INFORMATICA, 2005, Vol. 16, No. 1, 3–18

Probabilistic Argumentation Systems with Decision Variables

Bernhard ANRIG
Berne University of Applied Sciences, School of Engineering and Information Technology
CH–2501 Biel, Switzerland
e-mail: bernhard.anrig@hti.bfh.ch

Dalia BAZIUKAITĖ
Klaipėda University, Department of Computer Science
LT–92294 Klaipėda, Lithuania
e-mail: dalia.baziukaite@ik.ku.lt

Received: July 2004

Abstract. The general concept of probabilistic argumentation systems PAS is restricted to the two types of variables: assumptions, which model the uncertain part of the knowledge, and propositions, which model the rest of the information. Here, we introduce a third kind into PAS: so-called decision variables. This new kind allows to describe the decisions a user can make to react on some state of the system. Such a decision allows then possibly to reach a certain goal state of the system. Further, we present an algorithm, which exploits the special structure of PAS with decision variables.*

Key words: argumentation systems, probabilistic reasoning, decision, uncertainty, reliability.

1. Introduction

The concept of probabilistic argumentation systems PAS has been used for dealing with problems in different contexts and several examples from a wide spectrum have been treated (Anrig *et al.*, 1999; Haenni *et al.*, 2000; Kohlas *et al.*, 2000). Lately, Anrig & Kohlas (Anrig and Kohlas, 2002a) have considered the relation between reliability (Kohlas, 1987; Beichelt, 1993) and model-based diagnostics (Anrig, 2000; Kohlas *et al.*, 1998) on the basis of PAS. In this environment, the attention was brought to a small but very interesting example (cf. Section 2). Although the theory of PAS was able to compute the required result for this example, this was only possible after a quite sophisticated modelization of the knowledge. Yet it turns out that there is a more elegant and natural way.

Inspired from this example, we have extended the concept of PAS with *decision variables*. Consider the situation where we have modeled a system using a PAS (P, Σ) and the

*Some results related with this paper were published in (Anrig and Baziukaitė, 2003a).

elements of the set $A \subseteq P$ are the indicator variables of the components of the system. A requirement δ describes the expected behaviour of the system, typically its desired input-output relations. We are now interested in computing those system states which allow to deduce the requirement δ. Anrig & Kohlas (Anrig and Kohlas, 2002) have shown that those system states are just the supporting scenarios of δ. This is a well known problem in PAS.

Now consider that given a system state, the users can themselves set the values of some decision variables in order to guarantee the requirement. The interesting system states are now those for which the users can find at least one settings of "their" variables under which the requirement imposed on the system can be fulfilled. This can be seen as a game with two players, say nature against a user. Nature makes the first move and the user the second. If the requirement is fulfilled, the user wins (as the system is up), otherwise not.

For computing results for this generalized version of PAS, the original methods of a PAS can be applied followed by a post-processing. This is also addressed in (Anrig and Baziukaitė, 2003a). Yet this proves to be inefficient and a new approach is proposed.

Section 2 presents an introductory example. In Section 3 the general theory of argumentation systems is introduced; in Section 4, this is generalized to argumentation and decision systems. Section 5 discusses then the computational theory and introduces a new algorithm.

2. An Example: Energy Distribution System

EXAMPLE 1. The function of an electrical power distribution system as depicted in Fig. 1 is to supply a high-tension line $L3$ from one of two incoming high-tension lines $L1$ and $L2$ over the bus $A1$ (Frey and Reichert, 1973; Kohlas, 1987). The lines $L1$, $L2$, and $L3$ are protected by corresponding high-tension switches $S1$, $S2$, and $S3$. In case of a

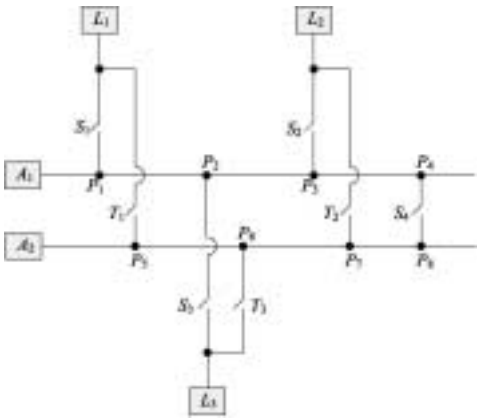

Fig. 1. Example of an energy distribution system.

broken switch it is possible to redirect the electric current on a second bus $A2$ which is protected by another high-tension switch $S4$. The corresponding switches $T1$, $T2$, and $T3$ can only be operated under no tension.

The energy distribution system is considered to be operating if $L3$ is linked to $L1$ or $L2$ over *at least two* protecting high-tension switches Si ($i = 1, \ldots, 4$). The problem is to determine the availability of an operating system. The lines are modeled by the corresponding binary propositions with the same name, e.g., $L1$ means that the line $L1$ is under tension, whereas $\neg L1$ means that it is not. A switch can be closed (e.g., $S1$) or open (e.g., $\neg S1$) and furthermore it can be in its correct working mode (e.g., ok_{S1}) or faulty (e.g., $\neg ok_{S1}$). For the working modes we have probabilistic information available:

$$p(ok_{Si}) = 0.95, \quad p(ok_{Ti}) = 0.97, \quad p(ok_{Ai}) = 0.99. \tag{1}$$

The energy can pass through a switch only if the switch is closed. If a switch is not intact, then it cannot be closed. So every switch can be modeled by the two logical formulas, where the second formula has always the form *if switch x is closed, then there is power on both lines or points specified on the right-hand side or on none of them*:

$$\begin{array}{ll}
\neg ok_{S1} \rightarrow \neg S1, & S1 \rightarrow (L1 \leftrightarrow P1), \\
\neg ok_{T1} \rightarrow \neg T1, & T1 \rightarrow (L1 \leftrightarrow P5), \\
\neg ok_{S2} \rightarrow \neg S2, & S2 \rightarrow (L2 \leftrightarrow P3), \\
\neg ok_{T2} \rightarrow \neg T2, & T2 \rightarrow (L2 \leftrightarrow P7), \\
\neg ok_{S3} \rightarrow \neg S3, & S3 \rightarrow (P2 \leftrightarrow L3), \\
\neg ok_{T3} \rightarrow \neg T3, & T3 \rightarrow (P6 \leftrightarrow L3), \\
\neg ok_{S4} \rightarrow \neg S4, & S4 \rightarrow (P4 \leftrightarrow P8).
\end{array} \tag{2}$$

The different connection points to the buses are modeled by the propositions $P1$ to $P8$; if the bus is working correctly, then either there is power on all of the respective connection points or on none of them:

$$ok_{A1} \rightarrow \Big((P1 \wedge P2 \wedge P3 \wedge P4) \vee (\neg P1 \wedge \neg P2 \wedge \neg P3 \wedge \neg P4)\Big) \tag{3}$$

$$ok_{A2} \rightarrow \Big((P5 \wedge P6 \wedge P7 \wedge P8) \vee (\neg P5 \wedge \neg P6 \wedge \neg P7 \wedge \neg P8)\Big) \tag{4}$$

An important part of the model is the fact that only one of $T1$, $T2$ and $T3$ is allowed to be closed at the same time, because otherwise there is no protection between the corresponding lines:

$$\neg(T1 \wedge T2), \qquad \neg(T1 \wedge T3), \qquad \neg(T2 \wedge T3). \tag{5}$$

Until now, we have specified different things:

- propositions whose values are set by nature:

$$A = \{ok_{S1}, ok_{S2}, ok_{S3}, ok_{S4}, ok_{T1}, ok_{T2}, ok_{T3}, ok_{A1}, ok_{A2}\}, \tag{6}$$

• propositions whose values are (or may be) set by the user (under some restrictions):

$$D = \{S1, S2, S3, S4, T1, T2, T3\}, \tag{7}$$

• other propositions: $U = \{P1, \ldots, P8\}$,
• knowledge about the system in form of logical sentences.

Consider now the requirement that given power at the input lines $L1$ and $L2$ the line $L3$ is really supplied. We will formulate this by the logical formula $\delta = L1 \wedge L2 \rightarrow L3$.

There are two main problems: the first one is to compute a (logical) description of those system states, where the user can react so that the requirement δ is fulfilled. Second, for every system state we want to have a description of possible decisions of the user (so that the requirement δ is fulfilled). In the rest of this paper we will discuss solutions to both problems.

Clearly, in reliability theory this very example has been treated successfully: this means that for example in (Kohlas, 1987), the path sets have been determined *by hand* and only then based on this information, the reliability function has been computed. Here, we will show how the structure function can be derived directly from the logical description of the system.

3. Argumentation Systems

Probabilistic assumption-based argumentation systems have been 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 of these arguments. These systems can be used for example for model-based diagnostics as has been shown in (Anrig, 2000; Kohlas *et al.*, 1998).

Argumentation systems can be based on different logics. 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 (Haenni *et al.*, 2000; Kohlas *et al.*, 1998; Anrig *et al.*, 1999). Note that such systems have been implemented in a system called ABEL, Assumption-Based Evidential Language (Anrig *et al.*, 1999), which is available on the internet (`http://diuf.unifr.ch/tcs/abel`), and which can be used to compute results of the examples of this paper as well as of other problems.

3.1. *Basic Definitions*

Propositional logic deals with declarative statements that can either be true or false. Such statements are called *propositions*. Let $P = \{p_1, \ldots, p_n\}$ be a finite set of propositions. The symbols $p_i \in P$ together with $\top$ (tautology) and $\bot$ (falsity), are called *atoms* or

atomic formulas. Compound formulas are built by usual syntactic rules (using the connectors $\neg$, $\wedge$, $\vee$, $\rightarrow$, and $\leftrightarrow$).

The set $\mathcal{L}_P$ of all formulas 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 of negative, and neither $\top$ nor $\bot$ does occur. Additionally, we say that $\top$ is a (empty) proper term and $\bot$ 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 term) 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, $\bot \subseteq \alpha \subseteq \top$ for every clause α and $\bot \supseteq \beta \supseteq \top$ for every term β. If we want to emphasize the interpretation as a set, we write $lit(\beta)$. Note that $lit(\neg\beta)$ for a term β is the set of literals of the clause $\neg\beta$. For a set of formulas X we define $\neg X = \{\neg\zeta : \zeta \in X\}$.

$N_P = \{0,1\}^n$ denotes the set of all 2^n different interpretations relative to P The set of all models of γ is denoted by $N_P(\gamma) \subseteq N_P$. A propositional sentence γ *entails* another sentence δ (denoted by $\gamma \models \delta$) if and only if $N_P(\gamma) \subseteq N_P(\delta)$. Sometimes, it is convenient to write $\mathbf{x} \models \gamma$ instead of $\mathbf{x} \in N_P(\gamma)$. Furthermore, two sentences γ and δ are *logically equivalent* (denoted by $\gamma \equiv \delta$), if and only if $N_P(\gamma) = N_P(\delta)$.

3.2. *Propositional Argumentation Systems*

Consider a finite set $P = \{p_1, p_2, \ldots, p_m\}$ of propositions. We consider a fixed set of formulas $\Sigma \subseteq \mathcal{L}_P$ 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.

DEFINITION 1. A **propositional argumentation system PAS** is a tuple (P, Σ) where P is a set of literals and $\Sigma \subseteq \mathcal{L}_P$ a set of formulas.

Note that in the present formulation (and opposed to previous work on PAS, e.g., (Haenni *et al.*, 2000; Anrig *et al.*, 1999)), we do not explicitly single out the assumptions, but allow them to vary from one situation to another (cf. below). This does not contradict the initial idea about argumentation system. Often, it is clear from the beginning which propositions are assumptions and which ones are not. Yet from the users perspective it is interesting to switch the type of variables from assumptions to propositions or vice versa in the development of a model. Further we will see in Section 4 that this can also be interesting for answering even more general questions.

DEFINITION 2. Consider a PAS (P, Σ) and a subset of **assumptions** $A \subseteq P$. For a hypothesis $h \in \mathcal{L}_P$ we define

Inconsistent Scenarios:	$CS^s_A\Sigma) = \{\mathbf{s} \in N_A : \mathbf{s}, \Sigma \models \bot\}$
Quasi-Support Set:	$QS^s_A(h, \Sigma) = \{\mathbf{s} \in N_A : \mathbf{s}, \Sigma \models h\}$
Support Set:	$SP^s_A(h, \Sigma) = QS^s_A(h, \Sigma) - QS^s_A(\bot, \Sigma)$
Plausible Set:	$PL^s_A(h, \Sigma) = N_A - QS^s_A(\neg h, \Sigma)$

The elements of N_A are called *scenarios* or *system states*. Inconsistent scenarios are in contradiction with the knowledge base and therefore to be considered as excluded by the knowledge. CS^s_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 a supporting scenario, 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 are important especially for technical reasons.

3.3. *Logical Representation*

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:

DEFINITION 3. Consider a PAS (P, Σ), a set of assumptions $A \subseteq P$ and a hypothesis $h \in \mathcal{L}_P$, then we call:

Conflicts:	$\alpha \in C_A$ such that $N_A(\alpha) \subseteq CS^s_A(\Sigma)$
Quasi-Supporting Argument for h:	$\alpha \in C_A$ such that $N_A(\alpha) \subseteq QS^s_A(h, \Sigma)$
Supporting Argument for h:	$\alpha \in C_A$ such that $N_A(\alpha) \subseteq SP^s_A(h, \Sigma)$
Possible Argument for h:	$\alpha \in C_A$ such that $N_A(\alpha) \subseteq PL^s_A h, \Sigma)$

We define $QS_A(h, \Sigma)$, $SP_A(h, \Sigma)$, and $PL_A(h, \Sigma) = SP^c_A(\neg h, \Sigma)$ to be the sets of quasi-supporting, supporting and possible arguments for h, respectively. $QS(\bot, \Sigma)$ denotes then the set of conflicts. These sets are all upward closed, e.g., if $\alpha \in SP_A(h, \Sigma)$ then every $\alpha' \supset \alpha$ is also in $SP_A(h, \Sigma)$.

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 $\alpha = \alpha'$. The sets of arguments defined above are already determined by their *minimal* elements. In general for an upward closed set S of conjunctions (or clauses), μS denotes the set of minimal elements of S.

3.4. *Probabilistic Argumentation Systems*

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 $\mathbf{s} = (s_1, \ldots, s_n)$ gets the probability

$$p(\mathbf{s}) = \prod_{i=1}^{n} p_i^{s_i}(1 - p_i)^{1-s_i}. \tag{8}$$

This induces a probability measure p on the language $\mathcal{L}_A$, $p(f) = \sum\{p(\mathbf{s}) : \mathbf{s} \in N_A(f)\}$ for $f \in \mathcal{L}_A$. A tuple (Σ, A, P, Π) with $\Pi = (p_1, \ldots, p_n)$ is then called a *probabilistic (propositional) argumentation system PAS*.

More generally, any probability measure p on $\mathcal{L}_A$ (or more rigorously on N_A) can be used. The local structure described above is just a very convenient but also frequent special case, yet there are other local structures which can be used.

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 $N_A - CS_A^s(\Sigma)$, hence their probability has to be conditioned on the event Σ. This conditional probability is defined by

$$p'(\mathbf{s}) = \frac{p(\mathbf{s})}{1 - p(QS_A(\bot, \Sigma))}.$$

for consistent scenarios $\mathbf{s}$. $p(QS_A(h, \Sigma)) = dqs_A(h)$ is the so-called degree of quasi-support for h. The degree of support dsp_A for hypotheses h is defined by

$$dsp_A(h) = p'(\mu SP_A(h, \Sigma)) = \frac{dqs_A(h, \Sigma) - dqs_A(\bot, \Sigma)}{1 - dqs_A(\bot, \Sigma)}.$$

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 i.e., $dps_A(h) = 1 - dsp_A(\neg h)$. For some information about computing these probabilities see Section 5.3.

Note that the degree of quasi-support $dqs_A(h)$ of h corresponds in fact to unnormalized belief, the degree of support to normalized belief in the Dempster–Shafer theory of evidence (Shafer, 1976; Kohlas and Monney, 1995; Haenni *et al.*, 2000).

4. Argumentation and Decision Systems

The concept of a PAS allows that the user can compute symbolical and numerical arguments for and against any hypotheses he likes, but it does not allow to include actions of the user. The introductory Example 1 is a typical situation where in some cases additional user interaction allows to fulfill a specified requirement. Hence we will incorporate a limited range of user actions (or better: possible user actions) into our system. In the sequel, we will especially look at the situation when the user makes a *re*action with respect to some previous action of "nature", in the sense that given some components are functioning, he selects which ones of these components should do the job.

4.1. *Propositional Argumentation and Decision Systems*

Consider a PAS (P, Σ) and a set of assumptions $A \subseteq P$. A subset of variables $D \subseteq P$ is singled out. This set D contains all those variables which can be set by the user. The idea is that given the knowledge base Σ, the variables specified by A are set without any possible intervention of the user, i.e., these are the variables set by nature or an enemy. The user cannot influence them. Yet after these variables are set, the user can set the values of the variables specified by D and try to deduce there with some hypothesis $h \in \mathcal{L}_P$. More precisely: given the setting of the variables in A, is there always a possibility for the user to select a setting of the variables in D so that together with the knowledge Σ these settings are not contradictory and imply the hypothesis h?

DEFINITION 4. A **propositional argumentation and decision system PADS** is a tuple (P, D, Σ) with $D \subseteq P$, and $\Sigma \subseteq \mathcal{L}_P$. A scenario $\mathbf{d} \in N_D$ is called a **user decision**.

We assume that the propositions in D can directly be influenced by the user, whereas the other propositions cannot. Especially the assumptions, i.e., a specified set $A \subseteq P - D$, cannot be influenced by the user but only by nature. Here we will consider the situation where $A \cap D = \emptyset$, i.e., there are no propositions which can be influenced directly by nature as well as by the user.

Formalizing the ideas from above, we are now interested in the following scenarios given a hypothesis $h \in \mathcal{L}_P$ and a set of assumptions $A \subseteq P - D$:

$$\{\mathbf{s} \in N_A : \text{ There is a } \mathbf{d} \in N_D \text{ s.t. } \mathbf{s}, \mathbf{d}, \Sigma \models h \text{ and } \mathbf{s}, \mathbf{d}, \Sigma \not\models \bot\} \tag{9}$$

In the special case when there are no decidable variables, $D = \emptyset$, the problem above is just the problem of computing the support $SP^s_A(h, \Sigma)$ of the hypothesis h (considered as an ordinary hypothesis). Therefore we re-use the same notation for the set (9):

DEFINITION 5. The **set of supporting scenarios** $SP^s_A(h, \Sigma; D)$ of a hypothesis h with respect to the decision variables in D is defined as

$$\begin{aligned} &SP^s_A(h, \Sigma; D) \\ &\quad = \{\mathbf{s} \in N_A : \text{ There is a } \mathbf{d} \in N_D \text{ s.t. } \mathbf{s}, \mathbf{d}, \Sigma \models h \text{ and } \mathbf{s}, \mathbf{d}, \Sigma \not\models \bot\} \end{aligned} \tag{10}$$

For every supporting scenario, i.e., for every setting of the variables in A chosen by nature, the users can find at least one setting of their variables in D, i.e., those which can be changed by them, so that the knowledge based together with these settings allow to deduce the hypothesis h, but at the same time are not contradictory. Clearly, $SP^s_A(h, \Sigma; \emptyset) = SP^s_A(h, \Sigma)$.

EXAMPLE 2. The information modeled in Example 1 is already in the form of a PADS, namely we have defined the assumptions A (see (6)) and the decision variables D (see

(7)). There are some further variables $U = \{P1, \ldots, P8\}$. The knowledge about the system has been modeled using several logical formulas in Equations (2) to (5), which together specify Σ.

These ingredients form the PADS (P, D, Σ) with $P = D \cup U \cup A$.

For the hypothesis $\delta = L1 \wedge L2 \rightarrow L3$ the following supporting scenarios can be computed:

$$SPA(\delta, \Sigma; D) = \left\{ \begin{array}{ll} (1,*,1,*,*,*,*,1,*), & (*,1,1,*,*,*,*,1,*), \\ (1,*,*,1,*,*,1,1,1), & (*,1,*,1,*,*,1,1,1), \\ (*,*,1,1,1,*,*,1,1), & (*,*,1,1,*,1,*,1,1) \end{array} \right\} \tag{11}$$

Here, the assumptions are ordered as in (6) and a star "$*$" signifies that either a 0 or a 1 can be inserted in the place. This gives finally a total of 170 supporting scenarios. For example, the scenario $\mathbf{s} = (1,0,1,0,0,0,0,1,0) \in SP_A^s(\delta, \Sigma; D)$ states that only the switches $S1$ and $S3$ are working correctly together with the bus $A1$ and indeed this scenario allows the user to take a specific action (namely close the switches which work correctly) so that the requirement is fulfilled.

Lemma 1. *$SP_A^s(h, \Sigma; D)$ is monotone in D, i.e., for every $D' \subseteq D$ we have $SP_A^s(h, \Sigma; D') \subseteq SP_A^s(h, \Sigma; D)$.*[2]

Several results from support functions (cf. Theorem 2.2 in (Haenni *et al.*, 2000)) can be restated for the present case. However, note the set inclusion instead of equality in point (3) below.

Lemma 2. *If h_1, h_2 are sentences in $\mathcal{L}_P$ then*

(1) $SP_A^s(\bot, \Sigma; D) = \emptyset$.
(2) $SP_A^s(\top, \Sigma; D) = CS_A^s(\Sigma)$.
(3) $SP_A^s(h_1 \wedge h_2, \Sigma; D) \subseteq SP_A^s(h_1, \Sigma; D) \cap SP_A^s(h_2, \Sigma; D)$.
(4) $SP_A^s(h_1 \vee h_2, \Sigma; D) \supseteq SP_A^s(h_1, \Sigma; D) \cup SP_A^s(h_2, \Sigma; D)$.
(5) $h_1 \models h_2$ *implies* $SP_A^s(h_1, \Sigma; D) \subseteq SP_A^s(h_2, \Sigma; D)$.

4.2. *Decisions Depending on System States*

The second question which has been raised in the introductory example was: given a system state which decision can be taken by the user? Given a PADS, this can be answered using the following concept:

DEFINITION 6. The **set of supporting decisions** $SD_A^s(h, \Sigma; D)$ of a hypothesis h with respect to the decision variables in D is defined as

$$SD_A^s(h, \Sigma; D) = \{(\mathbf{s}, \mathbf{d}) \in N_A \times N_D : \mathbf{s}, \mathbf{d}, \Sigma \models h \text{ and } \mathbf{s}, \mathbf{d}, \Sigma \not\models \bot\}$$

[2] For proofs of all lemmas see (Anrig and Baziukaitė, 2003).

A supporting decision $(\mathbf{s}, \mathbf{d}) \in SD^s_A(h, \Sigma; D)$ specifies first a system state $\mathbf{s}$ and a corresponding decision $\mathbf{d}$ which can be taken by the user in order to fulfill the hypothesis h. For a system state, there may be several possible decision, hence this is not an optimal representation.

4.3. *Logical Representation*

Analog to the case of ordinary PAS, we define the notion of argument for a PADS.

DEFINITION 7. Consider a PADS (P, D, Σ), a set of assumptions $A \subseteq P - D$ and a hypothesis $h \in \mathcal{L}_P$, then we call $\alpha \in C_A$ a **supporting argument for** h if and only if $N_A(\alpha) \subseteq SP^s_A(h, \Sigma; D)$. The **set of supporting arguments** for h is denoted by $SP_A(h, \Sigma; D)$. As in the case of PAS, sets of supporting arguments are upward closed. Hence $\mu SP_A(h, \Sigma; D)$ denotes the corresponding set of minimal arguments.

DEFINITION 8. Similarly, we call $\alpha \in C_{A\cup D}$ a **supporting decision (argument) for** h if and only if $N_{A\cup D}(\alpha) \subseteq SD^s_A(h, \Sigma; D)$. The **set of supporting decision (arguments)** for h is denoted by $SD_A(h, \Sigma; D)$; this set is upward closed. Hence $\mu SD_A(h, \Sigma; D)$ is the corresponding set of minimal ones.

EXAMPLE 3. A logical version of the supporting arguments is more readable:

$$\mu SP_A(h, \Sigma; D) = \left\{ \begin{array}{l} ok_{A1} \wedge ok_{S1} \wedge ok_{S3}, \\ ok_{A1} \wedge ok_{S2} \wedge ok_{S3}, \\ ok_{A1} \wedge ok_{A2} \wedge ok_{S1} \wedge ok_{S4} \wedge ok_{T3}, \\ ok_{A1} \wedge ok_{A2} \wedge ok_{S2} \wedge ok_{S4} \wedge ok_{T3}, \\ ok_{A1} \wedge ok_{A2} \wedge ok_{S3} \wedge ok_{S4} \wedge ok_{T1}, \\ ok_{A1} \wedge ok_{A2} \wedge ok_{S3} \wedge ok_{S4} \wedge ok_{T2} \end{array} \right\} \tag{12}$$

These six formulas represent indeed the result expected from reliability theory, i.e., they determine the structure function of the example. The formulas corresponds to the six minimal paths given in (Kohlas, 1987).

4.4. *Probabilistic Argumentation and Decision Systems*

So far we have only been concerned with symbolical results. However, given the numerical information we can weigh the arguments and compute – for example – the reliability. Along similar lines as in the case of PAS (cf. Section 3.4), we introduce probabilities into the framework of PADS. Using the same notation, we define the degree of support dsp_A for hypotheses h by

$$dsp_A(h, \Sigma; D) = p'(\mu SP_A(h, \Sigma; D)) = \frac{p(\mu SP_A(h, \Sigma; D))}{1 - p(\mu QS_A(\bot, \Sigma))}. \tag{13}$$

In the case of PAS, such a probability can be computed directly using the concept of Dempster–Shafer belief functions (Haenni and Lehmann, 2003; Haenni and Lehmann, 2001) with local computations (cf. Section 5.3). It is not yet clear if and how this approach can be extended to PADS.

5. Computation

Consider a PADS (P, D, Σ) and a set of assumptions $A \subseteq P - D$. In this section we focus on the problem of computing the set of minimal arguments for a hypothesis $h \in \mathcal{L}_P$.

5.1. *Arguments*

Clearly, the computation of a compact representation of the supporting scenarios and the supporting decisions is a key point. We re-use here the well-developed concepts from PAS. First, we have to introduce the concept of projection w.r.t. conjunctions. Consider a conjunction $\alpha \in C_P$. The projection $\alpha^{\downarrow P'}$ of α to the subset $P' \subseteq P$ means to eliminate all literals outside P' from the conjunction. The projection of a set of formulas $S \subseteq C_P$ to P' is computed using the projection of every element, $S^{\downarrow P'} = \{\alpha^{\downarrow P'} : \alpha \in S\}$.[3]

For a discussion of the computation of minimal arguments in PAS see Section 5.3 and (Haenni *et al.*, 2000; Kohlas *et al.*, 1999). For the computation of arguments in a PADS, we can use the techniques of computing arguments in PAS by changing temporarily the type of some variables:

Lemma 3. $N_A(SP_A(h, \Sigma; D)) = N_A((SP_{A\cup D}(h, \Sigma))^{\downarrow A})$.

The lemma above is stated on the level of scenarios. If we try to lift it up to the level of arguments, we can only prove the following result:

Lemma 4. $SP_A(h, \Sigma; D) \supseteq (SP_{A\cup D}(h, \Sigma))^{\downarrow A}$.

This means that the corresponding PAS allows to compute only a subset of the supporting arguments for the PADS, yet the "missing ones" are included in larger arguments:

COROLLARY 1. For every $\alpha \in SP_A(h, \Sigma; D)$ there is a $\alpha' \in (SP_{A\cup D}(h, \Sigma))^{\downarrow A}$ so that $\alpha \wedge \beta = \alpha'$ for some $\beta \in C_A$.

Hence from PAS, essentially a correct logical representation of the support in PADS can be computed; however, some arguments might be missing. The same situation arises in the computation of arguments in PAS, and the use of the operator *Cons*$_A$ for computing all arguments is discussed in (Haenni *et al.*, 2000). These results can be applied to our situation.

[3]Note that this operation denotes the projection; this is different from variable elimination (Haenni *et al.*, 2000) also known as variable forgetting!

DEFINITION 9. The operator $\overline{Cons}_A$ is defined on a set of conjunctions C as follows, with $A = \{a_1, \ldots, a_n\}$:

$$\overline{Cons}_A(C) = \neg(Cons_{a_1}(Cons_{a_2}(\cdots(Cons_{a_n}(\neg C))\cdots))) \tag{14}$$

and $Cons_a$ is the set of all resolvents with respect to the assumption a, i.e., $Cons_a(\Xi) = \Xi \cup \{\rho(\xi, \xi') : \xi, \xi' \in \Xi\}$ for a set of clauses Ξ and ρ as in (Haenni *et al.*, 2000).

Note that the definition is unambiguous since the order of the assumptions a_1 to a_n in (14) does not matter (Haenni *et al.*, 2000). The order however is crucial for the efficiency of computations (see also Section 5.3).

Lemma 5. $SP_A(h, \Sigma; D) = \overline{Cons}_A(SP_{[A\cup D}(h, \Sigma))^{\downarrow A})$ *and therefore we have also* $\mu SP_A(h, \Sigma; D) = \mu\overline{Cons}_A((\mu SP_{A\cup D}(h, \Sigma))^{\downarrow A})$.

Note that the negation applied to a set of conjunctions means to apply the negation to every conjunction in the set, which results in a set of clauses, and vice versa. The set of supporting arguments $\mu SP_{A\cup D}(h, \Sigma)$ can be computed from $\mu QS_{A\cup D}(h, \Sigma)$ and $\mu QS_{A\cup D}(\bot, \Sigma)$ (Haenni *et al.*, 2000) as follows:

Algorithm *Algo1*
Input: $\mu QS_{A\cup D}(h, \Sigma)$, $\mu QS_{A\cup D}(\bot, \Sigma)$
Output: $\mu SP_{A\cup D}(h, \Sigma)$

let $R = \emptyset$
for every $\alpha \in \mu QS_{A\cup D}(\delta, \Sigma)$
  transform $\alpha \wedge \bigwedge \neg\mu QS_{A\cup D}(\bot, \Sigma)$ into a DNF
  add the conjunctions to R
return $\mu\overline{Cons}_A(\mu R)$

The supporting decision arguments can be computed similarly using the following results:

Lemma 6. $SD_A(h, \Sigma; D) = SP_{A\cup D}(h, \Sigma), \quad \mu SD_A(h, \Sigma; D) = \mu SP_{A\cup D}(h, \Sigma)$.

5.2. *Improvements in the Computation*

The computations described in the previous chapter are correct, yet in applications, one becomes aware that some of them require a lot of time. Where is the problem? The computation of a typical support $\mu SP_A(h, \Sigma; D)$ as defined above requires according to Lemma 5 essentially that we determine $\mu SP_{A\cup P}(h, \Sigma)$. This is done by computing the two minimal sets of arguments $\mu QS_{A\cup D}(h, \Sigma)$ and $\mu QS_{A\cup D}(\bot, \Sigma)$, which both can be computed usually in short time. The computationally hard part is then the combination of them according to the algorithm *Algo1*. There, especially the conversion of a negated DNF into a DNF is time consuming, i.e., a translation from a CNF to a DNF. This is a

problem whose complexity is known to be non-polynomial (without the introduction of additional literals). This problem arose very clearly computing the example of energy distribution system with more redundancy (Frey and Reichert, 1973; Anrig and Baziukaitė, 2003).

Here, we are in the situation where we are not interested in the supporting arguments $\mu SP_{A\cup D}(h,\Sigma)$ but only in $\mu SP_A(h,\Sigma;D)$. Hence we essentially have to combine the computation of $\mu SP_{A\cup D}(h,\Sigma)$ with the subsequent projection to A. The following algorithm exploits the special situation:

Algorithm *Algo2*
Input: $\mu QS_{A\cup D}(\delta,\Sigma)$, $\mu QS_{A\cup D}(\bot,\Sigma)$
Output: $\mu SP_A(\delta,\Sigma;D)$

let $R=\emptyset$
for every $\alpha\in\mu QS_{A\cup D}(\delta,\Sigma)$
let $\Xi_\alpha=\left\{\beta^{\downarrow(var(\alpha))^c}:\begin{array}{l}\beta\in\mu QS_{A\cup D}(\bot,\Sigma),\\ var(\beta)\subseteq A\cup var(\alpha)\\ lit(\beta)\cap lit(\neg\alpha)=\emptyset\end{array}\right\}$
transform $\alpha\wedge\bigwedge\neg\Xi_\alpha$ into a DNF $\delta=\delta_1\vee\cdots\vee\delta_{q(\alpha)}$
and add $\delta_i^{\downarrow A}$ to R for every $i=1,\ldots,q(\alpha)$
return $\mu\overline{Cons}_A(\mu R)$

Remember that for a conjunction $\alpha=\alpha_1\wedge\cdots\wedge\alpha_q$, the set of literals $lit(\neg\alpha)$ consists of the literals contained in the clause $\neg\alpha_1\vee\cdots\vee\neg\alpha_q$.

Lemma 7. *The result of the algorithm is equal to* $\mu SP_A(\delta,\Sigma,D)$.

EXAMPLE 4. Consider the set of assumptions $A=\{a_1,a_2,a_3\}$ and the decision variables $D=\{d_1,d_2\}$. Assume that we have computed

$$\begin{aligned}\mu QS_{A\cup D}(\delta,\Sigma)&=\{a_1\wedge d_1,\ d_2,\ a_2\}\\ \mu QS_{A\cup D}(\bot,\Sigma)&=\{a_3\wedge d_2,\ \neg d_1\wedge a_2,\ a_1\wedge\neg a_2\wedge d_1\}\end{aligned}$$

and we have to compute the set of supporting arguments $\mu SP_A(\delta,\Sigma;D)$.

We use the algorithm *Algo2*. First, we set $R=0$. Consider the first element $a_1\wedge d_1$. According to the algorithm, we have to look for elements in $\mu QS_{A\cup D}(\bot,\Sigma)$ which respect the two conditions of the definition of Ξ_α. The first argument, $a_3\wedge d_2$ does not respect the first condition because $var(a_3\wedge d_2)=\{a_3,d_2\}\not\subseteq A\cup var(a_1\wedge d_1)=\{a_1,a_2,a_3,d_1\}$. Hence this argument can be omitted. The second one, $\neg d_1\wedge a_2$, does not respect the second condition because $lit(\neg d_1\wedge a_2)\cap lit(\neg(a_1\wedge d_1))=\{\neg d_1,a_2\}\cap\{\neg a_1,\neg d_1\}=\{\neg d_1\}\neq\emptyset$ and can therefore be omitted too. The third one, $a_1\wedge\neg a_2\wedge d_1$, does respect both conditions, hence we have to project it to $(var(a_1\wedge d_1))^c=\{a_2,a_3,d_2\}$, i.e., $(a_1\wedge\neg a_2\wedge d_1)^{\downarrow\{a_2,a_3,d_2\}}=\neg a_2$. Then, the resulting formula $a_1\wedge d_1\wedge\neg(a_2)$ has to be transformed into a DNF (which it already is in this case)

and its projection to A is added to R, hence $R = \{a_1 \wedge \neg a_2\}$. The same procedure is now done with the second element d_2 of $\mu QS_{A\cup D}(\delta, \Sigma)$. This yields $R = \{a_1 \wedge \neg a_2,\ \neg a_3\}$ and, using the third element a_2, we get $R = \{a_1 \wedge \neg a_2,\ \neg a_3,\ a_2\}$. Finally, we have to apply the minimality operator μ which yields $\mu R = \{\neg a_3,\ a_2\}$ and then apply the operator $\overline{Cons}_A$ to μR which, in this special case, does not change anything, hence the result is $\mu SP_A(\delta, \Sigma) = \{\neg a_3,\ a_2\}$.

5.3. *Local Computations and Approximation Techniques*

For the computation of supporting arguments, we can re-use techniques from PAS (Haenni *et al.*, 2000) based on the concept of variable elimination. One of the main problems of variable elimination (Haenni *et al.*, 2000) is the order in which these variables are actually eliminated (the results are equivalent for all orderings, but the computation time depends very much on the ordering). An ordering corresponds to a computation on a certain hypertree structure, a local computation technique based on ideas from Lauritzen & Shenoy (Lauritzen and Shenoy, 1995) which allows to distribute the computations on different virtual or real processors (Shenoy and Kohlas, 2000; Kohlas, 2003). We refer also to (Anrig and Kohlas, 2002a) for applications of these computation techniques to reliability and diagnostic.

Based on these symbolical results, the numerical results can then be computed using orthogonalization techniques in reliability theory (Abraham, 1979; Heidtmann, 1989; Heidtmann, 1997; Bertschy and Monney, 1996; Anrig and Beichelt, 2001). Besides, new promising approches focus on more general decomposition techniques (Darwiche and Marquis, 2001; Darwiche, 2002). If we are only interested in numerical results, then there is a second, often more efficient way, cf. Section 4.4.

When applying the framework of argumentation systems to larger problems, there is need for approximation techniques (Haenni, 2001; Haenni, 2001a); yet its application to PADS is subject to further research.

6. Conclusion

We have shown how the well-known formalism of PAS can be enriched with decision variables. A specialized algorithm for computing the respective arguments has been presented. This allows now to use this framework for example in model-based reliability theory (Anrig and Kohlas, 2002; Anrig and Kohlas, 2002a) for computing structure functions.

Several open questions arose during the research. First, the numerical counterpart of PAS is – in some sense – the computation with belief functions (Haenni and Lehmann, 2003; Haenni and Lehmann, 2001). But is it possible to apply the same concepts for PADS? Second, several approximation techniques are known for PAS and also for belief functions. Is it possible to use the same techniques for PADS? Using these approximation technique we would get then (symbolical) upper and lower bounds of structure functions in reliability theory.

Acknowledgments

Most of this work was done while the second author was visiting researcher at the Department of Informatics, University of Fribourg, Switzerland. The visit was supported by Swiss Baltic Net (Gebert Ruef Foundation).

We thank Jürg Kohlas and Norbert Lehmann for critical remarks on earlier drafts.

References

Abraham, J. A. (1979). An improved algorithm for network reliability. *IEEE Transactions on Reliability*, **28**, 58–61.

Anrig, B. (2000). *Probabilistic argumentation systems and model-based diagnostics*. In A. Darwiche and G. M. Provan (Eds.), DX'00, *Eleventh Intl. Workshop on Principles of Diagnosis*. Morelia, Mexico. pp. 1–8.

Anrig, B., and D. Baziukaitė (2003). Probabilistic argumentation and decision system. *Technical Report 03-03*, Department of Informatics, University of Fribourg.

Anrig, B., and D. Baziukaitė (2003a). Probabilistic argumentation and decision system. Environmental Reasearch. *Engineering and Management*, **4**(26), 56–61.

Anrig, B., and F. Beichelt (2001). Disjoint sum forms in reliability theory. *ORiON J.*, OR Society South Africa, **16**(1), 75–86.

Anrig, B., R. Bissig, R. Haenni, J. Kohlas, N. Lehmann (1999). Probabilistic argumentation systems: Introduction to assumption-based modeling with ABEL. *Technical Report 99-1*, University of Fribourg, Institute of Informatics.

Anrig, B., and J. Kohlas (2002). Model-based reliability and diagnostic: A common framework for reliability and diagnostics. In M. Stumptner and F. Wotawa (Eds.), DX'02, *13th Intl. Workshop on Principles of Diagnosis*. Semmering, Austria. pp. 129–136.

Anrig, B., and J. Kohlas (2002a). Probabilistic argumentation and decision systems. An application to reliability theory. In B. Wolfinger and K. Heidtmann (Eds.), *2. MMB-Arbeitsgespräch: Leistungs-, Zuverlässigkeits- und Verlässlichkeitsbewertung von Kommunikationsnetzen und verteilten Systemen*. pp. 75–82.

Beichelt, F. (1993). *Zuverlässigkeits- und Instandhaltungstheorie*. Teubner, Stuttgart.

Bertschy, R., and P.-A. Monney (1996). A generalization of the algorithm of Heidtmann to non-monotone formulas. *J. of Computational and Applied Math.*, **76**, 55–76.

Darwiche, A. (2002). A compiler for deterministic decomposable negation normal form. In *Proc. of the National Conf. on Artif. Intell (AAAI 2002)*. pp. 627–634.

Darwiche, A., and P. Marquis (2001). A perspective on knowledge compilation. In *Proc. 17th Int. Joint Conf. on Artif. Intell. IJCAI-01*.

Frey, H., and K. Reichert (1973). Anwendungen moderner Zuverlässigkeits-Analysenmethoden in der elektrischen Energieversorgung. *Elektrotechnische Zeitschrift ETZ-A*, **94**, 249–255.

Haenni, R. (2001). Cost-bounded argumentation. *Int. J. of Approximate Reasoning*, **26**(2), 101–127.

Haenni, R. (2001a). A query-driven anytime algorithm for assumption-based reasoning. *Technical Report 01-26*, University of Fribourg, Department of Informatics.

Haenni, R., J. Kohlas and N. Lehmann (2000). Probabilistic argumentation systems. In J. Kohlas and S. Moral (Eds.), *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, Vol. 5, Algorithms for Uncertainty and Defeasible Reasoning. Kluwer, Dordrecht. pp. 221–287.

Haenni, R., and N. Lehmann (2001). Implementing belief function computations. *Technical Report 01-28*, University of Fribourg, Department of Informatics.

Haenni, R., and N. Lehmann (2003). 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.

Heidtmann, K. (1989). Smaller sums of disjoint products by subproduct inversion. *IEEE Transactions on Reliability*, bf 38(3), 305–311.

Heidtmann, K. (1997). *Zuverlässigkeitsbewertung Technischer Systeme*, Teubner.

Kohlas, J. (1987). *Zuverlässigkeit und Verfügbarkeit*, Teubner.

Kohlas, J. (2003). *Information Algebras: Generic Structures for Inference*, Springer.

Kohlas, J., B. Anrig, R. Haenni and P.-A. Monney (1998). Model-based diagnostics and probabilistic assumption-based reasoning. *Artif. Intell.*, **104**, 71–106.

Kohlas, J., D. Berzati and R. Haenni (2000). Probabilistic argumentation systems and abduction. In C. Baral and M. Truszczyński (Eds.), *Proc. of the 8th Int. Workshop on Non-Monotonic Reasoning*. Breckenridge Colorado.

Kohlas, J., R. Haenni and S. Moral (1999). Propositional information systems. *J. of Logic and Computation*, **9**(5), 651–681.

Kohlas, J., and P.-A. Monney (1995). *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.

Laskey, K. B., and P.E. Lehner (1989). Assumptions, beliefs and probabilities. *Artif. Intell.*, **41**, 65–77.

Lauritzen, S. L., and P.P. Shenoy (1995). Computing marginals using local computation. *Working Paper 267*, School of Business, University of Kansas.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann Publ. Inc.

Provan, G.M. (1990). A logic-based analysis of Dempster–Shafer theory. *Int. J. of Approximate Reasoning*, **4**, 451–495.

Provan, G.M. (2000). An integration of model-based diagnosis and reliability theory. In A. Darwiche and G. M. Provan (Eds.), DX'00, *Eleventh Intl. Workshop on Principles of Diagnosis*. Morelia, Mexico. pp. 193–200.

Shafer, G. (1976). *The Mathematical Theory of Evidence*. Princeton University Press.

Shenoy, P.P., and J. Kohlas (2000). Computation in valuation algebras. In J. Kohlas and S. Moral (Eds.), *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, Vol. 5, Algorithms for Uncertainty and Defeasible Reasoning. Kluwer, Dordrecht. pp. 5–40.

B. Anrig is an associate professor at the Berne University of Applied Sciences in Biel, Switzerland. His main interests include Uncertainty in Artificial Intelligence, especially Assumption-Based Reasoning, as well as Cryptology and its Applications. He is member of the Virtual Identity and Privacy Research Center (http://www.vip.ch).

D. Baziukaitė recieved her MCs degree in Mathematics from Klaipėda University. Currently she is working towards her PhD degree in computer science and mathematics at Klaipėda University. Her research is focused on adaptivity, intelligence, and decision making processes in virtual learning environments. She is a member of the National Association of Distance Education.

Tikimybinės argumentavimo sistemos su sprendimo priėmimo kintamaisiais

Bernhard ANRIG, Dalia BAZIUKAITĖ

Tikimybinių argumentavimo sistemų sąvoka yra apribota dviejų tipų kintamaisiais: prielaidomis, kuriomis modeliuojama neapibrėžtoji žinių dalis, ir teiginiais, modeliuojančiais likusią informacijos dalį. Atskiri tikimybinių argumentavimo sistemų taikymo atvejai sutinkami įvairiose probleminėse srityse, sprendžiant skirtingos prigimties uždavinius. Šiame darbe skaitytojas supažindinamas su trečio tipo kintamaisiais, kuriuos vadiname sprendimo priėmimo kintamaisiais. Šis naujas kintamųjų tipas leidžia išreikšti vartotojo sprendimus, kuriuos šis gali priimti reaguodamas į tam tikrą sistemos būseną. Straipsnyje pristatomas argumentų su sprendimo priėmimo kintamaisiais apskaičiavimo algoritmas.