

Reclustering Techniques to Perform Network Reconfiguration Tasks *

Jaime Cerda Juan J. Flores

{jcerda,juanf}@zeus.ccu.umich.mx

Facultad de Ingeniería Eléctrica

Universidad Michoacana de San Nicolás de Hidalgo

Morelia, Michoacan, Mexico

Tel. (43) 16 90 71

Guillermo B. Morales Luna

gmorales@cs.cinvestav.mx

Sección de Computación

Departamento de Ingeniería Eléctrica

Centro de Investigación y de Estudios Avanzados del I.P.N.

Abstract

In this paper we present algorithms to perform clustering modification in the presence of ideal and non-ideal faults. In this process, we assume some other agent already derived the nature of the fault (on a single fault assumption), and we are just dealing with the process of reasoning about and explaining the ramifications of such faults. With these algorithms, we can address several reasoning tasks to explain a circuit fault. We can answer questions like “How does the circuit clustering change in the presence of fault F ?”. “What happened to the current in element x after fault F ?”, “Why did the current in element x increase after fault F ?”, or “How would the circuit topology change if fault F becomes an ideal fault (short or open circuit)?”. This modularity opens up the possibility to produce hybrid systems, which can combine several techniques. For instance, artificial neural networks for diagnosis and qualitative reasoning for explanation.

1 Introduction

In the presence of a fault, an electric circuit experiences topological changes that affect its behavior. A short-circuit makes two nodes collapse, while an open circuit eliminates a branch of the circuit. Real short-circuits are modeled as shorts through a fault resistor. In those cases another element is inserted in the circuit.

In the presence of a fault, the circuit behavior deviates from the expected behavior. Circuit diagno-

sis tries to determine what kind of fault may have taken the circuit from normal operation to the observed mode.

Several approaches have been taken in solving the diagnosis problem [Bennani & Bossaert1998, Deuker, Perrier, & Amy1998, Chantler *et al.*1998, Say1999, Mauss & Sachenbacher1999, Milde *et al.*1999]. Consistency based diagnosis finds a (faulty) mode of operation whose model is consistent with the observed behavior. Artificial neural networks learn a classification function, given a set of examples of the observed behavior under different circuit faults. Flores [Flores1997] proposed a diagnosis algorithm for electric circuits based on the clustering.

Now, suppose we have a diagnosis engine, perhaps based on any of the above mentioned mechanisms. There are several reasoning tasks that can be addressed after we have determined that a given circuit fault occurred. One would like to answer questions like: “What happened to the current in element x after fault F ?”, “Why did the current in element x increase after fault F ?”, “How would the circuit topology change if fault F becomes an ideal fault (short or open circuit)?”, or “How does the circuit clustering change in the presence of fault F ?”. In this paper we propose schemes to perform such reasoning tasks. All reasoning tasks we propose here are based on the assumption of single faults.

The paper is organized as follows: In section 2, we review the clustering and modeling of circuits, which are the underlining mechanisms of the methods presented here. Section 3 outlines the proposed algorithm to modify a circuit clustering in the presence of a fault. Section 4 addresses the asymptotic reason-

*Research supported by Conacyt grant No.31857A, by CIC-UMSNH grant No. 9.3

ing, section 5 shows the first order reasoning, section 6 shows how the clustering modification can be used to produce explanations. And finally, section 7 concludes the work presenting our main contributions and discusses some possible extensions for future research.

2 Clustering and Modeling

Clustering [Flores & Cerda1999, Mauss1998, Mauss1997] takes part of a circuit and reduces it to a single element, called equivalent element, between each pair of nodes connected to the eliminated region. The substitution is called equivalent because the rest of the circuit does not notice the substitution i.e. the rest of the circuit currents and voltages remain unaltered.

Working through a series of substitutions we transform the circuit, getting a simpler circuit each time. When the circuit is simple enough, we can compute the currents and voltages, then we go backwards through the sequence of simplifications, computing the rest of the variables for each step. At the end of the process, we return to the original circuit, having computed all variables.

The modeling and solution process does not have to be implemented that way. What we actually do is to obtain the reduction sequence, forming a graph (the clustering graph). We produce a set of algebraic and qualitative relations, called constraints, for each part of the circuit and for each reduction step. The result of this second phase is called a constraint-based model of the circuit. Finally, we perform constraint propagation on the circuit model, which solves the circuit for the unknown variables.

In the following subsections we review the clustering technique used for circuit modeling and analysis.

2.1 Generalized Star-Mesh Reduction

Star-Mesh reductions [Shen1947] allow us to eliminate a node from the network, creating an equivalent circuit represented by a full graph among the neighbors of the eliminated node. See Fig. 1.

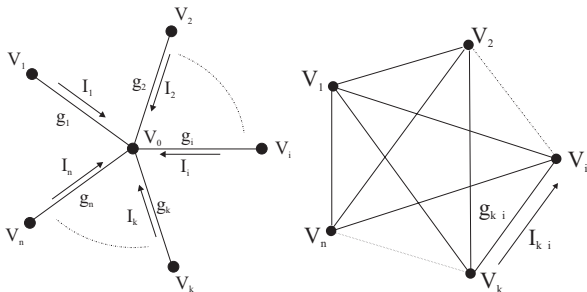


Figure 1: Star-Mesh Conversion

For the general case, eliminating a node produces $\binom{n}{2} = \frac{n(n-1)}{2}$ new elements, some of which may be in parallel to other circuit elements.

The main idea of clustering is to produce a series of equivalent circuits, reducing a node at a time. When all nodes except those connected to sources have been eliminated, we can compute currents for all elements. Going back in the reduction tree, we can compute currents and voltages for the previous reduction step. We do this until we return to the original circuit.

2.2 Circuit Modeling

In practice, all clustering is done first and then produce a constraint-based model of the circuit. The clustering process generates a graph containing the history of the transformations. In the clustering graph, we distinguish three kinds of nodes: admittance vertices (representing either original or equivalent admittances), node vertices (one such vertex for each node reduction, each element connected to the eliminated node points to that node vertex; equivalent admittances come out from the eliminated node vertex), and parallel vertices (the parallel elements point to the reduction vertex, which represents an admittance). On each step of the clustering process, we generate a node vertex in the clustering graph. The process stops when all remaining nodes are connected to sources.

The model we chose to produce is based in admittances, because the mathematical expressions for reductions are simpler in terms of admittances, instead of impedances. An admittance, denoted by g , is the multiplicative inverse of an impedance, and represents how capable is an element to conduct electrical currents, as opposed to the impedance, which is a measure of electrical resistance. So Ohm's law can be expressed in terms of admittance as $I = gV$.

To produce the circuit model we traverse the clustering graph, creating a set of algebraic and qualitative constraints for each vertex. Table 1 shows the algebraic and qualitative constraints for node reduction and parallel elements.

Fig. 2 shows an example circuit and a possible clustering graph for the same circuit.

3 Clustering Modification

A diagnosis program observe a device (a circuit in this case), and classifies its behavior as faulty or normal. In the case of a faulty circuit, the diagnosis program is supposed to provide a description of the fault. A normal circuit can be seen as a circuit with no fault. Any fault introduces a change in the circuit topology; it can be any of the following: introduction of a

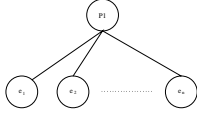
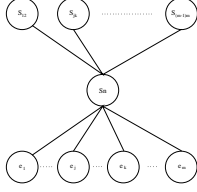
PARALLEL	STAR
	
$1 \leq j \leq n$ $i_j = V_{p1} g_j$ $V_j = V_{p1}$	$i, j \in \Gamma_n$ $g_n = \sum g_{in}$ $i_{in} = \sum_{k \in PK_n(i,n)} i_{ik}$ $V_i = \frac{i_i}{g_i}$ $V_n = \frac{1}{g_n} \sum g_{in} V_i$
$g_{p1} = \sum_{i=1}^n g_i$	$g_{ij} = \frac{g_{in} g_{jn}}{g_n}$
$\partial g_{p1} = \sum_{i=1}^n \partial g_i$	$\partial g_{ij} = \partial g_i + \partial g_j - \sum_{r \in \Gamma_n \setminus \{i,j\}} \partial g_r$

Table 1: Algebraic and Qualitative Constraints

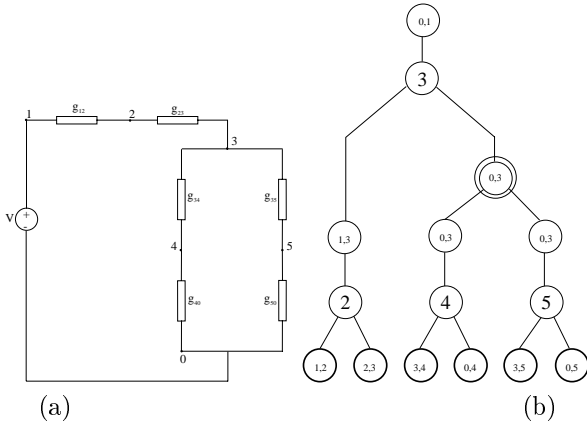


Figure 2: Clustering example

new component between two nodes (non-ideal short-circuit), introduction of a new component in a branch of the circuit (non-ideal open-circuit), elimination of a component (ideal open-circuit), or collapse of two nodes (ideal short-circuit). The last two cases can be seen as special cases of the first two, where the fault admittance is ∞ and 0, respectively.

The idea of clustering modification is to produce a clustering for a faulty circuit, as close as possible to the cluster we produced for the original (non-faulty) circuit. This task is accomplished under the assumption that we have already diagnosed the circuit, so we know the fault (or at least have a hypothesis of what it might be).

When we introduce a fault admittance to a circuit, the degree of the nodes (x and y) it is connected to increases by one. That increase reflects that the two faulty nodes are now neighbors. Let us assume that x was not clustered after y , and that $\text{degree}(x)=n$. Reconsidering the elimination of node x , we now have to increase the number of outgoing elements from node x in the clustering graph by n (these n elements will connect the old n neighbor nodes of x to y). If there already was an element in parallel with the inserted element, we just replace it by a parallel cluster. Otherwise, we have to recurse, considering the topological modifications that this new element causes to the circuit. This procedure is formally stated in Fig. 3.

Function clustering-modification takes as input clustering graph CG and fault element F connected to nodes x and y . It just calls reclustering with a copy of the clustering graph and a singleton queue, formed by the fault element. Function reclustering takes a clustering graph and a queue of elements to be processed, and returns a modified clustering graph with all the elements in the queue incorporated into the clustering graph. Function recluster-node takes a clustering graph, a node and an element; it adds the element to the node reduction in the clustering graph and returns the new cluster. Function elements returns all out-coming nodes of the cluster (i.e. the equivalent elements from the node reduction). The meaning of the rest of the functions is pretty much intuitive.

Fig. 4 shows the modifications (enclosed by a curved line) of the clustering of Fig. 2 for a fault between nodes 4 and 5.

4 Asymptotic Reasoning

An interesting question is ‘‘How does the circuit structure changes when the fault admittance is infinite/zero?’’. When an admittance takes on an infinite value, two nodes collapse; when it takes on a zero value, a section of the circuit gets disconnected. Asymptotic reasoning deals with these kind of topics

```

clustering-modification( $CG, F(x, y)$ )
   $MCG = CG$ 
  queue= $F(x, y)$ 
  return recluster( $MCG, queue$ )

recluster( $MCG, queue$ )
  while (queue)
    element( $w, z$ )= $remove(queue)$ 
    if (! element  $\in CG$ )
      if (par= $search-for-parallel(CG, element(w, z))$ )
        replace( $MCG, par, make-parallel-cluster(element(w, z), par)$ )
      else
        queue= $append(queue, elements(recluster-node(MCG, w, element(w, z))))$ 
  return( $MCG$ )

```

Figure 3: Reclustering

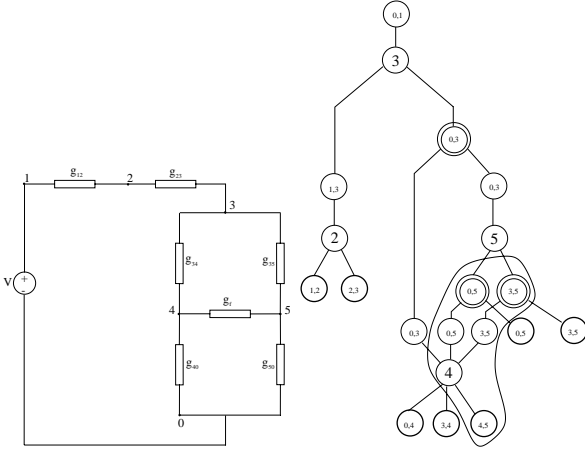


Figure 4: Clustering Modifications

where the value of the faulty element is 0 or infinity. In either case, the circuit structure changes, and the clustering has to be modified to reflect those changes.

An algorithm to compute such changes can be used to explain what happens in the presence of the fault. That algorithm can also be used to produce a simplified circuit and circuit model, which in turn can be used to verify that the exhibited behavior corresponds to the simplified circuit model.

As an example, Fig. 5 shows the clustering modification for a circuit region containing an ideal short-circuit. In that figure, if element g_{03} becomes a short, nodes 0 and 3 form a super-node. Under those conditions, we can see that the elements connected to node 0 are now directly connected to node 3. Also, there is no element (as there was none before) connecting nodes 1 and 2.

For this case, when an admittance takes on an infinite value, ($g \rightarrow \infty$), the reduction formulas taken to the limit, yield the appropriate changes to the clustering structure. The situation depicted in Fig. 5 gener-

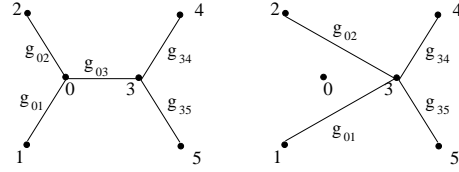


Figure 5: Ideal Short-circuit

alizes for the case when more elements are connected to nodes 0 and 3.

In general, when an admittance becomes infinity ($g \leftarrow \infty$), the reduction formulae taken to the limit, give us

$$\begin{aligned}
\forall i \in \Gamma_0, i \neq f \\
\lim_{g_{0f} \rightarrow \infty} (g_{if}) &= \frac{g_{0i}g_{0f}}{\sum_{j \neq f} g_{0j} + g_{0f}} \\
&= \frac{g_{0i}g_{0f}}{g_{0f}} \\
&= \frac{\sum_{j \neq f} g_{0j}}{g_{0f}} + \frac{g_{0f}}{g_{0f}} \\
&= \frac{g_{0i}}{0 + 1} \\
&= g_{0i}
\end{aligned}$$

$$\begin{aligned}
\forall i, j \in \Gamma_0; i \neq j; i, j \neq f \\
\lim_{g_{0f} \rightarrow \infty} (g_{ij}) &= \frac{g_{0i}g_{0j}}{\sum_{k \neq f} g_{0k} + g_{0f}} \\
&= 0
\end{aligned}$$

Nevertheless, we have to take care of several details. First, creating a super-node between two nodes may create parallel elements if both nodes have a common neighbor. This situation can be checked when we remove one node and move all elements to the

other node. Second, adding new elements to the second node increases its cardinality, therefore we have to add more equivalent elements coming out of that node in the clustering graph. In this case, we have the same situation as in clustering modification, so we can use the same algorithm. Third, a portion of the circuit can become inactive when short-circuiting two nodes. In that case, reclustering eliminates nodes until an equivalent element (of the whole region) is connected from and to the same node. The whole region is eliminated. Fig. 6 shows the algorithm.

```

short-reclustering( $CG, x, y$ )
 $MCG = CG$ 
remove-cluster( $MCG, x$ )
queue=out-elements( $MCG, y$ )
return reclustering( $MCG, queue$ )

```

Figure 6: Reclustering a Short Circuit

The other case is an ideal open-circuit. Fig. 7 shows the clustering modification for such an example. In that figure, if element g_{03} opens up, nodes 0 and 3 get disconnected. Under those conditions, equivalent elements g_{13} and g_{23} open up as well, and the formula for element g_{12} becomes the formula for admittances in series connection.

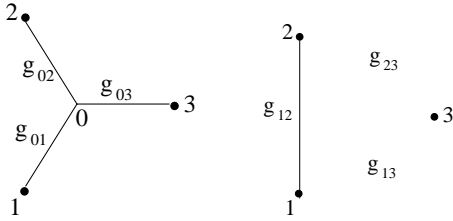


Figure 7: Ideal Open-circuit

For this case, the reduction formulas taken to the limit provide the appropriate changes to the circuit structure. The reduction applied in Fig. 7 generalizes for the case when more elements are connected to node 0. That is

$$\begin{aligned}
\forall i \in \Gamma_0, i \neq f \\
\lim_{g_{0f} \rightarrow 0} (g_{if}) &= \frac{g_{0i}g_{0f}}{\sum_{j \neq f} g_{0j} + g_{0f}} \\
&= 0
\end{aligned}$$

$$\begin{aligned}
\forall i, j \in \Gamma_0; i \neq j; i, j \neq f \\
\lim_{g_{0f} \rightarrow \infty} (g_{ij}) &= \frac{g_{0i}g_{0j}}{\sum_{k \neq f} g_{0k} + g_{0f}} \\
&= \frac{g_{0i}g_{0j}}{\sum_{k \neq f} g_{0k}}
\end{aligned}$$

If nodes 1 to n were neighbors to node 0 before, and now element g_{0n} opens up, the cluster that eliminates node 0 has fewer outgoing nodes. I.e. node n gets disconnected, so all equivalent elements g_{in} for $i \in \{0..n-1\}$ become zero. The effect of those zero elements has to be propagated upward in the clustering graph. Some of them will be in parallel with other elements; in that case, the parallel region reduces, remaining only the other element. Other elements may further disconnect other pairs of nodes. We apply the same procedure to those nodes. This yields the algorithm of Fig. 8.

Note that we may remove an element from a node with cardinality two. That would make the node a dangling node. Function zero-reclustering takes care of that case, removing the node from the clustering graph.

5 First Order Reasoning

In 2.1 we show how a set of confluences can be derived from the clustering graph. Those confluences capture how changes in parameters or variables of the circuit affect other variables.

To reason about how a fault affects other circuit variables, we start assuming nothing changes, except for the fault admittance. If the fault is a non-ideal short-circuit, we can assume the short was there all the time. Before the fault appeared, the fault admittance had a value of zero (no admittance equals infinite resistance). The change from zero admittance to any real valued admittance, produces an increase in its numerical value (i.e. a positive qualitative value). If the fault is a non-ideal open-circuit, we assume the fault admittance decreases.

Using those qualitative values, and zero for the rest of the admittances, we can run constraint propagation and derive causal chains of the changes in the rest of the circuit variables. Fig. 9 shows the results of the propagation process for the circuit of Fig. 4. In that figure, an up arrow means a positive qualitative value, i.e. an increase, a dash means zero, and a down arrow means negative.

This kind of reasoning allows us to answer questions like: “what happens to V_{R_3} when fault F occurs?”, or “why does V_{R_5} increases when fault F occurs?”. The system will generate structures that can be used to generate explanations like ¹ “the inclusion of F , $\partial F = +$, causes admittance d to increase, which in turn makes admittance j to decrease, ... , which makes admittance l decrease, increasing current I_l , ...”

For ideal faults, we can consider them as real faults in the limit. For a short circuit, we assume the admit-

¹no natural language has been generated, throughout the paper, we are paraphrasing the computer output

```

open-modification( $CG, F(x, y)$ )
   $MCG = CG$ 
  zero-reclustering( $MCG, x, F$ )
  queue=zero-elements( $MCG, x$ )
  return open-reclustering( $MCG, queue$ )

open-reclustering( $MCG, queue$ )
  while (queue)
    element( $w, z$ )=remove(queue)
    if (par=search-for-parallel( $MCG, element(w, z)$ ))
      reduce-parallel( $MCG, par, element(w, z)$ )
    else
      zero-reclustering( $MCG, w, element$ )
      queue=append(queue, zero-elements( $MCG, w$ ))
  return( $MCG$ )

```

Figure 8: Reclustering a Short Circuit

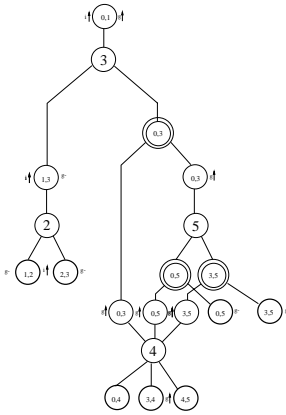


Figure 9: First Order Reasoning

tance was zero and now it takes on an infinite value, thus increasing. For an open circuit, the situation is the opposite and the fault admittance is considered to decrease.

6 Explaining Topological Changes

The clustering modification algorithms produce a new clustering graph with all changes marked. The topological modifications that occur in the presence of a fault can be explained using the changes introduced by the fault in the clustering graph. For example, the output that results from the introduction of fault into a circuit can be read as:

“when F occurs, the reduction of node 4 changes, since its cardinality increases. The equivalent admittance g_{345} , resulting of node 4 reduction, is now in parallel with ad-

mittance g_{35} , and the equivalent admittance g_{045} , resulting of node 4 reduction, is now in parallel with admittance g_{05} .”

7 Conclusions and Future Work

We are presenting an algorithm that modifies the clustering for a circuit in the presence of a fault. This algorithm takes a clustering and a fault, and produces a modified clustering, which reflects the inclusion of the fault into the circuit.

We also presented two algorithms to perform asymptotic analysis. That is, to reduce a clustering graph of a fault circuit under the assumption that the fault admittance is either zero or infinity.

Base on the modified clustering graph, we can produce causal chains that account for the changes in the values of the circuit variables, as a result of a fault on the circuit. We can also explain the changes in the circuit topology, based on the clustering modifications.

The clustering modification algorithms are linear on the number of elements of a circuit. The propagation algorithms we use are also linear on the size of the circuit, so all the analysis is efficient and can be performed on larger circuits than those used in the examples.

Since the methodologies proposed here take as input the circuit fault, we can use any method to diagnose the circuit. This opens up the possibility to use efficient diagnostic methods, such as artificial neural networks, and combine them (for the sake of explanation) with these reasoning methods. This hybridization allows us to maintain the best of both worlds, gaining efficiency and reasoning capabilities.

A bibliography revision shows some work on qualitative circuit analysis [Flores & Farley1996, Flores1997, Cerda & Flores1998, Mauss1998, Lee1999], some on circuit design [Flores & Farley1998] and design verification [McManus *et al.*1999], also some on diagnosis [Pettersson1996, Flores1997, Flores & Farley1997, Mauss & Sachenbacher1999], but the authors have no knowledge about any work on reasoning about structural changes in electrical circuits as a result of a fault.

References

- [Bennani & Bossaert1998] Bennani, Y., and Bossaert, F. 1998. Alarm generation in telephone network using multi-variate neural networks modelling and joint confidence interval. In *Proc. 9th International Workshop on Principles of Diagnosis (Dx98)*.
- [Cerda & Flores1998] Cerda, J., and Flores, J. J. 1998. Modelling multiple source circuits: An efficient clustering algorithm. *Sent to AICOM, special Issue on Model/Based Reasoning*.
- [Chantler *et al.*1998] Chantler, M.; Pogliano, P.; Aldea, A.; Torielli, G.; Wyatt, T.; and Jolley, A. 1998. The use of fault-recorder data for diagnosing timing and other related faults in electricity transmission networks. In *Proc. 9th International Workshop on Principles of Diagnosis (Dx98)*.
- [Deuker, Perrier, & Amy1998] Deuker, B.; Perrier, M.; and Amy, B. 1998. Fault-diagnosis using neuro-symbolic hybrid systems. In *Proc. 9th International Workshop on Principles of Diagnosis (Dx98)*.
- [Flores & Cerda1999] Flores, J. J., and Cerda, J. 1999. Modelling circuits with multiple grounded sources: An efficient clustering algorithm.
- [Flores & Farley1996] Flores, J. J., and Farley, A. M. 1996. Qualitative phasor analysis. In *Proc. 10th Int. Workshop on Qualitative Reasoning About Physical Systems*.
- [Flores & Farley1997] Flores, J. J., and Farley, A. M. 1997. Diagnosis of linear circuits. In *Proc. 4th International Congress on Expert Systems*, 340–347.
- [Flores & Farley1998] Flores, J. J., and Farley, A. M. 1998. Incremental design for linear circuits. In *Proc. 12th Int. Workshop on Qualitative Reasoning About Physical Systems*.
- [Flores1997] Flores, J. J. 1997. *Reasoning about Linear Circuits in Sinusoidal Steady State*. Ph.D. Dissertation, University of Oregon.
- [Lee1999] Lee, M. 1999. Qualitative modeling of linear networks in ecad applications.
- [Mauss & Sachenbacher1999] Mauss, J., and Sachenbacher, M. 1999. Conflict-driven diagnosis using relational aggregations.
- [Mauss1997] Mauss, J. 1997. *Analyse Kompositionales Modelle durch Serien-Parallel-Stern Aggregation*. Ph.D. Dissertation, DISKI-183, Sankt Augustin.
- [Mauss1998] Mauss, J. 1998. Local analysis of linear networks by aggregation of characteristic lines. In *Proc. 9th International Workshop on Principles of Diagnosis (Dx98)*.
- [McManus *et al.*1999] McManus, A.; Price, C.; Snooke, N.; and Joseph, R. 1999. Design verification of electrical systems.
- [Milde *et al.*1999] Milde, H.; Hotz, L.; Kahl, J.; and Wessel, S. 1999. Qualitative analysis of electrical circuits for computer-based diagnostic decision tree generation.
- [Pettersson1996] Pettersson, G. 1996. Impedance driven model-based diagnosis of electric power distribution system faults. Master's thesis, Department of Computer Engineering, University of Central Florida, Orlando, Florida.
- [Say1999] Say, C. 1999. Using inter-behavior contradictions for modeling and diagnosis.
- [Shen1947] Shen, D. W. C. 1947. Generalized star and mesh transformations. *Philosophical Magazine and Journal of Science* 38(7):267–275(2).