Automatic Derivation of Probabilistic Inference Rules *

Manfred Jaeger
Max-Planck-Institut für Informatik
Im Stadtwald, 66123 Saarbrücken,
Germany

email: jaeger@mpi-sb.mpg.de

Abstract

A probabilistic inference rule is a general rule that provides bounds on a target probability given constraints on a number of input probabilities. Example: from $P(A|B) \leq r$ infer $P(\neg A|B) \in [1-r, 1]$. Rules of this kind have been studied extensively as a deduction method for propositional probabilistic logics. Many different rules have been proposed, and their validity proved – often with substantial effort. Building on previous work by T. Hailperin, in this paper we show that probabilistic inference rules can be derived automatically, i.e. given the input constraints and the target probability, one can automatically derive the optimal bounds on the target probability as a functional expression in the parameters of the input constraints.

Keywords: Probabilistic Logic, Inference Rules, Conditional Probabilities, Symbolic Computation

* to appear in *Int. J. of Approximate Reasoning*

1 Introduction

Probabilistic information very often is obtained in the form of conditional probability statements. An expert on tropical diseases, for example, will typically express his knowledge in statements of the form "the probability of drowsiness given that a patient suffers from malaria is at least 0.8"; a robot engaged in some navigation task might have to process statements like "the probability that the exit is 90 degrees to the left given that the sensor readings are correct is at least 0.6". The formal representation and manipulation of such (conditional or unconditional) constraints, therefore, was one of the earliest topics that arose in probabilistic reasoning for AI applications [15].

As a formal representation language one typically uses a probabilistic extension of propositional logic. Simple probability statements like the ones given above then become equivalent to linear constraints on probability assignments to the (finitely many) truth assignments (or possible worlds) for the propositional variables. Probabilistic inference, then, consists of deriving probability bounds that are entailed by the given constraints for propositions of interest.

A general problem that arises in this approach is that very often these entailed bounds will be fairly unspecific, i.e. a knowledge base of probability statements (linear constraints) might entail only a lower bound of 0.05, say, and an upper bound of 0.9 for the proposition of interest. For this reason the focus of research has somewhat shifted from pure probabilistic (propositional) logics to representation and inference systems that guarantee unique probability values to be deducible – notably Bayesian networks. The precision of inference gained, however, here comes at the cost of very stringent requirements for the specification of probabilistic knowledge. As we cannot assume that these requirements can always be met, the general problem of deducing (bounds on) probability values from arbitrary collections of probability statements still is of fundamental importance.

For probability constraints expressed in propositional probabilistic logic there exist basically two distinct approaches to deduction: the global optimization approach, and the local inference rule approach. In the global optimization approach (e.g.[16, 13]) one obtains the bounds entailed by a knowledge base of probability constraints for a specific target (conditional) probability by linear (fractional) optimization on the space of all probability assignments to possible worlds. This method has the advantage of always yielding the exact bounds entailed for the target probability, but has the disadvantage of requiring space and time exponential in the size of the knowledge base.

In the local inference rule approach one derives entailed bounds step-by-step by applying probabilistic inference rules to the constraints in the knowledge base and/or intermediate constraints already derived. A very simple such inference rule is

$$\frac{P(A \mid B) \leq r}{P(\neg A \mid B) \in [1-r, 1]} \tag{1}$$

In a particular deduction problem, this rule can be applied if, for instance, we have already found that $P(A \mid B) \leq 0.7$. An application of rule (1) will then yield the bounds $P(\neg A \mid B) \in [0.3, 1]$. By successive applications of various rules one then tries to derive increasingly tight bounds for the target probability. This method has the advantage that it operates on the level of the propositional language, not on the level of truth assignments, and therefore avoids the initial construction of an exponentially large state space. At any point in the derivation, the method provides a current best bound derived, which can serve as an approximation to the ultimate solution ("anytime" character of probabilistic deduction [6]). Disadvantages of the rule-based derivations are that one usually cannot tell whether the current bounds already are optimal, or whether further inferences may lead to tighter bounds; and that one has to find a suitable proof strategy to derive any useful bounds at all. Moreover, depending on specific aspects of the underlying representation language (which, for instance, may only admit atomic constraints of the form $P(\phi \mid \psi) \leq r$, or else arbitrary linear constraints on probability values, or even Boolean combinations of linear constraints) and the available inference rules, rule-based deduction can be incomplete, i.e. it can happen that for a given knowledge base and a target probability no derivation of the optimal bounds exists.

Generally, rule based deduction tends to have its biggest advantage over global optimization when one works in a restricted setting where knowledge bases are of some specialized type that admit the design of special purpose inference rules and proof strategies. Thus, it is not surprising that much of the work on probabilistic deduction was either in the context of such specialized representation systems [20, 12, 14], or completeness results were only obtained for restricted classes of knowledge bases [6].

Results by Fagin, Halpern and Megiddo [5], on the other hand, show that even for very expressive representation languages one can devise complete local inference systems (in fact, the expressiveness of the language here furthers, not hinders, completeness, because it allows us to store more complex intermediate results in the course of the derivation). The system presented in [5] essentially consists of the axioms of probability theory, and axioms for reasoning with linear inequalities, together with a single inference rule, modus ponens. Rules of the form (1),

thus, do not directly appear in this system. They might be added to the system as derived rules, however, in order to allow for shorter proofs in this system.

A great number of different probabilistic inference rules have been proposed in the AI literature [1, 20, 6, 12, 14], mostly as part of a specialized representation and inference system. To devise suitable inference rules, one has to solve the following problem: given a probabilistic premise, e.g. $P(A \mid B) \leq r$ in (1), and a target probability ($P(\neg A \mid B)$), what are the best upper and lower bounds one can infer for the target probability as a function of the parameters (r) in the premise? Dubois, Prade and Toucas [4], for instance, solve the problem for the premises $P(A \mid B) = r_1$, $P(B \mid A) = r_2$, $P(C \mid B) = r_3$, $P(B \mid C) = r_4$, and the target probability $P(A \mid C)$. Lukasiewicz [14] solves the same problem with two generalizations: instead of point-valued probabilities in the premise he treats the more general case of interval-valued probabilities ($P(A \mid B) \in [l_1, u_1], \ldots, P(B \mid C) \in [l_4, u_4]$), and, more importantly, he adds various "taxonomic" constraints to the premises (e.g. $P(C \wedge \neg A) = 0$).

Independent from (and preceding) this work in AI, the same problem of deriving probabilistic inference rules has been studied by T. Hailperin [9, 10, 11], motivated by his studies of the work of George Boole, who, in turn, already was concerned with probabilistic inference rules [2, 3]. Hailperin pursued a general approach, and, beyond deriving particular inference rules, investigated general methods for their derivation. Based on this work of Hailperin we will describe in this paper a general algorithmic method that derives for arbitrary conditional probability bounds given as a premise the optimal bounds that can be inferred for a given conditional target probability. In other words, the algorithm we describe can be used to automatically generate probabilistic inference rules of any desired form. The contribution of this paper, thus, is methodological: the algorithm presented can be used as a design tool for probabilistic inference systems; it is not to be confused with probabilistic inference algorithms that are used at run time in actual application systems.

2 Rules, Polytopes, and Bound Functions

In this section we first make precise the problem statement of deriving a probabilistic inference rule, then present in Lemma 2.1 the mathematical foundation for the algorithm we will propose, an outline of which is given at the end of the section. Sections 3 and 4 contain a more detailed description of the algorithm.

The general form of probabilistic inference rules that we are going to consider is given by

$$\frac{\begin{array}{l} P(\phi_1 \mid \psi_1) \sim_1 r_1 \\ \vdots \\ P(\phi_N \mid \psi_N) \sim_N r_N \end{array}}{P(\chi \mid \gamma) \in [L(r_1, \ldots, r_N), U(r_1, \ldots, r_N)]} \tag{2}$$

where $\phi_1, \psi_1, \ldots, \phi_N, \psi_N, \chi, \gamma$ are propositional formulas in a language of K propositional variables $\{A_1, \ldots, A_K\}$, the $\sim_i$ are one of $\leq, \geq, =$, the r_i are *parameter symbols*, and $L(\ldots)$, $U(\ldots)$ are functions from $[0,1]^N$ to $[0,1]$ whose arguments are denoted by the parameter symbols. A formula ψ_i may, in particular, be a propositional tautology, in which case we obtain an unconditional premise $P(\phi_i) \sim_i r_i$.

To define the correctness and optimality of an inference rule, we briefly have to review some standard concepts pertaining to propositional probabilistic models and linear constraints. The propositional variables $\{A_1, \ldots, A_K\}$ generate a Boolean algebra $\mathscr{B}(A_1, \ldots, A_K)$ of propositions in these variables. Using $\bar{A}$ for the negation of A, and concatenation for conjunction, we can then identify $\mathscr{B}$ with its 2^K atoms $A_1^* A_2^* \ldots A_K^*$ ($A_i^* \in \{A_i, \bar{A}_i\}$ $(1 \leq i \leq K)$). Probability distributions on $\mathscr{B}$ are given by probability assignments to its atoms, and thus by elements of the set Δ^{2^K}, where, generally,

$$\Delta^n := \{(p_1, \ldots, p_n) \in [0,1]^n \mid \sum_{i=1}^{n} p_i = 1\}.$$

Now return to the inference rule (2). For brevity, in the sequel we denote the ith premise $P(\phi_i \mid \psi_i) \sim_i r_i$ by c_i, and sometimes refer to it as the *ith input constraint*. Furthermore, we let $\mathscr{C} := \{c_1, \ldots, c_N\}$. Let

$$I : \{r_1, \ldots, r_N\} \to [0,1]$$

be an *instantiation* of the parameter symbols. Then the *instantiated constraint*

$$I(c_i) := P(\phi_i \mid \psi_i) \sim_i I(r_i) \tag{3}$$

defines a set of probability distributions on $\mathscr{B}$: using x_k as a variable for the probability of the kth atom α_k of $\mathscr{B}$ ($k = 1, \ldots, 2^K$), we can write (3) as

$$\sum_{k:\alpha_k \to \phi_i \wedge \psi_i} x_k \;\Big/ \sum_{k:\alpha_k \to \psi_i} x_k \;\; \sim_i \;\; I(r_i), \tag{4}$$

or, equivalently, as the linear (in-)equation

$$\sum_{k:\alpha_k \to \phi_i \wedge \psi_i} (1 - I(r_i)) x_k \;\; - \sum_{k:\alpha_k \to \neg\phi_i \wedge \psi_i} I(r_i) x_k \;\; \sim_i \;\; 0. \tag{5}$$

$I(c_i)$, thus, defines the set of distributions $\boldsymbol{p} \in \Delta^{2^K}$ that are a solution of (5). We do not strictly distinguish between an (instantiated) input constraint (3) and its algebraic form (5), and use $I(c_i)$ also to refer to the latter.

Geometrically, the solutions of (5) in Δ^{2^K} are given by the intersection of Δ^{2^K} with a hyperplane (if $\sim_i$ is =) or a halfspace (if $\sim_i \in \{\leq, \geq\}$). The elements of Δ^{2^K} that satisfy $I(\mathscr{C}) := \{I(c_1), \ldots, I(c_N)\}$ then are the intersection of Δ^{2^K} with N hyperplanes or halfspaces, i.e. a polytope. This polytope we denote by $\Delta(I(\mathscr{C}))$.

For any instantiation $I(\mathscr{C})$ of the input constraints, we now are interested in bounds entailed by $I(\mathscr{C})$ for the target conditional probability $P(\chi \mid \gamma)$. Defining for a subset $\Pi \subseteq \Delta^{2^K}$

$$\Pi(\chi \mid \gamma) := \{\boldsymbol{p}(\chi \mid \gamma) \mid \boldsymbol{p} \in \Pi,\ \boldsymbol{p}(\gamma) > 0\},$$

this means that we wish to compute $\Delta(I(\mathscr{C}))(\chi \mid \gamma)$.

We can rephrase this in a more logical terminology: a set of instantiated constraints $I(\mathscr{C})$ is (part of) a knowledge base, the set $\Delta(I(\mathscr{C}))$ is the set of models of $I(\mathscr{C})$, and $\Delta(I(\mathscr{C}))(\chi \mid \gamma)$ is the set of conditional probabilities of χ given γ in some model of $I(\mathscr{C})$. Thus, to compute $\Delta(I(\mathscr{C}))(\chi \mid \gamma)$ is to deduce the bounds on $P(\chi \mid \gamma)$ that are entailed by $I(\mathscr{C})$.

Note that $\Delta(I(\mathscr{C}))(\chi \mid \gamma) = \emptyset$ can hold for two reasons: either because $\Delta(I(\mathscr{C})) = \emptyset$, i.e. because of the instantiated constraints being inconsistent, or because $\boldsymbol{p}(\gamma) = 0$ for all $\boldsymbol{p} \in \Delta(I(\mathscr{C}))$, i.e. the constraints imply that $P(\gamma) = 0$. In the following we will not distinguish between these two cases, i.e. we will be content to deduce $P(\chi \mid \gamma) = \emptyset$ in both cases. It would not be difficult, however, to modify the inference rules we derive in such a way, that they will let us deduce $P(\chi \mid \gamma) = \emptyset$ only when $\Delta(I(\mathscr{C})) = \emptyset$, whereas in the case $\Delta(I(\mathscr{C})) \neq \emptyset$, but $\boldsymbol{p}(\gamma) = 0$ for all $\boldsymbol{p} \in \Delta(I(\mathscr{C}))$, a distinct conclusion $P(\chi \mid \gamma) = \texttt{undefined}$ would be obtained.

The following lemma describes $\Pi(\chi \mid \gamma)$ for the case of Π being a polytope. It is instrumental for all that follows.

Lemma 2.1 Let $\Pi \subseteq \Delta^{2^K}$ be a polytope with vertices $\{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_m\}$, and $\chi, \gamma \in \mathscr{B}(A_1, \ldots, A_K)$. Then $\Pi(\chi \mid \gamma)$ is a closed interval $[l, u]$ with

$$\begin{aligned} l &= \mathit{min}\{\boldsymbol{p}_i(\chi \mid \gamma) \mid 1 \leq i \leq m,\ \boldsymbol{p}_i(\gamma) > 0\} \\ u &= \mathit{max}\{\boldsymbol{p}_i(\chi \mid \gamma) \mid 1 \leq i \leq m,\ \boldsymbol{p}_i(\gamma) > 0\} \end{aligned}$$

(assuming the conventions $\mathit{min}\,\emptyset = 1$, $\mathit{max}\,\emptyset = 0$, and $[1, 0] = \emptyset$). [1]

[1]The first part of this lemma has also been stated by Frisch and Haddawy [6], whose proof, however, did not

Proof: That $\Pi(\chi \mid \gamma)$ is an interval follows directly from the fact that $\boldsymbol{p} \mapsto \boldsymbol{p}(\chi \mid \gamma)$ is a continuous function on the connected domain $\{\boldsymbol{p} \in \Pi \mid \boldsymbol{p}(\gamma) > 0\}$. We thus have to show that the endpoints of the interval are actually attained at some vertices of Π. We first make a simple observation: if $\boldsymbol{p}, \boldsymbol{q} \in \Delta^{2^K}$ are such that $\boldsymbol{p}(\gamma) > 0$ and $\boldsymbol{q}(\gamma) > 0$, then the function $\lambda \mapsto (\lambda\boldsymbol{p} + (1-\lambda)\boldsymbol{q})(\chi \mid \gamma)$ ($\lambda \in [0,1]$) is monotone, i.e. the conditional probability of χ given γ increases or decreases monotonically along the line segment joining $\boldsymbol{p}$ and $\boldsymbol{q}$. If $\boldsymbol{p}(\gamma) > 0$, but $\boldsymbol{q}(\gamma) = 0$, then $(\lambda\boldsymbol{p} + (1-\lambda)\boldsymbol{q})(\chi \mid \gamma) = \boldsymbol{p}(\chi \mid \gamma)$ for all $\lambda \in (0,1]$, i.e.the conditional probability of χ given γ is constant on the half open line segment joining $\boldsymbol{p}$ and $\boldsymbol{q}$.

Now we can prove the lemma by induction on the dimension d of Π. For $d = 0$, Π is a single point, and the statement is trivially true. Now let $d > 0$, and let $\boldsymbol{p}$ be a vertex of Π with minimal value $\boldsymbol{p}(\chi \mid \gamma)$. Assume that there exists some other element $\boldsymbol{p}' \in \Pi$ with $\boldsymbol{p}'(\gamma) > 0$ and $\boldsymbol{p}'(\chi \mid \gamma) < \boldsymbol{p}(\chi \mid \gamma)$. Let l be the (unbounded) line through $\boldsymbol{p}$ and $\boldsymbol{p}'$. The intersection of l with Π is a line segment whose one endpoint is $\boldsymbol{p}$, and whose other endpoint is some $\boldsymbol{q}$. By our observation above we have $\boldsymbol{q}(\gamma) > 0$ and $\boldsymbol{q}(\chi \mid \gamma) < \boldsymbol{p}(\chi \mid \gamma)$, since otherwise $\boldsymbol{p}'(\chi \mid \gamma) < \boldsymbol{p}(\chi \mid \gamma)$ could not hold. Now we have found in $\boldsymbol{q}$ an element on the (relative) boundary of Π with $\boldsymbol{q}(\chi \mid \gamma) < \boldsymbol{p}(\chi \mid \gamma)$. But being on the boundary means that $\boldsymbol{q}$ lies in a $(d-1)$-dimensional facet of Π. To this facet the induction hypothesis applies, and shows that $\boldsymbol{q}(\chi \mid \gamma)$ cannot be smaller than the minimal conditional probability of χ given γ attained at some vertex of that facet, and hence cannot be smaller than $\boldsymbol{p}(\chi \mid \gamma)$. □

One can construct examples that show that Lemma 2.1 cannot be extended to arbitrary closed and convex subsets Π of Δ^{2^K}, i.e. there exist such Π for which $\Pi(\chi \mid \gamma)$ is not closed.

Letting $\Pi := \Delta(I(\mathscr{C}))$, Lemma 2.1 gives for a set $\mathscr{C}$ of input constraints, a target conditional probability $P(\chi \mid \gamma)$, and a parameter instantiation I the optimal bounds $l(I), u(I)$ for the target probability under the instantiated constraints. To obtain the probabilistic inference rule for the input constraints $\mathscr{C}$ and the target probability $P(\chi \mid \gamma)$, we have to determine explicit representations of the *bound functions*

$$L(r_1, \ldots, r_N), U(r_1, \ldots, r_N) : [0,1]^N \to [0,1]$$

correctly establish the closure of the entailed interval: the sentence that in their proof starts with "If for some point in the feasible region $W_\xi = 0, \ldots$ " should correctly read "If for *all* points $\ldots$ ". But then in the "Otherwise" case the domain of optimization is not closed, and hence the closure of the entailed interval does not follow by continuity arguments.

such that for all instantiations I

$$\begin{array}{l} I(L) := L(I(r_1), \ldots, I(r_N)) = l(I) \\ I(U) := U(I(r_1), \ldots, I(r_N)) = u(I). \end{array} \tag{6}$$

When the bound functions L and U satisfy (6) for all I, then we say that (2) is a correct and optimal [2] inference rule. If L and U only satisfy $I(L) \leq l(I)$, $I(U) \geq u(I)$, then rule (2) would be called correct, but not optimal. The algorithm we develop always derives correct and optimal inference rules.

Our strategy for finding L and U is derived from the second part of Lemma 2.1, and in broad outline is as follows: we shall first compute a complete list of the vertices of the $\Delta(I(\mathscr{C}))$ in parameterized form, i.e. we determine a list of M parameterized points

$$\boldsymbol{v}_j = (v_{j,1}(r_1, \ldots, r_N), \ldots, v_{j,2^K}(r_1, \ldots, r_N)) \tag{7}$$

$(1 \leq j \leq M)$ with functions

$$v_{j,i} : \ [0,1]^N \to [0,1],$$

such that for every instantiation I the set of all points

$$\begin{array}{rcl} I(\boldsymbol{v}_j) & := & (I(v_{j,1}), \ldots, I(v_{j,2^K})) \\ & := & (v_{j,1}(I(r_1), \ldots, I(r_N)), \ldots, v_{j,2^K}(I(r_1), \ldots, I(r_N))) \end{array}$$

$(1 \leq j \leq M)$ is just the set of vertices of $\Delta(I(\mathscr{C}))$. We then evaluate the target conditional probability at every parameterized vertex, obtaining functions $\boldsymbol{v}_j(\chi \mid \gamma)$ in the parameters $r_1, \ldots, r_N$. Finally, we put

$$\begin{array}{l} L(r_1, \ldots, r_N) := \mathit{min}\{\boldsymbol{v}_j(\chi \mid \gamma) \mid 1 \leq j \leq M, \ \boldsymbol{v}_j(\gamma) > 0\} \\ U(r_1, \ldots, r_N) := \mathit{max}\{\boldsymbol{v}_j(\chi \mid \gamma) \mid 1 \leq j \leq M, \ \boldsymbol{v}_j(\gamma) > 0\}, \end{array} \tag{8}$$

which, by Lemma 2.1, defines the optimal bound functions. We divide the following detailed description of this method into two parts: the first part (Section 3) describes how to compute the parameterized vertex list (7); the second part (Section 4) describes how to obtain L and U from the vertex list.

3 The Parameterized Vertex List

The first part of the algorithm outlined at the end of the previous section consists of the computation of the parameterized vertices (7). We first illustrate the general method of computation

[2] Also called "locally complete" [1] or "(quasi-) tight" [6]

by considering as an example the two input constraints

$$c_1 : \ P(A \mid B) \geq r \tag{9}$$

$$c_2 : \ P(A \mid A \vee B) \geq s \tag{10}$$

(the target probability of the desired inference rule plays no role at this point). We number the atoms of the algebra $\mathscr{B}$ generated by the two propositional variables A and B as follows

$$\alpha_1 := AB, \ \alpha_2 := A\bar{B}, \ \alpha_3 := \bar{A}B, \ \alpha_4 := \bar{A}\bar{B}.$$

The constraints (9) and (10) in algebraic form then become

$$(1-r)x_1 - rx_3 \geq 0 \tag{11}$$

$$(1-s)x_1 + (1-s)x_2 - sx_3 \geq 0. \tag{12}$$

Figure 1 shows the polytopes defined by (11) and (12) for three different instantiations of r and s. The individual figures show the set Δ^4 whose vertices $(1,0,0,0)$, $(0,1,0,0)$, $(0,0,1,0)$, $(0,0,0,1)$ correspond to the distributions that assign probability 1 to either of the four atoms of $\mathscr{B}$. The halfspaces defined by the instantiated constraints $I(c_1), I(c_2)$ are indicated as shaded regions, the intersection of both halfspaces by a darker shading. The vertices of the resulting polytopes $\Delta(I(c_1, c_2))$ are marked by dots. As this example shows, the number and position of the vertices of $\Delta(I(\mathscr{C}))$ can change substantially for different parameter instantiations. However, each vertex is given as the intersection of hyperplanes defined by the constraints with faces of Δ^4. Figure 2 shows the general position of these intersection points, and Table 1 presents them as a list. The third column of Table 1 states the conditions on the parameters for when the intersection point is an actual vertex of the polytope. Such conditions we will subsequently encounter in great numbers; they are formally introduced by the following definition.

Definition 3.1 A *parameter constraint* (p-constraint for short) for the parameters $r_1, \ldots, r_N$ is an inequality of the form

$$p(r_1, \ldots, r_N) \sim 0,$$

where $p(\ldots)$ is a polynomial in the variables $r_1, \ldots, r_N$ with rational coefficients, and $\sim$ is one of $\leq, \geq, <, >, =$.

We use π to denote a single p-constraint, and $\boldsymbol{\pi}$ for lists $\pi_1, \ldots, \pi_m$ of p-constraints (and often take the liberty to write a p-constraint in the form $p \sim q$ with polynomials p and q). The

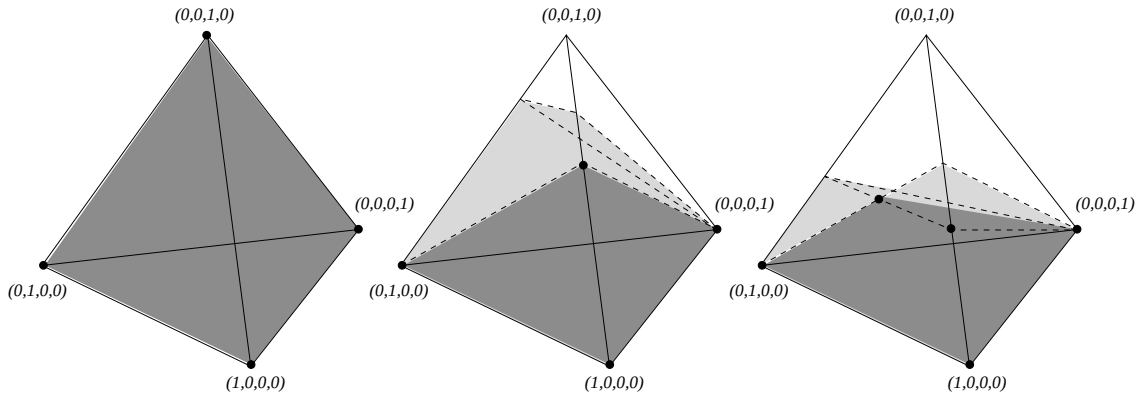

Figure 1: Polytopes for different parameter values

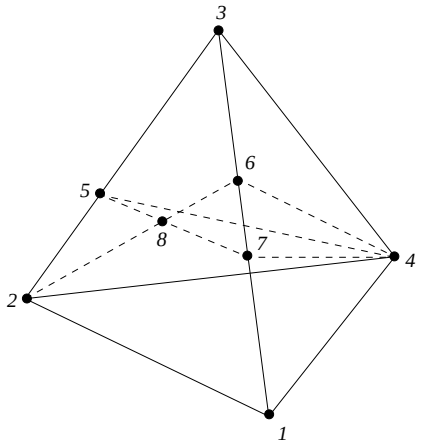

Figure 2: General vertex positions

problem statement for generating a complete parameterized vertex list can now be refined as follows: given input constraints $\mathscr{C}$, we have to find a list

$$\boldsymbol{v}_j : \boldsymbol{\pi}_j \quad (1 \leq j \leq M) \tag{13}$$

where each $\boldsymbol{v}_j$ is a parameterized vertex as in (7), and the $\boldsymbol{\pi}_j$ are lists of p-constraints, such that for every parameter instantiation I the set of vertices of $\Delta(I(\mathscr{C}))$ is just

$$\{I(\boldsymbol{v}_j) \mid \ I \text{ satisfies } \boldsymbol{\pi}_j\}$$

(where, naturally, I satisfies $\boldsymbol{\pi}$ iff for every $\pi_i \ \equiv \ p_i \sim 0 \ \in \boldsymbol{\pi}$: $I(p_i) \sim 0$).

Table 1 provides this list for the input constraints (9) and (10). To obtain a systematic method for generating such a list it is convenient to consider one by one the different faces of Δ^{2^K}, in

Vertex Position j	Coordinates $v_{j,1}$	$v_{j,2}$	$v_{j,3}$	$v_{j,4}$	Parameter constraints
1	1	0	0	0	
2	0	1	0	0	
3	0	0	1	0	$r = 0, s = 0$
4	0	0	0	1	
5	0	s	$1 - s$	0	$r = 0$
6	r	0	$1 - r$	0	$r \geq s$
7	s	0	$1 - s$	0	$s \geq r$
8	$\frac{r(1-s)}{1-r}$	$\frac{s-r}{1-r}$	$1 - s$	0	$r \neq 1, s \geq r$

Table 1: Parameterized vertices

increasing order of their dimension, say, and for any given face to compute the parameterized vertices lying in this face.

The d-dimensional faces f of Δ^{2^K} are just the convex hulls of any subset of $d+1$ vertices of Δ^{2^K}, and thus are given by $d+1$-element subsets H of $\{1, \ldots, 2^K\}$ according to

$$f(H) := \{(p_1, \ldots, p_{2^K}) \in \Delta^{2^K} \mid i \notin H \Rightarrow p_i = 0\}.$$

The (relative) interior of a face $f(H)$ is the set

$$\mathit{int}\, f(H) := \{(p_1, \ldots, p_{2^K}) \in \Delta^{2^K} \mid i \notin H \Leftrightarrow p_i = 0\}.$$

The vertices of $\Delta(I(\mathscr{C}))$ lying inside a specific face of Δ^{2^K} are characterized by the following lemma.

Lemma 3.2 Let $I(\mathscr{C}) = \{I(c_1), \ldots, I(c_N)\}$ be a set of instantiated input constraints. Let $H \subseteq \{1, \ldots, 2^K\}$ with $|H| = d+1$ and $\boldsymbol{p} \in \mathit{int}\, f(H)$. Then $\boldsymbol{p}$ is a vertex of $\Delta(I(\mathscr{C}))$ iff there exist d constraints $I(c_{i_1}), \ldots, I(c_{i_d})$ in $I(\mathscr{C})$ such that the following two conditions hold:

(i) $\boldsymbol{p}$ is the unique solution of the following system of 2^K linear equations

$$\sum_{i=1}^{2^K} x_i = 1 \tag{14}$$

$$x_i = 0 \qquad (i \notin H) \tag{15}$$

$$I(c_{i_j}^{=}) \qquad (1 \leq j \leq d) \tag{16}$$

where $c_{i_j}^{=}$ is the constraint obtained from c_{i_j} by replacing $\sim_{i_j}$ with $=$,

(ii) $\boldsymbol{p}$ satisfies the linear inequations

$$x_i \geq 0 \quad (i \in H) \tag{17}$$

$$I(c_j) \quad (c_j \in \mathscr{C} \setminus \{c_{i_1}, \ldots, c_{i_d}\}). \tag{18}$$

Proof: For any polytope $P \subseteq \mathbb{R}^n$ that is defined by k linearly independent linear equations and l linear inequations, we have that the vertices of P are the points that, first, are the unique solutions to a system of n equations obtained by combining the k given equations with the equality versions of $n - k$ of the given inequations, and, second, satisfy all the remaining inequations (see e.g. [18, Section 8.5]).

The polytope $\Delta(I(\mathscr{C}))$ is given by the one equation $\sum x_i = 1$, and the inequations $x_i \geq 0$ and $I(c_j)$ $(1 \leq j \leq N)$. With $n = 2^K$ we obtain that the vertices of $\Delta(I(\mathscr{C}))$ are just the points that satisfy (i) and (ii). Moreover, for $\boldsymbol{p} \in \mathit{int}\, f(H)$ we have that $\boldsymbol{p}$ satisfies the equations $x_i = 0$ exactly when $i \notin H$. If $\boldsymbol{p}$ is vertex, then the defining equations for $\boldsymbol{p}$, therefore, must contain d of the equations $I(c_i^=)$. □

To compute the parameterized vertex list we now solve the systems (14)-(16) of linear equations symbolically for the parameterized constraints $c_{i_j}^=$ instead of the instantiated constraints $I(c_{i_j}^=)$. Table 2 summarizes the algorithm. For the subroutine of solving the systems (14)-(16) in step 1 all methods for solving systems of linear equations are potentially applicable. A particularly suitable method is fraction free Gaussian elimination (see e.g. [7]). This is a variant of Gaussian elimination that avoids divisions, which is useful for us, as otherwise we would have to divide by symbolic expressions that might be zero for some parameter values and nonzero for others, thereby requiring us to make a number of case distinctions.

As an illustration for the working of the algorithm we retrace how vertex 8 in Table 1 was generated. This vertex is the solution of the system (14)-(16) defined by $d = 2$, $H = \{1, 2, 3\}$ and the (then mandatory) selection of both constraints c_1, c_2 for (16). This system leads to an initial matrix for Gaussian elimination as shown in the upper half of Table 3. Only one elimination step is needed to produce the bottom matrix in the table, which already is in upper triangular form. The system to have a unique solution now is equivalent to all the entries in the main diagonal being nonzero, which leads to the p-constraint $r \neq 1$. Under this constraint, the solution obtained is

$$\boldsymbol{v} = (\frac{r(1-s)}{1-r}, \frac{s-r}{1-r}, 1-s, 0).$$

procedure generate_vertex_list

Input: constraints $\mathscr{C} = \{c_1, \ldots, c_N\}$
Output: list of parameterized vertices of $\Delta(\mathscr{C})$.

For $d = 0..min\{2^K - 1, N\}$
  For all $H \subseteq \{1, \ldots, 2^K\}$ with $|H| = d + 1$
    For all subsets $\{c_{i_1}, \ldots, c_{i_d}\} \subseteq \mathscr{C}$
    1. solve the system
      $\sum_{i=1}^{2^K} x_i = 1,\ x_i = 0\ (i \notin H),\ c_{i_j}^{=}\ (1 \le j \le d)$
    # The solution returned is either the message "solution not
    # unique", or a parameterized point
    # $\boldsymbol{v} = (v_1(r_1, \ldots, r_N), \ldots, v_{2^K}(r_1, \ldots, r_N))$
    # together with a list $\boldsymbol{\pi}$ of p-constraints for the
    # point to be the unique solution of the system.
    If solution $\boldsymbol{v} : \boldsymbol{\pi}$ was returned
    2. For all $i \in H$
        append $v_i(r_1, \ldots, r_N) \ge 0$ to $\boldsymbol{\pi}$
      For all $c_j \in \mathscr{C} \setminus \{c_{i_1}, \ldots, c_{i_d}\}$
        append $c_j[x_1/v_1(r_1, \ldots, r_N), \ldots, x_{2^K}/v_{2^K}(r_1, \ldots, r_N)]$ to $\boldsymbol{\pi}$
    3. append $\boldsymbol{v} : \boldsymbol{\pi}$ to parameterized vertex list

Table 2: The first part of the algorithm

The solution returned by step 1 thus is $\boldsymbol{v}$ together with the p-constraint $r \neq 1$. In step 2 we now append the p-constraints

$$\frac{r(1-s)}{1-r} \ge 0,\ \frac{s-r}{1-r} \ge 0,\ 1 - s \ge 0$$

(the first and last of which, however, are vacuous because $r, s \in [0, 1]$). Step 3 here is void, because $\mathscr{C}$ does not contain any constraints not used in the definition of $\boldsymbol{v}$.

To conclude this section, we derive a bound on the number of parameterized vertices we will generate by this method, i.e. a bound on M in (13). For a fixed dimension d there are $\binom{2^K}{d+1}$ faces of dimension d in Δ^{2^K}, and for each face there are $\binom{N}{d}$ possibilities to select d constraints from the set of N input constraints. This leads to a number of

$$\binom{2^K}{d+1}\binom{N}{d} \le N^d 2^{K(d+1)}$$

systems of linear equations in the form of (14)-(16), each of which may lead to one parameterized vertex in a face of dimension d. Adding over $d = 0, \ldots, N$ leads to a (crude) upper

x_1	x_2	x_3	
1	1	1	1
$1-r$	0	$-r$	0
$1-s$	$1-s$	$-s$	0
1	1	1	1
0	$1-r$	1	$1-r$
0	0	1	$1-s$

Table 3: Solving linear equations

bound

$$(N+1)N^N 2^{K(N+1)} \sim N^{N+1}2^{K(N+1)} \tag{19}$$

for the total number of parameterized vertices.

4 The Bound Functions

In the second part of the algorithm the representation (8) of the bound functions is computed. This is a trivial step in principle, but as we shall see a nontrivial complication arises in practice. To illustrate the general method, we continue with our example, taking $P(\neg A \mid B)$ to be the target probability of the inference rule to be derived. The probability of $\neg A$ given B at the vertices listed in Table 1 is evaluated by computing $v_3/(v_1+v_3)$, which leads to the values listed in Table 4. Note that the possible values of $P(\neg A \mid B)$ are still annotated with the parameter constraints on the vertices at which they are attained, and that for vertices 5 and 8 the new p-constraint $s < 1$ has been added. This additional p-constraint specifies the condition under which $v_1 + v_3 > 0$, and thus $\boldsymbol{v}(\neg A \mid B)$ being well-defined. At vertex 4 this is not the case for any parameter values. Table 4 now gives for every instantiation of the parameters the possible extremal values of $\Delta(I(\mathscr{C}))(\neg A \mid B)$, and thus provides essentially the representation of the bound functions via (8). It is apparent, however, that this representation is highly redundant. For the lower bound, for instance, we always have the possible value 0, no matter how r and s are instantiated. Thus, we can simply put $L(r,s) :\equiv 0$. The upper bound function, too, can be simplified from the full representation (8). Here one finds that the maximum in (8) will always be attained at one of the vertices $\boldsymbol{v}_6, \boldsymbol{v}_7, \boldsymbol{v}_8$, so that vertices $\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3, \boldsymbol{v}_4, \boldsymbol{v}_5$ can be eliminated from the representation of $U(r,s)$. To turn the usually highly redundant representation (8) into a more succinct form will turn out to be the most difficult problem to handle in the general

j	$\boldsymbol{v}_j(\neg A \mid B)$	Parameter constraints
1	0	
2	0	
3	1	$r = 0, s = 0$
4	undefined	
5	1	$r = 0, s < 1$
6	$1 - r$	$r \geq s$
7	$1 - s$	$s \geq r$
8	$1 - r$	$r \neq 1, s \geq r, s < 1$

Table 4: Values of $P(\neg A \mid B)$

computation of the bound functions.

Turning now to the general procedure for computing $L(\dots)$ and $U(\dots)$ from the parameterized vertex list, let $P(\chi \mid \gamma)$ be the target probability of the desired inference rule. Let $\boldsymbol{v} : \boldsymbol{\pi}$ a parameterized vertex annotated with its p-constraints from the vertex list. To evaluate $\boldsymbol{v}(\chi \mid \gamma)$ we bring the expression

$$\frac{\sum_{k:\alpha_k \to \chi \wedge \gamma} v_k(r_1, \dots, r_N)}{\sum_{k:\alpha_k \to \gamma} v_k(r_1, \dots, r_N)} \tag{20}$$

into the form $p(r_1, \dots, r_N)/q(r_1, \dots, r_N)$ with polynomials p, q.

For a specific parameter instantiation I two conditions must be met for (20) to actually define the conditional probability of χ given γ at a vertex of $\Delta(I(\mathscr{C}))$: first, I must satisfy the p-constraints of $\boldsymbol{v}$, so that $I(\boldsymbol{v})$ is indeed a vertex of $\Delta(I(\mathscr{C}))$; second, $\sum_{k:\alpha_k \to \gamma} I(v_i) > 0$ must hold, so that the conditional probability of χ given γ is defined at the vertex $I(\boldsymbol{v})$. This latter condition can be expressed by yet another p-constraint π', which we append to the list $\boldsymbol{\pi}$ to obtain a new list $\boldsymbol{\pi}'$ of p-constraints. The preliminary definition (8) can now be brought into the form

$$L(r_1, \dots, r_N) := \mathit{min}[\boldsymbol{v}_1(\chi \mid \gamma) : \boldsymbol{\pi}'_1, \dots, \boldsymbol{v}_M(\chi \mid \gamma) : \boldsymbol{\pi}'_M] \tag{21}$$

$$U(r_1, \dots, r_N) := \mathit{max}[\boldsymbol{v}_1(\chi \mid \gamma) : \boldsymbol{\pi}'_1, \dots, \boldsymbol{v}_M(\chi \mid \gamma) : \boldsymbol{\pi}'_M] \tag{22}$$

with the semantics

$$I(L) = \mathit{min}\{I(\boldsymbol{v}_j(\chi \mid \gamma)) \mid 1 \leq j \leq M,\ I \text{ satisfies } \boldsymbol{\pi}'_j\}$$
$$I(U) = \mathit{max}\{I(\boldsymbol{v}_j(\chi \mid \gamma)) \mid 1 \leq j \leq M,\ I \text{ satisfies } \boldsymbol{\pi}'_j\}.$$

Note that Table 4 just gives the arguments of *min*[...], resp. *max*[...], in (21) and (22) for our example.

In principle, we now have solved the problem of deriving the inference rule for the given input constraints and the target conditional probability. However, this result is as yet unsatisfactory, because of the size of the representation of the bound functions. We will therefore now consider the problem of transforming $L(\dots), U(\dots)$ into a more manageable form. This will simply be done by deleting from the representation (21),(22) values that never define the optimal conditional probabilities.

Definition 4.1 Let $G \subseteq \{1, \dots, M\}$, $j \in G$. Value $\boldsymbol{v}_j(\chi \mid \gamma) : \boldsymbol{\pi}'_j$ is called *redundant for minimization (maximization)* in $\{\boldsymbol{v}_k(\chi \mid \gamma) : \boldsymbol{\pi}'_k \mid k \in G\}$ iff for every parameter instantiation I that satisfies $\boldsymbol{\pi}'_j$ there exists some $k \in G \setminus \{j\}$ such that I satisfies $\boldsymbol{\pi}'_k$, and $I(\boldsymbol{v}_j(\chi \mid \gamma)) \geq (\leq) I(\boldsymbol{v}_k(\chi \mid \gamma))$.

Our aim, now, will be to delete values that are redundant for minimization from (21) until a subset $\{\boldsymbol{v}_i(\chi \mid \gamma) : \boldsymbol{\pi}'_i \mid i \in G\}$ ($G \subseteq \{1, \dots, M\}$) without redundant values is found. Similarly, values redundant for maximization are to be deleted from (22). In our example we deleted from Table 4 all entries except the first for minimization, and the first five entries for maximization. Usually, there will exist more than one irredundant subset of the original value list. If, for instance, we begin with the list of the following five values

$$0 : \emptyset; \;\; 1 : \emptyset; \;\; 1 - r : r = 1; \;\; 1 - s : s = 1; \;\; 0 : r \neq 1, s \neq 1, \tag{23}$$

then both the subset consisting of the first value only, and the subset consisting of the last three values only are irredundant for minimization.

To compute irredundant representations for (21) and (22), a method is needed to decide whether a given value is redundant within a list of values. Many different heuristics can be used to determine redundancy in some cases (the simplest being that of checking for duplicates). A completely general decision procedure for redundancy, however, only seems to exist in the form of general decision procedures for the first-order theory of the real numbers (see [17] for a comprehensive treatment of this subject, also [8] for a recent exposition of the subject and its relevance to uncertain reasoning). To see why such a decision procedure can be used to decide redundancy, we express redundancy for minimization as stated in Definition 4.1 by the formula

$$\forall r_1 \dots \forall r_N (\boldsymbol{\pi}'_j \rightarrow \bigvee_{k \in G \setminus \{j\}} (\boldsymbol{\pi}'_k \wedge \boldsymbol{v}_k(\chi \mid \gamma) \leq \boldsymbol{v}_j(\chi \mid \gamma)). \tag{24}$$

As the expressions $\boldsymbol{\pi}$ and $\boldsymbol{v}(\chi \mid \gamma)$ that appear in this formula are (fractions of) polynomials, this formula can be rewritten as a (universal) sentence in pure first-order logic over the vocabulary

$\{0, 1, +, \cdot, \leq\}$. By classic results in model theory [19] the validity of such formulas, when interpreted over the real numbers, is decidable. For the special case of universal formulas, decision procedures exist with a time complexity essentially[3] of the form

$$m^N, \tag{25}$$

where m is the number of polynomial inequalities appearing in (24) [17].

Table 5 summarizes the second part of the algorithm, leading from the parameterized vertex list to the final representation of the bound functions. The algorithm calls subroutines `redundant_for_minimization` and `redundant_for_maximization`, which decide whether a given value is redundant in a given list of values. For simplicity, the algorithm of Table 5 checks the redundancy of values just in the order in which they appear in the original list. Applied to the list (23), for instance, this means that the algorithm will yield the last three values as an irredundant representation of the lower bound function, rather than the shorter representation given by the first value alone. In order to obtain smaller representations of the bound functions, one may modify the algorithm by checking for redundancy in a different order, e.g. by decreasing complexity of the expressions $\boldsymbol{v}_j(\chi \mid \gamma) : \boldsymbol{\pi}'_j$, thereby favoring simple expressions to be retained.

To obtain some bound on the complexity of the second part of the algorithm, we first have to find bounds on the complexity of the subroutines for redundancy checking. When these subroutines are implemented by a decision procedure for the universal first-order theory of the real numbers, this means that we have to find bounds on the number m of polynomial inequations appearing in (24). An examination of the procedure for generating the parameterized vertex list shows that the number of p-constraints in the $\boldsymbol{\pi}'_k$ is bounded by $2^K + N$, so that m is bounded by

$$|G|(2^K + N + 1). \tag{26}$$

In order to find a minimal irredundant subset of values, we have to check the redundancy of all M values, in the worst case always with respect to the full set $G = \{1, \ldots, M\}$. Combining (25) with (26), we find that a time bound for the computation of a minimal irredundant set of values is essentially of the form

$$M(M(2^K + N + 1))^N \approx M^{N+1}(2^K + N)^N. \tag{27}$$

[3]The exact bounds also depend on the maximal degrees of the polynomials and the maximal size of the coefficients in (24). This dependency appears to be non-critical for our application.

procedure generate_bound_functions

Input: parameterized vertex list $\boldsymbol{v}_j : \boldsymbol{\pi}_j \ (1 \leq j \leq M)$
target probability $P(\chi \mid \gamma)$
Output: irredundant representations of bound functions

1. Initialize: lower_bound_list $:= \emptyset$, upper_bound_list $:= \emptyset$
2. For $j = 1..M$
evaluate $\boldsymbol{v}_j(\chi \mid \gamma)$
generate $\boldsymbol{\pi}'_j$ by appending $\sum_{k:\alpha_k \to \gamma} v_{j,k} > 0$ to $\boldsymbol{\pi}_j$
append $\boldsymbol{v}_j(\chi \mid \gamma) : \boldsymbol{\pi}'_j$ to lower_bound_list
append $\boldsymbol{v}_j(\chi \mid \gamma) : \boldsymbol{\pi}'_j$ to upper_bound_list
3. For $j = 1..M$
If redundant_for_minimization $(\boldsymbol{v}_j(\chi \mid \gamma) : \boldsymbol{\pi}'_j$,lower_bound_list)
Then delete $\boldsymbol{v}_j(\chi \mid \gamma) : \boldsymbol{\pi}'_j$ from lower_bound_list
If redundant_for_maximization $(\boldsymbol{v}_j(\chi \mid \gamma) : \boldsymbol{\pi}'_j$,upper_bound_list)
Then delete $\boldsymbol{v}_j(\chi \mid \gamma) : \boldsymbol{\pi}'_j$ from upper_bound_list
4. $L(r_1, \ldots, r_N) := \mathit{min}$(lower_bound_list)
$U(r_1, \ldots, r_N) := \mathit{max}$(upper_bound_list)

Table 5: The second part of the algorithm

When we substitute for M the bound given in (19), the first of the two factors in (27) is seen to dominate the second, so that we obtain the bound

$$N^{(N+1)^2} 2^{K(N+1)^2}. \tag{28}$$

Clearly, this computation of an irredundant subset of values dominates the complexity of computing the initial set of values, so that (28) also expresses a bound on the overall complexity of the complete algorithm consisting of the generation of the parameterized vertex list and the subsequent computation of L and U.

5 An Example

In this section we demonstrate our general method by a second, slightly more complicated, example. We apply our method to derive an inference rule that was given as rule (v) by Frisch

and Haddawy [6] in the following form.

$$\frac{\begin{array}{l} P(\alpha \wedge \beta \mid \delta) \in [r,t] \\ P(\beta \mid \delta) \in [u,v] \end{array}}{P(\alpha \mid \beta \wedge \delta) \in [r/v, z]}$$

$$\text{where } z = \begin{cases} 1, & \text{if } t > u \\ 0, & \text{if } t = u = 0 \\ t/u, & \text{otherwise} \end{cases} \tag{29}$$

provided $r \leq v, v > 0$.

Here the applicability of the rule is constrained by the conditions $r \leq v$ and $v > 0$. We can easily extend the rule to also cover the degenerate cases $r > v$ or $v = 0$, and obtain the bound functions

$$L(r,t,u,v) = \begin{cases} 1 & \text{if } r > v \text{ or } v = 0 \\ r/v & \text{otherwise} \end{cases} \tag{30}$$

$$U(r,t,u,v) = \begin{cases} 1 & \text{if } t > u \text{ and } r \leq v \text{ and } v > 0 \\ 0 & \text{if } t = u = 0 \text{ or } r > v \text{ or } v = 0 \\ t/u & \text{otherwise} \end{cases} \tag{31}$$

(in agreement with the conventions of Lemma 2.1 we then obtain the bounds $P(\alpha \mid \beta \wedge \delta) \in [1,0] = \emptyset$ for parameter values that entail $P(\beta \wedge \delta) = 0$ or that lead to inconsistent premises).

To fit our general pattern (2), we have to express the premises of (29) by the four inequalities

$$P(A \wedge B \mid D) \geq r \tag{32}$$

$$P(A \wedge B \mid D) \leq t \tag{33}$$

$$P(B \mid D) \geq u \tag{34}$$

$$P(B \mid D) \leq v \tag{35}$$

Here α, β, δ that in [6] are meant to represent arbitrary propositional formulas have been replaced by simple propositional variables A, B, D. This changes the semantics of the rule only in an inessential way, and has no impact on the bound functions to be derived.

We number the atoms of $\mathscr{B}(A,B,D)$ as follows:

$$\alpha_1 \equiv ABD, \alpha_2 \equiv \bar{A}BD, \alpha_3 \equiv \bar{A}\bar{B}D, \alpha_4 \equiv A\bar{B}D, \ldots$$

(atoms with the conjunct $\bar{D}$ will not play any role), and can then write (32)-(35) in the algebraic

form (5) as

$$(1-r)x_1 - rx_2 - rx_3 - rx_4 \geq 0 \tag{36}$$

$$(1-t)x_1 - tx_2 - tx_3 - tx_4 \leq 0 \tag{37}$$

$$(1-u)x_1 - (1-u)x_2 - ux_3 - ux_4 \geq 0 \tag{38}$$

$$(1-v)x_1 - (1-v)x_2 - vx_3 - vx_4 \leq 0 \tag{39}$$

Our first task now is to generate the parameterized vertex list, i.e. we have to set up and solve the systems (14)-(16) of linear equations. The parameters of the given problem are $K = 3, N = 4$, so that our crude upper bound (19) for the number of systems to be solved is $4^5 2^{15} = 2^{25}$. Several simple observations enable us to reduce this number to 38. First, it follows from the fact that the equality forms of (36) and (37) are either linearly dependent (when $r = t$), or inconsistent (when $r \neq t$), that the system (14)-(16) has no unique solution when (16) contains both (36) and (37). Analogously for (38) and (39). In particular, at most two of the constraints (36)-(39) have to be used to generate all the solutions of (14)-(16), i.e. we have to set up the systems only for $d = 0, 1, 2$. A second useful observation is that eventually we are only interested in vertices at which the probability of $B \wedge D$ is positive. Given our numbering of variables, this means that we are looking for solutions of (14)-(16) with $x_1 + x_2 > 0$. A necessary condition for this to hold is that $H \cap \{1, 2\} \neq \emptyset$, where H is as in Lemma 3.2. Finally, we can exclude from our considerations systems defined by H with $H \cap \{5, 6, 7, 8\} \neq \emptyset$. This is because the variables $x_5, \ldots, x_8$ (representing the probabilities of atoms with negated D) do not appear in (36)-(39), so that a unique solution of (14)-(16) defined by such a H will simply be $x_i = 1, x_j = 0$ $(j \neq i)$ for some $i \in \{5, 6, 7, 8\}$. These solutions, again, are uninteresting because then $x_1 + x_2 = 0$.

In summary, we have to set up and solve system (14)-(16) for $d = 0, 1, 2$; for $H \subseteq \{1, \ldots, 4\}$ with $|H| = d + 1$, $H \cap \{1, 2\} \neq \emptyset$; and for all selections of constraints from (36)-(39) that do not contain both (36) and (37), or both (38) and (39). Note that analogous simplifications will be possible in the computation of other inference rules as well. We are left with 38 systems of equations, 24 of which possess unique solutions, listed in Table 6. Columns 2 and 3 give the parameters that define a particular system, column 4 their unique solution (blank spaces represent 0-entries; the assignments $v_5 = v_6 = v_7 = v_8 = 0$ are common to all these solutions, and therefore omitted). Column 5 gives the p-constraints for the solutions. Not listed in this column are the p-constraints $\boldsymbol{\rho} :\equiv r \leq t, u \leq v, r \leq v$, which are generated for every

vertex (except in some cases where stronger constraints like $t = 1$ or $r \leq u$ are generated).

The first part of the algorithm thus leaves us with 24 parameterized vertices $\boldsymbol{v}$ at which we have to evaluate the target probability $\boldsymbol{v}(\chi \mid \gamma)$, i.e. the quotient $x_1/(x_1 + x_2)$. The results of this evaluation are shown in column 6. Column 7 states the additional p-constraint π' for the denominator to be positive. It remains to bring the solution now found into a more compact, useful, form by deleting redundant values. Minimal irredundant sets of values for minimization and maximization of $P(A \mid B \wedge D)$ are indicated by the +-marks in the columns 8 and 9, respectively. The final bound functions we obtain now are

$$L(r,t,u,v) = \mathit{min}[r/v : \boldsymbol{\rho}, v > 0] \tag{40}$$

$$\begin{aligned} U(r,t,u,v) = \mathit{max}[&0 : r = 0, v = 1; \\ &1 : \boldsymbol{\rho}, u \leq r, r > 0; \\ &1 : \boldsymbol{\rho}, u \leq t, t \leq v, t > 0; \\ &1 : \boldsymbol{\rho}, r \leq u, u \leq t, u > 0; \\ &1 : \boldsymbol{\rho}, v \leq t, v > 0; \\ &t/u : \boldsymbol{\rho}, t \leq u, u > 0]. \end{aligned} \tag{41}$$

where the p-constraints $\boldsymbol{\rho}$ suppressed in Table 1 have been reinstated. Remembering the conventions $\mathit{min}\,\emptyset = 1,\ \mathit{max}\,\emptyset = 0$, and taking into account that the conditions $r \leq t, u \leq v$ are taken for granted in (29), these functions can be seen to be the same as (30) and (31).

The bound functions (40) and (41) here were derived essentially by a faithful (manual) execution of the general algorithm given in Tables 2 and 5. In two places, however, we did not follow the algorithm to the letter: first, initially we analyzed the given problem a little more closely in order to reduce the number of systems of linear equations we had to solve. This analysis is applicable in general, and the substantial reductions it gives rise to in most cases can easily be integrated into the first part of the algorithm. Second, the computation of minimal irredundant value sets in this example was not done by checking values for redundancy in the order in which they appear in the list, or, indeed, by following any other strictly mechanical procedure. It may very well be the case, therefore, that every reasonably simple fully automatic process for the computation of minimal irredundant value sets (given by some prescribed order in which values are checked for redundancy) will yield a representation of the bound functions that is somewhat different from the one given by (40) and (41).

1	2	3	4				5	6	7	8	9
j	H	constraints	$v_{j,1}$	$v_{j,2}$	$v_{j,3}$	$v_{j,4}$	$\boldsymbol{\pi}_j$	$\boldsymbol{v}_j(\chi \mid \gamma)$	π'_j	L	U
1	1	$\emptyset$	1				$t = 1, v = 1$	1			
2	2	$\emptyset$		1			$r = 0, v = 1$	0			+
3	1, 2	(36)	r	$1 - r$			$v = 1$	r			
4	1, 2	(37)	t	$1 - t$			$v = 1$	t			
5	1, 3	(36)	r		$1 - r$		$u \leq r$	1	$r > 0$		+
6	1, 3	(37)	t		$1 - t$		$u \leq t, t \leq v$	1	$t > 0$		+
7	1, 3	(38)	u		$1 - u$		$r \leq u, u \leq t$	1	$u > 0$		+
8	1, 3	(39)	v		$1 - v$		$v \leq t$	1	$v > 0$		+
9	1, 4	(36)	r			$1 - r$	$u \leq r$	1	$r > 0$		
10	1, 4	(37)	t			$1 - t$	$u \leq t, t \leq v$	1	$t > 0$		
11	1, 4	(38)	u			$1 - u$	$r \leq u, u \leq t$	1	$u > 0$		
12	1, 4	(39)	v			$1 - v$	$v \leq t$	1	$v > 0$		
13	2, 3	(38)		u	$1 - u$		$r = 0$	0	$u > 0$		
14	2, 3	(39)		v	$1 - v$		$r = 0$	0	$v > 0$		
15	2, 4	(38)		u		$1 - u$	$r = 0$	0	$u > 0$		
16	2, 4	(39)		v		$1 - v$	$r = 0$	0	$v > 0$		
17	1, 2, 3	(36), (38)	r	$u - r$	$1 - u$		$r \leq u$	r/u	$u > 0$		
18	1, 2, 3	(36), (39)	r	$v - r$	$1 - v$			r/v	$v > 0$	+	
19	1, 2, 3	(37), (38)	t	$u - t$	$1 - u$		$t \leq u$	t/u	$u > 0$		+
20	1, 2, 3	(37), (39)	t	$v - t$	$1 - v$		$t \leq v$	t/v	$v > 0$		
21	1, 2, 4	(36), (38)	r	$u - r$		$1 - u$	$r \leq u$	r/u	$u > 0$		
22	1, 2, 4	(36), (39)	r	$v - r$		$1 - v$		r/v	$v > 0$		
23	1, 2, 4	(37), (38)	t	$u - t$		$1 - u$	$t \leq u$	t/u	$u > 0$		
24	1, 2, 4	(37), (39)	t	$v - t$		$1 - v$	$t \leq v$	t/v	$v > 0$		

Table 6: Example

6 Conclusion

We have described an algorithm for the automatic derivation of probabilistic inference rules. The general form (2) of probabilistic inference rules we have considered covers most of the rules studied in the literature. An exception are rule schemata that have a variable number of input constraints (rule (viii) in [6]).

The approach presented here is based on previous work by Hailperin [9, 10, 11]. It extends the work of Hailperin in that it provides for a general method for conditional probabilities in the premise and the target probability. Hailperin presented his method in general form only for unconditional probabilities. This case is substantially simpler, because most of the symbolic calculations that we have to perform in our algorithm can be avoided. It should be noted, however, that several of the extensions that we presented are already implicit in some examples Hailperin [11] gives for inference rules with conditional probabilities. New elements in the present paper are the general strategy of optimization following Lemma 2.1, the concept of

p-constraints, and the final step of algorithmically eliminating redundant values.

In spite of the disheartening bound (28), computational complexity may not be a major obstacle for the applicability of the given method: first, the algorithm is mainly intended for relatively small inputs, because interesting probabilistic inference rules usually are of a relatively simple structure. Of the 32 different inference rules presented in [6], for instance, all but two have $K \leq 4$ and $N \leq 6$. Second, it must be born in mind that the algorithm is not to be used at run time in an application system, but only as a design tool for probabilistic inference systems. Here it will not matter much when the algorithm takes a day or two to obtain a result, if this saves the developer an at least equal amount of tedious manual computations.

References

[1] S. Amarger, D. Dubois, and H. Prade. Constraint propagation with imprecise conditional probabilities. In *Proceedings of the Seventh Annual Conference on Uncertainty in Artificial Intelligence (UAI–91)*, pages 26–34, San Mateo, California, 1991. Morgan Kaufmann Publishers.

[2] G. Boole. *Investigations of Laws of Thought on which are Founded the Mathematical Theories of Logic and Probabilities*. London, 1854.

[3] G. Boole. On the theory of probabilities. *Philosophical Transactions of the Royal Society of London*, 152, 1862.

[4] D. Dubois, H. Prade, and J.-M. Toucas. Inference with imprecise numerical quantifiers. In Z. Ras and M. Zemankova, editors, *Intelligent Systems: State of the Art and Future Directions*, pages 52–72. Ellis Horwood, 1990.

[5] R. Fagin, J. Y. Halpern, and N. Megiddo. A logic for reasoning about probabilities. *Information and Computation*, 87:78–128, 1990.

[6] A.M. Frisch and P. Haddawy. Anytime deduction for probabilistic logic. *Artificial Intelligence*, 69:93–122, 1994.

[7] K. O. Geddes, S.R. Czapor, and G. Labahn. *Algorithms for Computer Algebra*. Kluwer Academic Publishers, 1992.

[8] D. Geiger and C. Meek. Quantifier elimination for statistical problems. In *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–99)*, pages 226–235, San Francisco, CA, 1999. Morgan Kaufmann Publishers.

[9] T. Hailperin. Best possible inequalities for the probability of a logical function of events. *American Mathematical Monthly*, 72:343–359, 1965.

[10] T. Hailperin. *Boole's Logic and Probability*, volume 85 of *Studies in Logic and the Foundations of Mathematics*. North-Holland, 1976.

[11] T. Hailperin. *Sentential Probability Logic*. Lehigh University Press, Bethlehem, 1996.

[12] J. Heinsohn. Probabilistic description logics. In *Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–94)*, pages 311–318, Seattle, WA, 1994.

[13] C. Luao, C. Yu, J. Lobo, G. Wang, and T. Pham. Computation of best bounds of probabilities from uncertain data. *Computational Intelligence*, 12(4):541–566, 1996.

[14] T. Lukasiewicz. Local probabilistic deduction from taxonomic and probabilistic knowledge-bases over conjunctive events. *International Journal of Approximate Reasoning*, 21:23–61, 1999.

[15] N. Nilsson. Probabilistic logic. *Artificial Intelligence*, 28:71–88, 1986.

[16] G. Paaß. Probabilistic logic. In D. Dubois, P. Smets, A. Mamdani, and H. Prade, editors, *Non-Standard Logics for Automated Reasoning*, chapter 8, pages 213–251. Academic Press, 1988.

[17] J. Renegar. On the computational complexity and geometry of the first-order theory of the reals. parts i-iii. *Journal of Symbolic Computation*, 13:255–352, 1992.

[18] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, 1986.

[19] A. Tarski. *A Decision Method for Elementary Algebra and Geometry*. University of California Press, 1951.

[20] H. Thöne, U. Güntzer, and W. Kießling. Towards precision of probabilistic bounds propagation. In *Proceedings of the Eighth Annual Conference on Uncertainty in Artificial*

Intelligence (UAI–92), pages 315–322, San Francisco, CA, 1992. Morgan Kaufmann Publishers.