A SYSTEM OF PROGRAMS FOR THE SYNTHESIS OF SWITCHING CIRCUITS USING THE METHOD OF DISJOINT STARS

by

Ryszard S. Michalski
Zenon Kulpa

Information Processing 71, North-Holland Publishing Company, pp. 61-65, 1972.

INFORMATION PROCESSING 71 – NORTH-HOLLAND PUBLISHING COMPANY (1972)

A SYSTEM OF PROGRAMS FOR THE SYNTHESIS OF SWITCHING CIRCUITS USING THE METHOD OF DISJOINT STARS

Ryszard S. MICHALSKI and Zenon KULPA
Institute of Automatic Control, Polish Academy of Sciences, Warszawa, Poland

This paper describes a system of computer programs developed in the Institute of Automatic Control of the Polish Academy of Sciences for minimization of combinational switching circuits.

All these programs are based on an algorithm A^q, the general principles of which are also outlined. This algorithm was first presented by Michalski [1,3] for quasi-minimal solution of the generally stated cpvering problem. It gives an approximate minimal solution without inspection of irredundant covers and also provides an estimate of the maximum possible "distance" between the obtained cover and the minimal one.

The system consists of five programs; four of them are for minimization of disjunctive normal forms of incompletely specified, one- and multiple-output switching functions with the restriction $(n+m) \leq 31$ (n and m are the numbers of variables and outputs, respectively). The fifth program is intended for minimization of TANT circuits (i.e., three-level with NAND elements) for incompletely specified, one-output switching functions of up to 23 variables.

1. INTRODUCTION

At the Fifth FCIP Symposium we presented an algorithm A^q for the so-called quasi-minimal solution of the generally stated covering problem [1], based on the method of disjoint stars.

At the Sixth FCIP Symposium we presented an application of the same algorithm A^q to the automatic synthesis of minimal forms of incompletely specified multiple-output switching functions together with some computer-generated results [2].

The following programs are presently operational:
(1) A^q – NORMIN – 17a;
(2) A^q – NORMIN – w1;
(3) A^q – NORMIN – w2;
(4) A^q – NORMIN – w3;
(5) A^q – TANTMIN – 4J.

The first two of these programs were described in [2]. They permit the minimization of disjunctive forms of one-output (17a) and multiple-output (w1) switching functions.

The next two are new versions of A^q – NORMIN – w1, while the last one minimizes three-level circuits of NAND or NOR elements, the so-called TANT circuits [4].

2. GENERAL PRINCIPLES OF ALGORITHM A^q

Let

$$X_j = (\dot{x}_1,\ldots,\dot{x}_n)\,; \quad j \in \{0,1,\ldots,2^n-1\}\,; \quad \dot{x}_i \in \{0,1\}\,;$$

denote a sequence of values of the input variables $x_1, \ldots, x_n$ such that

$$j = \sum_{i=1}^{n} \dot{x}_i \cdot 2^{n-i}\,.$$

Let X denote the set of all X_j-sequences. An m-output switching function $f(x_1,\ldots,x_n)$ is then the mapping of the set X into set $\{0,1,*\}^m$, where * denotes an unspecified value 0 or 1 (don't care).

A one-output switching function can be uniquely determined by means of any two of the three sets F^1, F^0, F^* designating the indices j of the sequences X_j at which the function f takes the values 1, 0, *, respectively.

An m-output switching function is equivalent to a set of m one-output functions f^k of the same input variables and can be determined by m above-mentioned pairs of sets – e.g., $(F^{1,k}, F^{0,k})$.

We shall now describe a geometrical model repre-

senting switching functions, the so-called logical diagram [3].

Let $m = 1$. Supposing we are given an arbitrary rectangle, divided into $2^{[n/2]}$ rows and $2^{n-[n/2]}$ columns, where $[n/2]$ = entier $(n/2)$.

In lexical order we assign numbers 0, 1, ..., $2^n - 1$ to cells of the diagram. The cell e having number j will be denoted e^j. Now the numbers of cells correspond to the numbers of X_j sequences. Moreover, we can assign the letters x_i and $\bar{x}_i$ to the specified parts of the diagram (as in the Veitch one) so that every cell e^j will correspond to a sequence X_j (fig. 1).

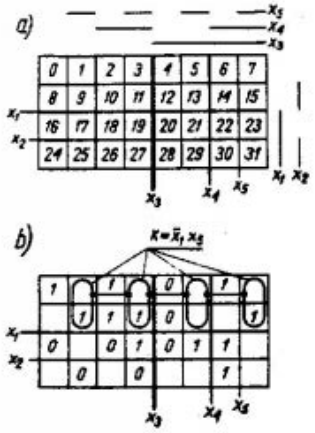

Fig. 1. (a) The logical diagram for $n = 5$, and (b) the image of function f of 5 variables; an unfilled cell denotes unspecified value *.

Furthermore, when the value $f(X_j)$; $j = 0, 1, ..., 2^n - 1$ of function f is assigned to the cell e^j of the diagram, we call it the image $T(f)$ of function f (fig. 1).

In the case where $m > 1$ we divide every cell e^j in the diagram into m smaller subcells e^{jk}, so an m-output function may be represented on the diagram if we assign the values $f^k(X_j)$ to the subcells e^{jk}.

When we have the image $T(f)$, a set of cells (subcells) corresponding to a prime implicant K_i of the function f (when $m > 1$ a multiple-output prime implicant) will be called a complex of cells.

The star $G(j)$ or $G(jk)$ of cell e^j or subcell e^{jk} such that $e^j \in F^1$ or $e^{jk} \in F^{1,k}$ is the set of all complexes covering the cell e^j or subcell e^{jk}. By G^r we denote the family of disjoint stars. We denote the minimal cover of image $T(f)$ by $M(T)$, the quasi-minimal cover (i.e., the cover obtained from algorithm A^q) by $M^q(T)$. $c(K)$ is the number of elements in a set K.

THEOREM 1. The number of elements $c(M(T))$ of the minimal cover of image $T(f)$ satisfies the relation

$$c(M(T)) \geqslant c(G^r). \tag{1}$$

Then an estimate Δ,

$$c(M^q(T)) - c(M(T)) \leqslant \Delta, \tag{2}$$

may be obtained as

$$\Delta = c(M^q(T)) - c(G^r). \tag{3}$$

An estimate δ,

$$z(M^q(T)) - z(M(T)) \leqslant \delta, \tag{4}$$

where $z(M(T))$ is the cost of the cover $M(T)$ – the sum of costs of its elements – may be obtained similarly (fig. 2) [1].

Fig. 2 presents the flow-diagram of algorithm A^q using the above concepts. F^1, F^0, F^p, M^q are variables the values of which are sets. F^p, M^q are auxiliary variables. $O^p(F, j_1)$ is the operation of selecting one number from the current value of F and

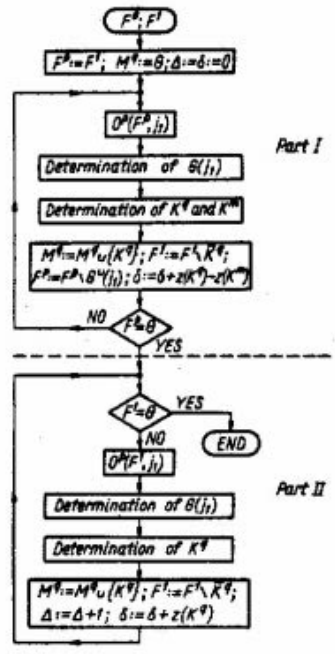

Fig. 2. Algorithm A^q in the case of minimization of switching circuits.

assigning the notation j_1 to it. K^q is a product of $G(j)$ which covers the maximal number of elements of the current value of F^1 (a quasi-extremal). K^m is a minimal cost product of $G(j)$. $z(K)$ is a cost of product K (the number of literals in it). $\bar{K}^q$, $G^u(j)$ are the sets of all elements of F^1 covered by K^q or by all products of $G(j)$ respectively.

The final value of M^q is a set of products – components of a quasi-minimal form $M^q(f)$; (cover $M^q(T)$) of f.

3. PROGRAMS A^q – NORMIN – 17a, A^q – NORMIN – w1

These programs were written in LYAPAS language [5] for the ODRA 1204 computer [6]. They were described in [2].

The first program permits the minimization of only one-output functions (up to 31 variables), the second one handles multiple-output provided $(n+m) \leqslant 31$, where n and m are the numbers of variables and outputs, respectively.

Now we describe the algorithm for generating a star $G(j)$. We shall determine it for a fundamental product $K(j)$ corresponding to cell e^j.

Let $m = 1$. Let $F^0 = \{j_z\}_{z=1}^{y}$ and K_z denote the product corresponding to number j_z.

An implicant of f included in $K(j)$ that is an element of $G(j)$ is a product consisting of literals of $K(j)$ and not included in any product K_z. The implicant is a prime implicant when it is minimal under inclusion. $\tilde{K}$, $\tilde{K}_z$ are sets of literals of products K and K_z, respectively.

Algorithm G^2

1. Determine sets

$$\tilde{D}_z = \tilde{K} \setminus \tilde{K}_z ; \quad z = 1, 2, ..., y .$$

2. Set up a function

$$f_g = \bigwedge_{z=1}^{y} D_z \tag{5}$$

where

$$D_z = \bigvee_{X_i^{\sigma i} \in \tilde{D}_z} X_i^{\sigma i} .$$

3. Find the irredundant disjunctive normal form $D(f_g)$ of function f_g.

The star $G(j)$ is the set of all components of $D(f_g)$.

In cases where $m > 1$ the general principles of generating a star are the same (described in detail in [2]).

Let us assume that the algorithm (fig. 2) has been realized and a cover $M^q(f)$ determined, but values Δ and δ are considered to be too large. If the algorithm is repeated whilst selecting the other quasi-extremals and/or generating stars of other numbers j, then better results may be obtained. There are many different heuristic approaches possible and some of them have been tested in our programs [2]. In that way a certain adaptive process is realized and more iterations may be performed.

4. PROGRAMS A^q – NORMIN – w2 AND w3

These are new versions of program w1, also written in LYAPAS language. The difference between them and w1 is in generating a star.

We notice that determining components of $D(f_g)$ in algorithm G^2 is also the covering problem, so we may apply algorithm A^q to it. The covered elements are disjunctions D_z of function f_g. They are covered by single literals chosen from them in order to algorithm A^q.

In effect these programs work several times faster than w1, but the results are a little worse.

Because the whole star is not generated, determination of $G^u(j)$ (fig. 2) is not possible. Thus we find an algorithm for checking whether or not the stars of given cells, say e^i, e^j, are disjoint ones.

Let $m = 1$. Let K_{jz} denote a fundamental product corresponding to cell e^{jz} so that

$$j_z \in F^0 ; \quad z = 1, 2, ..., c(F^0).$$

Let K' denote the product of literals obtained from K by complementing all its literals (e.g., if $K = x_1\bar{x}_2\bar{x}_3$ then $K' = \bar{x}_1x_2x_3$). $\tilde{K}$ denotes the set of all literals of K.

THEOREM 2. The stars $G(k)$ and $G(i)$ of cells e^k, e^i are disjoint iff there exist at least one z such that $1 \leqslant z \leqslant c(F^0)$ and

$$\tilde{K}'_{jz} \cap \tilde{K}_k \cap \tilde{K}_i = \Theta \tag{6}$$

where Θ denotes an empty set.

For $m > 1$, a similar theorem is valid.

5. PROGRAM A^q – TANTMIN – 4J

This program was written in assembly language "jp zero" for the ODRA 1204 computer for minimization of TANT circuits (*T*hree-Level *A*ND-*N*OT Networks with *T*rue Inputs) [4].

As Gimpel showed [4] the only type of TANT circuit that we need consider will have the form shown in fig. 3.

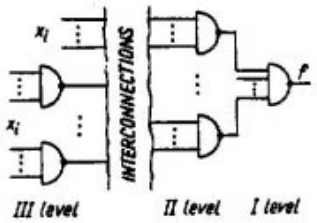

Fig. 3. General structure of TANT circuit.

The second-level elements realize complements of the so-called TANT-expressions (which are the forms of notation of TANT-implicants).

Definition 1. A TANT-implicant of the function f is an implicant of this function which can be written as $P = H\overline{T}_1 \cdots \overline{T}_k$, where H and T_i are frontal terms; H is the head of a TANT-implicant; the T_i are tail factors of TANT-implicants, with the feature that if any tail factor is removed, the resulting expression will not be an implicant of f.

So the minimization of TANT-circuits is equivalent to a problem of covering given functions by TANT-complexes (which correspond to TANT-implicants). In order to apply algorithm A^q we then need only an algorithm for generating a star of TANT-implicants.

Definition 2. A head $H(K)$ of product K is a product of all uncomplemented literals of the product K. A head $H(e)$ of cell e is a head of the fundamental product corresponding to cell e.

Definition 3. A frontal complex of cells, $L^p(H)$, is the complex corresponding to frontal term H. Complex $L^p(H(e))$ is called a frontal complex of cell e.

Analogously, we define backal complexes $L^n(G)$ and $L^n(e) = L^n(G(e))$.

Definition 4. A lower bound of a set of cells Z is the set $Z^d \subseteq Z$ with the property that the set-theoretic sum of frontal complexes of cells of set Z^d includes the set Z and no cell could be removed from Z^d without destroying this property.

$Z(P)$ is the complex of cells corresponding to the TANT-implicant P.

THEOREM 3. TANT-implicant with head H of the function f belonging to the star of cell $e_i \notin F^0$ exists iff

$$e(H) \in Q^i = L^n(e_i) \setminus \bigcup_{\{e_0\}} L^n(e_0) \tag{7}$$

where $e(H)$ is a cell with head H; $\{e_0\} = F^0 \cap L^n(e_i)$.

THEOREM 4. If cell $e(H) \notin F^0$ then we may obtain all TANT-complexes $Z(P_i)$ (corresponding to all TANT-implicants $P_i(H) = H\overline{T}_1 \cdots \overline{T}_k$ with head H) as

$$Z(P_i) = L^p(H) \setminus \bigcup_{s=1}^{t_r} L^p(T_s^i) \tag{8}$$

where $\{L^p(T_s^i)\}_{s=1}^{t_r}$ is an irredundant cover of a set R^d; R^d is a lower bound of set R

$$R = F^0 \cap L^p(H) \tag{9}$$

and for every T_s^i is $\widetilde{T}_s^i \cap \widetilde{H} = \Theta$.

Fig. 4 presents the flow-diagram of that algorithm.

It is worth noting that this algorithm generates all TANT-implicants without generating and inspecting

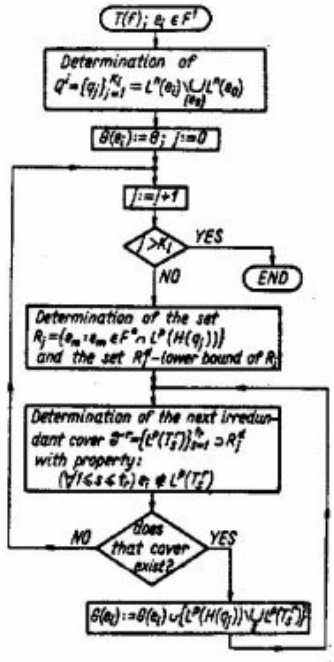

Fig. 4. Generation of a star of TANT-implicants.

the prime implicants of given functions, as was the case in [4] and [7].

In program A^q – TANTMIN – 4J we also use the algorithm A^q for minimization of the third level of circuit, because it is also a covering problem.

The program discussed here permits the minimization of TANT-circuits up to 23 input variables. It also permits the minimization of TANT-structure circuits built with NOR elements.

6. CONCLUSIONS

The system of programs presented here is based on the algorithm A^q for the quasi-minimal solution of the generally stated covering problem, presented in [1]. Experiments with this system show the great usefulness of that algorithm in applications to practice problems in automata theory, leading us to assume that it will also turn out to be useful in other applications as well.

REFERENCES

[1] R.S.Michalski, On the quasi-minimal solution of the general covering problem, Proc. V. Yugoslav Int. Symp. on Inf. Proc. (FCIP 69), Bled 1969, Vol. A3, 125–128.

[2] R.S.Michalski, Automatic synthesis of the quasi-minimal multiple-output switching circuits, Proc. VI Yugoslav Int. Symp. on Inf. Proc. (FCIP 70), Bled 1970, Vol. D1.

[3] R.S.Michalski, Synteza Wyrazen Minimalnych i Rozpoznawanie Symetrii Funkcji Logicznych (Warszawa 1970).

[4] J.F.Gimpel, The minimization of TANT networks, IEEE Trans., EC-16 (1967).

[5] Logicheskii Yazik dlya Predstavleniya Algoritmov Sinteza Releinikh Ustroistv (Moscov 1966).

[6] A.Michalski and T.Wiewiorowski, Odra Ljapas, CC PAS Reports, Vol. 4, (Warszawa 1970).

[7] K.K.Chakrabarti, A.K.Choudhury and M.S.Basu, Complementary function approach to the synthesis of three-level NAND network, IEEE Trans., C-19 (1970).