SYNTHESIS OF SEQUENTIAL MACHINES
A MATHEMATICAL APPROACH
JORGE AGUILÓ, ELENA VALDERRAMA, CARLOS SIMÓ

A new mathematical approach for encoding internal states in synchronous sequential machines is developed.

The relationship between transition matrices and the corresponding matrices of the isometric group is studied. An algorithm to determine a priori the null cost assignment is proposed. The cycle lengths which appear for each \( n \) are analysed. The results obtained for the \( D \) - or \( SR \) flip-flop can be extended to \( JK \) and \( CI \).

1. INTRODUCTION.

A mathematical approach for the synthesis of sequential machines is developed. In the synthesis, the state assignment problem remains still without solution. This topic is treated in this paper.

A great number of algorithms have been tested dealing with it from different points of view. Methods based on the search of SP-partitions (1),(2) from the states set, or in the search of non-disjoint covers, represent attempts to solve this problem, presenting intermediate results close to ours. Nevertheless, for every sequential machine, they have to look for a convenient set of - SPPs which do not let them an "a priori" definition over the existence of a null cost assignment. In addition, the general mathematical handling goes away from real conditions. That causes that, some particular case excepted, nor the structures that can be realized with a given number of memory elements, neither the number of flip flops you need to synthesize a given autonomous automaton have not been reach.

In other way, with SHR's method /3/, we have to test \( (2^n-1)!(2^n-R)!m! \) assignment schemes because of a previous condition about the existence of solutions does not exist. In some cases, when \( \text{SAN}_j > MNS_j \) \( \forall_j \), SHR's method is better in the sense that it gets acceptable results in a relatively easy way.

Combinational techniques /4/ are very powerful, full ones, but for the moment, have not reach relevant solutions to the problem, and their computing complexity is undoubtedly high.

In this paper, a first step towards a mathematical formulation in which there is no uncertainty in the final solution is given. The starting point is a geometrical representation of the Boolean algebra from a dinamic point of view. In spite of that the optimum cost of an automaton depends of the optimization criterium chose, independently of such criterium, the elements belonging to the same isomorphical class defined in the set of the automata behaviour graphes have all of them the same optimum cost. The study of this optimum is done by identifying the automata with the set of transformations of the Boolean hypercube. We get a first result dealing with the gateless synchronous counters synthesized with D or SR flip flops (ffs).

- J. Aguiló, E. Valderrama, C. Simó - Departament d'Informàtica de la Universitat Autònoma de Barcelona.
- Bellaterra (Barcelona)
- Article rebut al Gener de 1982.

Quèstió - V. 5, nº 4 (d'agost 1981)
An algorithm that finds when the synthesis -- with n memory elements \((2^n - 1 < R \leq 2^n)\); number of states) and without gates is possible is given. In particular cases, analytical formulas valid from a certain value of n are developed, and, in general, an upper bound of the number of memory elements needed in the synthesis without gates is found.

1. ANALYSIS

The transition table of a counter (coded states) can be seen as a one to one mapping from \(B^n\) into \(B^n\); where \(B^n = \{1,-1\}^n\) is a n-dimensional cube inside \(R^n\).

With this representation, the usual method of synthesis with D ffs. is reduced to an interpretation of the cube movement.

For example, let's take the following state table:

<table>
<thead>
<tr>
<th>Present state (P.S.)</th>
<th>Next state (N.S.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1       -1 -1</td>
<td>1      -1 -1</td>
</tr>
<tr>
<td>-1       -1 1</td>
<td>-1      -1 1</td>
</tr>
<tr>
<td>-1       1 -1</td>
<td>-1      -1 -1</td>
</tr>
<tr>
<td>-1       1 1</td>
<td>1       -1 1</td>
</tr>
<tr>
<td>1       -1 -1</td>
<td>1       1 -1</td>
</tr>
<tr>
<td>1       -1 1</td>
<td>1       1 1</td>
</tr>
<tr>
<td>1       1 -1</td>
<td>-1      1 -1</td>
</tr>
<tr>
<td>1       1 1</td>
<td>-1      1 1</td>
</tr>
</tbody>
</table>

This state table can be seen as a 3-cube movement (Fig. 1)

\[ x_1 \text{ and } D_3 = \bar{x}_1 x_3 + x_1 \bar{x}_3. \]

We infer that a given counter will be able to be synthesised without gates if and only if that mapping changes (n-1)-cubes into --- (n-1)-cubes (5)(6).

2. ISOMETRIC TRANSFORMATIONS ON A CUBE

We shall call \(\Delta_n\) the set of all \(B^n\)-isometric transformations. Being clear that \(\Delta_n\) is a subgroup of the symmetric group \(S_{2^n}\) and \(\sigma \in \Delta_n\) if and only if it changes r-subcubes into r-subcubes. Specifically, it changes (n-1)-cubes into (n-1)-cubes. Then, it represents a counter which can be synthesised without gates.

If \(d\) is the Euclidean distance restricted to an n-cube:

\[ d_{(2\sqrt{k})} = \binom{k}{j} \]

where,

\[ d_{x}: (x) \times B^n \rightarrow R^n \]

\[ (x, y) \rightarrow d_{x}(x, y) = d(x, y) \]

In such a way \(C \subset B^n\) is a k-subcube of \(B^n\) if and only if

\[ d_{x}(C \times C) = \{0, 2\sqrt{1}, ..., 2\sqrt{k}\} \]

and

\[ d_{(2\sqrt{j})} = \binom{k}{j}; \forall x \in C, j = 0, 1, ..., k \]

The face \(x_1 = 1\) is transformed into the two edges \((x_2 = 1, x_3 = 1)\) and \((x_2 = 0, x_3 = 0)\).

Then, the input to the first D ff. has to be

\[ D_1 = x_2 x_3 + x_2 \bar{x}_3, \]

and, in the same way, \(D_2 = \)

and we infer that \(\Delta_n\) is a group isomorphic to the group of all the linear mapping \(\sigma_s\) such that,

\[ \sigma_s(e_1) = + e_s(i), \text{ with } s \in S_{2^n} \]
As a consequence, any element \( \sigma \in \Lambda_0 \cap S_n \) can be represented by a \( nxn \) square matrix of 0,1 and -1, with a sole non null element by row and column.

For example, let's consider the following \( n \)-cube isometry \( \sigma \) (see Fig. 2)

\[
\sigma(\mathbf{e}_1) = \mathbf{e}_2; \quad \sigma(\mathbf{e}_2) = -\mathbf{e}_1; \quad \sigma(\mathbf{e}_3) = \mathbf{e}_3
\]

where \( \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\} \) is the canonical base. --

This isometry can be biunivocally represented by the matrix

\[
\sigma = \begin{pmatrix}
0 & 1 & 0 \\
-1 & 0 & 0 \\
0 & 0 & 1
\end{pmatrix}
\]

Accordingly with the above considerations, the counter whose state table is represented by the 3-cube movement (see Fig. 3) will be able to be synthesised with 3 D-flfs. and without gates. In addition, the portion of the non zero elements in the matrix gives us the input function for every ff. That is

\[ D_1 = B, \quad D_2 = \bar{A}, \quad D_3 = C \]

or, (see Fig. 4).

\[
\begin{array}{c|ccc}
\text{P.S.} & -1 & 1 & -1 \\
-1 & -1 & -1 & 1 \\
1 & 1 & -1 & 1 \\
1 & 1 & 1 & 1 \\
1 & -1 & -1 & -1 \\
-1 & 1 & 1 & -1 \\
-1 & -1 & 1 & 1 \\
-1 & 1 & 1 & 1 \\
\end{array}
\]

\[ S_1 = \bar{R}_1 = B, \quad S_2 = \bar{R}_2 = A, \quad S_3 = \bar{R}_3 = C \]

3. SYNTHESIS

We will call two autonomous automata \( m \) and \( m' \) equivalents if and only if their behaviour graphs are isomorphes.

In other words, if we call \( M \) and \( M' \) to the matrices associated to behaviour graphs of \( m \) and \( m' \), \((M,M' \in S_n^3)\),

\[ m \cong m' \quad \Leftrightarrow \quad U.M.U^{-1} = M' \]

Matrix \( U^{-1} \) gives the assignment done over the states of \( m \). In other words, \( m \) and \( m' \) are equivalents when we can obtain one of them from the other through an assignment.

The conjugation classes defined by the last relation represent sets of automata having the same optimum, independently of the optimisation criterium chosen.

Given a state table (no coded states), we will see that there exist a state assignment which allows us to synthesise the counter without gates if and only if its conjugation class has at least one element of \( \Lambda_0(5)(6)(7) \).
There exist a one to one correspondence between the matrices associated to $\sigma \in S_{2^n}$ and the transition matrices of the graph corresponding to a complete autonomous automation. The previous condition can be expressed in the following way: If $M_{\sigma}$ is the transition matrix associated to the autonomous automaton $M_{\sigma}$, $\sigma \in S_{2^n}$ and $M_{\sigma}$ is the transition matrix corresponding to an isometry $\sigma \in \Lambda_n$, the automaton $M_{\sigma}$ can be synthesised without gates through an appropriate assignment if and only if there exist $U \in S_{2^n}$ and $\sigma$ such that:

$$U M_{\sigma} U^{-1} = M_{\sigma}$$

As an example, let the automaton $M$ be:

<table>
<thead>
<tr>
<th>P.S.</th>
<th>N.S.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>$q_3$</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$q_6$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$q_1$</td>
</tr>
<tr>
<td>$q_3$</td>
<td>$q_4$</td>
</tr>
<tr>
<td>$q_4$</td>
<td>$q_7$</td>
</tr>
<tr>
<td>$q_5$</td>
<td>$q_2$</td>
</tr>
<tr>
<td>$q_6$</td>
<td>$q_5$</td>
</tr>
<tr>
<td>$q_7$</td>
<td>$q_0$</td>
</tr>
</tbody>
</table>

![Diagram](image)

Then,

$$M_{\sigma} = \begin{pmatrix}
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

and let's take the matrix $U$, and $M_{\sigma}$ corresponds to the isometry $\sigma$ of the last example, and, in consequence, the automaton $M_{\sigma}$ can be synthesised without gates as Fig. 4 shows. The optimum assignment is determined by $U^{-1}$:

Uncoded state 0 1 2 3 4 5 6 7
Coded state 0 1 5 2 6 7 3 4

Each state assignment which allows the synthesis of the automaton with 0-cost takes us the state table determined by $\sigma$. Generally, for a given $M_{\sigma}$, $M_{\sigma}$ will not be unique.

Let us note, in addition, that being $m$ and $m'$ autonomous automata, their behaviour graphs can be easily represented by their cycle decompositions. Then, $m$ and $m'$ will be equivalent machines if and only if they have the same cycle lengths and, in consequence, our aim will be to obtain the cycle decomposition of the n-cube isometries. To do so, and if we define the above equivalence relation in $\Lambda_n$, we only have to test the cycle lengths of the representative isometries.

4. PROCEDURE

From the considerations above mentioned, the conjugation classes containing at least one isometry can be obtained; and then the optimum assignment for a given state table can be reached from the representative isometry.
The algorithm to obtain these representative isometries has been programmed in a UNIVAC - 1108 computer. The time required for the classification is of the order of

\[ t(n) = 10^{-2} \cdot 1.5^n \text{ n! msec.} \]

and the memory used is of the order of \(1.8^n\) K-words.

5. RESULTS

The table in the next page gives the lengths of the cycles that can be synthesised with a given number of memory elements and without gates.

To make the table shorter, the lengths for every \(n\) which are twice the ones of \(n-1\) are omitted.

A set of theoretical results has been obtained which in some sense, are interesting by themselves, and, in the other way, represent hints that make easier the searching.

It has been proved that:

- For a given \(n\), the length of the longest cycle is a multiple of the lengths of the remaining cycles.

- The lowest common multiple of the lengths which appear for each \(n\) is such that,

\[
\text{l.c.m.}(n) = 2 \cdot \Pi \log_p n \quad \text{p prime} \leq n
\]

in such a way that for a given \(n\), we find -- all lengths of type \(\text{l.c.m.}(1_k, \ 1_{n-k})\) where \(1_k\) is any length obtained in a previous case.

This result includes other author's one (1) (8) that determines that the minimum \(n\) in which appears a cycle of length \(p\),

\[
p = 2 \cdot p_2 \cdots \cdot p_m
\]

is

\[
N = \sum_{i=1}^{m} r_i - \left\lfloor \frac{r_1}{2} \right\rfloor
\]

From this two results we can infer, for instance, that a sequential machine having one cycle of length 7 and another one of length 5, will be implemented by a number \(n\) of memory elements such that the longest length for this \(n\) will be a multiple of 35, and in accordance with the second result, this will be only possible for \(n \leq 12\).

- There exist only one class with a length -- \(2^{2n-2}\) cycle. Such a class has \(n(n-1)\) elements; that is, there are \(n(n-1)\) equally optimal assignments.

- For \(n > 4\), there exist classes \(2^3 \cdot 2^{n-3}\) and \(3^{2n-2}\) with \(12(n)\) and \(18(n)\) optimal assignments each one respectively.

- If \(n > 5\), there appear classes \(4 \cdot 2^{n-4}\), \(6 \cdot 2^{n-4}\), \(6 \cdot 3^{2n-3}\), \(2 \cdot 2^{n-4}\) and \(27 \cdot 2^{n-4}\).

We can generalize the results for other types of ffs. For the JK ff., the necessary and sufficient condition so that a counter might be able to be synthesised with only JKS is that the mapping \(B^n \rightarrow B^n\) transforms every \((n-1)\)-cube in two \((n-2)\)-cubes on the first one or on her complementary. In the case of the GL ffs., the transformation is every \((n-1)\)-cube in two \((n-2)\)-cubes, one on the first one, and the other on the \((n-1)\)-cube complementary to the cube corresponding to the input variable \(G\).

It is clear that for the JK and GL ffs., the conditions are less restrictive than for D ffs. In such a way that the former are included in the later. The number of state tables of zero cost will be greater and as a consequence, the number of null cost conjugation classes is also greater. For two memory elements and for JK ffs. we can synthesise -- without gates whatever automaton. For three memory elements there exist 712 possible -- transition tables with 0 cost, a total of 12 conjugation classes and whatever cycle -- length.
6. REFERENCES


/7/ AGUILO J., VALDEBERRA R., VILLANUEVA JJ. "On the state assignment problem" IEEE Journal on Computer and Digital techniques. (To be published).