Efficient Diagnosis based on Incomplete System Descriptions

Peter Fröhlich and Wolfgang Nejdl
Institut für Rechnergestützte Wissensverarbeitung
Universität Hannover
Lange Laube 3, 30159 Hannover, Germany
E-Mail: {froehlich, nejdl}@kbs.uni-hannover.de

Abstract

Fault diagnosis is an important task in technical domains. In the model–based diagnosis literature it has mostly been assumed that technical systems are described by complete, deterministic system descriptions.

We show that intuitive descriptions of technical systems are often incomplete, and need to be completed for providing correct diagnoses. Due to the structure of the axioms of such system descriptions completion is not straightforward, because predicate completion is not applicable and circumscription does not yield first order sentences. We introduce an algorithm which reduces the system descriptions to propositional representations which are then completed.

Furthermore, we discuss how diagnoses can be computed efficiently for the completed system descriptions. While the original descriptions of the system behaviour usually consist of horn clauses, their completions are typically not horn. We provide formal results, which allow to exploit the structure of the original horn clauses, when reasoning with the completed clause sets.

1 Introduction

Fault diagnosis is an important task in technical domains like Electronic Design Automation, Telecommunication or Autonomous Systems. The connection between the model–based approach to diagnosis and non–monotonic reasoning is twofold. On the one hand the computation of diagnoses usually includes the minimization of the *Ab*–predicate and is thus a non–monotonic reasoning problem. On the other hand predicate completion is used in the diagnosis domain to reduce abductive reasoning to consistency–based reasoning [4, 3, 9].

In the model–based diagnosis literature it has mostly been assumed that technical systems are described by complete, deterministic system descriptions. We show that intuitive descriptions of technical systems are often incomplete, and need to be completed for providing correct diagnoses. Due to the structure of the axioms of such system descriptions completion is not straightforward, because predicate completion is not applicable and circumscription does not yield first order sentences. We introduce an algorithm which reduces the system descriptions to propositional representations which are then completed.

Furthermore, we discuss how diagnoses can be computed efficiently for the completed system descriptions. We have recently introduced a technique for exploiting the structure of the horn clauses in system descriptions to speed up diagnosis computation [6]. While the original descriptions of the system behaviour usually consist of horn clauses, their completion is typically not horn. In the current paper we provide formal results, which allow to exploit the structure of the original horn clauses, when reasoning with the completed clause sets.

2 A Propagation Model for Cellular Networks

We are studying the application of model–based diagnosis to alarm correlation in cellular networks. Such networks consist of semi–intelligent network elements, which send alarm messages to indicate errors. Usually, the abstraction level of these messages is low, leading to a great number of alarm messages for any single cause. In large networks, like the current GSM networks, the alarm vectors supplied by the network elements tend to flood the workstations of the operators especially in critical situations like the passage of a thunderstorm front. To avoid overloading the operators, alarm correlation systems are required which filter and condense the incoming alarms to meaningful high-level alarms and diagnoses. The basic idea of our model–based approach to alarm correlation is to describe

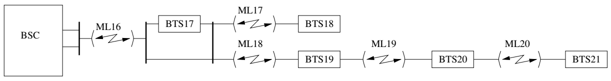


Figure 1: A Trunk of Components connected to a Base Station Controller

the propagation of different alarm messages by logical axioms. In [7] we give a more detailed treatment of the technical background and the different models needed for the different alarm types. In this paper, we only take a very simple example for one type of message in order to illustrate the modeling problems arising in the diagnosis of networked technical systems, and show how they are solved using the methods introduced in the current paper.

We consider a small part of a cellular network as depicted in figure 1. The picture shows a set of base transceiver stations (BTS17–BTS21), which can be contacted by handheld phones. These transmit their data via microwave links (ML16–ML20) to the base station controller (BSC). Let us now develop a logical model for the propagation of alive–messages sent by the base stations. Axiom $A1$ denotes that every base station sends an alive message. Each component has two ports, an upstream port (*Up*) and a downstream port (*Down*). Thus, the precise statement of axiom $A1$ is that each component of type BTS has an alive–signal at its upstream port.

$$A1: \quad \forall c\; Type(c, BTS) \rightarrow Signal(c, Up, Alive(c))$$

Connections between network elements forward signals in the direction of the base station controller (see axiom $A2$). If the upstream port of a network element c_1 is connected to the downstream port of another network element c_2, and the signal $Alive(c_{Sender})$ is present at the upstream port of c_1, it is forwarded by the connection to the downstream port of c_2.

$$\begin{aligned} A2: \quad & \forall c_1 \forall c_2 \forall c_{Sender}\; Conn(c_1, Up, c_2, Down) \wedge \\ & Signal(c_1, Up, Alive(c_{Sender})) \\ & \rightarrow Signal(c_2, Down, Alive(c_{Sender}))) \end{aligned} \tag{1}$$

Microwave links also propagate signals towards the base station controller. Axiom $A3$ states that a microwave link c, which is working properly ($Ok(c)$), transmits the alive message from its downstream port to its upstream port.

$$\begin{aligned} A3: \quad & \forall c \forall c_{Sender}\; Type(c, ML) \wedge Ok(c) \wedge \\ & Signal(c, Down, Alive(c_{Sender})) \\ & \rightarrow Signal(c, Up, Alive(c_{Sender}))) \end{aligned} \tag{2}$$

Finally, axiom $A4$ denotes that base transceiver stations also propagate messages. In contrast to microwave links, failures in base transceiver stations usually do not affect the transmission of signals, so that the premise $Ok(c)$ is omitted here.

$$\begin{aligned} A4: \quad & \forall c \forall c_{Sender}\; Type(c, BTS) \wedge \\ & (Signal(c, Down, Alive(c_{Sender})) \rightarrow \\ & Signal(c, Up, Alive(c_{Sender}))) \end{aligned} \tag{3}$$

The axioms describing the propagation behaviour are complemented by a set of facts for the predicates *Type* and *Conn*. These facts describe the configuration of our small network.

$$\begin{aligned} DB: \{\; & Type(Bts17, BTS), Type(Bts18, BTS), \\ & Type(Bts19, BTS), Type(Bts20, BTS), \\ & Type(Bts21, BTS), Type(ML16, ML), \\ & Type(ML17, ML), Type(ML18, ML), \\ & Type(ML19, ML), Type(ML20, ML) \\ & Conn(ML16, Up, Bsc1, Down), \\ & Conn(Bts17, Up, ML16, Down), \\ & Conn(ML18, Up, ML16, Down), \\ & Conn(ML17, Up, Bts17, Down), \\ & Conn(Bts18, Up, ML17, Down), \\ & Conn(Bts19, Up, ML18, Down), \\ & Conn(ML19, Up, Bts19, Down), \\ & Conn(Bts20, Up, ML19, Down), \\ & Conn(ML20, Up, Bts20, Down), \\ & Conn(Bts21, Up, ML20, Down)\} \end{aligned}$$

The axioms together with the closure of the database DB yield the system description $\mathcal{S}$, i.e. $\mathcal{S} = \{A1, A2, A3, A4\} \cup CWA(DB)$. We explicitly apply the closed world assumption to our database DB to keep the algorithm from changing the extensions of the predicates *Type* and *Conn* during the computation of diagnoses.

Comp denotes the set of all system components. Suppose that the alive-messages from $Bts17$, $Bts18$, and $Bts19$ have arrived at the BSC, while the signals from $Bts20$ and $Bts21$ did not arrive:

$$\begin{aligned} Obs = \{ & Signal(Bsc1, Down, Alive(Bts17)), \\ & Signal(Bsc1, Down, Alive(Bts18)), \\ & Signal(Bsc1, Down, Alive(Bts19)), \\ & \neg Signal(Bsc1, Down, Alive(Bts20)), \\ & \neg Signal(Bsc1, Down, Alive(Bts21))\} \end{aligned}$$

According to the definition of consistency–based diagnosis [14], we must find minimal sets of faulty components, so that the observations are consistent with the assumption that all other components are working.

Definition 2.1 *A* Diagnosis[1] *of* $(\mathcal{S}, Comp, Obs)$ *is a set* $\Delta \subseteq Comp$ *such that* $\mathcal{S} \cup Obs \cup \{Ok(c) | c \in Comp \backslash \Delta\} \not\models \perp$. *A*

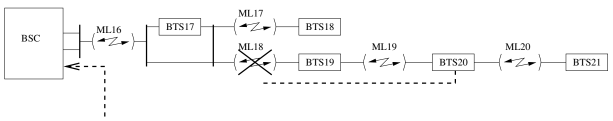
[1] This definition deviates slightly from the one in [14]. We use

diagnosis Δ *of* $\mathcal{S} \cup Obs$ *is called a* Minimal Diagnosis *is there is no diagnosis* Δ' *of* $\mathcal{S} \cup Obs$*, such that* $\Delta' \subset \Delta$.

Given the system description $\mathcal{S}$ and the observations Obs introduced above, we find two minimal diagnoses: $\Delta_1 = \{ML19\}$ and $\Delta_2 = \{ML18\}$. Reconsidering figure 1 we see that diagnosis Δ_1 is the intended diagnosis in the given scenario: If the mircowave link $ML19$ is not working, it is evident that the alive messages from $Bts20$ and $Bts21$ did not arrive at the base station controller.

The additional diagnosis $ML18$ is counterintuitive. If $ML18$ were faulty, we would expect that the alive–message from $Bts19$ gets lost, too. In the above model, it is consistent to assume that the faulty microwave link transmitted this message and lost the others. Thus, consistency–based diagnosis is too weak to draw the right conclusions from this intuitive system description. Our investigation of a more complex version of this application has shown that there is no way to manually define an intuitive, declarative model of this domain, from which consistency–based diagnosis draws the right conclusions efficiently.

As an example for the problems encountered, consider the following thought: One could try to eliminate the additional diagnosis by adding a fault model for the microwave link, stating that a faulty microwave link does not transmit messages. However, then we would still find a consistent model with the diagnosis $ML18$:



In this counter–intuitive scenario the farend–alarm is lost at $ML18$, but for some reason it reappears at the upstream port of $ML16$. If we extend the fault models by specifying that the absence of alarms is propagated, the contrapositives of the corresponding axioms cause alarms to be propagated from the base station controller into the direction of the base transceiver stations.

The simplified example discussed in this section already shows that it is easy and intuitive to positively describe how the network transmits messages, while it is hard to provide a complete description, which rules out all impossible situations. To draw the right conclusions from an intuitive (incomplete) model of our domain, we need to complete the model using a systematic method. As we will see in the next section, the straightforward application of circumscription/predicate completion to the first order logic representation is not possible.

a predicate Ok instead of the predicate Ab used by Reiter. The intended meaning is $\forall c\ Ok(c) \leftrightarrow \neg Ab(c)$. The use of Ok has the advantage that the clauses of our theory are horn.

3 How to Complete this System Description

In large technical systems usually certain values are propagated through the system, which consists of a set of components and their connections. In the above case the axioms describe the propagation of messages through the network. Other examples are the propagation of water through a system of pipes and valves or the propagation of current through a switching network. In their simplest form, such propagation descriptions consist of an axiom describing how a property P can be caused directly (e.g. a message is sent by a component as in axiom $A1$) and an axiom describing how the property is propagated over the connection of the system components (compare axiom $A2$) and within the system components (axioms $A3$ and $A4$). The simplest form of a propagation axiom is shown in the following formula:

$$\forall x \forall y\ (C_P(x) \rightarrow P(x)) \wedge (P(x) \wedge Conn(x,y) \rightarrow P(y)). \quad (4)$$

A property is present at x if x is the creator or cause of the property P ($C_P(x)$). Furthermore, properties are propagated over connections. This axiom is simpler than the axioms $A1$ to $A4$ above because it does not consider the propagation within the components.

To eliminate counter–intuitive models as the one presented at the end of section 2, we have to apply predicate completion to the propagation axiom, thereby expressing that an instance of $P(x)$ is true, if and only if this is indicated by axiom 4.

Definition 3.1 *Positive, Solitary. A literal occurs* positively *in a formula F, iff it occurs positively in the clause form of F. A set of clauses is* solitary *in P, iff P occurs at most only once in every clause in which it occurs positively.*

Definition 3.2 *Predicate Completion [8] (originally defined as Database Completion in [2]). Let* $\mathcal{S}$ *be a system description in clause form, which is solitary in P. The set of all clauses, in which P occurs positively can be written as*

$$\begin{array}{ccc} \forall x_1 \ldots \forall x_n\, E_1 & \rightarrow & P(x_1,\ldots,x_n) \\ & \vdots & \\ \forall x_1 \ldots \forall x_n\, E_r & \rightarrow & P(x_1,\ldots,x_n) \end{array}$$

The predicate completion of P in $\mathcal{S}$ *is defined as*

$$\begin{array}{l} COMP(\mathcal{S};P) = \\ \quad \mathcal{S} \cup (\forall x_1 \ldots \forall x_n\, P(x_1,\ldots,x_n) \leftrightarrow E_1 \vee \ldots \vee E_r) \end{array}$$

In the special case, where P does not occur in $\mathcal{S}$ *we define*

$$COMP(\mathcal{S};P) = \mathcal{S} \cup (\forall x_1 \ldots \forall x_n\, P(x_1,\ldots,x_n) \leftrightarrow False)$$

Console and Torasso propose the use of predicate completion for the completion of system descriptions in diagnosis [4], based on formal work by Console Theseider Dupré and Torasso in [3]. Unfortunately, not even the completion of the simple formula 4 is equivalent to a first–order formula. Thus, the first–order techniques proposed by Console, Theseider Dupré and Torasso are not directly applicable to descriptions of complex technical systems. Predicate completion is not applicable to formula 4 because this formula is not solitary in P. Often circumscription [11, 12] can be used to complete the theory in cases where predicate completion is not applicable. We can however prove that the circumscription of formula 4 in P does not collapse to a first–order formula by exploiting the compactness theorem of first–order logic: There is no first–order formalization of the fact that $P(x)$ is valid if and only if $C_P(x)$ holds or there exists an y such that $C_p(y)$ holds and x is connected (directly or indirectly) to y.

Proposition 3.3 *The completion of formula 4 cannot be formalized by a set of first–order formulas.*

We use the following notations: For an interpretation M, we denote the extension of the predicate constant P in M by $M|P|$. For a set of first–order sentences $\mathcal{T}$ we denote the set of all models of $\mathcal{T}$ by $MOD(\mathcal{T})$.

Proof of proposition 3.3: We show that there is no set of first order sentences $\mathcal{T}$ such that such that $MOD(\mathcal{T})$ has the property:

$$(*)\begin{cases}\text{For every } M \in MOD(\mathcal{T})\text{: } x \text{ is in } M|P|\text{, iff } x \text{ is in}\\ M|C_P| \text{ or there exists an } y \text{ such that } y \text{ is in } M|C_P|\\ \text{and } (y,x) \text{ is in the transitive closure of } M|Conn|.\end{cases}$$

The proof is by contradiction. Suppose $\mathcal{T}$ is such a theory. Let us define a set of frames $\{M^n | n \in \mathbf{N}\}$. The domain of M^k is the set $\{1,\ldots,k\}$. Further we define $M^k|P| := \{1,\ldots,k\}, M^k|C_P| := \{1\}, M^k|Conn| := \{(i,i+1)|1 \le i < k\}$. Then, obviously M^k satisfies $(*)$ for every $k \in \mathbf{N}$. Now let

$$\phi_k := P(x) \wedge \neg(\exists x_1 \ldots \exists x_k\, C_P(x_1) \wedge x_k = x \wedge Conn(x_1,x_2) \wedge \ldots \wedge Conn(x_{k-1},x_k)).$$

The formula ϕ_k expresses that P holds for x but x is not reached by k times following the relation $Conn$ from a point, where C_P holds. Let $\mathcal{T}' := \mathcal{T} \cup \{\phi_k | k \ge 2\}$. We have to show that every finite subset $\mathcal{T}_\subset$ of $\mathcal{T}'$ is satisfiable. For a given finite subset $\mathcal{T}_\subset$ there is a largest number j such that ϕ_j is contained in $\mathcal{T}_\subset$. For this j we have $\mathcal{T}_\subset \subseteq \underbrace{\mathcal{T} \cup \{\phi_k | 2 \le k \le j\}}_{\mathcal{T}_j}$. The interpretation (M^{j+1},β) with $\beta(x) = j+1$ is obviously a model of $\mathcal{T}_j$ because the frame M^{j+1} satisfies $(*)$ and $j+1$ applications of $Conn$ are needed to get from 1 to $j+1$, so that all ϕ_k for $2 \le k \le j$ are satisfied. Thus $\mathcal{T}_j$ has a model, which is also a model of $\mathcal{T}_\subset$, because $\mathcal{T}_\subset \subseteq \mathcal{T}_j$. Since every finite subset of $\mathcal{T}'$ has a model, by compactness theorem, we infer that $\mathcal{T}'$ itself has a model. However, this model does not satisfy $(*)$, because for some x, $P(x)$ holds, but it is not connected to a point where $C_P(x)$ holds. Since this model of $\mathcal{T}'$ is also a model of $\mathcal{T}$ (because $\mathcal{T} \subseteq \mathcal{T}'$), we have found a contradiction to the assumption that all models of $\mathcal{T}$ satisfy $(*)$.

Q.E.D.

Because of these problems and efficiency considerations we restrict the language of our system descriptions to fixed domain theories [10, 13], i.e. first order theories without function symbols in which the unique name assumption and the domain closure assumption hold. We reduce these fixed domain theories to sets of ground clauses by instantiating all quantifiers. Then we apply the propositional variant of predicate completion [2] to the resulting propositional problem.

We can dramatically reduce the size of the propositional representation if we eliminate the database predicates during the instantiation. As stated above we assume the CWA for the database DB. The following algorithm converts a fixed domain theory into a set of ground clauses and eliminates all facts from the clauses, whose truth value is defined by the database.

Algorithm 3.4 *Conversion of a Fixed Domain Theory into Clause Form.*
Let $\mathcal{S} = \{F_1,\ldots,F_n\}$ be a fixed domain theory. Let $\{c_1,\ldots,c_t\}$ be the set of all constants occurring in $\mathcal{S}$. For every formula $F \in \mathcal{S}$ do the following:

1. *Convert F into a set of propositional clauses $\mathcal{F}$, i.e. a set of clauses, in which the literals do not contain any variables:*
   (a) *Instantiate every existential quantifier by replacing $\exists x\varphi$ with $\bigvee_{c\in\{c_1,\ldots,c_t\}} \varphi_{[x/c]}$.*
   (b) *Instantiate every universal quantifier by replacing $\forall x\varphi$ with $\bigwedge_{c\in\{c_1,\ldots,c_t\}} \varphi_{[x/c]}$.*
   (c) *Convert the resulting propositional formula into conjunctive normal form.*
   (d) *Convert the formula into a set of clauses $\mathcal{F}$.*
2. *For every atom a which is an instance of a predicate defined by the database DB do the following:*
   - *If $a \in DB$ then delete all clauses from $\mathcal{F}$ which contain a (they are trivially satisfied) and delete*

$\neg a$ *from all clauses in which it occurs (because* $\neg a$ *contradicts* $\mathcal{S}$*).*

- *If* $a \notin DB$ *then delete all clauses from* $\mathcal{F}$ *which contain* $\neg a$ *and delete* a *from all clauses in which it occurs.*

Example 3.5 *Let us study the effect of this algorithm on the axiom A2 given the database DB as defined in section 2.*

$$A2: \quad \forall c_1 \forall c_2 \forall c_{Sender}\, Conn(c_1, Up, c_2, Down) \wedge Signal(c_1, Up, Alive(c_{Sender})) \rightarrow Signal(c_2, Down, Alive(c_{Sender})))$$

In the following, we denote the set of constant symbols occuring in $\mathcal{S}$ *by C. Instantiation of the quantifiers yields*

$$\bigwedge_{c_1 \in C} \bigwedge_{c_2 \in C} \bigwedge_{c_{Sender} \in C} (Conn(c_1, Up, c_2, Down) \wedge Signal(c_1, Up, Alive(c_{Sender})) \rightarrow Signal(c_2, Down, Alive(c_{Sender})))$$

Conversion into conjunctive normal form:

$$\bigwedge_{c_1 \in C} \bigwedge_{c_2 \in C} \bigwedge_{c_{Sender} \in C} (\neg Conn(c_1, Up, c_2, Down) \vee \neg Signal(c_1, Up, Alive(c_{Sender})) \vee Signal(c_2, Down, Alive(c_{Sender})))$$

Conversion into clause form:

$$\mathcal{F} : \{\{\neg Conn(c_1, Up, c_2, Down), \neg Signal(c_1, Up, Alive(c_{Sender})), Signal(c_2, Down, Alive(c_{Sender}))\} \mid c_1, c_2, c_{Sender} \in C\}$$

Let us finally study the effect of step 2 on some clauses from $\mathcal{F}$*:*

$$\{\neg Conn(ML16, Up, Bsc1, Down), \neg Signal(ML16, Up, Alive(Bts20)), Signal(Bsc1, Down, Alive(Bts20))\}$$

The above clause contains only one instance of a database predicate, namely $\neg Conn(ML16, Up, Bsc1, Down)$*. Since* $Conn(ML16, Up, Bsc1, Down) \in DB$*, following algorithm 3.4 we have to delete* $\neg Conn(ML16, Up, Bsc1, Down)$ *from the clause resulting in the simplified clause*

$$\{\neg Signal(ML16, Up, Alive(Bts20)), Signal(Bsc1, Down, Alive(Bts20))\}$$

Finally we consider

$$\{\neg Conn(ML16, Up, ML20, Down), \neg Signal(ML16, Up, Alive(Bts17)), Signal(ML20, Down, Alive(Bts17))\}$$

Since $Conn(ML16, Up, ML20, Down) \notin DB$ *and* $Conn(ML16, Up, ML20, Down) \notin CWA(DB)$ *we can delete this clause, because* $CWA(DB) \subseteq \mathcal{S}$*, therefore every model M of* $\mathcal{S}$ *must make* $\neg Conn(ML16, Up, ML20, Down)$ *true, which subsumes the clause.*

In our application domains (circuit diagnosis, alarm correlation in cellular networks) the size of the propositional representations generated by algorithm 3.4 is linear in the size of the system under consideration[2].

We can now apply the propositional variant of predicate completion to the propositional clause set. In the propositional case we compute a completion formula for every instance of the predicate we want to complete, e.g. we compute separate completion formulas for $Signal(Bsc1, Down, Alive(Bts20))$ and $Signal(Bsc1, Down, Alive(Bts19))$. Note, that in the propositional version, the condition that the clauses must be solitary is satisfied by every system description: A clause containing a propositional atom a must not contain other positive occurrences of a (this is trivial, because a clause is a set) and it must not contain $\neg a$ (this is also trivial because a clause containing a and $\neg a$ is a tautology and can be omitted).

Note, that it does not contradict proposition 3.3 that we can complete the propositional version: Although propagation is in general not formalizable in first–order logic, propagation for a given topology (as represented by the propositional clause set) is easily expressible. This observation is in line with comments by Aho and Ullman concerning the transitive closure [1].

The next step is to apply our efficient consistency–based algorithm to the completed propositional clause set. The basic algorithm is outlined in the next section. Later in the paper, we discuss our techniques for exploiting the structure of the horn clauses of the system description, as well as a new result, which allows us to exploit the fact that the original system description was a set of horn clauses when reasoning with the completed system description (which is no longer horn).

[2] In the case of the cellular networks this is due to the fact that the length of the branches (as shown in figure 1) is bounded by some technological constant. If we consider branches of arbitrary lengths this no longer holds.

4 Computing Diagnoses

Diagnoses are computed using the DRUM–II algorithm, which we discussed in great detail in [5, 6]. Here, we will only describe the basic ideas necessary to understand the focussing techniques presented in the next section. DRUM–II computes minimal diagnoses by computing models of $\mathcal{S} \cup Obs$, in which the set of components assumed Ok is maximal, or equivalently, in which the extension of $Ab \equiv \neg Ok$ is minimal.

Definition 4.1 *Given models M_1 and M_2, we define $M_1 \leq_{Ab} M_2$, iff $M_1|Ab| \subseteq M_2|Ab|$.*

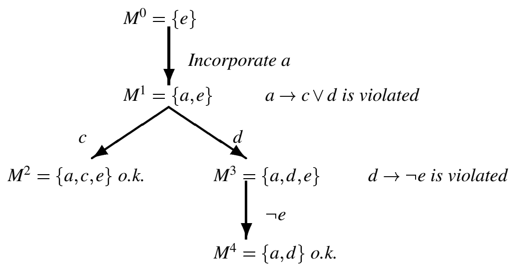
The minimal diagnoses corrspond to the $\leq_{Ab}$–minimal models of $\mathcal{S} \cup Obs$.

Proposition 4.2 *A set Δ of components is a minimal diagnosis of $(\mathcal{S}, Comp, Obs)$, iff there exists a $\leq_{Ab}$–minimal model M of $\mathcal{S} \cup Obs$, such that $\Delta = M|Ab|$.*

To prove this proposition, we have to assume that the predicate Ab is only applied to components, an assumption which is valid in model–based diagnosis. The DRUM–II algorithm computes the minimal models incrementally. It uses a model of $\mathcal{S}$ to focus the search for minimal models of $\mathcal{S} \cup Obs$. DRUM–II starts with a model M^0 of the theory $\mathcal{S} \cup \{Ok(c) | c \in Comp\}$. Then it adds the observations to M^0 giving M^1.

The interpretation M^1 possibly violates some clauses from $\mathcal{S}$. The basic idea of the DRUM–II algorithm is to find a model of $\mathcal{S}$ by iteratively inverting literals in the interpretation until a model of $\mathcal{S} \cup Obs$ is found. In contrast to GSAT [15] the search for models by inversion of literals is however exhaustive in the sense that all possible inversions are considered and every literal is only considered for inversion once on a given branch of the search tree.

Example 4.3 *Consider $\mathcal{S} = \{a \to c \vee d, d \to \neg e\}$, in clause form $\{\{\neg a, c, d\}, \{\neg d, \neg e\}\}$, together with the initial model $M^0 = \{e\}$ and the observations $Obs = \{a\}$.*



The model M^1 contradicts the axiom $a \to c \vee d$. We can change it in two ways in order to make the axiom hold again. Either we assume c or d, because these are the other literals in the corresponding clause, which can make it true again. While assuming c creates a consistent model (M^2) of $\mathcal{S} \cup Obs$, assuming d violates the axiom $d \to \neg e$. Thus, we have to delete e in order to obtain the second model M^4.

The minimality of the models generated in this fashion is achieved by applying the changes first to those models, in which the extension of Ab is minimal. DRUM–II uses an equivalence class approach to reduce the number of minimal models which have to be computed. Due to space limitations we cannot provide the formalization of this algorithm here but have to refer the reader to [5, 6], where we define the algorithm and provide soundness and completeness results for fixed domain theories.

5 Exploiting Structural Knowledge

In [6] we have proposed a technique which exploits knowledge about the structure of set $\mathcal{S}$ of horn clauses to avoid useless computations. In this section we generalize our previous results and show how we can still use structural knowledge for focussing after we have completed the theory (so that the resulting theory is not horn).

Let us consider a set of Horn Clauses $\mathcal{S}$. Let $\{a_1, \ldots, a_n\}$ be the set of atoms occurring in $\mathcal{S}$. We say that a_j *depends directly on* a_i, if there is a clause containing a_j and $\neg a_i$.

Definition 5.1 a_j depends directly on a_i, *denoted by $a_i \to^1_{\mathcal{S}} a_j$, iff there is a clause $C \in \mathcal{S}$ with $\{a_j, \neg a_i\} \subseteq C$.*

We define the transitive closure of the direct dependency relation $\to^1_{\mathcal{S}}$ and call it $\to^+_{\mathcal{S}}$ (depends on).

Definition 5.2 *The relation $\to^+_{\mathcal{S}}$ is defined inductively by*

1. *If $a_i \to^1_{\mathcal{S}} a_j$, then $a_i \to^+_{\mathcal{S}} a_j$.*
2. *If $a_i \to^1_{\mathcal{S}} a_k$ and $a_k \to^+_{\mathcal{S}} a_l$, then $a_i \to^+_{\mathcal{S}} a_l$.*

We say that a_j depends on *a_i, iff $a_i \to^+_{\mathcal{S}} a_j$.*

In the following, we will denote the difference between two sets M and N by $Diff(M,N)$.

Definition 5.3 *Let M and N be sets. $Diff(M,N) := (M \backslash N) \cup (N \backslash M)$*

During the computation of minimal models/diagnoses different extensions of the Ok–predicate are assumed. When we change the extension of Ok, we want to detect, which other atoms are possibly affected by this change. The next proposition shows that some positive atoms need not be considered for inversion when the set of assumed Ok–atoms changes.

Proposition 5.4 *Let $\mathcal{S}$ be a system description and $\mathcal{S}_c$ its completed variant. Let A and B be sets of Ok–atoms such that $\mathcal{S} \cup A \not\models \bot$ and $\mathcal{S}_c \cup B \not\models \bot$. Let a be an atom.*

If $\mathcal{S} \cup A \models a$ and there is no $b \in Diff(A,B)$ such that $b \rightarrow^+_{\mathcal{S}} a$, then $\mathcal{S}_c \cup B \models a$.

Proof Sketch: The *positive* literal a follows from $S \cup A$, iff it holds in the minimal model of $S \cup A$. This minimal model coincides with the minimal model of $\mathcal{S}^+ \cup A$ ($\mathcal{S}^+$ consists of those clauses of $\mathcal{S}$, which contain a positive literal). Thus, $\mathcal{S}^+ \cup A \models a$. Thus, there is a proof for a, based only on the clauses in S^+. If we delete all unnecessary steps from this proof, only those steps remain, which derive a itself or literals on which a depends. This proof remains possible, if we replace A by B since we assume that A and B do not disagree on atoms, on which a depends. Thus, $\mathcal{S}^+ \cup B \models a$ and, since $\mathcal{S} \cup B$ is consistent, we conclude $\mathcal{S} \cup B \models a$. Finally, since $\mathcal{S} \subseteq \mathcal{S}_c$ we conclude $\mathcal{S}_c \cup B \models a$. Q.E.D.

This proposition is exploited in DRUM–II as follows: The minimal model M^0 of the original horn system description under the assumption that all components are working, i.e. $\mathcal{S} \cup \{Ok(c) | c \in Comp\}$, is used as starting point. For each literal a, which holds in this unique minimal model, we know that $\mathcal{S} \cup \{Ok(c) | c \in Comp\} \models a$. Now if we assume some faulty components, which are not connected to a by $\rightarrow^+_{\mathcal{S}}$, we know that a is still entailed and all branches searched by the DRUM–II algorithm, in which a is inverted, can be cut off.

Let us now consider which negative literals remain unchanged, when the extension of Ok is changed.

Proposition 5.5 *Let $\mathcal{S}$ be a system description and $\mathcal{S}_c$ its completed variant. Let A and B be sets of Ok–atoms such that $\mathcal{S} \cup A \not\models \bot$ and $\mathcal{S}_c \cup B \not\models \bot$. Let a be an atom such that $\mathcal{S} \cup A \not\models a$ and there is no $b \in Diff(A,B)$ such that $b \rightarrow^+_{\mathcal{S}} a$.*

If $Comp \backslash \{c | Ok(c) \in B\}$ is a minimal diagnosis of $(\mathcal{S}_c, Comp, Obs)$, then it is inconsistent with a, i.e. there is no model M of $\mathcal{S}_c$, such that $M|Ok| = \{c | c \in B\}$ and $M \models a$.

Proof Sketch: The proof is by contradiction.

1. First, with a similar argumentation as in the proof of proposition 5.4, we can show that $\mathcal{S} \cup B \not\models a$. Consequently, $\mathcal{S} \cup B \not\models Obs \cup \{a\}$.

2. Now let us assume that $\Delta := Comp \backslash \{c | Ok(c) \in B\}$ is a minimal diagnosis of $(\mathcal{S}_c, Comp, Obs)$ and M is a model of $\mathcal{S}_c$, such that $M|Ok| = \{c | c \in B\}$ and $M \models a$. Then $\mathcal{S}_c \cup B \cup Obs \cup \{a\}$ has a model M and is thus consistent. We also know that B is the maximal set, for which $\mathcal{S}_c \cup B \cup Obs \cup \{a\}$ is consistent (Proof: assume there exists a set $B' \supset B$ such that $\mathcal{S}_c \cup B' \cup Obs \cup \{a\}$ is consistent, then $\Delta' = Comp \backslash \{c | Ok(c) \in B'\}$ is a diagnosis and $\Delta' \subset \Delta$ which contradicts the assumption that Δ is a minimal diagnosis).

   Following Konolige ([9], theorem 4), we can conclude that B is a minimal set of abducibles, such that $\mathcal{S} \cup B \models Obs \cup \{a\}$. This is a contradiction with 1.

Q.E.D.

# Branches	DRUM–II	DRUM–II with $\rightarrow^+_{\mathcal{S}}$
2	76	54
3	103	59
4	130	64
$\vdots$	$\vdots$	$\vdots$
n	49 + 27n	49 + 5n

Table 1: Effect of exploiting $\rightarrow^+_{\mathcal{S}}$

DRUM–II exploits this proposition as follows: Assume that a literal a is not true in the model M^0 of the correctly functioning device. If we now consider a set Δ of components faulty, such that for all $c \in \Delta : Ok(c) \not\rightarrow^+_{\mathcal{S}} a$ then we need not consider assuming a, because if Δ is a minimal diagnosis, it is inconsistent with a.

The basic DRUM–II algorithm is already very efficient on the cellular network application (using the completed system description). Table 1 shows the number of flippings (inversions of literals) needed by DRUM–II to compute the diagnosis for a symptom, which can be explained by a single fault for a tree–structured network comparable to the one in figure 1. The running time for the basic DRUM–II algorithm is linear in the number of branches. Exploiting $\rightarrow^+_{\mathcal{S}}$ allows nearly constant running time. The 5 flippings per additional branch in the third column of table 1 are due to the fact that we count the unsuccessful first flipping of branches in the search tree which are terminated by applying the propositions 5.4 and 5.5.

In the domain of these tree-structured networks the effect of our focusing techniques is a linear efficiency gain. We have discussed the diagnosis of complex combinatorial circuits in [6]. In this domain, exploiting the relation $\rightarrow^+_{\mathcal{S}}$ makes circuits with up to 3000 components solvable in less than 1 second, which are not solvable in acceptable time by the basic DRUM–II algorithm.

6 Conclusion

In this paper we have introduced steps and techniques for the efficient diagnosis of technical systems consisting of networks of connected components. Intuitive descriptions of such systems (e.g. cellular networks) are often incomplete. The propagation axioms present in these descriptions cannot be completed in first–order logic. However, we can reduce the first–order system descriptions to propositional logic for a given topology. Our algorithm for this transformation eliminates the database predicates and generates compact propositional clause sets, which can be easily completed. Starting from a model of the correct behaviour of the system, the DRUM–II algorithm computes diagnoses based on the completed system description. If the original system description consists of horn clauses the efficiency of DRUM–II can be increased considerably by exploiting a relation $\rightarrow_S^+$ which describes the dependencies among the atoms in the system description.

References

[1] A. V. Aho and J. D. Ullman. Universality of data retrieval languages. In *Proc. 6th ACM Symposium on Principles of Programming Languages*, 1979.

[2] K. Clark. Negation as failure. In *Logic and Databases*, pages 293–322. Plenum, 1978.

[3] Luca Console, Daniele Theseider Dupré, and Pietro Torasso. On the relationship between abduction and deduction. *Journal of Logic and Computation*, 1(5):661–690, 1991.

[4] Luca Console and Pietro Torasso. A spectrum of logical definitions of model-based diagnosis. *Computational Intelligence*, 7(3):133–141, 1991.

[5] Peter Fröhlich and Wolfgang Nejdl. A model–based reasoning approach to circumscription. In *Proceedings of the 12th European Conference on Artificial Intelligence*, 1996.

[6] Peter Fröhlich and Wolfgang Nejdl. A static model-based engine for model-based reasoning. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97)*, Nagoya, Japan, August 1997.

[7] Peter Fröhlich, Wolfgang Nejdl, Klaus Jobmann, and Hermann Wietgrefe. Model-based alarm correlation in cellular phone networks. In *Fifth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, January 1997.

[8] M. R. Genesereth and N. J. Nilsson. *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann Publishers, Inc., 1987.

[9] Kurt Konolige. Abduction versus closure in causal theories. *Artificial Intelligence*, 53:255–272, 1992.

[10] W. Lukaszewicz. *Non–monotonic reasoning: formalization of commonsense reasoning*. Ellis Horwood, 1990.

[11] J. McCarthy. Circumscription - a form of non–monotonic reasoning. *Artificial Intelligence*, 13:27–39, 1980.

[12] J. McCarthy. Applications of circumscription to formalizing commonsense knowledge. *Artificial Intelligence*, 28:89–118, 1986.

[13] R. Reiter. Equality and Domain Closure in First–Order Databases. *Journal of the ACM*, 32:57–97, 1987.

[14] Raymond Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987.

[15] Bart Selman, Hector Levesque, and David Mitchell. A new method for solving hard satisfiability problems. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 440–446, San Jose, July 1992. Morgan Kaufmann Publishers, Inc.