Interval-Type and Affine Arithmetic-Type Techniques for Handling Uncertainty in Expert Systems, with Applications to Geoinformatics and Computer Security

Martine Ceberio[1], Vladik Kreinovich[1], Sanjeev Chopra[1], Luc Longpré[1], Hung T. Nguyen[2], Bertram Ludäscher[3], and Chitta Baral[4]

[1]NASA Pan-American Center for Earth and Environmental Studies
University of Texas at El Paso
El Paso, TX 79968, USA
contact email vladik@cs.utep.edu
[2]Math. Dept., New Mexico State University
Las Cruces, NM 88003, USA
[3]San Diego Supercomputer Center
University of California at San Diego
9500 Gilman Drive, La Jolla, CA 92093-0505, USA
[4]Department of Computer Science & Engineering
Arizona State University, Tempe, AZ 85287-5406, USA

Abstract

Expert knowledge consists of statements S_j (facts and rules). The expert's degree of confidence in each statement S_j can be described as a (subjective) probability (some probabilities are known to be independent). Examples: if we are interested in oil, we should look at seismic data (confidence 90%); a bank A trusts a client B, so if we trust A, we should trust B too (confidence 99%). If a query Q is deducible from facts and rules, what is our confidence $p(Q)$ in Q? We can describe Q as a propositional formula F in terms of S_j; computing $p(Q)$ exactly is NP-hard, so heuristics are needed.

Traditionally, expert systems use technique similar to straightforward interval computations: we parse F and replace each computation step with corresponding probability operation. Problem: at each step, we ignore the dependence between the intermediate results F_j; hence intervals are too wide. Example: the estimate for $P(A \vee \neg A)$ is not 1. Solution: similarly to affine arithmetic, besides $P(F_j)$, we also compute $P(F_j \,\&\, F_i)$ (or $P(F_{j_1} \,\&\, \ldots \,\&\, F_{j_k})$), and on each step, use all combinations of l such probabilities to get new estimates. Results: e.g., $P(A \vee \neg A)$ is estimated as 1.

1 Formulation of the Problem

Expert knowledge usually consists of statements S_j: facts and rules. The main objective is, given a query Q, to check whether Q follows from the expert knowledge. For example, in the knowledge base

$$S_1 : a \leftarrow b. \quad S_2 : b \leftarrow . \quad S_3 : a \leftarrow c. \quad S_4 : c \leftarrow .$$

S_1 and S_3 are rules, and S_2 and S_4 are facts. If we ask a query $Q = a?$, then the answer is "yes": e.g., Q follows from S_1 and S_2. Prolog-type inference engines are tools that provide such inference; see, e.g., [9].

The problem with this approach is that experts are often not 100% confident in their statements. One way to describe the expert's degree of confidence in each statement S_j is by a (subjective) probability $p(S_j)$. A subjective probability of a statement can be defined, e.g., as a value p for which, to the expert, a lottery in which she gets \$100 with probability p is equivalent to the lottery in which she gets \$100 if S_j is true. This value can be determined by using bisection: we start with an interval $[p^-, p^+] = [0, 1]$ for which "\$100 with probability p^-" is less preferable than "\$100 if S_j", which is, in its turn, less preferable than "\$100 with probability p^+". By definition of a subjective probability p, this means that $p \in [p^-, p^+]$.

Once we have such an interval, we compute its midpoint p_m and compare the corresponding lottery with "\$100 if S_j". Depending on the result of this comparison, we get a half-size interval ($[p^-, p_m]$ or $[p_m, p^+]$) that contains p. Hence, in k steps, we can compute the interval of width 2^{-k} containing p. The midpoint of this interval is thus a $2^{-(k+1)}$-approximation to p.

The question is: if a query Q is deducible from facts and rules, what is our confidence $p(Q)$ in Q? For example, to find oil, we must look for certain subterranean structures (confidence 80%); to find these structures, we must analyze gravity data (confidence 90%). What is our confidence that to find oil, we must analyze gravity data?

Let us describe this problem in precise terms. We can usually describe deducibility of Q as a propositional formula F in terms of S_j. For example, for the above knowledge base, for Q to be true, either S_1 and S_2 must be true, or S_3 and S_4 must be true. In this case, $F = (S_1 \,\&\, S_2) \vee (S_3 \,\&\, S_4)$. The general algorithm for describing such a propositional formula is given in [6].

As a result, we arrive at the following problem:

- we have a propositional combination F of known statements S_j;
- we know the probabilities $p(S_j)$ of different statements;
- we must determine the probability $p(F)$.

Since the events S_j may be statistically dependent, we may get different values for $p(F)$ depending on whether the values are independent or, say, positively correlated. So, to be more precise:

- we must determine the interval $\mathbf{p}(F)$ of possible values of $p(F)$.

How is this problem solved now?

2 Traditional Approach

It is known that in general, the problem of finding the exact bounds for $p(F)$ is NP-hard; see, e.g., [9]. This problem is NP-hard even if all the probabilities $p(S_j)$ are equal to 1 – because it is equivalent to the propositional satisfiability problem, a known NP-hard problem.

Traditionally, expert systems use technique similar to straightforward interval computations [5]. Namely, for simple formulas we know the corresponding probability bounds [11]: if we know the bounds $[\underline{a}, \overline{a}]$ for $p(A)$ and $[\underline{b}, \overline{b}]$ for $p(B)$, then:

- $p(\neg A)$ is in the interval $[1-\overline{a}, 1-\underline{a}]$;
- $p(A \,\&\, B)$ is in the interval $[\max(\underline{a}+\underline{b}-1, 0), \min(\overline{a}, \overline{b})]$;
- $p(A \vee B)$ is in the interval $[\max(\underline{a}, \underline{b}), \min(\overline{a}+\overline{b}, 1)]$.

In the general case, we parse F and replace each computation step with the corresponding probability operation.

For example, let $F = (A \,\&\, B) \vee (A \,\&\, \neg B)$ and $p(A) = p(B) = 0.6$. The compiler would start with $F_1 = A$ and $F_2 = B$, then it would compute $F_3 = \neg B$, $F_4 = F_1 \,\&\, F_2$, $F_5 = F_1 \,\&\, F_3$, and finally $F = F_4 \vee F_5$. Thus, according to the above procedure, we first find the bounds for $p(F_3) = p(\neg B)$, then for $p(F_4) = p(A \,\&\, B)$ and $p(F_5) = p(A \,\&\, \neg B)$, and finally, the bounds for $p(F)$. As a result, we get $p(\neg B) = 1 - 0.6 = 0.4$,

$$\mathbf{p}(A \,\&\, B) = [\max(0.6+0.6-1, 0), \min(0.6, 0.6)] = [0.2, 0.6],$$

$$\mathbf{p}(A \,\&\, \neg B) = [\max(0.6+0.4-1, 0), \min(0.6, 0.4)] = [0, 0.4],$$

$$\mathbf{p}(F) = [\max(0, 0.2), \min(0.4+0.6, 1)] = [0.2, 1.0].$$

In this problem, F is equivalent to A, so $p(F) = 0.6$. Thus, similarly to interval computations, we can see that the resulting interval contained excess width.

3 Main Idea

In interval computations, one way to decrease the excess width is to use affine or Taylor arithmetic; see, e.g., [2, 3, 7]. For example, for affine arithmetic, the main idea is that for each intermediate result y_j, we not only keep the interval of its possible values, we also keep the relation between this value and the original inputs – in the form of a linear dependence $y_j = a_{0j} + a_{1j} \cdot x_1 + \ldots + a_{nj} \cdot x_n$.

For quadratic Taylor models, we also keep the relation between y_j and pairs of inputs (as terms $a_{jkl} \cdot x_k \cdot x_l$), etc.

Our main idea is that, similarly to affine and Taylor arithmetic, for each intermediate result F_j, besides an interval of possible values for $p(F_j)$, we also compute intervals of possible values for pairs: $p(F_j \,\&\, F_i)$, $p(F_j \vee F_i)$ for all previous expressions F_i and for all possible propositional functions of two variables. On each step, we use all such probabilities to get new estimates.

If this is not enough, we use an analog of k-th order Taylor methods – estimate intervals for propositional combinations $p(F_{j_1} \,\&\, \ldots \,\&\, F_{j_{k+1}})$ of $k+1$ variables. The higher the order k, the more accurate the results, but the longer the computations (since we must compute more terms).

4 Technical Details

A (minor) problem with the above idea is that it is not clear how, when we know, say, the probabilities for all triples of F_i for $i \le j$, we can estimate the new probabilities involving F_{j+1}. To transform this idea into a computationally efficient approach, we introduce, in addition to the parameter k, another parameter l such that, when estimating interval for $p(F_i \,\&\, \ldots)$, we take into account only $\le l$ known probability bounds, and then take the intersection of the resulting intervals corresponding to all sets of l probability bounds.

One we have such a restriction, we can estimate the new probabilities as follows. For each new result $F_{j+1} = F_{i_1} \oplus F_{i_2}$, we must estimate the probabilities of all combinations $f(F_{j+1}, F_{j_1}, \ldots, F_{j_{k-1}})$, i.e., combinations

$$f(F_{i_1} \oplus F_{i_2}, F_{j_1}, \ldots, F_{j_{k-1}})$$

involving $k+1$ formulas F_i. We assume that we know the probability intervals for l such combinations of $\le k$ values F_i. Between these $l+1$ propositional combinations, we involve $m \le (k+1) + l \cdot k$ variables $F_{k_1}, \ldots, F_{k_m}$. We can describe both known and estimated probabilities as sums of probabilities of atomic statements $F_{i_1}^{\varepsilon_1} \,\&\, \ldots \,\&\, F_{i_m}^{\varepsilon_m}$, where $\varepsilon \in \{-,+\}$, F^+ means F, and F^- means $\neg F$. Then, we use linear programming (LP) to get desired bounds on the unknown probability.

What is the running time of this algorithm? A propositional function $f(x_1, \ldots, x_k)$ of k propositional variables can be described as a function from the set $\{0,1\}^k$ of 2^k possible combinations x_i to the set $\{0,1\}$ of possible truth values. Thus, there are exactly 2^{2^k} such functions. For fixed k and l, this means that we have $O(1)$ such functions.

One the j-th step, we have j intermediate results $F_1, \ldots, F_j$. We have $O(j^k)$ possible combinations of $\le k$ such values, so we have $O(j^k)$ probability bounds. To compute each of $O(j^k)$ new bounds, we consider all possible subsets of l probabilities. There are $O((j^k)^l) = O(j^{k\cdot l})$ such subsets. For each subset, for fixed k and l, the value m is bounded by a constant: $m = O(1)$. There are $2^m = O(1)$ possible combinations, so each LP requires $O(1)$ time. So, overall, on step j, we need $O(j^k) \cdot O(j^{k\cdot l}) \le M \cdot j^{k\cdot(l+1)}$ steps for some constant M. Overall, we need $\le M(1^{k\cdot(l+1)} + \ldots + n^{k\cdot(l+1)})$ steps, where the number n of parsing steps is bounded by the length of the formula F. It is known that $1^a + 2^a + \ldots + n^a = O(n^{a+1})$, so overall, this algorithm requires $O(n^{k\cdot(l+1)+1})$ steps. In other words, the running time grows polynomially with the length of the formula F – so this algorithm is feasible.

It is worth mentioning that when $k \to \infty$ and $l \to \infty$, we get exact results; however, computation time grows exponentially with k and l, so we cannot realistically use too large values k and l.

5 Examples

Let us first illustrate the above algorithm on the example when we already know the analytical solution: we know $p(A) = a = 0.6$ and $p(B) = b = 0.6$ and we want to estimate $p(A \vee B)$. Here, we describe the following probabilities of atomic statements: $p_{++} = p(A \,\&\, B)$, $p_{+-} = p(A \,\&\, \neg B)$, $p_{-+} = p(\neg A \,\&\, B)$, $p_{--} = p(\neg A \,\&\, \neg B)$. In this problem, $p(A \vee B) = p(A \,\&\, B) + p(A \,\&\, \neg B) + p(\neg A \,\&\, B)$, so the corresponding LP problem takes the following form: $p_{++} + p_{+-} + p_{-+} \to \min(\max)$ under the conditions:

$$p_{++} + p_{+-} = a; \quad p_{++} + p_{-+} = b; \quad p_{++} + p_{+-} + p_{-+} + p_{--} = 1;$$

$$p_{++} \ge 0; \quad p_{+-} \ge 0; \quad p_{-+} \ge 0; \quad p_{--} \ge 0.$$

It is known that in general, the solution of a LP problem is attained at one of the vertices of the corresponding set, i.e., when the largest possible number of inequalities become equalities. In this particular case, $p(A \vee B)$ is the smallest when $p_{-+} = 0$, and $p(A \vee B)$ is the largest when $p_{--} = 0$. In both cases, we get the desired bounds $\max(a + b - 1, 0) = 0.2$ and $\min(a, b) = 0.6$.

The second example shows that we indeed get narrowed intervals. The problem is to estimate $p(A \vee \neg A)$ for $p(A) = 0.6$. The desired answer is, of course, $p(A \vee \neg A) = 1$. However, when parsing, we get $F_1 = A$, $F_2 = \neg A$, and $F = F_1 \vee F_2$. So, in the traditional approach, we estimate $p(F_1) = 0.6$, $p(F_2) = 1 - p(F_1) = 1 - 0.6 = 0.4$, and

$$\mathbf{p}(F_1 \vee F_2) = [\max(0.4, 0.6), \min(0.4 + 0.6, 1)] = [0.4, 1]$$

– excess width. In the new approach, in addition to estimating $p(F_2) = 0.4$, we also use the relation $F_2 = \neg F_1$ to estimate probabilities of other binary combinations, in particular, $p(F_1 \,\&\, F_2) = 0$, $p(F_1 \,\&\, \neg F_2) = 0.6$, $p(\neg F_1 \,\&\, F_2) = 0.4$, $p(F_1 \vee F_2) = 1$, $p(F_1 \vee \neg F_2) = 0.6$, $p(\neg F_1 \vee F_2) = 0.4$, and $p(\neg F_1 \vee \neg F_2) = 1$. Based on these estimates, for $F = F_1 \vee F_2$, we get $p(F_1 \vee F_2) = 1.0$. As a result, we get the exact desired probability, with no excess width.

Third example: for $(A \,\&\, B) \vee (A \,\&\, \neg B)$, the traditional method leads to excess width in comparison with A; however, if we use triples (analogue of quadratic Taylor approximations), then we can estimate the probability of $(A \,\&\, B) \vee (A \,\&\, \neg B)$ as $p(A)$.

Similarly, for $(A \,\&\, B) \vee (A \,\&\, C)$, the traditional method leads to excess width in comparison with $A \vee (B \,\&\, C)$; if we use higher-order methods, we get the exact interval for $p((A \,\&\, B) \vee (A \,\&\, C))$ – i.e., we get *distributivity.*

Comment. A general argument against expert systems and fuzzy logic (see, e.g., [8]) is that, e.g., $p(A \vee \neg A)$ is estimated as $f(p(A), p(\neg A))$ – e.g., as $\max(p(A), p(\neg A))$, while the correct value of $p(A \vee \neg A)$ is 1. Our solution: in addition to probabilities of individual intermediate statements, keep probabilities of pairs, triples, etc.

6 Case Study: Computer Security

In the traditional approach to trust, we either trust an agent or not. If we trust an agent, we allow this agent full access to a particular task. For example, I trust my bank to handle my account; the bank (my agent) outsources money operations to another company (sub-agent), etc. The problem with this approach is that I may have only 99.9% trust in bank, bank in its contractor, etc. As a result, for long chains, the probability of a security leak may increase beyond any give threshold. To resolve this problem, we must keep track of trust probabilities.

Let us describe this idea in precise terms. We have a finite set A; its elements are called *agents.* For some pairs (a, b) of agents, we know the probability $p_0(a, b)$ with which a directly trusts b. Our objective is to describe, for given two agents f and s, the probability $p_t(f, s)$ with which the agent f trusts the agent s.

In graph terms, we have each edge (a, b) with probability $p_0(a, b)$. We must find the probability $p_t(f, s)$ that there is a path from f to s: $f \xrightarrow{E} s$. Since we usually have no information on the dependence between different direct trust links, we should find the interval of possible values of $p_t(f, s)$. Specifically, it is sufficient to estimate the worst-case (smallest) probability of trust $\underline{p}_t(f, s)$.

In precise terms, we know a graph (A, E), and we know the values $p_0(a, b)$ for all $(a, b) \in E$. We consider all possible probability distributions $p(E')$ on the set of all subgraphs $E' \subseteq E$ for which, for every $(a, b) \in E$, we have

$\sum\limits_{E':(a,b)\in E'} p(E') = p_0(a,b)$. For every two edges f and s, $p_t(f,s) \stackrel{\text{def}}{=} \sum\limits_{E':f\xrightarrow{E'}s} p(E')$. We define $\underline{p}_t(f,s)$ as the exact lower bound of all such values $p_t(f,s)$. Our objective is to compute $\underline{p}_t(f,s)$.

For this problem, we have an explicit solution. Namely, let us define the *length* (distrust) of an edge is defined as $d_0(a,b) \stackrel{\text{def}}{=} 1 - p_0(a,b)$. The *length* $\ell(\gamma)$ of a path $\gamma = (a_0,\ldots,a_n)$ is defined as usual: $\ell(\gamma) \stackrel{\text{def}}{=} \sum\limits_{i=0}^{n-1} d_0(a_i,a_{i+1})$. The length of the shortest path from f to s is defined as follows:

$$\underline{d}_t(f,s) \stackrel{\text{def}}{=} \min\{\ell(\gamma)\,|\,\gamma \text{ is a path from } f \text{ to } s\}.$$

Proposition. $\underline{p}_t(f,s) = \max(1 - \underline{d}_t(f,s), 0)$.

So, we can use Dijkstra's algorithm (see, e.g., [1]) to find the shortest path in a graph, and then compute $\underline{p}_t(f,s)$.

Proof. Let $p(E')$ be consistent with the given information. We want to prove that $d_t(f,s) \le \underline{d}_t(f,s)$, where $d_t(f,s) \stackrel{\text{def}}{=} 1 - p_t(f,s)$.

Let $\gamma_0 = (a_0,a_1,\ldots,a_n)$ be the shortest path from $a_0 = f$ to $a_n = s$; then, $\underline{d}_t(f,s) = d_0(a_0,a_1)+\ldots+d_0(a_{n-1},a_n)$. If there is no path from f to s ($N_t(f,s)$), then at least one of the connections (a_i,a_{i+1}) is not present in E' ($N_0(a_i,a_{i+1})$):

$$N_t(f,s) \supset (N_0(a_0,a_1) \vee \ldots \vee N_0(a_{n-1},a_n)).$$

Hence, $d_t(f,s) \le P(N_0(a_0,a_1) \vee \ldots \vee N_0(a_{n-1},a_n))$. So, $d_t(f,s) \le d_0(a_0,a_1) + \ldots + d_0(a_{n-1},a_n) = \underline{d}_t(f,s)$.

To complete the proof, we produce a distribution $p(E')$ for which $p_t(f,s) \le \max(1 - \underline{d}_t(f,s), 0)$. Let $\pi(x) \stackrel{\text{def}}{=} x - \lfloor x \rfloor$. We define $E(\omega)$ for $\omega = U([0,1])$ as follows: For every $(a,b) \in E$, this edge is in $E(\omega)$ if and only if $\omega \notin \pi(I(a,b))$, where $I(a,b) \stackrel{\text{def}}{=} [\underline{d}_t(f,a), \underline{d}_t(f,a) + d_0(a,b)]$. Since $\pi(I(a,b))$ has width $p_0(a,b)$, the distribution $p(E')$ is consistent with $p_0(a,b)$.

Induction proves (see [4] for details) that for every path starting at $a_0 = f$, if all its edges (a_i,a_{i+1}) are in $E(\omega)$, then $\omega \ge \underline{d}_t(a_0,a_n)$. Hence,

$$p_t(f,s) \le \max(1 - \underline{d}_t(f,s), 0).$$

The statement is proven, so the above algorithm has been justified.

Acknowledgments

This work was supported in part by NASA under cooperative agreement NCC5-209, by NSF grants EAR-0112968, EAR-0225670, and EIA-0321328, by Army Research Laboratories grant DATM-05-02-C-0046, by NIH grant 3T34GM008048-20S1, and by Applied Biomathematics.

The authors are thankful to all the participants of SCAN'04 for valuable discussions.

References

[1] Th. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press, Cambridge, MA, 2001.

[2] L. H. de Figueiredo and J. Stolfi, *Self-Validated Numerical Methods and Applications*, Brazilian Mathematics Colloquium monograph, IMPA, Rio de Janeiro, Brazil, 1997.

[3] J. Hoefkens, M. Berz, and K. Makino, "Controlling the Wrapping Effect in the Solution of ODEs for Asteroids", *Reliable Computing*, 1999, Vol. 9, No. 1, pp. 21–41.

[4] P. Jaksurat, E. Freudenthal, M. Ceberio, and Vladik Kreinovich, "Probabilistic Approach to Trust: Ideas, Algorithms, and Simulations", *Proceedings of the Fifth International Conference on Intelligent Technologies InTech'04*, Houston, Texas, December 2–4, 2004.

[5] L. Jaulin, M. Kieffer, O. Didrit, and E. Walter, *Applied Interval Analysis: With Examples in Parameter and State Estimation, Robust Control and Robotics*, Springer, London, 2001.

[6] F. Lin and J. Zhao, "On tight logic programs and yet another translation from normal logic programs and propositional logic", *Proc. AAAI'03*, 2003, pp. 853–864.

[7] A. Neumaier, "Taylor Forms – Use and Limits", *Reliable Computing*, 2003, Vol. 9, No. 1, pp. 43–79.

[8] H. T. Nguyen and E. A. Walker, *First Course in Fuzzy Logic*, CRC Press, Boca Raton, Florida, 1999.

[9] S. Russell and P. Norvig, *Artificial Intelligence: a modern approach*, Prentice Hall, Englewood Cliffs, New Jersey, 2002.

[10] R. Trejo, V. Kreinovich, and C. Baral, "Towards Feasible Approach to Plan Checking Under Probabilistic Uncertainty: Interval Methods", *Proc. of the 17th National Conference on Artificial Intelligence AAAI'2000*, Austin, TX, July 30–August 3, 2000, pp. 545–550.

[11] P. Walley, *Statistical Reasoning with Imprecise Probabilities*, Chapman & Hall, N.Y., 1991.