Search in Horn's knowledge bases and the Simplex algorithm

by

Julián Aráoz

Departamento de Procesos y Sistemas
Universidad Simón Bolívar

ABSTRACT

It has been proved that deduction in propositional Horn's clauses systems is equivalent to feasibility of a linear program associated to a directed hype graph flow model which correspond to the Horn's clauses systems. Here, we show that bottom up search is equivalent to solve the linear program using the Simplex algorithm and top down search correspond to the application of the dual Simplex algorithm.

1. Introduction

In the last few years, expert systems [7,9,17] 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.

To represent a given knowledge base various schemes such as associative networks [11], frames [14], rule based models [7,16], and many others, are used. 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, any knowledge representation model is essentially a combinatorial data structure, since it can be viewed as a discrete set with some defined combining rules. Now then, a deduction is also a search in such structure. This leads to the formulation of models for deduction in knowledge-based systems taken from the combinatorial optimization area. Even so, there are no publications about this use of combinatorial models, but the set-covering model used for abductive systems by Reggia, Nau, Wang and Peng [21,24,25,26]. However, since this area has grown very fast in last twenty years [19,23], we believe that the systematic use of this approach will be useful for improving the efficiency of inference engines, and will establish a positive exchange between the two disciplines.

In this paper, we deal with propositional Horn clauses. This is not the most general knowledge representation scheme, but an expressive and widely used one. 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 has been proposed [29]. In this paper we show that bottom up search is equivalent to solve linear program using the Simplex algorithm with artificial variables and top down search correspond to the application of the dual Simplex algorithm.

2. Paths and flows in directed hypergraphs

In order to work with the dual of linear problems formulated on the incidence matrix of a graph, Berge [5] 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. This combinatorial model may be useful for modelling in a series of applications where a generalization of directed graphs is needed. For example, directed hypergraphs have been used as an expressive representation for functional dependencies in databases [2,28], Petri nets [22], and associative networks [6]. On the other hand, most of combinatorial problems and some algorithms on directed graphs can also be extended to directed hypergraphs.

Definition 2.1: 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 $h(e)$.

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

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

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

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

where

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

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

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

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

Now we present a generalization of the problem of flow in networks [15] to directed hypergraphs. Most of notation and definitions are extended from Rockafellar [27].

Definition 2.4: 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.

Definition 2.5: Let H be a directed hypergraph and let F be a flow in H. The *divergence function* of F is $\mathrm{Div}_F : V \to \mathbb{R}$, where:

$$\mathrm{Div}_F(v) = \sum_{\{e \in E : v \in h(e)\}} F(e) - \sum_{\{e \in E : v \in t(e)\}} F(e).$$

Note that divergence function for v_i can be calculated by multiplying ith row of the incidence matrix of H by the vector F. Therefore:

Lemma 2.6: *Let H be a directed hypergraph and F a flow in H, then:*

$$\overline{H} * \overline{F} = \overline{\mathrm{Div}_F}. \qquad \Box$$

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.

Definition 2.7: 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 *unit flow* from V' into V'' iff:

i) $\forall\, v \in V''\ (\mathrm{Div}_F(v) = 1)$,

ii) $\forall\, v \in V - (V' \cup V'')\ (\mathrm{Div}_F(v) = 0)$,

iii) $\forall\, e \in E\ (F(e) \geq 0)$.

Therefore F is an unit flow if and only if it is a feasible solution to the linear program defined in 2.7.

3. Horn systems and rule hypergraphs

Even when a variety of knowledge representation schemes exist [7,11,14,16], perhaps the most precise and general model is given by symbolic logic. In this way a knowledge base is simply a well formed formula in a formal logic system. It is known that any logical formula can be translated to an equivalent one in clausular form [20]. In a propositional system a clause is nothing more than a statement of the form: $\bigwedge_{\{p \in P\}} p \rightarrow \bigvee_{\{p' \in P'\}} p'$, where P and P' are both sets of atomic propositions, i.e. instances of propositional variables. A formula is in clausular form if it is a finite conjunction of clauses.

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 [18] in which there is only one atomic proposition at the right side, i.e. clauses of the form: $\bigwedge_{\{p \in P\}} p \rightarrow p'$. If a formula is a finite conjunction of Horn clauses it's said that it is in Horn clausular form. Note that a clause of the form $\bigwedge_{\{p \in P\}} p \rightarrow p'$, where $p' \in P$ is a tautology, so any formula in Horn clausular form can be assumed to have not such clauses.

Definition 3.1: 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.

Definition 3.2: 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 $(T, \{v_k\})$ will be denoted as $T \rightarrow v_k$. Any subset of V will be an *information state* of H.

Then every atomic proposition p of a logical system will be represented by an atom v_p and every clause of the form: $\bigwedge_{\{p\in P\}} p \rightarrow p'$ where $(p' \notin P)$ 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 $H(S)$.

Fagin [12,13] have established that derivation in Horn systems is equivalent to functional dependencies [1] derivation in a relational database model [8], that is, Fagin showed that a Horn clause is logically implied by a given set of Horn clauses iff the same holds interpreting each clause as a functional dependency. In other works [2,3,4,28], it's been established that derivation of functional dependencies is also equivalent to the determination of membership of a clause to the closure of the corresponding directed hypergraph.

Now, we introduce a path-based concept of derivation which is equivalent to deduction in Horn's systems [29].

Definition 3.3: 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 $t(e) \subseteq O$.

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.

Note also that the increasing path correspond to a bottom up search in the Horn's system.

Theorem 3.4 [29]: *Let $S = (P, C)$ be a Horn's system with an information state O and an atom d. The vertices of $H(S)$ are P and the edges of $H(S)$ are C. Let Q be the set of solutions F to the linear program*

$$\begin{aligned}&\mathrm{Div}_F(d) = 1.\\ &\forall\, v \in P - (O \cup \{d\})\ (\mathrm{Div}_F(v) = 0),\ \text{and}\\ &\forall\, e \in C\ (F(e) \geq 0).\end{aligned}$$

Then, the following statements are equivalent:

i) d is deductible from O in S;

ii) There is a derivation of d from O in $H(S)$;

iii) There exists an integral F in Q;

iv) Q is not empty. □

4. Bottom up search and the Simplex algorithm

In this section we will show that the Simplex algorithm with artificial variables applied to the system Q in Theorem 3.4 behaves exactly as a bottom up search. Hence bottom up search could be considered a simplex algorithm which take advantage of the structure of the problem in the same way that the transportation algorithm does.

Theorem 4.1: *Bottom up search is a Simplex algorithm.*

Proof. We prove the theorem by showing, using induction, that the derivation of d from O,

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

is obtained with $s_0 = O$ and at any pivoting step t, e_t is the edge corresponding to the pivot column. This is equivalent to prove that at step t we can pivot in a column only if all the preconditions of this column are in s_{t-1}.

Note that after adding the artificial variables, the problem is to minimize the summation of the artificial variables. At any pivoting step, a variable j could enter the basis only if its transformed cost $z_j - c_j > 0$, i.e. $z_j > 0$ since $c_j = 0$ for the real variables.

Let $z_j = \sum_{i \in I} h_{ij}$ with $I \subseteq P$. Then $z_j > 0$ is equivalent to $z_j = 1$ and $h_{ij} = 0$ for all $i \in I$ but one i' with $h_{i'j} = 1$, because h_{ij} is equal to 0, 1 or -1 with exactly one 1. That is $z_j > 0$ iff all preconditions of j are in $P \setminus I$.

(i) For $t = 1$, the basis is the identity matrix, hence $z_j = \sum_{i \in P-O} h_{ij} > 0$ iff all preconditions of j are in $O = s_0$.
Without loss of generality assume that $z_e = 1 = h_{e1}$. Hence, after pivoting, we have a new basis $\begin{vmatrix} 1 & O \\ O & I \end{vmatrix}$ where $\begin{vmatrix} 1 \\ O \end{vmatrix}$ is the column of edge e and $\begin{vmatrix} O \\ I \end{vmatrix}$ are the columns of the artificial variables. The path is now $Oe(O \cup \{h(e)\})$.

(ii) Assume that at pivoting step t we have built the path $s_0 e_1 s_1 \dots e_{t-1} s_{t-1}$ and we have rearranged the basis as $\begin{vmatrix} B & O \\ O & I \end{vmatrix}$ where $\begin{vmatrix} B \\ O \end{vmatrix}$ are the columns of $e_1, \dots, e_{t-1}$ and $\begin{vmatrix} O \\ I \end{vmatrix}$ are the columns of the artificial variables. Hence the inverse of the basis is $\begin{vmatrix} B^{-1} & O \\ O & I \end{vmatrix}$ and $z_j = \sum_{i \in P-s_{t-1}} h_{ij}$.
We have $z_j > 0$ iff all the preconditions of j are in s_{t-1}. Moreover, if we pivot in $z_j > 0$, since $h_{ij} = 0$ for the remaining artificial variables we keep the same structure for step $t+1$. □

It is not difficult to show that in Theorem 3.4 we can change the =’s by ≥’s and the Theorem is still valid. If we add the surplus variables we have an initial basis with the surplus variable corresponding to d equals to -1. If we apply the dual Simplex method, it will behave like a top down search.

References

[1] Armstrong, W.W.: Dependency structures of data base relationships. Information Processing Letters, Vol. 74, 1974, pp. 580–583.

[2] Ausiello, G., D'Atri, A., Saccà, D.: Graph algorithms for functional dependency manipulation. Journal of the ACM, Vol. 30, No. 4, October 1983, pp 752–766.

[3] Ausiello, G., D'Atri, A., Saccà, D.: Minimal representation of directed hyper-

graphs. SIAM Journal of Computing, Vol. 15, No. 2, May 1986, pp. 418–431.
[4] Beeri, C., Bernstein, P.A.: Computational problems related to the design of normal form relational schemes. ACM Transactions on Database Systems, Vol. 4, 1979, pp. 30–59.
[5] Berge, C.: Graphes et hypergraphes. Second edition, Dunod, Paris, 1976.
[6] Boley, H.: Directed recursive labelnode hypergraphs: A new representation language. Artificial Intelligence, Vol. 9, 1977, pp. 49–85.
[7] Buchanan, B.G., Shortliffe, E.H. (Editors): Rule-based expert systems. Addison-Wesley, Massachusetts, 1984.
[8] Codd, E.F.: A relational model of data for large shared data banks. Communications of the ACM, Vol. 13, No. 6, 1970, pp. 377–387.
[9] Charniak, E., McDermott, D.: Introduction to artificial intelligence. Addison-Wesley, Massachusetts, 1986.
[10] Dantzig, G.B.: Linear programming and extensions. Princeton University Press, New Jersey, 1963.
[11] Deliyanni, A., Kowalski, R.A.: Logic and semantic networks. Communications of the ACM, Vol. 22, No. 3, 1979, pp. 184–192.
[12] Fagin, R.: Functional dependencies in a relational database and propositional logic. IBM Journal of Research and Development, Vol. 21, No. 6, November 1977, pp. 534–544.
[13] Fagin, R.: Horn clauses and database dependencies. Proceedings of the 12th Annual ACM Symposium on the Theory of Computing, California, 1980, pp. 33–64.
[14] Filkes, R., Kehler, T.: The role of frame-based representation in reasoning. Communications of the ACM, Vol. 28, No. 9, 1985, pp. 904–920.
[15] Ford, L.R., Fulkerson, D.R.: Flows in networks. Princeton University Press, New Jersey, 1962.
[16] Hayes-Roth, F.: Rule-based systems. Communications of the ACM, Vol. 28, No. 9, 1985, pp. 921–932.
[17] Hayes-Roth, F., Waterman, D.A., Lenat, D. (Editors): Building expert systems. Addison-Wesley, Massachusetts, 1983.
[18] Horn, A.: On sentences which are true of direct unions of algebras. Journal of Symbolic Logic, Vol. 16, No. 14, 1951.
[19] Lawler, E.L.: Combinatorial optimization: Networks and matroids. Holt, Rinehart and Winston, 1984.
[20] Manna, Z.: Mathematical theory of computation. McGraw-Hill Inc., 1974.
[21] Nau, D.S., Reggia, J.A.: Relationships between deductive and abductive inference in knowledge-based diagnostic problem-solving. Expert Database systems, The Benjamin/Cummings Publishing Company, 1986, pp. 549–558.
[22] Petri, C.A.: Communication with automata. Tech. Rep. RADC-TR-65377, Vol. 1, Griffith Air Force Base, New York, 1966.
[23] Papadimitriou, C.H., Steiglitz, K.: Combinatorial optimization. Algorithms and complexity. Prentice-Hall Inc., New Jersey, 1982.
[24] Reggia, J.A., Nau, D.S., Wang, P.Y.: Diagnostic expert systems based on a set covering model. International Journal of Man-Machine Studies, Academic Press Inc., London, 1983, pp. 437–460.
[25] Reggia, J.A., Nau, D.S., Wang, P.Y.: A formal model of diagnostic inference. I. Problem formulation and decomposition. Information sciences, Vol. 37, 1985, pp. 227–256.
[26] Reggia, J.A., Nau, D.S., Wang, P.Y., Peng, Y.: A formal model of diagnostic in-

ference. II. Algorithmic solution and application. Information Sciences, Vol. 37, 1985, pp. 227–256.

[27] Rockafellar, R.T.: Network flows and monotropic optimization. John Wiley and Sons, 1984.

[28] Saccà, D.: Closures af database hypergraphs. Journal of the ACM, Vol. 32, No. 4, October 1985, pp. 774–803.

[29] Torres, A., Aráoz, J.: Combinatorial models for searching in knowledge bases. Acta Científica Venezolana, Vol. 39, 1988, pp. 387–394.