Appeared in: Working Papers of DX-95: Sixth International Workshop on Principles of Diagnosis, Goslar, Germany, October 2-4, 1995, pp. 73-80, 1995

Diagnosis by Algebraic Modeling and Fault-Tree Induction

Jakob Mauss
DAIMLER-BENZ AG, Forschung F1M/EK
D-70546 Stuttgart, Tel.: +49 711 17 41247, Fax: +49 711 17 41717
and
Bernd Neumann
Labor für Künstliche Intelligenz (LKI)
Vogt-Koelln-Str 30, 22527 Hamburg
E-Mail: {mauss,neumann}@informatik.uni-hamburg.de

Abstract

We outline relevant characteristics of the vehicle diagnosis domain and requirements for diagnosis support. We argue that a combination of a model-based and a fault-tree approach will meet the requirements. We motivate and specify the task of automated modeling and develop a method to derive linear algebraic models of a given device. Models are derived by performing series-parallel analysis and by applying Cramer's Rule in a tractable way in cases when sp-analysis fails. The models are used to predict observations under arbitrary multiple-fault assumptions as a basis for fault-tree induction. The models can also be used for purely model-based diagnosis. We sketch how to derive a fault dictionary from a model and how to induce a fault tree with an ID3-like algorithm.

1 Introduction

The research presented in this paper was conducted at a research division of a leading car manufacturer designing a knowledge-based system for car diagnosis. The knowledge bases should cover all faults of the electrical, electronic, mechanical, and hydraulic components of a car usually encountered in the workshop, such that the system can guide the workshop technician to the faulty component. We will first outline some relevant characteristics of our domain:

C1 Hybrid devices with feedback loops A car is a hybrid device containing electrical, electronic, mechanical, hydraulic and pneumatic components. Often, the functionality of the device is achieved by feedback coupling of many such components.

C2 Software-controlled devices A considerable portion of the behaviour relevant for the diagnosis of the device is coded in the software of the electronic control units controlling e.g. the motor, or the automatic transmission.

C3 Limited observability of physical quantities Many of the quantities predictable by purely physical (i.e. not functional) models of the device like torque, numbers of revolution, or hyraulic volume flow are not and cannot be measured in the workshop. Instead, the technician observes the overall behaviour (function) of a device, e.g. whether the up-shift from position Neutral to Drive works properly, whether the engine sounds fine, and so on.

C4 Non-obvious tests As a consequence of C3, the technician has to perform smart tests appropriate to detect the fault modes of the components. The design of such tests (experiments) often requires subtle knowledge about the behaviour of the device as well as common-sense understanding of safety conditions and manageability in the workshop.

C5 Non-physical knowledge Developers of the handbooks used by technicians in today's workshops take into account a "model" of the technician: They ascribe certain skills to him, try not to bore him with too detailed instructions while supporting him with helpful explanations and pictures when things get complicated. Besides this model of the technician, a developer uses other non-technical sources of knowledge to design a diagnostic strategy. For example, the costs of a repair are charged to the customer, to the workshop or to the car manufacturer, depending on the outcome of the diagnosis. This outcome depends on the diagnostic strategy applied in the workshop and hence can be influenced by the diagnostic expert.

C6 Broad coverage of faults Any approach proposed to assist the workshop technician should not only provide a framework for some selected but for all relevant faults. Here, a fault is considered to be relevant, if it occurs in the workshop with a reasonable probability. Under this definition, single faults are more relevant than most multiple faults.

C7 Uncertain and changing knowledge Knowledge acquisition (for paper handbooks as well as for knowledge-based diagnosis systems) has to be performed during the development of the device. Any method for knowledge acquisition should thus account for a situation with uncertain or changing information.

C8 Economy Knowledge acquisition (for paper handbooks as well as for knowledge-based diagnosis systems) has to meet certain economic constraints: If knowledge acquisition with method A takes one man year and with method B 10 man years, method A might be preferable for economic reasons, even if method B results in a broader coverage of faults.

C9 Automated Modeling If a knowledge acquisition method requires explicit models of the device, the processing of a model has to be completely automated: A developer will not accept a tool that asks him e.g. to solve $x = f(x)$, to do some ordering of equations, or to add missing energy constraints to a model. Also, the developer will refuse to model a device by e.g. functional decomposition, or by specifying different levels of abstraction, or by applying any similar sophisticated modeling technique known in the fields of AI or simulation. Ideally, from a developer's point of view, a device should be modeled by a simple device diagram and very little more.

Given these characteristics we conclude that a pure model based approach will not meet all the requirements: C1 ... C6 imply that we will need rich models, which almost in turn conflicts with C9. C9 forces us to restrict ourselves to models that can be derived and processed automatically. On the other hand, C7 and C8 give us a strong recommendation to use models compatible with C9 whenever possible.

Matching the classical fault-tree approach against the above characteristics we conclude: C9 is matched perfectly since the approach uses no models at all. C1 ... C5 are matched as with a fault tree we can "code" arbitrary, in particular non-physical diagnostic knowledge. C6 is matched if we assume that only single faults and very few multiple faults known in advance are relevant[1]. C7 and C8 recommend strongly not to use fault trees.

The above scorings are nearly complementary. Therefore we try to combine the two approaches in order to end up with an approach matching all the given requirements.

We combine the two approaches by deriving (see C8) portions of the fault tree of a device from simple (see C9) device diagrams. Where this cannot be done because the diagnostic knowledge needed cannot be derived from such a diagram (C1, C2, C4, C5), the diagnostic expert has to complete the fault tree by hand. A simple truth maintenance mechanism not described in this paper couples derived portions of the fault tree with the diagrams used, such that a certain class of changes of the diagram will cause an automatical update of the fault tree (C7, C8). For several reasons (C3, C4, C5) the diagnostic expert needs explicit control over the test sequences occuring later in the workshop. Post-editing a derived fault tree gives him this freedom in a way that could hardly be achieved with a purely model-based approach.

The rest of the paper is organized as follows: In the second section we describe our modeling technique, which represents the main contribution of this paper. In the third section we shortly sketch the derivation of fault trees from automatically derived models with an algorithm similar to Quinlan's ID3 [Quin 86].

[1]The technical diagnosis and repair handbooks used in our domain today actually only treat a fraction of all single faults, mainly because of the costs (C8) of producing and maintaining more sophisticated handbooks. Thus an exhaustive, automated treatment of the "trivial" single fault case could already be regarded as a substantial improvement. Recall also that the probabilities for independent coincidence of single faults multiply, i.e. multiple faults are, while probably hard to locate, in general much less likely, than single faults.

2 Automatic modeling for workshop diagnosis

A key observation is that in the workshop scenario considered here - car diagnosis - measurements are often performed in steady states of the faulty device. Voltage and resistance are measured in steady states (i.e. in states achieved e.g. by opening and closing certain switches and plugs, such that the observed values do not vary in time). The same holds for hydraulic and pneumatic pressure measurements, motoroil level, number of revolutions of an engine in idle state etc. Such steady states can often be modeled by resistive networks.

Because of the physical parallelism of voltage, pressure, force, torque on the one hand, and current, volume flow, velocity, and angular velocity on the other[2], hydraulic, pneumatic, and mechanical devices can be modeled by resistive networks as well. However, the values predicted by non-electrical models are often by far more difficult to observe in the workshop (C3). In the electrical case, the instruments to measure voltage and resistance[3] are at hand in every workshop today. Table 1 outlines the parallels in the different physical domains[4]. In the following we will focus on electrical networks although our techniques also apply to the other physical domains.

electrics	**hydraulics**	**translational mechanics**	**rotational mechanics**
voltage u	pressure p	force F	torque τ
current i	flow Q	velocity V	angular velocity ω
battery	oil pump	source of force	motor
wire	pressure line	rack	shaft
resistance lightbulb coil	oil filter	damper	bearing with friction
switch			pawl
relay	shift valve		
diode	check valve		freewheel
Z-diode			clutch
transformer	pressure line with two different cross sections	lever mechanics	gear wheel mechanics

Table 1: Analogies in different physical domains

2.1 Automatic modeling: specifying the task

A *device diagram* is a graphical representation of the device made for humans (see e.g. Fig. 1). A diagram is composed by a human fault-tree developer by choosing and connecting predefined component diagrams. A *component diagram* defines a graphic representation of a component as well as a set of quantities representing observable facts under given fault assumptions. Using the diagram, the developer states also, which of the quantities are actually observable in the workshop, what preconditions have to be satisfied (in terms of propositions about quantities) before an observation can take place and at what expences this can be done. A *model* is a function, derived completely automatically (C9) from the diagram that is able to predict observable quantities under given fault assumptions. So a model can be regarded as a function f with

observed value = f(fault assumption, observable quantity)

Automatic modeling is the derivation of such a computable function from a given device diagram. A *fault assumption* assigns a mode of behaviour to each component. The function f or model can be evaluated to fill a *fault dictionary*, also called *fault relation* here. This relation correlates the observations possible in the workshop with possible component faults. You might think of the relation as implemented by a table. Models are one of the possible sources of entries in that table.

[2]Such pairs of variables are generalized in the bondgraph formalism by an abstract pair of variables, called effort e and flow f, where the product e f is constrained to yield power, cf. [KaMaRo90].
[3]Current is measured only in rare cases during a car repair in a workshop.
[4]Regarding the analogy of Z-diode and clutch: The Z-voltage at which the resistance of the diode becomes zero parallels the maximal torque that can be transmitted by the clutch before the clutch starts slipping.

2.2 Example for the basic modeling technique

Fig. 1 shows a diagram of our examplary device. The device consists of a series connection of battery B, switch S, wire W and lightbulb L. There are 3 points accessible for measurements labeled 0 (= GND), 1 and 2.

Let us assume, the developer has further stated that voltages and resistances, but no currents are observable between arbitrary pairs of the 3 measurement points. Additionally, he has declared an experimental change of the lightbulb as a possible test (test is repair). The costs for measurements were defined as follows: There are 4 classes. Observing, whether the light is on or off is the simplest, followed by measurements involving points 0 and 2 only. Measurements involving point 1 (accessing 1 is complicated) characterize class 3. The highest costs in our example are caused by an experimental change of the lightbulb.

This example might appear too simple. But in the domain considered here (car diagnosis) devices of rather moderate complexity occur frequently. The problem for the developers of traditional diagnosis and repair handbooks is not that a single independent device is too complicated, but that there are too many such devices to consider them all in detail (C8). From the point of view of model-based diagnosis the problem is that for several reasons (C4, C5) the diagnostic expert wants explicit control over the test sequences to be carried out later in the workshop. Post-editing a partially derived fault tree gives him this freedom in a way that could hardly be achieved with a purely model-based approach.

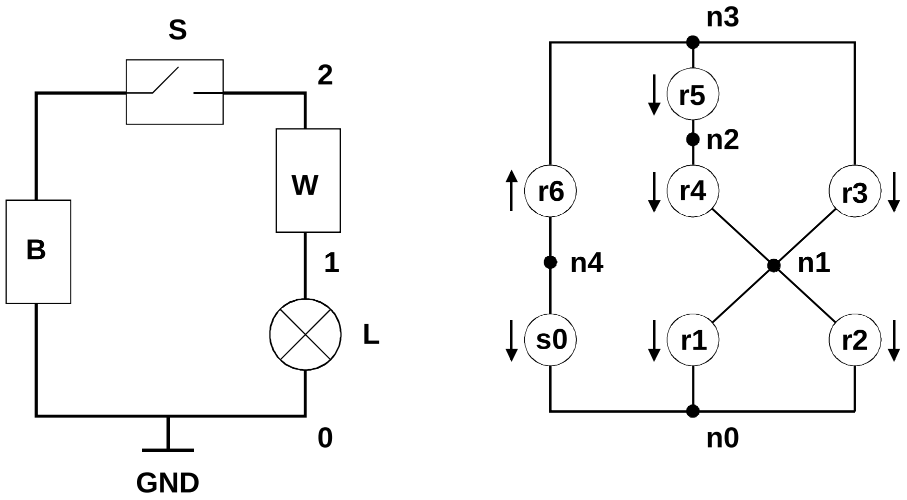

Figure 1: Diagram of a device and corresponding resistive net

In our modeling framework the normal and faulty behaviour of all components is represented by a resistive network (see Fig. 1). Each component class specifies a resistive net model. In our example, the model for wire is a resistance with R = 0 if the wire is ok and R = ∞ if the wire is broken. A short of the wire to an arbitrary point is modeled by a resistance R with R = ∞ if wire is ok and R = 0 if the wire is shorted. Each fault assumption translates to a certain assignment of R-values to the edges of a net covering the assumption. Our modeling enables us to initially cover all possible fault assumptions with a single resistive network. The battery in our example was modeled as a series connection of an ideal source s0 and an inner resistance r6. In summary, the voltage sources and resistances (the labeled edges of the graph) represent:

s0	battery as ideal voltage source
r1	resistance of lightbulb L; ∞, if burnt out
r2	0, if wire W is shorted to ground, ∞ else
r3	0, if wire W is shorted to plus, ∞ else
r4	resistance of wire W; 0 in the normal case and ∞, if W is broken
r5	switch S; 0, if closed and ∞, if open
r6	inner resistance of the battery

Automated modeling proceeds as follows: In a first stage we perform series-parallel analysis of the resistive net derived from the device diagram. Successful sp-analysis results in a tree with leaves representing the edges of the given net. The inner nodes aggregate serial and parallel subnets and the root represents the resistance of the whole net with respect to a given source. If sp-analysis succeeds we are done with modeling, since the resulting tree can be used to derive algebraic expressions for all voltages and currents: Resistances of inner nodes and of the root node are calculated in a top-down manner while the currents and voltages of leave nodes are calculated bottom-up using the precalculated resistances. If sp-analysis fails, i.e. the set of fault assumptions cannot be covered by a single sp-reduceable net, we simplify the net by deleting an edge representing a resistance with only two possible values $\{0, \infty\}$. This splits the set of covered fault assumptions into two disjoint subsets, each associated with one of the two resulting simplified networks. The procedure continues until - in the ideal case - all fault assumptions are covered by sp-reducable nets. We apply techniques here inspired by the ATMS framework and reuse sp-analysed subnets valid under more than one fault assumption. Details will be presented in a forthcoming paper. However, some nets, each associated with a non-empty set of fault-assumptions, may remain that can neither be simplified nor sp-reduced to a single resistance. These are processed in a second stage by applying Cramer's Rule as illustrated below.

Assume that the net given in Figure 1 was not series reducable (as it actually is). To apply Cramer's Rule, the inference mechanism analyses the net with its 13 unknown voltages and currents. The net has n = 5 nodes n0 ... n4 and m = 7 edges. Therefore it can be deduced that there are (n-1) = 4 independent current equations (Σ i = 0) and (m-n+1) = 3 independent voltage equations (Σ u = 0). Each of the 6 r-labeled edges gives us one equation of the form u = Ri. So we get 13 equations and exactly that many current and voltage unknowns. Therefore we can derive the following sparse matrix, representing a set of 13 independent linear equations:

i0	i1	i2	i3	i4	i5	i6	u1	u2	u3	u4	u5	u6	
1	1	1											
				-1	1								
			1		1	-1							
1						1							
							1		1			1	s0
									1	-1	-1		
							-1	1					
	-r1						1						
		-r2						1					
			-r3						1				
				-r4						1			
					-r5						1		
						-r6						1	

Empty matrix elements represent zero entries. Solving the matrix symbolically by applying Cramer's Rule (here performed with Mathematica) yields:

$$i0 = (s0/Det)\,(r13 + r23 + r14 + r24 + r15 + r25) \tag{1}$$
$$i1 = (s0/Det)\,(r23 + r24 + r25) \tag{2}$$
$$i2 = (s0/Det)\,(r13 + r14 + r15) \tag{3}$$
$$i3 = (s0/Det)\,(r14 + r24 + r15 + r25) \tag{4}$$
$$i4 = (s0/Det)\,(r13 + r23) \tag{5}$$
$$i5 = (s0/Det)\,(r13 + r23) \tag{6}$$
$$i6 = (s0/Det)\,(r13 + r23 + r14 + r24 + r15 + r25) \tag{7}$$
$$u1 = (s0/Det)\,(r123 + r124 + r125) \tag{8}$$
$$u2 = (s0/Det)\,(r123 + r124 + r125) \tag{9}$$
$$u3 = (s0/Det)\,(r134 + r234 + r135 + r235) \tag{10}$$
$$u4 = (s0/Det)\,(r134 + r234) \tag{11}$$
$$u5 = (s0/Det)\,(r135 + r235) \tag{12}$$
$$u6 = (s0/Det)\,(r136 + r236 + r146 + r246 + r156 + r256) \tag{13}$$

where *r123* is a shorthand for the product *r1 r2 r3* and

$$Det = r123 + r124 + r134 + r234 + r125 + r135 + r235 + r136 + r236 + r146 + r246 + r156 + r256 \tag{14}$$

Cramer's Rule states the following: Let $Ax = b$ be a system of linear equations with a unique solution, i.e. A non-singular. Then the x_i can be calculated by:

$$x_i = \frac{Det(A_i)}{Det(A)} \qquad \text{where } A_i = (A \text{ with the } i\text{-th column in } A \text{ replaced by } b)$$

Each fault assumption translates to a certain assignment of 0 and ∞ to R-quantities. The generally valid formulas (1) ... (13) can be evaluated for a given fault assumption in a simple way by computing the 0 and ∞ limits of the R-quantities. This yields symbolic expressions for all currents and voltage drops in the net. To our opinion, this is a remarkably compact representation of the behaviour of a device under arbitrary (also multiple) fault assumptions. Note that we have not used any quantitative information about parameters of the device components so far, therefore we could call this approach a qualitative one. However, because of its close relation to methods developed in the field of computer algebra (for an introduction see e.g. [DaSiTo 88]) we find this modeling approach best termed *algebraic modeling*.

If Cramer's Rule is applied in the naive form presented above, the resulting expressions for the observable quantities will always possess the simple and uniform structure shown in the example. But the size of the expressions will grow exponentially in the number n of the quantities as determinants are calculated as a sum of n! terms, where each term is the product of n matrix elements.

Therefore, Cramer's Rule is often considered to be useless for practical purposes. This is true as long as we are interested in numerical solutions of linear systems where simple Gaussian Elimination is known to work in $O(n^3)$. If we are interested in symbolic solutions of large, sparse linear systems, Cramer's Rule becomes applicable. [Smit 79] and [Smit 81] gives a conceptually simple algorithm called FDSLEM for solving such systems based on Cramer's Rule. This algorithm calculates the determinant in a factorized, i.e. not fully expanded form. It combines several sophisticated techniques to omit multiple calculations by storing intermediate results and by always choosing an optimal row or column for the expansion of subdeterminants. Experiments presented in [Smit 79] show that the calculation of a determinant is performed in linear time at least for the given examples. These are linear systems derived from ladder networks with 10 to 26 variables.

state	fault assumption / observation	no fault	wire W broken	wire W short to plus	wire W short to ground	light bulb L burnt out
S closed	light	○	●	○	●	●
S open	light	●	●	○	●	●
S open	r20	C	∞	!	0	∞
S open	u20	0	?	A	0	?
S closed	u20	A	B	A	0	B
S open	r21	0	∞	!	0	0
S open	r10	C	C	!	0	∞
S open	u21	0	?	0	0	?
S open	u10	0	B	A	0	?
S closed	u21	0	B	0	0	0
S closed	u10	A	B	A	0	B
S closed	light after bulb change	○	●	○	●	○

○ = lightbulb is on
● = lightbulb is off
? = value is with no model predictable
! = measurement of r not allowed, because current is not zero
A = r(lightbulb) u(battery) / (r(lightbulb) + r(battery))
B = u(battery)
C = r(lightbulb)

Table 2: Derived fault dictionary

Our hope is that the tractability of Cramer's Rule as indicated by these experiments extends to systems with up to a hundred variables. This would enable us to apply this method to the modeling of real-world devices. It should be noted however that we did not implement any of the techniques presented in this paper.

At this point we have derived algebraic expressions for currents and voltages of resistances in the net, either by sp-analysis in stage 1 or by computing limits of the formualae derived by Cramer's Rule in stage 2. These low-level expressions need further processing in order to derive propositions on the level of observations e.g. about the behaviour of the lightbulb in different fault situations. Merging these propositions into one table and sorting them by their costs yields the fault dictionary presented in Table 2. The rows (observation in a given state) and columns (fault assumption) in this table are subsets of the combinatorially possible ones and were selected by the developer while preparing the device diagram (see Fig. 1).

3 Fault-tree induction

The fault trees to be derived here are basically commented decision trees. Every path in a fault tree corresponds to a certain test sequence in the workshop. Since childnodes of a node n in the fault tree share the same path down to n, the test sequences to confirm or disconfirm them are equal down to node n. The comment or label of a node specifies the fault assumptions represented by that node, i.e. the current diagnostic candidates.

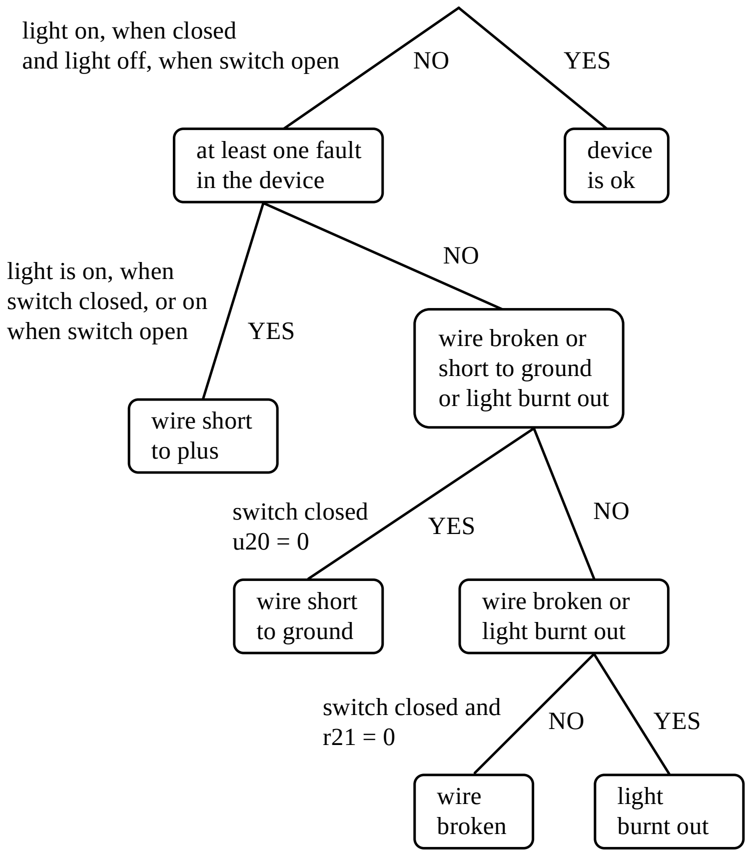

Figure 2: Fault tree derived from the diagram

For deriving the fault tree we start with the case "everything is ok, no fault" and deduce the test to confirm or disconfirm this case. All other cases are confirmed or disconfirmed in the order resulting from balancing the gain of information and the costs of a test in some reasonable way. The derivation proceeds in a straigtforward manner according to the ID3 algorithm as described in [Quin 86]. For our example we get the fault tree shown in Fig. 2.

Note that the example presented here could of course be mastered by a purely model-based approach. For a motivation of the use of fault trees recall our arguments given in Section 1.

It is not specified by ID3, in which way the costs of a test are balanced with the information gain of the test, since [Quin 86] considers only the information gain based on calculations of the entropy, not costs. Because there are no real-time constraints while deriving a fault tree (in this respect our situation differs from a purely model-based approach), we can apply computationally expensive techniques to do this. For example, we could apply the A* algorithm by defining a cost function f(n,test) = g(n) + h(n,test). g is the cost to confirm node n, and h is an optimistic estimation of the remaining costs, if we choose the given test. The function h introduces information not covered by other approaches purely based on one-step look-ahead entropy, as e.g. described in [KleWil 87]. Basically, h will help us to prevent a situation, where we postpone an expensive test although a look ahead would show that we have to execute it anyway.

However, these problems are far from having been solved and will be subject of further investigations.

4 Conclusion and future work

We outlined the relevant characteristics of vehicle diagnosis and presented a method to derive a fault dictionary from a given device diagram by performing series-parallel analysis and by applying Cramer's Rule in a tractable way in cases when sp-analysis fails. The derived fault dictionary serves as a basis for fault-tree induction with an ID3-like algorithm.

Current research concerns empirical validation of the methods presented here. Further research will extend the modeling techniques to piecewise linear models in order to cope with non-linear componentes like diodes, and to models with state to model e.g. a fuse, whose resistance becomes ∞ after a short time span Δt when the actual current exceeds a certain limit or a relay that will open or close after Δt if the actual current through its coil exceeds a given limit.

References

[DaSiTo 88] J. H. Davenport, Y. Siret, E. Tournier: Computer Algebra: Systems and Algorithms for Algebraic Computation. *Academic Press*, 1988.

[HaCoKl 92] Walter Hamscher, Luca Console, Johan de Kleer (Eds): Readings in Model-Based Diagnosis, *Morgan Kaufmann Publishers, San Mateo, CA*, 1992.

[KaMaRo 90] Dean C. Karnopp, Donald L. Margolis, Ronald C. Rosenberg: System Dynamics: A Unified Approach, *John Wiley & Sons, Second Edition*, 1990.

[KleWil 87] Johan de Kleer, Brian C. Williams: Diagnosing multiple Faults, *AI 32*, pp. 97-130, 1987 (improved version in [HaCoKl 92])

[LeeOrm 92] M. H. Lee, A. R. T. Ormsby: Qualitative Modeling of Electrical Circuits, *QR'92: Sixth Int. Workshop on Qualitative Reasoning about Physical Systems*, pp. 155-169, 1992.

[Quin 86] J. R. Quinlan: induction of decision trees. *Machine Learning, 1*, pp. 81-106, 1986.

[Smit 81] Jaap A. Smit: A Cancellation Free Algorithm, with Factoring Capabilities, for the Efficient Solution of Large Sparse Sets of Equations.*Proceedings of the 1981 ACM Symposium on Symbolic and Algebraic Computation SYMSAC*, ACM Inc., New York, pp. 146-154, 1981.

[Smit 79] Jaap A. Smit: New Recursive Minor Expansion Algorithms, A Presentation in a Comparative Context, *EUROSAM 79 Symposium on Symbolic and Algebraic Manipulation*, Lecture Notes in Computer Science LNCS 72, Springer, 1979.

[SteSus 77] Richard M. Stallmann, Gerald J. Sussmann: Forward Reasoning and Dependency-Directed Backtracking in a System for Computer-Aided Circuit Analysis, *AI 9*, pp. 135-196, 1977.

[SusSte 80] Gerald J. Sussmann, Guy L. Steele Jr. : CONSTRAINTS - A Language for Expressing Almost-Hierarchical Descriptions, *AI 14*, pp. 1-39, 1980.