L'INGÉNIEUR
et le technicien
DE L'ENSEIGNEMENT TECHNIQUE

MATÉRIELS DIDACTIQUES

BIMESTRIEL - PRIX DU N° : 9 F Voir sommaire page 3 30e ANNÉE - N° 193 SEPT.-OCTOBRE 1975

Informatique industriel et automatismes

Sous cette rubrique abondamment fournie nous présentons deux études importantes.
« L'une sur « *l'analyse algorithmique des systèmes automatiques séquentiels* » voir page 45.
L'autre ci-dessous, sur « *la simplification automatique des fonctions booléennes* ».

simplification des fonctions booléennes

P. BOURDET et M. BOSOM
Professeurs à l'E.N.S.E.T.

INTRODUCTION

On sait depuis près de 40 ans que l'analyse et la synthèse des circuits de commandes automatiques peuvent exploiter un modèle d'algèbre de Boole.

La réduction de fonctions booléennes de variables binaires a donc fait l'objet de nombreuses recherches motivées par le souci de créations peu coûteuses. L'économie des composants nécessaires à la réalisation des circuits est obtenue par deux attitudes, l'une visant à réduire le nombre de monômes de la fonction, l'autre le nombre de variables dans chaque monôme.

Dans l'enseignement technique, depuis une quinzaine d'années, le problème est abordé et résolu par l'utilisation d'une méthode « graphique » également répandue dans l'industrie. Cette méthode s'appuie sur deux axiomes et un théorème d'algèbre de Boole et exploite les propriétés du code binaire réfléchi. Son succès auprès des utilisateurs tient à sa simplicité et à son efficacité dans les cas de fonctions ayant un nombre réduit de variables binaires.

Il existe d'autres méthodes, permettant la réduction de fonctions ayant un nombre important de variables, mais elles nécessitent l'assistance d'un ordinateur. Actuellement, elles sont peu enseignées et peu exploitées industriellement. Cette situation devrait changer dans la décennie en cours : l'utilisation scientifique des calculateurs est inéluctable.

Nous nous proposons d'examiner les avantages et les inconvénients d'une méthode algorithmique par comparaison avec la méthode habituelle, pour des circuits de complexité croissante.

CIRCUITS COMPLEMENT DEFINIS

On sait qu'une fonction booléenne de n variables binaires peut s'exprimer et d'une seule façon sous forme canonique disjonctive : somme de monômes d'ordre n (mintermes). Or, la somme de deux monômes d'ordre n est un monôme d'ordre n — 1 si les deux monômes d'ordre n sont adjacents (ceux-ci ne diffèrent que par une seule variable, non complémentée pour l'un, et complémentée pour l'autre). En effet, par exemple dans le cas d'une écriture littérale, cette somme s'écrit :

$$\pi_{i \varepsilon (n-j)} x_i^* x_j + \pi x_i^* \overline{x_j} \qquad \text{avec } x_i^* = x_i \text{ ou } \overline{x_i}$$

soit

$$\pi_{i \varepsilon (n-j)} x_i^* (x_j + \overline{x_j}) = \pi x_i^* \text{ (monôme d'ordre } n-1)$$

On a utilisé l'axiome de complémentation ($x_j + \overline{x_j} = 1$) et l'existence de 1 comme élément neutre du produit booléen.

S'il est aisé de distinguer tous les monômes adjacents à un monôme donné, la réduction est facile.

Cette distinction peut se faire graphiquement sur un tableau qui exploite les symétries existant dans le code binaire réfléchi : c'est la méthode habituelle dite méthode de KARNAUGH.

Cette distinction peut se faire aussi par un groupement des monômes affectés de poids, et classés dans un ordre défini : c'est la méthode algébrique dite méthode de CLUSKEY.

Mais, grâce à l'idempotence de la somme booléenne ($x_i^* + x_i^* = x_i^*$) le même monôme d'ordre n peut s'ajouter avec l'un quelconque des monômes d'ordre n qui lui sont adjacents pour donner un monôme d'ordre (n — 1). Les possibilités de réduction croissent donc très rapidement avec le nombre des monômes adjacents. Cette prolifération de solutions peut dépasser à partir de 6 variables les possibilités visuelles d'un opérateur moyen qui utilise la méthode graphique : l'inventaire des solutions pour le choix optimal n'est plus raisonnablement possible. Certes, on peut toujours donner une expression réduite de la fonction, donc un circuit intéressant sans gros risque d'erreur, mais sans garantie sur la minimisation souhaitée.

Exemple élémentaire : considérons le cas simple d'une fonction booléenne de 4 variables binaires a, b, c, d, dont l'expression canonique disjonctive s'écrit :

$$f = a\bar{b}\bar{c}\bar{d} + \bar{a}b\bar{c}\bar{d} + ab\bar{c}\bar{d} + \bar{a}\bar{b}c\bar{d} + \bar{a}bc\bar{d}$$
$$+ a\bar{b}\bar{c}d + \bar{a}b\bar{c}d + ab\bar{c}d + \bar{a}\bar{b}cd$$
$$+ a\bar{b}cd + \bar{a}bcd + abcd$$

On sait que l'on peut interpréter cette fonction comme la fonction caractéristique d'une fonction booléenne de 4 parties A, B, C, D de l'ensemble de E. Il y a isomorphisme entre ($\mathcal{P}$ (E), ∩, U, C_E) et (F, . , + , ‾) cet isomorphisme étant défini par la fonction qui fait correspondre à chaque partie sa fonction caractéristique.

C'est cette interprétation qui permet la représentation graphique de KARNAUGH (fig. 1) les douze « 1 » correspondant aux douze mintermes de la forme canonique et de plus les mintermes adjacents apparaissent très visiblement. Il y a quatre façons différentes de « couvrir » les « 1 » par des réunions de « carrés » groupant chacun quatre « 1 » soit donc pour la fonction donnée, écrite sous forme disjonctive les quatre expressions réduites :

$$f_1 = \bar{a}c + a\bar{c} + \bar{a}b + ad$$
$$f_2 = \bar{a}c + a\bar{c} + \bar{a}b + cd$$

$$f_3 = \bar{a}\,c + a\,\bar{c} + b\,\bar{c} + a\,d$$
$$f_4 = \bar{a}\,c + a\,\bar{c} + b\,\bar{c} + c\,d$$

Chacune de ces 4 solutions utilise deux monômes essentiels ($\bar{a}$ c et a $\bar{c}$) et deux autres, non essentiels, parmi les quatre restants ($\bar{a}$ b, a d, b $\bar{c}$, c d).

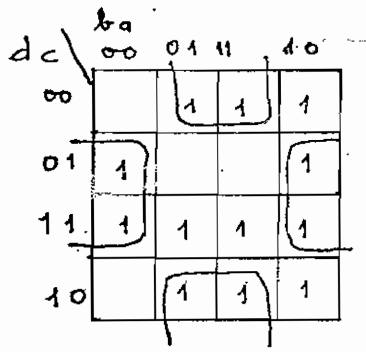


Fig. 1

Pour lire les monômes essentiels (ou « implicants premiers essentiels » I.P.E.), on cherche à « couvrir » un minterme présent dans la fonction donnée — adjacent à des mintermes absents dans la fonction — par un monôme d'ordre aussi faible que possible, c'est-à-dire correspondant à la « couverture » d'un maximum de « 1 ». Ici par exemple, le minterme $\bar{a}\,\bar{b}\,c\,\bar{d}$ conduit à trouver $\bar{a}$ c et le minterme $a\,\bar{b}\,\bar{c}\,\bar{d}$ ou le minterme $a\,b\,\bar{c}\,\bar{d}$ conduisent à a $\bar{c}$. Donc dans cette recherche, l'habilité de l'opérateur joue ici un rôle décisif et l'on rencontre des difficultés à partir de tableaux de KARNAUGH quadruple de celui-ci.

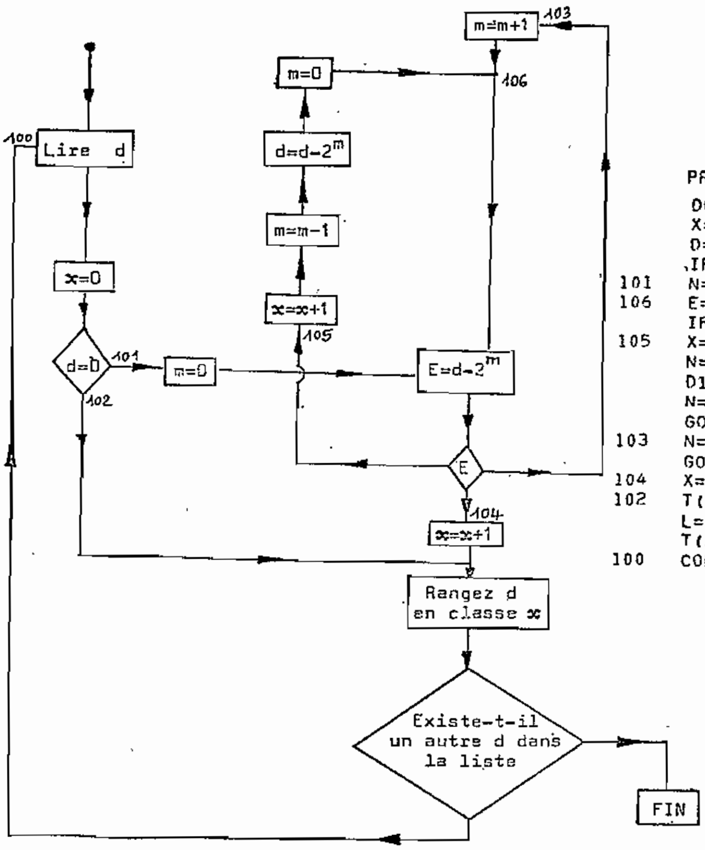
Voyons donc, comment conduire la recherche des implicants premiers (I.P.), distinguer les I.P.E. et faire l'inventaire des solutions d'une façon systématique par la méthode de CLUSKEY.

Nous allons voir la méthode générale et en parallèle l'application au cas particulier de l'exemple élémentaire précédent.

* **Recherche des implicants premiers** : l'écriture des monômes sous forme littérale n'est pas très adaptée aux calculateurs; pour faciliter la lecture, nous les désignons sous forme décimale d, chaque variable étant affectée d'un « poids » correspondant à une puissance de 2. Un minterme m_i d'ordre n d'une fonction de n variables binaires est désigné par d; d $\in \mathbb{N}$ avec d < 2 n.

Dans l'exemple, en adoptant des poids croissants dans l'ordre de l'alphabet, les mintermes seraient désignés respectivement par 1, 2, 3, 4, 6, 9, 10, 11, 12, 13, 14, 15.

— pour préparer l'examen exhaustif de toutes les adjacences, on groupe les mintermes m en classes C_x d'ordre croissant selon les algorithmes fig. 2 et fig. 3.




```
PROGRAMME ;
      DO 100 I=1,NB
      X=0
      D=DON(I)
      IF(D)101,102,101
101   N=0
106   E=D-2**N
      IF(E)103,104,105
105   X=X+1
      N=N-1
      D1=D-2**N
      N=0
      GO TO 106
103   N=N+1
      GO TO 106
104   X=X+1
102   T(X,100)=T(X,100)+1
      L=T(X,100)
      T(X,L)=DON(I)
100   CONTINUE
```

Fig. 2

Fig. 2 : algorithme programmable de la démarche manuelle.

a) si le nombre d est égal à zéro le minterme est rangé en classe x = O (C_0)

b) si le nombre d est différent de zéro, on cherche son excédent E minimal sur la puissance entière m de 2. On compte alors 1 pour la valeur provisoire de x

si E = O on prend x = 1 et le minterme d est rangé en classe x = 1 (C_1)

si E $\neq$ O on effectue le paragraphe c

c) on cherche le nouvel excédent E' minimal de E (nouveau d) sur une nouvelle puissance entière de 2. On augmente alors x d'une unité.

si E' = O on range le minterme en classe x (C_x)

si E' $\neq$ O on recommence le paragraphe c

on note que le rang x de la classe est égal au nombre d'excédents cherchés.

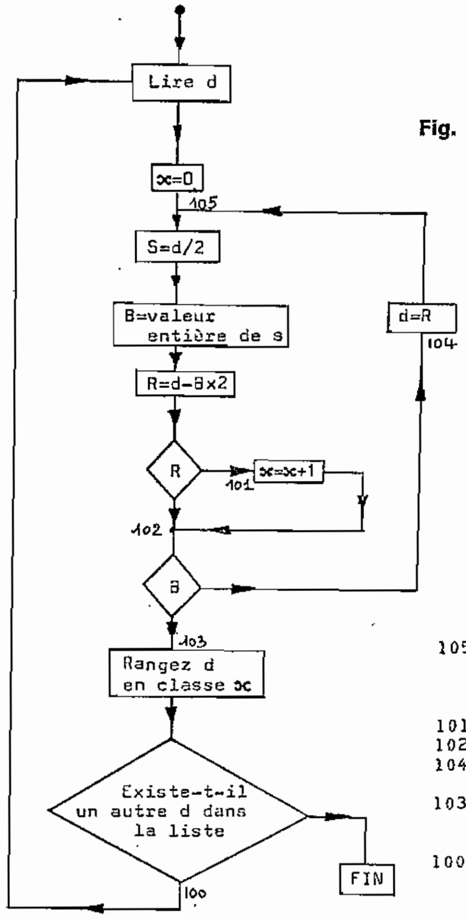


Fig. 3

PROGRAMME :

```
      DO 100 I=1,NR
      X=0
      D=DON(I)
105   S=D/2.
      B=AINT(S)
      R=D-(B*2)
      IF(R)102,102,101
101   X=X+1
102   IF(B)104,103,104
104   D=R
      GO TO 105
103   T(X,100)=T(X,100)+1
      L=T(X,100)
      T(X,L)=DON(I)
100   CONTINUE
```

To

C_0	néant
C_1	1 2 4.
C_2	3 6 9 10 12
C_3	11 13 14
C_4	15

Fig. 4

Fig. 3 : algorithme programmable utilisé effectivement.

Il exploite le fait que le rang x est aussi le nombre de 1 de l'écriture binaire du minterme.

a) on compte le nombre de reste égaux à 1 lors des divisions utilisées dans le passage de l'écriture décimale à l'écriture binaire.

b) on range d dans le tableau en classe x (C_x)

Fig. 4 : correspond à l'exemple traité :

— pour aboutir aux I.P. à partir du classement précédent, on cherche les monômes d'ordre décroissant.

L'algorithme général est présenté fig. 5.

Tout monôme d'ordre n est représenté par la suite des mintermes qu'il couvre ordonnés selon leur désignation décimale croissante D_i (di_1, di_2, di_3... di_n)

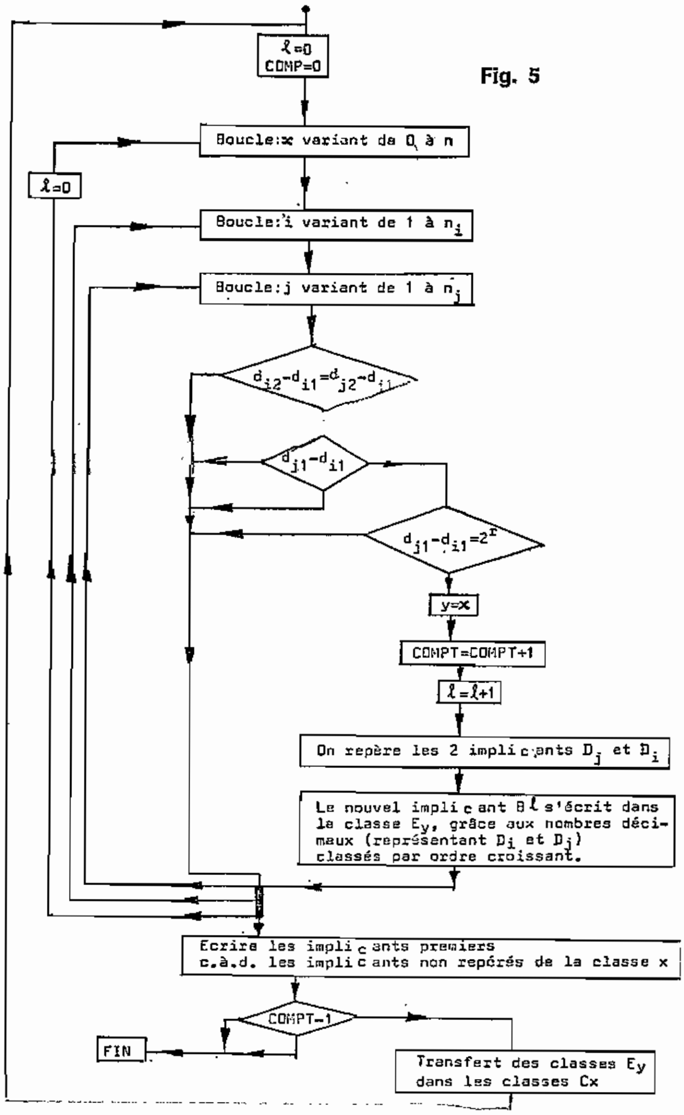


Fig. 5

d_{i1} est le premier minterme couvert par l'implicant $D_i \in C_x$

d_{j1} est le premier minterme couvert par l'implicant $d_j \in C_{x+1}$

Dans l'exemple traité fig. 6, il y a sept I.P. d'ordre 2, surnommés A, B, C, D, E, F, G. On utilise l'algorithme général mais, dans la procédure manuelle, on établit un classement partiel dans chaque classe.

*** Recherche des implicants premiers essentiels I.P.E. et des différentes solutions.**

— pour distinguer simplement les I.P.E. parmi les I.P. trouvés précédemment, il suffit de dresser un tableau à double entrée, permettant de voir la « couverture » des mintermes — occupant les colonnes — par les I.P. en lignes (cf. fig. 7 pour l'exemple). Toute colonne n'ayant qu'une croix indique que l'I.P. correspondant est le seul à inclure le minterme en tête de colonne : cet I.P. est essentiel. Dans l'exemple, il y en a deux, B et D : ils sont encadrés.

— pour avoir une solution, il suffit de choisir une somme d'I.P. qui couvre tous les mintermes.

La présence des I.P.E. permet de satisfaire cette couverture en partie seulement dans le cas général (les mintermes inclus sont différenciés). Pour les autres, s'il y a plusieurs éventualités, nous cherchons la ou les combinaisons contenant le plus petit nombre de lettres. Pour le cas examiné, nous pouvons dire : la couverture du minterme 2 exige la présence des I.P. A ou C, celle de 10 exige A ou B ou E etc., l'ensemble de ces conditions peut s'exprimer par une condition unique :

$$F = (A + C)(A + C + E)(F + G)(E + F + G) = 1$$

(F est appelée « fonction de choix ».)

Les combinaisons utilisant le moins d'I.P. sont : AF = 1, AG = 1, CF = 1, CG = 1. Chaque I.P. étant un monôme d'ordre 2, ces 4 combinaisons sont équivalentes.

En définitive, les 4 solutions du cas particulier correspondent à BDAF = 1, BDAG = 1, BDCF = 1 et BDCG = 1.

Se sont bien celles que nous avions trouvées par la méthode graphique.

L'algorithme général d'obtention des différentes solutions est présenté fig. 14.

Remarques : tout ce qui précède a été entrepris à partir d'une fonction booléenne de n variables binaires exprimée initialement sous forme canonique disjonctive : le résultat est une forme réduite disjonctive. On ne peut affirmer que c'est la forme minimale.

En effet, toute fonction booléenne peut s'exprimer sous forme canonique conjonctive, donc il existe une ou des formes réduites conjonctives qui pourraient exiger moins de lettres.

Pour en décider, il suffit — conformément à la méthode générale de passage d'une forme canonique à l'autre — de refaire la réduction sur la fonction complémentaire sous forme disjonctive et à prendre le ou les compléments du ou des solutions réduites.

Dans l'exemple traité, on trouve $\overline{y} = \overline{a}\,\overline{b}\,\overline{c} + a\,c\,\overline{d}$ soit $y = (a + b + c)(\overline{a} + \overline{c} + d)$ seule solution réduite conjonctive qui est en fait la solution minimale.

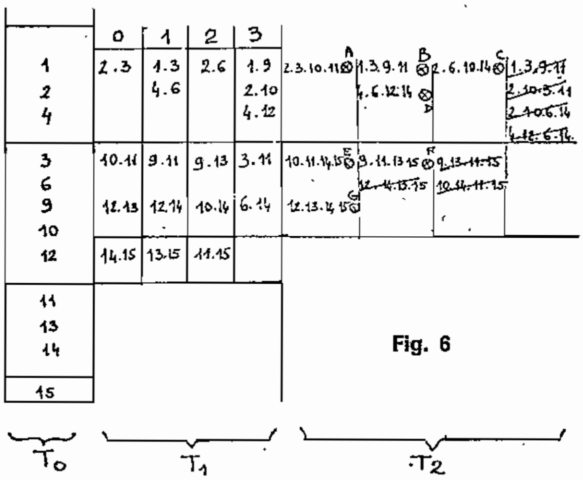


Fig. 6

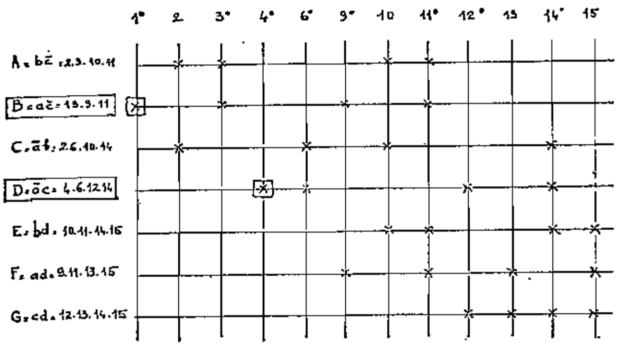


Fig. 7

— Tout ce que l'on a vu s'applique à une fonction dont on connaît la valeur (0 ou 1) pour chacun des mintermes. C'est le cas, en particulier, lorsqu'on fait l'analyse d'un circuit existant, mais dans un problème de synthèse, il est courant que, pour plusieurs mintermes d'ordre n, la valeur de la fonction ne s'impose pas, elle peut être indifféremment 1 ou 0.

Examinons ce cas.

CIRCUITS INCOMPLETEMENT DEFINIS

Dans la réalité, ce sont les plus fréquents.

On connaît en fait deux bornes pour la fonction à réaliser : la borne minimale, correspondant à l'affectation de la valeur 0 pour tous les mintermes indifférents, et la borne maximale correspondant à l'affectation de la valeur 1 pour ces mêmes mintermes. La fonction à réduire se situe entre les deux.

Comment procéder à la réduction ?

* **Méthode de KARNAUGH.** Raisonnons sur l'exemple présenté par le tableau fig. 8.

Fig. 8 a : on cherche une forme réduite disjonctive : on englobe le maximum de « 1 » adjacents en s'aidant des valeurs indifférentes (notées par un tiret) auxquelles on affecte si nécessaire la valeur 1. Ceci a été fait pour 10 d'entre eux, les autres ont la valeur 0 pour la forme réduite choisie mais il y en a d'autres.

Fig. 8 b : on cherche une forme réduite conjonctive : on englobe le maximum de « 0 » adjacents en s'aidant des valeurs indifférentes. Dans le groupement effectué 8 valeurs « 0 » sont attribuées.

Cet exemple montre déjà la difficulté pour faire l'inventaire des solutions par cette méthode pour un nombre de variables peu importantes et garantir une solution minimale.

y — ed \ cba	000	001	011	010	110	111	101	100
00	1	0	0	1	–	1	–	0
01	–	1	1	1	1	–	–	–
11	–	0	1	0	–	1	1	–
10	–	–	–	1	1	–	–	1

Fig. 8 a

y — ed \ cba	000	001	011	010	110	111	101	100
00	1	0	0	1	–	1	–	0
01	–	1	1	1	1	–	–	–
11	–	0	1	0	–	1	1	–
10	–	–	–	1	1	–	–	1

Fig. 8 b

* **Méthode de CLUSKEY.** On peut envisager la réduction de la façon suivante :

— on considère la borne maximale pour la recherche des I.P.;

— on ne retient que la borne minimale pour la recherche des I.P.E. et des différentes solutions.

Evidemment, ceci est à faire pour chacune des deux formes (disjonctive et conjonctive).

A titre d'exemple, on trouvera fig. 9 et 10 l'illustration de la méthode pour l'exemple précédent dans le cas des recherches de solutions réduites disjonctives.

On trouve 8 solutions réduites disjonctives à 12 lettres.

L'étude de la forme conjonctive permettrait de trouver une neuvième solution à 12 lettres, soit au total 9 solutions minimales.

L'examen des tableaux fig. 9 et 10 fait apparaître un inconvénient de cette méthode.

De nombreux calculs ne servent à rien : par exemple tous les groupements qui ont été fait pour trouver l'I.P. (F) sont inutiles. Celui-ci intéresse des regroupements de valeurs indifférentes de la fonction.

Or, dans la pratique les valeurs indifférentes sont les plus nombreuses : un problème moyen comprenant une douzaine de variables avec par exemple une trentaine de valeurs imposées, a donc plus de 4 000 valeurs indifférentes qui alourdissent le calcul ! Manuellement, les « indifférents » sont sources d'erreurs et, mécaniquement, ils sont encombrants dans les registres mémoires (pour 15 variables plus de 32 000 valeurs).

Ce sont les raisons majeures qui font que pratiquement, les programmes sur ordinateurs de faible taille ne peuvent

0		0.2		0.8	0.16		0.2.8.10⊗A 0.2.16.18⊗B		0.8.16.24⊗C	8.9.10.11.12.13.14.15⊗J 16.17.18.19.20.21.22.23⊗K	5.7.13.15.21.23.29.31
2 8 16	8-9 16-17	8.10 16-18	2.6 8.12 16.20	2-10 16.24	2.18 8.24	8.9.10.11 8.9.12.13 16.17.18.19 16.17.20.21	8.10.12.14 16.18.20.22	2.6.10.14⊗D 2.6.18.22⊗E 8.12.24.28⊗F 16.20.24.28⊗G		6.7.14.15.22.23.30.31⊗M 12.13.14.15.28.29.30.31⊗N 20.21.22.23.28.29.30.31⊗O	
5 6 9 10 12 17 18 20 24	6.7 10-11 12-13 16-19 20.21	5.7 9-11 12.14 17.19 20.22	9.13 10.14 17.21 18.22 24.28	5.13 6.14 20.28	5.21 6.22 12.28	6.7.14.15 6.7.22.23 10.11.14.15 12.13.28.29 12.13.14.15 18.19.22.23 20.21.22.23 20.21.28.29	5.7.13.15 5.7.21.23 9.11.13.15 12.14.28.30 17.19.21.23 20.22.28.30		5.13.21.29 6.14.22.30		
7 11 13 14 19 21 22 28	14-15 22.23 28.29	13-15 21-23 28.30	11.15 19.23	7.15 19.27 21.29 22.30	7.23 11.27 13.29 14.30	14.15.30.31 22.23.30.31 28.29.30.31	13.15.29.31 21.23.29.31	11.15.27.31⊗H 19.23.27.31⊗I	7.15.23.31		
15 23 27 29 30	30.31	29.31	27.31	23.31	15.31						

Fig. 9

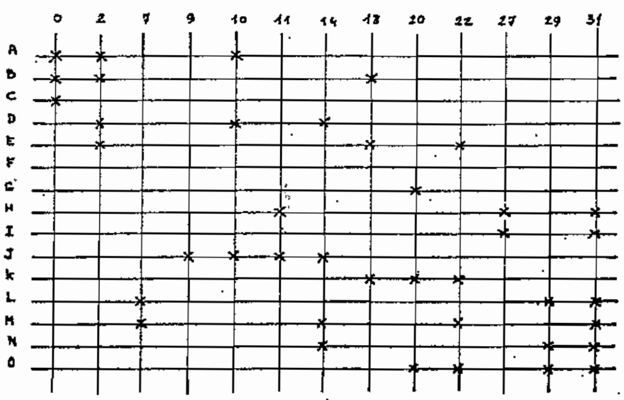


Fig. 10

pas utiliser exactement cet algorithme pour la recherche des I.P. : nous en avons fait l'expérience il y a quelque temps, sur l'IBM 1130.

*** Recherche des I.P. utiles par une méthode mieux adaptée** (explicitement elle ignore les indifférents).

Considérons la borne maximale f_M de la fonction f des n variables binaires x_i.

On sait qu'on peut exprimer f_M et d'une seule façon sous forme canonique conjonctive (produit de somme des n variables x_i^* : maxtermes d'ordre n).

soit $f_M = \pi_{i \in I} \alpha_i M_i$ avec $I = \{ 0,1 \ldots 2^n - 1 \}$; $M_i = \Sigma_{i \in n} x_i^*$

$\alpha_i = 0$ ou 1 selon que M_i est absent ou présent, et $x_i^* = x_i$ ou $\bar{x}_i$

On a, bien sûr : $f \subset f_M$

Mais dans chaque maxterme présent, il y a des x_i^* qui sont inutiles pour f : ce sont ceux qui ne couvrent aucun minterme présent. En faisant une partition dans les M_i des x_i^* utiles et des x_i^* inutiles, on peut écrire :

$f_M = \pi_{i \in I} \alpha_i \, (M'_i + M''_i)$

et choisir

$f = \pi_{i \in I} \alpha_i \, M'_i$

On peut retrouver à titre d'illustration les I.P. utiles de l'exemple précédent (fig. 11).

1re colonne : écriture des « zéros » et des « uns » de f sous forme binaire naturelle.

2e colonne : écriture des maxtermes M_i de f à partir des zéros.

3e à 6e colonne : recherche pas à pas des différents produits des M_i.

La liste des « uns » permet l'élimination aisée de chaque résultat inutile en cours de calcul.

Dans chaque étape, on transcrit en priorité les termes de π_i qui vont apparaître dans π_{i+1}, pour diminuer le nombre de comparaisons à effectuer pour former π_{i+1} (loi d'absorption).

Dans la dernière colonne, on a identifié les résultats avec ceux du tableau de la fig. 9.

Il faut remarquer la supériorité de ce mode opératoire sur le précédent, même manuellement, les sources d'erreurs par omission sont beaucoup plus réduites et on peut entreprendre des problèmes avec un nombre de variables plus important.

L'algorithme « mécanisé » suit une procédure semblable (cf. fig. 12-13-14).

"zéros" de f
0 0 0 0 1
0 0 0 1 1
0 0 1 0 0
1 1 0 0 1
1 1 0 1 0

"uns" de f
0 0 0 0 0
0 0 0 1 0
0 0 1 1 1
0 1 0 0 1
0 1 0 1 0
0 1 0 1 1
0 1 1 1 0
1 0 0 1 0
1 0 1 0 0
1 0 1 1 0
1 1 0 1 1
1 1 1 0 1
1 1 1 1 1

Maxtermes de f	
	1 - - - -
	- 1 - - -
M_0	- - 1 - -
	- - - 1 -
	- - - - 0
	1 - - - -
	- 1 - - -
M_1	- - 1 - -
	- - - 0 -
	- - - - 0
	1 - - - -
	- 1 - - -
M_2	- - 0 - -
	- - - 1 -
	- - - - 1
	0 - - - -
	- 0 - - -
M_3	- - 1 - -
	- - - 1 -
	- - - - 0
	0 - - - -
	- 0 - - -
M_4	- - 1 - -
	- - - 0 -
	- - - - 1

$\Pi_1 = M_0 . M_1$
1 - - - -
- 1 - - -
- - 1 - -
- - - - 0

$\Pi_2 = \Pi_1 . M_2$
1 - - - -
- 1 - - -
- - 0 - 0
- - 1 1 -
- - - 1 0
- - 1 - 1

$\Pi_3 = \Pi_2 . M_3$
- - 1 1 -
- - 1 - 1
- - 0 - 0
- - - 1 0
0 1 - - -
1 0 - - -
1 - 1 - -
- 1 1 - -
1 - - 1 -
- 1 - 1 -
1 - - - 0
- 1 - - 0

$\Pi_4 = \Pi_3 . M_4$	
0 1 - - -	J
1 0 - - -	K
- - 1 1 -	M
- - 1 - 1	L
1 - 1 - -	O
- 1 1 - -	N
0 - 0 - 0	A
0 - - 1 0	D
- 0 0 - 0	B
- 0 - 1 0	E
- - 0 0 0	C
1 - - 0 0	G
1 - - 1 1	I
- 1 - 1 1	H

Fig. 11

Algorithme : simplification d'une fonction booléenne (critère d'optimalisation, obtenir le minimum de lettres).

A – Introduction des données (sous forme octale).

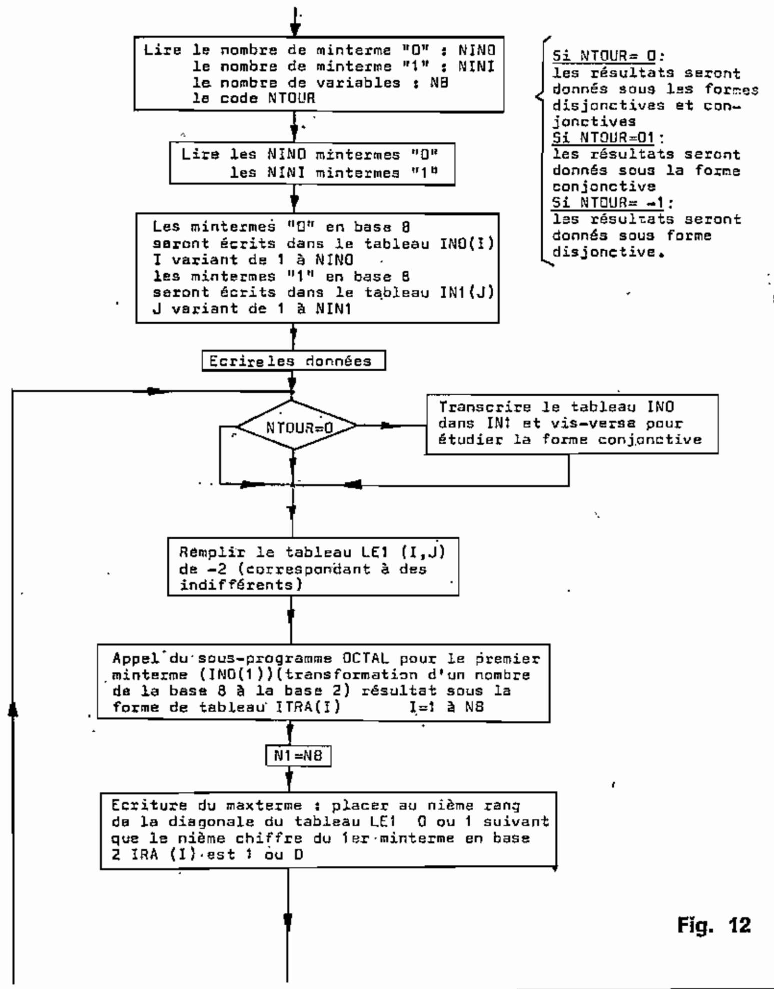


Fig. 12

Voir fig. 12 bis et 12 ter page suivante

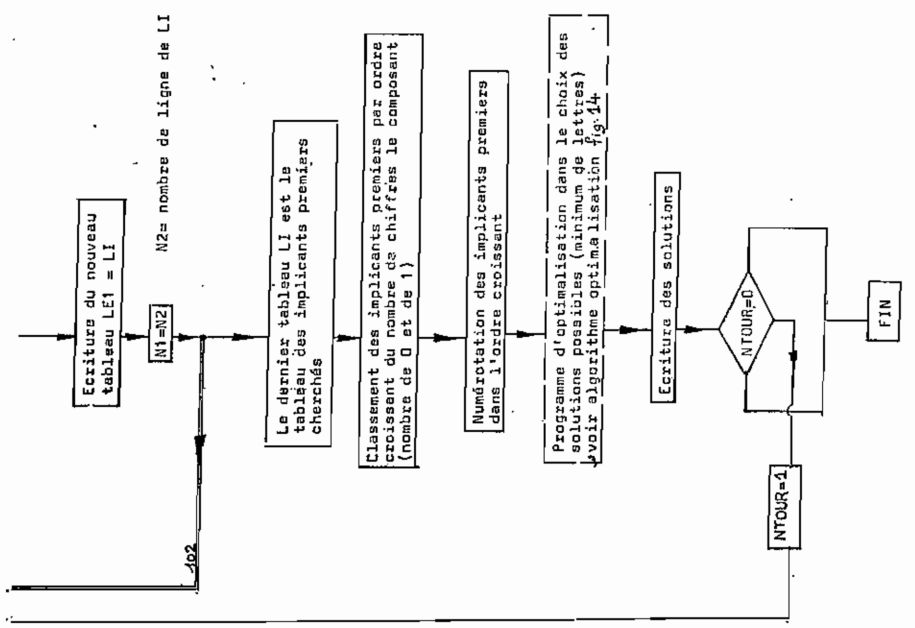


Fig. 12 ter

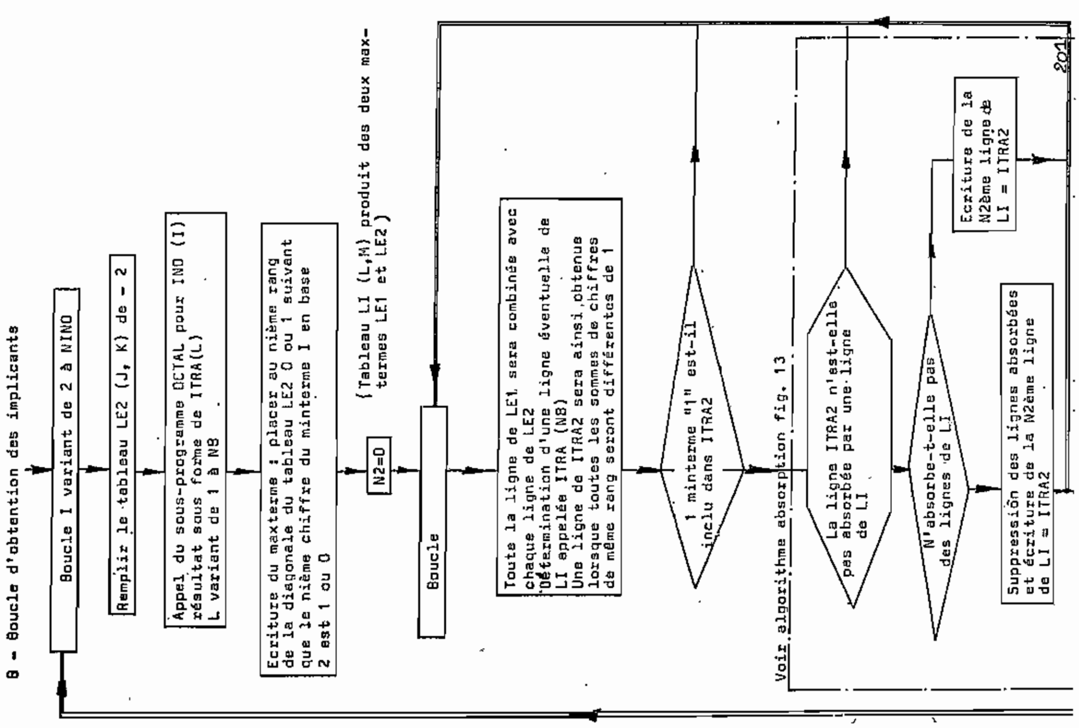


Fig. 12 bis

CIRCUITS MULTIPLES (multipôles)

— Dans la réalité, non seulement les fonctions sont incomplètement définies mais les circuits incluent plusieurs fonctions. Par exemple dans tout transcodage élémentaire, il y a autant de fonctions que de « bit » de sortie. Une analyse superficielle consiste à reconduire autant de réduction qu'il y a de fonctions. Hélas il est facile de voir que la somme de solutions minimales obtenues séparément n'est pas minimale.

En effet, un I.P. comportant 3 lettres par exemple mais qui serait commun à deux ou plusieurs fonctions peut se révéler plus économique que plusieurs I.P. de 2 lettres qu'on lui a successivement préféré, pourvu qu'il soit possible de l'utiliser simultanément dans plusieurs fonctions.

— Voyons cela sur un exemple volontairement très simple.

Soit à réaliser un circuit capable de satisfaire les trois fonctions simultanées données par la somme des mintermes présents S_1 ( ) et la somme des mintermes indifférents $S_{(-)}$ ( ) :

$f_1 = S_1$ (4, 11, 14) + $S_{(-)}$ (3, 6, 12, 15)
$f_2 = S_1$ (3, 6, 8, 9, 11) + $S_{(-)}$ (2, 7, 12, 14)
$f_3 = S_1$ (0, 1, 2, 9, 10, 14) + $S_{(-)}$ (5, 7, 8, 12, 15)

supposons que l'on s'intéresse à une forme réduite conjonctive.

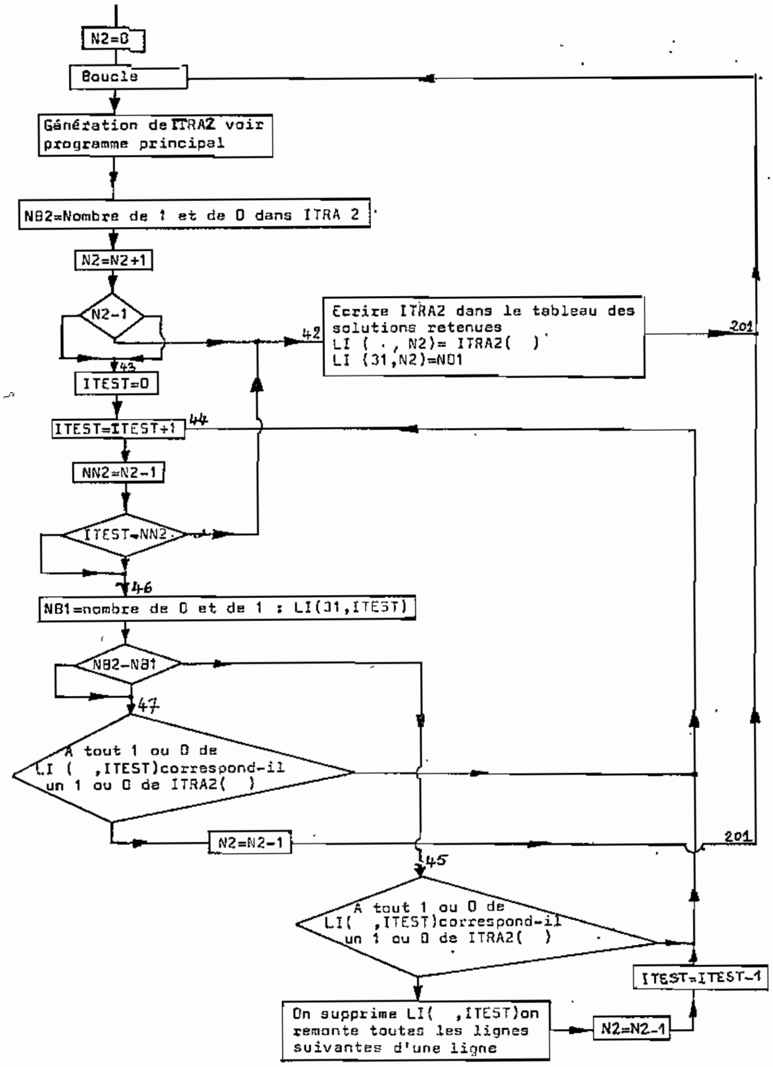
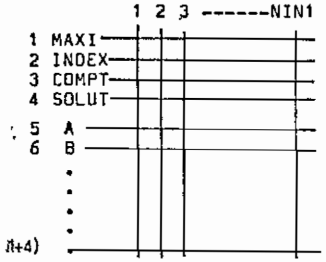


Fig. 13

INTRODUCTION A L'ALGORITHME DE LA FIG. 14

Etablissement du tableau permettant la recherche des différentes solutions.

Le tableau est constitué par NIN1 colonnes représentantes des NIN1 mintermes "1".



* les lignes 5 à (N1+4) représentent les implicants premiers classés par ordre croissant du nombre de lettres.

* la présence d'un 1, aux intersections des lignes 5 à (N1+4) et des colonnes, indique que le minterme "1" de la colonne est inclu dans l'implicant premier de la ligne.

* la ligne 1 (MAXI) représente le total des 1 compris dans chaque colonne.

* la ligne 2 (INDEX) sera constituée de 0 et d'un index écrit 1 repérant une colonne particulière.

* la ligne 3 (COMPT) représente le nombre de passage de l'index dans chaque colonne.

* la ligne 4 (SOLUT) sera constituée de 1 et de 0, chaque 1 indiquera que le minterme "1" de la colonne est présent dans la solution en cours d'élaboration (ainsi lorsque la ligne 4 sera constituée uniquement de 1, une solution sera trouvée).

LEGENDE :

SOMAX : nombre de lettres de chacune des solutions déjà retenues.

KP : nombre de solutions retenues.

NC : numéro de la colonne.

K1 : position de l'implicant premier retenu, dans le tableau à une dimension SOL (K1).

SOM : nombre de lettres contenues dans la solution en cours d'élaboration.

Fig. 14 : Algorithme Obtention des différentes solutions.

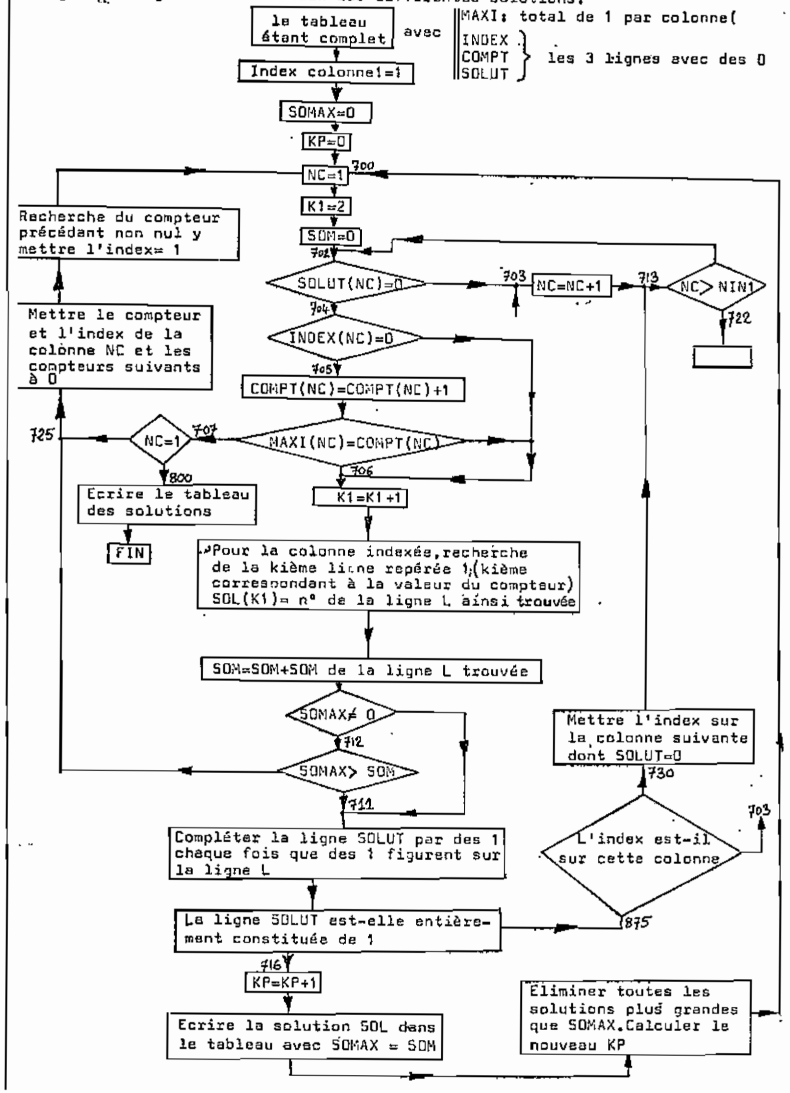


Fig. 14

• Méthode de KARNAUGH

L'opérateur le plus averti se sent vite débordé devant ce genre de problème dès qu'il y a 5 ou 6 variables et autant de fonctions. Ici, avec un peu de « métier » on peut voir la solution, qui fort heureusement est unique (fig. 15).

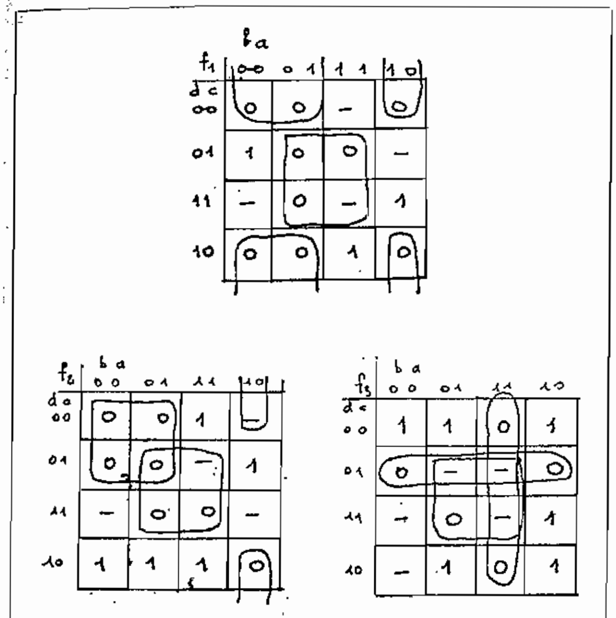


Fig. 15

• Méthode de CLUSKEY

La recherche des I.P. est conduite simultanément mais pourrait tout aussi bien être faite successivement. Cela fait, on obtient le tableau de recouvrement (fig. 16).

Le choix des I.P.E. pour la 2[e] et la 3[e] fonction permet la couverture des mintermes pointés.

Pour ces deux fonctions, le choix est ensuite évident. Dans la 2[e] pour « 10 » : B est prioritaire devant 0 puisqu'il intéresse 2 fonctions et dans la 3[e] pour « 13 » P est prioritaire devant M puisqu'il est commun aux 3 fonctions.

Il reste dans la 1[re] à couvrir 0, 1, 8, 9, soit la fonction de choix :

$$(A + E + G + H)(E + G + I + J)(C + E + H + N)(E + J + N) = 1$$

dont la solution la plus simple est E = 1.

En définitive on a donc pour :

$\overline{f_1}$: P B E = 1 qui donne $\overline{f_1} = a\,c + \bar{a}\,\bar{b}\,\bar{c} + \bar{b}\,\bar{c}$ soit :

$f_1 = (\bar{a} + \bar{c})(a + \bar{b} + c)(b + c)$

$\overline{f_2}$: F P B = 1 qui donne $\overline{f_2} = a\,c + \bar{b}\,\bar{d} + \bar{a}\,b\,\bar{c}$ soit :

$f_2 = (\bar{a} + \bar{c})(b + d)(a + \bar{b} + c)$

$\overline{f_2}$: L O P = 1 qui donne $\overline{f_3} = c\,\bar{d} + a\,b + a\,c$ soit :

$f_3 = (\bar{c} + d)(\bar{a} + \bar{b})(\bar{a} + \bar{c})$

Un peu longuement mais sûrement, on arrive au résultat lu sur les tableaux de KARNAUGH.

L'économie pour un circuit à diode sera au plus de 6 diodes (22 au lieu de 28).

Pour des fonctions plus complexes, il est toujours possible d'optimiser une fonction de choix globale puis de répartir dans chacune des fonctions les I.P. sélectionnés.

Le programme correspondant aux circuits multipôles suit sensiblement les mêmes algorithmes que précédemment (il est réalisé et disponible en FORTRAN V. Il permet la résolution d'un problème de N fonctions (de P variables) ayant chacune un nombre Q_i maximum de minterme « 0 » ou « 1 ». La capacité K de l'ordinateur est obtenue par la formule :

$$1\,000\,K = 310\,Q + 1\,300\,P + P^2 + 5\,000 + 2N \quad (\text{avec } Q = \Sigma Q_i)$$

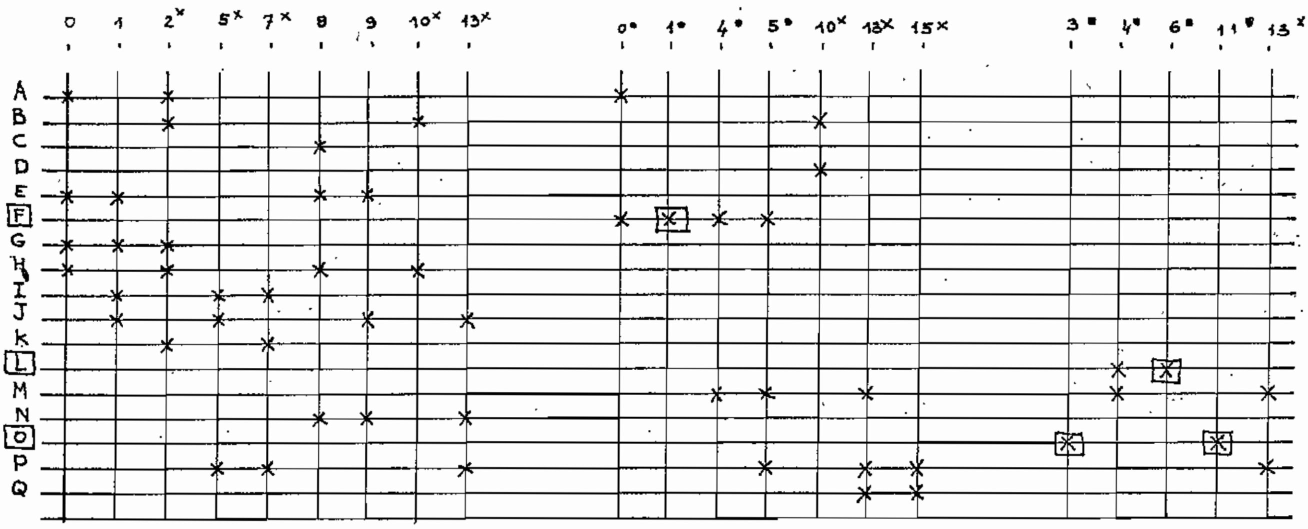


Fig. 16

CONCLUSION

Pratiquement, le bilan de la confrontation des deux méthodes graphique et algorithmique de simplification des circuits, est fait devant les moyens dont on dispose. Chaque entreprise, chaque établissement scolaire ne bénéficie pas d'un terminal d'ordinateur.

Il n'est pas douteux que la méthode habituelle doit être enseignée pour munir tout opérateur isolé d'un outil de travail mais il n'est pas moins douteux que c'est un outil artisanal. L'existence d'outils « industriels » ne doit pas être ignorée.

Actuellement, une question fondamentale se pose : quel est l'avenir des méthodes de simplification des circuits alors que nous quittons l'ère des composants discrets pour rentrer dans celle des composants intégrés ? Des modules très complexes réalisant une multitude de fonctions logiques sont sur le marché.

Très certainement la réduction des fonctions va céder la première place à la décomposition des fonctions. Aujourd'hui, dans l'enseignement, nous en sommes aux balbutiements de ces méthodes, réclamées par l'exploitation rentable du matériel : le prix du câblage, de l'entretien et de l'extension d'un circuit dépasse celui des composants. Mais une chose est évidente, c'est que ces méthodes ont aussi besoin d'un ordinateur. Qu'on le regrette ou non, les méthodes graphiques devront céder la place aux méthodes algorithmiques.

Bibliographie : systèmes logiques PERRIN DENOUETTE DACLIN (Dunod) mathématique et automatique BOSOM CHATY (Hachette) cours d'automatique ENSET.

PROGRAMME DE L'ALGORITHME DES FIGURES 12-13-14

```
PROGRAMME DE L'ALGORITHME DES FIGURES 12 - 13 - 14.

#RUN,8 REDUCT,CHNNST,CHN
#FOR,ISO
      PARAMETER LIGNE=600,IKV2=11,IKM=92
      PARAMETER IKSOL=100
      DIMENSION LE21(IKM,4),LE11(IKM,IKSOL)
      DIMENSION INO(IKM),IN1(IKM),FNO(IKM),ITRA(IKM),ITRA2(IKM)
C       IKV2=(NB DE VARIABLES+2)
C       IKM =SUP(NB DE MINTERMES 1 ) (NB DE MINTERMES 0 )
      DIMENSION LE1(IKV2,LIGNE),LE2(IKV2,IKV2),LI(IKV2,LIGNE),TAB(80)
      DIMENSION LUTION(IKM,IKSOL)
C                 LECTURE DES DONNEES ***********************************
      READ 6,TAR
6     FORMAT(80A1)
      PRINT 7,TAR
7     FORMAT(1H1,20X,80A1,///)
      PRINT 990
990   FORMAT(1H ,' AVEZ VOUS VERIFIE QUE LE PARAMETER IKV2 EST LE MEME DA
     1NS LE PROGRAMME PRINCIPAL ET LE SOUS PROG. OCTAL',//)
      READ 1,NINO,NIN1,NB,NTOUR
1     FORMAT(4I2)
      IKV=IKV2-2
      IF(IKV-NB)881,995,995
995   IF(IKM-NINO)886,880,880
880   IF(IKM-NIN1)886,882,882
881   NTU=NB+2
      PRINT 883,NTU
883   FORMAT(1H ,'PARAMETER IKV2 DES PROGRAMMES TROP FAIBLE  METTRE IKV2
     1=',I2)
      GO TO 999
882   READ 2,(FNO(I),I=1,NINO)
2     FORMAT(5F16.1)
      DO 100 I=1,NINO
100   INO(I)=IFIX(FNO(I))
      READ 2,(FNO(I),I=1,NIN1)
      DO 101 I=1,NIN1
101   IN1(I)=IFIX(FNO(I))
      PRINT 11,NB
11    FORMAT(1H ,'NOMBRE DE VARIABLES ',I2,/)
      PRINT 8,NINO
8     FORMAT(1H ,'NB DE MINTERMES A L ETAT 0  ',I2,/)
      PRINT 10,(INO(I),I=1,NINO)
      PRINT 9,NIN1
9     FORMAT(1H ,'NB DE MINTERMES A L ETAT 1  ',I2,/)
      PRINT 10,(IN1(I),I=1,NIN1)
10    FORMAT(1H ,5I15)
915   IF(NTOUR)890,890,891
891   DO 892 I=1,NINO
892   FNO(I)=INO(I)
      DO 893 I=1,NIN1
893   INO(I)=IN1(I)
      DO 894 I=1,NINO
894   IN1(I)=FNO(I)
      NAM=NINO
      NINO=NIN1
      NIN1=NAM
890   CONTINUE
C                 ENTREE DU TABLEAU 1 POUR LA PREMIERE DONNEES'*************
996   DO 104 J=1,NB
      DO 104 M=1,LIGNE
104   LE1(J,M)=-2
```

```
      IDES=1
      CALL OCTAL(INO(IDES),ITRA,NB)
CXXX  PRINT 1000,INO(IDES),(ITRA(I),I=1,NB)
1000  FORMAT( )
      N1=NB
      DO 103 J=1,NB
      IF(ITRA(J)-1)110,109,110
109   LE1(J,J)=0
      GO TO 103
110   LE1(J,J)=1
103   CONTINUE
CX    PRINT 1001
CX    DO 2000 IPI=1,NB
CX 2000  PRINT1000(LE1(IPR,IPI),IPR=1,NB)
C                 BOUCLE DE SIMPLIFICATION************************************
      DO 102 I=2,NINO
C                 ENTREE DU TABLAAU 2*****************************************
      DO 105 J=1,NB
      DO 105 M=1,N1
105   LE2(J,M)=-2
      CALL OCTAL(INO(I),ITRA,NB)
CX    PRINT 1000,INO(I),(ITRA(II),II=1,NB)
      DO 106 L=1,NB
      IF(ITRA(L)-1)108,107,108
107   LE2(L,L)=0
      GO TO 106
108   LE2(L,L)=1
106   CONTINUE
CX    PRINT 1001
1001  FORMAT(1H ,/,' ******************************************',/)
CX    DO 2001 IPI=1,NB
CX 2001  PRINT1000(LE2(IPR,IPI),IPR=1,NB)
C                 TABLEAU LI INTERSECTION DES TABLEAUX 1 ET 2 ***************
      N31=NB+1
      N2=0
      DO 200 L1=1,N1
      DO 201 L2=1,NB
C                 ECRITURE D UNE LIGNE DE LI ********************************
      DO 202 JC=1,NB
      LS=LE1(JC,L1)+LE2(JC,L2)
      IF(LS-1)203,201,203
203   IF(LE1(JC,L1)+2)205,204,205
204   ITRA2(JC)=LE2(JC,L2)
      GO TO 202
205   ITRA2(JC)=LE1(JC,L1)
202   CONTINUE
CX    PRINT 1001
CX    PRINT 1000,(ITRA2(JC),JC=1,NB)
C                 VERIFICATION SI LA LIGNE EST UTILE A LA SOLUTION ***********
      DO 206 KOU=1,NIN1
      CALL OCTAL(IN1(KOU),ITRA,NB)
      DO 207 ILM=1,NB
      LS=ITRA(ILM)+ITRA2(ILM)
      IF(LS-1)207,206,207
207   CONTINUE
      GO TO 211
206   CONTINUE
      GO TO 201
C                 ABSORPTION *************************************************
211   NCO2=0
      DO40 IM=1,NB
      IF(ITRA2(IM)+2)41,40,41
41    NCO2=NCO2+1
```

```
      N2=N2+1
      IF(N2-LIGNE)99,99,212
99    IF(N2-1)43,42,43
43    ITEST=0
44    ITEST=ITEST+1
      NN2=N2-1
      IF(ITEST-NN2)46,46,42
46    NC01=LI(N31,ITEST)
      IF(NC01-NC02)47,47,45
47    DO 70 MLF=1,NB
      IF(LI(MLF,ITEST)+2)71,70,71
71    IF(ITRA2(MLF)-LI(MLF,ITEST))44,70,44
70    CONTINUE
      N2=N2-1
      GO TO 201
45    DO 60 MLF=1,NB
      IF(ITRA2(MLF)+2)61,60,61
61    IF(ITRA2(MLF)-LI(MLF,ITEST))44,60,44
60    CONTINUE
      DO 62 MIA=ITEST,NN2
      MIO=MIA+1
      DO 62 MLF=1,N31
62    LI(MLF,MIA)=LI(MLF,MIO)
      N2=N2-1
      ITEST=ITEST-1
      GO TO 44
42    DO 65 MLF=1,NB
65    LI(MLF,N2)=ITRA2(MLF)
      LI(N31,N2)=NC02
CX    PRINT 3000,(LI(MLF,N2),MLF=1,N31)
CX 3000  FORMAT(1H+,80X,5I5,2X,I5)
201   CONTINUE
200   CONTINUE
C                     ECRITURE OU NOUVEAU TABLEAU LE1 ***************************
      DO 300 I3=1,N2
      DO 300 I4=1,NB
300   LE1(I4,I3)=LI(I4,I3)
CX    PRINT 1001
CX    DO 2002 IPI=1,N2
CX 2002  PRINT1000(LE1(IPR,IPI),IPR=1,NB)
      N1=N2
102   CONTINUE
      GO TO 400
212   PRINT 3
3     FORMAT(1H ,'PARAMETER LIGNE DU PROGRAMME PRINCIPAL TROP FAIBLE')
      GO TO 999
400   IF(NTOUR)895,895,896
895   PRINT 4
4     FORMAT(1H ,//,'   RESULTATS PROPOSES SOUS FORME DISJONCTIVE',//)
      GO TO 897
896   PRINT 898
898   FORMAT(1H ,///,'  RESULTATS PROPOSES SOUS FORME CONJONCTIVE',//)
C                     RECHERCHE DES SOLUTIONS LES PLUS SIMPLES 1 SEULE FONCTION
C                     CLASSEMENT DES IMPLIQUANTS PREMIERS TABLEAU LI ***********
897   N3=N2-1
603   MPAS=0
      DO 600 I1=1,N3
      I2=I1+1
      IF(LI(N31,I1)-LI(N31,I2))600,600,601
601   MPAS=1
      DO 602 I3=1,N31
      LM=LI(I3,I1)
      LI(I3,I1)=LI(I3,I2)
602   LI(I3,I2)=LM
600   CONTINUE
      IF(MPAS)603,604,603
604   CONTINUE
C                     NUMEROTATOON DES IMPLI PREMIERS ***************************
      N32=N31+1
      DO 605 I1=1,N2
605   LI(N32,I1)=I1
      IF(NTOUR)920,899,920
899   PRINT 921,N1
921   FORMAT(1H ,'LISTE DES ',I3,' IMPLICANTS PREMIERS DE LA FONCTION DO
     1NNEE DES SOLUTIONS DISJONCTIVES MINIMALES',//)
      GO TO 922
920   PRINT 923,N1
923   FORMAT(1H ,'LISTE DES ',I3,' IMPLICANTS PREMIERS DE LA FONCTION CO
     1MPLEMENTAIRE DES SOLUTIONS CONJONCTIVES MINIMALES',//)
C                     ETABLISSEMENT DU TABLEAU LE11 & MAXIMUM  COMPTEUR DE LE21***
922   DO 15 J=1,N1
15    PRINT 5,J,((LI(I,J),I=1,NB))
5     FORMAT(1H ,I2,3X,30(I2,2X))
      DO 805 I1=1,N1
      DO 805 I2=1,IKM
805   LE11(I2,I1)=0
      DO 606 K=1,NIN1
      ML=0
      CALL OCTAL(IN1(K),ITRA,NB)
      DO 607 I1=1,N2
      DO 608 I2=1,NB
      LS=ITRA(I2)+LI(I2,I1)
      IF(LS-1)608,607,608
608   CONTINUE
      ML=ML+1
      LE11(K,I1)=1
607   CONTINUE
      LE21(K,1)=ML
      LE21(K,2)=0
      LE21(K,3)=0
      LE21(K,4)=0
606   CONTINUE
CXX   DO 870 IV=1,4
CXX 870   PRINT 5,(LE2(MW,IV),MW=1,NIN1)
CXX   DO 871 IV=1,N2
CXX 871   PRINT 5,(LE1(MW,IV),MW=1,NIN1)
C                     BOUCLE GENERALE   ****************************************
      LE21(1,2)=1
      MAX=0
      KP=0
700   NC=1
      DO 701 K=1,NIN1
701   LE21(K,4)=0
      K1=2
      MOS=0
702   IF(LE21(NC,4))703,704,703
703   NC=NC+1
713   IF(NC-NIN1)702,702,722
722   PRINT 723
723   FORMAT(1H ,'ERREUR')
      GO TO 800
704   IF(LE21(NC,2)-1)706,705,706
705   LE21(NC,3)=LE21(NC,3)+1
      IF(LE21(NC,1)-LE21(NC,3))707,706,706
706   K1=K1+1
      MLF=0
      DO 708 I1=1,N2
      IF(LE11(NC,I1)-1)708,709,708
```

Voir suite page suivante

```
709     MLF=MLF+1
        IF(MLF-LE21(NC,3))708,710,708
708     CONTINUE
710     ITRA2(K1)=I1
        MOS=MOS+LI(N31,I1)
        IF(MAX)712,711,712
712     IF(MAX-MOS)725,711,711
711     NTOT=0
        DO 714 IW=1,NIN1
        IF(LE11(IW,I1)-1)714,715,714
715     LE21(IW,4)=1
714     CONTINUE
        DO 872 IW=1,NIN1
        IF(LE21(IW,4)-1)872,873,872
873     NTOT=NTOT+1
872     CONTINUE
        IF(NTOT-NIN1)875,716,875
716     KP=KP+1
        IF(IKSOL-KP)1886,885,885
886     PRINT 887
887     FORMAT(1H ,'PARAMETER IKM DU PROGRAMME TROP FAIBLE')
        GO TO 999
1886     PRINT 1887
1887    FORMAT(1H ,'PARAMETER IKSOL DU PROGRAMME PRINC. TROP FAIBLE',)
        GO TO 999
885     ITRA2(1)=K1-2
        ITRA2(2)=MOS
CX      PRINT 803,(ITRA2(IV),IV=1,15)
        IF(KP-1)718,719,717
717     IF(LUTION(1,2)-ITRA2(2))720,719,718
718     KP=1
719     DO 721 I2=1,K1
721     LUTION(KP,I2)=ITRA2(I2)
        MAX=MOS
        GO TO 700
720     KP=KP-1
        GO TO 700
707     IF(NC-1)725,800,725
725     DO 726 I3=NC,NIN1
        LE21(I3,2)=0
726     LE21(I3,3)=0
728     NC=NC-1
        IF(NC-1)722,727,729
729     IF(LE21(NC,3))727,728,727
727     LE21(NC,2)=1
        GO TO 700
875     IF(LE21(NC,2)-1)703,730,703
730     NC=NC+1
        IF(LE21(NC,4))730,731,730
731     DO 732 I4=1,NIN1
732     LE21(I4,2)=0
        LE21(NC,2)=1
        GO TO 713
800     DO 2100 I21=1,KP
        KPS=LUTION(I21,1)+2
2104    MPOT=0
        DO 2101 I23=4,KPS
        I22=I23-1
        IF(LUTION(I21,I23)-LUTION(I21,I22))2102,2101,2101
2102    MPOT=1
        MOTE=LUTION(I21,I23)
        LUTION(I21,I23)=LUTION(I21,I22)
        LUTION(I21,I22)=MOTE
2101    CONTINUE

        IF(MPOT)2100,2100,2104
2100    CONTINUE
        I10=1
2157    KPS=LUTION(I10,1)+2
        I11=I10
2150    I11=I11+1
        IF(KP-I11)2155,2154,2154
2154    DO 2151 I30=1,KPS
        IF(LUTION(I11,I30)-LUTION(I10,I30))2150,2151,2150
2151    CONTINUE
        I51=I10+1
        DO 2152 I30=I51,KP
        I12=I30-1
        DO 2153 I40=1,KPS
2153    LUTION(I12,I40)=LUTION(I30,I40)
2152    CONTINUE
        KP=KP-1
        I10=I10-1
2155    I10=I10+1
        IF(KP-I10)2156,2156,2157
2156    IF(NTOUR)910,910,911
911     PRINT 912,KP,LUTION(1,2)
912     FORMAT(1H ,//,2X,I3,' SOLUTIONS CONJONCTIVES MINILALES PROPOSEES A
       1VEC',I3,' LETTRES',//)
        GO TO 913
910     PRINT 801,KP,LUTION(1,2)
801     FORMAT(1H ,//,2X,I3,' SOLUTIONS DISJONCTIVES MINIMALES PROPOSEES AV
       1EC',I4,' LETTRES',//)
913     DO 802 MLF=1,KP
        KPS=LUTION(MLF,1)+2
802     PRINT 803,(LUTION(MLF,IF),IF=3,KPS)
803     FORMAT(1H ,40I3,/)
        IF(NTOUR)999,914,999
914     NTOUR=1
        GO TO 915
999     CONTINUE
        CALL EXIT
        END
```

```
@FOR,ISO ,OCTAL
        SUBROUTINE OCTAL(INJ,ITRA,NB)
        PARAMETER IKV2=11
        DIMENSION ITRA(IKV2),ND(8),ITR(IKV2)
        INV=INJ
        N=0
        MP=0
        ND(1)=0
        ND(2)=1
        ND(3)=10
        ND(4)=11
        ND(5)=100
        ND(6)=101
        ND(7)=110
        ND(8)=111
103     IRES=INV/10
        IF(IRES)101,200,101
200     MP=MP+1
        IF(MP-1)101,101,100
101     IA=INV-(10*IRES)
        IA=IA+1
        NC=ND(IA)
        DO 102 J=1,3
        N=N+1
        IF(N-30)107,107,105
107     NBA=NC/10
        ITR(N)=NC-(10*NBA)
        NC=NBA
102     CONTINUE
        INV=IRES
        GO TO 103
100     N=N+1
        IF(N-NB)104,104,500
104     DO 106 J=N,NB
106     ITR(J)=0
        GO TO 500
105     PRINT 1
1       FORMAT(1H ,'ERREUR OCTAL')
500     CONTINUE
        DO 210 IY=1,NB
        NN=NB+1-IY
210     ITRA(IY)=ITR(NN)
        RETURN
        END
```

par P. BOURDET et M. BOSOM