

Model-Based Reliability and Diagnostic *

A Common Framework for Reliability and Diagnostics

Bernhard Anrig, Jürg Kohlas

Department of Informatics
University of Fribourg
CH-1700 Fribourg, Switzerland
bernhard.anrig@unifr.ch, juerg.kohlas@unifr.ch

July 21, 2003

Abstract

Technical systems are in general not guaranteed to work correctly. They are more or less reliable. One main problem for technical systems is the computation of the reliability of a system as studied in reliability theory. A second main problem for technical systems is the problem of diagnostic, i.e. explanations why something does not work. In fact, these problems are — in some sense — dual to each other.

In this paper, we will use the concept of probabilistic argumentation systems PAS for modeling the system description as well as observation and specifications of behaviour in one common framework. We show that PAS provide a framework which allows to formulate and solve reliability and diagnostic problems, and all concepts for these two problems can clearly be defined therein. Using PAS, reliability and diagnostic can be considered as dual problems. PAS offer one common strategy for computing answers to the questions in these different situations.

1 Introduction and Overview

Technical systems are in general not guaranteed to work correctly. They are more or less reliable. One main problem for technical systems is the computation of the reliability of a system. The reliability depends on various factors like the quality and the age of components, the complexity of the system, etc. The reliability of a system conveys some information about the behavior of the system in the future, based on information about the components, for example probabilistic information about the reliability over time.

*Research supported by grant No. 2000-061454.00 of the Swiss National Foundation for Research.

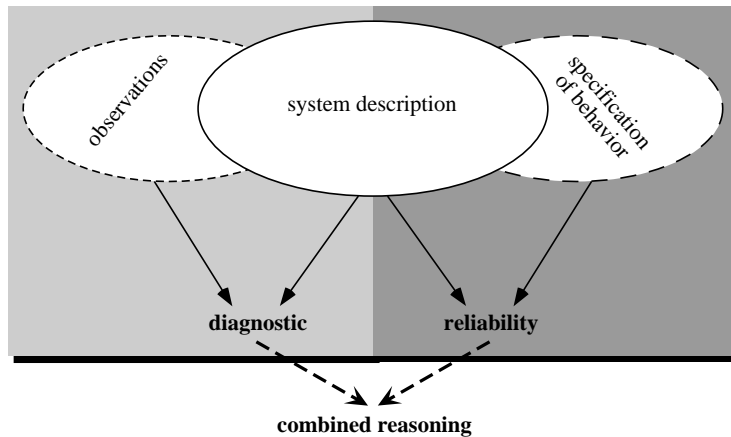


Figure 1: Reliability and diagnostic with respect to the available knowledge.

A second main problem for technical systems is the problem of diagnostic. Here, the problem is to explain the behavior of the system, usually based on measurements and observations of some parts of the system, together with the system description in some framework. That is, we have to identify those system components which are functioning abnormally and therefore explain the discrepancy between the predicted and the observed behavior of the system. The actual observations and the description of the system are the only ingredients for the computation of the diagnoses. Additionally, if probabilistic knowledge is available about the different operating modes of the components, then the likelihood of the system states can be defined and prior as well as posterior probabilities can be computed for the set of possible system states.

The two main problems of reliability and diagnostic depend both on a formalization of the system in some framework together with either observations, measurements, or requirements. The typical situation is presented in Fig. 1.

In this article we will use the concept of probabilistic argumentation systems PAS^{23, 22, 25} for modeling the system as well as observation and specifications of desired behaviour in one common framework. The goal of a PAS is to derive arguments in favor and against hypotheses of interest. An argument is a defeasible proof built on uncertain assumptions, i.e. a chain of deductions based on assumptions that makes the hypothesis true. If probabilistic information is available, a quantitative judgement of the situation is obtained by considering the probabilities that the arguments are valid (the reliability of proof). The resulting degrees of support and possibility correspond to belief and plausibility, respectively, in the sense of Dempster-Shafer theory of evidence.^{41, 34} In fact, PAS combines the strengths of logic and probability in one common framework.

In this paper we show that probabilistic argumentation systems are a framework which allows to formulate and solve both reliability and diagnostic problems. Several examples will illustrate the use of argumentation systems for both kinds of problems. The framework of probabilistic argumentation system offers thus a common framework for the different situations discussed in reliability and model-based diagnostic. It will especially permit to consider one common strategy for computing answers to the questions in these different situations.

The main element for both problems is a description of the system. We will focus here on a formalization using logic. In the case of reliability, we may have a specification which describes the goals which have to be fulfilled by the system. This information will be used to derive the structure function from the system description. Different specifications may lead to different structure functions. Even in the absence of an explicit specification of a reliability requirement, we may deduce a structure function by assuming that the system should be functioning at least if all components are working. This permits to detect redundancies of the system.

On the other hand, in the case of diagnostic, some observations of the system behavior may indicate that the system is not working as it is supposed to be. This information — together with the system description — allows then to determine possible diagnoses of the system, i.e. minimal sets of components whose malfunctioning “explains” the wrong behaviour of the whole system.

Fig. 1 shows that the two problems are connected by the common information contained in the system description. Then, by adding observations one can do diagnostic reasoning, or — by specifying desired behavior — one can compute the reliability of the system. Further, combining these approaches yields more information for both problems and a combined reasoning becomes possible.

The paper is organized as follows: In Section 2, we explain what we mean by model-based reliability using two examples. Section 3 explains then the main concepts of probabilistic argumentation systems PAS. We show how logic and probability are integrated in this common framework. A short introduction into the computational theory of PAS is added.

In Section 4, the concept of PAS is applied to reliability analysis and especially to deduce structure functions from the system description in PAS. Section 5 discusses the “dual” problem of model-based diagnostic modeled in PAS. Finally, in Section 6 we show that diagnostic and reliability problems may be present simultaneously in some situations.

On the side of probabilistic argumentation and especially diagnostic, a lot of work has been done; the main influence clearly was Reiter⁴⁰ followed by De Kleer et al.,^{17,18} Davis,¹⁶ Laskey & Lehner³⁶ and Provan.³⁸ For a short survey of related approaches see Haenni et al.,²² for a discussion of a general approach see Kohlas et al.³¹ Lately, the concept of process algebras has been used as a general formalization for systems diagnosis.¹² To the best of our knowledge, there was no discussion about the duality of the approaches of diagnostics and reliability besides some work of Provan³⁹ and Kohlas et al.,³⁰ which has been some inspiration for the present work.

2 Reliability

In the sequel we will use classical notation for elements of combinatorial reliability theory.^{9,10,28} Section 2.1 is a short introduction in this area. In contrast to this well-known theory, we will introduce in Section 2.2 the concept of model-based reliability.

2.1 Combinatorial Reliability

In binary combinatorial reliability, a system is assumed to be composed of a number of different components. Each component may be in one of two states: either it is intact or it is down. The whole system itself is also either functioning or down, depending on the states of its components. In order to formulate this, binary variables x_i are associated to components $i = 1, 2, \dots, n$ of the system. These variables represent the system state,

$$x_i = \begin{cases} \top & \text{if the component with number } i \text{ works,} \\ \perp & \text{otherwise,} \end{cases} \quad (1)$$

Let \mathbf{x} be the vector (x_1, x_2, \dots, x_n) of the component states. This state-vector has 2^n possible values. The set of all possible states can be decomposed into two disjoint subsets, the set S_\top of working states, for which the system as a whole is assumed to be functioning, and the set S_\perp of down-states, for which the system is supposed to not work properly. The corresponding system state is denoted by x . Its dependence on the state-vector \mathbf{x} is described by a Boolean function ϕ , defined as

$$x = \phi(\mathbf{x}) = \begin{cases} \top & \text{if } \mathbf{x} \in S_\top, \\ \perp & \text{if } \mathbf{x} \in S_\perp. \end{cases} \quad (2)$$

The Boolean function ϕ is called the *structure function* of the system. In combinatorial reliability it is assumed to be given and it forms the base for reliability analysis.

The structure function ϕ is usually assumed to be *monotone*. That is, if $\mathbf{x}_1 \leq \mathbf{x}_2$, then $\phi(\mathbf{x}_1) \leq \phi(\mathbf{x}_2)$. This makes sense in most cases: It says that a system, which is working, is still working if one or more components are repaired (changed from down state to working state). Or, alternatively, a monotone system remains down, if one or more components fail.

For monotone structure function, a subset $P \subseteq \{1, 2, \dots, n\}$ of components is called a *path*, if $\phi(\mathbf{x}) = \top$ for all state-vectors \mathbf{x} for which the components of the set P are working, $x_i = \top$ for all $i \in P$. That is, the elements of a path are sufficient to guarantee the functioning of the system, regardless of the state of the components outside the path. The family of paths is *upward closed*: If P is a path, and P' a superset of P , then P' is a path. This follows from the monotonicity of the structure function. We assume that the set $\{1, 2, \dots, n\}$ of all components is a path (otherwise the system would never be functioning). A system, for which the set of all components is the only path, is called a *series system*. A path P is called *minimal*, if no proper subset of P is still a path. Since the paths are upwards closed it is sufficient to know all minimal paths. Let \mathcal{P} denote the set of minimal paths. This set determines the structure function,

$$\phi(\mathbf{x}) = \bigvee_{P \in \mathcal{P}} \bigwedge_{i \in P} x_i. \quad (3)$$

This logical formula expresses the fact, the system is working, if all components of at least one minimal path are working (we refer to Section 3.1 for a short review of propositional logic).

A subset of components C is called a *cut*, if $\phi(\mathbf{x}) = \perp$ for all state-vectors \mathbf{x} for which the components of the set C are down, $x_i = \perp$ for all $i \in C$. That

is, the elements of a cut are sufficient to guarantee that the system is down, regardless of the state of the components outside the cut. Again, the family of cuts is upwards closed. If C is a cut and C' a superset of C , then C' is a cut. We assume that the set of all components is a cut (otherwise the system would never be down). If it is the only cut, then we have a *parallel system*. A cut C is called *minimal*, if no proper subset of C is a cut. Again, the knowledge of all minimal sets is sufficient to define the structure function. Let \mathcal{C} denote the set of all minimal cuts, then

$$\phi(\mathbf{x}) = \bigwedge_{C \in \mathcal{C}} \bigvee_{i \in C} x_i. \quad (4)$$

The system is functioning, if in all minimal cuts at least one component is working. This guarantees that there is no cut.

If for every component $i = 1, 2, \dots, n$ its respective probability p_i of functioning correctly is defined, then the probability that the system is functioning can be computed (assuming the components to be stochastically independent). In fact, $\phi(\mathbf{x})$ is a random variable, and the probability p that the system is functioning is

$$p = E(\phi(\mathbf{x})) = h(\mathbf{p}). \quad (5)$$

Here, \mathbf{p} denotes the vector (p_1, p_2, \dots, p_n) of probabilities. $h(\mathbf{p})$ is called the reliability function.

The practical computation of $h(\mathbf{p})$ is a nontrivial task. A classical approach consist in transforming (3) into a disjunction of disjoint terms, whose probabilities can be easily computed and then summed up.^{1,26,27,5} We refer to Ref. 9,10,28 for a discussion of this problem. Besides, new promising approaches focus on more general formulas as internal structures of the algorithm.^{14,13}

2.2 Model-Based Reliability

The structure function describes the conditions under which a system is functioning, depending on the states of its components. It is already a compilation of knowledge about the system and its structure. In this section we shall illustrate another approach, where a more physical description of a system is given. Additionally, a specification of the desired behavior of the system is given. These two elements will then allow the deduction of a structure function and its associated reliability function. This approach to reliability is more in the spirit of model-based diagnostics and is therefore called model-based reliability. The discussion in this section will be informal. The formal theory for this approach will be developed in the following sections. For further examples see also Ref. 7,6.

Example 1: Detector of Power Failure Consider a device which should detect a power failure. More precisely, the device watches a Boolean value and reports \top , if the value vanishes (becomes \perp). A simple version of such a device is depicted in Fig. 2.

The functionality of this device can be described with propositional logic. Let *in* be the variable which denotes the state of the input and *out* the variable which represents the output. Both variables are binary, i.e. represent the boolean values true or false respectively. Further, there are two internal variables x_1 and x_2 , also binary. For every component A , B or C , there is a respective binary variable ok_A , ok_B , and ok_C which describes the working mode of the component.

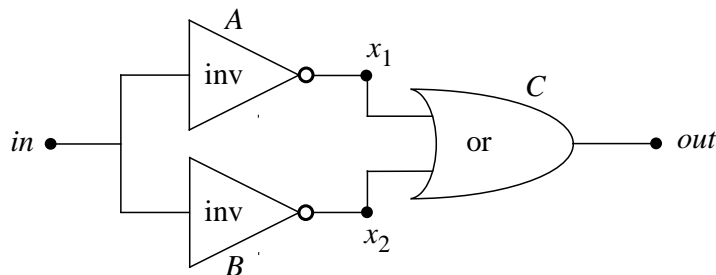


Figure 2: A simple device

Consider the inverter A : if it works correctly (ok_A is true), then its input is the negation of its output. We express this by the formula $in \leftrightarrow \neg x_1$, which reads as x_1 is true if and only if in is false. So the entire information is modeled as the logical implication $ok_A \rightarrow (in \leftrightarrow \neg x_1)$ which reads as *if ok_A is true then $in \leftrightarrow \neg x_1$* . This is one possible mode for the component. Note that so far nothing is said about the behavior of the component, if it is down (ok_A is false). There are several possibilities. One is that in this case the output of the component is always false. This possibility is described by $\neg ok_A \rightarrow \neg x_1$. Yet there may be further failure modes. For example, the output may be stuck at true, or the input may equal the output. Multiple failure modes are best described by variables with multiple states, which in this context are called set constraint variables, cf. Ref. 8 for further information.

For the component B , the same specification can be applied. For the or-gate, if it works correctly, then the output is true if at least one of its inputs is. This is described by the formula $out \leftrightarrow x_1 \vee x_2$. If it does not work correctly, then we assume that its output is zero. So the whole information about the device is modeled by six implications:

$$\begin{array}{lll}
 ok_A & \rightarrow & (in \leftrightarrow \neg x_1) & \neg ok_A & \rightarrow & \neg x_1 \\
 ok_B & \rightarrow & (in \leftrightarrow \neg x_2) & \neg ok_B & \rightarrow & \neg x_2 \\
 ok_C & \rightarrow & (out \leftrightarrow x_1 \vee x_2) & \neg ok_C & \rightarrow & \neg out
 \end{array} \tag{6}$$

This is the *system description*. In the sequel we denote the set of formulas describing the system by Σ .

We add now a specification of what we expect from the system. We expect, that negative (false) input is detected. This is expressed by

$$\neg in \rightarrow out. \tag{7}$$

However, this is a weak requirement. It does not exclude that out becomes true, even if in is true. More stringent would be the requirement

$$\neg in \leftrightarrow out. \tag{8}$$

This states that there is an alarm (out) if, and only if, in is false.

We may now ask under which states, described by the variables ok_A , ok_B , and ok_C , each one of these specifications is fulfilled. This defines then the *structure function* of the system associated with the corresponding specification of desired system behavior. We shall see in the next section, that it is a well-defined problem of propositional logic to deduce these structure functions from

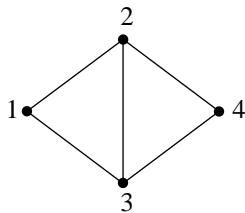


Figure 3: Simple Bridge Network

the system description and the specifications of desired behavior. If probabilities for the working modes of the three components are available, then the reliability functions corresponding to the specifications can be computed too. \ominus

Example 2: Stochastic Net Structure This example places the classical problem of network reliability in the framework of model-based reliability. It demonstrates that classical problems of reliability are covered by this approach. Consider the simple network of Fig. 3. For the description in propositional logic, we model this examples as follows: a node i is modeled by a binary variable (proposition) x_i . An edge between the nodes i and j is described by a variable ok_{ij} where the variable is true if this edge is available, i.e. if it is connecting the respective nodes, and false otherwise. We consider only undirected edges in this example. If an edge (i, j) is connecting the two nodes i and j , then $x_i \leftrightarrow x_j$. That is, the two nodes have the same logical value.

We have four nodes $N = \{1, 2, 3, 4\}$ and five edges

$$E = \{(1, 2), (1, 3), (2, 3), (2, 4), (3, 4)\}. \quad (9)$$

So the knowledge is modeled by five implications $ok_{ij} \rightarrow (x_i \leftrightarrow x_j)$ for every edge $(i, j) \in E$. Explicitly, the system description Σ looks as follows:

$$\Sigma = \left\{ \begin{array}{l} ok_{12} \rightarrow (x_1 \leftrightarrow x_2), \quad ok_{13} \rightarrow (x_1 \leftrightarrow x_3), \quad ok_{23} \rightarrow (x_2 \leftrightarrow x_3), \\ ok_{24} \rightarrow (x_2 \leftrightarrow x_4), \quad ok_{34} \rightarrow (x_3 \leftrightarrow x_4). \end{array} \right\} \quad (10)$$

Usually, there is also some probabilistic information given about the reliability of the edges, often as a probability $p(ok_{ij})$ for $(i, j) \in E$. Then, (E, N, p) is called a *stochastic net structure*.¹⁰

Given such a network, several questions can be of interest, for example:

Two-terminal reliability: Are nodes 1 and 4 connected?

Source-to-net reliability: Is node 1 connected with all other nodes?

Overall reliability: Are all nodes connected with each other?

A different specification or requirement corresponds to each one of these reliability problems. The two-terminal reliability problem for examples is specified by $x_1 \rightarrow x_4$, the source-to-net reliability by $(x_1 \rightarrow x_2) \wedge (x_1 \rightarrow x_3) \wedge (x_1 \rightarrow x_4)$. The overall reliability corresponds in the present example to the source-to-net reliability as there are only undirected edges.

Again, we ask under what states these requirements are fulfilled. This will define the corresponding structure functions. \ominus

These two examples show how the physical behavior of systems and the required behavior can be described in the language of propositional logic. We

shall examine this structure in the following section in a general context and show how reliability and also diagnostic analyses can be carried out using this framework.

3 Probabilistic Argumentation Systems

The examples considered in the previous section are modeled by instances of so-called probabilistic argumentation systems. These systems were developed as general formalisms for expressing uncertain and partial knowledge and information in artificial intelligence. They combine in an original way logic and probability. Logic is used to derive arguments and probability serves to compute the reliability or likelihood of these arguments. These systems can be used for model-based diagnostics in the sense of Reiter⁴⁰ as has been demonstrated in Ref. 2,31. Here we shall show how they relate to reliability theory.

Argumentation systems can be based on different logics, see for example Ref. 31 for a general framework. For the sake of simplicity we limit ourselves here to the case of propositional logic. In this section we give a short introduction into propositional probabilistic argumentation systems. For a more detailed presentation of the subject and further references we refer to Ref. 23. We remark also that such systems have been implemented in a system called ABEL, Assumption-Based Evidential Language,⁶ which is available on the internet (cf. Ref. 21), and which can be used to compute the examples of this paper as well as other problems.

We start with a short reminder of propositional logic, introduce then propositional argumentation systems and the basic concepts related to them, define the probability structure associated with them and give an overview of the inference methods for argumentation systems.

3.1 Propositional Logic

Propositional logic deals with declarative statements that can be either true or false. Such statements are called *propositions*. Let $P = \{p_1, \dots, p_n\}$ be a finite set of propositions. The symbols $p_i \in P$ together with \top (tautology) and \perp (falsity), are called *atoms* or *atomic formulas*. Compound formulas are built by the following syntactic rules:

- (1) atoms;
- (2) if γ is a formula, then $\neg\gamma$ is a formula;
- (3) if γ and δ are formulas, then $(\gamma \wedge \delta)$, $(\gamma \vee \delta)$, $(\gamma \rightarrow \delta)$, and $(\gamma \leftrightarrow \delta)$ are formulas.

Often, unnecessary parentheses can be omitted, e.g. $\gamma \wedge \delta$ instead of $(\gamma \wedge \delta)$. Furthermore, by assigning priority in decreasing ordering to \neg , \wedge , \vee , \rightarrow , some other parentheses can be eliminated, e.g. $\gamma \rightarrow \delta \wedge \lambda$ instead of $\gamma \rightarrow (\delta \wedge \lambda)$. The set \mathcal{L}_P of all formulas generated by the above recursive rules is called *propositional language* over P . A formula $\gamma \in \mathcal{L}_P$ is also called a *propositional sentence*.

A *literal* is either an atom p_i or the negation of an atom $\neg p_i$. A *term* is a conjunction of literals, and a *clause* is a disjunction of literals. Usually we

will consider only *proper* terms and clauses, where every atom occurs at most once in a term or a clause, either positive or negative, and neither \top nor \perp does occur. Additionally, we say that \top is a (empty) proper term and \perp a (empty) proper clause. The set of all proper terms is denoted by $C_P \subseteq \mathcal{L}_P$, and the set of all proper clauses by $D_P \subseteq \mathcal{L}_P$.

A clause (and similarly a conjunction) is interpreted also as a set of its literals; this allows to simplify the notations and algorithms. So for example for terms $\alpha, \alpha' \in \mathcal{L}_P$, we have $\alpha \subseteq \alpha'$ if and only if $\alpha' = \alpha \wedge \beta$ for some $\beta \in \mathcal{L}_P$. As a special case, $\perp \subseteq \alpha \subseteq \top$ for every clause α and $\perp \supseteq \beta \supseteq \top$ for every term β .

The meaning of a propositional sentence is obtained by assigning truth values true and false (represented by “1” and “0” respectively) to the propositions. The truth value of compound formulas can then recursively be obtained according to well-known truth tables.

An assignment of truth values to the elements of a set $P = \{p_1, \dots, p_n\}$ is called *interpretation* relative to P . $N_P = \{0, 1\}^n$ denotes the set of all 2^n different interpretations. Every interpretation $\mathbf{x} \in N_P$ can be seen as a point or a vector $\mathbf{x} = (x_1, \dots, x_n)$ in the n -dimensional binary product space N_P . Each $x_i \in \{0, 1\}$ denotes a binary variable which is associated to the corresponding proposition p_i .

Let \mathbf{x} be an arbitrary interpretation relative to P . If $\gamma \in \mathcal{L}_P$ evaluates to 1, then \mathbf{x} is called a *model* of γ . Otherwise, \mathbf{x} is a *counter-model* of γ . The set of all models of γ is denoted by $N_P(\gamma) \subseteq N_P$. If $N_P(\gamma) = \emptyset$, then γ is called *unsatisfiable*. Otherwise, it is called *satisfiable*.

The notion of models and counter-models links propositional logic to the Boolean algebra of subsets of interpretations:

- (1) $N_P(\perp) = \emptyset$,
- (2) $N_P(\top) = N_P$,
- (3) $N_P(\neg\gamma) = N_P - N_P(\gamma)$,
- (4) $N_P(\gamma \wedge \delta) = N_P(\gamma) \cap N_P(\delta)$,
- (5) $N_P(\gamma \vee \delta) = N_P(\gamma) \cup N_P(\delta)$.

A propositional sentence γ *entails* another sentence δ (denoted by $\gamma \models \delta$) if and only if $N_P(\gamma) \subseteq N_P(\delta)$. In that case, δ is also called a *logical consequence* of γ . For example, $\gamma \wedge \delta \models \gamma \vee \delta$. Sometimes, it is convenient to write $\mathbf{x} \models \gamma$ instead of $\mathbf{x} \in N_P(\gamma)$. Also we write $\gamma \models \perp$ if γ is not satisfiable. Furthermore, two sentences γ and δ are *logically equivalent* (denoted by $\gamma \equiv \delta$), if and only if $N_P(\gamma) = N_P(\delta)$. For example, $\gamma \rightarrow \delta \equiv \neg\gamma \vee \delta$. Note that logically equivalent sentences represent exactly the same information.

3.2 Basic Concepts of Argumentation Systems

Consider two finite sets $P = \{p_1, p_2, \dots, p_m\}$ and $A = \{a_1, a_2, \dots, a_n\}$ of propositional variables with $A \cap P = \emptyset$, the elements of P are called *propositions*, the elements of A *assumptions*.

We consider a fixed set of formulas $\Sigma \subseteq \mathcal{L}_{A \cup P}$ called the *knowledge base*, which models the information available; sets of formulas are interpreted conjunctively, i.e. $\Sigma = \bigwedge \{\xi \in \Sigma\}$. We assume that this knowledge base is satisfiable.

A triple $\mathcal{AS}_P = (\Sigma, A, P)$ is then called a *propositional argumentation system PAS*.

The elements of N_A are called *scenarios* or *system states*. A scenario represents a specification of all values of the assumptions in A . Given a scenario $\mathbf{s} = (s_1, \dots, s_n) \in N_A$ define the respective formula in \mathcal{L}_A by

$$\Upsilon_A(\mathbf{s}) := \bigwedge_{s_i=0} a_i \wedge \bigwedge_{s_i=1} \neg a_i \quad (11)$$

such that $N(\Upsilon_A(\mathbf{s})) = \mathbf{s}$. We abbreviate $\Upsilon_A(\mathbf{s}), \Sigma \models h$ by $\mathbf{s}, \Sigma \models h$.

Given the information encoded in Σ , we are interested first in the states which are excluded by the actual knowledge. Second, we would like to compute scenarios, which, together with the knowledge base imply some hypotheses h . Formally, the following concepts will be of interest:

Inconsistent Scenarios: $CS_A(\Sigma) := \{\mathbf{s} \in N_A : \mathbf{s}, \Sigma \models \perp\}$

Quasi-Supporting Scenarios of a formula $h \in \mathcal{L}_N$:
 $QS_A(h, \Sigma) := \{\mathbf{s} \in N_A : \mathbf{s}, \Sigma \models h\}$

Supporting Scenarios of a formula $h \in \mathcal{L}_N$:
 $SP_A(h, \Sigma) := QS_A(h, \Sigma) - CS_A(\Sigma)$

Possible Scenarios for $h \in \mathcal{L}_N$: $PL_A(h, \Sigma) := SP_A^c(\neg h, \Sigma)$

Inconsistent scenarios are in contradiction with the knowledge base and therefore to be considered as excluded by the knowledge. CS_A is also called conflict set. Supporting scenarios for a formula h are scenarios, which, together with the knowledge base imply h and are consistent with the knowledge. So, under supporting scenarios, the hypothesis h is true. Possible scenarios for h are scenarios, which do not imply $\neg h$ and thereby do not refute h . Quasi-supporting scenarios for h are the union of supporting scenarios and inconsistent scenarios. They are important for technical reasons only.

Scenarios are the basic concepts of assumption-based reasoning. However, sets of inconsistent, quasi-supporting, supporting and possible scenarios may become very large. Therefore, more economical, logical representations of these sets are needed. For this purpose, the following concepts are defined:

Conflicts: $\alpha \in C_A$ such that $N_A(\alpha) \subseteq CS_A(\Sigma)$

Quasi-Supporting Argument for h :
 $\alpha \in C_A$ such that $N_A(\alpha) \subseteq QS_A(h, \Sigma)$

Supporting Argument for h : $\alpha \in C_A$ such that $N_A(\alpha) \subseteq SP_A(h, \Sigma)$

Possible Argument for h : $\alpha \in C_A$ such that $N_A(\alpha) \subseteq PL_A(h, \Sigma)$

Conflicts or inconsistent terms represent sets of literals of assumptions, which are in contradiction to the knowledge base Σ . Supporting arguments for a hypothesis h represent sets of literals of assumptions which, together with the knowledge base Σ , are sufficient to guarantee the truth of h , whereas possible

arguments represent sets of literals of assumptions, which are sufficient to exclude the guarantee of the falsity of h . Quasi-supporting arguments are again only of technical importance. We remark, that supporting arguments are similar to paths for structure functions in reliability theory. This similarity will be exploited later. We define $QS(h, \Sigma)$, $SP(h, \Sigma)$, and $PL(h, \Sigma) = SP^c(\neg h, \Sigma)$ to be the sets of quasi-supporting, supporting and possible arguments for h , respectively. $QS(\perp, \Sigma)$ denotes then the set of conflicts. These sets are all upward closed, as the following theorem states, i.e. if a conjunction α is in the set, then so is every conjunction α' with $\alpha \subseteq \alpha'$.

Theorem 1 ²³

$$\begin{aligned} QS(h, \Sigma) &= \{\alpha \in C_A : \alpha, \Sigma \models h\}, \\ SP(h, \Sigma) &= \left\{ \alpha \in C_A : \begin{array}{l} \alpha, \Sigma \models h, \\ \alpha', \Sigma \not\models \perp \text{ for every } \alpha' \in C_A, \text{ with } \alpha' \supseteq \alpha, \end{array} \right\}, \\ PL(h, \Sigma) &= \{\alpha \in C_A : \alpha', \Sigma \not\models \neg h \text{ for every } \alpha' \in C_A, \text{ with } \alpha' \supseteq \alpha\} \end{aligned}$$

A conjunction α is a minimal element of a set of conjunctions if for every conjunction α' of this set satisfying $\alpha' \subseteq \alpha$ we have $\alpha = \alpha'$. According to the theorem above the sets of arguments are already determined by their *minimal* elements. We denote by $\mu QS(h, \Sigma)$, $\mu SP(h, \Sigma)$ and $\mu PL(h, \Sigma)$ the respective sets of minimal quasi-supporting, supporting and possible arguments. Then we define

$$\begin{aligned} \textbf{Conflict:} \quad \quad \quad \text{conf}(\Sigma) &:= \bigvee_{\alpha \in \mu QS(\perp, \Sigma)} \alpha, \\ \textbf{Quasi-Support of } h: \quad \text{qs}(h, \Sigma) &:= \bigvee_{\alpha \in \mu QS(h, \Sigma)} \alpha, \\ \textbf{Support of } h: \quad \quad \quad \text{sp}(h, \Sigma) &:= \bigvee_{\alpha \in \mu SP(h, \Sigma)} \alpha, \\ \textbf{Possibility of } h: \quad \quad \quad \text{pl}(h, \Sigma) &:= \bigvee_{\alpha \in \mu PL(h, \Sigma)} \alpha, \end{aligned}$$

Using these definition, we have $N_A(\text{conf}(\Sigma)) = CS_A(h, \Sigma)$, $N_A(\text{qs}(h, \Sigma)) = QS_A(h, \Sigma)$, $N_A(\text{sp}(h, \Sigma)) = SP_A(h, \Sigma)$ and $N_A(\text{pl}(h, \Sigma)) = PL_A(h, \Sigma)$.

A small example will illustrate these basic notions of propositional argumentation systems.

Example 3: cont. of Example 1 The information of Example 1 is modeled in an argumentation system as follows: $A = \{ok_A, ok_B, ok_C\}$, $P = \{in, x_1, x_2, out\}$ and

$$\Sigma = \left\{ \begin{array}{ll} ok_A \rightarrow (in \leftrightarrow \neg x_1), & \neg ok_A \rightarrow \neg x_1 \\ ok_B \rightarrow (in \leftrightarrow \neg x_2), & \neg ok_B \rightarrow \neg x_2 \\ ok_C \rightarrow (out \leftrightarrow x_1 \vee x_2), & \neg ok_C \rightarrow \neg out \end{array} \right\}. \quad (12)$$

There are no inconsistent states and for the hypothesis $h = \neg in \rightarrow out$ we have the following quasi-support and possible scenarios:

$$QS_A(h, \Sigma) = \{(0, 1, 1), (1, 0, 1), (1, 1, 1)\} \quad (13)$$

$$PL_A(h, \Sigma) = N_A. \quad (14)$$

As $CS(\Sigma) = \emptyset$, we have $QS = SP$ in this situation and there are some arguments in favor of the hypothesis, but none against it. The minimal version are $\mu QS(h, \Sigma) = \{ok_A \wedge ok_C, ok_B \wedge ok_C\}$ and $\mu PL(h, \Sigma) = \{\top\}$, such that $qs(h, \Sigma) = (ok_A \wedge ok_C) \vee (ok_B \wedge ok_C)$ and $pl(h, \Sigma) = \top$. \ominus

3.3 Probabilistic Information

On top of the structure of a propositional argumentation systems, we may easily add a probability structure. We assume that there is a probability $p(a_i) = p_i$ for every assumption $a_i \in A$ given. Assuming stochastic independence between assumptions, a scenario $\mathbf{s} = (s_1, \dots, s_n)$ gets the probability

$$p(\mathbf{s}) = \prod_{i=1}^n p_i^{s_i} (1 - p_i)^{1-s_i}. \quad (15)$$

This induces a probability measure p on the language \mathcal{L}_A ,

$$p(f) = \sum_{\mathbf{s} \in N_A(f)} p(\mathbf{s}) \quad (16)$$

for $f \in \mathcal{L}_A$. A quadruple $\mathcal{PAS}_P = (\Sigma, A, P, \Pi)$ with $\Pi = (p_1, \dots, p_n)$ is then called a *probabilistic (propositional) argumentation system* PAS. Note that in fact the stochastic independence is not really needed, but very useful for computational purposes.

Once we have such a probability structure on top of a propositional argumentation system, we can exploit it to compute likelihoods (or in fact, reliabilities) of supporting and possible arguments for hypotheses h . First, we note, that the knowledge base Σ imposes that we eliminate the inconsistent scenarios and condition the probability on the consistent ones. In other words, Σ is an event that restricts the possible scenarios to the set $N_A - CS_A(\Sigma)$, hence their probability has to be conditioned on the event Σ . This conditional probability is defined by

$$p'(\mathbf{s}) = \frac{p(\mathbf{s})}{1 - p(qs(\perp, \Sigma))}. \quad (17)$$

for consistent scenarios \mathbf{s} . $p(qs(h, \Sigma)) = dqs(h)$ is the so-called degree of quasi-support for h . Now, the degree of support dsp for hypotheses h is defined by

$$dsp(h) = p'(sp(h, \Sigma)) = \frac{p(sp(h, \Sigma))}{1 - p(qs(\perp, \Sigma))} = \frac{dqs(h, \Sigma) - dqs(\perp, \Sigma)}{1 - dqs(\perp, \Sigma)}. \quad (18)$$

This result explains the technical importance of quasi-support. It is sufficient to compute degrees of quasi-supports. Further, we obtain the degree of plausibility of h ,

$$dpl(h) = p'(pl(h, \Sigma)) = \frac{p(pl(h, \Sigma))}{1 - dqs(\perp, \Sigma)} = \frac{1 - dqs(-h, \Sigma)}{1 - dqs(\perp, \Sigma)} = 1 - dsp(-h). \quad (19)$$

We remark, that the degree of quasi-support $dqs(h)$ of h corresponds in fact to unnormalized belief functions in the Dempster-Shafer theory of evidence, the degree of support $dsp(h)$ to normalized belief.^{41,34,23}

Example 4: cont. of Example 3 Consider again the hypothesis $h = \text{in} \rightarrow \text{out}$. Assume the probabilities of the assumption in A to be $p_A = p(ok_A) = 0.95$,

$p_B = p(ok_B) = 0.95$, and $p_C = p(ok_C) = 0.99$. Using the quasi-support of h computed in Example 3 above (13), the degree of quasi-support is computed as follows:

$$\begin{aligned}
dqs(h) &= p(QS_A(h)) = p(\{(0, 1, 1), (1, 0, 1), (1, 1, 1)\}) \\
&= (1 - p_A)p_B p_C + p_A(1 - p_B)p_C + p_A p_B p_C \\
&= 0.05 \cdot 0.95 \cdot 0.99 + 0.95 \cdot 0.05 \cdot 0.99 + 0.95 \cdot 0.95 \cdot 0.99 \\
&= 0.9875
\end{aligned}$$

In this situation, as $CS(\Sigma) = \emptyset$, we have $dsp(h) = dqs(h)$. Since $pl(h, \Sigma) = \top$, we have $dpl(h) = 1$. \ominus

3.4 Computing Symbolic Results

Computing quasi-supports is the basic operation in PAS. It can be based on resolution and variable elimination, also known as forgetting or Davis-Putnam procedure.¹⁵ Corresponding algorithms are described in Ref. 23, where also approximate algorithms are discussed which compute arguments using explicit cost-functions.¹⁹ Newer work allows to compute upper and lower bounds of arguments using an anytime-algorithm.²⁰ In the sequel, we will sketch the main concepts for computation.

First, note that the computation of the quasi-support of a hypothesis h can be reduced to the computation of the conflicts with respect to an updated knowledge base:

$$qs(h, \Sigma) = qs(\perp, \Sigma \cup \{-h\}). \quad (20)$$

So for any hypothesis h , the quasi-supporting arguments $qs(h, \Sigma)$ can be determined by computing the conflicts with respect to $\Sigma \cup \{-h\}$. Hence in the sequel, we focus on the computation of the conflicts with respect to a knowledge base $\Sigma_h = \Sigma \cup \{-h\}$.

The algorithms presented in the sequel are based on representations of the knowledge Σ in conjunctive normal form (CNF), i.e. a conjunction of proper clauses ξ . It is well-known that the computation of a CNF representation for an arbitrary formula is NP-hard. But usually in practical applications, Σ is a set of pieces of information, hence a set of formulas which are interpreted conjunctively, so the computational effort for switching from this representation to a CNF may be not so big. Therefore we suppose that the knowledge base Σ (or Σ_h) is given as CNF.

The main step is based on the principle of resolution. Let $x \in A \cup P$ be an atom. A disjoint decomposition of the knowledge base Σ is then defined as follows:

$$\begin{aligned}
\Sigma_x^+ &= \{\xi \in \Sigma : x \in Lit(\xi)\} \\
\Sigma_x^- &= \{\xi \in \Sigma : \neg x \in Lit(\xi)\} \\
\Sigma_x^0 &= \{\xi \in \Sigma : x \notin Lit(\xi) \text{ and } \neg x \notin Lit(\xi)\}
\end{aligned} \quad (21)$$

where $Lit(\Sigma)$ denotes the set of all literals occurring in Σ .

Consider two clauses $\xi^+ = x \vee \delta^+$ and $\xi^- = \neg x \vee \delta^-$ in Σ_x^+ and Σ_x^- respectively. The clause $\rho(\xi^+, \xi^-) = \delta^+ \vee \delta^-$ is called the resolvent; note that we simplify implicitly the resolvent so that $\rho(\xi^+, \xi^-)$ is assumed to be again a *proper* clause.

Eliminating a variable $x \in P \cup A$ from Σ means now to compute

$$Elim_x(\Sigma) = \mu(\Sigma_x^0 \cup \{\rho(\xi^+, \xi^-) : \xi^+ \in \Sigma_x^+, \xi^- \in \Sigma_x^-\}) \quad (22)$$

Several atoms can now be deleted sequentially and the following result shows that the order of elimination does not matter:

Theorem 2 ²³

$$\begin{aligned} Elim_x(Elim_x(\Sigma)) &= Elim_x(\Sigma), \\ Elim_y(Elim_x(\Sigma)) &= Elim_x(Elim_y(\Sigma)). \end{aligned}$$

So the result does not depend on the very order of the elimination of atoms; yet note that the efficiency of the computations depend *critically* on a “good” ordering, see Ref. 23, 24 for a discussion as well as relations to the theory of local computation (in the sense of Ref. 43, 42, 33).

Consider a set $Q \subseteq P \cup A$. We define now, for $Q = \{q_1, \dots, q_r\}$,

$$Elim_Q(\Sigma) = Elim_{q_r}(\dots(Elim_{q_2}(Elim_{q_1}(\Sigma)))\dots) \quad (23)$$

This allows then to compute the quasi-supporting arguments of a knowledge base Σ as follows:

Theorem 3 ²³ $QS_A(h, \Sigma) = N_A^c(Elim_P(\Sigma \cup \{-h\}))$, or in other words this means $\mu QS(h, \Sigma) = \neg\mu Cons_A(Elim_P(\Sigma \cup \{-h\}))$

$Cons_A(\Xi)$ denotes the set of all terms in \mathcal{L}_A which are logical consequences of the argument Ξ . This is again computed using resolution, i.e. for $a \in A$

$$Cons_a(\Xi) = \Xi \cup Elim_a(\Xi) \quad (24)$$

and results analog to Theorem 2 can be proved²³ so that we can define for $A = \{a_1, \dots, a_n\}$,

$$Cons_A(\Xi) = Cons_{a_n}(\dots(Cons_{a_2}(Cons_{a_1}(\Xi)))\dots) \quad (25)$$

The concept of elimination allows to compute quasi-supporting and therefore also supporting as well as possible arguments for hypotheses. In the sequel, we will not only be interested in elimination of all atoms in P , but more general situations will be considered. Therefore, we introduce the following notation. Consider a knowledge base $\Sigma \subseteq \mathcal{L}_{A \cup P}$ and a subset $O \subseteq A \cup P$ of atoms. Then the projection of Σ to O is defined as the elimination of all but the atoms in O from Σ ,

$$\Sigma^{\downarrow O} = Elim_{(A \cup P) - O}(\Sigma). \quad (26)$$

This notation connects the concepts presented here also to the more general information algebras, which is a general theory for representing, combining and focusing pieces of information Ref. 29, 35.

Clearly, if arguments for several different hypotheses h_i with respect to the same knowledge base Σ have to be computed, then it is more efficient to first compute the conflicts $qs(\perp, \Sigma)$ and use this intermediate result to compute $qs(h_i, \Sigma)$. Depending on the number of hypotheses, different computation schemes are possible, which rely all on the concept of local computation.^{23, 33, 37}

3.5 Computing Numerical Results

The problem of computing the probability $p(f)$ of a logical formula f is similar to the problem of computing the reliability of a structure function, except that monotonicity cannot be assumed in general. Hence, the practical computation of $p(f)$ is a nontrivial task. A classical approach consist in transforming the formula f (which is often already a disjunction of conjunctions, yet not disjoint) into a disjunction of disjoint terms, whose probabilities can be easily computed and then summed up.^{1,26,27,34,11,5} Besides, new promising approaches focus on other canonical forms of knowledge compilation.^{14,13}

4 Reliability Analysis Using Probabilistic Argumentation Systems

4.1 Reliability based on Requirement Specification

In this section, we discuss how probabilistic argumentation systems can be used to formulate and solve reliability problem. The basic idea is simple: The system behavior is described in terms of the states of its components. In addition the desired or required behavior of the system is specified. The system description forms a probabilistic argumentation system. The question is then: how likely (probable) is it that the specified requirement is satisfied? In order to answer this question, the specification of required behavior is taken as a hypothesis. The *support* of this specification determines then essentially the structure function of this reliability problem, and the degree of support of the specified requirement is the reliability of the system with respect to required behavior. Note that — depending on different goals a system should attain, or services it should provide — different requirements may be formulated. So the corresponding reliability analysis has to be differentiated, but can be carried out within the same framework of probabilistic argumentation systems.

We start with an example to illustrate this idea before we formalize it in general terms.

Example 5: cont. of Example 1 We consider again the introductory Example 1. There, the system description specifying the expected behavior of the components both in the working state as well as in the down state has been formulated in the language of propositional logic. This forms the knowledge base Σ ,

$$\Sigma = \left\{ \begin{array}{ll} ok_A \rightarrow (in \leftrightarrow \neg x_1), & \neg ok_A \rightarrow \neg x_1, \\ ok_B \rightarrow (in \leftrightarrow \neg x_2), & \neg ok_B \rightarrow \neg x_2, \\ ok_C \rightarrow (out \leftrightarrow x_1 \vee x_2), & \neg ok_C \rightarrow \neg out. \end{array} \right\} \quad (27)$$

The required system behavior was also already specified in terms of constraints on the input-output relation of the system. We formulated in fact, two different, although related specifications,

$$\delta_1 = \neg in \rightarrow out, \quad \delta_2 = \neg in \leftrightarrow out.$$

The first one is the weaker one. It requires that a power failure is signaled. This satisfies the minimal safety requirement. However it allows that a power failure may be signaled also when there is none. The second one is more stringent. It

requires that a power failure is indicated if, *and only if*, there is one. It does not allow false alarms.

Once the system description and the functional requirements are specified, we can compute the supports of these two specifications. It turns out, that both are the same,

$$sp(\delta_1, \Sigma) = sp(\delta_2, \Sigma) = (ok_A \wedge ok_C) \vee (ok_B \wedge ok_C).^1 \quad (28)$$

Note that this is just the path representation of the expected structure function of the system. In fact this structure function could be reformulated as $(ok_A \vee ok_B) \wedge ok_C$, which shows that it is a series system composed of component C and a parallel module of the components A and B . The remarkable fact is, that this structure function has been automatically deduced from the system description and the specification of requirements.

The system description is an essential element for this analysis. If it is changed, then this may influence the results of the analysis. Suppose that, in contrast to the model above, we do not know how the faulty components behave. Then we may decide to give no indication for these states. The knowledge base contains now only the first part of (27),

$$\Sigma' = \left\{ \begin{array}{l} ok_A \rightarrow (in \leftrightarrow \neg x_1), \\ ok_B \rightarrow (in \leftrightarrow \neg x_2), \\ ok_C \rightarrow (out \leftrightarrow x_1 \vee x_2). \end{array} \right\} \quad (29)$$

With this less complete model, the structure function of the two specifications above become different,

$$\begin{aligned} sp(\delta_1, \Sigma') &= (ok_A \wedge ok_C) \vee (ok_B \wedge ok_C), \\ sp(\delta_2, \Sigma') &= ok_A \wedge ok_B \wedge ok_C. \end{aligned}$$

Now, the stronger requirement δ_2 can only be guaranteed if all three components work correctly (a series system), whereas the weaker one still has the same redundancy as before.

If we define probabilities for the components, for example $p(A) = p(B) = 0.99$ and $p(C) = 0.95$, then we can compute the reliabilities corresponding to the different specifications as the degrees of support. In the first case we obtain $dsp(\delta_1, \Sigma) = dsp(\delta_2, \Sigma) = 0.95$, in the second case $dsp(\delta_1, \Sigma') = 0.95$ as before, whereas $dsp(\delta_2, \Sigma') = 0.931$. \ominus

In the general case, we have a PAS (Σ, A, P) , where the assumable symbols in A correspond to the components of the system. Positive assumptions correspond to working components (that is why assumptions in this context are often denoted by ok). Accordingly in the context of reliability analysis, we shall call the scenarios of this argumentation system *system states*. The propositional symbols in P are needed to describe the system behavior. We assume that the system description Σ alone excludes no system states, i.e. there are no conflicts,

$$QS_A(\perp, \Sigma) = \emptyset. \quad (30)$$

¹This results as well as similar ones in other examples has been computed using the software system ABEL.²¹

A knowledge base Σ which satisfies (30) is called A -consistent.

The required behavior is specified by a formula δ . Usually δ will not contain assumptions, but there is no reason to exclude this in general. δ formulates a reliability goal. There may be several such goals.

The set of system states $SP_A(\delta, \Sigma)$ supporting δ contains all states guaranteeing the required specification from the system description. Its complement $SP_A^c(\delta, \Sigma) = PL_A(\neg\delta, \Sigma)$ contains the system states where this guarantee is no more assured, i.e. $\neg\delta$ is possible but not guaranteed, hence δ is still possible. These are the unreliable states. So $SP_A(\delta, \Sigma)$ defines the *structure function* associated with the specification δ

$$s = \phi_{\delta, \Sigma}(\mathbf{s}) = \begin{cases} \top & \text{if } \mathbf{s} \in SP_A(\delta, \Sigma), \\ \perp & \text{if } \mathbf{s} \notin SP_A(\delta, \Sigma). \end{cases} \quad (31)$$

The index Σ in $\phi_{\delta, \Sigma}$ will be omitted if Σ is indicated by the context. Here, s denotes the ‘‘system state’’, which is \top , when the reliability specification is assured and \perp otherwise. Since the set $SP_A(\delta, \Sigma)$ has a logical representation based on minimal arguments, the same holds for the structure function ϕ_δ ,

$$\phi_\delta = \bigvee_{\alpha \in \mu SP(\delta, \Sigma)} \alpha. \quad (32)$$

In the same way, based on minimal possible arguments $PL(\neg\delta, \Sigma)$, we obtain

$$\neg\phi_\delta = \bigvee_{\beta \in \mu PL(\neg\delta, \Sigma)} \beta. \quad (33)$$

By de Morgan laws this transforms into

$$\phi_\delta = \bigwedge_{\beta \in \mu PL(\neg\delta, \Sigma)} \neg\beta. \quad (34)$$

Note that $\neg\beta$, the negation of a term, is a clause. This is a second logical representation of the structure function ϕ_δ .

A structure function ϕ is often assumed to be *monotone*. That is, if $\mathbf{x}_1 \leq \mathbf{x}_2$, then $\phi(\mathbf{x}_1) \leq \phi(\mathbf{x}_2)$. This makes sense in most cases: It says that a system, which is working, is still working if one or more components are repaired (changed from down state to working state). Or, alternatively, a monotone system remains down, if one or more components fail.

A comparison with the minimal path and minimal cut representation of monotone structure functions shows that minimal supporting arguments α for δ and minimal possible arguments β for $\neg\delta$ correspond to minimal paths and minimal cuts. In fact, consider the term α as the set of the literals it contains. Then a minimal supporting argument α is a set of literals guaranteeing the specification δ whatever the value of the assumptions outside α . But in contrast to an ordinary path in the sense of reliability theory, α may contain negative literals. Similarly, considering a term $\neg\beta$ as the set of the literals it contains, then a minimal possible argument β for $\neg\delta$ fixes a set of literals, which assure that the system is unreliable, i.e. δ is not guaranteed, whatever values the assumptions outside of β have.

Note that the structure function ϕ_δ is *not necessarily monotone*. There is a very simple characterization of monotone structure functions in terms of minimal supporting arguments for δ and minimal possible arguments for $\neg\delta$.

Theorem 4 ϕ_δ is monotone, if, and only if, at least one of the two following conditions are satisfied:

- a) All minimal supporting arguments in $\mu SP(\delta, \Sigma)$ contain only positive literals.
- b) All minimal possible arguments in $\mu PL(\neg\delta, \Sigma)$ contain only negative literals.

Proof

“ \Rightarrow ” If $\mu SP(\delta, \Sigma) = \{\top\}$ and hence $\phi_\delta = \top$, the theorem follows. Assume now that there is an $\alpha \in \mu SP(\delta, \Sigma)$ with $\alpha = \alpha_1 \wedge \dots \wedge \alpha_k$ which contains negated literals (at least one). Without restriction, we assume that $\alpha_1 = \neg a_1$. We show that this implies that ϕ_δ is not monotone.

Let $S = N_A(\alpha)$. By definition, we have $\emptyset \neq S \subseteq SP_A(\delta, \Sigma)$. Let

$$S' = \{\mathbf{s}' = (1, s_2, \dots, s_n) : (0, s_2, \dots, s_n) \in S\}. \quad (35)$$

There is an $\mathbf{s}' \in S'$ with $\mathbf{s}' \notin SP_A(\delta, \Sigma)$ because otherwise $\alpha_2 \wedge \dots \wedge \alpha_k \in \mu SP(\delta, \Sigma)$ which violates the assumption that α is minimal.

Let $\mathbf{s}' = (1, s_2, \dots, s_n) \in S'$ but $\mathbf{s}' \notin S$ and $\mathbf{s} = (0, s_2, \dots, s_n)$. Then $\mathbf{s} \in S$ and $\mathbf{s} \leq \mathbf{s}'$. Now $\mathbf{s} \in S \subseteq SP_A(\delta, \Sigma)$ and $\mathbf{s}' \notin SP_A(\delta, \Sigma)$ imply that $\phi_\delta(\mathbf{s}) = 1$ and $\phi_\delta(\mathbf{s}') = 0$ which shows that ϕ_δ is not monotone. This proves a). The proof of b) is similar.

“ \Leftarrow ” First assume that a) is true, i.e. every element of $\mu SP(\delta, \Sigma)$ contains only positive literals. Consider \mathbf{s} and \mathbf{s}' from N_A with $\mathbf{s}' \geq \mathbf{s}$. Let $\alpha \in \mu SP(\delta, \Sigma)$. Then $\mathbf{s} \models \alpha$ implies $\mathbf{s}' \models \alpha$, hence $\phi_\delta(\mathbf{s}) = \top$ implies $\phi_\delta(\mathbf{s}') = \top$ by (32). Further, if $\mathbf{s}' \not\models \alpha$ for every $\alpha \in \mu SP(\delta, \Sigma)$ then also $\mathbf{s} \not\models \alpha$ for every $\alpha \in \mu SP(\delta, \Sigma)$. Thus $\phi_\delta(\mathbf{s}') = \perp$ implies $\phi_\delta(\mathbf{s}) = \perp$. Therefore $\phi_\delta(\mathbf{s}) \leq \phi_\delta(\mathbf{s}')$ which shows that ϕ_δ is monotone.

Second assume that b) is true, i.e. every element of $\mu PL(\neg\delta, \Sigma)$ contains only negative literals. Consider \mathbf{s} and \mathbf{s}' from N_A with $\mathbf{s}' \geq \mathbf{s}$. Let $\beta \in \mu PL(\neg\delta, \Sigma)$. Then $\mathbf{s}' \not\models \neg\beta$ implies $\mathbf{s} \not\models \neg\beta$, hence $\phi_\delta(\mathbf{s}') = \perp$ implies $\phi_\delta(\mathbf{s}) = \perp$ by (34). Further, if $\mathbf{s} \models \neg\beta$ for every $\beta \in \mu PL(\neg\delta, \Sigma)$ then also $\mathbf{s}' \models \neg\beta$ for every $\beta \in \mu PL(\neg\delta, \Sigma)$. Thus $\phi_\delta(\mathbf{s}) = \top$ implies $\phi_\delta(\mathbf{s}') = \top$. Therefore $\phi_\delta(\mathbf{s}) \leq \phi_\delta(\mathbf{s}')$ which shows that ϕ_δ is monotone.

□

This theorem shows that monotonicity of a structure function can easily be decided by inspecting the minimal arguments computed from the argumentation system. In this case the usual definition of paths and cuts can be used. Otherwise this is not the case. Also in the case of Theorem 4, if there is at least one supporting argument for δ , then $\phi_\delta(\mathbf{1}) = \top$ and if there is at least one possible argument for $\neg\delta$, then $\phi_\delta(\mathbf{0}) = \perp$.

Example 6: cont. of Example 5 The minimal supporting arguments in Example 5 satisfy the first condition of Theorem 4. The minimal possible arguments are $\{\neg ok_C, \neg ok_A \wedge \neg ok_B\}$ for $\neg\delta_1$ and $\{\neg ok_A, \neg ok_B, \neg ok_C\}$ for $\neg\delta_2$ in the second, partial system. In the first system both specifications have possible arguments $\{\neg ok_C, \neg ok_A \wedge \neg ok_B\}$ for their negations. So the second condition

of Theorem 4 is also satisfied in all cases. Hence, all structure functions are monotone. The minimal cut representation in the first case, for example, is

$$\phi_{\delta_1} = ok_C \wedge (ok_A \vee ok_B). \quad (36)$$

⊖

We remind from Sections 3.2 and 3.4 that

$$\begin{aligned} SP_A(\delta, \Sigma) &= QS_A(\delta, \Sigma) - QS_A(\perp, \Sigma) \\ &= QS_A(\perp, \Sigma \cup \{\neg\delta\}) - QS_A(\perp, \Sigma). \end{aligned}$$

According to our assumption of A -consistency, we have $QS_A(\perp, \Sigma) = \emptyset$. Thus

$$SP_A(\delta, \Sigma) = QS_A(\perp, \Sigma \cup \{\neg\delta\}). \quad (37)$$

On the other hand, we have also similarly

$$PL_A(\neg\delta, \Sigma) = SP_A^c(\delta, \Sigma) = QS_A^c(\perp, \Sigma \cup \{\neg\delta\}). \quad (38)$$

This shows, that a reliability analysis of a system Σ relative to a requirement specification δ , requires essentially the computation of the conflict states $QS_A(\perp, \Sigma \cup \{\neg\delta\})$. We shall see below, that this is exactly what is required for diagnosis. This is a first hint to the duality between the problems of reliability and diagnosis.

Once probabilities for the assumptions, i.e. component availabilities or reliabilities are defined, system reliability relative to a specification δ is simply the degree of support of δ ,

$$p_{\delta, \Sigma} = dsp(\delta, \Sigma) = dqs(\delta, \Sigma) = p(QS_A(\perp, \Sigma \cup \{\neg\delta\})), \quad (39)$$

since $QS_A(\perp, \Sigma) = \emptyset$. The indices of $p_{\delta, \Sigma}$ will be omitted if they are obvious from the context.

So, we conclude that all the reliability analysis of a system with a given specification of systems requirement can be carried out within the framework of probabilistic argumentation systems. All the necessary inference mechanisms and methods for computing probability are provided by this framework.

As a further application of formulating reliability problems within the frame of probabilistic argumentation systems we refer back to Example 2. This indicates that classical reliability problems may be embedded into argumentation systems.

Example 7: In Example 2 we have already indicated two possible requirements, namely the two-terminal connection and the source-to-net connections,

$$\begin{aligned} \delta_1 &= x_1 \rightarrow x_4, \\ \delta_2 &= x_1 \rightarrow x_2 \wedge x_1 \rightarrow x_3 \wedge x_1 \rightarrow x_4. \end{aligned}$$

The support of the first specification relative to Σ as defined in Ex. 2 is given by the minimal supporting arguments (written as sets of literals) $\{ok_{13}, ok_{34}\}$, $\{ok_{12}, ok_{24}\}$, $\{ok_{12}, ok_{23}, ok_{34}\}$, and $\{ok_{13}, ok_{23}, ok_{24}\}$ for δ_1 . These correspond clearly to the (minimal) paths linking the nodes x_1 and x_4 .

Similarly, the minimal possible arguments for $\neg\delta_1$ are the sets $\{\neg ok_{24}, \neg ok_{34}\}$, $\{\neg ok_{12}, \neg ok_{13}\}$, $\{\neg ok_{13}, \neg ok_{23}, \neg ok_{24}\}$, and $\{\neg ok_{12}, \neg ok_{23}, \neg ok_{34}\}$. They correspond to the minimal cuts.

The minimal supporting arguments with respect to δ_2 correspond to spanning trees of the bridge network and the minimal possible arguments with respect to $\neg\delta_2$ to cut sets which disconnect the connected network. We remark that both systems are monotone.

If we define link probabilities of 0.9 for each edge, we obtain for the two-terminal problem the reliability $dsp(\delta_1, \Sigma) = 0.978$ and for the source-to-net reliability $dsp(\delta_2, \Sigma) = 0.977$. \ominus

4.2 Implicitly Defined Reliability

Usually a system is designed such that all its required functionality is guaranteed at least, if all components are working; this is reflected by the requirement $\phi(\mathbf{1}) = 1$ for the structure function ϕ . We call a specification δ *consistent* with the system description Σ , if the system state $\mathbf{1}$ belongs to $SP_A(\delta, \Sigma)$. This is surely true for δ inducing monotone systems with at least one minimal supporting argument. In this section we only consider specifications consistent with the system description.

A system description Σ often contains, besides assumptions, another set O of special propositional atoms, namely those which are considered to be *observable*. Then specifications δ can be assumed to be formulated with observables only, $\delta \in \mathcal{L}_O$. Observables are typically input and output variables of some system. This is illustrated in the following example.

Example 8: In Example 5 the non-assumables *in* and *out* can be considered as observables. Both requirement specifications in Example 6 are expressed in terms of observables only. Of course, it is a modeling decision to single out the observables. x_1 and x_2 could under other circumstances well be declared as observables too. \ominus

Assume then that in a system description (Σ, P, A) a set of observable variables O is singled out. Usually, $O \subseteq P$, i.e. component states can not be observed directly. But it does no harm to assume more generally $O \subseteq P \cup A$. Then we define an implicit specification

$$\hat{\delta} \equiv (\Sigma \cup \{a_1 \wedge a_2 \wedge \dots \wedge a_n\})^{\downarrow O}.$$

That is, $\hat{\delta}$ represents all the functionality of the system in terms of observables which can be obtained from a system with all components working. We call this the *implicit* reliability specification with respect to O . Now, the system may be — with respect to this specification — as good as “new” also for some states including faulty components. Therefore we define the implicit structure function by the set of up-states relative to $\hat{\delta}$, i.e. $SP_A(\hat{\delta}, \Sigma)$. That is, we obtain

$$\phi_{\hat{\delta}} = \bigvee_{\alpha \in \mu SP(\hat{\delta}, \Sigma)} \alpha, \quad \text{or} \quad \phi_{\hat{\delta}} = \bigwedge_{\beta \in \mu PL(\neg\hat{\delta}, \Sigma)} \neg\beta. \quad (40)$$

Accordingly, the implicit reliability of such a system can be obtained as the degree of support $dsp(\hat{\delta}, \Sigma)$. This approach helps to decide whether a system has some implicit redundancy, namely, whether $\phi_{\hat{\delta}}$ represents not simply a series system, that is, $\mu SP(\hat{\delta}, \Sigma)$ has only the set of all assumptions as minimal supporting argument for $\hat{\delta}$.

Example 9: In Example 8 we have defined the observables $O = \{in, out\}$. In this example the implicit reliability specification with the fully defined system

turns out to be $\delta_2 \equiv (\Sigma \cup \{a_1 \wedge a_2 \wedge \dots \wedge a_n\})^{\downarrow O}$. The corresponding structure function is thus defined by the minimal supporting arguments $\{ok_A, ok_C\}$ and $\{ok_B, ok_C\}$. As we have seen, this structure function is monotone, which is not always the case for implicit structure functions. \ominus

We stated above that $\hat{\delta}$ covers the whole functionality attainable with all components working. This will now be restated in a more precise form.

Lemma 5 *If $\delta \in \mathcal{L}_O$ is a consistent specification with respect to Σ , then $\hat{\delta} \models \delta$.*

Proof Consistency of δ means that $\Sigma \cup \{a_1 \wedge a_2 \wedge \dots \wedge a_n\} \models \delta$. The assumption that $\delta \in \mathcal{L}_O$ implies then $(\Sigma \cup \{a_1 \wedge a_2 \wedge \dots \wedge a_n\})^{\downarrow O} \models \delta$ because for $h \in \mathcal{L}_O$ we have $\Sigma \models h$ if and only if $\Sigma^{\downarrow O} \models h$.²³ But this means that indeed $\hat{\delta} \models \delta$. \square

This lemma shows that $\hat{\delta}$ is the most stringent, consistent specification over observables O . Intuitively we expect therefore, that for all specifications over O the implicit specification has least reliability. This is indeed so, as the next lemma shows.

Lemma 6 *If $\delta \in \mathcal{L}_O$ is a consistent specification w.r.t. Σ , then $SP_A(\hat{\delta}, \Sigma) \subseteq SP_A(\delta, \Sigma)$.*

Proof By Lemma 5 $\hat{\delta} \models \delta$, hence $\mathbf{s}, \Sigma \models \hat{\delta}$ implies $\mathbf{s}, \Sigma \models \delta$. \square

As a corollary we see that the reliability relative to a partial specification is at least the reliability relative to the implicit specification, i.e. the full functionality.

Corollary 7 *If $\delta \in \mathcal{L}_O$ is a consistent specification with respect to Σ , then $p_\delta \geq p_{\hat{\delta}}$.*

Proof By Lemma 6 we have $SP_A(\hat{\delta}, \Sigma) \subseteq SP_A(\delta, \Sigma)$, hence $p_{\hat{\delta}} = dsp(\hat{\delta}, \Sigma) = p'(SP_A(\hat{\delta}, \Sigma)) \leq p'(SP_A(\delta, \Sigma)) = dsp(\delta, \Sigma) = p_\delta$. \square

4.3 Compatible System Description

In a system composed of several components we will in general know exactly how intact components work. But we may be less sure how faulty components behave. E.g. in Example 1, it is clear that a working inverter inverts the input. But a faulty inverter may show one of several behaviors. In Example 1 we assumed that a faulty inverter sticks to a zero output. But another possibility would be that it simply passes the input through instead of inverting it. Still other possibilities exist. It is even possible that different failure modes mix randomly. So, if we are not sure what the fault behavior is, we may leave it open. What is the effect of such a simplification on the reliability analysis?

Let Σ be a “complete” system description, where the behavior of components is specified both for the working as well as for the down state. If we eliminate the specification of the behavior of faulty components, then we obtain a new “more crude” system description Σ' . We may assume that

$$\Sigma \models \Sigma'. \quad (41)$$

This relation expresses exactly the relation of “coarsening” a model. The “coarser” model describes the system behavior only partially or less completely than Σ , but is still consistent with it. Inversely, we may also start with a crude, partial model Σ' and “refine” it to a more complete model Σ satisfying the relation above. We call Σ' *compatible* with Σ , if (41) holds and Σ is A -consistent. An example will illustrate this notion.

Example 10: cont. of Example 1 We consider again the introductory Example 1. There the system description Σ specifying the expected behavior of the components both in the working state as well as in the down state has been formulated in the language of propositional logic.

$$\Sigma = \left\{ \begin{array}{ll} ok_A \rightarrow (in \leftrightarrow \neg x_1), & \neg ok_A \rightarrow \neg x_1, \\ ok_B \rightarrow (in \leftrightarrow \neg x_2), & \neg ok_B \rightarrow \neg x_2, \\ ok_C \rightarrow (out \leftrightarrow x_1 \vee x_2), & \neg ok_C \rightarrow \neg out. \end{array} \right\} \quad (42)$$

If we delete the specification of the behavior of faulty components, we obtain

$$\Sigma' = \left\{ \begin{array}{l} ok_A \rightarrow (in \leftrightarrow \neg x_1), \\ ok_B \rightarrow (in \leftrightarrow \neg x_2), \\ ok_C \rightarrow (out \leftrightarrow x_1 \vee x_2). \end{array} \right\} \quad (43)$$

Clearly the relation $\Sigma \models \Sigma'$ is satisfied. Also both system descriptions are A -consistent. Hence Σ' is compatible with Σ . \ominus

We have the following simple result.

Theorem 8 *Let Σ and Σ' be two system descriptions such that Σ' is compatible with Σ , and δ a requirement specification, consistent with Σ' . Then δ is also consistent with Σ and $SP_A(\delta, \Sigma') \subseteq SP_A(\delta, \Sigma)$.*

Proof We remark that $\mathbf{s}, \Sigma' \models \perp$ implies $\mathbf{s}, \Sigma \models \perp$, hence $QS_A(\perp, \Sigma') \subseteq QS_A(\perp, \Sigma)$. Therefore, if $QS_A(\perp, \Sigma) = \emptyset$, then $QS_A(\perp, \Sigma') = \emptyset$ too and Σ' is A -consistent. Further, $\Sigma \models \Sigma'$ implies also

$$QS_A(\delta, \Sigma') = \{\mathbf{s} : \mathbf{s}, \Sigma' \models \delta\} \subseteq \{\mathbf{s} : \mathbf{s}, \Sigma \models \delta\} = QS_A(\delta, \Sigma). \quad (44)$$

Hence we obtain that $SP_A(\delta, \Sigma') = QS_A(\delta, \Sigma') \subseteq QS_A(\delta, \Sigma) = SP_A(\delta, \Sigma)$. Thus, if δ is consistent with Σ' , that is, $\mathbf{1} \in SP_A(\delta, \Sigma')$, then $\mathbf{1} \in SP_A(\delta, \Sigma)$ and δ is consistent with Σ too. \square

As a corollary we note that reliability relative to specification δ grows, if the system description is refined.

Corollary 9 *Let Σ and Σ' be two system descriptions such that Σ' is compatible with Σ , and δ a requirement specification. Then $p_{\delta, \Sigma} = dsp(\delta, \Sigma) \geq dsp(\delta, \Sigma') = p_{\delta, \Sigma'}$.*

Proof This follows from Theorem 8. \square

According to this corollary, a partial system description gives always a lower bound of the reliability of a specification. So, working with a partial model at

least leaves us on the safe side, whereas assuming a wrong finer model may lead to a too optimistic view of system reliability.²

Example 11: We refer back to Example 5, where the complete model yields

$$sp(\delta_1, \Sigma) = sp(\delta_2, \Sigma) = (ok_A \wedge ok_C) \vee (ok_B \wedge ok_C). \quad (45)$$

The partial model on the other hand gives only

$$\begin{aligned} sp(\delta_1, \Sigma') &= (ok_A \wedge ok_C) \vee (ok_B \wedge ok_C), \\ sp(\delta_2, \Sigma') &= ok_A \wedge ok_B \wedge ok_C. \end{aligned}$$

For the weaker specification δ_1 both system description give the same structure function. For the sharper specification δ_2 however we have $sp(\delta_2, \Sigma') \models sp(\delta_2, \Sigma)$. The partial system description yields a series system, whereas the complete model exhibits the parallel redundancy of the two inverters. \ominus

The implicit reliability specifications of two compatible system description are also related.

Theorem 10 *Let Σ and Σ' be two systems such that Σ' is compatible with Σ . Then $\hat{\delta}_\Sigma \models \hat{\delta}_{\Sigma'}$.*

Proof This follows since $\Sigma \models \Sigma'$ implies $(\Sigma, \mathbf{1})^{\downarrow O} \models (\Sigma', \mathbf{1})^{\downarrow O}$. \square

So, a more complete model induces a sharper implicit reliability specification, than a partial system description. This implies also that a finer model has less implicit reliability than a partial model. We take this as a hint that implicit reliability should be used only, if the complete model is known.

Example 12: cont. of Example 5 Once more we refer back to Example 5. We saw already in Example 9 that for the complete system we have $\hat{\delta}_\Sigma = \delta_2$. For the partial model, without specification of the behavior of the faulty components, we obtain $\hat{\delta}_{\Sigma'} = \delta_1$ and indeed $\delta_2 \models \delta_1$. \ominus

4.4 Examples

In this section, we present two typical examples of classical reliability theory to show how the “expected” structure function — in the sense of classical reliability theory — is computed using the approach of implicit structure function defined above.

Example 13: Three Components Consider the simple reliability diagram in Fig. 4. The system consists of an input K_E , two components e_2 and e_3 in parallel with one other components e_1 connected serially, and an output K_A . The connections between in- and output is established if there is a path of working components from K_I to K_O . Therefore, the structure function ϕ can for this simple example be deduced from the graphic, i.e.

$$\phi(x_1, x_2, x_3) = x_1 \wedge (x_2 \vee x_3), \quad (46)$$

where x_i denotes the state of component e_i .

²See also Ref. 32 for more information about complete models.

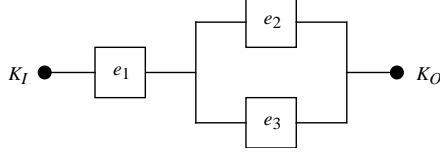


Figure 4: A small device with three components.

We reconsider now this example in the framework of PAS. Consider three assumptions, i.e. $A = \{ok_1, ok_2, ok_3\}$ and three propositions $P = \{e, a, x\}$, where the assumption ok_i is true when the component e_i is functioning (i.e. when $x_i = 1$). Further, we assume here that $O = \{e, a\} \subset P$, i.e. the user can only observe the input and output variables, but not the internal connection x . The knowledge base Σ consists then of three formulas:

$$\Sigma = \{ok_1 \rightarrow (e \leftrightarrow x), ok_2 \rightarrow (x \leftrightarrow a), ok_3 \rightarrow (x \leftrightarrow a)\} \quad (47)$$

We want now to compute the implicitly defined reliability, hence we first determine $\hat{\delta}$. For the computation, the knowledge base is converted to a conjunctive normal form:

$$\Sigma_{\text{CNF}} = \left\{ \begin{array}{l} \neg ok_1 \vee \neg e \vee x, \neg ok_1 \vee e \vee \neg x, \\ \neg ok_2 \vee \neg x \vee a, \neg ok_2 \vee x \vee \neg a, \\ \neg ok_3 \vee \neg x \vee a, \neg ok_3 \vee x \vee \neg a. \end{array} \right\} \quad (48)$$

Clearly, $\Sigma_{\text{CNF}} \equiv \Sigma$. The marginal of $\Sigma_{\text{CNF}} \cup \{ok_1 \wedge ok_2 \wedge ok_3\}$ to the sublanguage \mathcal{L}_O is computed using variable elimination and gives

$$(\Sigma_{\text{CNF}} \cup \{ok_1 \wedge ok_2 \wedge ok_3\})^{\downarrow O} = \{e \vee \neg a, \neg e \vee a\} \quad (49)$$

where the right hand side is a description (in conjunctive normal form) of the possible observations, namely $e \wedge a$ and $\neg e \wedge \neg a$, because indeed $(e \vee \neg a) \wedge (\neg e \vee a) \equiv (e \wedge a) \vee (\neg e \wedge \neg a)$. So the implicit reliability $\hat{\delta}$ is

$$\hat{\delta} = (e \wedge a) \vee (\neg e \wedge \neg a) \equiv e \leftrightarrow a. \quad (50)$$

The implicit structure function is then given by $SP_A(\hat{\delta}, \Sigma)$ and hence

$$\phi_{\hat{\delta}} = sp(\hat{\delta}, \Sigma) = (ok_1 \wedge ok_2) \vee (ok_1 \wedge ok_3) \equiv ok_1 \wedge (ok_2 \vee ok_3) \quad (51)$$

which is clearly the same as (46). \ominus

Example 14: Fault Tree Consider the simple fault tree of Fig. 5. We model this as a PAS. First, the propositional atoms are $P = \{p_1, p_2, p_3\}$, i.e. all atoms in the tree which are not leaf nodes. The observable nodes in the tree are the faults of the system p_1 , p_2 and p_3 , hence the observable atoms are $O = \{p_1, p_2, p_3\}$. There are four assumptions $A = \{c_1, \dots, c_4\}$ corresponding to the leaf nodes. The knowledge base reflects the structure of the tree, i.e. a fault is triggered by its children in the tree as follows:

$$\Sigma = \{p_1 \leftrightarrow (c_1 \vee c_2 \vee c_3), p_2 \leftrightarrow (c_3 \vee c_4), p_3 \leftrightarrow (p_1 \wedge p_2)\} \quad (52)$$

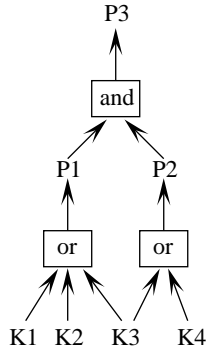


Figure 5: A simple fault tree

The implicit reliability is then $\hat{\delta} = (\Sigma \cup \{c_1 \wedge c_2 \wedge c_3 \wedge c_4\})^{\downarrow O} = p_1 \wedge p_2 \wedge p_3$. The implicit structure function $sp(\hat{\delta}, \Sigma) = (c_1 \wedge c_4) \vee (c_2 \wedge c_4) \vee c_3$ represents then exactly what we have in mind when inspecting Fig. 5.

Now consider the situation where only the top most failure can be observe, i.e. the set of observable propositions is $O' = \{p_3\}$. Using the same knowledge base as above, we get $\hat{\delta}' = (\Sigma \cup \{c_1 \wedge c_2 \wedge c_3 \wedge c_4\})^{\downarrow O'} = p_3$. The implicit structure function $sp(\hat{\delta}', \Sigma) = (c_1 \wedge c_4) \vee (c_2 \wedge c_4) \vee c_3$ which is the same information as above. \ominus

5 Model-Based Diagnostic

5.1 Duality Between Reliability and Diagnostics

A problem of diagnostics arises, if an observation indicates that a requirement specification δ is violated. Then the question is: how can the required functionality be recovered? That is, one would like to find out those components whose failure caused the system failure and which have to be fixed or replaced. This analysis will be based on the system description Σ and on the specification δ which is violated.

In fact, we ask, which system states are compatible or consistent with the system description Σ and the violation of the specification δ , expressed by $\neg\delta$. Well, these are of course all states which are consistent with $\Sigma \cup \{\neg\delta\}$, i.e. the set

$$QS_A^c(\perp, \Sigma \cup \{\neg\delta\}) = PL_A(\neg\delta, \Sigma). \quad (53)$$

Remark that this is exactly the set of down states relative to the specification δ (see (38)). Here we have the basic *duality* between *reliability analysis* relative to a requirement specification δ and the *diagnostic problem* relative to the same specification. The conflict set $QS_A(\perp, \Sigma \cup \{\neg\delta\})$ is the computational key to both reliability analysis and diagnostics. It gives the up-states which define reliability and its complement gives the possible states explaining the violation of the reliability specification, that is possible diagnostics. It is well known in model-based diagnostics that such conflict sets play a key role.^{40,16,31} The duality implies that they play an equally important role for model-based reliability.

We remind that $\mu PL(-\delta, \Sigma)$ are the minimal terms representing the set $QS_A^c(\perp, \Sigma \cup \{-\delta\}) = PL_A(-\delta, \Sigma)$. $\beta \in \mu PL(-\delta, \Sigma)$ are minimal possible arguments for the violation $-\delta$ of the specification δ . According to Theorem 4, if the structure function $\phi_{\delta, \Sigma}$ is *monotone*, then to the minimal possible arguments $\beta \in \mu PL(-\delta, \Sigma)$ correspond the minimal cuts $\neg\beta$. They represent minimal sets of failed components, which explain the violation of the specification δ , independently on the state of the other components.

Minimal cuts correspond to kernel diagnoses in model-based diagnostics.⁴⁰ Usually model-based diagnostics goes not beyond such concepts of diagnostics. It neglects the important role of probability (see however Ref. 31, 3). The observation of the violation of the specification $-\delta$ in fact defines the event $QS_A^c(\perp, \Sigma \cup \{-\delta\})$ in the sample space N_A . That is, the prior probabilities $p(\mathbf{s})$ defined on the states have now to be conditioned on this event. This gives us the *posterior probabilities*

$$p(\mathbf{s}|\neg\delta) = \frac{p(\mathbf{s})}{1 - p(QS_A^c(\perp, \Sigma \cup \{-\delta\}))} = \frac{p(\mathbf{s})}{dpl(-\delta, \Sigma)}, \quad (54)$$

for diagnostic states $\mathbf{s} \in QS_A(\perp, \Sigma \cup \{-\delta\})$. This underlines once more the key role of the conflict set $QS_A(\perp, \Sigma \cup \{-\delta\})$. Its prior probability is sufficient to compute the posterior probabilities of the possible diagnostic states explaining the violation of a specification δ .

These posterior probabilities represent important additional diagnostic information. For example we may look for diagnostic states with maximal posterior probability. A state $\tilde{\mathbf{s}}$ is called a *maximal likelihood state*, if

$$p(\tilde{\mathbf{s}}|\neg\delta) = \max_{\mathbf{s} \in QS_A^c(\perp, \Sigma \cup \{-\delta\})} p(\mathbf{s}|\neg\delta). \quad (55)$$

There may be several such states. They represent most likely states explaining the violation of the specification δ .

Reiter⁴⁰ proposed to look especially at possible diagnostic states with a minimal number of faulty components. Intuitively this makes sense: The failure should be explained with a minimal number of down components. If \mathbf{s} is a state, we define s^- to be the set of its negative (down) components. Note that $\mathbf{s}' \leq \mathbf{s}$ means that $s'^- \subseteq s^-$, so we look for diagnoses $\mathbf{s} \in QS_A^c(\perp, \Sigma \cup \{-\delta\})$ which are minimal with respect to this partial order, i.e. *Reiter diagnoses*. We make the reasonable assumption that for every component i we have $p_i > 0.5$ such that $p_i > 1 - p_i$. That is, it is more probable that a component works than that it is down. Then it is easy to verify that $\mathbf{s}' < \mathbf{s}$ implies that $p(\mathbf{s}'|\neg\delta) > p(\mathbf{s}|\neg\delta)$. So maximum likelihood states are Reiter diagnoses. The inverse of course does not hold necessarily. Also, if the structure function ϕ_δ is monotone, the s^- of Reiter diagnoses correspond to minimal cuts relative to the specification δ .

The posterior fault probabilities of the components, $p(-a_i|\neg\delta)$, are also of interest. The larger this probability, the more critical is component i for the requirement specification δ . So this is a possible *importance measure* for component i relative to the specification (for other importance measures see Ref. 4).

Example 15: We consider once more the detector of power failure, Example 1. Suppose we observe that, although $\neg in$ we have also $\neg out$, i.e. a power system failure is not detected. Note that $\neg in \wedge \neg out \equiv \neg\delta_1$ (cf. Example 5). So we consult the minimal cuts relative to the specification $\neg\delta_1$.

In Example 6 we have computed the minimal cuts; there is one with one literal, and one with two: $\{\neg ok_C\}$ and $\{\neg ok_A, \neg ok_B\}$. To any minimal cut corresponds a Reiter diagnosis, namely, $\{ok_A, ok_B, \neg ok_C\}$ to the first cut, and $\{\neg ok_A, \neg ok_B, ok_C\}$ to the second one. One of these two diagnoses must be the maximum likelihood state. The first one has prior probability $0.99 \cdot 0.99 \cdot 0.05 = 0.049$, whereas the second one has probability $0.01 \cdot 0.01 \cdot 0.95 = 0.000095$. So clearly, the first one is the by far the most likely state. The posterior probability is obtained, when the prior probability is divided by the unreliability 0.05 relative to δ_1 (see example 5). We obtain for the maximum likelihood state a posterior probability of 0.98. All other possible states have posterior probability less than 0.01. So the diagnosis is pretty clear.

We may also compute the posterior fault probability of each component, given the observation of $\neg\delta_1$. We obtain 0.145 for each inverter and 0.726 for the or-gate, which is clearly much more important or critical than each one of the inverter. \ominus

5.2 Diagnostics Based on Observations of System Behavior

As the last example in the previous Section 5.1 indicates, the actual observation is not necessarily the negation of a system requirement, but may be something stronger, which implies the violation of a specification. Indeed, as we saw in Example 15 we observed $\neg in \wedge \neg out \equiv \neg\delta_1$, but $\neg in \wedge \neg out \models \neg\delta_2$. So, we should reconsider the duality between reliability and diagnostics. In fact, assume we make some observation of the system behavior, expressed in a formula ω over observables. Then we may test whether $\omega \models \neg\hat{\delta}_\Sigma$. If this is the case, then we have a diagnostic problem, i.e. the specification is violated.

The solution of this diagnostic problem is found along similar lines as in the previous section. Possible states are those, which are consistent with the system description and the observation. Or, in other words, the states in the conflict set $QS_A(\perp, \Sigma \cup \{\omega\})$ are those which are excluded by the observation. So, the possible diagnostic states are those of the set $PL_A(\omega, \Sigma) = QS_A^c(\perp, \Sigma \cup \{\omega\})$. We see that this diagnostic problem is dual to a (fictitious) reliability problem with a “requirement” specification $\neg\omega$. Note that the specification $\neg\omega$ is always consistent with Σ , since $\hat{\delta}_\Sigma$ is consistent and $\omega \models \neg\hat{\delta}_\Sigma$.

Of course, we get a much sharper diagnostic with an observation $\omega \models \neg\hat{\delta}$, than with the information of $\neg\hat{\delta}$ alone. This follows, because according to Lemma 6, we have

$$PL_A(\omega, \Sigma) \subseteq PL_A(\hat{\delta}, \Sigma). \quad (56)$$

So, the more precise the observation, the more states are eliminated. A mere statement that a given reliability specification is violated is less informative than a precise observation implying a violation of a requirement specification.

5.3 Diagnostics Using Compatible System Descriptions

What happens, if a system description is refined or completed? Assume that Σ' is a partial system description, and ω an observation such that $\omega \models \neg\delta$ for some specification δ consistent with Σ' . This means, we have a diagnostic problem, as in the previous Section 5.2. Indeed, $\mathbf{1} \in SP_A(\delta, \Sigma)$ implies $\mathbf{1} \notin PL_A(\neg\delta, \Sigma')$,

and since $PL_A(\omega, \Sigma') \subseteq PL_A(\neg\delta, \Sigma')$, we conclude that $\mathbf{1} \notin PL_A(\omega, \Sigma')$. So observation ω excludes the possibility that all components are in their working mode.

How does the diagnosis change, if the partial system description Σ' is replaced by a more complete one, Σ such that $\Sigma \models \Sigma'$? First, we note that by Theorem 8, δ is also consistent with respect to Σ . So, we have still a diagnostic problem relative to Σ too. Also according to Theorem 8, we have that $PL_A(\omega, \Sigma) \subseteq PL_A(\omega, \Sigma')$. So turning to a more complete model focuses the diagnostics to less possible states. However, we remark that, if the completed model uses a wrong description of the behavior of faulty components, then it may be that $PL_A(\omega, \Sigma)$ does not contain the true state generating the observation, whereas the partial model still faithfully contains the true state in $PL_A(\omega, \Sigma')$. We do not want to enter in a discussion of these important issues here. A more detailed examination of these questions is contained in Ref. 23.

6 Combining Diagnostic and Reliability

We conclude this discussion of duality between reliability and diagnostics by remarking that we may have an observation of the system behavior which does neither entail a specification δ nor its violation $\neg\delta$. But still this observation is information and we can use it to improve reliability analysis and also to perform a preventive diagnostic analysis.

In fact, let $\omega \neq \perp$ be a logical formula representing some information about the system described by Σ , and δ a consistent specification. We may simply add ω to the system description Σ , to obtain a new, more complete one $\Sigma' = \Sigma \cup \{\omega\}$. Clearly we have then $\Sigma' \models \Sigma$, but Σ' will in general not be compatible with Σ , because Σ' will be no more A -consistent. Anyway, we can compute the set of states which guarantee δ under $\Sigma \cup \{\omega\}$, that is, $SP_A(\delta, \Sigma \cup \{\omega\})$. But this time, we do not have $SP_A(\delta, \Sigma \cup \{\omega\}) = QS_A(\delta, \Sigma \cup \{\omega\})$, since $QS_A(\perp, \Sigma \cup \{\omega\})$ will be no more empty, in general. So, we have

$$SP_A(\delta, \Sigma \cup \{\omega\}) = QS_A(\delta, \Sigma \cup \{\omega\}) - QS_A(\perp, \Sigma \cup \{\omega\}). \quad (57)$$

We shall see in a moment, that the second term, the conflict set $QS_A(\perp, \Sigma \cup \{\omega\})$, is again essential for the diagnostic analysis. The first term can also be rewritten as a conflict term, such that

$$SP_A(\delta, \Sigma \cup \{\omega\}) = QS_A(\perp, \Sigma \cup \{\neg\delta, \omega\}) - QS_A(\perp, \Sigma \cup \{\omega\}). \quad (58)$$

Consider the two extreme cases:

$\omega \models \delta$: In this case, ω is a confirmation that the system really works, i.e. $\Sigma \cup \{\omega\}$ is A -consistent. Hence the numerical reliability is 1.

$\omega \models \neg\delta$: Here, ω is a confirmation that the requirement δ is not fulfilled, or — in other words — $SP_A(\delta, \Sigma \cup \{\omega\}) = \emptyset$.

In the case where $\omega \not\models \delta$ as well as $\omega \not\models \neg\delta$, nothing can be said in general about the behaviour of the set of possible scenarios when new information is taken into consideration as it behaves non-monotonically, i.e. the set may increase or decrease (and so does the reliability).

However, for reliability as well as for diagnostic, additional measurements — or more general any additional information — can be taken into account in the framework presented above and helps to focus the reasoning. We use the following example to sketch the possible use.

Example 16: cont. of Example 4 Consider the situation, where we have the possibility to make measurements not only at the in- and output of the device, but also at point x_1 , hence $O = \{in, out, x_1\}$. Assume that we actually measure $\neg in$ and $\neg x_1$, therefore $\omega = \neg in \wedge \neg x_1$. This measurement does neither imply the specification $\delta_1 = \neg in \rightarrow out$, nor contradict it, i.e. we have $\omega \not\models \delta_1$ as well as $\omega \not\models \neg \delta_1$. Yet, knowing the measurement ω implies that the inverter A is faulty, because $qs(\perp, \Sigma \cup \{\omega\}) = \neg ok_A$. This means, that we have a diagnosis which tells us that in the actual situation, the inverter A cannot be working correctly. Nevertheless, there is still a possibility that the device is functioning as it is supposed to be, i.e. according to δ_1 . The reliability according to δ_1 is $sp(\delta_1, \Sigma \cup \{\omega\}) = \neg ok_A \wedge ok_B \wedge ok_C$, hence there is only one possible explanation for the system to be working according to δ_1 .

Clearly, the numerical reliability decreases with the information ω : Before the measurement, the reliability was $dsp(\delta_1, \Sigma) = 0.988$, whereas including ω , we get $dsp(\delta_1, \Sigma \cup \{\omega\}) = 0.941$. This decreasing comes from the fact that we know for sure now that the inverter A is broken.

The same results are obtained in this example when the specification δ_1 is replaced by $\delta_2 = \neg in \leftrightarrow out$. \ominus

7 Conclusions

In this paper we have shown how reliability and model-based diagnostic are closely connected. The framework of probabilistic argumentation system appears to be a framework which covers both approaches. Therefore the generic structure of PAS can be used for solving problems in both domains. The approaches can even be combined and the information specified can be used in the common framework.

Further, from the system description of an argumentation system, we can derive the appropriate structure function and — if desirable — take into consideration a reliability requirement.

The concept of argumentation system allows to use local computation architectures and approximation techniques.^{43,23} This complements the computational theory of reliability theory.

The ideas have been presented in the framework of propositional argumentation systems, but these systems can be generalized to more general situations,⁴⁰ see Ref. 29,31,32,3 for further literature.

PAS can also be seen as a theory of the reliability of arguments. This is another link between PAS and reliability theory. Further work has to investigate this aspect and transport ideas from one field to the other one. For example, the notion of importance measures from reliability theory has been transported to the framework of argumentation systems.⁴

References

- [1] J. A. Abraham. An improved algorithm for network reliability. *IEEE Transactions on Reliability*, 28:58–61, 1979.
- [2] B. Anrig. Probabilistic argumentation systems and model-based diagnostics. In A. Darwiche and G. M. Provan, editors, *DX'00, Eleventh Intl. Workshop on Principles of Diagnosis, Morelia, Mexico*, pages 1–8, 2000.
- [3] B. Anrig. *Probabilistic Model-Based Diagnostics*. PhD thesis, University of Fribourg, Institute of Informatics, 2000.
- [4] B. Anrig. Importance measures from reliability theory for probabilistic assumption-based reasoning. In S. Benferhat and P. Besnard, editors, *European Conf. ECSQARU'01, Toulouse*, pages 692–703. Lecture Notes in Artif. Intell., Springer, 2001.
- [5] B. Anrig and F. Beichelt. Disjoint sum forms in reliability theory. *ORiON J. OR Society South Africa*, 16(1):75–86, 2001.
- [6] B. Anrig, R. Bissig, R. Haenni, J. Kohlas, and N. Lehmann. Probabilistic argumentation systems: Introduction to assumption-based modeling with ABEL. Technical Report 99-1, University of Fribourg, Institute of Informatics, 1999.
- [7] B. Anrig, R. Haenni, and N. Lehmann. ABEL — a new language for assumption-based evidential reasoning under uncertainty. Technical Report 97–01, University of Fribourg, Institute of Informatics, 1997.
- [8] B. Anrig, N. Lehmann, and R. Haenni. Reasoning with finite set constraints. Technical Report 97–11, University of Fribourg, Institute of Informatics, 1997.
- [9] R. E. Barlow and R. Proschan. *Statistical Theory of Reliability and Life Testing*. New York, 1975.
- [10] F. Beichelt. *Zuverlässigkeits- und Instandhaltungstheorie*. Teubner, Stuttgart, 1993.
- [11] R. Bertschy and P. Monney. A generalization of the algorithm of Heidtmann to non-monotone formulas. *J. of Computational and Applied Math.*, 76:55–76, 1996.
- [12] L. Console, C. Picardi, and M. Ribaud. Process algebras for systems diagnosis. *Artif. Intell.*, 142:19–51, 2002.
- [13] A. Darwiche. A compiler for deterministic decomposable negation normal form. In *Proc. of the National Conf. on Artif. Intell. (AAAI 2002)*, pages 627–634, 2002.
- [14] A. Darwiche and P. Marquis. A perspective on knowledge compilation. In *Proc. 17th Int. Joint Conf. on Artif. Intell. IJCAI-01*, 2001.
- [15] M. Davis and H. Putnam. A computing procedure for quantification theory. *J. of the ACM*, 5:394–397, 1962.

- [16] R. Davis. Diagnostic reasoning based on structure and behaviour. *Artif. Intell.*, 24:347–410, 1984.
- [17] J. De Kleer. *Local Methods for Localizing Faults in Electronical Circuits*. MIT AI Memo 394, MIT Cambridge, MA., 1976.
- [18] J. De Kleer and B. C. Williams. Diagnosing multiple faults. *Artif. Intell.*, 32:97–130, 1987.
- [19] R. Haenni. Cost-bounded argumentation. *Int. J. of Approximate Reasoning*, 26(2):101–127, 2001.
- [20] R. Haenni. A query-driven anytime algorithm for assumption-based reasoning. Technical Report 01-26, University of Fribourg, Department of Informatics, 2001.
- [21] R. Haenni, B. Anrig, R. Bissig, and N. Lehmann. ABEL homepage. <http://diuf.unifr.ch/tcs/abel>, 2000.
- [22] R. Haenni, B. Anrig, J. Kohlas, and N. Lehmann. A survey on probabilistic argumentation. In *ECSQARU'01, Toulouse. Workshop: Adventures in Argumentation*, pages 19–25, 2001.
- [23] R. Haenni, J. Kohlas, and N. Lehmann. Probabilistic argumentation systems. In J. Kohlas and S. Moral, editors, *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, volume 5: Algorithms for Uncertainty and Defeasible Reasoning, pages 221–287. Kluwer, Dordrecht, 2000.
- [24] R. Haenni and N. Lehmann. Efficient hypertree construction. Technical Report 99-3, University of Fribourg, Institute of Informatics, 1999.
- [25] R. Haenni and N. Lehmann. Probabilistic argumentation systems: a new perspective on Dempster-Shafer theory. *Intl. J. of Intelligent Systems, Special Issue on Dempster-Shafer Theory of Evidence*, 18(1):93–106, 2003.
- [26] K. Heidtmann. Smaller sums of disjoint products by subproduct inversion. *IEEE Transactions on Reliability*, 38(3):305–311, August 1989.
- [27] K. Heidtmann. *Zuverlässigkeitsbewertung technischer Systeme*. Teubner, 1997.
- [28] J. Kohlas. *Zuverlässigkeit und Verfügbarkeit*. Teubner, 1987.
- [29] J. Kohlas. *Information Algebras: Generic Structures for Inference*. Springer, 2003.
- [30] J. Kohlas, B. Anrig, and R. Bissig. Reliability and diagnostic of modular systems. *ORiON J. OR Society South Africa*, 16(1), 2001.
- [31] J. Kohlas, B. Anrig, R. Haenni, and P. Monney. Model-based diagnostics and probabilistic assumption-based reasoning. *Artif. Intell.*, 104:71–106, 1998.
- [32] J. Kohlas, D. Berzati, and R. Haenni. Probabilistic argumentation systems and abduction. *Annals of Mathematics and Artif. Intell., Special Issue (AMAI)*, 34:177–195, 2002.

- [33] J. Kohlas, R. Haenni, and S. Moral. Propositional information systems. *J. of Logic and Computation*, 9 (5):651–681, 1999.
- [34] J. Kohlas and P. Monney. *A Mathematical Theory of Hints. An Approach to the Dempster-Shafer Theory of Evidence*, volume 425 of *Lecture Notes in Economics and Mathematical Systems*. Springer, 1995.
- [35] J. Kohlas and R. F. Stärk. Information algebras and information systems. Technical Report 96–14, University of Fribourg, Institute of Informatics, 1996.
- [36] K. B. Laskey and P. E. Lehner. Assumptions, beliefs and probabilities. *Artif. Intell.*, 41:65–77, 1989.
- [37] N. Lehmann and R. Haenni. An alternative to outward propagation for Dempster-Shafer belief functions. In A. Hunter and S. Parsons, editors, *European Conf. ECSQARU'99, London*, pages 256–267. Lecture Notes in Artif. Intell., Springer, 1999.
- [38] G. M. Provan. A logic-based analysis of Dempster-Shafer theory. *Int. J. of Approximate Reasoning*, 4:451–495, 1990.
- [39] G. M. Provan. An integration of model-based diagnosis and reliability theory. In A. Darwiche and G. M. Provan, editors, *DX'00, Eleventh Intl. Workshop on Principles of Diagnosis, Morelia, Mexico*, pages 193–200, 2000.
- [40] R. Reiter. A theory of diagnosis from first principles. *Artif. Intell.*, 32:57–95, 1987.
- [41] G. Shafer. *The Mathematical Theory of Evidence*. Princeton University Press, 1976.
- [42] P. P. Shenoy and J. Kohlas. Computation in valuation algebras. In J. Kohlas and S. Moral, editors, *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, volume 5: Algorithms for Uncertainty and Defeasible Reasoning, pages 5–40. Kluwer, Dordrecht, 2000.
- [43] P. P. Shenoy and G. Shafer. Axioms for probability and belief functions propagation. In R. D. Shachter, T. S. Levitt, L. N. Kanal, and J. F. Lemmer, editors, *Uncertainty in Artif. Intell. 4*. North Holland, 1990.