COMPOSED AUTOMATA

Ilie Borcosi, ***"Constantin Brancusi" University of Targu Jiu, Romania***
Ion Marian Popescu, ***"Constantin Brancusi" University of Targu Jiu, Romania***
Adrian Runceanu, ***"Constantin Brancusi" University of Targu Jiu, Romania***

Abstract: *The composed automata can be studied with the help of automata theory, which deals with the study of study of abstract and automatic machines, but also computing problems can be solved with them. So, automata theory deals with the study of virtual machines which operates in understanding the logic of input and output, with or without intermediate steps of calculation. In this paper the synthesis of a composed automata is studied, which has a particular stucture, that has in composition blocks of delay.*

Keywords: *automata, composed automata, automata theory, synchronizing tree.*

INTRODUCTION

In the figure 1 is illustrates a machine which is composed of stable states (represented in the figure by circles) and transitions (represented by arrows) [1], [2]. When to the machine applied an input variable is applied (whose value is symbolized near the transition arrow), it makes a transition (or jump) into another stable condition, according to the transition function (which becomes the current state / actual and may evolve into another state depending on the variable input).

Automata theory is also closely related to formal language theory. An automaton can be considered as a finite representation of a formal language. The automatons are often classified according to the class of formal languages that are able to recognize them.

An automaton must evolve for some input sequences in discrete points of time. At each time point, an automaton has certain values for the input variables, which is a set of symbols or letters and forming the alphabet of input variables.

An automaton contains a finite set of states and each time it is in one of its states. At each step, when the machine reads an input variable, it transits in another state, which is determined by a function which the end is current state and parameter is input variable. This function is called the transition function. The automaton reads a set of input variable values (the input sequence forming) one after another and transits from one state to another by the function of transition, until the entire sequence is read. Once the input sequence has been read, the automaton is said to have stopped and the state that stopped the automaton is called final state [1]…[4].

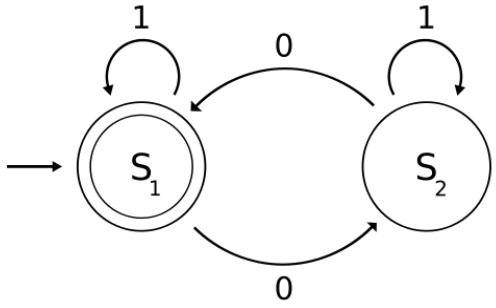

Fig.2.

PROBLEM FORMULATION

A composed automaton A is formed from A1 and A2, with the structure in figure 2, where x is the input variable and the output variable is z. If A1 has a required structure, it is asked to determine the structure of A2, so that A has a functioning imposed (A it is known).

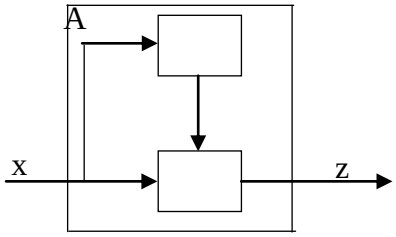

Fig.2.

PROBLEM SOLUTION

The problem formulated above can be addressed in two ways:

-A1 is linearly independent automaton made of delay blocks (D-type flip flops).
- A1 has an arbitrarily chosen structure.

In the first case, when A1 is linearly independent automaton, made of delay blocks, the automaton structure is shown in figure 3, where A1 is a shift register realized with delay bistables and:

$$x1=x(t-D)$$
$$x2=x(t-2D)$$
$$\dots$$
$$xk=x(t-kD)$$

The Automaton structure determination of A2 can be done using the following algorithm for synthesis of automatons [5]:

- it builds the synchronization tree of automaton A. The maximum length of paths in the tree is equal to the number of delay blocks of A1
- it analyzes the uncertain vectors of the final nodes and it determines the number of variables necessary to reduce uncertainty in each component of the vectors. These variables will define the states necessary to the functioning of A2
- it builds the transition table of A2, so it meet the requirements of the previous paragraph and in the same time to cover the operation of A
- it implements A2 and the structure obtained is coupled with A1

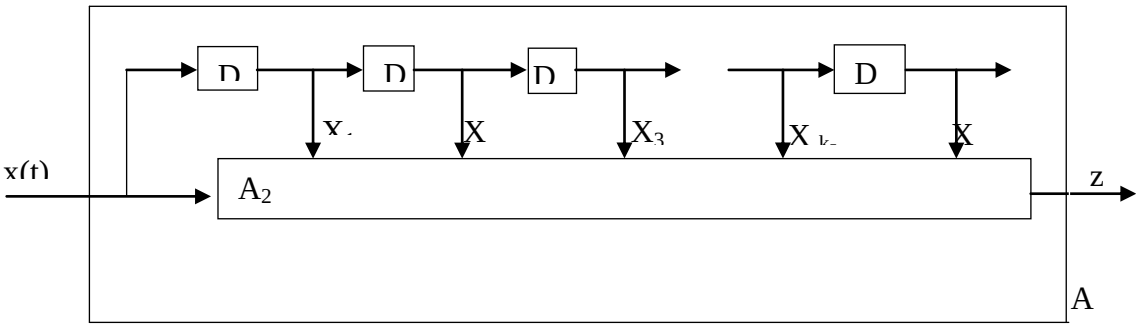

Fig. 3

For example it considers the automaton A1 with two delay blocks and the automaton A described by the operating tables Table 1 and Table 2 from below and that has the block configuration shown in figure 3.
In these tables the automaton A (consisting of A1 implemented with D flip-flop and A2 whose structure must be determined and then implemented with logic gates) may have as initial states ones 5 states from the first column denoted by Si and may evolve in one of the states of the second columns if x = 0 or in one of the states of the third column if x = 1.

Table 1

S_i \ x	0	1
A	B, 0	E, 0
B	D, 0	C, 1
C	A, 1	A, 0
D	B, 0	C, 1
E	D, 0	A, 0

Table 2

S_i \ x	0	1
S_1	S_2, 0	S_5, 0
S_2	S_4, 0	S_3, 1
S_3	S_1, 1	S_1, 0
S_4	S_2, 0	S_3, 1
S_5	S_4, 0	S_1, 0

The first step of the synthesis algorithm relates to the construction of the synchronization tree for automaton A (shown in figure 4). The levels of branches of the tree determines the number of delay blocks of A1, thus being the three levels are used three variables x1, x2, x3 associated to these levels.

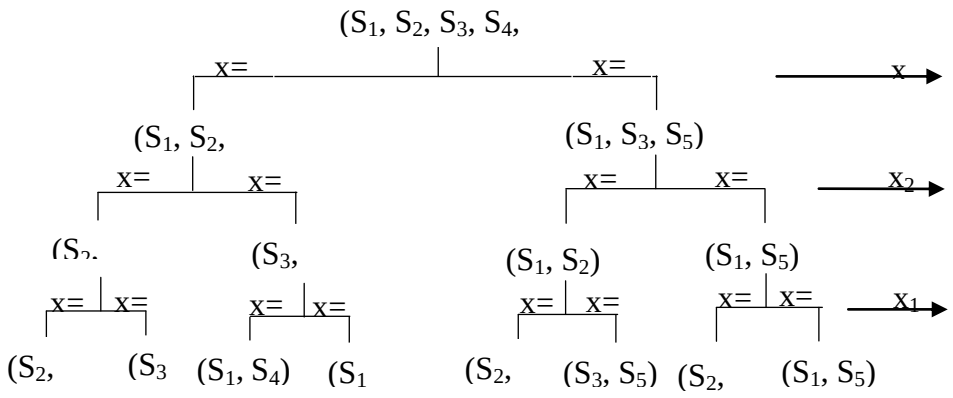

Fig. 4

For the second step of the synthesis algorithm the uncertain vectors, which correspond to the finals nodes of the synchronizing tree are analyzed. To eliminate the uncertainties of the automaton evolutions in one of the states (the vector components of uncertainty) is necessary to introduce two additional variables s_1' and s_2 'These variables define the conditions for the operation of A_2.
The evolution table of A_1 is built as in Table 3. The final vectors of uncertainty in the graph obtained for all branches of the synchronism tree (s_2, s_4), (s_3), (s_1, s_4), (s_1), (s_2, s_4), (s_3, s_5), (s_2, s_4), (s_1, s_5) are included in the columns of Table 3 for appropriate combinations of the variables x_1, x_2, x_3. It is observed that A_2 is defined on the set of states {s_1',s_2'}.The next step is to build the table of evolution (transition) of automaton A_2 according to Table 4.

The table is created by taking into account the transformations below.

Table 3

x_1 / x_2 / x_3	0 0 0	0 0 1	0 1 1	0 1 0	1 1 0	1 1 1	1 0 1	1 0 0
s_1'	s_2	s_2	s_2	s_1	s_1	s_1	s_3	s_3
s_2'	s_4	s_4	s_4	s_4	-	s_5	s_5	-

Table 4

x / x_1 / x_2 / x_3	x=0								x=1							
	0 0 0	0 0 1	0 1 1	0 1 0	1 1 0	1 1 1	1 0 1	1 0 0	0 0 0	0 0 1	0 1 1	0 1 0	1 1 0	1 1 1	1 0 1	1 0 0
S'$_1$	S'$_{2,}$0	S'$_{2,}$0	S'$_{2,}$0	S'$_{1,}$0	S'$_{1,}$0	S'$_{1,}$0	S'$_{1,}$0	S'$_{1,}$1	S'$_{1,}$1	S'$_{1,}$1	S'$_{1,}$1	S'$_{2,}$0	S'$_{2,}$0	S'$_{2,}$0	S'$_{1,}$0	S'$_{1,}$0
S'$_2$	S'$_{1,}$0	S'$_{1,}$0	S'$_{1,}$0	S'$_{1,}$0	-	S'$_{2,}$0	S'$_{2,}$0	-	S'$_{1,}$1	S'$_{1,}$1	S'$_{1,}$1	S'$_{1,}$1	-	S'$_{1,}$0	S'$_{1,}$0	-

$$S_1' \longrightarrow S_2(A_1) \xrightarrow{x=0} S_4,0\ (A) \Rightarrow S_2',0$$

$$S_2' \longrightarrow S_4(A_1) \xrightarrow{x=0} S_2,0\ (A) \Rightarrow S_1',0$$

From the transition table it is deduced that:

$$s_1' = s_2' \cdot \overline{x_1} \cdot \overline{x} + \overline{x} \cdot \overline{x_1} \cdot x_2 \cdot \overline{x_3} + s_1' \cdot x_1 \cdot \overline{x} + \overline{x_1} \cdot \overline{x_2} \cdot x + \overline{x_1} \cdot x_2 \cdot x_3 \cdot x + s_2' \cdot \overline{x_1} \cdot x +$$

$$+ s_1' \cdot x_1 \cdot \overline{x_2} \cdot x + s_2' \cdot x_1 \cdot x_3 \cdot x;\ \ s_2' = \overline{s_1'};\ \ Z = s_1' \cdot x_1 \cdot \overline{x_2} \cdot \overline{x} + \overline{x_1} \cdot \overline{x_2} \cdot x + \overline{x_1} \cdot x_2 \cdot x_3 \cdot x + \overline{x_1} \cdot x$$

In the figure 5 it is presented the implementation of automaton A.

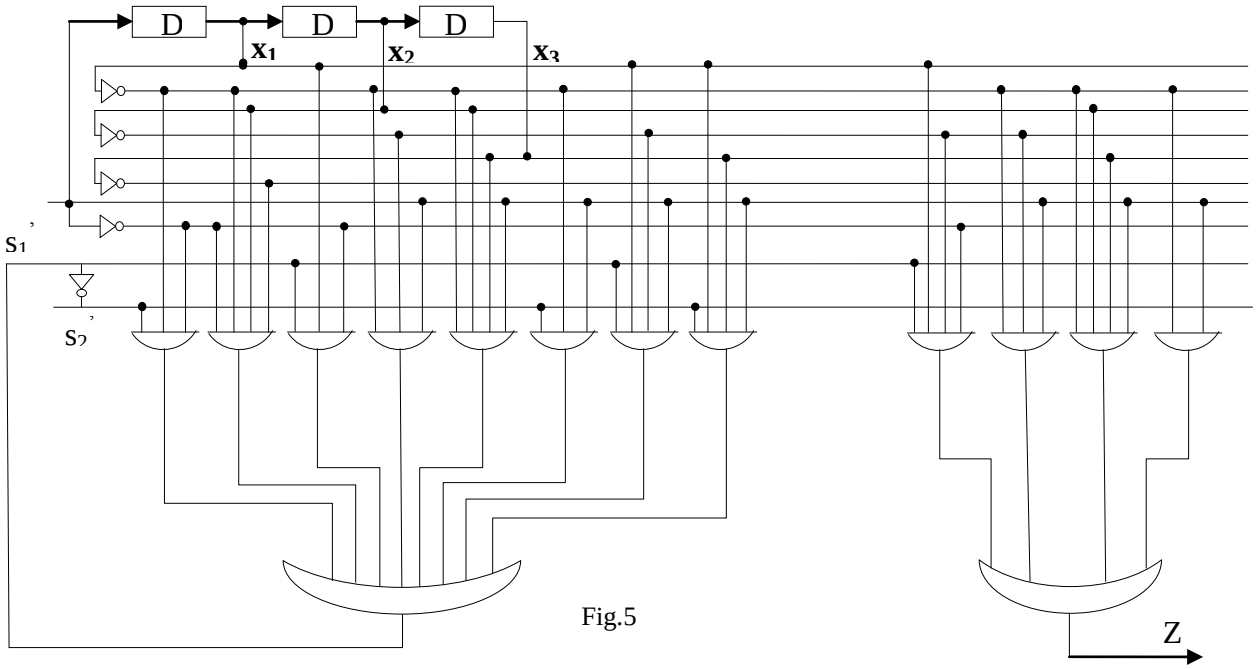

Fig.5

CONCLUSIONS

The automatons have an important role in the theory of computation, compiler design, artificial intelligence, analysis and formal verification. Their study is very important especially when you have composed automatons with a complex structure that we want to determine using the synthesis algorithm described in paper.

REFERENCES

[1] http://en.wikipedia.org/wiki/Automata_theory

[2] www1.fs.cvut.cz/cz/u12110/ui/A-**theory**.**doc**

[3] www.cs.duke.edu/courses/.../**lectures**/**lecture** 02....

[4] http://cs.stanford.edu/people/eroberts/courses/soco/projects/2004-05/automata-theory/basics.html

[5] Ilie Diaconu, Automate secvenţiale şi programabile- note de curs

[6] M. Ivănescu, I. Cauţil, Automate industriale, Ed. Scrisul Românesc, Craiova, 1984.