DIMACS Technical Report 94-9
March 1994

Decompositions of Partially Defined Boolean Functions[1]

by

Endre Boros[2]
RUTCOR, Rutgers University
New Brunswick, NJ 08904

Vladimir Gurvich[3]
RUTCOR, Rutgers University
New Brunswick, NJ 08904

Peter L. Hammer[2]
RUTCOR, Rutgers University
New Brunswick, NJ 08903

Toshihide Ibaraki
Department of Applied Mathematics and Physics
Kyoto University, Kyoto, Japan 606

Alexander Kogan[2]
Faculty of Management and RUTCOR, Rutgers University
New Brunswick, NJ 08903

[1]Research supported in part by the Office of Naval Research (Grants N0001492F1375 and N0001492F4083) and by the Air Force Office of Scientific Research (Grant F49620-93-1-0041).
[2]Permanent Member
[3]Visiting RUTCOR, on leave from the International Institute of Earthquake Prediction Theory and Mathematical Geophysics, Moscow, Russia.

DIMACS is a cooperative project of Rutgers University, Princeton University, AT&T Bell Laboratories and Bellcore.

DIMACS is an NSF Science and Technology Center, funded under contract STC–91–19999; and also receives support from the New Jersey Commission on Science and Technology.

ABSTRACT

The problem of recognizing decomposability of incompletely defined Boolean relations is considered. The results include polynomial time algorithms for certain important types of decompositions, as well as NP-completeness proofs for more complex structures.

1 Introduction

In this paper we address the problem of identifying a meaningful structure in a given set of data. In general, learning is a formation process of a theory that can explain the given set of data. This process can be greatly enhanced if the underlying structure that holds among the relevant variables is known in advance. The problem can also be regarded as one form of knowledge acquisition – an important topic in applications of artificial intelligence.

For simplicity, we concentrate on sets of deterministic data, which are encoded as finite sets of $0-1$ vectors of n components. Such a data set consists of a set of positive examples, and a set of negative examples; thus the theory we want to build is actually a Boolean function f, which can separate exactly these two sets. By "structure" we mean a decomposition of the set of all variables into small groups, such that f can be composed of subfunctions, each of which depends on one of the subgroups of variables. To be precise, the problem considered here is to decide whether a given set of data permits a theory having a prescribed decomposition structure among the variables.

In addition to learning theory, such decompositions can be encountered in various applications such as relational databases, where decomposition not only saves storage but also speeds up future queries (see e.g. [6, 9]), VLSI design where decomposition can help both minimization and testability (see e.g. [3, 10]), etc.

To formulate the problem precisely, we introduce the notion of a *partially defined Boolean function* (pdBf), which is defined as a pair of disjoint sets $(\mathbf{T}, \mathbf{F})$ of binary n-vectors. The set $\mathbf{T}$ denotes the set of true vectors (or positive examples) and $\mathbf{F}$ is the set of false vectors (or negative examples). A Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$ is called an *extension* (or a theory) of the pdBf $(\mathbf{T}, \mathbf{F})$, if $f(\mathbf{x}) = 1$ for all $\mathbf{x} \in \mathbf{T}$ and $f(\mathbf{y}) = 0$ for all $\mathbf{y} \in \mathbf{F}$.

Evidently, a pdBf $(\mathbf{T}, \mathbf{F})$ has an extension if and only if the sets $\mathbf{T}$ and $\mathbf{F}$ are disjoint. It may not be evident, however, to find out whether a given pdBf has an extension with a specified decomposition structure. There is a vast literature on the decomposition of completely specified Boolean functions, with particular attention given to the monotone case (see [12]). Unfortunately, these results cannot be applied directly to partially defined Boolean functions. Typically, a pdBf may have exponentially many different extensions, many of which may have exponential size, and the level of their decomposability as well as the complexity of its recognition may vary wildly.

In this paper we formulate the problem of deciding the existence of an extension with a specified decomposition structure (without the need of a complete description of such an extension), and study its computational complexity. We obtain computationally efficient algorithms in some cases, and prove NP-completeness in some other cases.

2 Main Results

In this section we shall state the main results, leaving the technical details and proofs for Section 3.

Let $V = \{x_1, x_2, \ldots, x_n\}$ denote the set of Boolean variables (attributes, or components). For a Boolean vector $\mathbf{x}$ and for a subset $S \subseteq V$ of the variables, let $\mathbf{x}[S]$ denote its projection on S, i.e., the vector restricted to the variables in S. For instance, if $\mathbf{x} = (1011100)$, $\mathbf{y} = (0010011)$, and $S = \{x_2, x_3, x_5\}$, then $\mathbf{x}[S] = (011)$, and $\mathbf{y}[S] = (010)$. To simplify notation, for a Boolean function h depending only on variables of $S \subseteq V$, we write simply $h(S)$ instead of $h(\mathbf{x}[S])$.

Given a pdBf $(\mathbf{T}, \mathbf{F})$ and a family of subsets $\{S_i | i \in \{1, \ldots, k\}, S_i \subseteq V\}$, we formulate the decomposability problem as the problem of deciding whether there exists an extension f of $(\mathbf{T}, \mathbf{F})$ for which there exist Boolean functions h_1, ..., h_k, and g such that each h_i depends only on variables of S_i $(i = 1, ..., k)$, $g : \{0,1\}^k \to \{0,1\}$, and

$$f = g(h_1(S_1), h_2(S_2), \ldots, h_k(S_k)). \tag{1}$$

If S_i is a singleton, say $S_i = \{x_j\}$, then without loss of generality we may assume that $h_i(\{x_j\}) = x_j$. If there are singleton subsets $\{x_{j_1}\}$, $\{x_{j_2}\}$, ..., $\{x_{j_m}\}$, we write $S_k = \{x_{j_1}, x_{j_2}, \ldots, x_{j_m}\}$ and

$$f = g(h_1(S_1), h_2(S_2), \ldots, h_{k-1}(S_{k-1})), S_k). \tag{2}$$

Let us note that $S_1, S_2, \ldots, S_k$ are not assumed to be disjoint, and that it is also possible that some of these subsets are equal, e.g. $S_1 = S_2$.

The specification given by (1) or by (2) will be called a *scheme*. In these terms, our objective is to find out whether a given pdBf has an extension of a given scheme, i.e. whether it has a representation of the type (1) or (2) It is also important to establish whether this question can be answered in time which is polynomial in n and $m = |\mathbf{T}| + |\mathbf{F}|$.

Let us remark that establishing the existence of an extension of a given scheme does not require the complete specification of the functions h_1, h_2, ..., h_k and g. It is sufficient to determine binary values $h_j(\mathbf{x}[S_j])$ for $j = 1, ..., k$ and $\mathbf{x} \in \mathbf{T} \cup \mathbf{F}$ such that the sets $\mathbf{T}' = \{(h_1(\mathbf{x}[S_1]), h_2(\mathbf{x}[S_2]), \ldots, h_k(\mathbf{x}[S_k])) \,|\, \mathbf{x} \in \mathbf{T}\}$ and $\mathbf{F}' = \{(h_1(\mathbf{y}[S_1]), h_2(\mathbf{y}[S_2]), \ldots, h_k(\mathbf{y}[S_k])) \,|\, \mathbf{y} \in \mathbf{F}\}$ are disjoint. Indeed, this condition is both necessary and sufficient for the existence of an extension g of the pdBf $(\mathbf{T}', \mathbf{F}')$.

Let us state now the main result of this paper.

Theorem 1 *Let $(\mathbf{T}, \mathbf{F})$ be a given pdBf, $n = |V|$ and $m = |\mathbf{T}| + |\mathbf{F}|$. Then the existence of extensions of the following schemes*

(a) $f = g(h_1(S_1), S_2)$,

(b) $f = g(h_1(S_1), h_2(S_2))$, *and*

(c) $f = g(h_1(S_1), h_2(S_2), S_3)$ *if* $|S_3| = O(\max(\log\log n, \log\log m))$,

can be decided in time polynomial in n and m.

In general, deciding the existence of extensions of the schemes *(d)* $f = g(h_1(S_1), h_2(S_2), S_3)$*, or*

(e) $f = g(h_1(S_1), h_2(S_2), h_3(S_3))$

is NP-complete.

In the next section we give complete proofs for (c), (d) and (e) of the above theorem. Obviously, case (c) implies cases (a) and (b), thus no separate proofs are required for these. Note however that in cases (a) and (b) there exist more efficient algorithms than what follows from (c), see [2].

Let us remark that it may be important to consider additional restrictions on the properties of g and h_i ($i \in \{1, \dots, k\}$), e.g. one may require these functions to be positive, Horn, etc. For instance, if all the functions h_1, ..., h_k and g are required to be positive, we shall call the specification (1) or (2) a *positive scheme*. It is shown in [2] that it can be decided in polynomial time whether a given pdBf $(\mathbf{T}, \mathbf{F})$, has an extension of positive schemes (a), (b), (d), or (e).

3 Proofs

To a given pdBf $(\mathbf{T}, \mathbf{F})$ and a given scheme $f = g(h_1(S_1), \dots, h_k(S_k))$ or $f = g(h_1(S_1), \dots,$ $g_{k-1}(S_{k-1}), S_k)$ we associate a *structure hypergraph* $H = (V, E)$ defined as follows. Let

$$\begin{aligned} V_i &= \{\mathbf{x}[S_i] \mid \mathbf{x} \in \mathbf{T} \bigcup \mathbf{F}\},\ i = 1, 2, \dots, k, \\ E_1 &= \{(\mathbf{x}[S_1], \mathbf{x}[S_2], \dots, \mathbf{x}[S_k]) \mid \mathbf{x} \in \mathbf{T}\},\ \text{ and} \\ E_0 &= \{(\mathbf{x}[S_1], \mathbf{x}[S_2], \dots, \mathbf{x}[S_k]) \mid \mathbf{x} \in \mathbf{F}\}, \end{aligned} \tag{3}$$

and let $V = V_1 \cup V_2 \cup \ldots \cup V_k$ and $E = E_1 \cup E_0$.

The hyperedges $e = (v_1, ..., v_k) \in E_1$ and $e' = (v'_1, ..., v'_k) \in E_0$ will be refered simply as *true-edges* and *false-edges*, respectively. In pictures true- and false-edges will be shown as solid and dotted lines, respectively.

3.1 Solvable Cases of Scheme $f = g(h_1(S_1), h_2(S_2), S_3)$

In this section we present an algorithm that, given a pdBf $(\mathbf{T}, \mathbf{F})$, decides the existence of an extension of the scheme $f = g(h_1(S_1), h_2(S_2), S_3)$. It is also shown that if

$$|S_3| = O(\log \log m + \log \log n),$$

where $m = |\mathbf{T} \cup \mathbf{F}|$ and $n = |V|$, then this algorithm runs in polynomial time.

Let us introduce the notations $T(g)$ and $F(g)$ for the sets of all true and false vectors of the Boolean function g.

To illustrate the basic idea of the algorithm, let us consider first a simple observation on Boolean functions of two variables:

Proposition 1 *For any Boolean function g in two variables, the sets $T(g)$ and $F(g)$ can be characterized by quadratic Boolean equations. The number of equations for each set is not more than* 3.

For example, if $g = \overline{x}\,y \vee x\overline{y}$ (exclusive or), then

$$T(g) = \{(0,1),(1,0)\}, \text{ and}$$
$$F(g) = \{(0,0),(1,1)\},$$

and both of these sets can be characterized by quadratic Boolean equations, e.g. $T(g)$ by $\overline{x}\,\overline{y} = 0$, $xy = 0$, and $F(g)$ by $\overline{x}\,y = 0$, $x\overline{y} = 0$. Similar argument can be applied to any Boolean function in two variables.

Now, given a pdBf $(\mathbf{T},\mathbf{F})$ and the scheme $f = g(h_1(S_1), h_2(S_2), S_3)$, let us construct its structure hypergraph $H = (V, E)$. As remarked in Section 2, an extension of scheme $f = g(h_1(S_1), h_2(S_2), S_3)$ exists if and only if there are binary assignments h_1 and h_2 to the vertices of V_1 and V_2, respectively, for which $T_w \cap F_w = \emptyset$ for every $w \in V_3$, where

$$\begin{aligned} T_w &= \{(h_1(u), h_2(v)) | (u,v,w) \in E_1\}, \text{ and} \\ F_w &= \{(h_1(u), h_2(v)) | (u,v,w) \in E_0\}. \end{aligned}$$

This means simply that for every $w \in V_3$ the sets T_w and F_w define a Boolean function $g_w(x,y)$ in two variables.

Although we do not know the functions g_w in advance, we can test all possibilities, since there are only $2^{2^2} = 16$ different functions in two variables.

ALGORITHM G-3

1. Construct the structure hypergraph H for the given pdBf $(\mathbf{T}, \mathbf{F})$, and associate Boolean variables X_u and Y_v to all vertices $u \in V_1$ and $v \in V_2$.
2. For each $w \in V_3$, select a two-variable Boolean function g_w (there are 16 possibilities for each w, and $16^{|V_3|}$ cases in total.)
3. For each selection of functions g_w, $w \in V_3$, set up the following system of quadratic Boolean equations:
   - For every true-edge $(u, v, w) \in E_1$ add the equations characterizing $T(g_w)$ using the variables X_u and Y_v.
   - For every false-edge $(u, v, w) \in E_0$ add the equations characterizing $F(g_w)$ using the variables X_u and Y_v.

   Let $B = 0$ denote the resulting system of quadratic Boolean equations.
4. If $B = 0$ has a solution for one of the selections, then $h_1(u) = X_u$ for $u \in V_1$ and $h_2(v) = Y_v$ for $v \in V_2$ show the existence of an extension of the required scheme, thus we can STOP.

   If none of the selections yield a consistent equation, then we can conclude that the pdBf $(\mathbf{T}, \mathbf{F})$ has no extension of the desired scheme.

Since the number of equations in each of the systems is not more than $3|E| = 3m$, and since a quadratic Boolean equation can be solved in time linear in its size, the running time of algorithm G-3 is $O(m16^{|V_3|})$. This proves the following result.

Proposition 2 *Deciding if a pdBf $(\mathbf{T}, \mathbf{F})$ has an extension of scheme $f = g(h_1(S_1), h_2(S_2), S_3)$ can be done in polynomial time if $|V_3| = O(\max(\log n, \log m))$, in particular, if $|S_3| = O(\max(\log\log n, \log\log m))$.*

3.2 Scheme $f = g(h_1(S_1), h_2(S_2), S_3)$

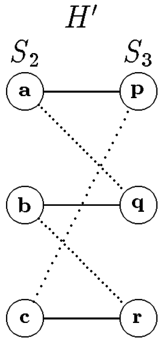
In this section we show that to decide about the existence of an extension of scheme $f = g(h_1(S_1), h_2(S_2), S_3)$ is, in general, NP-complete.

To show this we shall reduce our problem to the so called *hypergraph 2-coloring* problem, which is known to be NP complete (see e.g. [7]).

HYPERGRAPH 2-COLORING
Instance: Hypergraph $H^* = (V^*, E^*)$, with $|\Delta| = 3$ for every $\Delta \in E^*$.
Question: Is H^* 2-colorable, i.e. does there exist an assignment $c : V^* \to \{0, 1\}$ such that for every $\Delta \in E^*$ there exist elements $u, v \in \Delta$ with $c(u) = 1$ and $c(v) = 0$.

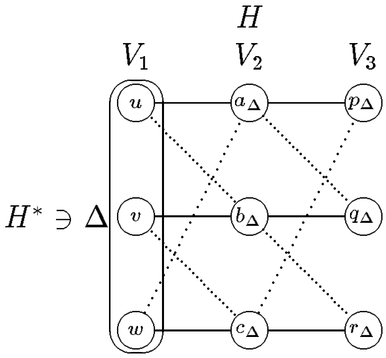
In the reduction the following small pdBf $(\mathbf{T}', \mathbf{F}')$ will play a key role.

	S_2	S_3
$\mathbf{T}'$	**a**	**p**
	b	**q**
	c	**r**
$\mathbf{F}'$	**a**	**q**
	b	**r**
	c	**p**



The letters $\mathbf{a}, \mathbf{b}, \ldots, \mathbf{r}$ in this example refer to distinct Boolean vectors of appropriate dimensions. It is easy to verify that this pdBf $(\mathbf{T}', \mathbf{F}')$ has no extension of scheme $f' = g'(h_2(S_2), S_3)$: Since $(\mathbf{a}, \mathbf{p})$ is a true-edge and $(\mathbf{c}, \mathbf{p})$ is a false edge, having the same components in S_3, $h_2(\mathbf{a}) \neq h_2(\mathbf{c})$ is necessary. Similarly, we can see that the non-equalities $h_2(\mathbf{b}) \neq h_2(\mathbf{c})$ and $h_2(\mathbf{a}) \neq h_2(\mathbf{b})$ must also hold for any extension of the required scheme. However, this leads to a contradiction, since h_2 can have only two different values.

Given a hypergraph H^*, we construct now a pdBf $(\mathbf{T}, \mathbf{F})$ as follows: To every $\Delta \in E^*$, $\Delta = \{u, v, w\}$, we associate nodes a_Δ, b_Δ, c_Δ, p_Δ, q_Δ, and r_Δ and define true- and false-edges as in the figure below.



Formally, we define the structure hypergraph $H = (V_1 \cup V_2 \cup V_3, E_0 \cup E_1)$ by setting

$$
\begin{aligned}
V_1 &= V^*, \\
V_2 &= \bigcup_{\Delta \in E^*} \{a_\Delta, b_\Delta, c_\Delta\}, \\
V_3 &= \bigcup_{\Delta \in E^*} \{p_\Delta, q_\Delta, r_\Delta\}, \\
E_1 &= \{(u, a_\Delta, p_\Delta), (v, b_\Delta, q_\Delta), (w, c_\Delta, r_\Delta) | \Delta = \{u, v, w\} \in E^*\}, \text{ and} \\
E_0 &= \{(w, a_\Delta, q_\Delta), (u, b_\Delta, r_\Delta), (v, c_\Delta, p_\Delta) | \Delta = \{u, v, w\} \in E^*\}.
\end{aligned}
\tag{4}
$$

Let finally $(\mathbf{T}, \mathbf{F})$ be a pdBf which has H as its structure hypergraph. (Obviously, using $|S_1| > \log |V_1| = \log |V^*|$ variables, one can construct a binary encoding, which associates different binary strings to each of the vertices in V_1. Analogously, using $|S_2| = |S_3| > \log|V_2| = \log|V_3| = \log 3|E^*|$ variables, the vertices of V_2 and V_3 can also be encoded as different binary strings.)

We claim that this pdBf $(\mathbf{T},\mathbf{F})$ has an extension of scheme $f = g(h_1(S_1), h_2(S_2), S_3)$ if and only if h_1 induces a proper 2-coloring of the hypergraph H^*. For, if h_1 is not defining a proper coloring of H^*, then there exists a hyperedge $\Delta = (u,v,w) \in E^*$ for which $h_1(u) = h_1(v) = h_1(w)$, implying that the pdBf $(\mathbf{T}',\mathbf{F}')$, consisting of the following rows,

	S_1	S_2	S_3
$\mathbf{T}'$	u	a_Δ	p_Δ
	v	b_Δ	q_Δ
	w	c_Δ	r_Δ
$\mathbf{F}'$	u	a_Δ	q_Δ
	v	b_Δ	r_Δ
	w	c_Δ	p_Δ

must have an extension of the scheme $f = g'(h_2(S_2), S_3)$, which is impossible as we observed above.

Conversely, let h_1 be any binary assignment to the vertices $V^* = V_1$, defining a proper 2-coloring of H^*. We show below that this assignment on V_1 can be extended to V_2, proving the existence of an extension of $(\mathbf{T},\mathbf{F})$ of the required scheme.

It follows from the definitions (4) that for every node $d \in V_2$ there exists a unique hyperedge $\Delta \in E^*$, and a unique node $u \in \Delta$, such that $d \in \{a_\Delta, b_\Delta, c_\Delta\}$, and $\{u,d\} \subset e$ for some true-edge $e \in E_1$. Let us define then $h_2(d) = h_1(u)$. It is easy to see then that any two hyperedges $e \in E_1$, $e' \in E_0$ of H with $e \bigcap V_3 = e' \bigcap V_3 = s$, have different pairs of bits assigned by (h_1,h_2). That is, if $e = (u,d,s)$ and $e' = (u',d',s)$, then either $h_1(u) \neq h_1(u')$ or $h_2(d) \neq h_2(d')$. This proves that $(\mathbf{T},\mathbf{F})$ has an extension of the desired scheme.

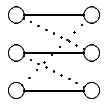
The arguments above show that the pdBf $(\mathbf{T},\mathbf{F})$ has an extension of the scheme $f = g(h_1(S_1), h_2(S_2), S_3)$ if and only if the hypergraph H^* has a proper 2-coloring. Since the decision problem about the existence of an extension of scheme $f = g(h_1(S_1), h_2(S_2), S_3)$ is obviously in NP, we get

Proposition 3 *Deciding if a pdBf $(\mathbf{T},\mathbf{F})$ has an extension of scheme*

$$f = g(h_1(S_1), h_2(S_2), S_3)$$

is NP-complete.

3.3 Scheme $f = g(h_1(S_1), h_2(S_2), h_3(S_3))$

By an argument similar to the one in the previous subsection, we show that this case is also NP-complete, by reducing the problem to hypergraph 2-coloring. As in the previous construction, the key in this reduction will be again the structure  , which does not have an extension of the scheme $g(h_2(S_2), h_3(S_3))$.

Let us recall now the construction of a pdBf $(\mathbf{T}, \mathbf{F})$ associated to a hypergraph $H^* = (V^*, E^*)$, in which $|\Delta| = 3$ for every edge $\Delta \in E^*$.

First, we associate to every edge $\Delta = \{u, v, w\} \in E^*$ six new nodes a_Δ, b_Δ, c_Δ, p_Δ, q_Δ, and r_Δ, and define

$$\begin{aligned} V_1 &= V^*, \\ V_2 &= \bigcup_{\Delta \in E^*} \{a_\Delta, b_\Delta, c_\Delta\}, \quad \text{and} \\ V_3 &= \bigcup_{\Delta \in E^*} \{p_\Delta, q_\Delta, r_\Delta\}. \end{aligned} \tag{5}$$

Next, we define the true- and false-edges of the structure hypergraph $H = (V_1 \cup V_2 \cup V_3, E_0 \cup E_1)$ by

$$\begin{aligned} E_1 &= \{(u, a_\Delta, p_\Delta), (v, b_\Delta, q_\Delta), (w, c_\Delta, r_\Delta) | \Delta = \{u, v, w\} \in E^*\}, \text{ and} \\ E_0 &= \{(w, a_\Delta, q_\Delta), (u, b_\Delta, r_\Delta), (v, c_\Delta, p_\Delta) | \Delta = \{u, v, w\} \in E^*\}. \end{aligned} \tag{6}$$

Finally, we set the pdBf $(\mathbf{T}, \mathbf{F})$ as one of the pdBfs which have H as their structure hypergraph. Without any loss of generality we may assume that the sets S_1, S_2 and S_3 are disjoint and that $|S_1| = 1 + \lceil \log |V^*| \rceil$ and $|S_2| = |S_3| = 1 + \lceil \log 3|E^*| \rceil$, since $|V_1| = |V^*|$, and $|V_2| = |V_3| = 3|E^*|$, and thus all of the vertices can be encoded as distinct binary strings of appropriate lengths.

We can argue again, as in the previous section, that if all the nodes of some hyperedge $\Delta \in E^*$ are assigned the same binary value α, then no extension of scheme $f = g(\alpha, h_2(S_2), h_3(S_3))$ exists. This implies that in any extension of $(\mathbf{T}, \mathbf{F})$ of scheme $f = g(h_1(S_1), h_2(S_2), h_3(S_3))$ the function h_1 must induce a proper 2-coloring of the hypergraph H^*.

On the other hand, if h_1 defines a 2-coloring for H^*, then we can define h_2 and h_3 uniquely on V_2 and V_3 by setting $h_2(d) = h_3(s) = h_1(u)$ for all $(u, d, s) \in E_1$. For illustration, consider the example in Figure 1.

It is not difficult to see now that for every false-edge $(u, d, s) \in E_0$, $(h_1(u), h_2(d), h_3(s)) \notin \{(0, 0, 0), (1, 1, 1)\}$, i.e. no binary triple encodes simultaneously a true-edge and a false-edge of $(\mathbf{T}, \mathbf{F})$. This implies the existence of an extension of scheme $f = g(h_1(S_1), h_2(S_2), h_3(S_3))$.

It follows that the pdBf $(\mathbf{T}, \mathbf{F})$ has an extension of scheme $f = g(h_1(S_1), h_2(S_2), h_3(S_3))$ if and only if the hypergraph H^* is 2-colorable. Since the decision problem about the existence of an extension of this scheme is obviously in NP, we have

Proposition 4 *Deciding if a pdBf* $(\mathbf{T}, \mathbf{F})$ *has an extension of scheme*

$$f = g(h_1(S_1), h_2(S_2), h_3(S_3))$$

is NP-complete.

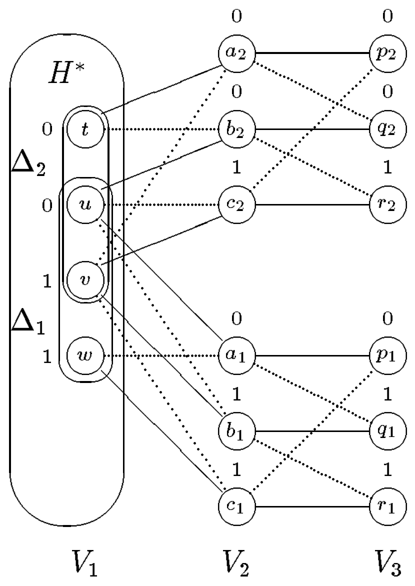


Figure 1:

References

[1] R. L. Ashenhurst, The decomposition of switching functions, *Proc. Int. Symp. Theory of Switching*, Harvard University, pp. 74-116, 1957.

[2] E. Boros, V. Gurvich, P.L. Hammer, T. Ibaraki, and A. Kogan, Structural analysis of partially defined Boolean functions, manuscript, 1993.

[3] R.K. Brayton, G.D. Hachtel, and A.L. Sangiovanni-Vincentelli, Multilevel logic synthesis, *Proc. IEEE* **78** (2) (1990).

[4] Y. Crama, P.L. Hammer and T. Ibaraki, Cause-effect relationships and partially defined Boolean functions, *Annals of Operations Research* **16** (1988) 299-325.

[5] R. Dechter, Decomposing a relation into a tree of binary relations, *J. Comput. Syst. Sci.*, Vol. 41, pp. 2-24, 1990.

[6] R. Dechter and J. Pearl, Structure identification in relational data, *Artificial Intelligence*, **58**, pp. 237-270, 1992.

[7] M.R. Garey and D.S. Johnson, *Computers and Intractability: A guide to the theory of NP-completeness*, New York, Freeman, 1979.

[8] T. Ibaraki and T. Kameda, A theory of coteries: Mutual exclusion in distributed systems, *IEEE Trans. on Parallel and Distributed Systems*, Vol.4, No.7, pp. 779-794, 1993.

[9] D. Maier, *The theory of relational databases* (Computer Science Press, Rockville, MD, 1983).

[10] R. Murgai, R.K. Brayton and A. Sangiovanni-Vincentelli, Some results on the complexity of Boolean Functions for Table Look Up Architectures.

[11] J. R. Quinlan, Induction of decision trees, *Machine Learning*, Vol. 1, pp. 81-106, 1986.

[12] K.G. Ramamurthy, *Coherent Structures and Simple Games*, Kluwer Academic Publishers, 1990.

[13] L. S. Shapley, On committees, *New Methods of Thought and Procedures*, edited by F. Zwycky and A. G. Wilson, New York, Springer, 1967.