Testing Programmable Logic Controllers from Finite State Machines specification

Julien Provost, Jean-Marc Roussel, Jean-Marc Faure

Testing Programmable Logic Controllers from Finite State Machines specification

Julien Provost, Jean-Marc Roussel and Jean-Marc Faure
LURPA, ENS Cachan
61 Avenue du Président Wilson, 94235 Cachan Cedex, France.
Email: {provost,roussel,faure}@lurpa.ens-cachan.fr

***Abstract*—This paper shows, on the basis of experiments, that execution of conformance tests of programmable logic controllers with minimum-length test sequences built from specifications in FSMs may yield spurious results. A new approach to build test sequences is then proposed to remove, or at least strongly lessen, this issue.**

I. Introduction

Conformance test is a model-based functional test technique whose aim is to check whether an implementation behaves correctly with respect to a specification. In this paper, the implementation is a PLC (Programmable Logic Controller) which executes cyclically a control algorithm and it will be assumed that the specification is described in the form of a Finite State Machine (FSM) with inputs and outputs. Industrial specifications of logic control are obviously not described as FSMs but it has been shown [1] that any specification in the IEC 60848 standardized language [2] can be translated into this formalism. Moreover, the approach proposed in the previous reference can be applied to other industrial specification languages; the assumption on the specification model is then quite realistic.

Generally speaking, conformance test comprises two phases (Fig. 1):

- Construction of a test sequence from the specification model.
- Execution of this sequence on the implementation to obtain a conformance verdict.

Many significant results have been obtained in the domain of construction of test sequences from FSMs; the interested reader is referred to [3] for a good synthesis. Execution of a test sequence is not so widely addressed in the literature; contributing to fill this gap is the overall objective of this paper which is based on experimental results obtained with a specific test-bench for PLCs.

Construction of a test sequence requires a test objective and an optimization criterion be previously defined. In the sequel of this paper, the objective is to fire at least once each transition of the FSM; the coverage rate of transitions is then equal to 100%, what is mandatory for critical systems. The usual optimization criterion of the test sequence is minimum-length, to limit the duration of the execution. The first contribution of this paper is to show, on the basis of experiments, that a sequence built with this criterion (minimum-length test sequence) may lead to spurious results, however: implementation errors may be not detected while correct controllers may be declared non-conform.

A detailed analysis of these experimental results has permitted to pinpoint the origin of these issues: synchronous events issued from the test-bench may be interpreted as asynchronous by the controller under test; hence, the input sequence used by the PLC to compute its outputs is not really that defined during test sequence construction. This can be avoided by using a SIC (Single Input Change) test sequence, in which the value of only one input can change at one and the same time; synchronous input values changes are then not possible. Nevertheless, it is not always possible, with the test objective selected, to build such a sequence from an FSM, as explained in [4]. To limit the error risk during test execution in this case, the second contribution of this work is a method to construct a sequence, termed maximum consecutive SIC (mc-SIC) sequence, that comprises a set of consecutive SIC test steps to explore entirely the SIC-testable part of the FSM.

The outline of this paper is the following. The next section is a brief reminder on conformance test of FSMs. Section III describes the experiments which were performed by using a minimum-length test sequence and discusses the results obtained. Construction of a mc-SIC test sequence is detailed in section IV; the minimum-length and mc-SIC test sequences are then compared. Last, prospects for other strategies to construct test sequences that limit the risk of spurious results during test execution are sketched in the last section.

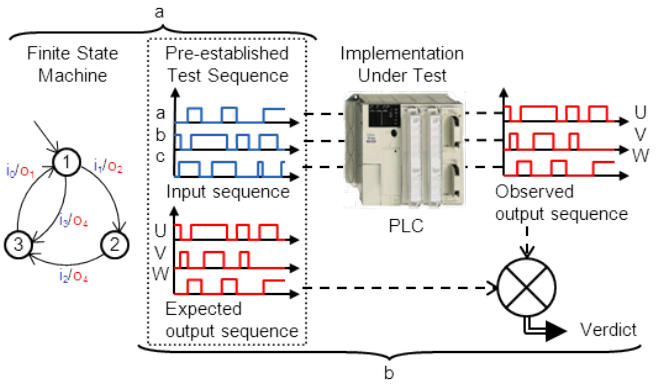

Fig. 1. Test sequence construction (a) and execution (b)

II. Background

A. PLCs behavior specification with FSMs

Among the numerous formalisms developed to describe the behavior of discrete event systems (DES), finite state machines (FSMs) with inputs/outputs, e.g. Mealy machines, are well suited to formal specification of PLCs that receive and send logic data from/to the plant. This formalism cannot be used by control engineers obviously, but it is possible to translate models in industrial standardized languages in this kind of formal models, as presented in [1].

The contributions of this work will be illustrated with a simple example: a logic controller with 3 logic inputs (a, b and c), and 3 logic outputs (U, V and W) (Fig. 2). The specification of this controller is given figure 3. The input and output alphabets of this FSM are respectively the controller inputs valuations (v_I) and outputs valuations (v_O) (Table I). As the state space comprises 5 states ($s_1 = s_{Init}$: initial state, s_2, s_3, s_4 and s_5), the transition and output functions $\delta(s, v_I)$ and $\lambda(s, v_I)$ of the FSM are defined for 40 ($5 \cdot 2^3$) couples (s, v_I). A couple (v_I, v_O) is associated to each transition of the machine; the first element of this couple represents the inputs valuation which provokes the transition from the source state, the second element the outputs valuation emitted when this transition is fired. The target state s_t reached when firing, from a source state s_s a transition labeled v_I/v_O is defined through the transition function ($s_t = \delta(s_s, v_I)$), while the emitted outputs valuation is defined through the output function ($v_O = \lambda(s_s, v_I)$). In this example, the specification is complete (the transition function is completely defined), minimal (there are no equivalent states) and every state is reachable.

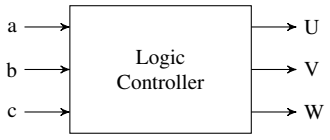

Fig. 2. Inputs/Outputs of the controller

	Input alphabet									Output alphabet							
	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7		o_0	o_1	o_2	o_3	o_4	o_5	o_6	o_7
a	0	1	0	1	0	1	0	1	U	0	1	0	1	0	1	0	1
b	0	0	1	1	0	0	1	1	V	0	0	1	1	0	0	1	1
c	0	0	0	0	1	1	1	1	W	0	0	0	0	1	1	1	1

TABLE I
Correspondence inputs and outputs valuations / FSM alphabets

B. Finite State Machine conformance test principle

This section recalls briefly the principle of the conformance test of FSM; the reader is referred to [3] for further details.

The problem of conformance test of FSM can be described as follows: Let S be a known machine (the specification) and I an unknown machine (the implementation under test) which can be only observed through its inputs and outputs, determine

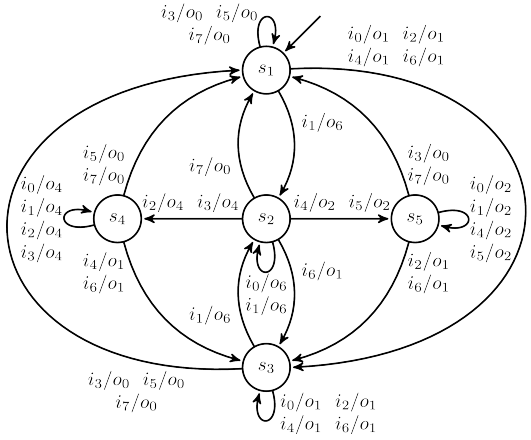

Fig. 3. Graphical representation of the FSM

by a test that includes a finite sequence of inputs and expected outputs whether I is equivalent to S or not.

In order to solve this problem, it is generally assumed that the specification is minimal and strongly connected (each state is reachable, immediately or not, from any other state). Then, the equivalence between an implementation I and a specification S consists in verifying that none of the following errors happens during the test of I:

- Output error: s_s being the active state, when the input i occurs, I produces the output o' instead of the expected output o.
- Transfer error: s_s being the active state, when the input i occurs, the transition labeled i/o is fired but the arrival state is s'_t instead of s_t.

The test sequence is constructed from S and must permit to detect these two kinds of errors, for each state and each transition. Hence, each elementary test corresponds to a transition $s_s \xrightarrow{i/o} s_t$ of S and is defined as follows:

1) Go to s_s (synchronization).
2) Apply input i and check whether the emitted output is o.
3) Check whether the arrival state is s_t (identification).

C. Minimum-length test sequence construction

The construction of a test sequence, using the transition tour method [5], can be automated to obtain a minimum-length test sequence. This optimization problem is a particular solution of a well-known problem in graph theory: the Chinese Postman Problem [6]. The general formulation of this problem is the following: Find a minimum-length closed walk that traverses at least once each **edge** (or arc) of the graph that describes the structure of the FSM. As this graph is directed, but not weighted, the optimization problem is simplified.

For the example given figure 3 this sequence contains 40 test steps (Table II) and fulfills the test objective (coverage rate of the transitions equals 100%). In practice, each elementary test permits to test sequentially two arcs of the FSM: (s_s, v_I, s_t, v_O) and (s_t, v_I, s_t, v_O). This is possible because the inputs of the PLC under test can be scanned several times

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
s_s:	s_1	s_2	s_3	s_2	s_2	s_4	s_3	s_2	s_4	s_3	s_2	s_4	s_4	s_4	s_1	s_2	s_4
s_t:	s_2	s_3	s_2	s_2	s_4	s_3	s_2	s_4	s_3	s_2	s_4	s_4	s_4	s_1	s_2	s_4	s_1
a:	1	0	1	0	1	0	1	0	0	1	0	1	0	1	1	0	1
b:	0	1	0	0	1	1	0	1	0	0	1	0	0	1	0	1	0
c:	0	1	0	0	0	1	0	0	1	0	0	0	0	1	0	0	1

18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
s_1	s_2	s_5	s_3	s_2	s_5	s_3	s_2	s_5	s_5	s_5	s_1	s_2	s_5	s_1	s_3	s_2
s_2	s_5	s_3	s_2	s_5	s_3	s_2	s_5	s_5	s_5	s_1	s_2	s_5	s_1	s_3	s_2	s_1
1	1	0	1	0	0	1	0	1	0	1	1	0	1	0	1	1
0	0	1	0	0	1	0	0	0	0	1	0	0	1	1	0	1
0	1	1	0	1	0	0	1	0	0	1	0	1	0	1	0	1

35	36	37	38	39	40
s_1	s_3	s_1	s_3	s_1	s_3
s_3	s_1	s_3	s_1	s_3	s_1
0	1	0	1	0	1
0	1	1	0	0	1
1	1	0	1	0	0

TABLE II
INPUTS VALUATIONS FOR THE MINIMUM-LENGTH TEST SEQUENCE

during one test step if the duration of the step is greater than several PLC scanning cycles.

III. EXPERIMENTATION

This section focuses on execution of the conformance test of a PLC from a minimum-length test sequence. The experimental conditions are first given. As spurious test results may be obtained, an explanation is then proposed and verified by complementary experiments. The last sub-section provides some guidelines for conformance test from FSM.

A. Experimental conditions

The test-bench used in this work is built from a remote input/output module (RIOM) with 16 logic inputs and 16 logic outputs that can be remotely set/reset from a computer via a Modbus TCP/IP network (Fig. 4). Each output of this RIOM is connected upstream from an input of the logic controller under test, while each of its inputs is connected downstream from an output of this controller. This open solution provides high flexibility at a very low cost.

A modular PLC (Phoenix Contact ILC 170 ETH 2TX) was selected for these experiments. I/O scanning can be cyclic or periodic and the programming environment complies with the IEC 61131-3 standard [7]. The control algorithm is implemented in the form of a Structured Text code obtained automatically from the specification according to the method proposed in [8]. This solution allows either to generate an a priori correct code or to artificially introduce implementation errors. Once programmed, the controller is disconnected from its programming environment for the entire test execution.

Once the controller initialized, test execution consists, for each step of the test sequence, in:

- sending to the controller the input signals that correspond to this step;
- waiting for outputs stabilization;
- reading the values of the output signals emitted by the controller under test;

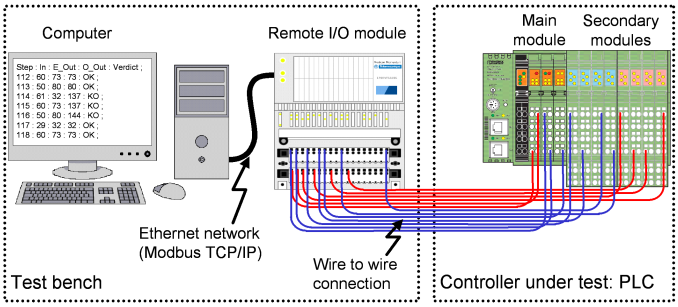

Fig. 4. Experimental device

- comparing these values to the expected ones and delivering a verdict;
- recording the results.

It matters to underline that the test-bench cycle is not synchronized with the PLC I/O scanning cycle so that the test conditions be as close as possible to the operation conditions of a closed-loop DES; the behavior of a plant is indeed not synchronized with that of its controller.

B. Results and discussion

The first experiments with short test sequences were conclusive. Unfortunately, erroneous detections of non-conformance, which led to reject correct controllers, and non-detections of non-conform controllers occurred during experiments with longer test sequences. These issues appeared whatever the PLC.

A detailed analysis of the results files showed that these issues occurred always for test steps where several input values were changed simultaneously by the test-bench, e.g. steps 2 or 3 of table II; test steps where only one input value was changed never caused such issues. An explanation was then put forward: synchronous input changes generated by the bench may be seen as asynchronous by the controller (Fig. 5) because all inputs are not read simultaneously. The input values used by the implementation to compute its outputs are then different from those planned.

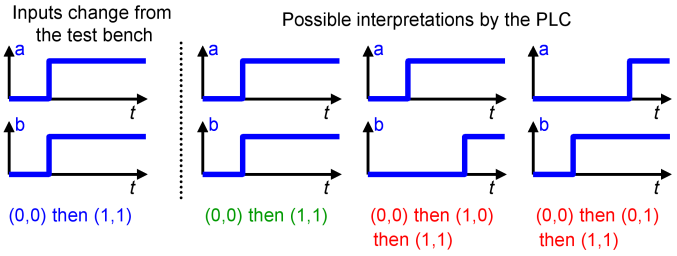

Fig. 5. Possible interpretations of synchronous changes of 2 inputs

C. Experimental verification

To verify this explanation, the following experiment has been then carried out:

- The test-bench sends to the controller a set of 8 signals that vary simultaneously and in an identical manner (from 0 to 1, then from 1 to 0) with a period far greater (more than 10 times) than the maximal PLC cycle time. Hence, the changes (rising and falling edges) of these 8 signals are strictly synchronous.
- When the controller detects the first change of one of its inputs, the current value of each input is copied on the corresponding output (8 outputs are used, each one associated to an input). Thus, the value of the output vector is equal to the value of the input vector at the moment when the first inputs change has been detected.
- The test-bench reads the emitted output values and compares them with the expected values, which should be identical to the input values sent to the controller if no inputs reading error (synchronous changes seen asynchronous) has occurred. A counter is incremented when an error is detected, i.e. when the outputs vector differs from the inputs one.

Table III gives the error rates according to the PLC operating mode (cyclic or periodic I/O scanning) and the distribution of the inputs on the PLC internal components (all inputs connected on the main module, connected on the secondary module, or distributed on both main and secondary modules). These figures were obtained from series of 200,000 experiments for each configuration and each distribution.

These results validate clearly the proposed explanation. As the PLC scanning cycle is not synchronized with the test-bench cycle, synchronous events issued from the bench may be seen asynchronous. In the case of inputs distributed on several modules (last two lines of table III), the errors rate is strongly increased due to the communication between the modules that induces extra delays.

The difficulty to implement theoretical results issued from DES theory on real PLCs has already been underlined in [9] when attempting to implement supervisors.

D. Guidelines for conformance test from FSM specifications

These experiments have clearly showed that using a minimum-length test sequence obtained from an FSM to test a PLC may yield spurious results. This approach relies indeed on the implicit assumption that all inputs changes generated by the test-bench are correctly detected by the controller under test, even if several input values are simultaneous modified, which is not always true. To overcome this issue, several solutions can be imagined:

- synchronization of the test-bench with the controller under test;
- test execution for the configurations of the controller that lessen the error rate (slow periodic inputs scanning and inputs distribution on a single module);
- multiple executions of the same test and statistical analysis of results;
- construction of a test sequence, termed Single Input Change (SIC) test sequence, that does not include simultaneous changes of the input values.

The first solution requires additional communication between the test-bench and the controller be introduced and, above all, does not represent the operation conditions of a real closed-loop DES, as mentioned above. The second and third solutions are not really suitable for controllers of highly critical systems because they imply statistical analyses of results. Hence, the last one shall be privileged and is discussed in the next section.

Distribution \ Configuration		Cyclic	Periodic 10ms	Periodic 20ms
main module	Change 0 to 1	0.052 %	0.022 %	0.014 %
	Change 1 to 0	1.597 %	0.886 %	0.455 %
secondary module	Change 0 to 1	0.073 %	0.032 %	0.013 %
	Change 1 to 0	2.490 %	1.059 %	0.525 %
distributed on the two modules	Change 0 to 1	30.05 %	39.59 %	19.92 %
	Change 1 to 0	37.62 %	42.51 %	21.66 %

TABLE III
ERROR RATES

IV. SIC-TESTABILITY AND MC-SIC TEST SEQUENCE GENERATION

A. Formal definition of test sequences

During test execution, a test sequence is seen as an ordered list of couples (inputs valuation, expected outputs valuation) which represents its external view:

$$\left[(v_I^0, v_O^0), (v_I^1, v_O^1), ..., (v_I^n, v_O^n)\right] \in (V_I \times V_O)^* \quad (1)$$

These two valuations are not independent, however. The outputs valuation is that associated to the transition which goes from a source state s_s to a target state s_t for the inputs valuation. Hence, an elementary conformance test step et is defined by the following 4-tuple:

$$et = (s_s, v_I, s_t, v_O) \in S \times V_I \times S \times V_O \quad \text{where} \begin{cases} s_t = \delta(s_s, v_I) \\ v_O = \lambda(s_s, v_I) \end{cases} \quad (2)$$

Thus, a test sequence is an ordered list of elementary test steps, and a **consistent** test sequence TS is a test sequence such as the source state of the k^{th} elementary test step is equal to the target state of the $(k-1)^{th}$ step.

$$TS = [(s^0, v_I^0, \delta(s^0, v_I^0), \lambda(s^0, v_I^0)), \cdots, (s^n, v_I^n, \delta(s^n, v_I^n), \lambda(s^n, v_I^n))] \mid \forall k > 1, s^k = \delta(s^{k-1}, v_I^{k-1}) \quad (3)$$

B. Properties of test sequences

A test sequence TS may be:

P1: initializable, i.e. the source state of the first test step is the initial state of the FSM, and the input valuation v_I^0 is such that this state is stable (the target and source states are identical):

$$\begin{cases} s^0 = s_{Init} \\ \delta(s^0, v_I^0) = s^0 \end{cases} \quad (4)$$

P2: complete, i.e. there is at least one test step for each element of the transition function:

$$\forall (s, v_I) \in (S \times I),\ (s, v_I, \delta(s, v_I), \lambda(s, v_I)) \in TS \quad (5)$$

P3: based on a SIC test sequence.

To express formally this latter property, the SIC relation between two inputs valuations must be first defined. The definition below is based on the representation of an inputs valuation by the subset of logic inputs that only contains the inputs that are True (also noted 1) for this valuation. Thus, two inputs valuations v_I and v_I' satisfy a SIC relation iff[1]:

$$dim((v_I \backslash v_I') \cup (v_I' \backslash v_I)) = 1 \quad (6)$$

For example, the inputs valuations used in steps 3 and 4 of the test sequence given table II satisfy a SIC relation since $dim((\{a\}\backslash\{\}) \cup (\{\}\backslash\{a\})) = dim(\{a\} \cup \{\}) = 1$

Unfortunately, any FSM does not always permit to construct a test sequence that satisfies these three properties. Thus, before considering the construction of a SIC test sequence, the SIC-testability of the FSM – ability of this FSM to satisfy the three properties previously pinpointed – must be checked.

C. Checking SIC-testability of an FSM

A detailed presentation of the algorithm to check whether a specification is SIC-testable or not can be found in [4]. For room reasons, only its principles and an illustration are presented in this paper. This algorithm is based on the following two observations:

- An elementary test step (s_s, v_I, s_t, v_O) is SIC-testable if it can be included into an initializable SIC test sequence. If the elementary test step (s_s, v_I, s_t, v_O) is SIC-testable, the elementary test step (s_t, v_I, s_t, v_O) is also SIC-testable.
- If the elementary step (s_t, v_I, s_t, v_O) is SIC-testable, it is always possible to add to the test sequence an elementary step $(s_t, v_I', \delta(s_t, v_I'), \lambda(s_t, v_I'))$ where v_I and v_I' satisfy a SIC relation.

On these bases, the set of elementary test steps which are SIC-testable can be obtained by a fixed-point calculation on the elementary test steps; this calculation begins with the set of elementary steps which start from the initial state of the FSM and satisfy $\delta(s_{Init}, v_I^0) = s_{Init}$. At the end of this iterative calculation, the FSM is SIC-testable iff the final set contains all elementary test steps that can be defined from its behavior description; it is then possible to build a SIC and complete (and also consistent and initializable) test sequence. Otherwise, this final set defines the SIC-testable part of the FSM.

[1] $dim(A)$ is the dimension of set A.
An input valuation can be represented either by a minterm or by the set of input variables that are True (noted 1) for this valuation (see table IV).

s \ v_I	i_0 or $\{\}$	i_4 or $\{c\}$	i_6 or $\{b,c\}$	i_2 or $\{b\}$	i_3 or $\{a,b\}$	i_7 or $\{a,b,c\}$	i_5 or $\{a,c\}$	i_1 or $\{a\}$
s_1	s_3	$^1 s_3$	$^1 s_3$	$^1 s_3$	$^0(s_1)$	$^0(s_1)$	$^0(s_1)$	$^1 s_2$
s_2	$^2(s_2)$	$^3 s_5$	s_3	$^3 s_4$	$^2 s_4$	s_1	$^2 s_5$	$^1(s_2)$
s_3	$^2(s_3)$	$^1(s_3)$	$^1(s_3)$	$^1(s_3)$	$^2 s_1$	$^2 s_1$	$^2 s_1$	$^3 s_2$
s_4	$^4(s_4)$	$^5 s_3$	$^4 s_3$	$^3(s_4)$	$^2(s_4)$	$^3 s_1$	$^4 s_1$	$^3(s_4)$
s_5	$^4(s_5)$	$^3(s_5)$	$^4 s_3$	$^5 s_3$	$^4 s_1$	$^3 s_1$	$^2(s_5)$	$^3(s_5)$

TABLE IV
TRANSITION FUNCTION AND SIC-TESTABLE PART CALCULATION

D. Illustration on the example

Table IV gives a tabular representation of the transition function of the example. In this Huffman table [10], each cell represents an elementary behavior ($\delta(s_s, v_I) = s_t$), whose source state s_s, inputs valuation v_I and target state s_t are respectively given by the corresponding line, column and content of the cell.

Circled cells represent stable couples (s, v_I), i.e. couples (s, v_I) such as $\delta(s, v_I) = s$, and non-circled cells couples that imply a change of the active state ($\delta(s_s, v_I) = s_t \neq s_s$).

An initializable and complete test sequence starts from a circled cell of the first line ($s_{Init} = s_1$) and covers all cells of the table. To be consistent, the test sequence must be built according to the following two rules:

R1: From circled cells, only horizontal changes of cells are possible. Indeed, the inputs valuations of two successive test steps can be different, but the source state of the second step must be identical to the target state of the first one.

R2: From non-circled cells (source and target states of the associated test step are different), only one vertical change is possible. Indeed, these cells represent non-stable couples (s, v_I), the change of the active state leads to the circled cell that represents $\delta(s, v_I)$.

To build a SIC test sequence, rule R1 must be restricted:

R3: Since a SIC test sequence does not include simultaneous changes of input values, an horizontal change must correspond to a couple of inputs valuations which satisfy a SIC relation.

Rule R3 implies that in the case of a controller with n logic inputs, only n cells among $(2^n - 1)$ can be reached from a circled cell.

Table IV also presents the result of the fixed point calculation for the example. The number k of the iteration during which the test step was found SIC-testable is indicated at the top-left corner of each cell. For example, the elementary test step associated to the cell $(s_1, \{a, b, c\})$ is obtained at iteration 0 (initialization). The test steps associated to cells $(s_1, \{b, c\})$

and $(s_3, \{b, c\})$ are obtained at iteration 1, since $\{a, b, c\}$ and $\{b, c\}$ satisfy a SIC relation, and so on. Calculation stops at the fifth iteration. The final set contains only 37 test steps; the steps that do not belong to this set are represented by grayed cells. Hence, the FSM is not SIC-testable; its SIC-testable part is given by the cells which are not grayed.

E. Construction of the mc-SIC test sequence

When the FSM is not SIC-testable, a maximum consecutive SIC (mc-SIC) test sequence must be built to limit the risk of spurious results during test execution. This sequence is composed of an initializable minimum-length SIC test sequence, which is computed from the SIC-testable part, followed by a minimum-length MIC test sequence, obtained from the non-SIC-testable part. Both sequences start from and end on the initial state of the FSM; there is no gap between the final state of the SIC sequence and the initial state of the MIC sequence.

The construction of the SIC sequence is based on a directed graph whose nodes represent all couples (s, v_I) that can be defined on the SIC-testable part. For a controller with n logic inputs, the arcs between the nodes are defined as follows:

- only one arc starts from a node that corresponds to a couple (s, v_I) such as $\delta(s, v_I) \neq s$; the target node of this arc is the node that corresponds to $(\delta(s, v_I), v_I)$;
- n arcs start from a node that corresponds to a couple (s, v_I) such as $\delta(s, v_I) = s$; the target nodes of these arcs correspond to couples (s, v'_I) such as v_I and v'_I satisfy a SIC relation.

The minimum-length SIC test sequence can then be obtained by using the results of graph theory on a well-known problem: the Traveling Salesman Problem [11]. The general formulation of this problem is the following: Find a minimum-length closed walk that traverses each **node** of the graph at least once.

The inputs valuations for a mc-SIC test sequence of the example are given Table V. This test sequence contains 45 SIC test steps, to test the 37 couples (s, v_I) of the SIC-testable part, and 5 MIC test steps, to test the 3 couples (s, v_I) of the non-SIC-testable part. Comparison to Table II shows that the use of a mc-SIC test sequence instead of a MIC test sequence increases the length of the test sequence (50 test steps instead of 40) but strongly reduces the number of test steps where synchronous changes of several inputs occur (only 3 couples (s, v_I) are tested in these conditions with the mc-SIC sequence, instead of 35 with the MIC sequence). Then, the test execution on a PLC will last a bit longer, but the risk of erroneous verdicts is widely lessened.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
s_s:	s_1	s_2	s_2	s_4	s_4	s_3	s_3	s_2	s_5	s_1	s_1	s_3	s_3	s_2	s_5	s_5	s_1
s_t:	s_2	s_2	s_4	s_4	s_3	s_3	s_2	s_5	s_1	s_1	s_3	s_3	s_2	s_5	s_5	s_1	s_2
a:	1	0	0	0	0	0	1	1	1	1	0	0	1	1	1	1	1
b:	0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	1	0
c:	0	0	0	0	1	0	0	1	1	0	0	0	0	1	0	0	0

	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
	s_2	s_5	s_5	s_5	s_3	s_1	s_2	s_4	s_1	s_3	s_3	s_1	s_3	s_3	s_2	s_4	s_4
	s_5	s_5	s_5	s_3	s_1	s_2	s_4	s_1	s_3	s_3	s_1	s_3	s_3	s_2	s_4	s_4	s_1
	1	1	0	0	1	1	1	1	0	0	1	0	0	1	1	1	1
	0	0	0	1	1	0	1	1	1	0	0	0	0	0	1	0	0
	1	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	1

	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
	s_1	s_2	s_2	s_4	s_3	s_1	s_1	s_2	s_2	s_5	s_3	s_1	s_3	s_2	s_3	s_2
	s_2	s_2	s_4	s_3	s_1	s_1	s_2	s_2	s_5	s_3	s_1	s_3	s_2	s_3	s_2	s_1
	1	0	0	0	1	1	1	0	0	0	1	0	1	0	1	1
	0	0	1	1	1	0	0	0	0	1	1	0	0	1	0	1
	0	0	0	1	1	1	0	0	1	1	1	0	0	1	0	1

TABLE V
INPUTS VALUATIONS FOR THE MC-SIC TEST SEQUENCE

V. CONCLUSIONS AND PROSPECTS

The experiments presented in this paper have shown that minimum-length is not always the right criterion to build a test sequence for conformance test of PLCs; test execution with such sequences may yield spurious results. SIC, if the specification is SIC-testable, or mc-SIC test sequences must be privileged to remove or lessen this risk and to improve the confidence level in test results.

On-going work focuses on construction of other kinds of test sequences for non-SIC-testable specifications, e.g. test sequences where the number of MIC test steps is minimized according to the number of inputs changes in the steps, and on construction of non error-prone test sequences for other test objectives.

ACKNOWLEDGMENT

This work is funded by the French National Research Agency (TESTEC project, Ref. TLOG 07-022)

REFERENCES

[1] J. Provost, J.-M. Roussel, and J.-M. Faure, "Translating Grafcet specifications into Mealy machines for conformance test purposes," Control Engineering Practice, 2010. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00547891

[2] IEC 60848, GRAFCET specification language for sequential function charts. International Electrotechnical Commission, 2002, no. 2.

[3] D. Lee and M. Yannakakis, "Principles and methods of testing finite state machines - a survey," in Proceedings of the IEEE, vol. 84, no. 8, 1996, pp. 1090–1123.

[4] J. Provost, J.-M. Roussel, and J.-M. Faure, "SIC-testability of sequential logic controllers," in Proceedings of 10th International Workshop on Discrete Event Systems (WODES 2010), August 2010, pp. 203–208. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00512767

[5] S. Naito and M. Tsunoyama, "Fault detection for sequential machines by transitions tours," in Proceedings of the IEEE Fault Tolerant Computer Symposium, 1981, pp. 238–243.

[6] K. Mei-Ko, "Graphic programming using odd or even points," Chinese Mathematics, vol. 1, pp. 273–277, 1962.

[7] IEC 61131-3, Programmable controllers - Part 3: Programming languages. International Electrotechnical Commission, 1993.

[8] J. Machado, B. Denis, J.-J. Lesage, J.-M. Faure, and J. Ferreira Da Silva, "Logic controllers dependability verification using a plant model," in Proceedings of the 3rd IFAC Workshop on Discrete-Event System Design, DESDes'06, 2006, pp. 37–42. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00361815

[9] M. Fabian and A. Hellgren, "PLC-based implementation of supervisory control for discrete event systems," in Proceedings of the 37th IEEE Conference on Decision and Control, 1998., vol. 3. IEEE, 1998, pp. 3305–3310.

[10] D. Huffman, "The synthesis of sequential switching circuits," Journal of the Franklin Institute, vol. 257, no. 3, pp. 161 – 190, 1954.

[11] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," Journal of the Operations Research Society of America, vol. 2, no. 4, pp. 393–410, 1954.