Propositional Information Systems

JÜRG KOHLAS and ROLF HAENNI, *Institute of Informatics, University of Fribourg Switzerland.*
E-mail: {juerg.kohlas,rolf.haenni}@unifr.ch

SERAFÌN MORAL, *Department of Computer Science, University of Granada, Spain.*
E-mail: smc@decsai.ugr.es

Abstract

Resolution is an often used method for deduction in propositional logic. Here a proper organization of deduction is proposed which avoids redundant computations. It is based on a generic framework of decompositions and local computations as introduced by Shenoy and Shafer. The system contains the two basic operations with information, namely marginalization (or projection) and combination; the latter being an idempotent operation in the present case. The theory permits the conception of an architecture of distributed computing. As an important application assumption-based reasoning is discussed.

Keywords: Propositional logic, information systems, assumption-based reasoning, marginalization problem, valuation networks.

1 Introduction

In many practical applications knowledge and information are essentially encoded in propositional logic. Using and exploiting such propositional knowledge bases involve essentially deduction and theorem proving. There is of course a wealth of well-known and established methods and procedures in propositional logic for doing exactly this. Nevertheless, we propose here a new look at this problem based on a decomposition of the knowledge base. This is an unusual point of view in logic. However, it is a widespread method in other formalisms of reasoning such as Bayesian networks [24], evidence theory [29], and others.

It will be argued and shown that propositional logic fits well into the corresponding axiomatic framework of local propagation in decomposed systems as introduced by Shafer and Shenoy [29]. Therefore, the computational methods derived from this formalism may be valuable alternatives to the usual deduction and theorem proving methods. In particular, this is true with respect to assumption-based reasoning, a variant of ATMS (Assumption-Based Truth Maintenance Systems). In fact, it is well known that assumption-based reasoning is closely related to evidence theory [23, 26]. Therefore, it is obvious that methods useful in evidence theory are also valuable in logic.

The techniques presented in this paper have been successfully implemented in ABEL[1], a logic-based language for assumption-based reasoning under uncertainty [2, 3]. The inference mechanism of ABEL is based on a combination of classical deduction techniques such as resolution and the ideas of decomposition and local propagation.

[1]More information about ABEL as well as a free copy of the software can be obtained from `http://www-iiuf.unifr.ch/tcs/abel`.

1.1 Formulation of the problem and overview

Formally, suppose that knowledge is encoded in a set $\Sigma = \{\xi_1, \xi_2, \ldots, \xi_n\}$ of propositional formulae over a set of propositional symbols P. Basically we are interested in consequences h of Σ which belong to the propositional sub-language over a subset of symbols $Q \subseteq P$. For example, we may want to know all prime implicates of Σ belonging to this sub-language or we may want to decide on many queries h in the sub-language, whether Σ entails h, written $\Sigma \models h$, or not (see Section 1.2 for a discussion of these problems and their interest). It may be worthwhile in such a case to first compile the knowledge Σ into a set Σ' of formulae over Q and then use Σ' instead of Σ to see whether the hypotheses can be deduced or not. Of course Σ' must satisfy the conditions

(C1) $\Sigma \models \Sigma'$,

(C2) $\Sigma \models h$ implies $\Sigma' \models h,$ for all h expressible with symbols in $Q \subseteq P$.

Such a set of formulae Σ' will be called a marginal of Σ with respect to Q. A special case arises for $Q = Ø$. Then, only $\perp$ and $\top$ are possible marginals with respect to Ø. In the first case Σ is not satisfiable. Thus, the problem of computing marginals encloses also the problem of deciding about satisfiability.

Once a marginal Σ' of Σ to Q is found, there are cases for which it is conceivably simpler to test whether $\Sigma' \models h$ or to derive consequences of Σ', than to decide whether $\Sigma \models h$ or to compute consequences of Σ. However, note that in particular cases the size of Σ' grows exponentially with the number of symbols in $P - Q$. Thus, marginalization is not always preferable. The usefulness of marginalization depends on the structure of the particular problem to which it is applied. For the problem of satisfiability checking, a comparison of the marginalization technique with other classical algorithms can be found in [11].

The problem of finding a marginal Σ' of Σ with respect to some Q is called the **marginalization problem**. Its solution is discussed in Section 2. It is well known that it can be solved by resolution for systems of clauses [13]. The same problem can also be solved by methods of mathematical programming [13, 34]. Here, also more general systems of disjunctive normal forms will be considered (Section 2.5). Often one wants the marginal to several subsets Q, not to just one subset. A convenient organization of the computations may then reduce the effort considerably by avoiding repeating the same computations. It may also help in updating marginals when new knowledge is added. That is where decomposition and local computation in join trees enters just in the same way as in comparable problems of marginalizations of probability distributions in Bayesian networks or of belief functions in evidence nets. The propagation algorithms provide a compilation of the knowledge base, from which sound and faster deductions can be carried out. In fact, it will be shown that propositional information Σ satisfies the basic axioms introduced by Shenoy, Shafer [29] for local propagation in join trees (Section 2.2). Actually, propositional information systems satisfy an additional idempotency axiom which permits to simplify computations.

In Section 3 an important application of marginalization is discussed. Assumption-based reasoning is closely related to abduction and circumscription [14] but also to evidence theory [23, 26]. And it has already been shown that it can be put into the framework of propagating information in join trees [16, 10]. However, an alternative approach based on the theory of propositional information systems will be exploited in Section 3 and its connection to the former approach will be outlined. The method described here corresponds to the inference mechanism implemented for ABEL. This implementation was the main motivation for the development of the theory described in this paper.

The idempotent algebraic structure underlying these local propagation methods is very appealing. In fact, information must be combined and it must be possible to reduce information to some coarser frame. Propositional information systems are a case. For propositional information Σ_1 and Σ_2 combination is just union. The reducing of information corresponds to marginalization. The related general abstract algebraic structure of information is discussed in [20].

1.2 *Application of marginalization*

The marginalization problem is basic to a number of important application fields related to consequence finding. Assumption-based reasoning in particular is an important application domain of consequence finding. We will use this application to illustrate the importance of marginalization and to compare it with other approaches to consequence finding. The fundamental problem of assumption-based reasoning can be described as follows:

Let Σ be a finite set of propositional formulae over propositional symbols in a set N. A subset $A \subseteq N$ is singled out and the symbols in A are called assumptions. Let C_A denote the set of all conjunctions of literals from A not containing simultaneously a literal and its negation. The elements of C_A are called arguments. If h is another propositional formula over N, then an argument $a \in C_A$ is called support for h given Σ, if

(1) $\Sigma, a \models h$,
(2) $\Sigma, a \not\models \emptyset$ (that is Σ and a are satisfiable).

The set

$$QS(\Sigma, h) = \{a \in C_A : \ \Sigma, a \models h\} \tag{1.1}$$

is called the set of quasi-supports of hypothesis h. In particular,

$$QS(\Sigma, \perp) = \{a \in C_A : \ \Sigma, a \models \emptyset\} \tag{1.2}$$

is called the set of contradictions. The set

$$SP(h, \Sigma) = QS(\Sigma, h) - QS(\Sigma, \perp) \tag{1.3}$$

is then the set of all supports for h given Σ. Support can therefore be expressed in terms of quasi-support. Thus, the main problem of assumption-based reasoning consists in computing sets of quasi-supports. Note that such a system is closely related to abduction and circumscription as Inoue [14] has pointed out.

Generally, the set $QS(\Sigma, h)$ of all quasi-supports of a hypothesis h is too big to be computed or stored explicitly. Therefore, all methods developed in the domain of assumption-based reasoning use a shorter representation of $QS(\Sigma, h)$. Every subset $QS' \subseteq QS(\Sigma, h)$ for which

$$\bigvee\{a \in QS'\} \equiv \bigvee\{a \in QS(\Sigma, h)\} \ \ (\equiv \text{means logical equivalence}) \tag{1.4}$$

can be used as an alternative representation of $QS(\Sigma, h)$. One particular subset is the set of minimal quasi-supports $\mu QS(\Sigma, h)$, that is all elements $a \in QS(\Sigma, h)$ such that no proper sub-conjunction of a belongs to $QS(\Sigma, h)$. Note that $\mu QS(\Sigma, h)$ is the set of prime implicants of any subset QS' for which (1.4) holds. From this point of view it becomes clear that often a set QS' exists which is considerably smaller than $\mu QS(\Sigma, h)$.

In view of these remarks let us formulate two basic problems:

(P1) For a formula h over Q, find a subset $QS' \subseteq QS(\Sigma, h)$ which is logically equivalent to $QS(\Sigma, h)$ in the sense of (1.4).

(P2) For a formula h over Q, find the set $\mu QS(\Sigma, h)$ of all minimal quasi-supports.

Note that (P2) is clearly a special case of (P1). (P2) is essentially the problem of finding minimal labels in ATMS. In probabilistic assumption-based reasoning [18], if probabilities are assigned to the assumptions, then the problem is to compute numerical degrees of support, i.e. the probability that h is supported given Σ. For this problem, solving (P1) is sufficient and often much simpler than (P2).

Inoue [14] solves (P2) by linear resolution. He defines the notion of characteristic clauses of Σ with respect to P, $Carc(\Sigma, P)$, which are the minimal clauses over P that are consequences of Σ (i.e. prime implicates of Σ containing only literals of symbols from P). Then, it can be shown [14, 19] that

$$\begin{aligned} \mu QS(\Sigma, h) &= {\sim}Carc(\Sigma \cup \{{\sim}h\}, A), && (1.5)\\ \mu QS(\Sigma, \bot) &= {\sim}Carc(\Sigma, A), && (1.6)\\ \mu SP(\Sigma, h) &= {\sim}(Carc(\Sigma \cup \{{\sim}h\}, A) - Carc(\Sigma, A)) \\ &= {\sim}Newcarc(\Sigma, {\sim}h, A). && (1.7) \end{aligned}$$

Here ${\sim}Carc(\Sigma, A)$ denotes the set of conjunctions obtained by negating the clauses in $Carc(\Sigma, A)$. The basic operation of Inoue is to compute $Newcarc(\Sigma, {\sim}h, A)$, or more generally $Newcarc(\Sigma, F, A)$ for an arbitrary formula F. This operation is first used to compute incrementally $Carc(\Sigma, A)$, and then, each time a formula h arises, to obtain $\mu QS(\Sigma, h)$ and $\mu SP(\Sigma, h)$.

A well-known result of Reiter and de Kleer [27] also solves (P2). The idea is that if h is a clause, then the minimal quasi-supports can easily be filtered from the set of prime implicates $PI(\Sigma)$. The problem here is that in most cases the set $PI(\Sigma)$ is too big and cannot be determined explicitly. If Σ consists of Horn clauses, then very efficient algorithms exist [9].

Marginalization can help to solve the problems (P1) and (P2) in two different ways:

(1) From Theorem 3.3 given later in Section 3.1 we know that if Σ' is a marginal of Σ to $Q \cup A$, then for all formula h over Q

$$QS(\Sigma, h) = QS(\Sigma', h). \qquad (1.8)$$

This theorem tells us that we may first marginalize Σ to $Q \cup A$ and only then solve (P1) or (P2) with respect to the marginal Σ'. If there are different hypotheses, all expressible in Q, then the marginal Σ' has only to be computed once. In this way, a number of redundant resolutions can be avoided. If, for example, the problem is to compute the set of supports $SP(\Sigma, h)$, then by (1.3) it is necessary to know the contradictions $QS(\Sigma, \bot)$, and therefore we have at least one other hypothesis $h = \bot$ which is always expressible in Q. Furthermore, as we will see in Subsection 2.2, if there are hypotheses on different subsets $Q_1, Q_2, \ldots$, then the computation can be organized by join trees which again helps to avoid many redundant resolutions.

(2) The problem (P1) can be solved by computing a marginal Σ'' of $\Sigma \cup \{{\sim}h\}$ relative to the set A of assumptions. Alternatively, it is also possible to use Σ' from above and to compute a marginal Σ'' of $\Sigma' \cup \{{\sim}h\}$ relative to A. Then, according to Theorem 3.2 given

in Section 3.1, we know that $QS' = {\sim}\Sigma''$ is a subset of $QS(\Sigma, h)$ for which (1.4) holds. Again, note that this set is often considerably smaller than $\mu QS(\Sigma, h)$, but nevertheless, sufficient for the computation of numerical degrees of support, and for solving (P2) by computing the prime implicants.

If we combine the techniques of computing $Newcarc$ proposed by Inoue and the marginalization method presented in this paper, then as illustrated in Figure 1 it is possible to solve the problems (P1) and (P2) in a number of different ways. The method recommended in this

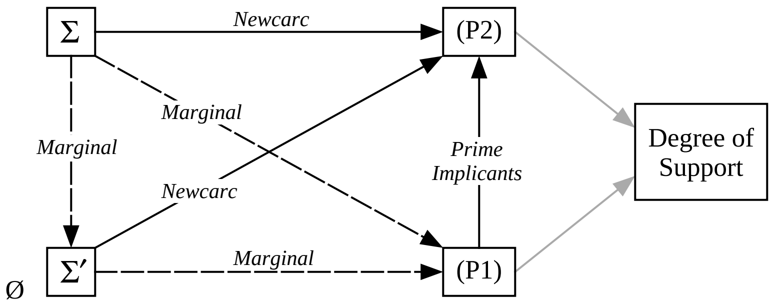

FIGURE 1. Different ways of computing quasi-support

paper consists then of three sequential steps: (1) use a join tree to obtain a marginal Σ' of Σ to $Q \cup A$; (2) use Σ' to solve the problem (P1) as described above; (3) compute either numerical degrees of support, or if necessary solve (P2).

The main advantage of this method is that the intermediate results stored in the join tree (for example Σ') can be reused for other hypotheses (for example $h = \perp$). Another important point is that the result obtained for (P1) is often considerably smaller than the result for (P2).

The application of marginalization to assumption-based reasoning will be discussed more in detail in Section 3.

2 Computation with propositional information

A propositional information is given by a set of well-formed propositional formulae $\Sigma = \{\xi_1, \ldots, \xi_m\}$. These formulae are of course to be interpreted in a conjunctive way: ξ_1 and ξ_2 and ... and ξ_m are true. We denote by $c(\Sigma)$ the set of propositional symbols occurring in the formulae of Σ. Furthermore, $\mathcal{L}_P$ denotes the set of well-formed formulas over the set P of propositional symbols. The elements of Σ can be considered as belonging to $\mathcal{L}_P$, whenever $c(\Sigma) \subseteq P$. Now, for the purpose of the computational theory to be developed, a propositional information Σ should always be considered as belonging to a $\mathcal{L}_P$ for a determined set P of propositional symbols. More precisely, a **propositional information** is considered to be a pair (Σ, P) with $c(\Sigma) \supseteq P$. Note that (Σ, P) and (Σ, P') are to be considered as two different propositional informations unless $P = P'$. P is called the **label** of a propositional information (Σ, P).

EXAMPLE 2.1
Consider a propositional information (Σ, P) with $\Sigma = \{\xi_1, \ldots, \xi_{17}\}$ defined as follows:

$$\begin{array}{lll}
\xi_1 : & \sim b \vee \sim a_1 \vee a & = b \wedge a_1 \rightarrow a, \\
\xi_2 : & \sim e \vee \sim a_1 \vee a & = e \wedge a_1 \rightarrow a, \\
\xi_3 : & \sim a_2 \vee a & = a_2 \rightarrow a, \\
\xi_4 : & b \vee e \vee a_2 \vee \sim a & = \sim b \wedge \sim e \wedge \sim a_2 \rightarrow \sim a, \\
\xi_5 : & a_1 \vee a_2 \vee \sim a & = \sim a_1 \wedge \sim a_2 \rightarrow \sim a, \\
\xi_6 : & \sim a \vee w & = a \rightarrow w, \\
\xi_7 : & \sim a_3 \vee w & = a_3 \rightarrow w, \\
\xi_8 : & a \vee a_3 \vee \sim w & = \sim a \wedge \sim a_3 \rightarrow \sim w, \\
\xi_9 : & \sim a \vee a_4 \vee g & = a \wedge \sim a_4 \rightarrow g, \\
\xi_{10} : & \sim a_4 \vee \sim g & = a_4 \rightarrow \sim g, \\
\xi_{11} : & a \vee \sim g & = \sim a \rightarrow \sim g, \\
\xi_{12} : & \sim a \vee \sim a_5 \vee d & = a \wedge a_5 \rightarrow d, \\
\xi_{13} : & a_5 \vee \sim d & = \sim a_5 \rightarrow \sim d, \\
\xi_{14} : & a \vee \sim d & = \sim a \rightarrow \sim d, \\
\xi_{15} : & \sim e \vee \sim a_6 \vee c & = e \wedge a_6 \rightarrow c, \\
\xi_{16} : & e \vee \sim c & = \sim e \rightarrow \sim c, \\
\xi_{17} : & a_6 \vee \sim c & = \sim a_6 \rightarrow \sim c.
\end{array}$$

The label of this propositional information can be taken as

$$P = c(\Sigma) = \{a, b, c, d, e, g, w, a_1, a_2, a_3, a_4, a_5, a_6\}.$$

This propositional information describes a small story around an alarm system [25]. The propositional symbols occurring in the clauses have to be interpreted as follows:

a:	the alarm system in the house of Mr Holmes is ringing,
b:	there is a burglary,
e:	an earthquake has occurred,
c:	there is confirmation of the earthquake on the radio,
w:	the neighbour of Mr Holmes, Mr Watson phones Mr Holmes,
g:	the neighbour, Mrs Gibbson phones Mr Holmes,
d:	the daughter of Mr Holmes phones.

The first four rules (ξ_1 to ξ_5) tell us, that a burglary generates an alarm in the house of Mr Holmes, if the alarm system is functioning (a_1), but so does also an earthquake. Other causes (a_2) may also cause an alarm. And these are the only ways an alarm can arise (rules ξ_4 and ξ_5). Then the next three rules ξ_6 to ξ_8 say that the neighbour of Mr Holmes, Mr Watson, phones Mr Holmes, if there is an alarm. But Mr Watson may also alarm Mr Holmes as a joke (a_3). The other neighbour of Mr Holmes, Mrs Gibbson, phones also Mr Holmes, when there is an alarm and she is able to hear it (a_4) (rules ξ_9 to ξ_{11}). Furthermore, if the daughter of Mr Holmes is at home (a_5), then she surely phones also, if there is an alarm. Finally, if there is an earthquake, there is a confirmation of it on the radio, if the earthquake was registered (a_6) (this is what rules ξ_{15} to ξ_{17} say).

If additional facts become known like $\xi_{18} = w$ (Mr Watson phones), $\xi_{19} = \sim g$ (Miss Gibbson does not phone) , $\xi_{20} = \sim c$ (there is no confirmation of an earthquake), the propositional information Σ is enlarged by adding these three formulae. The label does not change in this particular case.

As explained in the introduction one might be interested especially in hypotheses which can be expressed by propositional formula over a subset $Q \subseteq P$ of propositional symbols. In the example above one might be interested especially whether the alarm rang or not and whether there is a burglary or not, hypotheses which can be expressed by symbols in $Q = \{a, b\}$. In fact, in this example the symbols in $A = \{a_1, \ldots, a_6\}$ play a particular role and we may want to include them into the computation. This is made clear in Section 3. It means that we are interested in formulae which can be expressed using the symbols in $Q \cup A = \{a, b, a_1, \ldots, a_6\}$.

2.1 The marginalization problem

It has been argued in Section 1 that it is often advantageous to **marginalize** propositional information to a subset Q of propositional symbols. Here we take up this idea and formalize first the notion of a marginalization relation. If (Σ, P) is a propositional knowledge with label P, and (Σ', Q) a propositional information with label Q, then (Σ, P) and (Σ', Q) are said to satisfy the **marginalization relation** M, $((\Sigma', Q), (\Sigma, P)) \in M$, if

(C1) $\quad \Sigma \models \Sigma'$,

(C2) $\quad \Sigma \models h$ implies $\Sigma' \models h,$ for all h expressible with symbols in $Q \subseteq P$.

Σ' is then said to be a **marginal** of Σ with respect to Q. This means that Σ' is as informative as Σ when it comes to decide whether a formula $h \in \mathcal{L}_Q$ is a consequence of Σ or not. Note that the marginal of a propositional information Σ with respect to some Q is not unique. However, if Σ' and Σ'' are two marginals of Σ with respect to Q , then it is evident that they are (logically) equivalent, that is, $\Sigma' \models \Sigma''$ and $\Sigma'' \models \Sigma'$.

The fundamental problem which will be considered in this section is then the **marginalization problem**: given a propositional information (Σ, P) and a subset $Q \subseteq P$, compute a marginal (Σ', Q) of (Σ, P) such that $((\Sigma', Q), (\Sigma, P)) \in M$.

In order to solve this problem we use the Davis–Putnam procedure to eliminate the propositional symbols [7, 8]. However, note that our goal is different: Davis and Putnam were concerned with satisfiability, whereas we are interested in marginalization. So, even though the basic operation is the same, its overall organization and use in the computations will be different. Dechter and Rish [11] also highlight this role of the Davis–Putnam resolution procedure as a compilation algorithm. However, they do not achieve the full capabilities of this approach that will be obtained in this paper through the use of the join tree structure. Order the elements of P in an arbitrary way, such that $P = \{p_1, p_2, \ldots, p_n\}$. The goal is then to eliminate the symbol p_1, that is to marginalize (Σ, P) to $Q = P - \{p_1\}$. For the following we suppose that all formulae of Σ are **clauses** (for a more general case see Section 2.5). Essentially, Σ is then a conjunctive normal form. It is well known, that any Σ can be transformed into such an equivalent normal form, if necessary.

As a preparation define the sets

$$\Sigma_+ = \{\xi_i \in \Sigma : p_1 \in \xi_i\}, \qquad (2.1)$$

$$\Sigma_- = \{\xi_i \in \Sigma : {\sim}p_1 \in \xi_i\}. \qquad (2.2)$$

One or both of these sets may be empty. If ξ_i is a clause containing p_1 and ξ_j a clause containing ${\sim}p_1$, then the resolvent $\rho(\xi_i, \xi_j)$ can be formed by concatenating ξ_i and ξ_j, eliminating p_1 and ${\sim}p_1$, as well as all multiple occurrences of literals. $\rho(\xi_i, \xi_j)$ is set equal to $\top$ (tautology) if ξ_i and ξ_j contain another literal *and* its negation beside p_1 and ${\sim}p_1$. The procedure goes then as follows:

(1) If both Σ_+ and Σ_- are empty, that is, if the literal p_1 does not occur in the formulae of Σ, then put $(\Sigma', P - \{p_1\}) = (\Sigma, P - \{p_1\})$.

(2) If Σ_- is empty, but Σ_+ not, then put $(\Sigma', P - \{p_1\}) = (\Sigma - \Sigma_+, P - \{p_1\})$, that is eliminate all clauses in Σ containing p_1.

(3) Similarly, if Σ_+ is empty, but Σ_- not, then put $(\Sigma', P - \{p_1\}) = (\Sigma - \Sigma_-, P - \{p_1\})$, that is eliminate all clauses in Σ containing ${\sim}p_1$.

(4) Finally, if neither Σ_+ nor Σ_- is empty, then put

$$(\Sigma', P-\{p_1\}) = (\Sigma \cup \{\rho(\xi_i, \xi_j) :\ \xi_i \in \Sigma_+,\ \xi_j \in \Sigma_-\} - (\Sigma_+ \cup \Sigma_-)), P-\{p_1\}). \quad (2.3)$$

Here all the clauses containing either p_1 or $\sim p_1$ are removed from Σ, but all the clauses obtained by resolving these clauses with respect to p_1 are added.

Clearly, this procedure eliminates the symbol p_1 from Σ, it no longer occurs in Σ'. And the following theorem tells us that the resulting propositional information is indeed a marginal of (Σ, P) with respect to $P - \{p_1\}$.

THEOREM 2.2
If $(\Sigma', P - \{p_1\})$ is defined as in the procedure above, then

$$((\Sigma', P - \{p_1\}), (\Sigma, P)) \in M. \quad (2.4)$$

(Proofs of theorems are to be found in the Appendix.)

This indicates that the marginalization problem could be solved by eliminating the symbols in the set $P - Q$ sequentially from Σ. However, in order that this is indeed a way to solve the marginalization problem, it must be verified that if p_1 and then p_2 are sequentially eliminated, that this gives a marginal to $P - \{p_1, p_2\}$; or, more generally, that a sequential marginalization first to $Q' \supseteq Q$ and then to Q is also a marginalization directly to Q. This is what the following theorem affirms.

THEOREM 2.3
If $Q' \supseteq Q''$, and $((\Sigma', Q'), (\Sigma, P)) \in M$ and $((\Sigma'', Q''), (\Sigma', Q')) \in M$, then also $((\Sigma'', Q''), (\Sigma, P)) \in M$.

According to this theorem and Theorem 2.2 a marginal of (Σ, P) with respect to Q can be computed by eliminating the symbols in $P - Q$ from Σ in any sequence. This is essentially a sequence of resolutions which solves the marginalization problem. Note that the resolutions $\rho(\xi_i, \xi_j)$ may introduce redundant clauses. A clause subsuming (containing) another clause is redundant and may be eliminated. If Σ is a set of clauses, then $\mu\Sigma$ denotes the subset of clauses of Σ which are not subsuming another clause of Σ. The tautology $\top$ is assumed to subsume any other clause, hence $\top$ appears never in $\mu\Sigma$, except in $\Sigma = \{\top\}$. Point (4) of the procedure above can then be changed into

$$\begin{aligned}&(\Sigma', P - \{p_1\}) = \\ &\qquad (\mu(\Sigma \cup \{\rho(\xi_i, \xi_j) : \xi_i \in \Sigma_+, \xi_j \in \Sigma_-\} - (\Sigma_+ \cup \Sigma_-)), P - \{p_1\}) \,. \end{aligned} \quad (2.5)$$

There exist even more involved methods to reduce the size of Σ' (see for example [6]). In the next subsection this procedure will be studied in more detail and it will be shown that it permits one to solve **several** marginalization problems at the same time with little additional effort.

Before discussing this issue, let us illustrate the procedure with the simple example introduced above.

EXAMPLE 2.4
Consider the propositional knowledge $(\Sigma, c(\Sigma))$, where Σ contains the clauses ξ_1 to ξ_{20} introduced in Example 2.1. Suppose we want to eliminate the symbols w, g, d, c in this order. To eliminate w we consider the clauses

$$\xi_6 : \ {\sim}a \vee w, \qquad \xi_8 : \ a \vee a_3 \vee {\sim}w,$$
$$\xi_7 : \ {\sim}a_3 \vee w, \qquad \xi_{18} : \ w.$$

With respect to the symbol w the set Σ_+ is formed by the clauses ξ_6, ξ_7 and ξ_{18}, whereas the set Σ_- contains only ξ_8. All these clauses are removed from Σ. Added are all the resolvents of these clauses with respect to w. There is actually only one, which is not a tautology:

$$\rho(\xi_8, \xi_{18}) = a \vee a_3.$$

Next, to eliminate g , the clauses

$$\xi_9 : \ {\sim}a \vee a_4 \vee g, \qquad \xi_{11} : \ a \vee {\sim}g,$$
$$\xi_{10} : \ {\sim}a_4 \vee {\sim}g, \qquad \xi_{19} : \ {\sim}g,$$

have to be removed. Again only one resolvent is to be added:

$$\rho(\xi_9, \xi_{19}) = {\sim}a \vee a_4.$$

The symbol d is eliminated by removing the clauses ξ_{12} to ξ_{14}. No resolution can be added. Finally, to eliminate the symbol c, the clauses ξ_{15} to ξ_{17} and ξ_{20} are to be removed and the resolvent

$$\rho(\xi_{15}, \xi_{20}) = {\sim}e \vee {\sim}a_6$$

is to be added. The resulting marginal to the remaining symbols $a, b, e, a_1, \ldots, a_6$ is thus:

$$\begin{aligned} \xi_1 &: \ {\sim}b \vee {\sim}a_1 \vee a, & \xi_5 &: \ a_1 \vee a_2 \vee {\sim}a, \\ \xi_2 &: \ {\sim}e \vee {\sim}a_1 \vee a, & \rho(\xi_8, \xi_{18}) &: \ a \vee a_3, \\ \xi_3 &: \ {\sim}a_2 \vee a, & \rho(\xi_9, \xi_{19}) &: \ {\sim}a \vee a_4, \\ \xi_4 &: \ b \vee e \vee a_2 \vee {\sim}a, & \rho(\xi_{15}, \xi_{20}) &: \ {\sim}e \vee {\sim}a_6. \end{aligned}$$

Suppose now, in a second phase, we want to marginalize to $a, g, a_1, \ldots, a_6$. Clearly, if the intermediate results of the previous computation were stored, then it is not necessary to start from scratch for this second marginalization. For example, if we chose an elimination sequence w, c, b, e, then at least the elimination of the first symbol w is exactly as before. Although in the former elimination sequence g was eliminated next, which appears not to match the present sequence, we may note that the clauses involved in the elimination of g do in no way interfere with those involved in the elimination of c. Thus, it seems that the previous computations can even be reused for eliminating c.

Such considerations will be discussed more systematically in the following subsection. It will be shown how computations can be organized such that intermediate results can be reused for different marginalizations with a minimum of additional effort.

2.2 The organization of computations

We need to introduce some preliminary notions before discussing the organization of computations of marginals. If (Σ, P) is a propositional information, then we associate to every clause ξ_i of Σ the set $c(\xi_i) \subseteq P$ of the symbols it contains. The family of the sets $c(\xi_i)$

form a hypergraph $\{c(\xi_i), \xi_i \in \Sigma\}$. The sets $c(\xi_i)$ are called its hyperedges. The notion of join (or Markov) tree plays a central role in the following discussion. A join tree is a tree whose nodes are subsets of symbols, such that if a given symbol belongs to two nodes, then it belongs to every node on the unique path between the two nodes. A join tree is said to cover the hypergraph $\{c(\xi_i), \xi_i \in \Sigma\}$ if every hyperedge $c(\xi_i)$ is contained in at least one node of the join tree.

It is well known that any elimination sequence of symbols generates a join tree covering the hypergraph $\{c(\xi_i), \xi_i \in \Sigma\}$ (see for example [1] or [19] for a discussion of these issues). Figure 2 displays a covering join tree for the hypergraph associated with the example of the previous subsection. If we number the nodes of a covering join tree in some way by $i = 1, 2, \ldots$ then let Σ_i be the subset of clauses covered by node i. There may be clauses covered by several nodes of the join tree; such clauses are arbitrarily affected to one of the covering nodes, such that $\cup_i \Sigma_i = \Sigma$ and $\Sigma_i \cap \Sigma_j = \emptyset$ if $i \neq j$.

EXAMPLE 2.5
Again, consider the propositional information of Example 2.1. According to Figure 2 we may take $\Sigma_1 = \{\xi_1, \xi_2, \xi_3, \xi_4, \xi_5\}$, $\Sigma_2 = \{\xi_6, \xi_7, \xi_8, \xi_{18}\}$, $\Sigma_3 = \{\xi_9, \xi_{10}, \xi_{11}, \xi_{19}\}$, $\Sigma_4 = \{\xi_{12}, \xi_{13}, \xi_{14}\}$, and $\Sigma_5 = \{\xi_{15}, \xi_{16}, \xi_{17}, \xi_{20}\}$.

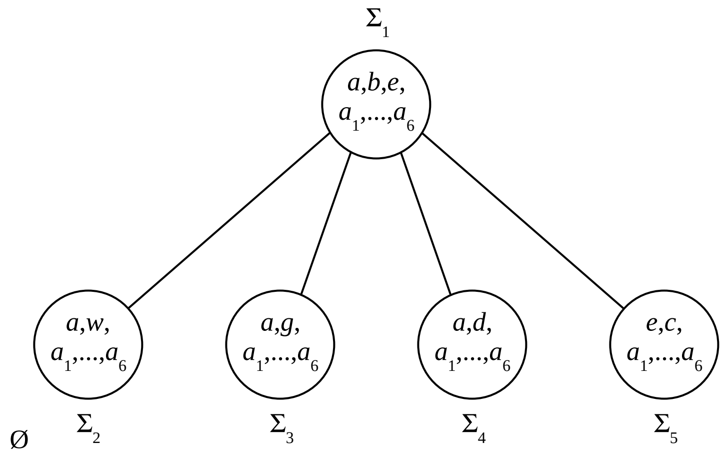

FIGURE 2. A covering join tree for the hypergraph associated with the example of Section 2.1

An important issue in the elimination process is the order in which symbols are removed. Though with different orders the final calculated set of clauses are equivalent, the complexity of the computations can be quite different. The problem of obtaining an optimal sequence in relation to the associated computations (for example that minimizing the number of resolutions) looks a difficult one. It is similar to the problem of obtaining an optimal triangulation of an undirected graph, which is known to be NP-hard (see for example Kjærulff [15]).

However, there are several heuristics which can be used [11]. For example, eliminate in each case the literal for which the product of the number of elements of Σ_+ and the number of elements of Σ_- is minimal. That is, in each case the symbol involving a minimal number of resolutions is removed. This does not guarantee that the total number of resolutions is minimal, but it is better than choosing an arbitrary order.

To each node of the covering join tree a propositional information (Σ_i, P_i) is associated, where P_i is the set of symbols forming node i. This represents then somehow a decomposition of the original propositional information into distinct pieces which together reconstitute the original information. This is a very fruitful point of view which is widespread in fields like Bayesian or belief networks and also in linear equations with sparse matrices, but not so in logics. In order to exploit this way of looking at information we introduce the operation of combination of propositional information.

If (Σ_1, P_1) and (Σ_2, P_2) are two items of propositional information, then they combine naturally into $(\Sigma_1, P_1) \oplus (\Sigma_1, P_1) = (\Sigma_1 \cup \Sigma_2, P_1 \cup P_2)$. Combination is thus union. Note that here too, subsuming clauses can be removed, replacing $\Sigma_1 \cup \Sigma_2$ by $\mu(\Sigma_1 \cup \Sigma_2)$. Clearly, $(\Sigma, c(\Sigma)) = \oplus_{i=1,\ldots,m}(\{\xi_i\}, c(\xi_i))$ and, more importantly,

$$(\Sigma, P) = \bigoplus_i (\Sigma_i, P_i), \tag{2.6}$$

if the (Σ_i, P_i) are the propositional information associated with the nodes of a covering join tree for (Σ, P).

It is from here on convenient to consider classes of logically equivalent propositional information with identical labels. If φ denotes such a class, then $\varphi = (\Sigma, P)$ means that (Σ, P) belongs to the class φ and if $\varphi = (\Sigma', P)$ and $\varphi = (\Sigma'', P)$, then Σ' and Σ'' are logically equivalent. The label of the class, its domain, is denoted by $d(\varphi)$; hence, if $\varphi = (\Sigma, P)$, then $d(\varphi) = P$. The combination carries over to these classes: if $\varphi_1 = (\Sigma_1, P_1)$ and $\varphi_2 = (\Sigma_2, P_2)$, then $\varphi_1 \oplus \varphi_2 = (\mu(\Sigma_1 \cup \Sigma_2), P_1 \cup P_2)$. The operation $\oplus$ is associative and commutative. The propositional information φ form a commutative semigroup with respect to combination. If a propositional information is decomposed according to a covering join tree (like (2.6)), then this defines a corresponding decomposition

$$\varphi = \bigoplus_i \varphi_i. \tag{2.7}$$

If $\varphi = (\Sigma, P)$ is a propositional information, then we denote by $\varphi^{\downarrow Q}$ the equivalence class of its marginals to Q. In the following theorem two fundamental properties of marginalization and combination of propositional information are affirmed.

THEOREM 2.6
(1) If φ is an item of propositional information with $d(\varphi) = P$, and $Q'' \subseteq Q' \subseteq P$, then

$$(\varphi^{\downarrow Q'})^{\downarrow Q''} = \varphi^{\downarrow Q''}. \tag{2.8}$$

(2) If φ_1 and φ_2 are two items of propositional information with $d(\varphi_1) = P$ and $d(\varphi_2) = Q$, then

$$(\varphi_1 \oplus \varphi_2)^{\downarrow P} = \varphi_1 \oplus (\varphi_2^{\downarrow P \cap Q}). \tag{2.9}$$

(2.8) and (2.9) show that propositional information satisfies the axioms introduced by Shenoy and Shafer [29] which permit one in many cases to improve the efficiency of the computations for the solution of the marginalization problem. The important property is (2.9). In our case it means the following (compare also the proof of Theorem 2.6): in both marginals on the left and the right-hand side the symbols in $Q - P$ must be eliminated. This involves in both cases, if $\varphi_1 = (\Sigma_1, P)$ and $\varphi_2 = (\Sigma_2, Q)$, only resolutions with clauses from Σ_2, because the clauses of Σ_1 contain no symbols outside P. However, the search for clauses containing symbols from $Q - P$ is simpler in Σ_2 than in $\Sigma_1 \cup \Sigma_2$. In this sense it is better first to eliminate symbols in the propositional information (Σ_2, Q) and only then to combine with (Σ_1, P), rather than first to combine the two items of propositional information and to eliminate the symbols afterwards. That is, once a decomposition of the propositional information has been organized, it is convenient to use it.

A covering join tree of propositional information $\varphi = (\Sigma, P)$ can now, on the base of properties (2.8) and (2.9), be used to compute marginals $\varphi^{\downarrow Q}$ for any sets Q which are subsets of some node set P_i of the join tree. Computational schemes to do this have been discussed in [29]. These methods can be adapted to the present case, which, in addition to the basic properties (2.8) and (2.9), exhibits a further property, which can be exploited. This is the fact, that the combination of propositional information is clearly **idempotent**, or, more generally, for any $Q \subseteq d(\varphi)$

$$\varphi \oplus (\varphi^{\downarrow Q}) = \varphi. \tag{2.10}$$

One way to organize the computations is to direct the edges of the join tree such that it becomes a rooted tree. This can be done using any construction sequence of the tree. Once this is done, every node i of the tree, except the root node r, has a unique successor $s(i)$. Assume that the nodes are numbered corresponding to the construction sequence, then the root is number 1 and $s(i) < i$ for all $i = 2, \ldots, m$. Define $H_i = \cup_{j=1,\ldots,i} P_j$. It is well known [29], that for a join tree

$$P_i \cap P_{s(i)} = P_i \cap H_{i-1}. \tag{2.11}$$

Theorem 2.6 has then the following corollary [29].

COROLLARY 2.7
Define $\varphi_j^{(m)} = \varphi_j$, for $j = 1, \ldots, m$. If, for $i = m, \ldots, 2$

$$\begin{aligned} \varphi_{s(i)}^{(i-1)} &= (\varphi_i^{(i)})^{\downarrow P_i \cap P_{s(i)}} \oplus \varphi_{s(i)}^{(i)}, && (2.12)\\ \varphi_j^{(i-1)} &= \varphi_j^{(i)}, \quad \text{for } j = 1, \ldots, i-1;\ j \neq s(i), && (2.13) \end{aligned}$$

then, for $i = m-1, \ldots, 1$

$$\begin{aligned} &\varphi^{\downarrow H_i} = \varphi_1^{(i)} \oplus \varphi_2^{(i)} \oplus \cdots \oplus \varphi_i^{(i)}, \\ &\text{and } d(\varphi_j^{(i)}) = P_j. \end{aligned} \tag{2.14}$$

The proof of this theorem will not be given here. It is an immediate consequence of Theorem 2.6 and has been proved by Shafer and Shenoy [29].

From this theorem the following computational scheme on the join tree can be derived: Node m (necessarily a leaf in the tree) computes $(\varphi_m^{(m)})^{\downarrow P_m \cap P_{s(m)}}$ and sends this marginal as a message to its successor node $s(m)$ which combines it with its own stored information $\varphi_{s(m)}^{(m)}$. More generally, for $i = m, \ldots, 2$, we have the following computations:

node p_i: memory $\varphi_i^{(i)}$		**node** $p_{s(i)}$: memory: $\varphi_{s(i)}^{(i)}$
- marginalize $\psi_i = (\varphi_i^{(i)})^{\downarrow P_i \cap P_{s(i)}}$		
- send message ψ_i	$\longrightarrow$	combine $\varphi_{s(i)}^{(i-1)} = \psi_i \oplus \varphi_{s(i)}^{(i)}$

At the end, for $i = 1$, we obtain, according to the theorem,

$$\varphi_1^{(1)} = \varphi^{\downarrow P_1}. \tag{2.15}$$

Note that in this computational scheme, always only propositional information with respect to subsets P_j of P have to be handled. It can be expected that this is much simpler than to work within the whole set P of symbols.

Any node i of the join tree is the root of a subtree $T(i)$ which is itself a join tree. It is then clear that by analogy to (2.15) , if we restrict the consideration above to the subtree $T(i)$ that

$$\varphi_i^{(i)} = \left(\bigoplus_{j \in T(i)} \varphi_j \right)^{\downarrow P_i}. \tag{2.16}$$

This is the first part of a two-phase scheme. It is called the **inward** (or **collect**) phase. The computations at the end of the previous subsection correspond exactly to such an inward propagation in the join tree of Figure 2 is rooted at node 1. This illustrates that the inward phase is nothing other than the elimination of symbols in the sequence which generated the covering join tree.

Note that the idempotency of the combination of propositional information systems has not been used in this phase, that is the general Shafer–Shenoy theory applies just as for example in Bayesian networks. This is different for the second phase, the **outward** (or **distribute**) phase, where marginals of φ with respect to all other P_j, $j = 1, 2, \ldots, m$, are computed. The following theorem makes use of the idempotency of the information calculus.

THEOREM 2.8
Let $\varphi_i^{(i)}$ be the propositional information obtained at step i ($i = m, m-1, \ldots, 2$) of the inward phase. Then

$$\varphi^{\downarrow P_i} = \varphi^{\downarrow P_i \cap P_{s(i)}} \oplus \varphi_i^{(i)}. \tag{2.17}$$

On the basis of this theorem, we may, starting with node 1, for each of its predecessors j (that is, j such that $s(j) = 1$) compute the marginal $\varphi^{\downarrow P_1 \cap P_j}$ and send this marginal as a message to node j, where it is combined with $\varphi_j^{(j)}$ to get a marginal $\varphi^{\downarrow P_j}$. More generally, for $i = 2, \ldots, m$ we compute according to the following scheme:

<u>**node** $p_{s(i)}$</u>: memory $\varphi^{\downarrow P_{s(i)}}$		<u>**node** p_i</u>: memory $\varphi_i^{(i)}$
- marginalize $\psi_i = \left(\varphi^{\downarrow P_{s(i)}}\right)^{\downarrow P_i \cap P_{s(i)}}$		
- send message ψ_i	$\longrightarrow$	combine $\varphi^{\downarrow P_i} = \psi_i \oplus \varphi_i^{(i)}$

In this computational scheme again only propositional information with respect to subsets P_j of P have to be handled. Thus the computational simplification of the first phases is maintained in this second phase.

In this outward phase we compute thus indeed, as proposed, a marginal of (Σ, P) to every set P_j of the join tree. Now, finally if we look for a marginal with respect to some subset $Q \subseteq P_j$, then according to Theorem 2.6 (1), all we need to do is to eliminate the symbols in $P_j - Q$ from $(\Sigma, P)^{\downarrow P_j}$.

Here we have thus a computational organization which caches intermediate results in order to efficiently obtain the marginals to a multitude of subsets Q.

The calculation of messages in this distribution phase is different from the case of Bayesian networks [29]. In that case, the message ψ_i contains the value

$$\left(\bigoplus_{j\notin T(i)} \varphi_j\right)^{\downarrow P_i\cap P_{s(i)}} . \tag{2.18}$$

Here, since the idempotency property is verified, the final result can be obtained if the message ψ_i contains

$$\left(\bigoplus_{j\in\{1,\ldots,m\}} \varphi_j\right)^{\downarrow P_i\cap P_{s(i)}} . \tag{2.19}$$

The same type of messages as in the Bayesian networks is possible here, but the messages we propose are not only simpler to express, but also easier to calculate. The reason is as follows: our message is the Bayesian message combined with the message of the collect phase. Let us call the message of the collect phase ψ_i^o. This information was already calculated and defined on symbols $P_i \cap P_{s(i)}$. Our message is the Bayesian message combined with this message and marginalized on $P_i \cap P_{s(i)}$. That is,

$$\left(\left(\bigoplus_{j\notin T(i)} \varphi_j\right) \oplus \psi_i^o\right)^{\downarrow P_i\cap P_{s(i)}} . \tag{2.20}$$

This computation is a particular case of a calculation of $(\varphi \oplus \varphi')^{\downarrow P'}$, where $d(\varphi') = P' \subseteq P = d(\varphi)$. In general, this computation is easier than the computation of $\varphi^{\downarrow P'}$. The reason is as follows. Assume $\varphi = (\Sigma, P)$, $\varphi' = (\Sigma', P')$, and $\varphi^* = (\Sigma^*, P)$, where Σ^* is the set of clauses in Σ not subsumed by a clause in Σ' (i.e. $\Sigma^* = \mu(\Sigma \cup \Sigma') - \Sigma'$). Under these conditions:

$$(\varphi \oplus \varphi')^{\downarrow P'} = (\varphi^* \oplus \varphi')^{\downarrow P'} = (\varphi^*)^{\downarrow P'} \oplus \varphi'. \tag{2.21}$$

That is, in this case we have to carry out a marginalization but of a reduced set of clauses Σ^*.

EXAMPLE 2.9

Consider again the example of the previous subsection. As already said, the computation corresponds to the inward propagation, if the join tree, Figure 2, is rooted at node 1. The outward propagation is then as follows: for the message to node 2, the symbols b and e must be eliminated from (Σ'_r, P_r), that is from

$$\begin{aligned} \xi_1 &: \ {\sim}b \vee {\sim}a_1 \vee a, & \xi_5 &: \ a_1 \vee a_2 \vee {\sim}a,\\ \xi_2 &: \ {\sim}e \vee {\sim}a_1 \vee a, & \rho(\xi_8,\xi_{18}) &: \ a \vee a_3,\\ \xi_3 &: \ {\sim}a_2 \vee a, & \rho(\xi_9,\xi_{19}) &: \ {\sim}a \vee a_4,\\ \xi_4 &: \ b \vee e \vee a_2 \vee {\sim}a, & \rho(\xi_{15},\xi_{20}) &: \ {\sim}e \vee {\sim}a_6. \end{aligned}$$

This gives the message

$$\begin{aligned} \xi_3 &: \ {\sim}a_2 \vee a, & \rho(\xi_8,\xi_{18}) &: \ a \vee a_3,\\ \xi_5 &: \ a_1 \vee a_2 \vee {\sim}a, & (\xi_9,\xi_{19}) &: \ {\sim}a \vee a_4. \end{aligned}$$

The same message will also be sent to nodes 3 and 4. For the message to node 5, the symbols a and b must be eliminated from (Σ'_r, P_r) in a similar way. The resulting messages are then combined with the cached information on each node and we finally get the following results on the nodes 2 to 5:

$$\begin{aligned}
&Node_2: \{\sim a_2 \vee a,\ a \vee a_3,\ \sim a \vee a_4,\ a_1 \vee a_2 \vee \sim a,\ w\},\\
&Node_3: \{\sim a_2 \vee a,\ a \vee a_3,\ \sim a \vee a_4,\ a_1 \vee a_2 \vee \sim a,\ \sim g\},\\
&Node_4: \{\sim a_2 \vee a,\ a \vee a_3,\ \sim a \vee a_4,\ a_1 \vee a_2 \vee \sim a,\ \sim a \vee \sim a_5 \vee d,\ a_5 \vee \sim d,\ a \vee \sim d\},\\
&Node_5: \{\sim e \vee \sim a_6,\ \sim e \vee \sim a_1 \vee a_4,\ \sim a_2 \vee a_4,\ a_3 \vee a_4,\ a_1 \vee a_2 \vee a_3,\ \sim c\}.
\end{aligned}$$

2.3 *Incremental procedure*

If some new propositional information is added in a node of the join tree, then an identical distribute phase can be used to update the marginals with respect to every node P_j. Suppose that a new propositional information $\varphi'_1 = (\Sigma'_1, P_1)$ is added to node 1, then according to Theorem 2.6 (2), if $\varphi' = \varphi \oplus \varphi'_1$ is the new, updated global information,

$$\varphi'^{\downarrow P_1} = (\varphi \oplus \varphi'_1)^{\downarrow P_1} = \varphi^{\downarrow P_1} \oplus \varphi'_1. \tag{2.22}$$

Thus, the updating on node 1 is readily made. The next theorem shows how the marginals $\varphi'^{\downarrow P_j}$ with respect to the other nodes of the join tree can be computed.

THEOREM 2.10
If $\varphi' = \varphi \oplus \varphi'_1$, with $d(\varphi'_1) \subseteq P_1$, then

$$\varphi'^{\downarrow P_i} = \varphi'^{\downarrow P_i \cap P_{s(i)}} \oplus \varphi^{\downarrow P_i}. \tag{2.23}$$

This theorem shows in fact, that the new information can be propagated outwards using the old marginals stored in the nodes of the join tree and using exactly the outward phase computational scheme. If the new information is added to a node i different from node 1, then there is always a constructing sequence starting with node i. Hence the updating can be done in the same way starting with this node.

In general, we can think of an incremental procedure in which the pieces of information are incorporated and propagated in subsequent stages. Assume that we have a join tree and several pieces of information $\{\varphi_1, \ldots, \varphi_m\}$ such that for each piece of information φ_i there is a node P_j of the tree with $d(\varphi_i) \subseteq P_j$.

For the incremental algorithm we assume that there is an information φ'_j stored in each node P_j and that there is a message $\psi_{i,j}$ stored for each pair of connected nodes P_i and P_j. This message will be used to store messages from P_i to P_j and vice versa. Sending a message from P_i to P_j means to calculate $\psi_{i,j} = \varphi'_i{}^{\downarrow P_i \cap P_j}$ and change φ'_j to $\varphi'_j \oplus \psi_{i,j}$.

The algorithm starts with an initial state in which the information stored in the nodes and the messages are vacuous: $\varphi'_j = (\emptyset, P_j)$ and $\psi_{i,j} = (\emptyset, P_i \cap P_j)$. Then, the pieces of information $\{\varphi_1, \ldots, \varphi_m\}$ are added one by one. For each one of them, a node P_j is selected such that $d(\varphi_i) \subseteq P_j$. Then, φ_i is integrated into φ'_j by changing it to $\varphi'_j \oplus \varphi_i$, and the corresponding messages are sent out from this node to the rest of the network.

Each time a piece of information is integrated into the network, the information in the nodes φ'_j and the messages $\psi_{i,j}$ become more informative (more formulae can be deduced from them). The advantage of organizing the computations this way is the following: originally only very few clauses are in the network and the messages contain a small number of clauses. When new information φ_i is added into the network, i.e. each time a message is sent from P_j to P_k, then messages have already been sent from P_k to P_j. These messages are defined on $P_j \cap P_k$ and by the arguments given in the previous section, this simplifies the computation by reducing the number of clauses to be marginalized.

The disadvantage is that messages have to be recalculated several times. This problem is not very important if some procedure is determined by which the same resolution is never repeated. This can be done in the following way: imagine that node P_k has received a message from node P_j and a message from this node to a different adjacent node P_ℓ has to be computed. Assume that the marginalization to $P_k \cap P_\ell$ requires the deletion of the propositional symbol p_1 and consider the following sets:

- Σ_+^{old} the set of positive p_1 clauses stored in φ'_k before receiving the message,
- Σ_-^{old} the set of negative p_1 clauses stored in φ'_k before receiving the message,
- Σ_0^{old} the set of clauses not containing p_1 stored in φ'_k before receiving the message,
- Σ_+^{rec} the set of positive p_1 clauses contained in the received message,
- Σ_-^{rec} the set of negative p_1 clauses contained in the received message,
- Σ_0^{rec} the set of clauses not contained in the received message.

From these clauses calculate new clauses as follows:

- $\Sigma_+^{new} = \mu(\Sigma_+^{rec} \cup \Sigma_+^{old} \cup \Sigma_0^{old} \cup \Sigma_0^{rec}) - (\Sigma_+^{old} \cup \Sigma_0^{old} \cup \Sigma_0^{rec})$,
- $\Sigma_-^{new} = \mu(\Sigma_-^{rec} \cup \Sigma_-^{old} \cup \Sigma_0^{old} \cup \Sigma_0^{rec}) - (\Sigma_-^{old} \cup \Sigma_0^{old} \cup \Sigma_0^{rec})$.

That is, remove from the received clauses those clauses which are subsumed by clauses not containing p_1, or by old p_1 clauses. The message from P_k to P_ℓ can then be calculated as

$$\mu(\Sigma_0^{rec} \cup \{\rho(\xi,\xi') : \xi \in \Sigma_+^{new}, \xi' \in (\Sigma_-^{new} \cup \Sigma_-^{old})\} \cup \{\rho(\xi,\xi') : \xi \in \Sigma_+^{old}, \xi' \in \Sigma_-^{new}\}). \tag{2.24}$$

Therefore, it is not necessary to carry out the resolutions of old clauses with old clauses, because this information was been previously sent. Note that the former messages from P_ℓ to P_k, which are incorporated in Σ_0^{old}, are used to reduced the number of clauses for which a resolution has to be carried out.

Another problem of the incremental procedure is that the new information has to be integrated in one of the nodes of the tree. If some piece of information arrives which is defined for a set of propositional symbols which is not included in any of the nodes of the tree, then the structure of the tree has to be recalculated, in order to encompass this new set of clauses. This may happen if the pieces of knowledge arrive in a sequential way and they are not known when we build the join tree.

However, in a concrete situation two kinds of knowledge can be distinguished [12]: the general knowledge known in advance (rules generally involving several variables and representing relationships verified for all the elements of a population) and the facts (representing observations for a particular case and involving generally less variables). Since general knowledge usually involves more variables than the facts, it is more important in the construction of the join tree. On the other hand it represents the more static part of the knowledge and a lot of time can be allocated to the compilation of this knowledge. The compilation involves the construction of a reasonable join tree and the propagation.

Observations change from case to case. The observations are introduced for a particular case in order to produce deductions for some questions of interest. The observations generally involve very few propositional symbols and can be integrated into a tree node. So propagation can be done incrementally starting from the previous propagation of generic knowledge. In this way, the deductions can be speeded up with respect to a global propagation of the general and particular knowledge on a simple propagation algorithm: a part of the resolutions have been done with the generic knowledge only and the messages sent in the first stage can be used to reduce the additional resolutions when observations arrive.

2.4 Implementation and complexity

It is well known that in order to solve sparse linear systems of equations hypertree or join trees are very useful to maintain sparsity of the matrices involved during the solution process [28]. It will be shown here that essentially the same idea applies to marginalization in propositional information systems. In fact, one way to store a set Σ of clauses is to define a table with columns corresponding to the propositional symbols N appearing in Σ and a row for each clause. Appearances of a positive literal in a clause are noted by a '$+$' in the corresponding column, a negative literal by a '$-$', all other columns containing '0'. In practice such a table of dimension $|\Sigma| \times |N|$ associated to a set of clauses Σ often contains many '0'. Then the table is called **sparse** and the organization of the computation as described in the previous subsection is useful.

In fact, in eliminating propositional symbols one would like to keep the newly generated clauses short. This corresponds to the purpose of selecting an elimination sequence of symbols which generates a join tree covering a hypergraph $\{c(\xi_i), \xi_i \in \Sigma\}$ such that the cardinalities of the hyperedges remain as small as possible. This guarantees small clauses during the marginalizations because one never generates clauses beyond the nodes of the join tree.

In other words, such a join tree allows to replace the table of dimension $|\Sigma| \times |N|$ by a set of much smaller tables, each one corresponding to a node of the join tree. The number of columns of each table corresponds to the cardinality of the node in the join tree. This is already a considerable reduction in memory space. All the elimination of propositional symbols can be done within these smaller tables. This reduces the time to search for clauses containing the symbol to be eliminated because of the smaller number of rows of the small table. Given that only clauses containing literals of the symbols of the node of the join tree are generated, this keeps the number of new clauses conceivably small.

The search for an optimal covering join tree is known to be NP-hard. But in practice there exist good heuristics [5, 35, 22, 31, 32, 21, 4, 1, 33] for the construction of such join trees if the underlying table is sparse. So, just as in the case of systems of linear equations, the organization of the computations based on a covering join tree can be very efficient.

2.5 A more general approach: disjunctive normal forms

The procedure for the elimination of propositional symbols considered so far assumes that all formulae in Σ are clauses. Although it is always possible to transform a set of propositional formulae Σ into an equivalent set of clauses, this may not be efficient. Though the introduction of artificial symbols could transform this formulae into clauses without a significant increment in the size of the representation, these artificial symbols should be finally deleted, with then the possibility of increasing the size of the representations. Furthermore, the inclusion of the artificial symbols will make more difficult the determination of an optimal deletion sequence.

Therefore, we consider here elimination of symbols in the more general sets of formulae in disjunctive normal form. That is, each formula $\xi \in \Sigma$ is supposed to be of the form

$$\xi = \psi_1 \vee \psi_2 \vee \cdots \vee \psi_k \tag{2.25}$$

where each ψ_i is a conjunction of literals; that is, $\psi_i = a_1 \wedge a_2 \wedge \ldots \wedge a_h$, and a_i is either a p_i or a $\sim p_i$. It will be assumed that in a conjunction never appears a literal and its negation. If this were the case, the conjunction would be the falsity and could be removed from the formula, yielding an equivalent formula.

First we define the operation of the **deletion** of a symbol p from a formula ξ in such disjunctive normal form: in all conjunctions of ξ which contain p or $\sim p$ this literal will be removed and then all the conjunctions subsuming (containing) other conjunction will be deleted. The result of this operation will be called ξ^{-p}. If one of the conjunction of ξ is p or $\sim p$, then the resulting formula ξ^{-p} is the tautology.

Let ξ_1 and ξ_2 be formulae in disjunctive normal form such that ξ_1 contains p and ξ_2 contains $\sim p$, such that

$$\xi_1 = (p \wedge \psi_1^1) \vee \cdots \vee (p \wedge \psi_1^i) \vee \psi_1^{i+1} \vee \cdots \vee \psi_1^k, \tag{2.26}$$
$$\xi_2 = (\sim p \wedge \psi_2^1) \vee \cdots \vee (\sim p \wedge \psi_2^j) \vee \psi_2^{j+1} \vee \cdots \vee \psi_2^h, \tag{2.27}$$

where the conjunctions $\psi_1^{i+1}, \ldots, \psi_1^k$ do not contain p and $\psi_2^{j+1}, \ldots, \psi_2^h$ do not contain $\sim p$. Then a generalized resolution between ξ_1 and ξ_2 with respect to p can be defined as follows:

$$\rho_p(\xi_1, \xi_2) = \left(\psi_1^{i+1} \vee \cdots \vee \psi_1^k \vee \psi_2^{j+1} \vee \cdots \vee \psi_2^h\right)^{-p}. \tag{2.28}$$

Note that $\sim p$ may be present in the conjunctions $\psi_1^{i+1}, \ldots, \psi_1^k$ or p in the conjunctions $\psi_2^{j+1}, \ldots, \psi_2^h$. That is why it is necessary to delete p in the generalized resolution. Furthermore, note that in general $\rho_p(\xi_1, \xi_2) \neq \rho_p(\xi_2, \xi_1)$.

Consider now propositional information systems (Σ, P), where all formulae ξ of Σ are in disjunctive normal form. The elimination of a propositional symbol p_1 form this system is defined as follows: let Σ_+ be the set of formulae from Σ containing p_1 and Σ_- the set of formulae containing $\sim p_1$. Note that these sets are in general not disjoint. Form then

$$\begin{aligned}&(\Sigma'', P - \{p_1\}) = \\ &\quad (\{\xi^{-p_1} : \xi \in \Sigma\} \cup \{\rho_{p_1}(\xi_1, \xi_2) : \xi_1 \in \Sigma_+, \xi_2 \in \Sigma_-, \xi_1 \neq \xi_2\}, P - \{p_1\}).\end{aligned} \tag{2.29}$$

Clearly, this propositional information system contains no more p_1.

EXAMPLE 2.11
To illustrate and clarify this procedure consider $\Sigma = \{(a \wedge b) \vee (\sim a \wedge c), (a \wedge c) \vee b, (a \wedge d) \vee (\sim a \wedge b)\}$. To eliminate the symbol a, form

$$\begin{aligned}\Sigma_+ &= \{(a \wedge b) \vee (\sim a \wedge c), (a \wedge c) \vee b, (a \wedge d) \vee (\sim a \wedge b)\}, \\ \Sigma_- &= \{(a \wedge b) \vee (\sim a \wedge c), (a \wedge d) \vee (\sim a \wedge b)\}.\end{aligned}$$

Then we have

$$\begin{aligned}\{\xi^{-p_1} : \xi \in \Sigma\} &= \{b \vee c, c \vee b, b \vee d\}, \\ \{\rho_{p_1}(\xi_1, \xi_2) : \xi_1 \in \Sigma_+, \xi_2 \in \Sigma_-, \xi_1 \neq \xi_2\} &= \{c \vee d, b \vee b, b \vee d, b \vee b\}.\end{aligned}$$

Of course, we can simplify here by eliminating identical formulae and replacing of $b \vee b$ by b. Finally, we get thus the following result of eliminating a:

$$(\{b \vee c, b \vee d, c \vee d, b\}, \{b, c, d\}).$$

For this procedure of elimination of a propositional symbol in sets of disjunctive norm forms a result analogous to Theorem 2.2 holds.

THEOREM 2.12
If $(\Sigma'', P - \{p_1\})$ is defined by (2.29), then

$$((\Sigma'', P - \{p_1\}), (\Sigma, P)) \in M. \tag{2.30}$$

Furthermore, Theorem 2.3 and its proof carry over to this more general case. Thus, even in this case marginalization can be done by eliminating the symbols in an arbitrary sequence, just as in Section 2.1.

Combination of two such propositional informations (Σ_1, P_1) and (Σ_2, P_2) is defined as before:

$$(\Sigma_1, P_1) \oplus (\Sigma_2, P_2) = (\Sigma_1 \cup \Sigma_2, P_1 \cup P_2). \tag{2.31}$$

With this, not only Theorem 2.6 (1), but also (2) and its proof are valid in the more general case of disjunctive normal forms. This is to say that the whole computational theory of Section 2.2 applies to this case.

The computation can even be more refined by considering the following form of subsumption: if $\xi_1 = \psi_1^1 \vee \psi_2^1 \vee \cdots \vee \psi_k^1$ and $\xi_2 = \psi_1^2 \vee \psi_2^2 \vee \cdots \vee \psi_h^2$ are two disjunctive normal forms, then ξ_1 is subsumed by ξ_2 if for each conjunction ψ_i^2 in ξ_2 there is a conjunction ψ_j^1 in ξ_1 such that ψ_j^1 is subsumed, that is contained in ψ_i^2. Thus clearly $\xi_2 \models \xi_1$. A disjunctive normal form which is subsumed by another one can therefore be deleted without loss of information. Thus, if $\mu\Sigma$ denotes the set of disjunctive normal forms of Σ which are not subsumed by another form of Σ, then the combination can also be defined as

$$(\Sigma_1, P_1) \oplus (\Sigma_2, P_2) = (\mu(\Sigma_1 \cup \Sigma_2), P_1 \cup P_2). \tag{2.32}$$

And the elimination of a symbol (2.29) can be rewritten as

$$\begin{aligned}&(\Sigma'', P - \{p_1\}) = \\ &\quad (\mu(\{\xi^{-p_1} : \xi \in \Sigma\} \cup \{\rho_{p_1}(\xi_1, \xi_2) : \xi_1 \in \Sigma_+, \xi_2 \in \Sigma_-, \xi_1 \neq \xi_2\}), P - \{p_1\}).\end{aligned} \tag{2.33}$$

This may help to avoid excessive growth of the number of formulae when combining and marginalizing propositional information systems.

3 Assumption-based information systems

In this section we look more into the details of the subject introduced in Subsection 1.2. The idea of assumption-based reasoning is now developed from the point of view of information systems.

3.1 Basic notions

Let (Σ, N) be a propositional information as introduced in the previous section. If a subset of propositions $A \subseteq N$ is declared as **assumptions**, $P = N - A$, then the triple (Σ, A, P) is called **assumption-based information**. The idea behind this view is that assumptions are propositions for which we cannot be sure whether they are true or not. They are used to define uncertain logical relations. If a is an assumption, then $a \wedge b \rightarrow c$, for example, means that $b \rightarrow c$ is an uncertain rule (implication) whose validity depends on whether the assumption a holds or not.

EXAMPLE 3.1
In the example introduced at the beginning of Section 2.1 the propositions a_1 to a_6 can in fact be considered as assumptions. For example, the clause $\xi_1 = \sim b \vee \sim a_1 \vee a$ says that a burglary (b) implies an alarm (a) under the assumption that the alarm system functions properly (a_1); $\xi_2 = \sim a_2 \vee a$ says that under some other circumstances (a_2) the alarm (a) triggers itself without an explicit reason; etc.

The concepts of quasi-support and support can now be adapted for assumption-based information (Σ, A, P). For that purpose, we use again φ to denote the whole class of logically equivalent assumption-based information. $\varphi = (\Sigma, A, P)$ means again that (Σ, A, P) belongs to the class φ. The label of φ is now its domain $d(\varphi) = (A, P)$ of assumptions and other propositions. If h is a hypothesis expressible in $Q \subseteq A \cup P$, then we write $QS_\varphi(h)$ and $SP_\varphi(h)$ instead of $QS(\Sigma, h)$ and $SP(\Sigma, h)$ to denote the corresponding sets of quasi-supports and supports. Then, the main problem is again to compute a subset $QS' \subseteq QS_\varphi(h)$ which is logically equivalent to $QS_\varphi(h)$. One particular solution is the set $\mu QS_\varphi(h)$ of minimal quasi-supports. Different methods are known for this problem (see Subsection 1.2). The method we propose here is based on marginalization.

Let $\sim H$ be a set of clauses representing the negated hypothesis $\sim h$ and $\varphi_h = (\sim H, A, P)$ the corresponding assumption-based information. The following theorem tells us then how to compute the quasi-supports for h.

THEOREM 3.2
Let $\varphi = (\Sigma, A, P)$ and $\varphi_h = (\sim H, A, P)$ be two items of assumption-based information as described above. If $\varphi' = (\varphi \oplus \varphi_h)^{\downarrow A} = (\Sigma', A, \emptyset)$ is a marginal of $\varphi \oplus \varphi_h$ to A, then

$$QS' = \sim \Sigma' \tag{3.1}$$

is a set of quasi-supports of h which is equivalent to $QS_\varphi(h)$.

The resulting set $\sim \Sigma'$ is a set of conjunctions obtained by negating the clauses in Σ'. This theorem describes an alternative way of computing quasi-supports by means of marginalization. The advantage is that the resulting set of arguments is logically equivalent but often considerably smaller than $\mu QS_\varphi(h)$. This is of particular importance when numerical degrees of supports are computed.

Another important theorem describes a second way that marginalization can help to find the quasi-supports.

THEOREM 3.3
Let (Σ, A, P) be an assumption-based information and $Q \subseteq A \cup P$. If $\varphi' = \varphi^{\downarrow A \cup Q}$ is a marginal of φ to $Q \cup A$, then

$$QS_\varphi(h) = QS_{\varphi'}(h) \quad \text{for all } h \in \mathcal{L}_Q. \tag{3.2}$$

This theorems tells us that we may first marginalize φ to $Q \cup A$ and then compute the quasi-supports relative to this reduced information, instead of working with the full original information φ. If there are different hypotheses, all expressible in Q, then the marginal φ' has only to be computed once. In this way, a number of redundant resolutions can be avoided.

3.2 Algebraic structure of assumption-based reasoning

In Section 2 we introduced an algebraic structure of propositional information systems consisting of a commutative semigroup with respect to the combination operation $\oplus$ and of an operation of marginalization. This structure will be adapted here to assumption-based information and then extended also to the domain of quasi-supports. These algebraic structures will clarify the different ways to compute quasi-supports.

The operations of combination and marginalization of propositional systems can without difficulty be extended to assumption-based information. In fact, if $\varphi_i = (\Sigma_i, A_i, P_i)$ for $i = 1, 2$, then

$$\varphi_1 \oplus \varphi_2 = (\Sigma_1 \cup \Sigma_2, A_1 \cup A_2, P_1 \cup P_2). \tag{3.3}$$

Here, as in the sequel we assume that sets of assumptions A_i and sets of propositions P_j are always disjoint, i.e. $A_i \cap P_j = \emptyset$. If φ is the set of all assumption-based information over finite sets A and P, then the operation $\oplus$ provides this set with the structure of **commutative semigroup**. If we define $(A_1, P_1) \cup (A_2, P_2) = (A_1 \cup A_2, P_1 \cup P_2)$ (similarly for the intersection), then the labels satisfy furthermore the relation

$$d(\varphi_1 \oplus \varphi_2) = d(\varphi_1) \cup d(\varphi_2). \tag{3.4}$$

If A' and P' are subsets of A and P respectively, then the marginal of assumption-based information $\varphi = (\Sigma, A, P)$ to (A', P') can also be defined in terms of marginals of the corresponding propositional information (Σ, N). Indeed, if (Σ', N') is a marginal of (Σ, N) with respect to $N' = A' \cup P'$, then we define

$$\varphi^{\downarrow(A',P')} = (\Sigma', A', P'). \tag{3.5}$$

Of course we have

$$d(\varphi^{\downarrow(A',P')}) = (A', P'). \tag{3.6}$$

This process is called the marginalization of assumption-based information. Note that often the assumptions are not reduced, that is $A' = A$, because in general one does not want to eliminate assumptions from the consideration (see for example Theorem 3.3). In certain cases however, in order to simplify, one may be ready to eliminate some assumptions, considered as not so relevant for some questions. This yields, on the level of supports, of course only an approximation of the complete support. Nevertheless it is worthwhile to include this possibility for the sake of generality.

It is evident, that Theorem 2.6 can be adapted to assumption-based information:

(1) If $\varphi = (\Sigma, A, P)$ and $A'' \subseteq A' \subseteq A$, $P'' \subseteq P' \subseteq P$, then

$$(\varphi^{\downarrow(A',P')})^{\downarrow(A'',P'')} = \varphi^{\downarrow(A'',P'')}. \tag{3.7}$$

(2) If $\varphi_1 = (\Sigma_1, A_1, P_1)$ and $\varphi_2 = (\Sigma_2, A_2, P_2)$, then

$$(\varphi_1 \oplus \varphi_2)^{\downarrow(A_1,P_1)} = \varphi_1 \oplus (\varphi_2^{\downarrow(A_1,P_1)\cap(A_2,P_2)}). \tag{3.8}$$

This is nothing new, it is only a way of writing adapted to assumption-based information rather than to propositional information.

To assumption-based information $\varphi = (\Sigma, A, P)$ corresponds a family of up-sets $QS_\varphi(h)$ of quasi-supports, $h \in \mathcal{L}_N$, $N = A \cup P$. QS_φ can be considered as a mapping of $h \in \mathcal{L}_N$ into the family of up-subsets of C_A. This mapping QS_φ has the following basic properties (proved in [16]):

(S1) $QS_\varphi(\top) = C_A$,

(S2) $QS_\varphi(h_1 \wedge h_2) = QS_\varphi(h_1) \cap QS_\varphi(h_2)$.

These properties can also be viewed in a slightly different way. For a set of propositional symbols N let Li_N denote the Lindenbaum algebra of $\mathcal{L}_N$, that is the Boolean algebra of logically equivalent formulae of $\mathcal{L}_N$ (see for example Sikorski [30]). Denote the equivalence class of a formula $h \in \mathcal{L}_N$ by $[h]$. Then a quasi-support set $QS_\varphi(h)$ can also be represented by a uniquely determined element of Li_A, namely

$$qs_\varphi([h]) = \bigvee_{a \in QS_\varphi(h)} [a]. \tag{3.9}$$

qs_φ becomes then a mapping from the Lindenbaum algebras Li_N into Li_A. (S1) and (S2) translate into the equivalent properties

(S1) $qs_\varphi([\top]) = [\top]$,

(S2) $qs_\varphi([h_1] \wedge [h_2]) = qs_\varphi([h_1]) \wedge qs_\varphi([h_2])$.

Such a meet-homomorphism between Boolean algebras is called an **allocation of support** [17].

Now, if $\varphi_1 = (\Sigma_1, A_1, P_1)$ and $\varphi_2 = (\Sigma_2, A_2, P_2)$ are two assumption-based informations, then $\varphi_1 \oplus \varphi_2 = (\Sigma_1 \cup \Sigma_2, A_1 \cup A_2, P_1 \cup P_2)$. To it corresponds an allocation of support from $Li_{N_1 \cup N_2}$ into $Li_{A_1 \cup A_2}$. It can also be obtained from the allocations of support relative to the two original assumption-based informations as follows:

$$qs_{\varphi_1 \oplus \varphi_2}([h]) = \bigvee \{qs_{\varphi_1}([h_1]) \wedge qs_{\varphi_2}([h_2]) : h_1 \in \mathcal{L}_{N_1}, h_2 \in \mathcal{L}_{N_2}, h_1 \wedge h_2 \models h\} \tag{3.10}$$

for every $h \in \mathcal{L}_{N_1 \cup N_2}$ (see [17] for a proof). This defines then a combination operation $\oplus$ on the domain Ψ of allocations of support, such that $qs_{\varphi_1 \oplus \varphi_2} = qs_{\varphi_1} \oplus qs_{\varphi_2}$. It has been shown that Ψ becomes an idempotent, commutative semigroup under the operation $\oplus$ as defined by the right-hand side of (3.10) (see [17]).[2] Furthermore, we define the label of an allocation of support qs from a Lindenbaum algebra Li_N, $N = A \cup P$, to a Lindenbaum algebra Li_A by $d(qs) = (A, P)$, such that, in particular, $d(qs_\varphi) = d(\varphi)$.

If $\varphi = (\Sigma, A, P)$ is assumption-based information, $A' \subseteq A$, $P' \subseteq P$, then to the marginalized assumption-based information $\varphi^{\downarrow(A',P')}$ corresponds again an allocation of support from $Li_{N'}$, with $N' = A' \cup P'$, into $Li_{A'}$. The following theorem is a generalization of Theorem 3.3:

[2] Note that we have not proved that every allocation of support is induced by an assumption-based information.

THEOREM 3.4
If $\varphi = (\Sigma, A, P)$ is assumption-based information, $A' \subseteq A$, $P' \subseteq P$, and $\varphi^{\downarrow(A',P')} = (\Sigma', A', P')$ the marginalized assumption-based information, then, for $h \in \mathcal{L}_{N'}$, $N' = A' \cup P'$,

$$qs_{\varphi^{\downarrow(A',P')}}([h]) = \bigvee\{[a'] : a' \in C_{A'}, [a'] \leq qs_{\varphi}([h])\}. \tag{3.11}$$

Note that, if $A' = A$, then $qs_{\varphi^{\downarrow(A,P')}}([h]) = qs_{\varphi}([h])$ for $h \in \mathcal{L}_{A \cup P'}$. This is the special case of Theorem 3.3.

(3.11) defines a marginalization operation in the domain Ψ of allocations of support, that is

$$qs_{\varphi}{}^{\downarrow(A',P')} = qs_{\varphi^{\downarrow(A',P')}}. \tag{3.12}$$

The marginalization of allocations of support has been defined in this way in [17].

The association of an allocation of support qs_{φ} to any assumption-based information φ defines a mapping m from Φ into Ψ, $m(\varphi) = qs_{\varphi}$. This mapping is in view of (3.10) and Theorem 3.4 a **homomorphism** with respect to combination and marginalization,

$$m(\varphi_1 \oplus \varphi_2) = m(\varphi_1) \oplus m(\varphi_2), \quad m(\varphi^{\downarrow(A',P')}) = (m(\varphi))^{\downarrow(A',P')}. \tag{3.13}$$

Furthermore, the mapping maintains labels,

$$d(m(\varphi)) = d(\varphi). \tag{3.14}$$

Note that this homomorphism m carries properties (1) and (2) of Theorem 2.6 over to allocations of support. Indeed,

(1) If φ is an assumption-based information with $d(\varphi) = (A, P)$ and $(A'', P'') \subseteq (A', P') \subseteq (A, P)$, then by (2.8) and (3.13)

$$\begin{aligned} (m(\varphi)^{\downarrow(A',P')})^{\downarrow(A'',P'')} &= m((\varphi^{\downarrow(A',P')})^{\downarrow(A'',P'')}), \\ &= m(\varphi^{\downarrow(A'',P'')}) = (m(\varphi))^{\downarrow(A'',P'')}. \end{aligned} \tag{3.15}$$

(2) If φ_1 and φ_2 are two assumption-based informations with $d(\varphi_1) = (A', P')$ and $d(\varphi_2) = (A'', P'')$, then by (2.9) and (3.13))

$$\begin{aligned} (m(\varphi_1) \oplus m(\varphi_2))^{\downarrow(A',P')} &= m((\varphi_1 \oplus \varphi_2)^{\downarrow(A',P')}), \\ &= m(\varphi_1 \oplus (\varphi_2)^{\downarrow(A',P') \cap (A'',P'')}), \\ &= m(\varphi_1) \oplus (m(\varphi_2))^{\downarrow(A',P') \cap (A'',P'')}. \end{aligned} \tag{3.16}$$

This important remark allows us to develop two different computational approaches to assumption-based reasoning, both based on local computations in join trees as will be shown in the next subsection.

3.3 Computational structure of assumption-based reasoning

Consider an assumption-based information $\varphi = (\Sigma, A, P)$ and suppose there is a join tree covering this propositional information such that with node i of this tree the assumption-based

$$\begin{tikzcd}
\boxed{\text{Assumption-Based Information}} & \boxed{\text{Allocation of Support}} \\
\varphi = \bigoplus_i \varphi_i \arrow[r, "m"] \arrow[d, "\text{marginalization}"] & m(\varphi) = \bigoplus_i m(\varphi_i) \arrow[d, "\text{marginalization}"] \\
\emptyset \quad \varphi^{\downarrow(A_i,P_i)} \arrow[r, "m"] & m(\varphi^{\downarrow(A_i,P_i)}) = m(\varphi)^{\downarrow(A_i,P_i)}
\end{tikzcd}$$

FIGURE 3. The commutative diagram for computing marginalized allocation of support

information $\varphi_i = (\Sigma_i, A_i, P_i)$ is associated and $\varphi = \oplus_i \varphi_i$. The corresponding allocations of support are $m(\varphi) = m(\oplus_i \varphi_i) = \oplus_i m(\varphi_i)$. The problem is now to compute the marginals $m(\varphi)^{\downarrow(A_i,P_i)}$ of the allocation of support $m(\varphi)$ for the nodes of the join tree.

As the diagram in Figure 3 illustrates, there are two ways to compute these marginals. In the first approach, we compute in a first step the marginal $\varphi^{\downarrow(A_i,P_i)}$ of the assumption-based information by the methods of Section 2.2. In a second step the allocations of support $m(\varphi^{\downarrow(A_i,P_i)})$ of these marginalized assumption-based information are then derived. By (3.13) this equals the marginalized allocations of support $m(\varphi)^{\downarrow(A_i,P_i)}$.

The second approach consists in first computing the allocations of support $m(\varphi_i)$ relative to the assumption-based information φ_i. Then, in the second step, these allocations of support are combined and marginalized, that is

$$m(\varphi)^{\downarrow(A_i,P_i)} = (\oplus_i m(\varphi_i))^{\downarrow(A_i,P_i)} \tag{3.17}$$

is computed. This gives according to (3.13) the same result as the first approach, in other words, the diagram in Figure 3 is **commutative**.

As (3.15) and (3.16) are valid for allocations of support, the combination and marginalization of allocations of support can be similarly organized in the join tree as the combination and the marginalization of assumption-based information. This is illustrated in Figure 4.

If a node i of the join tree, at a given moment of the propagation (in- or outwards) contains assumption-based information $\varphi'_i = (\Sigma'_i, A_i, P_i)$, then the message to a neighbour node j is the assumption-based information $\varphi_i'^{\downarrow(A_i,P_i)\cap(A_j,P_j)}$. In the receiving node j this message will be combined with the assumption-based information already there,

$$\varphi'_j \oplus \varphi_i'^{\downarrow(A'_i,P'_i)\cap(A'_j,P'_j)}. \tag{3.18}$$

Similarly, in the computation with allocations of support, at the same moment, the nodes i of the join tree contain the associated allocations of support $m(\varphi'_i)$ of the assumption-based information φ'_i contained in the nodes in the first approach. The message sent

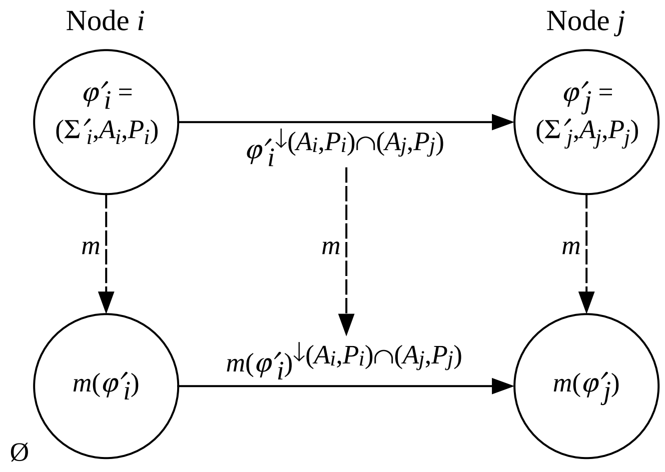
Node i
Node j
$\varphi'_i = (\Sigma'_i, A_i, P_i)$
$\varphi'_j = (\Sigma'_j, A_j, P_j)$
$\varphi'^{\downarrow(A_i,P_i)\cap(A_j,P_j)}_i$
m
m
m
$m(\varphi'_i)$
$m(\varphi'_i)^{\downarrow(A_i,P_i)\cap(A_j,P_j)}$
$m(\varphi'_j)$
Ø

FIGURE 4. Message passing on the join tree in the two formalisms

$m(\varphi'_i)^{\downarrow(A'_i,P'_i)\cap(A'_j,P'_j)} = m(\varphi'^{\downarrow(A'_i,P'_i)\cap(A'_j,P'_j)}_i)$ and the combination in the receiving node j correspond also to the associated allocations of support

$$m(\varphi'_j) \oplus m(\varphi'_i)^{\downarrow(A_i,P_i)\cap(A_j,P_j)} = m(\varphi'_j \oplus \varphi'^{\downarrow(A_i,P_i)\cap(A_j,P_j)}_i). \tag{3.19}$$

The details of the computations with allocations of support in this second approach are described in [16].

In both approaches, there is the need to pass from assumption-based information $\varphi = (\Sigma, A, P)$ to the corresponding allocation of support $m(\varphi)$. In the first approach this is done at the end, in the second one at the beginning. As noted in Section 3.1 this passage to the allocation of support is essentially the problem underlying ATMS. Different methods for its solution have been described in [18, 19]. It is not evident which one of the two approaches above is computational more efficient. The general computational scheme discussed in this paper for both propositional and assumption-based information systems is an instance of a general theory of information systems. It displays the general issues of combination and marginalization of information. This generic point of view permits to introduce new distributed architectures into classical techniques such as propositional logic.

The methods presented in this paper can be applied to different computational problems related with propositional information. An important example is the problem of finding arguments for hypotheses in the domain of assumption-based reasoning. Practical experimentation and a comparison with other existing approaches will be necessary in the future.

Acknowledgements

Research supported by grant No.2100–042927.95 of the Swiss National Foundation for Research and by the Commission of the European Communities under ESPRIT III BRA 6165, DRUMS 2.

References

[1] R. Almond and A. Kong. Optimality issues in constructing a markov tree from graphical models. Research Report A3, Department of Statistics, Harvard University, November 1991.

[2] B. Anrig, R. Haenni, J. Kohlas and N. Lehmann. Assumptionbased modeling using ABEL. In *Symbolic and Quantitative Approaches to Uncertainty, European Conference ECSQARU'97'*, Bonn, 1997.

[3] B. Anrig, R. Haenni and N. Lehmann. ABEL a new language for assumptionbased evidential reasoning under uncertainty. Technical Report 9701, University of Fribourg, Institute of Informatics, 1997.

[4] S. Arnborg, D. Corneil and A. Proskurowski. Complexity of finding embedings in a ktree. *SIAM Journal of Algebraic and Discrete Methods*, **38**, 277–284, 1987.

[5] U. Bertele and F. Brioschi. *Nonserial Dynamic Programming*. Academic Press, 1972.

[6] G. Birkhoff and T.C. Bartee. *Modern Applied Algebra.* McGraw Hill, New York, 1970.

[7] M. Davis and H. Putnam. A computing procedure for quantification theory. *Journal of the ACM*, **5**, 394–397, 1962.

[8] M. Davis and H. Putnam. A computing procedure for quantification theory. In *Automation of Reasoning 1: Classical Papers on Computational Logic 1957/1966*, J. Siekmann and G. Wrightson, eds. pp. 125–139. Springer, Berlin, Heidelberg, 1983.

[9] J. de Kleer. An assumptionbased TMS. *Artificial Intelligence*, **28**, 127–162, 1986.

[10] R. Dechter and A. Dechter. Structuredriven algorithms for truth maintenance. *Artificial Intelligence*, **82**, 120, 1996.

[11] R. Dechter and I. Rish. Directional resolution: The Davis–Putnam procedure revisited. In *Principles of Knowledge Representation and Reasoning*, pp. 134–145, 1994.

[12] D. Dubois and H. Prade. A survey of belief revision and updating rules in various uncertainty models. *International Journal of General Systems*, **9**, 61–100, 1994.

[13] J. N. Hooker. Logical inference and polyhedral projection. In *Computer Science Logic*, volume 626 of *Lecture Notes in Computer Science*, E. Börger, G. Jäger, H. Kleine-Büning and M. M. Richter, eds. pp. 184–200. Springer Verlag, Berlin, Heidelberg, New York, 1992.

[14] K. Inoue. Linear resolution for consequence finding. *Artificial Intelligence*, **56**, 301–353, 1992.

[15] U. Kjærulff. Triangulation of graphsalgorithms giving small total space. Technical Report R9009, Department of Mathematics and Computer Science, Aalborg University, Denmark, 1990.

[16] J. Kohlas. Symbolic evidence, arguments, supports and valuation networks. In *Symbolic and Quantitative Aproaches to Reasoning and Uncertainty*, M. Kruse M. Clarke and S. Moral, eds. pp. 186–198. Springer, 1993.

[17] J. Kohlas. Mathematical foundations of evidence theory. In *Mathematical Models for Handling Partial Knowledge in Artificial Intelligence*, G. Coletti, D. Dubois, and R. Scozzafava, eds. pp. 31–64. Plenum Press, 1995.

[18] J. Kohlas and P. A. Monney. Probabilistic assumption-based reasoning. In *Proceedings of the 9th Conference on Uncertainty in Artificial Intellitence* Heckermann and Mamdani, eds. Morgan Kaufmann, 1993.

[19] J. Kohlas and P. A. Monney. *A Mathematical Theory of Hints. An Approach to the DempsterShafer Theory of Evidence*, volume 425 of *Lecture Notes in Economics and Mathematical Systems.* Springer, 1995.

[20] J. Kohlas and R. F. Stärk. Information algebras and information systems. Technical Report 9614, University of Fribourg, Institute of Informatics, 1996.

[21] A. Kong. *Multivariate Belief Functions and Graphical Models*. PhD thesis, Department of Statistics, Harvard University, 1986.

[22] K. Lange and M. Boehnke. Extensions to pedigree analysis: Optimal calculations of mendelian likelihoods. *Human Heredity*, **33**, 291–301, 1983.

[23] K. B. Laskey and P. E. Lehner. Assumptions, beliefs and probabilities. *Artificial Intelligence*, **41**, 65–77, 1989.

[24] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of Royal Statistical Society*, **50**, 157–224, 1988.

[25] J. Pearl. *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufmann, 1988.

[26] G. M. Provan. A logic based analysis of Dempster–Shafer theory. *International Journal of Approximate Reasoning*, **4**, 451–495, 1990.

[27] R. Reiter and J. de Kleer. Foundations of assumption based truth maintenance systems. *Proceedings of the American Association in AI*, pp. 183–188, 1987.

[28] D. J. Rose. Triangulated graphs and the elimination process. *Journal of Mathematical Analysis and Applications*, **32**, 597–609, 1970.

[29] P. P. Shenoy and G. Shafer. Axioms for probability and belief function propagation. In *Uncertainty in Artificial Intelligence 4*, R. D. Shachter *et al.*, eds. pp. 169–198. North Holland, 1990.
[30] R. Sikorski. *Boolean Algebras*. Springer, Berlin, 1960.
[31] R. E. Tarjan and M. Yannakakis. Simple linear time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal of Computing*, **13**, 566–579, 1984.
[32] A. Thomas. Optimal computation of probability functions for Pedigree Analysis. *J. Math. Appl. Med. and Bio.*, **3**, 167–178, 1986.
[33] W. X. Wen. Optimal decomposition of belief networks. In *Uncertainty in Artificial Intelligence*. P. Bonissone, ed. North Holland, 1991.
[34] H. P. Williams. FourierMotzkin elimination exension to integer programming problems. *Journal of Combinatorial Theory*, **21**, 118–123, 1976.
[35] M. Yannakakis. Computing the minimum fillin is NPcomplete. *SIAM Journal of Algebraic and Discrete Methods*, **2**, 77–79, 1981.

Appendix

A Proof of Theorems

A.1 Proof of Theorem 2.2

In all cases (1) to (4) of the procedure $c(\Sigma) \subseteq P$ implies $c(\Sigma') \subseteq P - \{p_1\}$, s.t. $P - \{p_1\}$ is a label of Σ'.

Let $x \in \{0,1\}^{|P|}$ and x_+ the Boolean vector x with $x_1 = 1$, x_- the Boolean vector with $x_1 = 0$ s.t. $x_+ \models p_1$, $x_- \models {\sim}p_1$, and either $x = x_+$ or $x = x_-$.

(1) In case (1) of the procedure condition (1) and (2) of the marginalization relation are trivially fulfilled.

(2) In case (2) clearly $\Sigma \models \Sigma' \subseteq \Sigma$. Now, if $x \models \Sigma'$ then $x_+ \models \Sigma$, but then $x_+ \models h$ and $x_- \models h$ (because $h \in \mathcal{L}_{P-\{p_1\}}$), hence $x \models h$ and thus $\Sigma' \models h$.

(3) In case (3) a similar argument to that in case (2) holds.

(4) $\Sigma \models \rho(\xi_i, \xi_j)$ for $\xi_i, \xi_j \in \Sigma$. This implies $\Sigma \models \Sigma'$.

Let $x \models \Sigma'$. Then $x \models \{\rho(\xi_i, \xi_j) : \xi_i \in \Sigma_+, \xi_j \in \Sigma_-\}$. If $x = x_+$, then clearly $x \models \{\xi_i \in \Sigma_+\}$. There are two possible cases:

If $\xi_i - p_1$ are the clauses ξ_i where p_1 is eliminated then, either

(i) $x \models \{\xi_i - p_1 : \xi_i \in \Sigma_+\}$. But in this case $x_- \models \{\xi_i \in \Sigma_+\}$ and $x_- \models \{\xi_j \in \Sigma_-\}$ and hence finally $x_- \models \Sigma$.

(ii) otherwise there is a $\xi_i \in \Sigma_+$ s.t. $x \not\models \xi_i - p_1$, but then $x \models \rho(\xi_i, \xi_j)\ \forall \xi_j \in \Sigma_-$ implies $x \models \{\xi_j \in \Sigma_-\}$. Hence $x = x_+ \models \Sigma$.

Similarly, if $x = x_-$ either $x = x_- \models \Sigma$ or $x_+ \models \Sigma$.

Thus, $x \models \Sigma'$ implies always $x_+ \models \Sigma$ or $x_- \models \Sigma$. But in both cases x_+ and $x_- \models h \in \mathcal{L}_{P-\{p\}}$ if $\Sigma \models h$, hence $x \models h$ and therefore finally $\Sigma' \models h$. □

A.2 Proof of Theorem 2.3

We have

(1) $\Sigma \models \Sigma' \models \Sigma''$, hence $\Sigma \models \Sigma''$,

(2) Suppose $\Sigma \models h \in \mathcal{L}_{Q''} \subseteq \mathcal{L}_{Q'}$. Thus $\Sigma' \models h$, hence $\Sigma'' \models h$.

This shows that $((\Sigma'', Q''), (\Sigma, P)) \in M$. □

A.3 Proof of Theorem 2.6

(1) follows from Theorem 2.3. (2) If Q is a set of symbols, then let $\sigma(Q)$ denote an ordered sequence of these elements. Furthermore for a set Σ of clauses, let $\Sigma^{-\sigma(Q)}$ denote the marginal obtained from Σ by eliminating the symbols of Q in the sequence $\sigma(Q)$. If Σ_{+Q} denotes a subset of clauses of Σ containing symbols of Q and Σ_{-Q} a subset of clauses of Σ containing no symbols of Q, and $\Sigma = \Sigma_{+Q} \cup \Sigma_{-Q}$, then clearly

$$\Sigma^{-\sigma(Q)} = \Sigma_{+Q}^{-\sigma(Q)} \cup \Sigma_{-Q}.$$

Now, let $\varphi_1 = (\Sigma_1, P)$, $\varphi_2 = (\Sigma_2, P)$. Thus we have

$$\varphi_1 \oplus \varphi_2 = (\Sigma_1 \cup \Sigma_2, P \cup Q)$$

and in order to marginalize this information to P we have to remove the symbols in $Q - P$. But Σ_1 contains no such symbols. Hence, it follows that

$$(\varphi_1 \oplus \varphi_2)^{\downarrow P} = \left(\Sigma_1 \cup \Sigma_2^{-\sigma(Q-P)}, P\right) .$$

On the other hand

$$\varphi_2^{\downarrow P \cap Q} = \left(\Sigma_2^{-\sigma(Q-P\cap Q)}, P \cap Q\right) .$$

Note that $Q - P \cap Q = Q - P$, hence

$$\varphi_1 \oplus \varphi_2^{\downarrow P \cap Q} = \left(\Sigma_1 \cup \Sigma_2^{-\sigma(Q-P)}, P\right) = (\varphi_1 \oplus \varphi_2)^{\downarrow P}$$

which proves the theorem. □

A.4 Proof of Theorem 2.8

Let

$$\begin{aligned} \psi_1 &= \varphi_1^{(i)} \oplus \varphi_2^{(i)} \oplus \cdots \oplus \varphi_{i-1}^{(i)} \\ \psi_2 &= \varphi_i^{(i)} \end{aligned}$$

such that $d(\psi_1) = H_{i-1}$, $d(\psi_2) = P_i$ (Theorem 2.7). Also by Theorem 2.7 we have then

$$\varphi^{\downarrow H_i} = \psi_1 \oplus \psi_2 .$$

It follows now from the joint-tree property (4) $P_i \cap P_{s(i)} = P_i \cap H_{i-1}$ that

$$(\psi_1 \oplus \psi_2)^{\downarrow P_i \cap P_{s(i)}} \oplus \psi_2 = (\psi_1 \oplus \psi_2)^{\downarrow P_i \cap H_{i-1}} \oplus \psi_2 .$$

By Theorem 2.6 (1) and (2) we have

$$\begin{aligned} (\psi_1 \oplus \psi_2)^{\downarrow P_i \cap H_{i-1}} &= \left((\psi_1 \oplus \psi_2)^{\downarrow P_i}\right)^{\downarrow P_i \cap H_{i-1}} \\ &= \left(\psi_1^{\downarrow P_i \cap H_{i-1}} \oplus \psi_2\right)^{\downarrow P_i \cap H_{i-1}} = \psi_1^{\downarrow P_i \cap H_{i-1}} \oplus \psi_2^{\downarrow P_i \cap H_{i-1}} . \end{aligned}$$

Using the idempotency of the combination of propositional information and again Theorem 2.6 (2), it follows therefore

$$\begin{aligned} (\psi_1 \oplus \psi_2)^{\downarrow P_i \cap P_{s(i)}} \oplus \psi_2 &= \psi_1^{\downarrow P_i \cap H_{i-1}} \oplus \psi_2^{\downarrow P_i \cap H_{i-1}} \oplus \psi_2 \\ &= \psi_1^{\downarrow P_i \cap H_{i-1}} \oplus \psi_2 = (\psi_1 \oplus \psi_2)^{\downarrow P_i} . \end{aligned}$$

As $P_i \subseteq H_i$, we obtain from this result

$$\begin{aligned} \varphi^{\downarrow P_i \cap P_{s(i)}} \oplus \varphi_i^{(i)} &= \left(\varphi^{\downarrow H_i}\right)^{\downarrow P_i \cap P_{s(i)}} \oplus \varphi_i^{(i)} = (\psi_1 \oplus \psi_2)^{\downarrow P_i \cap P_{s(i)}} \oplus \psi_2 \\ &= (\psi_1 \oplus \psi_2)^{\downarrow P_i} = \left(\varphi^{\downarrow H_i}\right)^{\downarrow P_i} = \varphi^{\downarrow P_i} . \end{aligned}$$

This proves the theorem. □

A.5 Proof of Theorem 2.10

Apply Theorem 2.8 to

$$\varphi' = (\varphi_1 \oplus \varphi'_1) \oplus \varphi_2 \oplus \cdots \oplus \varphi_m$$

to obtain

$$\varphi'^{\downarrow P_i} = \varphi'^{\downarrow P_i \cap P_{s(i)}} \oplus \varphi_i^{(i)}. \tag{A.1}$$

Note here that, according to the inward phase, $\varphi_i^{(i)}$ is not changed by the new information φ'_1. Let now $T(i)$ be the subtree of the join tree associated with node i and put

$$\psi_1 = \bigoplus_{j \in T(i)} \varphi_j, \quad \psi_2 = \bigoplus_{j \notin T(i)} \varphi_j$$

such that $\varphi = \psi_1 \oplus \psi_2$, $P_i \subseteq d(\psi_1)$, $P_i \subseteq H_m = d(\varphi)$ and $\psi_1^{\downarrow P_i} = \varphi_i^{(i)}$ (see (2.16)). We obtain then, using Theorem 2.6 (2) and the idempotency of the combination of propositional information

$$\begin{aligned}
\varphi^{\downarrow P_i} \oplus \varphi_i^{(i)} &= (\psi_1 \oplus \psi_2)^{\downarrow P_i} \oplus \psi_1^{\downarrow P_i} = (\psi_1 \oplus \psi_2)^{\downarrow P_i \cap H_m} \oplus \psi_1^{\downarrow P_i} \\
&= \left(\psi_1^{\downarrow P_i} \oplus (\psi_1 \oplus \psi_2)\right)^{\downarrow P_i} = (\psi_1 \oplus \psi_2)^{\downarrow P_i} = \varphi^{\downarrow P_i},
\end{aligned}$$

and, similarly

$$\begin{aligned}
\varphi'^{\downarrow P_i} \oplus \varphi^{\downarrow P_i} &= (\varphi \oplus \varphi'_1)^{\downarrow P_i} \oplus \varphi^{\downarrow P_i} = (\varphi \oplus \varphi'_1)^{\downarrow P_i \cap H_m} \oplus \varphi^{\downarrow P_i} \\
&= \left(\varphi^{\downarrow P_i} \oplus (\varphi \oplus \varphi'_1)\right)^{\downarrow P_i} = (\varphi \oplus \varphi'_1)^{\downarrow P_i} = \varphi'^{\downarrow P_i}.
\end{aligned}$$

From this and (A.1) it follows then that

$$\begin{aligned}
\varphi'^{\downarrow P_i} &= \varphi'^{\downarrow P_i} \oplus \varphi^{\downarrow P_i} = \varphi'^{\downarrow P_i \cap P_{s(i)}} \oplus \varphi_i^{(i)} \oplus \varphi^{\downarrow P_i} \\
&= \varphi'^{\downarrow P_i \cap P_{s(i)}} \oplus \varphi^{\downarrow P_i}.
\end{aligned}$$

This proves the theorem. □

A.6 Proof of Theorem 2.12

Let Σ_1 be the set of all clauses ξ_1, which can be obtained by selecting a formula $\xi \in \Sigma$ and then choosing a literal from each one of the conjunctions of ξ. If a symbol appears in two conjunctions and it is selected in one of them, then it will also be selected in the other one. We never choose a symbol and its negation.

This set of clauses Σ_1 is equivalent to Σ because every disjunction ξ of conjunctions from Σ is converted into a conjunctive normal form (interpreted as a set of clauses) where subsumed clauses are eliminated. Let Σ'_1 be the set of clauses obtained from Σ_1 by deleting p_1 in clausal form and Σ'' the set of formulae obtained from Σ by deleting p_1 in disjunctive normal form. We will show that Σ'_1 and Σ'' are equivalent, with which this theorem will be a consequence of Theorem 2.2.

First, we are going to show that all the formulae in Σ'_1 are a consequence of Σ''. If a clause ξ'_1 belongs to Σ'_1, then we have two possibilities:

1. Clause ξ'_1 was in Σ_1, which means it does not contain p_1 nor $\sim p_1$. In this case it can be obtained by choosing a literal from each one of the conjunctions of a formula in disjunctive normal form, $\xi \in \Sigma$. None of the chosen literals is p_1 or $\sim p_1$. In this case it is immediate that ξ'_1 is a consequence of ξ^{-p_1} which is a formula belonging to Σ''.
2. Clause ξ'_1 was not in Σ_1. In this case, ξ'_1 is the result of the resolution of two clauses from Σ_1, ξ_1^+ and ξ_1^-, containing p_1 and $\sim p_1$ respectively. Two situations are now possible:

(a) ξ_1^+ and ξ_1^- come from the same formula $\xi \in \Sigma$. We can express ξ as

$$\xi : \quad (p_1 \wedge \psi_1) \vee \ldots \vee (p_1 \wedge \psi_k) \vee (\sim p_1 \wedge \psi_{k+1}) \vee \ldots \vee (\sim p_1 \wedge \psi_l) \vee (\psi_{l+1}) \vee \ldots \vee (\psi_s),$$

where $\psi_{l+1}, \ldots, \psi_s$ do not contain p_1 nor $\sim p_1$. ξ_1^+ and ξ_1^- will be expressed as

$$\begin{aligned} \xi_1^+ \quad &: \quad p_1 \vee t_{k+1} \vee \ldots \vee t_s, \\ \xi_1^- \quad &: \quad r_1 \vee \ldots \vee r_k \vee \sim p_1 \vee r_{l+1} \vee \ldots \vee r_s, \end{aligned}$$

where $t_{k+1}, \ldots, t_s$ are literals from $\psi_{k+1}, \ldots, \psi_s$ respectively and $r_1, \ldots, r_k, r_{l+1}, \ldots, r_s$ are literals from $\psi_1, \ldots, \psi_k, \psi_{l+1}, \ldots, \psi_s$ respectively.
The resolution of ξ_1^+ and ξ_1^- will be expressed as

$$t_{k+1} \vee \ldots \vee t_s \vee r_1 \vee \ldots \vee r_k \vee r_{l+1} \vee \ldots \vee r_s$$

and it is immediately clear that this formula is a consequence of

$$\xi^{-p_1} \quad : \quad \psi_1 \vee \ldots \vee \psi_k \vee \psi_{k+1} \vee \ldots \vee \psi_l \vee \psi_{l+1} \vee \ldots \vee \psi_s.$$

(b) ξ_1^+ comes from a formula $\xi \in \Sigma$ and ξ_1^- from a different formula $\varphi \in \Sigma$. Assume that,

$$\begin{aligned} \xi \quad &: \quad (p_1 \wedge \psi_1) \vee \ldots \vee (p_1 \wedge \psi_k) \vee (\psi_{k+1}) \vee \ldots \vee (\ \psi_l), \\ \varphi \quad &: \quad (\sim p_1 \wedge \zeta_1) \vee \ldots \vee (\sim p_1 \wedge \zeta_s) \vee (\zeta_{s+1}) \vee \ldots \vee (\zeta_d), \end{aligned}$$

where $\psi_{k+1}, \ldots, \psi_l$ do not contain p_1 and $\zeta_{s+1}, \ldots, \zeta_d$ do not contain $\sim p_1$. Furthermore,

$$\begin{aligned} \xi_1^+ \quad &: \quad p_1 \vee t_{k+1} \vee \ldots \vee t_l, \\ \xi_1^- \quad &: \quad \sim p_1 \vee r_{s+1} \vee \ldots \vee r_d, \end{aligned}$$

where $t_{k+1}, \ldots, t_l$ are literals from conjunctions $\psi_{k+1}, \ldots, \psi_l$ respectively and $r_{s+1}, \ldots, r_d$ are literals from $\zeta_{s+1}, \ldots, \zeta_d$, all these literals being different from p_1 and $\sim p_1$.
The resolution of ξ_1^+ and ξ_1^- is:

$$t_{k+1} \vee \ldots \vee t_l \vee r_{s+1} \vee \ldots \vee r_d$$

and this formula is a consequence of formula

$$\rho_{p_1}(\xi, \varphi) \quad = \quad ((\psi_{k+1}) \vee \ldots \vee (\ \psi_l) \vee (\zeta_{s+1}) \vee \ldots \vee (\zeta_d))^{-p_1}\ ,$$

which belongs to Σ''.

Inversely, every formula in Σ'' is a consequence of Σ_1'. Let $\xi \in \Sigma''$, then this formula does not contain p_1 and it is a consequence of Σ. As Σ_1' generates all the logical consequences of Σ (Σ is equivalent to Σ_1) not containing p_1, then ξ can be deduced from Σ_1', such that finally $\Sigma_1' \models \Sigma''$. □

A.7 Proof of Theorem 3.2

This theorem can be proved by transforming the definition of quasi-supports as follows:

$$\begin{aligned} QS_\varphi(h) \quad &= \quad \{a \in C_A : \ \Sigma, a \models h\} \\ &= \quad \{a \in C_A : \ \Sigma, \sim h \models \sim a\}. \end{aligned}$$

Then, according to (C2) and knowing that $\sim a$ are clauses in A, we can replace $(\Sigma, \sim h)$ by Σ':

$$\begin{aligned} QS_\varphi(h) \quad &= \quad \{a \in C_A : \ \Sigma' \models \sim a\} \\ &= \quad \{a \in C_A : \ a \models \sim \Sigma'\}. \end{aligned}$$

Therefore, Σ' and $QS_\varphi(h)$ are logically equivalent. □

A.8 Proof of Theorem 3.3

Let $\varphi = (\Sigma, A, P)$, $\varphi' = (\Sigma', A, Q)$. First, let $a \in QS_\varphi(h)$ s.t. $a, \Sigma \models h$ or $\Sigma \models h \vee {\sim}a$. But, then $h \vee {\sim}a \in \mathcal{L}_{A \cup Q}$ and $\Sigma' \models h \vee {\sim}a$ or $a, \Sigma' \models h$ s.t. $a \in QS_{\varphi'}(h)$.

Inversely, let $a \in QS_{\varphi'}(h)$ s.t. $a, \Sigma' \models h$ or $\Sigma' \models h \vee {\sim}a$. But $\Sigma \models \Sigma'$ implies $\Sigma \models h \vee {\sim}a$ or $a, \Sigma \models h$ hence $a \in QS_\varphi(h)$. □

A.9 Proof of Theorem 3.4

Let $\varphi = (\Sigma, A, P)$ and $\varphi^{\downarrow(A',P')} = (\Sigma', A', P')$ and

$$a' \in QS_{\varphi^{\downarrow(A',P')}}(h) = \{a' \in \mathcal{C}_{A'} : a', \Sigma' \models h\}$$

for some $h \in \mathcal{L}_{A' \cup P'}$. Now, $a', \Sigma' \models h$ is equivalent to $\Sigma' \models h \vee {\sim}a' \in \mathcal{L}_{A' \cup P'}$. As $\Sigma \models \Sigma'$ we have $\Sigma \models h \vee {\sim}a'$ or $a', \Sigma \models h$. This shows that $a' \in QS_\varphi(h)$ or $[a'] \leq qs_\varphi([h])$.

On the other hand, if

$$[a'] \leq qs_\varphi([h]) = \bigvee_{a \in QS_\varphi(h)} [a], \qquad a' \in \mathcal{C}_{A'}$$

for some $h \in \mathcal{L}_{A' \cup P'}$, then this inequality implies

$$a' \models \bigvee_{a \in QS_\varphi(h)} a$$

and therefore,

$$a', \Sigma \models \bigvee_{a \in QS_\varphi(h)} a, \Sigma \models h,$$

which shows that $a' \in QS_\varphi(h)$. But then this implies $\Sigma \models h \vee {\sim}a' \in \mathcal{L}_{A' \cup P'}$ which in turn implies $\Sigma' \models h \vee {\sim}a'$, hence $a', \Sigma' \models h$ and thus finally

$$a' \in QS_{\varphi^{\downarrow(A',P')}}(h).$$

This proves that for every $h \in \mathcal{L}_{A' \cup P'}$

$$QS_{\varphi^{\downarrow(A',P')}}(h) = \{a' \in \mathcal{C}_{A'} : [a'] \leq qs_\varphi([h])\}$$

and this proves the theorem. □

Received 16 December 1996