Katholieke Universiteit Leuven

Departement Elektrotechniek

ESAT-SISTA/TR 1997-36

BondLab, a Toolbox for Integrated Mechatronic Design

Willem Minten, Sven Vranckx, Bart De Moor, and Joos Vandewalle[1]

Accepted for ESM'97

[1]ESAT (SISTA) - Katholieke Universiteit Leuven, Kardinaal Mercierlaan 94, 3001 Leuven (Heverlee), Belgium, Tel. 32/16/32 17 09, Fax 32/16/32 19 70, WWW: *http://www.esat.kuleuven.ac.be/sista.* E-mail: {Willem.Minten,Bart.Demoor,Joos.Vandewalle}@esat.kuleuven.ac.be.
Willem Minten is Research Assistant, Sven Vranckx is Graduate Student, Bart De Moor is a senior Research Associate with the F.W.O. (Fund for Scientific Research-Flanders), and Joos Vandewalle is Full Professor, all at ESAT.
This work is supported by several institutions: the Flemish Government (Concerted Research Action GOA-MIPS, the FWO (Fund for Scientific Research - Flanders: G.0292.95, G.0256.97), the FWO Research Communities (ICCoS, Advanced Numerical Methods for Mathematical Modeling), the IWT (Flemish Institute for Science and Technology in Industry: IWT/VCST project (1992-1997), IWT Action Program on Information Technology (ITA/GBO/T23) - IT-Generic Basic Research: ISIS)), the Belgian Government (Inter-university Attraction Poles: IUAP P4-02 (1997-2001) and IUAP P4-24 (1997-2001)), and the European Commission (Human Capital and Mobility Network SIMONET; SCIENCE-ERNSI SC1-CT92-0779; TMR Network ALAPEDES; Keep In Touch SYSIDENT-KIT124).
The scientific responsibility is assumed by its authors.

BONDLAB,
A TOOLBOX FOR INTEGRATED MECHATRONIC DESIGN

Willem Minten, Sven Vranckx, Bart De Moor, and Joos Vandewalle
ESAT - Katholieke Universiteit Leuven
Kardinaal Mercierlaan 94, 3001 Leuven (Heverlee), Belgium
E-mail: {Willem.Minten,Bart.Demoor,Joos.Vandewalle}@esat.kuleuven.ac.be

KEYWORDS
bond graph methodology, physical systems modeling, software, interactive programs, object oriented models.

ABSTRACT

A prototype of a new mechahytronic design environment, named *BondLab*, is proposed. This environment offers the ability to exploit both the behavioral and causal model description languages during a design. Since MatLab is a de-facto standard in the treatment of the causal models of a design (in e.g. parameter optimization and control objectives) BondLab is implemented as an add-on to MatLab. And since Bond Graphs can cover the behavioral part of the modeling of physical systems, the design environment is adapted to these graphs. Some implementation aspects and current features are described shortly.

INTRODUCTION

The design of mechahytronic products is characterized by an attempt to consider simultaneously the mechanical, hydraulic, electrical/electronic, and control aspects of the final product during the design cycle. Today the importance of the first step of the design cycle, namely the modeling phase, is largely underestimated. This is due to the extensive use of 'causal' model descriptions. Such causal models are necessary at the end of the design, where control and implementation aspects pop up. But on the other hand they are not adapted to fulfill the tasks to be done in the beginning of a design. Bond Graphs [Karnopp *et al.* 1990] belong to the family of the 'behavioral' model description languages, and are very well (but not solely) suited to describe physical processes. Since the translation of a Bond Graph model to a causal model can be automated it is a good modeling tool candidate in the design of mechahytronic products.
In this paper a prototype of a new mechahytronic design environment, named *BondLab* (an acronym of *Bond* Graph *Lab*oratory), is proposed. This environment facilitates the use of both the behavioral and causal model description languages during a design.
The need for such an integrated design environment came up some years ago [Minten *et al.* 1993]. Since even now there exists no excellent *integrated* software dealing with (behavioral) modeling, simulation analysis and design together, the need for such a tool was, and still is high. During that time a lot of useful stand-alone software packages (e.g. CAMAS -and his successor 20SIM-, MS1, ...) became available. However they lack integration of all the aspects of a mechahytronic design[1].
During that time the first version of BondLab was created in MatLab4. Due to the lack of rich data structures, symbolic formula manipulation facilities, and object orientation this task was very cumbersome. With the advent of the Symbolic Math toolbox and release 5 of MatLab the software development became much more transparant. This paper is organized as follows: In Section 2 some implementation aspects of BondLab are covered. In Section 3 some current features are illustrated with an example. In the last Section some conclusions are made.

[1]In late releases of these packages they aim some port to MatLab. However it is left for the designer to verify the statement that the embedding of a behavioral modeling tool within MatLab is more preferable than stand-alone software.

SOME IMPLEMENTATION ASPECTS

Two constraints are focussed while embedding BondLab in MatLab: *fast* and *powerful* code. The term 'powerful' is used for specifications like transparency, compactness and extensibility. Since these aspects have influenced a lot the implementation, they are outlined short in the sequel.

Powerful code

The introduction of enhanced data types in MatLab5 provides structural means for constructing a proper database in BondLab. Especially the `structure` and the `cell` seemed to be very useful for the construction of transparent, extensible and compact code. Also the new possibility of object orientation is very powerful and improved the readability of the code. and hence it has been exploited in BondLab also. However it must be said that the MatLab implementation of object properties like *inheritance* differ considerably of the common usage like in e.g. C++. This difference is shown in Table 1.

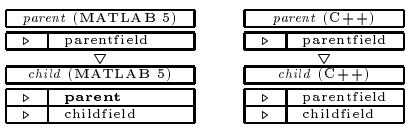

Table 1: the *parent*- and *child*-object type in MatLab5 and C++

A method of the *parent*-type has in C++ only access to the *parentfield*. When a *child*-object of a *parent*-type is passed to such a method, then the *childfield* is invisible such that the *child*-object behaves just like a *parent*-object. The MatLab implementation however is very awkward: a *child*-object has for a *parent*-method two fields: **parent** and *childfield* (this last one is only *visible*, it can not be manipulated). Consequently, the *parent*-method has to change its behavior with regard to the objects on which it is used. Especially this difference has influenced the implementation and use of objects in BondLab: BondLab objects are not inferred by a parent object, they create a parent object. And both the parent object and child object have a pointer to each other. This can be shown in Figure 3 *(b)*.

Speed

An important feature of BondLab is its user-friendly graphical interface. The user interface is designed corresponding the basic guidelines described in [Preece *et al.*1994] and [Tognazzini 1991]. The basic graphical manipulations like *move, copy, paste, cut, flip horizontal, flip vertical, rotate* are provided in such a way that minimal key strokes or mouse manipulations are needed.

In order to be *interactive*, not only the code but also the data structures have to be optimized with regard to speed. It is well known that MatLab code can be speed up by *vectorization* of loops or algorithms. This vectorization treats whole rows of data with a single build-in function. It has been found that inheritance must be abandoned in favor of speed optimization of the code. The most important reason for this is that different types of objects cannot be placed in one row on which vectorization can be applied.

Two-layer implementation of the Bond Graph database

The BondLab database has been implemented as two layers. Figure 3 *(b)* shows these layers containing the objects of the Bond Graph shown on top of Figure 3 (a), `[db_example]`. The graphical objects (nodes and bonds) are placed above the dashed line, the Bond Graph objects (0, 1, R, C, I, TF GY) are placed below of it. These objects are all implemented as cells. This layer implementation is beneficial in manipulation of *either* the graph itself *or* the Bond Graph technical aspects, which in most cases is a separate event.
All the objects are stored in the first field of the structure array pool, `pool.data` (see Figure 3 *(c)*). In order to improve the search speed to the objects in `pool.data`, the other pool fields serve as an interface on which optimized code is used. Beside this single Bond Graph database, every window is managed by its own *window data base* `db(i)`. This is shown in Figure 3 *(a)*. This database is also implemented as a structure array. It contains information about e.g. the grid (size, visibility) and the selected graph objects in the respective window.

CURRENT FEATURES

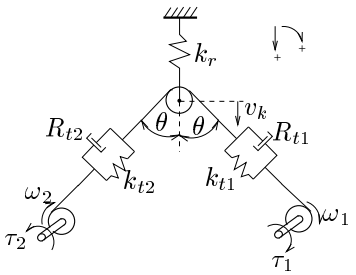

Figure 1: Bond Graph of two coupled drives, the drive side is located on the right hand side.

Current features of BondLab include (1) a highly perceptive user friendly graphical interface, (2) an easy to use model entry drawing environment, (3) a unifying treatment of both linear and nonlinear models, (4) the ability to answer already some basic questions on the model quality, (5) several smooth transitions between the behavioral and causal model descriptions (Transfer Fcn, State Space, Simulink2 .mdl block diagram format), and (6) contains also direct simulation, visualization and animation facilities. Some of them will be shown through an example, of which in Figure 1 the physical model is shown [Minten *et al.* 1996b] [Minten *et al.* 1997].
Figure 2 (top) shows e.g. the entry of a transformer in the main window of BondLab. In order to minimize the manipulations by the user, several push-buttons are provided below the pull-down menus. Keyboard shortcuts for creation an modification purposes are also available. Below the graph there is a possibility to enter or change the most important properties of the selected element. Using a properties window (Figure 2 (bottom)) it is possible to have direct access to the whole database.
While (or after) drawing the Bond Graph, parameter labels and values can be entered, and a simplification routine can be altered to remove e.g. redundant junction structure constructs. Thereafter a consistency check can modify questionable Bond

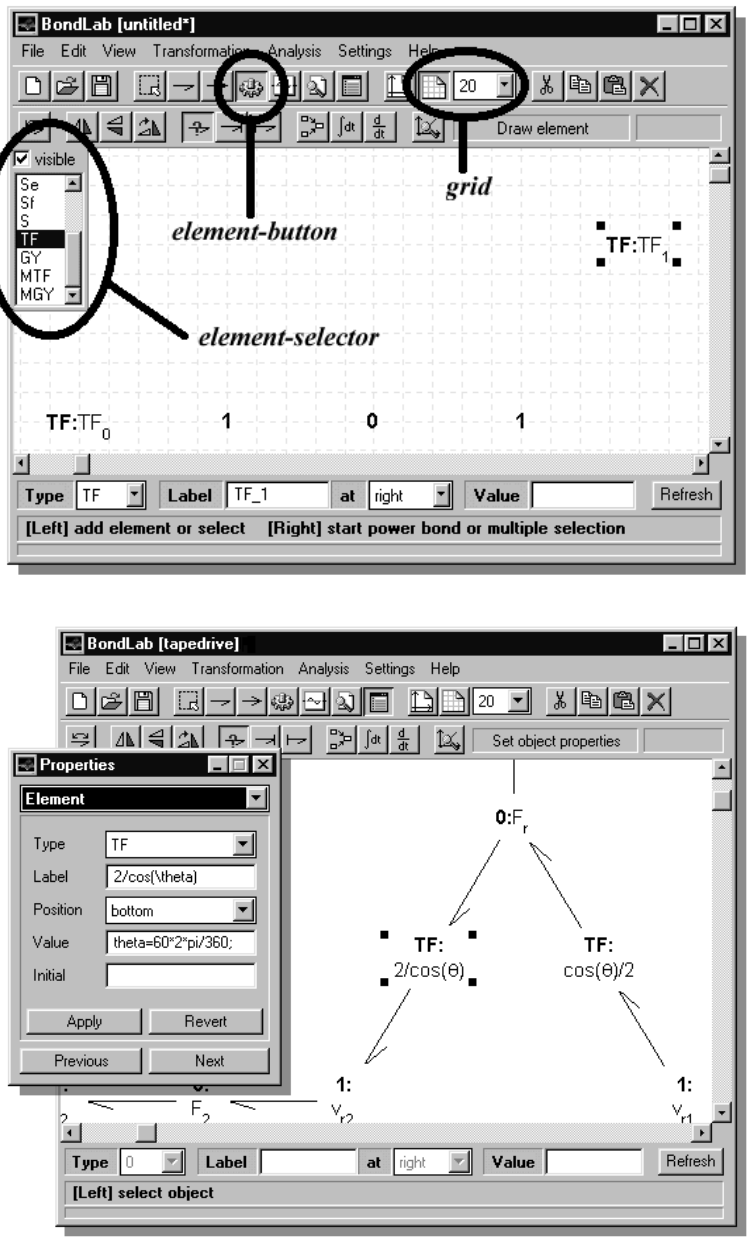

Figure 2: Main window of BondLab. (top) element-*mode, (bottom) properties mode.*

Graph parts [Minten *et al.* 1996a] and remove not connected elements. Thereafter, the causal assignment routines assigns the element equations, and a symbolic or numerical output can be saved to the MatLab workspace. This last can be done from the *Transformation* tab in the *Analysis* window, shown in Figure 4 (top).
This output can be a (non-)linear state space form or transfer matrix form, shown in Table 2.
Remark that all this is integrated in MatLab in a flexible way. E.g. LaTeX code can accompany the element labels resulting in very readable graphs[2]. Also nonlinear behavior can be modeled in several ways, since MatLab code can be entered right in the value field of the elements. Even user defined `.m` function files can be used in the same straightforward manner.
BondLab also provides internal facilities for direct numerical simulation. Therefore the set of ODE's can be simulated from within the *Simulation* tab of the *Analysis* window, shown in

[2]The LaTeX mark up code is removed automatically when a symbolic form of the equations is desired.

$$\begin{bmatrix} \dot{p}_3 \\ \dot{p}_7 \\ \dot{q}_{11} \\ \dot{q}_{12} \\ \dot{q}_{13} \end{bmatrix} = \begin{bmatrix} -\frac{R2}{J2} & 0 & 0 & 0 & \frac{kr\ \cos(th)\, r2}{2} \\ 0 & -\frac{R1}{J1} & 0 & 0 & -\frac{kr\ \cos(th)\, r1}{2} \\ 0 & 0 & -\frac{k_t2}{R_t2} & 0 & \frac{kr\ \cos(th)}{2\ R_t2} \\ 0 & 0 & 0 & -\frac{k_t1}{R_t1} & \frac{kr\ \cos(th)}{2\ R_t1} \\ -\frac{r2\ \cos(th)}{2\ J2} & \frac{r1\ \cos(th)}{2\ J1} & \frac{k_t2\ \cos(th)}{2\ R_t2} & \frac{k_t1\ \cos(th)}{2\ R_t1} & -\frac{kr\,(\cos(th))^2}{4}\left(\frac{1}{R_t2}+\frac{1}{R_t1}\right) \end{bmatrix} \cdot \begin{bmatrix} p_3 \\ p_7 \\ q_{11} \\ q_{12} \\ q_{13} \end{bmatrix} + \begin{bmatrix} -1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} T_2 \\ T_1 \end{bmatrix}$$

$$\begin{bmatrix} e_9 \\ f_9 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1/2\, kr\ \cos(th) \\ 0 & \frac{r1}{J1} & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} p_3 \\ p_7 \\ q_{11} \\ q_{12} \\ q_{13} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} T_2 \\ T_1 \end{bmatrix}$$

$$\frac{f_9}{T_1} = \frac{num(s^4)}{den(s^5)} = \frac{51/100 * (s+10) * (544 * s^3 + 6522 * s^2 + 10742 * s + 9945)}{17/400 * (s+10) * (4 * s + 7) * (1088 * s^3 + 13044 * s^2 + 23473 * s + 39780)}$$

Table 2: state space form (symbolic) and transfer function description (numerical) of the Bond Graph of Figure 1

Figure 4 (middle). If unavoidable derivative causalities occur, or if some not solvable nonlinear algebraic loops appear, then BondLab can automatically create a SimuLink `.mdl` block diagram, which consequently can be simulated also.
At last some initial model reduction tools, based on the power distribution in the bond graph are available [Minten *et al.* 1996b]. This is especially useful for the reduction of nonlinear systems, in which the nonlinear behavior in time should be evaluated. In Figure 4 (bottom) a snap-shot of such an animation based analysis is shown. In this example it turned out that 96% of the power-picture remains the same when the tape dynamics are neglected. This observation makes it possible to reduce the bond graph from a fifth order to a third order system, while maintaining the most important dynamics. For this model it turns out that the fastest poles are omitted in this reduction step. With classical methods (i.e. by means of singular value or eigenvalue decomposition) it is very hard to achieve such generic results[3].

CONCLUSIONS

In this paper a Bond Graph oriented graphical user interface to MatLab5 has been proposed. The data structure has been created with regard to code transparency, available data structures, and speed considerations. Especially for this last constraint it turns out that a pure object oriented solution seems at this time not applicable in MatLab5.
This paper also presents some actual features of *BondLab*: (1) the support of the design of the (bond graph) model by means of creation and reduction tools, (2) the analysis of the model quality which covers basic model adaption facilities (like energy related methods), and (3) the automated model transformation of (non-)linear bond graphs to a mathematical more tractable form (like a (symbolic or numerical) transfer matrix, nonlinear state-space form or SimuLink .mdl block diagram) useful for several numeric or symbolic manipulations. Future developments of this experimental design environment include the implementation of substructures, enhanced block diagram support within BondLab, and the ability of having more bond graph related design objectives in the framework of mechatronic design.

ACKNOWLEDGMENTS

This work is supported by several institutions: the Flemish Government (Concerted Research Action GOA-MIPS, the FWO (Fund for Scientific Research - Flanders: G.0292.95, G.0256.97), the FWO Research Communities (ICCoS, Advanced Numerical Methods for Mathematical Modeling), the IWT (Flemish Institute for Science and Technology in Industry: IWT/VCST project (1992-1997), IWT Action Program on Information Technology (ITA/GBO/T23) - IT-Generic Basic Research: ISIS)), the Belgian Government (Inter-university Attraction Poles: IUAP P4-02 (1997-2001) and IUAP P4-24 (1997-2001)), and the European Commission (Human Capital and Mobility Network SIMONET; SCIENCE-ERNSI SC1-CT92-0779; TMR Network ALAPEDES; Keep In Touch SYSIDENT-KIT124). The scientific responsibility is assumed by its authors.

[3]The problems that occur can be cast back to the Galois theory (polynomial root finding), but is not subject of interest here.

REFERENCES

[Karnopp *et al.* 1990] Karnopp D.C., Margolis D.L., and Rosenberg R.C. 1990, *System dynamics: a unified approach. Second edition.*, John Wiley & Sons, New York, USA, (first print in 1975).

[Minten *et al.* 1993] Minten W., Vanvuchelen P., De Moor B., "Modeling of an electronically driven continuously variable transmission", *Proc. of the 1993 International Conference on Bond Graph Modeling and Simulation (ICBGM'93)*, San Diego, California, Jan. 1993, pp. 223-228.

[Minten *et al.* 1996a] Minten W., De Moor B., and Vandewalle J. 1996 "How Can Different Interpretations and Use of Bond Graph Parameters be Avoided.", *J. Franklin Institute*, Vol334b, No3, pp.389-406, 1997.

[Minten *et al.* 1996b] Minten W., De Moor B., and Vandewalle J. "Power and Power Variables in Graphical Modeling of Physical Dynamic Systems", ESAT-SISTA, TR96-92, 1996, K.U.Leuven, Belgium.

[Minten *et al.* 1997] Minten W., De Moor B., Vandewalle J., "Condensed bond graph modeling", *Proc. of the 3rd International Conference on Bond Graph Modeling and Simulation, Part of the WMC'97 Multiconference (ICBGM'97)*, Phoenix, Arizona, Jan. 1997, pp. 50-55.

[Preece *et al.*1994] , Preece J., Keller L., *Eds*, *Human-Computer Interaction*, Prentice Hall, UK, 1994.

[Tognazzini 1991] Tognazzini B., *Tog on Interface*, Addison Wesley, New York, USA, 1991.

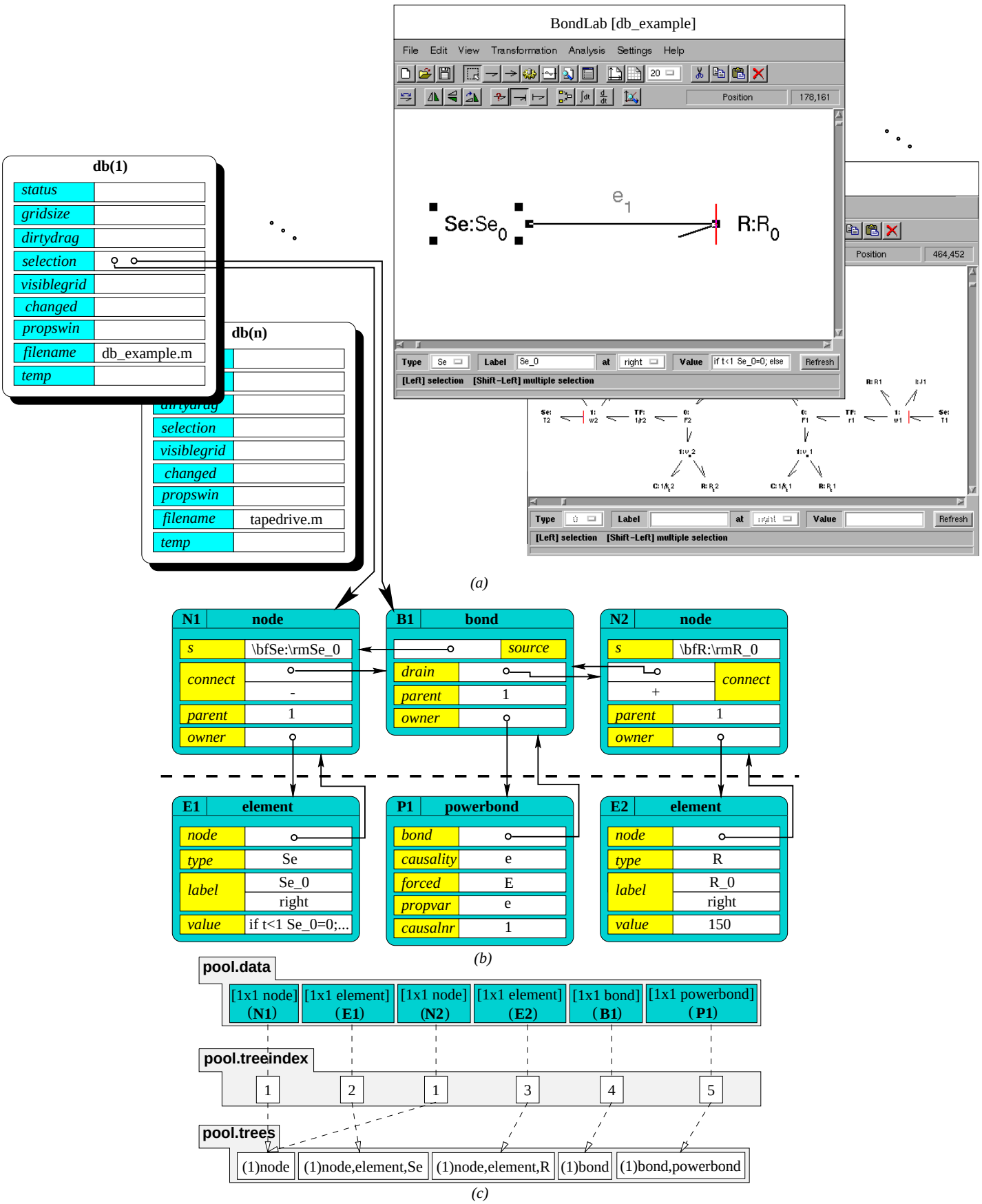

Figure 3: Internal structure of BondLab. (a) *Window manager databases,* (b) *partial view on the cells in* `pool.data`, (c) *view on the* `pool`. *The arrows visualize pointers.*

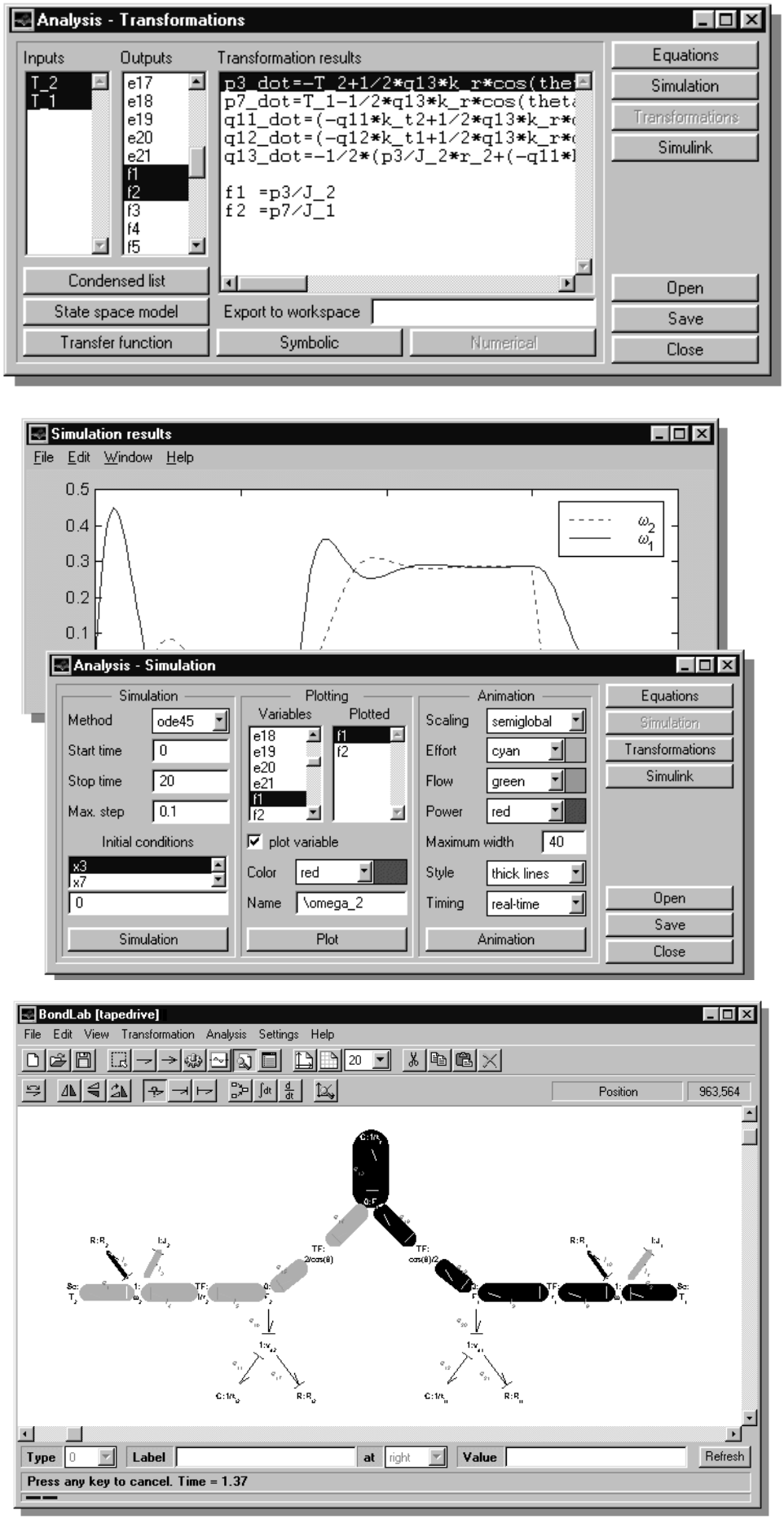

Figure 4: Analysis window. (top) Transformation tab: symbolic manipulation of the causal bond graph. (middle) Simulation tab and view on the plot window. (bottom) Snap-shot of the animation of the Bond Graph (thick-lines-mode).