CLASSIFICATION AND MINIMIZATION OF SWITCHING FUNCTIONS *)

by N. C. de TROYE

Summary

Electronic computers consist of a large number of switching elements, of which there are relatively few types, which together are capable of handling or forming a large number of conditions. A special switching technique, with the aid of diodes, is now being used for those parts of the machine where extremely short switching times are required. For several reasons it is advisable to keep the number of diodes as small as possible. To study the various conditions to be realized with diode circuits, Boolean algebra may be profitably applied. It may be demonstrated that two forms of notation in Boolean algebra, viz. the minimal sum of products and the minimal product of sums, are of particular importance in diode-circuit configurations. Several authors have attempted to arrive at these two forms of notation with varying degrees of success. This paper, too, is an attempt to find from a given Boolean function, also called switching function in view of its application in computer techniques — either the minimal sum of products or the minimal product of sums. It is demonstrated that it is possible to transform any given switching function into a matrix containing only the elements 0 or 1. The number of elements 1 in the various submatrices indicates whether a simplified notation of the switching function is possible. The possibility of easily finding the prime implicants of the switching function is likewise shown. These prime implicants can then be used to determine the minimum sum of products. It is found that this process can be carried out by means of electronic computers. The number of switching functions of \( n \) variables is \( 2^{2n} \). As is demonstrated, it is not necessary to determine the minimum sum of products for all these switching functions if the concept of equivalence class, i.e. the set of all switching functions that are invariant as regards permutation and negation of variables, is introduced. Every equivalence class has a representative and it is only of this representative that the minimal sum of products has to be obtained. Determining the equivalence class for any given switching function of 3 or 4 variables is a relatively simple matter.

Résumé

Les machines à calculer électroniques se composent d'un certain nombre d'éléments de commutation, relativement peu différents, qui ensemble sont en mesure de créer un grand nombre de conditions. Une technique spéciale de la commutation, utilisant des diodes, est pour le moment très employée dans les parties de la machine où des temps de commutation très courts sont exigés. À différents points de vue, il est important d'avoir un nombre de diodes aussi petit que possible. Pour étudier les diverses conditions qui doivent être réalisées à l'aide de circuits à diodes, le mieux est de se servir de l'algèbre de Boole. On peut montrer que deux notations tirées de l'algèbre de Boole: la somme minimum de produits et le produit minimum de sommes, sont particulièrement importantes pour la technique de commutation par diodes. Divers auteurs ont dirigé leurs recherches vers ces deux notations avec un succès variable. Cet article est

également une tentative pour trouver la somme minimum de produits ou le produit minimum de sommes, en partant d’une fonction de Boole donnée qu’on nomme parfois fonction de commutation, étant donné ses applications à la technique de la commutation. On peut montrer qu’il est possible de transformer une fonction de commutation donnée en une matrice qui ne contient que les éléments 0 ou 1. Le nombre d’éléments 1 dans les diverses matrices partielles indique si une forme simplifiée de la fonction de commutation est possible. On peut montrer qu’il est possible d’une façon simple de trouver ce qu’on appelle les implicants primes de la fonction de commutation. Partant de ces implicants primes, on peut déterminer la somme minimum des produits. Il se révèle possible de faire effectuer cette opération par des machines à calculer électroniques. Le nombre de fonctions de commutation de \( n \) variables est \( 2^n \).

On montre qu’il n’est pas nécessaire de déterminer la somme minimum des produits de toutes ces fonctions de commutation. On peut introduire la notion de classe d’équivalence, à savoir la réunion de toutes les fonctions de commutation qui sont invariantes pour une permutation et une négation des variables. Chaque classe d’équivalence a un représentant et il est suffisant de déterminer la somme minimum des produits de ces représentants. Si l’on donne une fonction de commutation arbitraire de 3 ou 4 variables, il est possible d’une manière simple de déterminer à quelle classe d’équivalence appartient cette fonction.

Zusammenfassung

CONTENTS

I. INTRODUCTION
   I.1. The minimization problem

II. FUNDAMENTAL DEFINITIONS
   II.1. Introduction
   II.2. The standard function
   II.3. The number of standard functions

III. THE STANDARD FUNCTION
   III.1. The standard vector
   III.2. The distance matrix and the development of a matrix
   III.3. Computation of the standard functions
   III.4. The classification of the switching functions of 3 and 4 variables

IV. THE MINIMUM DIODE PROBLEM
   IV.1. Introduction
   IV.2. Definition of a minimum diode circuit
   IV.3. The prime implicant
   IV.4. The implicant vector
   IV.5. Simplification of the switching function
   IV.6. Discussion of the results
   IV.7. The "don't care" circuits

APPENDIX

Mechanization of the minimization problem

References
I. INTRODUCTION

I.1. The minimization problem

The analysis and synthesis of relay circuits, which was first given by Shannon 1) in 1938, show that this problem can be approached by means of truth tables and Boolean algebra. If, for instance, a variable $x$ is associated with a relay, one can attach the value $x = 1$ to the activated state and the value $x = 0$ to the inactivated state of the relay. Shannon has shown that the variable $x$ can be considered as an element of a Boolean algebra. Series and parallel circuits of relay contacts then can be represented by the logical product and the logical sum, operations which also are taken from Boolean algebra. A complicated relay circuit then can be described by a Boolean function. One can attempt to reduce this Boolean function to an equivalent one. For our purpose functions with a minimum number of literals are of interest. From this expression other relay circuits can be synthesized which then have fewer elements than the original one. The Boolean function which describes the behaviour of the relay circuit is called switching function, and the problem of reducing a switching function to the simplest equivalent form is called the minimization problem.

Aiken 2), Veitch 3), Karnaugh 4) and Ledley 8) all have developed methods for carrying out this minimization procedure. Their methods, however, have the disadvantage that for more than 5 variables the procedure becomes difficult. A second drawback is that the equivalence of two switching functions which are obtained by permutation or negation of the variables is difficult to notice. Urbano and Muller 5) and Harris 6) succeeded in developing a method which remains clear also for more variables, but recognition of equivalence is still difficult. In McCluskey’s 7) method equivalence is clear only for special types of functions (so-called symmetric functions).

The purpose of this paper is to give a classification of all switching functions of 3 and 4 variables and a method which enables us to tabulate those switching functions of $n$ variables which are essentially different without having to tabulate all possible switching functions of $n$ variables. Also we shall present a procedure for finding that equivalent form of a switching function which leads to the simplest realization as a diode network. The treatment shows that several authors are approaching the minimization problem from an improper direction.
II. FUNDAMENTAL DEFINITIONS

II.1. Introduction

Let B represent a box, which holds a number of switching elements. This box has \( n \) input terminals and 1 output terminal, as illustrated in fig. 1.

On the input terminals we can apply voltages, which have only two discrete values, viz. high or low. The absolute value in volts is of little importance and, therefore, we define the value as 1 if the voltage is high and the value as 0 if the voltage is low.

The voltages \( x_i (i = 0, 1, \ldots, n-1) \) are independent and are called input variables. By means of the switching elements in the box, the input variables generate on the output terminal an output voltage \( y \) which, in the same way, is 0 or 1. The output voltage \( y \) can be expressed as a function of the input voltages \( x_0, \ldots, x_{n-1} \). Shannon has shown that this expression is a Boolean function and, therefore, the rules of Boolean algebra can be applied to the function. The most important rules can be arranged in the following table:

\[
\begin{align*}
    x + x &= x \\
    x + y &= y + x \\
    x + (y + z) &= (x + y) + z \\
    x + (y \cdot z) &= (x + y) \cdot (x + z) \\
    0 + x &= x \\
    1 + x &= 1 \\
    x + \bar{x} &= 1 \\
    \bar{x} + \bar{y} &= \bar{x} \cdot \bar{y} \\
    \bar{x} &= x
\end{align*}
\]

In the formulae (2.1.1) the dot indicates the logical product and the plus sign the logical sum of 2 Boolean variables; 0 (zero) and 1 (one) are called the zero-element and the one-element respectively. The symbol \( \bar{x} \) is called the negation of \( x \). For the sake of brevity one can write \( xy \) instead of \( x \cdot y \).
II.2. The standard function

A constant \( f_m \), a variable \( x_t \), or a function \( f^k \) or \( \phi^k \) will be called binary if it can take only the value 0 or 1. Since each binary quantity \( x \) is idempotent, superscripts are never needed to indicate powers and hence no confusion will arise by using superscripts as indices for functions. Only binary functions of binary variables will be used, \( \phi^k (x_{n-1}, \ldots, x_0) \). Let the number \( j \) be defined by

\[
j = \sum_{i=0}^{n-1} x_i 2^i,
\]

i.e., the value of the binary number we get by considering the \( x_i \) as the \( i \)th digit of that number \(*\)). Apparently \( j \) can have \( 2^n \) values, viz. \( 0(1)2^n-1 \). We can also consider \( (x_{n-1}, \ldots, x_0) \) as the coordinates of one of the vertices of an \( n \)-dimensional unit cube. Then \( j \) numbers the points \( (x_{n-1}, \ldots, x_0) \) and therefore can be used, if necessary, to replace the set \( (x_{n-1}, \ldots, x_0) \). We write

\[
\phi^k (x_{n-1}, \ldots, x_0) = f^k (j).
\]

Special functions of great importance are

\[
\begin{align*}
x_t &= \begin{cases} 0 & \text{if } x_t = 0 \\ 1 & \text{if } x_t = 1 \end{cases} \\
\bar{x}_t &= \begin{cases} 1 & \text{if } x_t = 0 \\ 0 & \text{if } x_t = 1 \end{cases}
\end{align*}
\]

(2.2.3)

By taking products of these functions we can introduce the primitive functions, e.g., \( x_3x_2x_1x_0, \bar{x}_3x_2x_1\bar{x}_0, x_3\bar{x}_0 \).

Of special importance are the complete primitive functions,

\[
\prod_{i+k}^{n-1} x_i \prod_{k}^{n-1} \bar{x}_k,
\]

(2.2.4)

in which \( i \) and \( k \) together take all values \( 0(1)n-1 \). There are \( 2^n \) complete primitive functions which all take the value 1 for only one specific value of \( j \) and the value 0 for all other values of \( j \). This specific value \( m \) of \( j \) can be obtained from (2.2.4) by replacing all variables \( x_1 \) by 1 and all \( x_k \) by 0. The function \( x_3x_2x_1x_0 \), e.g., equals 1 for \( j = 15 \); \( x_3\bar{x}_2x_1\bar{x}_0 \) equals 1 for \( j = 10 \). The complete primitive functions will be denoted by \( g^m \), where \( m \) denotes the value of \( j \) for which the function is equal to 1, e.g., \( g^{15} = x_3x_2x_1x_0 \) and \( g^{10} = x_3\bar{x}_2x_1\bar{x}_0 \). Now it is obvious that each function \( f^k(j) \) can be expressed unambiguously as

\[
f^k(j) = \sum_{m=0}^{2^n-1} f_m g^m.
\]

(2.2.5)

\(*\) We shall distinguish between a logical sum, which we shall denote by \( S \), and the ordinary algebraic sum, which will be denoted by \( \Sigma \).
The function $f^k(j)$ will be called switching function. The number of switching functions $f^k(j)$ is $2^{2n}$ since the $f_m$ can be chosen independently. For theoretical reasons and also for applications to be made further on, it is important to introduce the concept of equivalence classes, i.e. the set of all functions which can be derived from each other by permutations and negations of the variables.

In order to determine the number of equivalence classes of $n$ variables we could use the following procedure. We list all functions $f^k(j)$. To do that, the functions $f^k(j)$ are ordered, their order being

$$k = \sum_{m=0}^{2n-1} f_m 2^m.$$  \hspace{1cm} (2.2.6)

The list of numbers $k = 0 \ldots 2^{2n} - 1$ uniquely represents, therefore, all $f^k(j)$. We take function number 0. This one forms the whole equivalence class 0. The function is marked in the list by means of an asterisk. The next function is number 1. Hence $f_0 = 1$, $f_1 = \ldots = f_{2n-1} = 0$, thus $f^1(j) = \bar{x}_{n-1} \ldots \bar{x}_0$. The function itself is marked in the list. Now we take all permutations and negations of the variables in $f^1(j)$ and obtain in this way the functions of equivalence class 1. These all are removed from the list. The next function to be marked is that with the smallest number which has neither an asterisk nor is removed, and so on. As an example we take the case $n = 2$.

The first function is number 0. The function is $f^0 = 0$. Next is number 1. Hence $f_0 = 1$, $f_1 = f_2 = f_3 = 0$, and $f^1(j) = \bar{x}_1 \bar{x}_0$. The other functions of the class are $x_1 \bar{x}_0$, $x_1 x_0$ with the number 2, 4 and 8, and are removed from the list. The next number is $f^3$. Hence $f_0 = f_1 = 1$, $f_2 = f_3 = 0$, and the function is $\bar{x}_1 \bar{x}_0 + \bar{x}_1 x_0 = \bar{x}_1$. The other members of the class are $x_0$, $\bar{x}_0$, $x_1$ with numbers 10, 5 and 12 which are removed from the list. The next number is 6. Hence $f_0 = f_3 = 0$, $f_1 = f_2 = 1$ and the function is $x_1 x_0 + x_1 \bar{x}_0$. The other member is $x_1 \bar{x}_0 + x_1 x_0$ with number 9 which is removed from the list. The next number is $f^7$. Hence $f_0 = f_1 = f_2 = 1$, $f_3 = 0$, and the function is $\bar{x}_1 \bar{x}_0 + \bar{x}_1 x_0 + x_1 \bar{x}_0 = \bar{x}_1 \bar{x}_0 + \bar{x}_1 x_0 + \bar{x}_1 \bar{x}_0 + x_1 \bar{x}_0 = \bar{x}_1 + \bar{x}_0$. The other functions of the class are $x_1 + \bar{x}_0$, $\bar{x}_1 + x_0$, $x_1 + x_0$ with numbers 13, 11 and 14 which are removed from the list.

The next and last function is number 15. Hence $f_0 = f_1 = f_2 = f_3 = 1$, and the function is $\bar{x}_1 \bar{x}_0 + \bar{x}_1 x_0 + x_1 \bar{x}_0 + x_1 x_0 = 1$. There are, therefore, 6 equivalence classes. Of every class one function, e.g., the function with the lowest number, can be taken as representing the class. This function will be called standard function.

II.3. The number of standard functions

The determination of the number of standard functions is possible without writing down the whole list of possible functions, but the problem is far from
trivial even for rather small values of $n$. For $n$ up to 6 the number has been calculated. According to Semon \(^{10}\) one has

<table>
<thead>
<tr>
<th>Number of variables</th>
<th>Number of functions $f^k(j)$ $F(n)$</th>
<th>Number of equivalence classes $S(n)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>256</td>
<td>22</td>
</tr>
<tr>
<td>4</td>
<td>65 536</td>
<td>402</td>
</tr>
<tr>
<td>5</td>
<td>4 294 967 296</td>
<td>1 228 158</td>
</tr>
<tr>
<td>6</td>
<td>18 446 744 073 709 551 616</td>
<td>400 507 806 843 728</td>
</tr>
</tbody>
</table>

The number of standard functions $S(n)$, although much smaller than the number $F(n) = 2^{2^n}$ of switching functions, is nevertheless exceedingly large when $n > 4$. But without performing the lengthy calculations that give rise to the exact values given above we can easily get a lower bound of $S(n)$ that is surprisingly accurate. Indeed, one has

$$S(n) T(n) \geq F(n),$$

$T(n) = n! 2^n$ being the number of possible permutations and negations. This inequality can at once be improved to

$$S(n) T(n) > F(n),$$

since for each $n$ there are obvious examples of functions which permit only a smaller number of transformation than $T(n)$. For instance, each function of less than $n$ variables, e.g., $x_i$, permits fewer than $n!2^n$ transformations. Hence $S(n) > 2^{(2^n-n)/n!}$. This yields the following lower bounds:

<table>
<thead>
<tr>
<th>$n$</th>
<th>$2^{2^n-n}/n!$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>170</td>
</tr>
<tr>
<td>5</td>
<td>1 118 481</td>
</tr>
<tr>
<td>6</td>
<td>400 319 966 877 377</td>
</tr>
</tbody>
</table>

In the next chapter a method will be developed that will enable the computation of the standard functions without calculating the whole list of all possible functions. At the same time the procedure has the advantage of yielding for a given function the standard function of the equivalence class to which it belongs.
III. THE STANDARD FUNCTION

III.1. The standard vector

A switching function \( f^k(j) \) is determined by its components \( f_m \) which have been defined in section II.2. The function can be represented by a matrix. The rows of this matrix are formed by the digits of those indices \( m \), written in binary notation, for which \( f_m = 1 \). For the switching function

\[
f(x_3x_2x_1x_0) = \bar{x}_3\bar{x}_2\bar{x}_1\bar{x}_0 + \bar{x}_3\bar{x}_2x_1x_0 + \bar{x}_3x_2\bar{x}_1\bar{x}_0 + x_3\bar{x}_2\bar{x}_1x_0 + x_3x_2x_1x_0,
\]

(3.1.1)

for instance, \( f_0 = f_3 = f_4 = f_9 = f_{15} = 1 \), and therefore the matrix representation of this function is

\[
\begin{array}{c}
0000 \\
0001 \\
0011 \\
0100 \\
1001 \\
1111
\end{array}
\]

(3.1.2)

The matrix representation, at the same time, has a geometrical meaning. If, e.g., every column is associated with a coordinate \( x_i \) in an \( n \)-dimensional space, then every row in the matrix will represent the coordinates of one of the vertices of an \( n \)-dimensional unit cube. Permutation of variables is equivalent to permutation of the corresponding columns of the matrix, while negation of a variable means the same as negation of the corresponding column of the matrix. Thus, the matrix (3.1.2) is transformed into the matrix (3.1.3) if the variables \( x_2 \) and \( x_1 \) are permuted and if the negation of \( x_0 \) is taken:

\[
\begin{array}{c}
0001 \\
0000 \\
0100 \\
0011 \\
1000 \\
1110
\end{array}
\]

(3.1.3)

Because there are \( n! \) possible permutations and \( 2^n \) possible negations, there are \( n!2^n \) possible transformations of the variables which carry over the function \( f^k(j) \) into the other functions of its equivalence class. From every transformation follows a set of values of \( m \) for which \( f_m = 1 \). We rearrange the values of \( m \) as an ascending progression and introduce the concept of size, indicated by the number \( F \), say, which we obtain by considering the values of \( m \) as the digits
of a number in a numeral system with a sufficiently high radix. If the \( n!2^n \) transformations are applied to the representations of every equivalence class, a number of sets of arranged values for \( m \) is found, a system which can be arranged in a table according to the size \( F \). This table, which even for 4 variables becomes quite extensive, is called the table of input rearrangements.

In order to find out to which equivalence class a certain function \( f^k(j) \) belongs, we must determine the set of values \( f_m \) of this function and search through our table. Now, we will show that this table can be very much reduced by introducing three invariants. If we add to the matrix a column, the elements of which are equal to the algebraic sum of the digits of the respective rows, it will be clear that this "sum column" is invariant with respect to permutation of the variables. If we add, in the same way, a "sum row", the elements of which are equal to the algebraic sum of the digits in the respective columns, this sum row upon permutation of the variables will be subject to the same permutation.

The elements of the sum row and sum column will be denoted by the weights of the respective columns and rows. These weights can also be considered as elements of a row vector and a column vector \(^*\). Now we may rearrange the elements of the column vector as a non-descending progression and apply the same procedure to the elements of the row vector. The combined set of rearranged column elements followed by the set of rearranged row elements defines the weight vector. For clarity a semi-colon will be put between the two groups of elements. The weight vector found in this way, clearly, is invariant under permutation of the variables \( x_t \).

Negation of a variable corresponds to negation of the related column of the matrix. The weight vector, in general, will change under this procedure. We now introduce the following conventions. If the weight of a column is greater than half the number of rows, we must take the negation of the column in question. The weight vector must be determined only after these transformations have been carried out. If the weight of a column is equal to half the number of rows, both versions of the matrix should be considered.

Also, we can introduce the concept "size" of a weight vector, indicating the number which we obtain by considering the elements of the vector as the digits of a number in a numeral system with sufficiently high radix. If several versions of a matrix are to be considered, that version must be chosen which leads to a weight vector of minimal size. If there is more than one matrix leading to a weight vector of this smallest size, at least one of them should be taken into account. In every case, the weight vector found in this way will be called standard vector. By virtue of definition, the standard vector is invariant

\(^*\) The term vector is used here merely for convenience, with the meaning only of a set of numbers. No other properties of vectors are necessary or will be assumed.
under permutations and negations of the variables. By no means can we say that, conversely, to every standard vector belongs uniquely a matrix. First of all, there are cases that more than one matrix are leading to this smallest sized weight vector; secondly, for all functions of the same equivalence class the same standard vector is the invariant; finally, also two functions belonging to different equivalence classes may lead to the same standard vector. As the function representing the equivalence class, we will choose from the class that function which on the one hand has the standard vector as its weight vector and on the other hand has a minimal $F$. This function will be called the standard function of the class and its matrix the standard matrix. For example we take the switching function $f^k(j)$, having the following matrix:

\[
\begin{array}{cccc}
0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 \\
\end{array}
\]

Here we find one column of weight 4, of which column, therefore, we must take the negation. Also there is a column of weight 3, of which we must consider two versions. The two versions, however, have the same weight vector $(01122; 2223)$ and even the two matrices in this case can be obtained from each other by permutations; consequently, being the smallest weight vector, the vector $(01122; 2223)$ is at the same time the standard vector. The matrix with this vector as its weight vector and having a minimal $F$ is the standard matrix:

\[
\begin{array}{cccc}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 \\
\end{array}
\]

It will be clear that 2 matrices having standard vectors with unequal row vectors never can be derived from each other. This means that such matrices belong to different equivalence classes. However, the converse is not true because it is possible that there belong two or more standard matrices to one standard vector.

An example of this case may be taken from the standard vector $(01122; 1122)$. From this vector we can derive matrices representing three equivalence classes,
This complication does not arise for 2 and 3 variables. However it is possible, at least in the case of 4 variables, by introducing two more invariants, to distinguish between matrices belonging to different equivalence classes having the same standard vector.

### III.2. The distance matrix and the development of a matrix

Consider row \( i \) and row \( j \) of an arbitrary matrix. These rows consist of a sequence of 1's and 0's. Now define a quantity \( \Delta_y \) as the number of positions in which the rows differ from each other. Then \( \Delta_y \) is invariant against permutations of the columns, for by permuting columns no differences in corresponding positions of two rows can appear or disappear.

We can, if we want, also attach a geometrical meaning to the quantity \( \Delta_y \), because, if we recall that every row of the matrix has a representation in one of the vertices of a unit cube, the number of positions in which two rows differ from each other is, in fact, the square of the geometrical distance of the corresponding vertices. We will simply call \( \Delta_y \) a "distance".

If, now, we have a number of standard matrices having equal standard vectors, then for every one of these matrices we compute the numbers \( \Delta_y \). These numbers can be arranged in a square symmetric matrix, the "distance matrix", of which the element in row \( i \) and column \( j \) is \( \Delta_y \). For our purpose it will be better to arrange the elements in the rows in order of magnitude. It then appears that for every one of the standard matrices, the corresponding distance matrix has one or more typical rows.

If we apply, e.g., this procedure to the 3 matrices (3.1.6) we find the following matrices \( \Delta \):

\[
\begin{align*}
01122 & \quad 01122 & \quad 01122 \\
10213 & \quad 10213 & \quad 10213 \\
12031 (a) & \quad 12031 (b) & \quad 12033 (c) \\
21304 & \quad 21304 & \quad 21302 \\
23340 & \quad 23140 & \quad 23320
\end{align*}
\]

or after ordering

\[
\begin{align*}
01122 & \quad 01122 & \quad 01122 \\
01123 & \quad 01123 & \quad 01123 \\
01123 (a) & \quad 01123 (b) & \quad 01233 (c) \\
01124 & \quad 01234 & \quad 01223 \\
02334 & \quad 01234 & \quad 02233
\end{align*}
\]
Inspection shows that in this latter matrix the distance 4 does not occur at all whereas the other two differ in, e.g., the rows 01234 and 02334 (or 01124).

Yet it may also be possible that there are matrices having equal standard vectors and equal distance matrices. In this case we need a third invariant to distinguish between these matrices. For that purpose we may define a submatrix \( D_0 \) of a column \( k \) as the matrix that remains after striking out column \( k \) and all rows \( i \) for which \( a_{ik} = 1 \). In the same manner we may define a submatrix \( D_1 \) as the matrix that is left after striking out the \( k \)th column and all rows with \( a_{ik} = 0 \). Then the original matrix can be developed symbolically as

\[
M = 0 \cdot D_0 + 1 \cdot D_1. \tag{3.2.3}
\]

As this development will be used only for columns specified by a particular weight, \( k \) in the above definition has, therefore, no absolute meaning.

As an example we may develop the matrix (3.2.4) with respect to the column of weight 2, which yields:

\[
\begin{align*}
0000 & = 0 \cdot 000 + 1 \cdot 001 \\
0001 & = 0 \cdot 001 + 1 \cdot 001 \\
0010 & = 0 \cdot 010 + 1 \cdot 010 \\
0101 & = 0 \cdot 11 \quad 1 \cdot 011 \\
0110 & = 0 \cdot 11 \quad 1 \cdot 011 \\
1011 & = 0 \cdot 11 \quad 1 \cdot 011
\end{align*}
\]

Of the submatrices, again, we can determine the weight vectors, and the weight vector of (3.2.4) can be written symbolically, with respect to the column of weight 2, as follows:

\[
(011223; 1233) = 0 \ (0113; 122) + 1 \ (11; 011). \tag{3.2.5}
\]

Expression (3.2.5) is equally invariant under permutation of the columns of the matrix.

### III.3. Computation of the standard functions

In this section a method that enables us to compute the representations of the equivalence classes will be developed. First it is possible to compute the standard vectors in a simple way. From these standard vectors we can then immediately find the standard matrices. We start to examine the weights of the rows. The weight \( k \) of a row denotes the number of times that the digit 1 appears in a row of \( n \) elements. Because the weight is independent of the arrangement of the digits, the number of ways to obtain a weight \( k \) is

\[
G_k^n = \frac{n!}{k! \ (n-k)!}. \tag{3.3.1}
\]
Now consider matrices that have \( r \) rows of weight \( k \). If there are \( G_k^n \) possibilities to obtain a weight \( k \), there will be

\[
M_r^G = \frac{G_k^n!}{(G_k^n - r)! r!}
\]  

(matrices of \( r \) rows which all have weight \( k \)).

The weight vector of such a matrix will be called a substandard vector (ss-vector).

On the other hand, several matrices may belong to one ss-vector. At least for \( n \leq 4 \), we have found that these matrices can be all obtained from one another by permutations. Thus, they belong to one equivalence class, of which they form a subclass. As the representation of this subclass we choose the matrix that has a minimal \( F \). In general this will not be the same matrix as the representation of the total equivalence class as defined in section III.1.

In Table IV we have listed for \( n = 1, 2, 3 \) and 4 the ss-vectors and their representations given by means of their components \( f_m \). For example we will compute the standard vectors of 4 variables. At first we shall write down the column vectors. In order to carry out this procedure it is useful to recall the following properties. The number of times that weight \( k \) can appear is given by (3.3.1). For 4 variables we have the following table:

<table>
<thead>
<tr>
<th>( k )</th>
<th>( G_k^4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>

From the definition of the standard vector, it follows that the sum of the digits of the column vector is equal to the sum of the digits of the row vector, since both are the number of 1's in the matrix. Because only standard vectors are considered, in the case of a matrix of \( p \) rows and \( n \) columns the sum of the digits in the column vector can never exceed

\[
\frac{1}{2} np,
\]

the weight of a column never being greater than half the number of rows.

Now we may proceed to find the column vectors. Because \( n = 4 \) we need only consider \( p \leq 8 \) (see the last but one paragraph of this section).

From (3.3.1) we deduce how the smallest column vectors are formed; i.e.
The arranged column vector will consist of \( p \) digits
\[
s_0 s_1 \ldots s_{p-2} s_{p-1}.
\] (3.3.5)

The column vector that follows in size after \( s_0 s_1 s_2 \ldots s_{p-2} s_{p-1} \) will be \( s_0 s_1 \ldots s_{p-2} a \) where \( a = s_{p-1} + 1 \).

In the next column vector, again, the last digit is increased by 1; etc., until we have reached the value 4 as the highest weight that can occur in a matrix.

Now \( s_{p-2} \) will be increased by 1 and \( s_{p-1} \) decreased to this same value and the process repeated. In this way we continue, mindful of not exceeding \( G_k^4 \) and of not violating the inequality
\[
\sum_{i=0}^{p-1} s_i \leq \frac{1}{2} np.
\] (3.3.6)

For 6 rows, e.g., the following list of arranged column vectors can be drawn up

\[
\begin{align*}
011112 \\
011113 \\
011114 \\
011122 \\
\vdots \\
022233
\end{align*}
\]

The sum of the digits of this last column vector is 12. In view of (3.3.6) the table goes on as follows:

\[
\begin{align*}
111122 \\
111123 \\
111124 \\
111133 \\
\vdots \\
222222
\end{align*}
\]
Now that the column vectors are known, the appertaining row vector should be found. This may be done with the aid of the ss-vectors. Suppose the column vector to be

\[ s_0 \ s_1 \ \ldots \ s_{p-2} \ s_{p-1} \]  

(3.3.7)

A matrix belonging to (3.3.7) can be subdivided into a number of smaller matrices, each having rows of one weight \( k \). These smaller matrices have ss-vectors according to table IV. By permuting the elements of the row-vector part of the ss-vector independently, and adding these new-found row vectors, we get, upon rearrangement of the digits, the possible row-vector combinations of the standard vector, the column vector of which was given.

For example we choose the column vector 011222. The ss-vectors then are

\[(0; 0000), (11; 0011) \text{ and } (222; 0222) \text{ or } (222; 1113) \text{ or } (222; 1122).\]

The first two ss-vectors combined produce a row vector 0011. This row vector can be added to the ss-vector (222; 0222) in two ways that yield different vector sums. Equal sums are of no interest with standard vectors. The two ways are:

\[
\begin{array}{c}
0011 \\
0222 \\
0233
\end{array}
\begin{array}{c}
1001 \\
0222 \\
1223
\end{array}
\]

This may be done in the same way for the other ss-vectors. The result is

\[
\begin{array}{cccccccc}
0011 & 1001 & 0011 & 0110 & 0011 & 0101 & 1100 \\
0222 & 0222 & 1113 & 1113 & 1122 & 1122 & 1122 \\
0233 & 1223 & 1124 & 1223 & 1133 & 1223 & 2222
\end{array}
\]

The third row vector should be omitted, because a weight 4 cannot occur in a standard vector from six rows.

The following row vectors remain:

\[
\begin{array}{c}
0233 \\
1133 \\
1223 \\
2222
\end{array}
\]

In this way we found 4 standard vectors. Applying this procedure to every column vector, we can find all possible standard vectors. These have been arranged in table V.

Now that the standard vectors have been determined, the appertaining matrices should be calculated. We start from a given standard vector, which, as we know, consists of a column vector and a row vector. Now, from table IV we choose those ss-vectors the column vectors of which combined yield the
column vector of the standard vector. In order to avoid the labour of obtaining all possible permutations of the row vector of the standard vector, we choose one of the ss-vectors, preferably the one with maximal $r$ (see table IV), which we agree beforehand not to permute.

Now consider permutations of the row vectors of the other ss-vectors and add the corresponding elements. Only those combinations which yield the row vector of the given standard vector may produce the standard matrices. If among these matrices there are matrices with columns the weights of which are equal to half the number of rows, the negation of such columns is taken and the weight vector is again determined. It may then occur that a smaller weight vector is found. These matrices thus are not standard matrices of the given standard vector, and can therefore be rejected. For example we may consider the standard vector (012223; 2233).

The ss-vectors that together form the column vector are

\[
\begin{align*}
0; 0000 & \quad 0; 0000 & \quad 0; 0000 \\
1; 0001 & \quad 1; 0001 & \quad 1; 0001 \\
222; 0222 & \quad 222; 1113 & \quad 222; 1122 \\
3; 0111 & \quad 3; 0111 & \quad 3; 0111
\end{align*}
\]

The ss-vectors (222; 0222) or (222; 1113) or (222; 1122) will not be subjected to permutation. Therefore to these ss-vectors, all 16 possible permutations of (1; 0001) and (3; 0111) should be added. The only 6 combinations that produce the row vector of the given standard vector are

\[
\begin{align*}
0; 0000 & \quad 0; 0000 & \quad 0; 0000 \\
1; 1000 & \quad 1; 0010 & \quad 1; 0001 \\
222; 0222 & \quad 222; 1113 & \quad 222; 1122 \\
3; 1011 & \quad 3; 1110 & \quad 3; 1110 \\
\hline
012223; 2233 & \quad 012223; 2233 & \quad 012223; 2233
\end{align*}
\]

It is not necessary actually to perform all 16 permutations, for from the desired row vector of the standard vector and from the given fixed ss-vector the permutations of the other ss-vectors practically follow at once. The 6 combinations which have just been found lead to the matrices
Negation of the columns of weight 3 shows that the matrices (b), (c) and (d) yield the weight vectors:

011233; 2233
011134; 2233
011134; 2233

Because these weight vectors are smaller than the standard vector, the matrices are no standard matrices of the given standard vector. Thus they must be rejected. In as much as the matrices (3.3.8) (e) and (f) are equivalent, they change into each other under permutation of columns. We are therefore left with 2 matrices, which by permutations of columns can be written as matrices with minimal $F$.

<table>
<thead>
<tr>
<th></th>
<th>0000</th>
<th>0001</th>
<th>0110</th>
<th>0111</th>
<th>1010</th>
<th>1100</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>0000</td>
<td>0000</td>
<td>0011</td>
<td>0011</td>
<td>0100</td>
<td>1000</td>
</tr>
<tr>
<td>(b)</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0100</td>
<td>1000</td>
</tr>
<tr>
<td>(c)</td>
<td>0000</td>
<td>0011</td>
<td>0011</td>
<td>0011</td>
<td>0100</td>
<td>1000</td>
</tr>
<tr>
<td>(d)</td>
<td>0000</td>
<td>0011</td>
<td>0011</td>
<td>0011</td>
<td>0100</td>
<td>1000</td>
</tr>
<tr>
<td>(e)</td>
<td>0000</td>
<td>0011</td>
<td>0011</td>
<td>0011</td>
<td>0100</td>
<td>1000</td>
</tr>
<tr>
<td>(f)</td>
<td>0000</td>
<td>0011</td>
<td>0011</td>
<td>0011</td>
<td>0100</td>
<td>1000</td>
</tr>
</tbody>
</table>

These matrices are the standard matrices of the given standard vector and they represent the two equivalence classes belonging to this standard vector. Also, it will not be necessary here really to perform all possible permutations in order to obtain the matrices (3.3.9) (a) and (b).

Consider, e.g., (3.3.8) (a). If possible, the row 0000 ($m = 0$) should be present. It is present, and invariant under every permutation. Next a row 0001 ($m = 1$) should possibly appear. This can be arranged by permutation of the columns 0 and 3 with the effect that row 1000 changes in 0001. The matrix now reads:

<table>
<thead>
<tr>
<th></th>
<th>0000</th>
<th>0001</th>
<th>1010</th>
<th>1100</th>
<th>0110</th>
<th>1011</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>0000</td>
<td>0001</td>
<td>1010</td>
<td>1100</td>
<td>0110</td>
<td>1011</td>
</tr>
</tbody>
</table>

and column 0 should further be left unaltered. A row $m = 2$ cannot exist, nor
can \( m = 3, 4, 5 \). A row \( m = 6 \) can be arranged in 6 different ways but a row \( m = 7 \) in only 2 ways, i.e. by permutation of columns 2 and 3 or by successive permutations of the columns 1, 2 and 1, 3, which both produce a row \( m = 6 \).

The resulting matrices, however, are identical, viz.

\[
\begin{align*}
0000 \\
0001 \\
0110 \\
1100 \\
1010 \\
0111
\end{align*}
\]

Here only columns 1 and 2 may be interchanged, but this will have no effect on the matrix.

With the method given above all standard matrices of the equivalence classes of 3 and 4 variables have been computed. Computation of matrices of more than 4 rows (3 variables) or 8 rows (4 variables) is not of interest as they can be derived from matrices with fewer than 4 or 8 rows, respectively.

In connection with a function \( f^k(j) \), given by its components \( f_m \), we may also consider a function \( f^k(j) \) which is determined by those components \( f_m \) which do not occur in \( f^k(j) \). The function \( f^k(j) \) is called the complementary function of \( f^k(j) \). It will be obvious that the complementary functions of all functions of an equivalence class also form an equivalence class, the so-called complementary equivalence class. For several reasons it will be useful, for \( r > \frac{1}{2}n \), slightly to change the definition of standard function. In such cases we will take as standard function the complementary function of the standard function of the complementary equivalence class. This will spare us the computation of the 164 representations of the equivalence classes of more than 8 rows, i.e. about 40% of the 402 equivalence classes of 4 variables.

### III.4. The classification of the switching functions of 3 and 4 variables

With the aid of the 3 invariants given in sec. III.2, we are able to classify all switching functions of 3 and 4 variables. Two tables have been made, which are used in the following way. Of an arbitrary function \( f^k(j) \) we first determine whether the number of components \( f_m \) is more than 4 (3 variables) or 8 (4 variables). When such is the case we compute the function \( f^k(j) \), i.e. the function with those components \( f_m \) which did not appear in the original function \( f^k(j) \). In this way we shall always obtain a function with 4, respectively 8, or fewer components. Of this function we determine the standard vector, which we search for in the list of standard vectors (table I). This list has 4 columns. The first column gives the number of rows of the matrix of \( f^k(j) \). The second column gives the standard vectors arranged according to their sizes. The third column
denotes the place in the table of the function representing the equivalence classes to which \( f^k(j) \) belongs. The fourth column, “indications”, tells us which invariants we must use, if to the standard vector there belong several standard matrices. In practice it proves convenient not to determine all elements \( \Delta y \). Often it is sufficient to state that, e.g., 3 pairs of rows have a mutual distance equal to 4. This will be denoted by 3. \( (\Delta = 4) \). An element \( \Delta y = 4 \) is possible only if the sum of the weights of 2 rows of the matrix is equal to 4.

The table representing the equivalence classes (table II) is made up as follows. The first column again gives the number of rows of the representing matrices. The second and third columns, \( s \) and \( \bar{s} \), denote an arbitrarily assigned prescript by which the equivalence class is designated. To the number \( s \) belongs the equivalence class \( f^k(j) \), the components \( f_m \) of which are given in column 4. To the number \( \bar{s} \) belongs the complementary class, the representations of which have the components \( f_m \) that do not occur in column 4; \( s \) and \( \bar{s} \) also refer to table III, which will be treated in chap. IV. Column \( p \) denotes the number of different functions in the equivalence class. The last column gives the standard vector belonging to the equivalence class \( f^k(j) \).
IV. THE MINIMUM DIODE PROBLEM

IV.1. Introduction

A function which is expressed in the form

\[ f^k(j) = \sum_{m=0}^{2^n-1} f_m g^m \]  

(4.1.1)

is said to be written in the conjunctive normal form. The 402 standard functions which are computed in chapter III, and given in table II, have been defined by their components \( f_m \) and, therefore, are written in the conjunctive normal form. With the aid of (2.1.1), however, many other notations of the same function are possible. Furthermore, it may be that one of them is preferred for certain reasons above the conjunctive normal form. It is, therefore, necessary to define what we understand by the simplest notation.

IV.2. Definition of a minimum diode circuit

An expression

\[ \bar{x}_0 x_1 \bar{x}_2 x_3 \]  

(4.2.1)

can be realized by a series circuit of relays. A relay consisting of a coil \( X_i \) and a contact \( x_i \) that is closed in the activated state will be called \( x_i \). A relay consisting of a coil \( X_i \) and a contact \( x_i \) that is open in the activated state will be called \( \bar{x}_i \). The expression (4.2.1) can be realized by a series circuit of 4 relays, viz. 2 relay contacts \( x_1 \) and \( x_3 \) that are closed if \( X_1 \) and \( X_3 \) are activated and 2 relay contacts \( x_0 \) and \( x_2 \) that are closed if \( X_0 \) and \( X_2 \) are not activated. The first two relays are normally *) open, whereas the other two relays are normally closed. The circuit is shown in fig. 2.

![Fig. 2. A series circuit of 4 relays.](image)

An expression

\[ \bar{x}_0 + x_1 \]  

(4.2.2)

can be realized by a parallel circuit of 2 relays. The circuit (4.2.2) is a parallel circuit of a normally open (\( x_1 \)) and a normally closed (\( \bar{x}_0 \)) relay contact. The circuit is shown in fig. 3.

![Fig. 3. A parallel circuit of 2 relays.](image)

*) Normally means "not activated".
An expression
\[ \overline{x}_0x_1x_2 + x_0x_1x_2 \]  
represents a series-parallel circuit and which can be constructed as shown in fig. 4.

We can, however, apply the rules given in (2.1.1) and we therefore find

\[ \overline{x}_0x_1x_2 + x_0x_1x_2 = (\overline{x}_0 + x_0)x_1x_2 = x_1x_2. \]  

The corresponding circuit is shown in fig. 5. The circuit is obviously simpler than the circuit drawn in fig. 4.

Fig. 4. A series-parallel circuit of 6 relays. Fig. 5. The simplified circuit of fig. 4.

An expression
\[ \overline{x}_0\overline{x}_1x_2x_3 + x_0x_1x_2 + x_0x_1x_3 \]  
or, what is the same,

\[ \overline{x}_0\overline{x}_1x_2x_3 + x_0x_1(x_3 + x_2) \]  
can be realized as shown in fig. 6.

This relay circuit requires 8 relay contacts. The same function (4.2.5) can be realized, however, by 7 relay contacts by introducing a bridge circuit (fig. 7).

Fig. 6. A series-parallel circuit of 8 relays. Fig. 7. The simplified circuit of fig. 6, a so-called bridge circuit.

Nevertheless it is impossible to transform (4.2.5) into a seven-letter formula. Thus a problem is to make a given switching function with as few relay contacts as possible. This is called the minimization problem.

A second minimization problem which is important in telephone technique, reads, e.g.: how can we construct a circuit where the number of switching
operations done by the relays is distributed as uniformly as possible over the relays? From a technical point of view it is not desirable that the number of contact variations of some relays is unproportionally large, because the wear will be too high. This problem is known as the "minimal springload distribution".

A third minimization problem which often occurs in the construction of computers is the "minimum diode problem".

The expression (4.2.1) can be realized by means of a diode circuit (fig. 8).

Here and in what follows, it is assumed that of all variables both polarities, i.e. the variable and its negation, are available.

If all the voltages \( x_0, x_1, x_2 \) and \( x_3 \) are high (=1), then and only then the output voltage will be high (=1). If one or more of the voltages \( x_0, x_1, x_2 \) or \( x_3 \) is low (=0) then the output voltage will be low (=0). Such a circuit is called an AND-circuit.

The expression (4.2.2) can be realized by means of a diode circuit (fig. 9).

If one or more voltages \( x_0, x_1 \) are high (=1) then the output voltage will be high (=1). If \( x_0 \) and \( x_1 \) are low (=0), then and only then the output voltage will be low (=0). Such a circuit is called an OR-circuit.

It is possible to make a combination of AND- and OR-circuits. The expression (4.2.5) can be realized as shown in fig. 10.

With present-day techniques it is not easily possible to make a series circuit of more than one AND- and one OR-circuit. This means that it is not allowed to simplify (4.2.5) into (4.2.6). Thus parentheses are not permitted. The minimum diode problem, therefore, is equivalent to the finding of a function, which is called the minimum sum of products, consisting of as few literals as possible.
IV.3. The prime implicant

To simplify a switching function we start from the conjunctive normal form. We take, e.g., a switching function of the following form:

\[ f(x_3x_2x_1x_0) = \bar{x}_3\bar{x}_2\bar{x}_1\bar{x}_0 + \bar{x}_3\bar{x}_2x_1x_0 + \bar{x}_3x_2\bar{x}_1\bar{x}_0 + \ldots \]

(4.3.1)

This function can be written

\[ f(x_3x_2x_1x_0) = \bar{x}_3x_2(x_1\bar{x}_0 + x_1x_0 + x_1x_0) + \bar{x}_3\bar{x}_1(x_2\bar{x}_0 + x_2x_0 + x_2x_0). \]

(4.3.2)

The terms in parentheses are equal to 1, so that the function can be simplified to

\[ f(x_3x_2x_1x_0) = \bar{x}_3 + \bar{x}_2 + \bar{x}_1. \]

(4.3.3)

This function is the simplest form of (4.3.1) (without parentheses). It can be realized with 6 diodes.

The function

\[ f(x_3x_2x_1x_0) = \bar{x}_3\bar{x}_2\bar{x}_1\bar{x}_0 + \bar{x}_3\bar{x}_2x_1x_0 + \bar{x}_3x_2\bar{x}_1\bar{x}_0 + \bar{x}_3x_2x_1\bar{x}_0 + \bar{x}_2x_1x_0 \]

(4.3.4)

can be written as follows:

\[ f(x_3x_2x_1x_0) = \bar{x}_3\bar{x}_2(x_2\bar{x}_1 + x_2x_1 + x_2x_1) + \bar{x}_3\bar{x}_1(x_2\bar{x}_0 + x_2x_0 + x_2x_0 + x_2x_0) + \bar{x}_2x_1x_0(\bar{x}_3 + \bar{x}_3). \]

(4.3.5)

The terms in parentheses are equal to 1, and the function can be simplified to

\[ f(x_3x_2x_1x_0) = \bar{x}_3\bar{x}_2 + \bar{x}_3\bar{x}_1 + \bar{x}_2x_1x_0. \]

(4.3.6)

We can prove, however, that the term \( \bar{x}_3\bar{x}_2 \) is redundant. The simplest notation of (4.3.4) is, actually,

\[ f(x_3x_2x_1x_0) = \bar{x}_3\bar{x}_0 + \bar{x}_3\bar{x}_1 + \bar{x}_2x_1x_0. \]

(4.3.7)

The function can be realized with 10 diodes.

The procedure described above is equivalent to an attempt to bring a product outside the parentheses in part of the conjunctive normal form in such a manner that the expression inside the parenthesis equals 1. We now use the following definition: a function \( q \) implies \( f \), or \( q \) is an implicant of \( f \), if from \( q = 1 \) it follows \( f = 1 \).

For instance if

\[ f = \pi^1 + \pi^2 + \ldots + \pi^k \]

then each \( \pi^i \) is an implicant of \( f \). We shall restrict ourselves further on to
implicants $\pi^i$ that are primitive functions, that is, products of a number of variables or their negations. If the omission of a variable in $\pi$ results in a new product that does not imply $f$, then we call $\pi$ a prime implicant. The geometrical meaning of a prime implicant is, therefore, a unit cube of dimension $j$ on all vertices of which $f = 1$, which is not part of a unit cube of higher dimensions on all vertices of which again $f = 1$.

A cover $\sigma$ of $f$ is, by definition, any set of implicants $\pi^1, \pi^2 \ldots \pi^t$ of $f$ with the property that

$$f = \pi^1 + \pi^2 + \ldots + \pi^t.$$ 

By means of the diode technique described in section IV.2 one can realize $f$ by realizing the $\pi^i$ as AND-circuits and combining these by means of an OR-circuit. The number of diodes which is necessary for this realization is called the cost of the cover $\sigma$. Our problem is how to find a cover of minimal cost, a so-called optimal cover.

A necessary condition for $\sigma$ to be an optimal cover is that all $\pi^i$ are prime implicants. Indeed, the cost of a cover is the sum of the costs of its implicants, where, in general, the cost of an implicant of dimension $j$ is $n-j+1$, since $n-j$ diodes are needed for the AND-circuit and 1 diode for the OR-circuit. If the dimension is $n-1$, i.e., if the implicant is equal to one of the variables, then the cost is not 2 as suggested by the general formula but 1 since the AND-diode is no longer necessary. On the other hand, if the cover consists of one implicant only then its cost is $n-j$ instead of $n-j+1$ since the OR-diode is no longer necessary. If the cover consists of one implicant only and the dimension is $n-1$, then its cost is 0.

Now, if an implicant $\pi^i$ in $\sigma$ is no prime implicant, then by omitting a variable in $\pi^i$ such that the new $\pi^i*$ is again an implicant, which is possible by definition, the set $\sigma*$ consisting of all implicants in $\sigma$ without $\pi^i$ and with $\pi^i*$ is again a cover, the cost of which is one less than that of $\sigma$. Hence we shall restrict ourselves to covers consisting only of prime implicants.

In order to simplify a switching function $f$, i.e., to find an optimal cover of $f$, in any case, we must determine prime implicants of $f$. A sufficient number of prime implicants forms a cover and in general there are very many covers. In a given cover we can search for terms which can possibly be left out to limit the cost, and if there are such terms, we must determine the best combination of them. The procedure is far from trivial and in the next section a method will be developed.

**IV.4. The implicant vector**

An implicant of 4 variables and dimension 2 can be, e.g.,

$$\bar{x}_2x_2$$

(4.4.1)
and the matrix notation of this implicant is

\[
\begin{array}{cccc}
0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 1 & 1 \\
\end{array}
\]

(4.4.2)

The weight vector of this implicant is

\[0112; 0022\]

It is also possible to take the negation of columns; then, it appears that there are still 2 other weight vectors, viz.

\[1223; 0224\]
\[2334; 2244\]

By permutation and negation of the columns of (4.4.2), an arbitrary implicant of 4 variables and dimension 2 can be formed. The definition of weight vector gives rise to only 3 weight vectors. Consideration of the sum row that follows from the matrix, delivers immediately the form of the implicant. The sum row of (4.4.2) is (0022). Permutation of the first and last columns of the matrix (4.4.2) gives the matrix (4.4.3) which represents the implicant \(\bar{x}_2x_0\):

\[
\begin{array}{cccc}
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 \\
\end{array}
\]

(4.4.3)

In this case the sum row of this implicant is (2020).

If we write \(x_3x_2x_1x_0\) above the digits, only those \(x_i\) in the implicant for which the corresponding elements of the sum row are 0 or 4 will appear. A 4 represents the variable, a 0 the negation of the variable. So (2024) represents the implicant \(\bar{x}_2x_0\), (2420) the implicant \(x_2x_0\).

It is possible without computation of the corresponding matrices to construct all the weight vectors of implicants of \(n\) variables and dimension \(j\). For convenience, we start with the case of dimension 0.

A weight vector of an implicant of \(n\) variables and dimension 0 belongs to a matrix with only 1 row. The column vector therefore consists of only 1 element. If there are \(k\) elements in the row of the matrix equal to 1, and, therefore, \(n-k\) elements equal to 0, the column vector will be simply the number \(k\). The list of possible weight vectors of implicants of dimension 0 (after rearrangement) reads:
A weight vector of an implicant of \( n \) variables and dimension 1 belongs to a matrix with 2 rows, which differ in one and only one column. The corresponding element in the row vector of this column is equal to 1, while the corresponding elements in the row vector of the remaining columns are 0 or 2, depending on the (constant) elements in the columns of the matrix, which are either 0 or 1. After rearrangement of the row vector, we find a number of 0’s, then one element 1 and finally a number of 2’s. The elements of the column vector obviously differ by 1 and their sum equals the sum of the elements of the column vector. The list of weight vectors of implicants of dimension 1 reads:

\[
\begin{align*}
0; & \quad 00 \ldots 0 \\
1; & \quad 00 \ldots 01 \\
2; & \quad 00 \ldots 011 \\
\vdots & \quad \ldots \\
n; & \quad 11 \ldots 111
\end{align*}
\]

An implicant with dimension \( j \) has a matrix with \( 2^j \) rows and \( n \) columns. In \( c \) columns all the elements are 1, in \( n-c-j \) columns all the elements are 0, whereas in the \( j \) remaining columns successively all possible \( 2^j \) combinations of \( j \) zero’s and ones occur in the different rows. The row vector, therefore, has \( c \) elements equal to \( 2^j \), \( j \) elements equal to \( 2^{j-1} \), whereas the remaining elements are equal to 0.

The elements of the column vector are equal to \( c + \) the number of ones in the variable columns. Consequently there is 1 element \( c \), \( j \) elements \( c+1 \), \( \binom{c}{2} \) elements \( c+2 \) and in general \( \binom{c}{i} \) elements \( c+i \). The list of weight vectors of implicants of \( n \) variables and dimension \( j \) are, if \( a_b \) denotes a row of \( b \) elements \( a \):

\[
\begin{array}{cccccc}
0 & 1\binom{c}{1} & 2\binom{c}{2} & \ldots & j; & 0_{n-j} & 2^{j-1}_j \\
1 & 2\binom{c}{1} & 3\binom{c}{2} & \ldots & j+1; & 0_{n-j-1} & 2^{j-1}_j & 2^1_j \\
\vdots & \vdots & \vdots & \vdots & \vdots; & \vdots \vdots & \vdots \\
c & c+1\binom{c}{1} & c+2\binom{c}{2} & \ldots & c+j; & 0_{n-j-c} & 2^{j-1}_j & 2^c_j \\
\vdots & \vdots & \vdots & \vdots & \vdots; & \vdots \vdots & \vdots \\
n-j & n-j+1\binom{c}{1} & n-j+2\binom{c}{2} & \ldots & n; & 2^{j-1}_j & 2^{j}_{n-j} \\
\end{array}
\]
We can now simplify the notation of the weight vector of an implicant. For the computation of the weight vector of an implicant of \( n \) variables and dimension \( j \), at first the row vector is determined. This row vector, however, completely determines the column vector. Therefore it is not necessary to compute the entire column vector. If we remember that the sum of the digits of the column vector is equal to the sum of the digits of the row vector, the column vector can be replaced by this sum. In this way we get an abbreviated notation for the weight vector of an implicant of \( n \) variables and dimension \( j \). The weight vector

\[
0 \begin{pmatrix} 1 \end{pmatrix} 2 \begin{pmatrix} 1 \end{pmatrix} \ldots j; \quad 0_{n-j} 2^{j-1}
\]

can be replaced by

\[
2^{j-1}; \quad 0_{n-j} 2^{j-1}.
\]

The weight vector

\[
1 \begin{pmatrix} 2 \end{pmatrix} 2 \begin{pmatrix} 2 \end{pmatrix} \ldots j+1; \quad 0_{n-j-1} 2^{j-1} 2^j
\]

can be replaced by

\[
j. \quad 2^{j-1} + 1.2^j; \quad 0_{n-j-1} 2^{j-1} 2^j.
\]

This modified weight vector will be called an implicant vector. To prevent confusion, the semi-colon in the implicant vector will be replaced by a \( / \).

The number before the \( / \) is the sum of the digits of all elements of the matrix which represents the implicant. This sum will be called the matrix sum. A complete list of the implicant vectors up to dimension 6 is given in table VI. Consideration of the unarranged row vector that follows from the matrix yields immediately the form of the implicant. If we write \( x_{n-1}x_{n-2} \ldots x_1x_0 \) above the digits of the sum row, only those \( x_t \) in the implicant for which the corresponding elements of the sum row are 0 or \( 2^t \) will appear in the implicant. A \( 2^t \) represents the variable, a 0 the negation of the variable.

IV.5. Simplification of the switching function

It is possible with the aid of the implicant vectors to simplify a switching function, and the process consists of two parts. In the first part we determine all the prime implicants of the switching function; in the second part with the prime implicants we search for an optimal cover.

The determination of the prime implicants may proceed as follows. We write the switching function as a matrix \( M \). If we work with pencil and paper, it is advantageous to order the rows according to nondescending weights. If we imagine to the left of the matrix \( M \) the sum column, a matrix \( M_0 \) arises, which is a list of implicant vectors of dimension 0. We write \( M_0 \) at square angles above and to the right of \( M \)\(^*\). A row of \( M_0 \) forms the heading of a column of a connection matrix \( N_0 \), while a row of \( M \) is forming the heading of a row of \( N_0 \).

\(^*\) See the table, 3 pages further on.
On the main diagonal the elements of $N_0$ are equal to 1, the other elements equal to 0. We shall write the 1's only. From the matrix $M_0$ we derive a matrix $M_1$, viz. a list of implicant vectors of dimension 1. For that purpose the sum of every two rows of $M_0$ must be determined for which:

(i) the leading elements (matrix sums) differ by one,
(ii) the remaining vectors of $n$ elements differ in one element only. If we find such an implicant vector, we add this vector to the matrix $M_1$, which is written next to the matrix $M_0$. A row of $M_1$ now forms the heading of a column of a connection matrix $N_1$ which is built up as follows. If the implicant vector on the heading of the column $\gamma$ of $N_1$ was formed by the two implicant vectors of $M_0$ which are heading the columns $a$ and $\beta$ of $N_0$, then the column $\gamma$ is the logical sum of the columns $a$ and $\beta$.

Every implicant vector of $M_0$ that can be combined with another implicant vector of $M_0$ is marked by a 1, written underneath the connection matrix $N_0$ in the corresponding column. If there is an implicant vector which cannot be combined a 0 is written under the connection matrix $N_0$ in the corresponding column. In this way a row $u$ arises, which marks the prime implicants.

Now we compute the sum column $S$ of $N_1$. The $k$th element of $S$ is said to correspond to those implicant vectors which are heading the columns containing a 1 in the $k$th row. A 0 element of $S$ means that the corresponding implicant vector of $M_0$ (a point!) cannot be combined with another implicant vector of $M_0$, i.e., it is a prime implicant.

In this case they even are essential prime implicants, i.e., prime implicants which are necessary elements of the cover. In the columns of those prime implicants we write down 0 in a row $v$ under the matrix $N_0$, and in the other columns a 1 is written. The row $v$ is meant to mark the essential prime implicants. In the row $v$ the only marks of interest are those columns in which the $u$ row contains a 0. Thus so far the $u$ row and $v$ row are identical.

If there are elements 1 in $S$, then the corresponding point in $M_0$ can be combined and must be combined with another point to a line (= implicant vector of $M_1$). This line is a prime implicant, since at least one of its points can no more be combined with a third point. Moreover it is an essential prime implicant, since it is the only prime implicant that contains the said point. Hence, on the basis of the elements of $S$ equal to 1, the essential prime implicants of $M_1$ are recognized and marked with a 0 in the $u$ and $v$ rows.

In general, from the matrix $M_j$ a matrix $M_{j+1}$ is formed, viz. a list of implicants of dimension $j+1$. For this purpose the sum of each two rows of $M_j$ must be determined for which:

(i) the leading elements (matrix sums) differ by $2^j$,
(ii) the remaining vectors of $n$ elements differ in one element only.

If we find such an implicant vector, we shall add it to the matrix $M_{j+1}$.
which is written next to the matrix $M_j$. Care must be taken, however, that each element of $M_{j+1}$ will be found $j+1$ times.

A row of $M_{j+1}$ forms the heading of a column of the connection matrix $N_{j+1}$ obtained in the following way. If the implicant vector on the heading of column $\gamma$ of $N_{j+1}$ is formed by the implicant vectors of $M_j$ which are heading columns $\alpha$ and $\beta$ of $N_j$, then the column $\gamma$ is the logical sum of the columns $\alpha$ and $\beta$.

As soon as $M_{j+1}$ is formed the following information can be drawn regarding prime implicants. First of all, implicants of $M_j$ which have not contributed to $M_{j+1}$ are prime implicants. This is easily recognized if one has marked with a 1 in the $u$ row those implicants which have been combined to form an implicant in $M_{j+1}$. If the $u$ row originally contained all zeros the prime implicants in $M_j$ would be automatically marked by a zero in the $u$ row.

Moreover, if a column of $M_{j+1}$ contains a 1 in a row in which the sum column $S$ of $N_1$ contains a $j+1$, then that implicant of $M_{j+1}$ is an essential prime implicant which is marked by a zero in the $v$ row which originally contained only 1's.

The process ends automatically if a matrix $M_j$ is empty. Then the prime implicants are those implicants which correspond to a 0 in the $u$ row and to the essential ones that correspond to a 0 in the $v$ row.

The process described above will be illustrated by an example in 5 variables. Let the matrix $M$ of the switching function be:

$$
\begin{align*}
00000 \\
00001 \\
00011 \\
00101 \\
00111 \\
01000 \\
01100 \\
10000 \\
10011 \\
10100 \\
10101 \\
10111 \\
11000 \\
11100
\end{align*}
$$

We get the following scheme and list of prime implicants (see next page).

Now we must carry out the second part of the process, i.e., to search for an optimal cover with these prime implicants. At first in $M$ the rows $r_k$ forming the essential prime implicants are marked. These rows are indicated by the 1's
<table>
<thead>
<tr>
<th>$M_0$</th>
<th>$M_1$</th>
<th>$M_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>00001</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>01000</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>00011</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>00101</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>01100</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>10000</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>00111</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>10011</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>10100</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>11000</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>11100</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>10111</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>$N_0$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$N_1$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$N_2$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Essential prime implicants**

<table>
<thead>
<tr>
<th>Dim. 0</th>
<th>Dim. 0</th>
<th>Dim. 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1/00001 = x_4x_3x_2x_1$</td>
<td>$8/00224 = x_4x_3x_0$</td>
<td>$4/22000 = \bar{x}_2\bar{x}_1\bar{x}_0$</td>
</tr>
<tr>
<td>$8/24200 = x_3\bar{x}_1\bar{x}_0$</td>
<td>$5/20201 = x_4x_3x_2x_1$</td>
<td>$8/42200 = x_4\bar{x}_1x_0$</td>
</tr>
<tr>
<td>$12/20244 = \bar{x}_3x_2x_0$</td>
<td></td>
<td>$12/20424 = \bar{x}_3x_2x_0$</td>
</tr>
</tbody>
</table>

**Non essential prime implicants**
in the columns of the connection matrices, which columns belong to the corresponding essential prime implicants. If it turns out that all rows of $M$ are marked then the logical sum of all essential prime implicants will be equal to the optimal cover.

If there are still unmarked rows in $M$, which geometrically means that there are still uncovered points, then we must cover the points with the non-essential prime implicants. At first we examine whether there are non-essential prime implicants that cover points which are already covered by the essential ones. We can strike out these redundant prime implicants. With the remaining prime implicants we must determine the optimal cover. Let there be $N$ of these non-essential prime implicants. On every non-essential prime implicant a number $j$ is designated.

A logical sum of a set of non-essential prime implicants can be represented by a number $q$,

$$q = \sum_{j=0}^{N-1} p_j 2^j,$$

where $p_j = 0$ or $1$, viz. 0 if the $j$th prime implicant does not occur and 1 if the $j$th prime implicant does occur in the sum.

Certainly the number

$$Q = \sum_{j=0}^{N-1} 2^j$$

represents a cover. Whereas we must take into account that in the optimal cover even few $p_j = 0$, we start with $Q$ and we search for an optimal cover by introducing zeros systematically. Let

$$q_{12 \ldots k} = \sum_{j=0}^{N-1} p_j 2^j \quad j = j_1, j_2, \ldots, j_k.$$

By leaving out terms $p_j 2^j$, there arises from $Q$ a number $q_{j_1}$. At every $j_1$ $q_{j_1}$ represents a cover, for otherwise $p_{j_1}$ must be an essential prime implicant. All $q_{j_1}$ are taken up in a list of covers of order 1. Next all $q_{j_1j_2}$ are determined. This happens by collation of $q_{j_1}$ and $q_{j_2}$. Since this operation is symmetric, it is sufficient to collate every $q_{j_1}$ with a $q_{j_2}$ for which $j_2 \succ j_1$. It does not follow as a matter of course that $q_{j_1j_2}$ is a cover and we must test whether $q_{j_1j_2}$ perhaps is a cover. Only then is $q_{j_1j_2}$ taken up into the list of covers of order 2. At the same time $q_{j_1}$ and $q_{j_2}$ in the list of covers of order 1 are labelled by an asterisk, which indicates that $q_{j_1}$ and $q_{j_2}$ never forms an optimal cover, for $q_{j_1j_2}$ is a cover with a lower cost. Let us suppose that the list of order 2 is classified in such a way that the row of numbers $q_{j_1j_2}$ with the same $j_2$ are written one after the other in order of ascending $j_2$, and that they are recognizable as such.

Next we must determine the possible covers $q_{j_1j_2j_3}$ with $j_1 \prec j_2 \prec j_3$. This happens by collation of $q_{j_1j_2}$ with $q_{j_1j_3}$, $j_3 \prec j_2$, i.e. by collating 2 numbers in the aforementioned sublist of order 2. We test whether such $q_{j_1j_2j_3}$ forms a cover.
and if that be the case, it is written in the list of order 3 and \( q_{112} \) and \( q_{113} \) in the list of order 2 are labelled by an asterisk. We continue until the list of order \( k \) is empty.

The candidates for an optimal cover are those numbers in all lists of covers of different orders that are not labelled with an asterisk. We now compute the cost of all these covers in order to find the optimal cover(s).

As an example we shall search for the optimal cover belonging to the switching function (4.5.1). In that case there are 2 essential and 6 non-essential prime implicants. We can easily show that under the 6 non-essential prime implicants there are no redundant prime implicants.

The non-essential prime implicants are:

\[
\begin{align*}
    j = 0 & \quad .01.1 \\
    1 & \quad 1.00 \\
    2 & \quad 00.1 \\
    3 & \quad .000 \\
    4 & \quad 1010 \\
    5 & \quad 0000 .
\end{align*}
\]

and the points that must be covered are:

00000
00001
10000
00101
10100
10101

The lists are then as follows:

\[
\begin{align*}
    111110 & * \\
    111100 & * \quad 011100 (13) \\
    110110 (18) \\
    011110 & * \\
    111101 & * \quad 110011 (18) \\
    011101 (17) \\
    111011 & * \quad 100011 (13) \\
    101011 & * \\
    110111 & * \quad 100111 (17) \\
    101111 & * \quad 001111 (16) \\
    011111 & *
\end{align*}
\]

Behind the numbers without an * the cost is stated in between parentheses. The cost of that part of the cover which is covered by the non-essential prime implicants is 13, and the cost of that part of the cover which is covered by the essential prime implicants is 8. The total cost is 21 and the optimal covers are:
It is, however, still possible that there is a diode circuit that has a cost which is less than the cost which we find by the procedure just described. This procedure leads to the minimal sum of products. The other possibility is the minimal product of sums which can easily be derived from the minimal sum of products.

Let the switching function $f$ be given in the conjunctive normal form. Calculate the conjunctive normal form of $\overline{f}$ and apply the procedure to find the minimal sum of products for $\overline{f}$. If we now take the negation of $\overline{f}$, we shall find the minimal product of sums. Because this minimal product of sums is derived by negation only from the minimal sum of products of $\overline{f}$, the cost will be the same as the cost for the cover of $\overline{f}$. This may be less than the cost of the minimal sum of products.

If we apply the procedure to the matrix (4.5.1), we find an expression for $\overline{f}$ that reads

$$\overline{f} = \overline{x_1}x_0 + x_2x_0 + \overline{x_3}x_2x_0 + x_4\overline{x_3}x_1x_0.$$  \hspace{1cm} (4.5.4)

This function can be realized with 16 diodes. By taking the negation of $\overline{f}$ we find the minimal product of sums, which of course also needs 16 diodes. Because the minimal sum of products needs 21 diodes, the minimal product of sums is less expensive. The most simple notation of (4.5.1) is, therefore,

$$f = (x_1 + x_0)(\overline{x_3} + x_0)(x_4 + x_3 + \overline{x_2} + x_0)(\overline{x_4}x_2x_1x_0).$$  \hspace{1cm} (4.5.5)

For 3 and 4 variables we have determined the minimal sum of products only (table III). In table II the conjunctive normal form for 3 and 4 variables is given. As is mentioned, the number $s$ corresponds to the conjuncted normal form of $f$ and $\overline{s}$ corresponds to the conjuncted normal form of $\overline{f}$. The number $s$ in table II refers to the number $s$ in table III, i.e., the minimal sum of products of $f$.

It is possible that the cost of $\overline{s}$ is less than the cost of $s$, and then we must take the negation of $\overline{s}$ to find the simplest notation of $f$. In table III the cost $c$ of the covers is given in the second column. In this table $x_0$ is replaced by $x$, $x_1$ by $y$, $x_2$ by $z$, and $x_3$ by $u$.

IV.6. Discussion of the results

Ledley 8) as well as McCluskey 11) developed a method by which it is possible to determine the prime implicants of a switching function. They start by the determination of the optimal cover with the essential prime implicants,
and go on (if necessary) with the non-essential prime implicants with the highest dimension \( j \), say. If all prime implicants of dimension \( j \) cover all points of the \( n \)-dimensional figure, which represents the switching function, then we must examine whether there are redundant prime implicants. If all points have not yet been covered, then we try the prime implicants of dimension \( j-1 \), and so on.

This method gives rise to difficulties for switching functions of 5 or more variables. An example is the switching function (4.5.1). We find 6 prime implicants of dimension 2. These 6 prime implicants completely cover the 5-dimensional figure representing the switching function. There are even no redundant prime implicants, thus the minimal sum of products can be realized by 24 diodes. In section IV.5 we have found that 4 prime implicants of dimension 2 and 1 prime implicant of dimension 1 also completely cover the switching function. This last cover can be realized with 21 diodes and is, therefore, less expensive.

Figure 11 represents a 2-dimensional projection of a 5-dimensional unit cube.

![Figure 11. The geometrical representation of the switching function (4.5.1).](image-url)
The coordinates of the vertices are defined by

\[ j = \sum_{i=0}^{4} x_i 2^i. \]  

(4.6.1)

The switching function (4.5.1) is represented in this unit cube by black dots. This drawing shows that there are 6 prime implicants of dimension 2 (planes) which cover the black dots completely. All black dots are also covered by 4 planes and 1 line. The 4 planes and the line are represented by the thick lines. The 2 planes that can be eliminated are pointed out by the dotted lines. The figure represents the notation (4.5.3). The other notation (4.5.2) is represented by the planes that are determined by the points

\[
\begin{array}{cccc}
8 & 12 & 24 & 28 \\
0 & 8 & 16 & 24 \\
3 & 7 & 19 & 23 \\
1 & 3 & 5 & 7 \\
\end{array}
\]

and a line is now determined by the points

\[
20 \quad 21
\]

IV.7. The "don't care" circuits

Up to now we have started from the principle that all components \( f_m \) of the switching function are prescribed. In practice, however, it can happen that the value of some components is not of importance. By a suitable use of these "don't care" terms, it may perhaps be possible to reduce the number of diodes that realize the switching function. The method described in section IV.5 can also be used for the "don't care" circuits.

The process is as follows. The switching function \( f^k(j) \) which is defined by the components \( f_m = 1 \) that must be 1, is written in the matrix notation. The switching function \( f^l(j) \) which is defined by the components \( f_m = 1 \) that may be 1, is also written in the matrix notation.

The matrices are written one under the other and considered as one matrix. We now apply the simplification process. The cover \( \sigma \) is now defined as follows. A cover \( \sigma \) is a logical sum of implicants of \( f^k(j) \) and \( f^l(j) \) which are implied by \( f^k(j) \), such that from \( f^k(j) = 1 \) follows \( \sigma = 1 \). We can determine the cost of the cover, thus a cover with minimal cost is an optimal cover.

Again we distinguish between essential and non-essential prime implicants. It is, however, clear that the essential prime implicants and the non-essential prime implicants of \( f^l(j) \) can never be a part of \( \sigma \). Those prime implicants can be omitted.

By an example we shall illustrate the process. Let the switching function \( f^k(j) \) be

\[ f^k(j) = \bar{x}_3 \bar{x}_2 \bar{x}_1 x_0 + \bar{x}_2 x_2 x_1 \bar{x}_0 + \bar{x}_3 \bar{x}_2 x_1 x_0 + x_3 x_2 \bar{x}_1 \bar{x}_0, \]  

(4.7.1)
and let the switching function \( f^t(j) \) be
\[
 f^t(j) = \bar{x}_3\bar{x}_2\bar{x}_1x_0 + x_3x_2\bar{x}_1x_0 + x_3x_2x_1x_0.
\] (4.7.2)

The matrices of \( f^k(j) \) and \( f^t(j) \) are, respectively,

\[
\begin{array}{cccc}
0001 & 0000 \\
0010 & 1101 \\
0011 & 1111 \\
1100 & \\
\end{array}
\] (4.7.3)

The matrix \( M \) arises by combining the two matrices. To this matrix we apply the simplification procedure, so that

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>u</td>
<td></td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0010</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0011</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1100</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1101</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1111</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

There are 3 prime implicants, i.e.,
\[
\begin{align*}
5/2201 &= x_3x_2\bar{x}_1 \\
7/2212 &= x_3x_2x_0 \\
4/0022 &= \bar{x}_3\bar{x}_2.
\end{align*}
\]

The prime implicant \( x_3x_2x_0 \) cannot be a part of the cover, for this is a prime implicant of \( f^t(j) \). The two remaining prime implicants are essential prime implicants; thus the optimal cover reads:
\[
f = \bar{x}_3\bar{x}_2 + x_3x_2\bar{x}_1.
\] (4.7.4)

The cost is 7 diodes. If we had made no use of the “don’t care” terms, and had applied the simplification on (4.7.1) only, then the result would have been:
\[
f = \bar{x}_3\bar{x}_2x_0 + \bar{x}_3\bar{x}_2x_1 + x_3x_2\bar{x}_1\bar{x}_0
\] (4.7.5)

with a cost of 13 diodes.
APPENDIX

Mechanization of the minimization process

Electronic computing machines will carry out large numbers of operations without intervenience of the operator. A list of operations that is to be executed by the machine is called a programme. In practice it appears that these programmes are very complicated, and the unravelling is like looking for a needle in a hay-stack. It is, therefore, convenient to divide up a programme into a number of blocks. In every block the procedure that the machine must carry out is indicated in short. We call such a simplified programme a flow diagram.

We shall make the assumption that we shall enter a block at the top and leave it at the bottom. If a "yes" or "no" decision is asked for in a block, then we shall leave the block on the right side in case of an affirmation, on the left side in the other case. If a connecting line between 2 blocks would be very long, then we end it in a circle in which a number is written and which we term connector. The number refers to a connector with the same number, which is put before the block that now must be executed.

An advantage of such flow diagrams is that we need not be familiar with the code in which the machine is working. The same flow diagram can, therefore, be used for different kinds of machines.

The process described in IV.5 can be mechanized very well. For understanding the flow diagrams the following remarks are important.

The rows of the matrix $M$ are ordered according to nonascending weights. During the input programme the number of rows of $M$ is counted; call it $K$.

At the same time that a memory place $k$ is filled with a row $r_{kt}$, a corresponding memory place $a^0_k$ is filled with $n+1$.

The flow diagrams now read:

\begin{equation}
\begin{array}{c}
\text{Determination of } M_0. \\
\end{array}
\end{equation}
Determination of $M_{j+1}$ from $M_j$, $\sigma_k$ is the cost of the prime implicants.
Determination of the \( v \)-row; \( w \) is a working space.

Determination of the essential and non-essential prime implicants.

Determination of the cover with the essential prime implicants.
Determination of the redundant non-essential prime implicants.

\[ s_j = \text{sum of the costs of the prime implicants} \]
\[ t_l \text{ is a digit that marks those covers which never can form an optimal cover} \]
\[ x_l \text{ is a symbol that indicates the end of a particular list} \]
Determination of the optimal covers.

List of covers of order \( j \).
REFERENCES


(To be continued)