Reasoning with Conditional Probabilities
NumQuant : a linear programming based method.

Stéphane AMARGER[1, 2], Richard EPENOY and Stéphane GRIHON

Institut de Recherche en Informatique de Toulouse
Université Paul Sabatier
118, route de Narbonne
31062 TOULOUSE Cedex
FRANCE

Abstract

In this paper, starting from the problem of reasoning with conditional probabilities, we expose a mathematical programming based method. We show that this method is not really efficient and, for a certain kind of problems, gives too imprecise results. Then we propose an exact method and finally we compare the efficiencies of these two methods.

Keywords

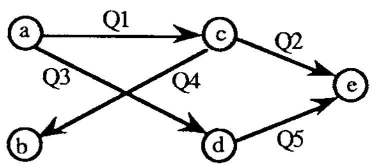
conditional probabilities, mathematical programming, linear programming, reasoning

1. Introduction

The problem of reasoning with probabilities was studied by Nilsson in [NILSSON 1986] handling the probability of the material implication, i.e. P(a → b).

In [PEARL 1988], Pearl reasoned with conditional probabilities, i.e. the material implication (a → b) is viewed as the conditional event (b | a).

To reason with conditional probabilities, Pearl used Bayesian networks ([PEARL 1988]),



where Q1 is P(c | a), Q2 is P(e | c), and so on.

In such networks, loops (undirected cycles, e.g., <a, c, e, d>) are allowed, but cycles are prohibited.

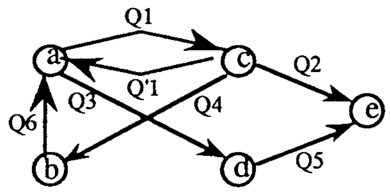
1 This work was partially supported by the European ESPRIT Basic Research Action number 3085, entitled "Defeasible Reasoning and Uncertainty Management Systems" (DRUMS)

2 For more information, contact Stéphane AMARGER by p-mail at this address, or by e-mail at amarger@irit.fr, or, by voice-mail at +33 61 55 66 11 extension 73 66.

Reasoning with conditional probabilities

But what we want to do is to reason with conditional probabilities viewed as a generalization of numerical quantifiers ([DUBOIS, PRADE 1988]); and we then need to handle cycles ([DUBOIS, PRADE, TOUCAS 1989]).

So, we represent graphically the problem, using an inference network, which generalize the notion of Bayesian network, because cycles are allowed, e.g.,



where the arc between "a" and "c" and valued "Q1" means that "Q1 a's are c's," i.e. Q1 elements of the set "a" are in the set "c."

In such a framework, Q1 is called a numerical quantifier and is viewed as a constraint acting on the cardinality of "c" relative to "a," i.e.,

$$\frac{|c \cap a|}{|a|} \in Q1 \subseteq [0,\ 1]$$

and, that way, Q1 is viewed as the answer to the question: "how many a's are c's." For instance, "how many students are young."

So, noticing that relative cardinality is a particular case of conditional probability, since

$$P\,(c\,|\,a) = \frac{|c \cap a|}{|a|}$$

we are going to manipulate conditional probabilities only.

The problem is now to answer to queries like "Q5 a's are b's" (i.e. "how many a's are b's") in the preceding network. Which is equivalent to calculate P (b | a).

We are now going to present an approximate method to compute an interval for the lower bound and, an interval for the upper bound of P (b | a).

2. The mathematical programming based method

In [PAASS 1988], Paass presents an approximate method based on mathematical programming to reason with conditional probabilities.

Let $\mathcal{U} \equiv \{U_1, \ldots, U_{nU}\}$ a set of relevant propositions, we have to construct the smallest set $\mathcal{W} \equiv \{W_1, \ldots, W_{nw}\}$ of "elementary propositions" that fulfil the following conditions:

(a) each U_i is the disjunction of some of the W_j : $U_i = \bigvee_{j \in J(i)} W_j$

(b) the W_j are exclusive : $W_{j_1} \wedge W_{j_2}$ is false for $j_1 \neq j_2$

(c) the W_j are exhaustive : $W_1 \vee \ldots \vee W_{nw}$ is true

We have to notice that because of (b) and (c),

$$\sum_{j=1}^{nw} p(W_j) = 1$$

and $p(U_i) = \sum_{j \in J(i)} p(W_j)$, where $U_i = \bigvee_{j \in J(i)} W_j$

We can now adopt a vector notation. Each specified (prior) probability π_i can be considered as a conditional probability[3] , then

$$\pi_i = \frac{p(U_i^+)}{p(U_i^-)}$$

where, $U_i^+ = A_{i1} \wedge A_{i2} \in \mathcal{U}$ and $U_i^- = A_{i2} \in \mathcal{U}$, so

$$p(U_i^+) = \sum_{j \in J^+(i)} p(W_j) \text{ and } p(U_i^-) = \sum_{j \in J^-(i)} p(W_j)$$

Then we define two (0-1)-matrix $R^+ = [r_{ij}^+]_{nU,\, nw}$ and $R^- = [r_{ij}^-]_{nU,\, nw}$ such that

$p(U_i^+) = r_{ij}^+ . pw$ and $p(U_i^-) = r_{ij}^- . pw$, with

$$pw \equiv \begin{bmatrix} p(W_1) \\ \vdots \\ p(W_{nw}) \end{bmatrix}$$

---

[3] $P(A) = P(A \mid \mathbb{1})$ where $\mathbb{1}$ is the ever true event

then $\pi_i = r_i^+.pw/r_i^-.pw \Leftrightarrow (\pi_i.r_i^- - r_i^+).pw = c_i^\pi.pw$.

So, in a matrix form, $C^\pi.pw = 0$. Then, if we want to determine the probability of some proposition

$$U^* = \bigvee_{j \in J^*} W_j$$

we are going to calculate $p(U^*) = g^*.pw$, where g* is a (0-1) vector $\left(g_j^* = 1 \text{ iff } j \in J^*\right)$.

Then, solving the two following linear programs, where C^π_{high} and C^π_{low} are constraints matrix,

$$P_{high} \equiv \begin{cases} P(U^*)_{high} = \underset{pw}{Max}\ g^* \cdot pw \\ C^\pi_{high} \cdot pw \geq 0 \\ C^\pi_{low} \cdot pw \leq 0 \\ \sum_i P(W_i) = 1 \\ \forall\, i, P(W_i) \geq 0 \end{cases} \qquad P_{low} \equiv \begin{cases} P(U^*)_{low} = \underset{pw}{Min}\ g^* \cdot pw \\ C^\pi_{high} \cdot pw \geq 0 \\ C^\pi_{low} \cdot pw \leq 0 \\ \sum_i P(W_i) = 1 \\ \forall\, i, P(W_i) \geq 0 \end{cases}$$

we obtain an interval as follows

$$P(U^*) \in \left[P(U^*)_{low}, P(U^*)_{high}\right]$$

In [PAASS 1988], Paass calls this approach the *worst-case analysis*.

So, now, in the framework of conditional probabilities, if we want to determine $P(U_1 | U_2) = \dfrac{P(U_1 \wedge U_2)}{P(U_2)}$, we let $P(U_1 \wedge U_2) = g_+ \cdot pw$ and $P(U_2) = g_- \cdot pw$.

Then, $P(U_1 | U_2) \in \left[P(U_1 | U_2)_{low}, P(U_1 | U_2)_{high}\right]$, solving the two following mathematical[4] programs (in P_{high} and P_{low}, the last constraint expresses the prior probability distribution, and S denotes the set of constraints):

[4] linear with fractional objective functions

P_{high}

$$\begin{cases} P(U_1 | U_2)_{high} = \text{Max} \dfrac{g_+ \cdot pw}{g_- \cdot pw} \\ C^{\pi}_{high} \cdot pw \geq 0 \\ C^{\pi}_{low} \cdot pw \leq 0 \\ \sum_i P(W_i) = 1 \\ \forall\, i,\ P(W_i) \geq 0 \\ P_a \cdot pw = a \end{cases}$$

P_{low}

$$\begin{cases} P(U_1 | U_2)_{low} = \text{Min} \dfrac{g_+ \cdot pw}{g_- \cdot pw} \\ C^{\pi}_{high} \cdot pw \geq 0 \\ C^{\pi}_{low} \cdot pw \leq 0 \\ \sum_i P(W_i) = 1 \\ \forall\, i,\ P(W_i) \geq 0 \\ P_a \cdot pw = a \end{cases}$$

where "P_a" is the prior probabilities vector.

According to Paass, it is not possible to apply the pure SIMPLEX algorithm, because we maximize (or minimize) a ratio where the denominator and numerator are not independent; so, we have no more linear programs. To solve this problem we may use a "scanning" method, i.e.

> we compute first an interval $[p_{low}, p_{high}]$ by minimizing (for p_{low}), and maximizing (for p_{high}), the denominator of the initial objective function, over S. Then we divide this interval in equal parts, called subdivisions $[p^i_{low}, p^{i+1}_{high}]$ and solve two (one with Min - with p^{*i}_{low} as solution, and, one with Max - with p^{*i}_{high} as solution) new linear program given by the integration of the constraint $p \in [p^i_{low}, p^{i+1}_{high}]$ to S, and whith the numerator of the initial objective function as new objective function. So, if N is the number of subdivisions, we have to solve 2.N + 2 linear programs (N to approximate the lower bound, and N to approximate the upper bound and two to compute $[p_{low}, p_{high}]$). As great is the precision we want, as great is the number of intervals, and, the same the number of linear programs to solve.
>
> Then, for instance, if p^*_{low} denotes the solution of P_{low}, according to Paass, "the maximum of all upper bounds and the minimum of all lower bounds gives a globally valid range for " p^*_{low}.

But, this method can be improved. That's what we are going to show now.

3. An improvement which leads to an exact method

Here we show that we can transform the two following mathematical programs,

$$P_{high} \quad \begin{cases} P(U_1 | U_2)_{high} = \text{Max} \dfrac{g_+ \cdot pw}{g_- \cdot pw} \\ C^{\pi}_{high} \cdot pw \geq 0 \\ C^{\pi}_{low} \cdot pw \leq 0 \\ \sum_i P(W_i) = 1 \\ \forall i, P(W_i) \geq 0 \\ P_a \cdot pw = a \end{cases} \qquad P_{low} \quad \begin{cases} P(U_1 | U_2)_{low} = \text{Min} \dfrac{g_+ \cdot pw}{g_- \cdot pw} \\ C^{\pi}_{high} \cdot pw \geq 0 \\ C^{\pi}_{low} \cdot pw \leq 0 \\ \sum_i P(W_i) = 1 \\ \forall i, P(W_i) \geq 0 \\ P_a \cdot pw = a \end{cases}$$

to obtain linear programs.

In [SCHAIBLE, IBARAKI 1983], such nonlinear programs are called fractional programs. A fractional can be transformed in an equivalent linear program ([SCHAIBLE, IBARAKI 1983], [CHARNES, COOPER 1962]).

In the following, we are going to show the transformation only for P_{high} (for P_{low}, the method is the same).

Assume we have (P), the following fractional program,

$$(P) \quad \begin{cases} \text{Max} \dfrac{f(x)}{g(x)} \\ x \in S \end{cases}$$

where $S = \{x \in C : h_i(x) \leq 0, i \in]m]\}$ is the set of constraints. Then, with the following variable transformation [CHARNES, COOPER 1962],

$$y = \frac{x}{g(x)} \text{ and } t = \frac{1}{g(x)}$$

we reduce (P) to

$$(P') \quad \begin{cases} \text{Max } t\, f({}^{y}/_{t}) \\ t\, h_i({}^{y}/_{t}) \leq 0, \ i \in]m] \\ t\, g({}^{y}/_{t}) \leq 1 \\ {}^{y}/_{t} \in C \\ t > 0 \end{cases}$$

If, f and g are linear, (P) is called a linear fractional program and may be written, as follow,

(P) $\begin{cases} \text{Max } \dfrac{c^T x + \alpha}{d^T x + \beta} \\ B\,x \le b \\ x \ge 0 \end{cases}$ which yields the new linear program (P'), (P') $\begin{cases} \text{Max } c^T y + \alpha\, t \\ B\,y - b\,t \le 0 \\ d^T y + \beta\, t = 1 \\ y \ge 0\,,\, t > 0 \end{cases}$

So, in this new formalism, P_{high} is ($x = pw$, $\alpha = \beta = b = 0$, $c^T = g_+$, $d^T = g_-$),

$$P_{high}$$
$$\begin{cases} \text{Max } \dfrac{c^T . x}{d^T . x}\,,\ x \ge 0 \\ C^{\pi}_{high} \cdot x \ge 0 \\ C^{\pi}_{low} \cdot x \le 0 \\ \sum_i x_i = 1 \\ P_a \,.\, x = a \end{cases}$$

which yields the linear program,

$$P'_{high}$$
$$\begin{cases} \text{Max } c^T . y\,,\ y \ge 0\,, t > 0 \\ C^{\pi}_{high} \cdot y \ge 0 \\ C^{\pi}_{low} \cdot y \le 0 \\ d^T . y = 1 \\ \sum_i (y_i - y_i) = 0 \quad (1) \\ P_a \cdot y - a.t = 0 \quad (2) \end{cases}$$

noticing that the constraint (1) has to be removed.

Now, we define the following vector,

$$qw \equiv \begin{bmatrix} \dfrac{p(W_1)}{\vdots} \\ \dfrac{p(W_{nw})}{t} \end{bmatrix}$$

Then, if, $P_a \equiv \begin{bmatrix} P_{a1} \\ \vdots \\ P_{ap} \end{bmatrix}^T$, we can define $P'_a \equiv \begin{bmatrix} P_{a1} \\ \vdots \\ P_{ap} \\ -a \end{bmatrix}^T$, so, the constraint (2) becomes

$P'_a . qw = 0.$

Thus, for P_{high}, we derive the equivalent linear program Q ($y = qw$),

$$
Q \quad \begin{cases} \text{Max } g_+ \cdot qw \\ C^{\pi}_{high} \cdot qw \geq 0 \\ C^{\pi}_{low} \cdot qw \leq 0 \\ \forall\, i,\ P(W_i) \geq 0 \\ P'_a \cdot qw = 0 \\ g_- \cdot qw = 1 \end{cases}
$$

And then, the same way for P_{low}.

So, now, we are going to solve only two (one to compute the exact lower bound and one to compute the exact upper bound) linear program, instead of 10002 (2 to compute the first interval, 5000 to approximate the lower bound and 5000 to approximate the upper bound) or more if we want a great precision of the result, by Paass method.

4. A counter-example

Here, we show, with an example, that the method presented in [PAASS 1988] is not always correct. For a certain kind of problems, we show that the precision of the result does not increase with the number of subdivisions.

Let us consider the following mathematical program (see example 2 in section 5 of this paper),

$$
PM \quad \begin{cases} \text{MIN } \dfrac{f(x)}{g(x)} = \dfrac{x_1 + x_3}{x_1 + x_2 + x_3 + x_4} \\ x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 = 1 \\ 10x_1 + 10x_2 - 90x_3 - 90x_4 = 0 \\ 75x_1 + 75x_2 - 25x_5 - 25x_6 = 0 \\ 10x_1 - 90x_2 + 10x_5 - 90x_6 = 0 \\ 40x_1 - 60x_3 + 40x_5 - 60x_7 = 0 \end{cases}
$$

from the minimisation and the maximization of g(x), under the preceding constraints, we obtain $g(x) \in [t_0, t_N]$ ($t_0 = 0$ and $t_N = 0.174$). So, we derive the new mathematical program,

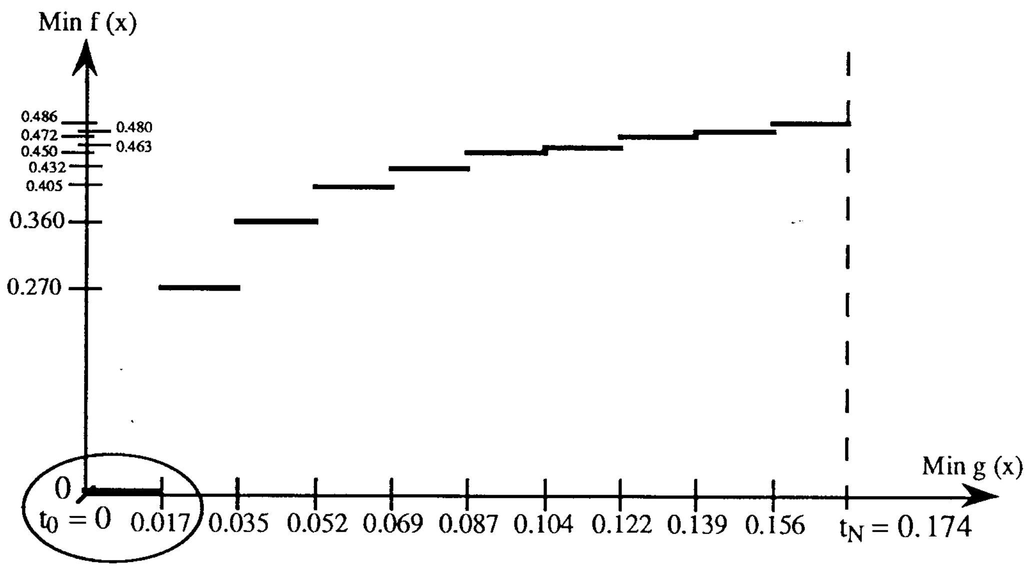
$$
\text{PL}\quad
\begin{cases}
\text{MIN } f(x) = x_1 + x_3 \\
x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 = 1 & (1) \\
10x_1 + 10x_2 - 90x_3 - 90x_4 = 0 & (2) \\
75x_1 + 75x_2 - 25x_5 - 25x_6 = 0 & (3) \\
10x_1 - 90x_2 + 10x_5 - 90x_6 = 0 & (4) \\
40x_1 - 60x_3 + 40x_5 - 60x_7 = 0 & (5) \\
x_1 + x_2 + x_3 + x_4 \geq t_0 & (6) \\
x_1 + x_2 + x_3 + x_4 \leq t_N & (7)
\end{cases}
$$

with two new constraints.

The interval $[t_0, t_N]$ is then split into intervals $[t_i, t_{i+1}]$, and, we solve as many problems as intervals, where t_0 is changed into t_i and t_N into t_{i+1}, for all i.

What we are going to show now is that, even with a great number of subdivisions over $[t_0, t_N]$, we wont get a more precise result for the lower bound; because $t_0 = 0$.

Indeed, if we consider a subdivision over the interval $[0, t_N]$, and the associated solutions of (PL) in the following figure,



and, because the lower bound of Min(f(x)/g(x)) is the minimum of all the minima of f(x) on each interval of the subdivision of $[0, t_N]$, the lower bound of the solution of (PM) is 0.

Then, even if N is very big, i.e. if the first interval, $[0, t_1]$, of the subdivision of $[0, t_N]$ is very narrow, the solution of (PL) in this interval is always 0.

So, now, we can derive the following result

Let (P),

$$\text{(P)}\quad \begin{cases} \text{MIN}\ \dfrac{f(x)}{g(x)} \\ A.x = b \end{cases}$$

the mathematical program to solve (p* the optimal solution), and, if in (P') and (P''),

$$\text{(P')}\quad \begin{cases} \text{MIN}\ g(x) \\ A.x = b \end{cases} \qquad \text{(P'')}\quad \begin{cases} \text{MAX}\ g(x) \\ A.x = b \end{cases}$$

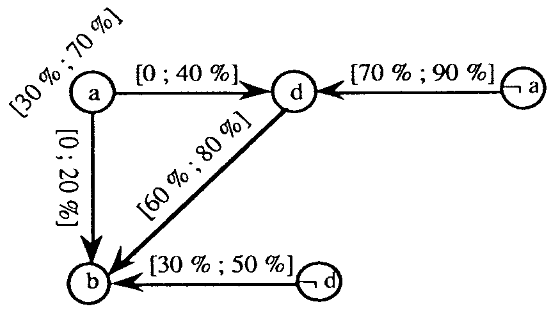
0 is solution for (P') and t is solution for (P''), then, the Paass method will always give 0 as lower bound for p*; whatever the number of subdivisions over [0, t].

5. Comparative tests

Here, we give three examples. One given by Paass in [PAASS 1988], and, two others which are relevant for the problem we want to treat. The second example is the one exposed as counter-example in section 4.

<u>Example 1:</u> see [PAASS 1988] page 226

First, we give the associated inference network



Then (PM1), the mathematical program proposed by Paass to answer to the query "Q1 (¬a ∧ b)'s are d's," and (PL1), the linear program we propose, PM1

$$\begin{cases} \mathrm{OPTI}\ \dfrac{f(x)}{g(x)} = \dfrac{x_5}{x_5+x_7} \\ x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 = 1 \\ 60x_1 + 60x_2 - 40x_3 - 40x_4 \leq 0 \\ 80x_1 - 20x_2 + 80x_3 - 20x_4 \geq 0 \\ 40x_1 - 60x_2 + 40x_5 - 60x_6 \geq 0 \\ 20x_1 - 80x_2 + 20x_5 - 80x_6 \leq 0 \\ 30x_5 - 30x_6 - 70x_7 - 70x_8 \geq 0 \\ 10x_5 + 10x_6 - 90x_7 - 90x_8 \leq 0 \\ 70x_3 - 30x_4 + 70x_7 - 30x_8 \geq 0 \\ 50x_3 - 50x_4 + 50x_7 - 50x_8 \leq 0 \\ 70x_1 + 70x_2 + 70x_3 + 70x_4 - 30x_5 - 30x_6 - 30x_7 - 30x_8 \geq 0 \\ 30x_1 + 30x_2 + 30x_3 + 30x_4 - 70x_5 - 70x_6 - 70x_7 - 70x_8 \leq 0 \end{cases}$$

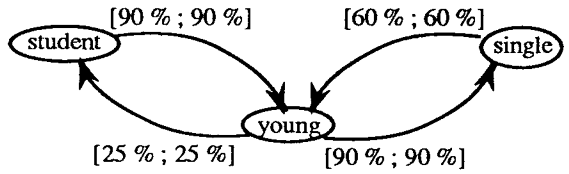
PL1

$$\begin{cases} \mathrm{OPTI}\ f(x) = x_5 \\ x_5 + x_7 = 1 \\ 60x_1 + 60x_2 - 40x_3 - 40x_4 \leq 0 \\ 80x_1 - 20x_2 + 80x_3 - 20x_4 \geq 0 \\ 40x_1 - 60x_2 + 40x_5 - 60x_6 \geq 0 \\ 20x_1 - 80x_2 + 20x_5 - 80x_6 \leq 0 \\ 30x_5 - 30x_6 - 70x_7 - 70x_8 \geq 0 \\ 10x_5 + 10x_6 - 90x_7 - 90x_8 \leq 0 \\ 70x_3 - 30x_4 + 70x_7 - 30x_8 \geq 0 \\ 50x_3 - 50x_4 + 50x_7 - 50x_8 \leq 0 \\ 70x_1 + 70x_2 + 70x_3 + 70x_4 - 30x_5 - 30x_6 - 30x_7 - 30x_8 \geq 0 \\ 30x_1 + 30x_2 + 30x_3 + 30x_4 - 70x_5 - 70x_6 - 70x_7 - 70x_8 \leq 0 \end{cases}$$

where OPTI is either MIN or MAX.

<u>Example 2:</u> the simplest one

First, the associated inference network



Then (PM2), the mathematical program proposed by Paass to answer the query "Q2 students are single," and (PL2), the linear program we propose

PM2

$$\begin{cases} \mathrm{OPTI}\ \dfrac{f(x)}{g(x)} = \dfrac{x_1+x_3}{x_1+x_2+x_3+x_4} \\ x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 = 1 \\ 10x_1 + 10x_2 - 90x_3 - 90x_4 = 0 \\ 75x_1 + 75x_2 - 25x_5 - 25x_6 = 0 \\ 10x_1 - 90x_2 + 10x_5 - 90x_6 = 0 \\ 40x_1 - 60x_3 + 40x_5 - 60x_7 = 0 \end{cases}$$

PL2

$$\begin{cases} \mathrm{OPTI}\ f(x) = x_1 + x_3 \\ x_1 + x_2 + x_3 + x_4 = 1 \\ 10x_1 + 10x_2 - 90x_3 - 90x_4 = 0 \\ 75x_1 + 75x_2 - 25x_5 - 25x_6 = 0 \\ 10x_1 - 90x_2 + 10x_5 - 90x_6 = 0 \\ 40x_1 - 60x_3 + 40x_5 - 60x_7 = 0 \end{cases}$$

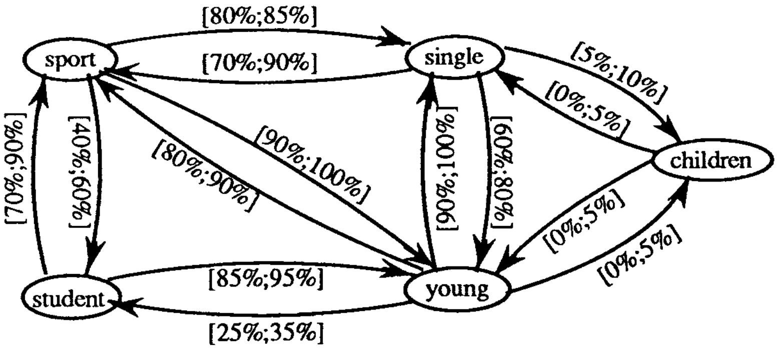
where OPTI is either MIN or MAX.

<u>Example 3:</u> the "big one" (which is not so big in fact)

This example corresponds to the syllogism problem solved analytically in [DUBOIS, PRADE, TOUCAS 1989].

First we give the associated inference network

Reasoning with conditional probabilities



Then (PM3), the mathematical program given by Paass. This program is too big to be written in a formal form, so, we give bellow the SIMPLEX array, noticing that we have 32 variables and 26 constraints

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-	1
30	30	30	30	30	30	30	30	-70	-70	-70	-70	-70	-70	-70	-70	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	≥	0
10	10	10	10	10	10	10	10	-90	-90	-90	-90	-90	-90	-90	-90	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	≤	0
15	15	15	15	-85	-85	-85	-85	15	15	15	15	-85	-85	-85	-85	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	≥	0
5	5	5	5	-95	-95	-95	-95	5	5	5	5	-95	-95	-95	-95	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	≤	0
60	60	60	60	60	60	60	60	0	0	0	0	0	0	0	0	-40	-40	-40	-40	-40	-40	-40	-40	0	0	0	0	0	0	0	0	≥	0
40	40	40	40	40	40	40	40	0	0	0	0	0	0	0	0	-60	-60	-60	-60	-60	-60	-60	-60	0	0	0	0	0	0	0	0	≤	0
20	20	-80	-80	20	20	-80	-80	0	0	0	0	0	0	0	0	20	20	-80	-80	20	20	-80	-80	0	0	0	0	0	0	0	0	≥	0
15	15	-85	-85	15	15	-85	-85	0	0	0	0	0	0	0	0	15	15	-85	-85	15	15	-85	-85	0	0	0	0	0	0	0	0	≤	0
10	10	10	10	-90	-90	-90	-90	0	0	0	0	0	0	0	0	10	10	10	10	-90	-90	-90	-90	0	0	0	0	0	0	0	0	≥	0
0	0	0	0	-100	-100	-100	-100	0	0	0	0	0	0	0	0	0	0	0	0	-100	-100	-100	-100	0	0	0	0	0	0	0	0	≤	0
75	75	75	75	0	0	0	0	75	75	75	75	0	0	0	0	-25	-25	-25	-25	0	0	0	0	-25	-25	-25	-25	0	0	0	0	≥	0
65	65	65	65	0	0	0	0	65	65	65	65	0	0	0	0	-35	-35	-35	-35	0	0	0	0	-35	-35	-35	-35	0	0	0	0	≤	0
20	20	20	20	0	0	0	0	-80	-80	-80	-80	0	0	0	0	20	20	20	20	0	0	0	0	-80	-80	-80	-80	0	0	0	0	≥	0
10	10	10	10	0	0	0	0	-90	-90	-90	-90	0	0	0	0	10	10	10	10	0	0	0	0	-90	-90	-90	-90	0	0	0	0	≤	0
10	10	-90	-90	0	0	0	0	10	10	-90	-90	0	0	0	0	10	10	-90	-90	0	0	0	0	10	10	-90	-90	0	0	0	0	≥	0
0	0	-100	-100	0	0	0	0	0	0	-100	-100	0	0	0	0	0	0	-100	-100	0	0	0	0	0	0	-100	-100	0	0	0	0	≤	0
95	-5	95	-5	0	0	0	0	95	-5	95	-5	0	0	0	0	95	-5	95	-5	0	0	0	0	95	-5	95	-5	0	0	0	0	≤	0
30	30	0	0	30	30	0	0	-70	-70	0	0	-70	-70	0	0	30	30	0	0	30	30	0	0	-70	-70	0	0	-70	-70	0	0	≥	0
10	10	0	0	10	10	0	0	-90	-90	0	0	-90	-90	0	0	10	10	0	0	10	10	0	0	-90	-90	0	0	-90	-90	0	0	≤	0
40	40	0	0	-60	-60	0	0	40	40	0	0	-60	-60	0	0	40	40	0	0	-60	-60	0	0	40	40	0	0	-60	-60	0	0	≥	0
20	20	0	0	-80	-80	0	0	20	20	0	0	-80	-80	0	0	20	20	0	0	-80	-80	0	0	20	20	0	0	-80	-80	0	0	≤	0
95	-5	0	0	95	-5	0	0	95	-5	0	0	95	-5	0	0	95	-5	0	0	95	-5	0	0	95	-5	0	0	95	-5	0	0	≥	0
90	-10	0	0	90	-10	0	0	90	-10	0	0	90	-10	0	0	90	-10	0	0	90	-10	0	0	90	-10	0	0	90	-10	0	0	≤	0
95	0	95	0	-5	0	-5	0	95	0	95	0	-5	0	-5	0	95	0	95	0	-5	0	-5	0	95	0	95	0	-5	0	-5	0	≤	0
95	0	-5	0	95	0	-5	0	95	0	-5	0	95	0	-5	0	95	0	-5	0	95	0	-5	0	95	0	-5	0	95	0	-5	0	≤	0

and the function to optimize (maximize or minimize) is

$$\frac{f(x)}{g(x)} = \frac{x_1 + x_3 + x_9 + x_{11}}{\sum_{i=1}^{16} x_i}$$

to answer to the query "Q3 student's are young and have children."

And (PL3), the linear program we propose, given in the preceding form, with always 32 variables and 26 constraints

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	1
30	30	30	30	30	30	30	30	-70	-70	-70	-70	-70	-70	-70	-70	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	≥	0
10	10	10	10	10	10	10	10	-90	-90	-90	-90	-90	-90	-90	-90	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	≤	0
15	15	15	15	-85	-85	-85	-85	15	15	15	15	-85	-85	-85	-85	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	≥	0
5	5	5	5	-95	-95	-95	-95	5	5	5	5	-95	-95	-95	-95	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	≤	0
60	60	60	60	60	60	60	60	0	0	0	0	0	0	0	0	-40	-40	-40	-40	-40	-40	-40	-40	0	0	0	0	0	0	0	0	≥	0
40	40	40	40	40	40	40	40	0	0	0	0	0	0	0	0	-60	-60	-60	-60	-60	-60	-60	-60	0	0	0	0	0	0	0	0	≤	0
20	20	-80	-80	20	20	-80	-80	0	0	0	0	0	0	0	0	20	20	-80	-80	20	20	-80	-80	0	0	0	0	0	0	0	0	≥	0
15	15	-85	-85	15	15	-85	-85	0	0	0	0	0	0	0	0	15	15	-85	-85	15	15	-85	-85	0	0	0	0	0	0	0	0	≤	0
10	10	10	10	-90	-90	-90	-90	0	0	0	0	0	0	0	0	10	10	10	10	-90	-90	-90	-90	0	0	0	0	0	0	0	0	≥	0
0	0	0	0	-100	-100	-100	-100	0	0	0	0	0	0	0	0	0	0	0	0	-100	-100	-100	-100	0	0	0	0	0	0	0	0	≤	0
75	75	75	75	0	0	0	0	75	75	75	75	0	0	0	0	-25	-25	-25	-25	0	0	0	0	-25	-25	-25	-25	0	0	0	0	≥	0
65	65	65	65	0	0	0	0	65	65	65	65	0	0	0	0	-35	-35	-35	-35	0	0	0	0	-35	-35	-35	-35	0	0	0	0	≤	0
20	20	20	20	0	0	0	0	-80	-80	-80	-80	0	0	0	0	20	20	20	20	0	0	0	0	-80	-80	-80	-80	0	0	0	0	≥	0
10	10	10	10	0	0	0	0	-90	-90	-90	-90	0	0	0	0	10	10	10	10	0	0	0	0	-90	-90	-90	-90	0	0	0	0	≤	0
10	10	-90	-90	0	0	0	0	10	10	-90	-90	0	0	0	0	10	10	-90	-90	0	0	0	0	10	10	-90	-90	0	0	0	0	≥	0
0	0	-100	-100	0	0	0	0	0	0	-100	-100	0	0	0	0	0	0	-100	-100	0	0	0	0	0	0	-100	-100	0	0	0	0	≤	0
95	-5	95	-5	0	0	0	0	95	-5	95	-5	0	0	0	0	95	-5	95	-5	0	0	0	0	95	-5	95	-5	0	0	0	0	≤	0
30	30	0	0	30	30	0	0	-70	-70	0	0	-70	-70	0	0	30	30	0	0	30	30	0	0	-70	-70	0	0	-70	-70	0	0	≥	0
10	10	0	0	10	10	0	0	-90	-90	0	0	-90	-90	0	0	10	10	0	0	10	10	0	0	-90	-90	0	0	-90	-90	0	0	≤	0
40	40	0	0	-60	-60	0	0	40	40	0	0	-60	-60	0	0	40	40	0	0	-60	-60	0	0	40	40	0	0	-60	-60	0	0	≥	0
20	20	0	0	-80	-80	0	0	20	20	0	0	-80	-80	0	0	20	20	0	0	-80	-80	0	0	20	20	0	0	-80	-80	0	0	≤	0
95	-5	0	0	95	-5	0	0	95	-5	0	0	95	-5	0	0	95	-5	0	0	95	-5	0	0	95	-5	0	0	95	-5	0	0	≥	0
90	-10	0	0	90	-10	0	0	90	-10	0	0	90	-10	0	0	90	-10	0	0	90	-10	0	0	90	-10	0	0	90	-10	0	0	≤	0
95	0	95	0	-5	0	-5	0	95	0	95	0	-5	0	-5	0	95	0	95	0	-5	0	-5	0	95	0	95	0	-5	0	-5	0	≤	0
95	0	-5	0	95	0	-5	0	95	0	-5	0	95	0	-5	0	95	0	-5	0	95	0	-5	0	95	0	-5	0	95	0	-5	0	≤	0

with the following function to optimize (maximize or minimize)

$$f(x) = x_1 + x_3 + x_9 + x_{11}$$

And in the following table, we compare the results given by the Paass method and our method.

In the columns denoted "s" we give the running times in seconds. And, for the Paass method we give in the columns "answer" the intervals of the lower and upper bounds; while, for our method, we give the interval where the exact value lies[5].

	Improved Method		Paass Method 2 subdivisions		Paass Method 10 subdivisions		Paass Method 100 subdivisions		Paass Method 500 subdivisions		Paass Method 10000 subdivisions	
	answer	s	answer	s	answer	s	answer	s	answer	s	answer	s
Q1	[0.538;1.000]	2	[0.294;0.560] [0.988;1.981]	3	[0.477;0.544] [0.999;1.142]	8	[0.532;0.539] [0.999;1.012]	63	[0.537;0.539] [1.000;1.003]	306	[0.538;0.538] [1.000;1.000]	6053
Q2	[0.540;1.000]	2	[0.000;0.540] [1.000;2.000]	2	[0.000;0.540] [1.000;2.000]	4	[0.000;0.540] [1.000;2.000]	22	[0.000;0.540] [1.000;2.000]	104	[0.000;0.540] [1.000;2.000]	2070
Q3	[0.000;0.121]	12	[0.000;0.000] [0.121;0.243]	56	[0.000;0.000] [0.121;0.243]	210	[0.000;0.000] [0.121;0.243]	1739	[0.000;0.000] [0.121;0.243]	8618	[0.000;0.000] [0.121;0.243]	167211

These tests have been done on a SUN WORKSTATION 3/50, diskless and without arithmetical coprocessor. The algorithms, including the SIMPLEX, are written in Pascal.

The implementation of the SIMPLEX algorithm is extracted from [LEMAIRE 1988] and is not very efficient, but, more than the efficiency itself, what we must note is the gain in time. This gain will be the same with a more efficient SIMPLEX and/or computer.

6. Conclusion

In this paper, we have shown that the Paass method in [PAASS 1988] is not always correct, and, even so, can't be used in a system reasoning with conditional probabilities.

The method we have as an improvement of the Paass method is correct and is more efficient, but, it is still impossible to use it in a real system, because it may be not efficient enough for large networks.

In fact, such a method (i.e. a linear programming based method) is not directly applicable in an artificial intelligence system, because it is then impossible to explain the results.

---

[5] 167211 seconds = 46 H 26 Mn 51 s

We think that our method could be used locally, i.e. on small parts of the inference network, to guide the inference over the whole net. Moreover, it can useful to evaluate the performance of local propagation techniques such as derived from inference rules described in [DUBOIS, PRADE, TOUCAS 1989].

References

[CHARNES, COOPER 1962] A. Charnes, W.W. Cooper, "Programming with linear fractional functionals", *Naval Res. Logist. Quart. 9 (1962)* pp. 181-186

[DUBOIS, PRADE 1988] Didier DUBOIS, Henri PRADE, "On fuzzy syllogisms," *Comput. Intell. 4 (1988),* pp. 171-179

[DUBOIS, PRADE, TOUCAS 1989] Didier DUBOIS, Henri PRADE, Jean-Marc TOUCAS, "Inference with imprecise numerical quantifiers," *to appear in "Intelligent Systems : State of the Art and Future Directions," Edited by Z. Ras and M. Zemankova, Ellis Horwood Ltd.*

[LEMAIRE 1988] B. LEMAIRE, "Programmation linéaire sur micro-ordinateur," Eds. MASSON, Paris (FRANCE), 1988

[MINOUX 1983] Michel MINOUX, "Programmation Mathématique. Théorie et algorithmes" (Tome 1 and Tome 2), Eds. DUNOD, Paris (FRANCE), 1983

[NILSSON 1986] Nils J. NILSSON, "Probabilistic logic," *Artificial Intelligence* **28** (1986) 71-87

[SCHAIBLE, IBARAKI 1983] Siegfried SCHAIBLE, Toshihide IBARAKI, "Fractional Programming," *European Journal of Operation Research* 12 (1983) 325-338

[PAASS 1988] Gerhard PAASS, "Probabilistic logic," in *Non-Standard Logics for Automated Reasoning*, (ed. Philippe SMETS, Abe MAMDANI, Didier DUBOIS and Henri PRADE), Eds. ACADEMIC PRESS, 1988, pp 213-251

[PEARL 1988] Judea PEARL, "Probabilistic reasoning in intelligent systems: networks of plausible inference," Morgan Kaufmann Publishers, 1988