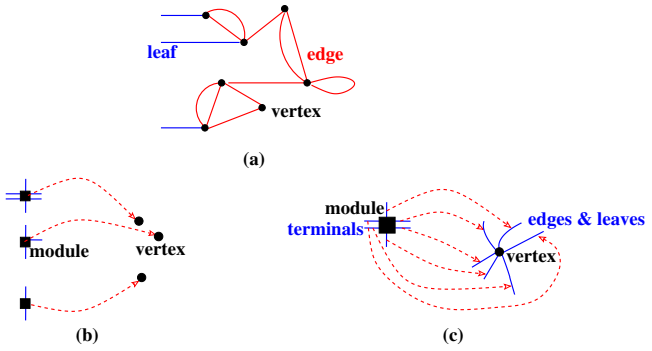

Fig. 2. Interconnection architecture

This layout is formalized by the interconnection architecture and the module embedding. The *interconnection architecture* or the *interconnection graph* is a graph with leaves. Recall that a *graph* is defined as $\mathscr{G} = (\mathbb{V}, \mathbb{E}, \mathscr{A})$, where $\mathbb{V}$ is a set of *vertices*, $\mathbb{E}$ is a set of *edges*, and $\mathscr{A}$ is the *adjacency map*. The adjacency map $\mathscr{A}$ associates with each edge $e \in \mathbb{E}$ an unordered pair $\mathscr{A}(e) = [v_1, v_2]$ with $v_1, v_2 \in \mathbb{V}$; the edge e is *adjacent* to v_1 and v_2. A graph with leaves (see Figure 2(a)) is a graph in which some special 'edges', called leaves, are adjacent to only one vertex. Formally, a *graph with leaves* is defined as $\mathscr{G} = (\mathbb{V}, \mathbb{E}, \mathbb{L}, \mathscr{A})$, where $\mathbb{V}$ is a set of *vertices*, $\mathbb{E}$ is a set of *edges*, $\mathbb{L}$ is a set of *leaves*, and $\mathscr{A}$ is the *adjacency map*. The adjacency map $\mathscr{A}$ associates with each edge $e \in \mathbb{E}$ an unordered pair $\mathscr{A}(e) = [v_1, v_2]$ with $v_1, v_2 \in \mathbb{V}$, and with each leaf $\ell \in \mathbb{L}$ an element $\mathscr{A}(\ell) = v \in \mathbb{V}$; e is *adjacent* to v_1 and v_2, while ℓ is adjacent to v. The *degree* of a vertex is the sum of the number of edges and the number of leaves that are adjacent to the vertex. A *self-loop*, that is, an edge with $\mathscr{A}(e) = [v, v]$, contributes 2 to the degree of v.

Modeling an interconnected system requires specifying the laws of the subsystems, as well as the interconnection of the subsystems. The concept that formalizes the way in which the subsystems are embedded in the overall system is the *module embedding*, which associates a module with each vertex of the interconnection architecture, as illustrated in Figure 2(b). The degree of the vertex is assumed to be equal to the number of terminals of the associated module. Moreover, the module embedding determines, for every vertex, a one-to-one assignment between the terminals of the module that has been associated with the vertex and the edges and leaves adjacent to the vertex, as illustrated in Figure 2(c). The edges serve to specify how terminals of subsystems are connected, while the leaves allow for unconnected terminals, for example, terminals by which the interconnected system can interact with its environment.

Since each edge is adjacent to two vertices, the module embedding assigns two terminals to each edge. We postulate that this assignment results in two terminals that are of the same type if the terminals are of physical type — both electrical, mechanical, hydraulic, or thermal — or of opposite type — one input, one output — if the terminals are of logical type. In other words, if the edge e is adjacent to vertices v_1 and v_2, then the module embedding must imply that v_1 and v_2 are either of the same physical type, or of opposite logical type. In this way, each vertex is labeled as a module, and each edge and leaf are labeled by a terminal type.

IV. Examples

The following examples illustrate interconnection architectures and module assignments.

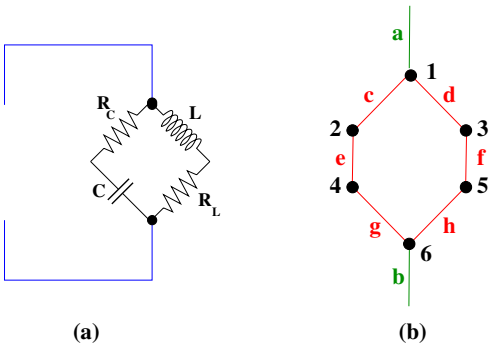

Fig. 3. RLC circuit

Consider the electrical circuit of Figure 3. The goal is to model the external port behavior the circuit. The port variables consist of the difference of the voltages of the external terminals and the current that flows into the circuit along the upper terminal. This circuit has 6 modules, two ohmic resistors denoted by R_C and R_L, respectively, one capacitor denoted by C, one inductor denoted by L, and two connectors denoted by connector1 and connector2, respectively. The parameter value of modules R_C, R_L, C, and L are denoted by the same symbol as the corresponding module. The parameter value of the modules connector1 and connector2 are both 3, meaning that they connect 3

terminals. All of the terminals of all of the modules are of electrical type, the resistors, capacitor, and inductor each have 2 terminals, and the connectors each have 3 terminals. We denote the 2 terminals of R_C by $R_{C,1}$ and $R_{C,2}$, the 3 terminals of connector1 by connector1_1, connector1_2, and connector1_3, and use a similar notation for the terminals of the other modules.

The interconnection architecture, shown in Figure 3(b), has six vertices, labeled $1,2,3,4,5,6$, six edges, labeled c,d,e,f,g,h, and two leaves, labeled a,b. The module embedding first requires that we associate a module with each vertex. For the example at hand, this association is given by

$$R_C \mapsto 2, R_L \mapsto 5, C \mapsto 4, L \mapsto 3,$$

$$\text{connector1} \mapsto 1, \text{connector2} \mapsto 6.$$

The module embedding also requires that for each vertex, we assign to each edge and leaf adjacent to a vertex, a terminal of the module associated to the vertex. For the RLC example, this assignment is given by

$$\begin{aligned}
\text{vertex } 1: &\ \text{connector}1_1 \mapsto a, \text{connector}1_2 \mapsto c,\\
&\ \text{connector}1_3 \mapsto d,\\
\text{vertex } 2: &\ R_{C,1} \mapsto c, R_{C,1} \mapsto e,\\
\text{vertex } 3: &\ L_1 \mapsto d, L_1 \mapsto f,\\
\text{vertex } 4: &\ C_1 \mapsto e, C_1 \mapsto g,\\
\text{vertex } 5: &\ R_{L,1} \mapsto f, R_{L,1} \mapsto h,\\
\text{vertex } 6: &\ \text{connector}2_1 \mapsto g, \text{connector}2_2 \mapsto h,\\
&\ \text{connector}2_3 \mapsto b.
\end{aligned}$$

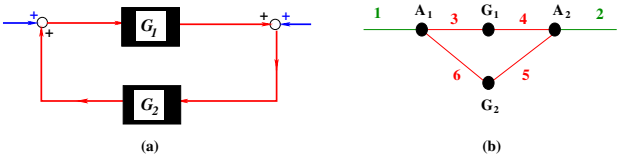

Fig. 4. Feedback system

The second example is the feedback system shown in Figure 4. The interconnection architecture is the graph with vertices A_1, A_2, G_1, G_2, edges $3,4,5,6$, and leaves $1,2$. The modules consist of two adders, associated with vertices A_1 and A_2, each with 2 inputs and 1 output, and two input/output systems, associated with vertices G_1 and G_2. The specification of the module embedding is obvious from Figure 4.

V. The Interconnection Constraints

The behavioral equations that govern an interconnected system combine module equations with interconnection constraints. We now explain how the interconnection constraints are obtained. The edges of the interconnection architecture specify how the terminals of the modules are linked. A module embedding guarantees that the terminals associated with the same edge are of the same physical type or of opposite logical type.

We postulate that there are universal rules, originating from the physical nature of the interconnections, that specify relations among the variables on the terminals that are linked. For instance, if an edge is electrical type, and hence connects two electrical terminals, the connection rule equates the voltages on the two terminals and equates the sum of the currents on the terminals to zero, where currents are counted positive when they run into a module. If the connected terminals are hydraulic, the connection rule equates the pressures and equates the sum of the mass flows to zero. If the terminals are logical, the connection rule equates the values of the associated input and the associated output.

The behavioral equations of the interconnected system are obtained as follows. For each vertex of the interconnection architecture, we obtain behavioral equations relating the variables that live on the terminals of the module associated with the vertex. These behavioral equations are the *module equations*. For each edge of the interconnection architecture, we obtain behavioral equations relating the variables that live on the terminals and that are linked by the edge. These behavioral equations are the *interconnection equations*, or *interconnection constraints*. Although no interconnection equation results from the leaves, the associated terminal variables nevertheless enter in the module equations.

The module equations and the interconnection equations together specify the behavior of all of the variables on all of the terminals involved. Note that each vertex of the interconnection graph is in the end labeled as a module, while each edge is labeled as a terminal of a specific type. We thus have systems in the vertices and interconnections in the edges, in contrast to, for example, conventional electrical circuit theory, which has modules in the edges, and interconnections in the vertices.

The interconnection equations are usually very simple. Typically they equate potential variables and equate the sum of flow variables to zero. We therefore think of interconnection as variable sharing.

VI. The Module and Interconnection Equations for the RLC Circuit

For the RLC circuit, the module equations involve the currents and voltages on the terminals of the module associated with each of the vertices. The voltage and current of terminal $R_{C,1}$ are denoted by $V_{R_{C,1}}$ and $I_{R_{C,1}}$, respectively, and a similar notation is used for the remaining terminals. The module equations are given by

$$\begin{aligned}
\text{vertex } 1: &\ V_{\text{connector}_1,1} = V_{\text{connector}_1,2} = V_{\text{connector}_1,3},\\
&\ I_{\text{connector}_1,1} + I_{\text{connector}_1,2} + I_{\text{connector}_1,3} = 0;\\
\text{vertex } 2: &\ V_{R_C,1} - V_{R_C,2} = R_C I_{R_C,1},\ I_{R_C,1} + I_{R_C,2} = 0;\\
\text{vertex } 3: &\ L\tfrac{d}{dt} I_{L,1} = V_{L,1} - V_{L,2},\ I_{L,1} + I_{L,2} = 0;\\
\text{vertex } 4: &\ C\tfrac{d}{dt}(V_{C,1} - V_{C,2}) = I_{C,1},\ I_{C,1} + I_{C,2} = 0;\\
\text{vertex } 5: &\ V_{R_L,1} - V_{R_L,2} = R_L I_{R_L,1},\ I_{R_L,1} + I_{R_L,2} = 0;\\
\text{vertex } 6: &\ V_{\text{connector}_2,1} = V_{\text{connector}_2,2} = V_{\text{connector}_2,3},\\
&\ I_{\text{connector}_2,1} + I_{\text{connector}_2,2} + I_{\text{connector}_2,3} = 0.
\end{aligned}$$

The module embedding for the RLC circuit implies that the pairs of terminals

$$\text{edge } c:\{R_{C,1},\text{connector1}_2\},\text{edge } d:\{L_1,\text{connector1}_3\},$$
$$\text{edge } e:\{R_{C,2},C_1\},\text{edge } f:\{L_2,R_{L,1}\},$$
$$\text{edge } g:\{C_2,\text{connector2}_1\},\text{edge } h:\{R_{L,2},\text{connector2}_2\}$$

share their terminal variables. The interconnection equations, given by

$$\begin{aligned}
&\text{edge c}: && V_{R_{C,1}} = V_{\text{connector1}_2},\ I_{R_{C,1}} + I_{\text{connector1,2}} = 0;\\
&\text{edge d}: && V_{L_1} = V_{\text{connector1}_3},\ I_{L_1} + I_{\text{connector1}_3} = 0;\\
&\text{edge e}: && V_{R_{C,2}} = V_{C_1},\ I_{R_{C,2}} + I_{C_1} = 0;\\
&\text{edge f}: && V_{L_2} = V_{R_{C,1}},\ I_{L_2} + I_{R_{L,1}} = 0;\\
&\text{edge g}: && V_{C_2} = V_{\text{connector2}_1},\ I_{C_2} + I_{\text{connector2}_1} = 0;\\
&\text{edge h}: && V_{R_{L,2}} = V_{\text{connector2}_2},\ I_{R_{L,2}} + I_{\text{connector2}_2} = 0,
\end{aligned}$$

equate the voltages of each of the connected terminals, and equate the sum of the currents to zero.

The module equations together with the interconnection constraints specify the behavior of the terminal variables.

For the feedback system example, we obtain, in the obvious notation, the module equations

$$\text{vertex } \mathrm{G}_1:(u_3,y_4)\in\mathscr{B}_{\mathrm{G}_1};\quad \text{vertex } \mathrm{G}_2:(u_5,y_6)\in\mathscr{B}_{\mathrm{G}_2};$$
$$\text{vertex } \mathrm{A}_1: y_2 = u_1+u_6;\quad \text{vertex } \mathrm{A}_2: y_4 = u_2+u_4.$$

Here $\mathscr{B}_{\mathrm{G}_1}$ and $\mathscr{B}_{\mathrm{G}_2}$ denote, respectively, the behavior of the input/output systems in the forward loop and the feedback loop of the feedback system. The interconnection equations are given by

$$\text{edge 3: } y_3 = u_3;\ \text{edge 4: } y_4 = u_4;$$
$$\text{edge 5: } y_5 = u_5;\ \text{edge 6: } y_6 = u_6.$$

VII. The Manifest Variable Assignment

The final step of the modeling procedure consists of the *manifest variable assignment*, a map that assigns the manifest variables as a function of the terminal variables. The terminal variables are henceforth considered as latent variables.

For the RLC circuit the manifest variable assignment consists of the specification

$$V_{\text{externalport}} = V_{\text{connector}_1,1} - V_{\text{connector}_2,3},\ I_{\text{externalport}} = I_{\text{connector1}_1}$$

of the external port voltage and port current in terms of the terminal variables. It is easy to deduce from the behavioral equations obtained in the section 'The Module and Interconnection Equations for the RLC Circuit' that the equations imply that $I_{\text{connector1}_1} = -I_{\text{connector2,3}}$. In words, the current that flows into the circuit through terminal connector1_1 flows out of the circuit through terminal connector2_3. In many circuit theory applications, modeling aims at obtaining equations of the voltage across a port and the current that flows into a port. For the feedback system, the manifest variable assignment consists of

$$u_{\text{external}} = (u_1,u_2),\quad y_{\text{external}} = (y_6,y_5).$$

The module equations, combined with the interconnection constraints and the manifest variable assignment, define the full behavior. These equations contain many latent variables — in fact, all of the terminal variables are latent variables — in addition to the manifest variables the model aims at. This model is the end result of the modeling process based on tearing (the interconnection architecture), zooming (leading to the module equations and manifest variable assignment), and linking (leading to the interconnection constraints).

VIII. Concluding remarks

The tearing, zooming, and linking modeling methodology is systematic, modular, adaptable to computer-assisted implementation with the module equations in parametric form and the interconnection equations stored in a database, and hierarchical, since a model of an interconnected systems can be used as a module on a higher level. A model library supporting this methodology is thus re-useable, extendable, modifiable, and flexible. A disadvantage of this methodology is that the model equations involve many variables. This drawback can be alleviated by eliminating variables when possible. The interconnection equations, for example, allow the elimination of many of the variables.

The philosophy of tearing, zooming, and linking is to keep the interconnections highly standardized and simple, and to deal with complex features of a model by means of modules. For instance, in the circuit example, a multi-terminal connector is viewed as a module, rather than as a connection. In mechanical systems, joints, hooks, and hinges are viewed as modules, rather than as connections. The variable-sharing approach of tearing, zooming, and linking formalizes the modeling practice followed in computer-assisted modeling packages such as Spice and Modelica, in contrast to Matlab's output-to-input assignment-based, and therefore limited, Simulink.

Acknowledgments The SISTA-SMC research program is supported by the Research Council KUL: GOA AMBioRICS, CoE EF/05/006 Optimization in Engineering (OPTEC), IOF-SCORES4CHEM, several PhD/postdoc and fellow grants; by the Flemish Government: FWO: PhD/postdoc grants, projects G.0452.04 (new quantum algorithms), G.0499.04 (Statistics), G.0211.05 (Nonlinear), G.0226.06 (cooperative systems and optimization), G.0321.06 (Tensors), G.0302.07 (SVM/Kernel, research communities (ICCoS, ANMMM, MLDM); and IWT: PhD Grants, McKnow-E, Eureka-Flite; by the Belgian Federal Science Policy Office: IUAP P6/04 (DYSCO, Dynamical systems, control and optimization, 2007-2011) ; and by the EU: ERNSI.

References

[1] T. Cotroneo and J.C. Willems, The simulation problem for high order differential equations, *Applied Mathematics and Computation*, volume 145, pages 821–851, 2003.

[2] J.W. Polderman and J.C. Willems, *Introduction to Mathematical Systems Theory: A Behavioral Approach*, Springer-Verlag, 1998.

[3] J.C. Willems, The behavioral approach to open and interconnected systems, Modeling by tearing, zooming, and linking, *Control Systems Magazine*, volume 27, number 6, December 2007.

Parametric Identification of Plants with Multiple Delays and Internal Feedbacks using Genetic Algorithm

N. Salkanović, B. Lačević, B. Peruničić
Faculty of Electrical Engineering
University of Sarajevo
Sarajevo, Bosnia & Herzegovina
brana_p@hotmail.com

Ž. Jurić
Faculty of Science, Department of Mathematics
University of Sarajevo
Sarajevo, Bosnia & Herzegovina
zjuric@utic.net.ba

Abstract — **This paper proposes a procedure for parametric identification of plants having multiple sources of the pure time delays and internal feedbacks, which are extremely complicated for the identification. The procedure is based on a genetic algorithm applied to the samples of the frequency response on the plant.**

Identification; time delay; feedback; genetic algorithm

I. INTRODUCTION

The experimental identification of the transfer function of unknown plant is a very important task, because most of the control design procedures rely on some knowledge of the plant transfer functions. Strictly speaking, the transfer function is defined only for linear plants. However, controllers for many standard industrial plants are designed using linear approximation of the plant dynamics.

In general, all identification methods may be roughly divided into the time-response methods and the frequency-response methods. The time-response methods are mostly simple open loop methods, performed in a short time. They are considered adequate only for simple plant models, and they usually require the isolation of the plant from the control loop. Frequency response methods are based on finding the samples of the frequency response of the plant. The points of the plant's Nyquist plot, which are typical input data for frequency response identification methods, are obtained either using special signal generators in open loop, or using forced oscillations in the closed-loop caused by appropriate controller settings. The well-known Ziegler-Nichols (ZN) method for PID controller tuning [1] may be regarded as the simplest closed loop frequency-response method, as it detects one point on the Nyquist plot of the plant. More advanced frequency-response methods give more information about the plant, but they usually require much longer time.

For some control design methods, knowledge of a few characteristic points on the Nyquist curve is all that is needed for design. However, many controller design methods require full knowledge of the transfer function, at least in some approximate form. Therefore, it is often necessary to obtain some model of the unknown transfer function based on experimental data. This task is usually accomplished using parametric identification, where some model structure is chosen in advance. Then, the problem of identification is reduced to the problem of finding parameters such that the chosen model structure best fits the experimental data, according to some pre-selected criteria. One quite general method for such identification under the assumption that the plant transfer function is rational is given by Levy [2]. Some improvements of Levy's approach in cases when the plant contains finite zeros are given in [3].

The dynamic of nearly all of the industrial plants contain pure time delay. There are a lot of procedures for the identification of various models that contain pure time delay. One detailed comparison of such procedures is given in [4]. There is also a generalization of Levy's approach for the plants that contain pure time delay [5]. However, all such procedures are based on the supposition that the plant transfer function may be modeled as a product of a rational function of s that describes the inertial part of the plant dynamics, and the transcendental factor $e^{-\tau s}$ that describes the pure delay. Such assumption is valid only if there are no internal feedbacks in the plant that have a pure delay in the loop. In the presence of any such feedback, the transfer function of the plant is not a product of one rational and one simple transcendental factor. Such transfer functions are very complex, as they have an infinite number of poles, and sometimes also an infinite number of finite zeros [6]. The situation is even more complex if there is more than one source of pure time delay in the plant. The overall transfer function of such plant then depends on delay terms like $e^{-\tau s}$ with different values of τ. Finding exact values of the plant parameters seems to be extremely difficult. Until now no systematic procedure for the identification of such plants is known. This paper proposes an approach based on a genetic algorithm that may be useful for the approximate identification of parameters of such plants.

The identification procedure proposed in this paper uses the values of the transfer function for some known values of s, called samples in the further text, as the input data. For the application of the procedure, it is completely irrelevant how these samples are obtained. Such samples may be a set of points on the Nyquist plot of the plant, which may be obtained

978-1-4244-1688-2/08/$25.00