XIV Conferencia Latinoamericana de Informática
17avas. Jornadas Argentinas de Informática
e Investigación Operativa
Buenos Aires, Setiembre 1988.

Computing Certainty Factors in Expert Systems Using Dihypergraphs Flows

Julián Aráoz D.
Departamento de Procesos y Sistemas
Universidad Simón Bolívar

Edgardo Broner
Departamento de Matemáticas y Ciencias de la Computación
Universidad Simón Bolívar

Alberto Torres F.
Departamento de Matemáticas y Ciencias de la Computación
Universidad Simón Bolívar

Abstract: We use a Directed Hypergraph model that allows the reduction of derivation in Horn Systems to the calculation of a Linear Programming problem feasible solution associated to a Unit Flow searching. We explore the application of this tool in the calculation of Certainty Factors. Since several Expert Systems Knowledge Bases are expressed, or can be expressed, with Horn clauses associated with Certainty Factors, the results of this work can be applied to Knowledge Searching in such bases. These results show the importance of the systematic application of Combinatorial Optimization models to Expert Systems.

Resumen: Se utiliza un modelo de Hipergrafos Dirigidos que permite reducir una derivación en un sistema de Horn al cálculo de una solución factible en un problema de Programación Lineal asociado a la búsqueda de flujos unitarios. Se explora la aplicación de esta herramienta en el cálculo de Factores de Certeza. Como las Bases de Conocimiento de muchos Sistemas Expertos estan expresadas, o se pueden expresar, en cláusulas de Horn con Factores de Certeza asociados, el resultado de este trabajo se puede aplicar directamente en la búsqueda en tales Bases de Conocimiento. Estos resultados muestran la importancia de la aplicación sistemática de los modelos de Optimización Combinatoria al área de Sistemas Expertos

1.- Introduction:

In last few years, Expert Systems [2,5] have become one of the most important areas in Artificial Intelligence. This technology has been developed for a wide variety of practical problems resolution, and has been established as a new programming methodology paradigm. In this way, the knowledge used building a system transcends program limits and become part of the data. That is why Expert Systems are also called Knowledge-based Systems.

Various schemes are used to represent a given Knowledge Base. Such a Knowledge Base is usually modifiable by user interaction and is handled by the system in order to produce new knowledge or to check if a given fact is a consequence of the represented knowledge. Generally, these operations have been seen as a particular case of logical deduction and have been treated with logical methods.

From other point of view, in [11], it was presented a directed hypergraph based model for Horn clauses systems, which leads to the equivalence between logical deduction in such systems and a linear programming problem corresponding to a generalization of flow to directed hypergraphs. With this approach, we consider in this work the inclusion of Certainty Factors in the model.

In section 2, we present the definitions and results of Directed Hypergraphs, Flows and Horn Systems that we use in this work. In section 3, we consider various scheemes for Certainty Factors and how can be implemented in this Linear Model.

2.- Directed Hypergraphs and Horn Systems

In order to work with the dual of linear problems formulated on the incidence matrix of a graph, Berge [1] introduced the definition of hypergraph, a generalization of the concept of graph, which consist of a set of vertices and a family of hyperedges, each one a subset of vertices with cardinality not restricted to be equal to two (as it is on graphs). In this way any matrix with only zero or one valued entries is the incidence matrix of a hypergraph and so it is its transpose. Usually empty hyperedges are undesired, and isolated vertices are also forbidden, in order to assure that the dual holds the same property. The same idea may be trivially extended to directed graphs, leading to the concept of directed hypergraphs or dihypergraphs.

2.1 - *Definition*: Let V be a finite nonempty set, and let $E \subseteq \mathcal{P}(V)^2$. Then the pair $H = (V,E)$ is a directed hypergraph iff $\forall\, e \in E$ such that $e = (t(e),h(e))$:

(i) $t(e) \cap h(e) = \emptyset$

(ii) $t(e) \cup h(e) \neq \emptyset$

An element of V will be called a vertex and an element of E will be called a hyperedge or simply an edge when not leading to confusion. Henceforth, the set of all vertices of a directed hypergraph H will be denoted as V_H and the set of all its edges as E_H, when not leading to confusion the subindex H will be omitted. If $e = (V', V'')$ is an edge of a directed hypergraph, V' will be called the tail of e, and V'' the head of e. The tail of an edge e will also be denoted as $t(e)$ and its head as $h(e)$.

A directed hypergraph is therefore a pair of sets: a set of vertices and a set of hyperedges, where the head and the tail of a hyperedge are allowed to be any pair of disjoint set of vertices, not both empty.

In this work any finite set is allowed to be an index set. A vector A with index on a set I will be denoted by $A = (a_i : i \in I)$, and a_i will be the entry of A indexed by i. A matrix M indexed on two sets I and J, will be denoted by $M = (m_{i,j} : i \in I, j \in J)$, where $m_{i,j}$ will be the entry of M indexed by i and j. M may be interpreted as a vector indexed on the cartesian product of I and J, that is $M = (m_{(i,j)} : (i,j) \in I \times J)$. A function F with domain in a finite set D will be represented by the vector $\bar{F} = (f_d : d \in D)$ where $f_d = F(d)$.

2.2 - *Definition:* Let H be a directed hypergraph. Then the incidence matrix of H will be $\bar{H} = (h_{v,e} : v \in V, e \in E)$, where:

$$h_{v,e} = \begin{cases} -1 & \text{if } (v \in t(e)) \\ 1 & \text{if } (v \in h(e)) \\ 0 & \text{if } (v \notin t(e) \cup h(e)) \end{cases}$$

Any matrix with only 0, 1 and -1 valued entries is then the incidence matrix of a directed hypergraph if it does not have two equal columns or a zero column.

Now we extend the concept of path to directed hypergraphs.

2.3.- *Definition:* A path, from s_0 to s_k ($k \geq 0$), in a directed hypergraph H is a sequence of form:

$$s_0 e_1 s_1 e_2 s_2 \ldots s_{k-1} e_k s_k$$

Where:

(i) $\forall i \in [0..k] (s_i \in \mathcal{P}(V))$

(ii) $\forall i \in [1..k] (e_i \in E)$

(iii) $\forall i \in [1..k] ((t(e_i) \cap s_{i-1} = \emptyset) \wedge (h(e_i) \cap s_i \neq \emptyset))$

The edge sequence associated to a path is $e_1 e_2 \ldots e_k$. The length of a path P (denoted by length(P)) is the total number of occurrences of edges in

its edge sequence, i. e. length($s_0e_1s_1e_2s_2...s_{k-1}e_ks_k$) = k . The set of edges in a path P will be denoted as S(P), i. e. S ($s_0e_1s_1e_2s_2...s_{k-1}e_ks_k$) = { e_i : $i \in [1..k]$ }.

This is a very weak path concept. Next, we introduce the stronger concept of increasing paths where the initial set of vertices s_0 grows in the path by mean of the edges in it.

2.4 - *Definition:* An increasing path in a directed hypergraph H is a path of the form:

$$s_0\, e_1\, s_1\, e_2\, s_2\, \ldots\, s_{k-1}\, e_k\, s_k$$

Where:

(i) $\forall i \in [1..k]\ (t(e_i) \subseteq s_{i-1})$

(ii) $\forall i \in [1..k]\ (s_i = s_{i-1} \cup h(e_i))$

Now we present a generalization of the problem of flow in networks [4] to directed hypergraphs. Most of notation and definitions are extended from [10].

2.5 - *Definition:* Let H be a directed hypergraph. Then a flow F in H is just a function of the edges of H into the real numbers, i. e. $F : E \to \mathbb{R}$. The value F(e) will be called the flux of the edge e. A flow F is an integral flow iff for all e in E, the flux of e is integer, that is Range(F) $\subset \mathbb{N}$. For a flow F, the set of edges with positive flux will be denoted as P(F), i. e. P(F) = { $e \in E : F(e) > 0$ }. A flow F will be called a nonnegative flow iff it has nonnegative flux for all edges, i. e. P(-F) = $\emptyset$.

2.6 - *Definition:* Let H be a directed hypergraph and let F be a flow in H. The divergence function of F is $Div_F : V \to \mathbb{R}$, where:

$$Div_F(v) = \sum_{\{e \in E : v \in h(e)\}} F(e) - \sum_{\{e \in E : v \in t(e)\}} F(e)$$

Note that divergence function for v_i can be calculated by multiplying i^{th} row of the incidence matrix of H by the vector $\bar{F}$. Therefore:

2.7 - *Lemma:* Let H be a directed hypergraph and F a flow in H, then:

$$\bar{H} * \bar{F} = \overline{Div}_F$$

□

It is of special interest the consideration of nonnegative flows with restrictions on the divergences. These restrictions will be, for the divergence of each node, of one of the following types: equal to a given value, less than or equal, greater than or equal, or unrestricted. Now, if R is a vector of restrictions and D a constant vector, both indexed on V_H then the set of nonnegative flows satisfying such restrictions, will be equal to the set of feasible solutions to the linear programming problem $\overline{H} * \overline{F}\ R\ D\ ;\ \overline{F} \geq 0$. Since for the scope of this paper only nonnegative flows are relevant, thereafter whenever a flow appears it will be assumed to be a nonnegative flow.

2.8 - *Definition:* Let H be a directed hypergraph, V' and V'' disjoint subsets of V, and let F be a flow in H. Then F is an <u>unit flow</u> from V' to V'' iff:

(i) $\forall v \in V''$ $(Div_F(v) = 1)$

(ii) $\forall v \in V - (V' \cup V'')$ $(Div_F(v) = 0)$

Note that a unit flow is a special case of restricted flow, so a flow is a unit flow iff it is a feasible solution of the linear problem $\overline{H} * \overline{F}\ R\ D\ ;\ \overline{F} \geq 0$, where r_v is an equality for every vertex in V'' - V' and no restriction for every vertex in V', and d_v is 1 for all vertices in V'' and 0 for all vertices in $V - (V' \cup V'')$. This can be stated in the following lemma:

2.9 - *Lemma:* Let H be a directed hypergraph, V' y V'' two disjoint subsets of V, and F a flow in H. Then F is a unit flow iff it is a feasible solution of the linear problem $\overline{H} * \overline{F}\ R\ D\ ;\ \overline{F} \geq 0$, where R and D are as described above.

□

Now we introduce the logic model we use in this work. It is based in a particular class of clauses, very important for the simplicity of the derivations with them and their suitableness for expressing most of knowledge, are Horn clauses [6] in which there is only one atomic proposition at the right side, i. e. clauses of the form: $\wedge_{\{p \in P\}} p \rightarrow p'$. If a formula is a finite conjunction of Horn

clauses it is said that it is in Horn clausular form. Note that a clause of the form: $\wedge_{(p \in P)} p \rightarrow p'$, where $p' \in P$ is a tautology, so any formula in Horn clausular form can be assume to have not such clauses.

2.10.- *Definition:* Let P be a set of atomic propositions then a Horn system S is a pair (P,C), where C is a set of nontautological Horn clauses referencing only the atomic propositions in P.

A Horn system is then naturally modelled by a directed hypergraph with all edges having a unitary head. In this model vertices represent the atomic propositions of the system and edges represent the clauses.

2.11.- *Definition:* A directed hypergraph H is a rule hypergraph iff:

$$\forall e \in E (|h(e)| = 1)$$

Any edge of H will be called a rule and any vertex an atom of H. A rule of the form $(T, \{v_h\})$ will be denoted as $T \rightarrow v_h$. Any subset of V will be an information state of H, and the set of all information states of H will be denoted by IS_H. ($IS_H = \mathcal{P}(V)$).

Then every atomic proposition p of a logical system will be represented by an atom v_p and every clause of the form: $\wedge_{(p \in P)} p \rightarrow p'$ ($p \notin P$) by a rule $e = V_P \rightarrow v_{p'}$, where $V_P = \{ v_p : p \in P \}$. In this way any Horn system S can be modelled by a rule hypergraph, that will be denoted H(S).

Now, we introduce a path-based concept of derivation:

2.12.- *Definition:* Let H be a rule hypergraph. A derivation of d from O in H, where d is a vertex and O an information state, is an increasing path, of the form

$$s_0 e_1 s_1 e_2 s_2 \ldots s_{k-1} e_k s_k$$

Where:

(i) $s_0 = O$

(ii) $d \in s_k - O$

The edge sequence of a derivation will be called rule sequence. A rule contained in the rule sequence of a derivation is said to be applied in it. A rule e is said to be applicable given an information state O iff t(e) ⊆ O.

Note that the concept of derivation in rule hypergraphs corresponds to that of logical derivation with Horn clauses by *modus ponens*. If information states are interpreted as the conjunction of the propositions represented by its atoms, which are assumed to be true, then a rule is applicable if its preconditions are true, and the set of true propositions after the rule is applied is augmented by its consequent.

This completes the path leading to the main results of [11], that state that existence of a derivation in a rule hypergraph is equivalent to existence of a feasible unit flow in it. This reduces the problem of logical deduction with propositional Horn clauses to a linear programming problem.

2.13.- *Theorem:* ([11]) Let S = (P,C) be a Horn system and let P' be a subset of P. Then the nontautological Horn clause $\wedge_{\{p' \in P'\}} p' \rightarrow p''$ is logically derivable from the clauses in S iff there is an unit flow from P' to p'' in H.

□

2.14.- *Theorem:* ([11]) Let S = (P,C) be a Horn system and let P' be a subset of P. Then the nontautological Horn clause $\wedge_{\{p' \in P'\}} p' \rightarrow p''$ is logically derivable from the clauses in S iff there is a feasible solution of the linear problem $\overline{H(S)} * \overline{F}$ R D ; $\overline{F} \geq 0$, where r_v is an equality for every vertex in $V - \{ v_{p'} : p' \in P' \}$ and no restriction for every vertex in $\{ v_{p'} : p' \in P' \}$, and d_v is 1 for $v_{p''}$ and is equal to 0 for all vertices in $V - (\{ v_{p'} : p' \in P' \} \cup \{ p'' \})$.

□

3.- Computing Certainty Factors in the Linear Model

Several Certainty Factors Algebras have been developed to deal with uncertain knowledge in Expert Systems. The most important models ones are the models given by Probabilistic Logic [9] and Fuzzy Logic [8,12], even when many others exist [3,7]

Using the result of (2.11) any Certainty Factors Algebra can be reduced to the optimization of a functional over a linearly bounded space. It is of special interest the models that can be translated to linear functions thus reducing the searching problem in Knowledge Bases with uncertainty to linear programming problems. In this sense, even when Probabilistic Logic produces a complex product functional, Fuzzy Logic allows a linear formulation, that we present in this section.

In Fuzzy Logic the Certainty Factors of a derivation is the minimum of all the factors of the rules and facts used in the derivation. Hence, the problem is to find a derivation which maximize this minimum, Since we consider facts as special rules, let α_i be the Certainty Factor of rule i. Our problem is then:

$$\max \{ \min \{ \alpha_i : \text{rule i is used in solution } \bar{F} \} : \bar{F} \text{ is a solution} \}$$

This is a max-min problem with setup costs α_i if $f_i > 0$ or 0 if $f_i = 0$ over solutions F to $\overline{H(S)} * \bar{F} \; R \; D$; $\bar{F} \geq 0$ by theorem (2.14), where R is as described in such theorem.

Let $\bar{X}$ be an integer vector with indexes on the rules of S, where x_i is equal to 0 if $f_i = 0$, and is equal to 1 otherwise. The feasible set is:

$$3.1.- \quad \overline{H(S)} * \bar{F} \; R \; D \; ; \; \bar{F} \geq 0 \; ; \; \bar{F} \leq M * \bar{X}$$

where M is a bound for f_i (like n, the number of rules in S)

Now then, the functional is:

$$\max \{ \min \{ \alpha_i : x_i = 1 \} : (\bar{F},\bar{X}) \text{ is a solution of (3.i)} \}$$

But; $\min \{ \alpha_i : x_i = 1 \} = \min \{ \alpha_i * x_i + L * (1 - x_i) : i = 1 .. n \}$, where: $L > \max \{ \alpha_i : i = 1 .. n \}$.

Therefore, to find the best Certainty Factor is equivalent to the problem:

$$\max \min \{ \alpha_i * x_i + L * (1 - x_i) : i = 1 .. n \}$$

Over:

$$\overline{H(S)} * \bar{F} \; R \; D \; ; \quad \bar{F} \geq 0 \; ; \quad \bar{F} \leq M * \bar{X} \; ; \quad \bar{X} \in (0,1)^n$$

And this problem can be solved as a mixed integer linear programming, since any max-min functional can be translated to a linear one. This can be stated, for this case as follows:

3.2.- *Theorem:* Let S = (P,C) be a Horn system with Fuzzy Certainty Factors associated to each rule i. Then the Certainty factor of the clause $\Lambda_{\{ p' \in P' \}} p' \rightarrow p''$ is:

$$\max b$$

Over

$$b \leq \alpha_i * x_i + L * (1 - x_i) : i = 1 .. n ;$$
$$b \geq 0 ;$$
$$\overline{H(S)} * \bar{F} \; R \; D \; ;$$
$$\bar{F} \geq 0 \; ,$$
$$\bar{F} \leq M * \bar{X} \; ,$$
$$\bar{X} \in (0,1)^n$$

Where $L > \max \{ \alpha_i : i = 1 .. n \}$, r_v is an equality for every vertex in $V - \{ v_{p'} : p' \in P' \}$ and no restriction for every vertex in $\{ v_{p'} : p' \in P' \}$, d_v is 1 for $v_{p''}$ and is equal to 0 for all vertices in $V - (\{ v_{p'} : p' \in P' \} \cup \{ p'' \})$, and M is a bound for f_i (like n, the number of rules in S).

□

5.- Conclusions:

In this paper we have shown the usefulness of looking at logical derivation and calculation of Certainty Factors in Expert and Knowledge-based Systems from a Combinatorial Optimization point of view . In particular, the problem of derivation with propositional Horn clauses with Fuzzy Certainty Factors is reduced, by theorem (4.7), to a mixed integer linear programming problem associated to the existence of a Unit Flow in Directed Hypergraph. Such result is not only of theoretical interest, but also practical, since many Knowledge-based Systems or big subsystems of them can be modelled or translated to propositional Horn systems, and Fuzzy Logic is one of the most widely used Certainty Factor Algebra

This work opens two lines for future research. First, the study of more general or structured combinatorial models for searching in Knowledge Bases with Uncertainty. Second, the inclusion of other Certainty Factor Algebras in the linear model.

References

[1] Berge, C., *Graphes et Hypergraphes,* Second Edition, Dunod, Paris, 1976.

[2] Buchanan, B. G., and Shortliffe, E. H. (Editors), *Rule-Based Expert Systems,* Addison Wesley, Massachusetts, 1984.

[3] Duda, R. O., Hart, P. E., and Nilson, N. J., "Subjective Bayesian Methods for Rule-Based Inference Systems", TR-124, Artificial Intelligence Center, Stanford Research Institute, California, 1976.

[4] Ford, L. R., and Fulkerson, D. R., *Flows in Networks,* Princeton University Press, New Jersey, 1962.

[5] Hayes-Roth, F., Waterman, D. A., and Lenat, D. (Editors), *Building Expert Systems,* Addison-Wesley, 1983.

[6] Horn, A., "On Sentences Which are True of Direct Unions of Algebras", *Journal of Symbolic Logic,* Vol. 16, Nº 14, 1951.

[7] Michalski, R. S., and Winston, P. H., "Variable Precision Logic", *Artificial Intelligence,* Vol. 29, Nº 2, pp. 121-146, 1986.

[8] Negoita, C. V., and Ralescu, D. A., *Applications of Fuzzy Sets to Systems Analysis,* John Wiley & Sons, 1975.

[9] Nilsson, N. J., "Probabilistic Logic", *Artificial Intelligence,* Vol. 28, Nº 1, pp. 71-87, 1986.

[10] Rockafellar, R. T., *Network Flows and Monotropic Optimization,* John Wiley and Sons, 1984.

[11] Torres, A., and Aráoz, D., "Combinatorial models for searching in Knowledge Bases", Acepted for publication in *Acta Cientifica Venezolana.*

[12] Zadeh L. A., "Fuzzy Sets", *Information and Control,* Vol. 8, pp. 338-353, 1965.