A New Algorithm to Solve Synchronous FSM Equations (Extended Abstract)

N. Yevtushenko[¶] S. Tikhomirova[¶] T. Villa[†]

[¶]Tomsk State University, 36 Lenin Str., Tomsk, Russia
yevtushenko@elefot.tsu.ru, szh@ultranet.tomsk.ru

[†]DI, University of Verona, Strada le Grazie, 15 - 37134 Verona, Italy
and PARADES, Via S.Pantaleo, 64 - 00186 Roma, Italy
tiziano.villa@univr.it

June 11, 2008

Abstract

Many problems over discrete event systems can be reduced to solving a synchronous FSM inequality $A \bullet X \subseteq S$ or a synchronous FSM equation $A \bullet X = S$, where X is a free variable and $\bullet$ is the synchronous composition operator. In this paper we address the problem of solving a multi-component FSM equation, We study the most general solution of a synchronous FSM equation defined over several FSMs. In particular, we show that a solvable equation has always a largest solution, then we consider the largest alphabet of actions over which a solution exists, from which it is possible to extract the largest solution over a restricted set of alphabets.

1 Introduction

Many problems over discrete event systems can be reduced to solving a synchronous FSM inequality $A \bullet X \subseteq S$ or a synchronous FSM equation $A \bullet X = S$, where X is a free variable and $\bullet$ is the synchronous composition operator [18]. The applications range from logic synthesis [13, 3, 11], supervisory control and model matching [4, 1, 10] to formal verification [2, 14], testing [20, 19, 9, 8] and protocol conformance [17], discrete games [6, 5]. An inequality as well as a solvable equation is known to have a largest solution [15, 16, 12, 7, 18]. To find optimal solutions with respect to some criteria, one approach is first to find the most general solution containing any particular solution, then to extract a desired solution from the largest solution. Most papers on the subject consider only binary synchronous FSM inequalities and/or equations. However, when solving an equation for designing an optimal component, e.g., in logic synthesis, multi-component FSM equations may occur. In principle, since the synchronous composition operator is associative, we could solve a multi-component FSM equation by converting it into a binary FSM equation, by composing successively all the known components FSMs into a single joint context FSM. However, it seems worth to investigate how to solve such an equation directly, without lumping together the network of given FSMs into a single one.

In this paper, we propose a formula of the composition operator of several FSMs and introduce a multi component FSM equation. We then propose two ways how a largest solution over the given alphabet can be derived. One of these ways proposes to consider the largest alphabet of a possible solution. The complexity of solving an equation over the largest alphabet is polynomial when the specification FSM is deterministic. If an equation is unsolvable over the largest alphabet then it is unsolvable over any alphabet over which the unknown is computed. If the equation is solvable over the largest alphabet then we propose an algorithm for deriving a special reduction of this solution.

The paper is structured as follows. Sec 2 contains preliminaries; the notion of the synchronous operator over several FSMs and multi-component FSM equation are introduced there. In Sec. 3 we propose a new procedure for finding the largest solution over the largest alphabet of a multi component FSM equation. Sec. 4 summarizes the

paper and sketches some results under investigation implied by the new procedure.

2 Preliminaries

2.1 Multi-component FSM composition

In this paper, an FSM is a 5-tuple $M = (S, I, O, T, s_0)$ where S is a finite non-empty set of states with the initial state s_0, $I = I_1 \times \cdots \times I_k$ and $O = O_1 \times \cdots \times O_p$ where $I_1, \ldots I_k$ and $O_1, \ldots, O_p$ are respectively input and output alphabets associated to the input and output ports, $T = I \times S \times S \times O$ is the behaviour relation. An FSM M is observable if for each triple $(i, s, o) \in I \times S \times O$ there exists at most one next state $s' \in S$ such that $(i, s, s', o) \in T$.

Consider now a collection $\mathcal{N}$ of n interacting FSMs $M_1, \ldots M_n$ each of which has input and output ports with associated input and output alphabets (see [21]). Let $\Gamma = \{X_1, \ldots, X_m\}$ be the collection of all the alphabets of the component FSMs, where we assume that the same alphabet corresponds to two ports if and only if these ports are connected or it is a common input alphabet for them; let $\theta \subseteq \Gamma$ be the subset of alphabets corresponding to external input and output ports of the overall system. Therefore the sets θ and Γ and the set of component FSMs completely specify the structure of the network $\mathcal{N}$. In order to establish a normal form of an FSM network we assume the following restrictions:

1. One and the same alphabet cannot be an output alphabet of different component FSMs.

2. One and the same alphabet cannot be an input and an output alphabet of the same component FSM.

3. If an alphabet X_j is only an input alphabet of some component FSM then $X_j \in \theta$ and X_j is an input alphabet of the system.

4. If an alphabet X_j is only an output alphabet of some component FSM then $X_j \in \theta$ and X_j is an output alphabet of the system.

The behaviour of the whole network is described by an FSM that is derived in the following way. Each component FSM M_i is converted into a corresponding automaton $A(M_i) = A_i$. Each automaton A_i is lifted to all the alphabets of Γ(i.e., it is lifted to each alphabet that is not already in the set of input and output alphabets of M_i). The intersection of languages $A_{i\uparrow\Gamma}$has all possible sequences which can occur in the system. In order to define the external behaviour of the system the intersection is projected on the alphabets of the set θ. A reduced observable FSM that corresponds to the obtained automaton is a synchronous composition $\bullet_{\Gamma,\theta}(M_1, \ldots, M_n)$.

In this paper, we further assume that all component FSMs are complete and deterministic and that their composition too is complete and deterministic. Instead the specification is not required to be complete and deterministic.

2.2 Solving an FSM equation for a network of FSMs

Suppose that a network $\mathcal{N}$ of FSMs is composed by known components (the context) $M_1, \ldots, M_{n-1}$ and that the required behaviour of the whole system is described by an FSM M_S. Given some alphabets from Γ, the set of alphabets of the network, we would like to derive an FSM M_n over these alphabets such that $\bullet_{\Gamma,\theta}(M_1, \ldots, M_n) = M_S$, i.e., the language of the composition of M_n with the rest of the components is equal to the language of the specification. The existence of a solution and its features depend on the alphabets over which the unknown M_X is defined. In order to get the normal form of a system of interacting FSMs, the following restrictions must be satisfied by the input and output alphabets of the unknown M_n, i.e., by $I_1^n, \ldots, I_k^n$ and $O_1^n, \ldots O_m^n$ respectively, where $\{I_1^n, \ldots, I_k^n, O_1^n, \ldots, O_m^n\} \subseteq \Gamma$:

1. The set of all input alphabets of the unknown M_X includes a) each input alphabet of the FSM M_S that is not an input alphabet of another known component FSM, and b) each output alphabet of each known component FSM that is not an output alphabet of the FSM M_S or an input alphabet of another known component FSM.

2. The set of all output alphabets of the unknown M_X contains a) each output alphabet of the FSM M_S that is not an output alphabet of another known component FSM, and b) each input alphabet of each known component FSM that is not an input alphabet of the FSM M_S or an output alphabet of another known component FSM.

The set of input alphabets of the unknown M_X can be further extended with input alphabets of other component FSMs, whereas the set of output alphabets of the unknown M_X is precisely the one described in 2., because the sets of output alphabets of two component FSMs must not intersect. Any subset $\rho \subseteq \Gamma$ that satisfies the two previous conditions can be selected as a set of alphabets of the unknown M_X.

As a straightforward corollary to the theorem in [17] stating that the largest solution of the language/FSM inequality $M_A \bullet M_X \subseteq M_S$ is given by $M_X = \overline{M_A \bullet \overline{M_S}}$ we have the following statement.

Theorem 2.1 *The largest solution M_{LS} of the FSM inequality $\bullet_{\Gamma,\theta}(M_1, \ldots, M_{n-1}, M_X) \subseteq M_S$, when M_X is defined over the set of alphabets ρ, is the reduced observable FSM whose language is the largest prefix closed subset of the language of the FSM $\overline{\bullet_{\Gamma,\rho}(M_1, \ldots, M_{n-1}, \overline{M_S})}$.*

The largest complete solution M_{LCS} is the largest complete submachine of the largest FSM solution (if it exists).

Proof. The notation $L(M_i)$ denotes the language of the automaton associated to the FSM M_i.

A sequence α is not in a solution of the inequality $\bullet_{\Gamma,\theta}(L(M_1), \ldots, L(M_{n-1}), L(M_X)) \subseteq L(M_S)$ if and only if $\bullet_{\Gamma,\theta}(L(M_1), \ldots, L(M_{n-1}), \{\alpha\}) \not\subset L(M_S)$. The following statements are equivalent:
$\bullet_{\Gamma,\theta}(L(M_1), \ldots, L(M_{n-1}), \{\alpha\}) \subseteq L(M_S) \Leftrightarrow$
$[L(M_1)_{\uparrow\Gamma} \cap \cdots \cap L(M_{n-1})_{\uparrow\Gamma} \cap \{\alpha\}_{\uparrow\Gamma}]_{\downarrow\theta} \cap \overline{L(M_S)} = \emptyset \Leftrightarrow$
$L(M_1)_{\uparrow\Gamma} \cap \cdots \cap L(M_{n-1})_{\uparrow\Gamma} \cap \{\alpha\}_{\uparrow\Gamma} \cap \overline{L(M_S)}_{\uparrow\Gamma} = \emptyset \Leftrightarrow$
$\alpha \notin [L(M_1)_{\uparrow\Gamma} \cap \cdots \cap L(M_{n-1})_{\uparrow\Gamma} \cap \overline{L(M_S)}_{\uparrow\Gamma}]_{\downarrow\rho} \Leftrightarrow$
$\alpha \notin \bullet_{\Gamma,\rho}(L(M_1), \ldots, L(M_{n-1}), \overline{L(M_S)}) \Leftrightarrow$
$\alpha \in \overline{\bullet_{\Gamma,\rho}(L(M_1), \ldots, L(M_{n-1}), \overline{L(M_S)})}$ □

Theorem 2.2 *Let M_{LS} be the largest solution of the FSM inequality $\bullet_{\Gamma,\theta}(M_1, \ldots, M_{n-1}, M_X) \subseteq M_S$ and let M_{LCS} be the largest complete solution of the same inequality.*

If $\bullet_{\Gamma,\theta}(M_1, \ldots, M_{n-1}, M_{LS}) = M_S$ then M_{LS} is the largest solution of the FSM equation $\bullet_{\Gamma,\theta}(M_1, \ldots, M_{n-1}, M_X) = M_S$.

If $\bullet_{\Gamma,\theta}(M_1, \ldots, M_{n-1}, M_{LCS}) = M_S$ then M_{LCS} is the largest complete solution of the FSM equation $\bullet_{\Gamma,\theta}(M_1, \ldots, M_{n-1}, M_X) = M_S$.

If $\bullet_{\Gamma,\theta}(M_1, \ldots, M_{n-1}, M_{LS}) \neq M_S$, then the FSM equation has no solution and therefore no complete solution.

It may happen that the largest solution of an FSM equation is not a complete solution.

3 The largest solution over the largest alphabet

In the previous section we assumed that the alphabets of the unknown FSM are given. However, when there is no solution over the given set of alphabets a solution may exist over a larger set of input alphabets, i.e., for another network topology. In this section, we show that there exists the largest set of input alphabets such that an equation is solvable if and only if it is solvable over the Cartesian product of these alphabets. The idea behind deriving such a set is as follows. The set of input alphabets is selected as large as possible while preserving the normal form of the network. Therefore, all the alphabets of the set Γ which are not output alphabets of the unknown component are input alphabets of the unknown component, in order to comply with 2. In other words, each alphabet of the set Γ is an input or an output alphabet of the unknown. We then derive a largest solution $M_{LS,\Gamma}$ to the inequality $\bullet_{\Gamma,\theta}(M_1, \ldots, M_{n-1}, M_X) \subseteq M_S$ over the set Γ of alphabets.

The procedure of deriving the FSM $M_{LS,\Gamma}$ includes the following steps.

Step 1 Derive the automaton $A = A(M_1)_{\uparrow\Gamma} \cap \cdots \cap A(M_{n-1})_{\uparrow\Gamma} \cap A(M_S)_{\uparrow\Gamma}$.

Step 2 Derive an FSM M as follows.

1. FSM M has a transition $(s, x_1 \ldots x_k, y_1 \ldots y_t, s')$, where $x_1 \ldots x_k$ is an item of the Cartesian product of the input alphabets (of the unknown component FSM) and $y_1 \ldots y_t$ is an item of the Cartesian product of the output alphabets (of the unknown component FSM), if the automaton A has a transition $(s, x_1 \ldots x_k, y_1 \ldots y_t, s')$.
2. FSM M has a transition $(s, x_1 \ldots x_k, y_1 \ldots y_t, \textit{DNC})$ if there is a component FSM $M_i, i = 1, \ldots, n-1$, which has no transition from the state corresponding to s under the projection of $x_1 \ldots x_k$ and $y_1 \ldots y_t$ onto the alphabets of this component FSM.
3. FSM M has a transition $\textit{DNC}, x_1 \ldots x_k, y_1 \ldots y_t, \textit{DNC})$ for all pairs $x_1 \ldots x_k, y_1 \ldots y_t$.

Step 3 Derive the FSM $M_{LS,\Gamma}$ as the reduced and observable representation of the largest complete submachine of FSM M.

The complexity of deriving $M_{LS,\Gamma}$ is polynomial as the determinization operator is not used.

Theorem 3.1 *If $\bullet_{\Gamma,\theta}(M_1, \ldots, M_{n-1}, M_{LS,\Gamma}) = M_S$, then the equation is solvable at least over the set Γ of alphabets. Otherwise, the equation has no solution over any alphabet.*

Theorem 3.2 *Assume that* $\bullet_{\Gamma,\theta}(M_1, \ldots, M_{n-1}, M_{LS,\Gamma}) = M_S$. *Given a set* $\rho \subseteq \Gamma$ *of input and output alphabets and an FSM* M_B *over the set* ρ *of alphabets that is a solution of the equation, then the FSM obtained from the automaton* $A(M_B)_{\uparrow\Gamma}$*is a reduction of the FSM* $M_{LS,\Gamma}$.

Thus, according to Th. 3.2, given the solution $M_{LS,\Gamma}$ and a set of alphabets $\rho \subseteq \Gamma$ over which to compute the unknown component FSM, a largest solution $M_{LS,\rho}$ over ρ (if it exists) can be extracted from $M_{LS,\Gamma}$ using the following steps.

Step 1 Derive the largest solution $M_{LS,\Gamma}$ over the set Γ of alphabets.

Step 2 Derive an FSM M_ρ from $M_{LS,\Gamma}$ as follows.

1. For each state and each absent input/output pair (over the set Γ) add a transition to the designated *Fail* state.
2. Add to the *Fail* state a transition to itself for each input/output pair.
3. For each transition, erase each label that corresponds to an alphabet that is not in ρ.

Step 3 Derive the automaton $A(M_\rho)$ and determinize it. Then delete from the obtained deterministic automaton each subset of states that contains the *Fail* state and convert the resulting automaton into an FSM $M_\rho^{'}$.

Step 4 $M_{LCS,\rho}$, i.e., the largest complete solution of the equation over the set of alphabets $\rho \subseteq \Gamma$, is the largest complete submachine of $M_\rho^{'}$ (if it exists).

Example 3.1 *Fig. 1 shows the steps of the algorithm: (a) shows the topology of the problem with the FSM* M_A *and* M_C *that compose the context; (b) shows the FSM conponent* M_A*; (c) shows the FSM conponent* M_C*; (d) shows the FSM specification* M_S*; (e) shows the automaton of the product* $\bullet_\Gamma(A, C, S)$*; (f) shows* $M_{LS,\Gamma}$ *the largest FSM solution over the largest alphabet* Γ*; (g) shows the topology of the problem with the solution* $M_{LS,\Gamma}$ *depending over the largest alphabet* Γ*; (h) shows the projection of the solution* $M_{LS,\Gamma}$ *over the alphabet* ρ *over which the original unknown is defined; (i) shows the determination of the projection of* $M_{LS,\Gamma}$ *to the alphabet* ρ*; (l) shows the final solution* $M_{LS,\rho}$.

4 Conclusions and Work in Progress

The new procedure to compute the largest solution of a synchronous FSM equation is leading also to some new interesting equisolvability results that we will report when the investigation will be completed. In particular, if all component FSMs are complete and deterministic and the specification is also described by a complete and deterministic FSM, we can restrict the largest set of input alphabets of the unknown over which the solvability of the equation should be checked. The reason is that in this case we can delete from the set of input alphabets of the unknown each output alphabet of another component FSM that corresponds to some external output. Given such an alphabet X_i, consider a transition in M_{LS} from state s under input/output pair $x_1 \ldots x_k / y_1 \ldots y_t$ and let an output depend on $x_1 \ldots x_{i-1}$. As the specification FSM and all other component FSMs are deterministic, there exists exactly one value x_i such that the transition under $x_1 \ldots x_k / y_1 \ldots y_t$ does not lead to the *DNC* state. Correspondingly if we delete a component from all tuples we will still get a complete FSM as a solution to the equation.

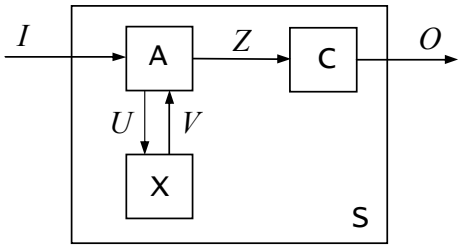
5 Acknowledgments

The first and the second authors gratefully acknowledge the support of grants by the Russian Fund of Basic Research, in particular, RFBR-NSC Grants 06-08-89500 and 07-08-12243. The third author gratefully acknowledges the support of the projects FP6-2005-IST-5-033709 (VERTIGO) and FP7-2007-IST-1-217069 (COCONUT). All the authors gratefully acknowledge the support of the NATO Collaborative Linkage Grants No 971217, 979698 and 982314.

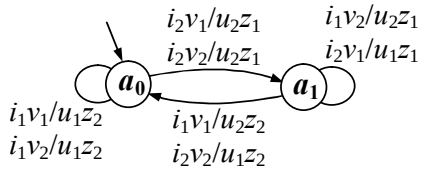
References

[1] M. Di Benedetto, A. Sangiovanni-Vincentelli, and T. Villa. Model Matching for Finite State Machines. *IEEE Transactions on Automatic Control*, 46(11):1726–1743, December 2001.

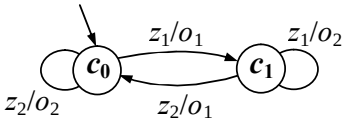
[2] J.R. Burch, D. Dill, E. Wolf, and G. DeMicheli. Modelling hierarchical combinational circuits. In *The Proceedings of the International Conference on Computer-Aided Design*, pages 612–617, November 1993.


I
A
Z
C
O
U
V
X
S

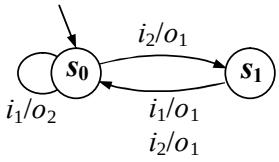
a)


i_2v_1/u_2z_1
i_2v_2/u_2z_1
i_1v_2/u_2z_1
i_2v_1/u_1z_1
a_0
a_1
i_1v_1/u_1z_2
i_1v_2/u_1z_2
i_1v_1/u_2z_2
i_2v_2/u_1z_2

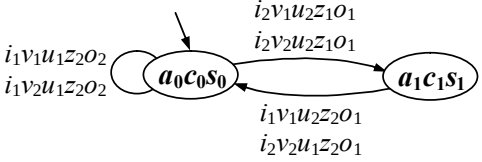
b)


z_1/o_1
z_1/o_2
c_0
c_1
z_2/o_2
z_2/o_1

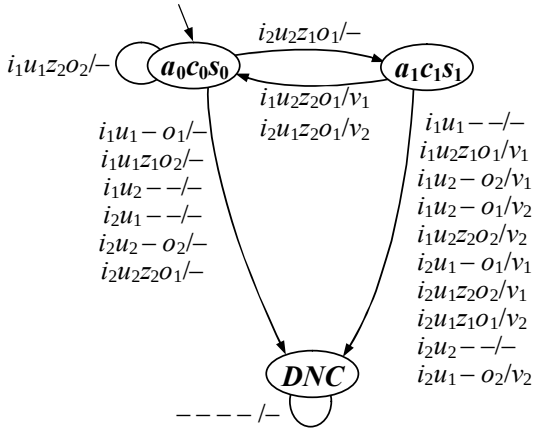
c)


i_2/o_1
s_0
s_1
i_1/o_2
i_1/o_1
i_2/o_1

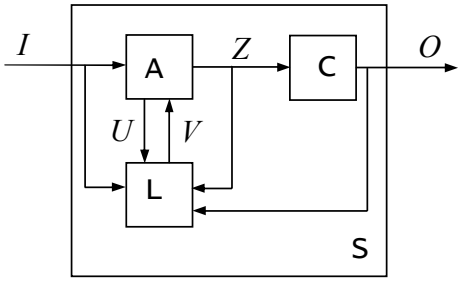
d)


$i_2v_1u_2z_1o_1$
$i_2v_2u_2z_1o_1$
$i_1v_1u_1z_2o_2$
$i_1v_2u_1z_2o_2$
$a_0c_0s_0$
$a_1c_1s_1$
$i_1v_1u_2z_2o_1$
$i_2v_2u_1z_2o_1$

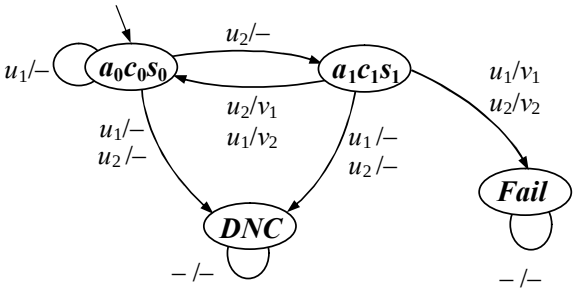
e)


$i_2u_2z_1o_1/-$
$i_1u_1z_2o_2/-$
$a_0c_0s_0$
$a_1c_1s_1$
$i_1u_2z_2o_1/v_1$
$i_2u_1z_2o_1/v_2$
$i_1u_1-o_1/-$
$i_1u_1z_1o_2/-$
$i_1u_2--/-$
$i_2u_1--/-$
$i_2u_2-o_2/-$
$i_2u_2z_2o_1/-$
$i_1u_1--/-$
$i_1u_2z_1o_1/v_1$
$i_1u_2-o_2/v_1$
$i_1u_2-o_1/v_2$
$i_1u_2z_2o_2/v_2$
$i_2u_1-o_1/v_1$
$i_2u_1z_2o_2/v_1$
$i_2u_1z_1o_1/v_2$
$i_2u_2--/-$
$i_2u_1-o_2/v_2$
DNC
$----/-$

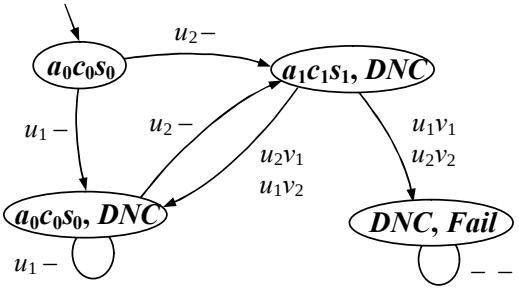
f)


I
A
Z
C
O
U
V
L
S

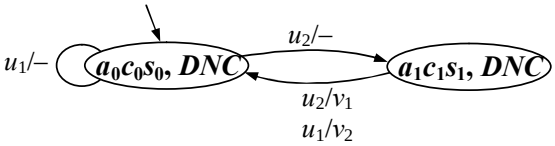
g)


u2/–
u1/–
a0c0s0
a1c1s1
u1/v1
u2/v2
u2/v1
u1/v2
u1/–
u2/–
u1/–
u2/–
Fail
DNC
– /–
– /–

h)


u2 –
a0c0s0
a1c1s1, DNC
u1 –
u2 –
u1v1
u2v2
u2v1
u1v2
a0c0s0, DNC
DNC, Fail
u1 –
– –

i)


u2/–
u1/–
a0c0s0, DNC
a1c1s1, DNC
u2/v1
u1/v2

l)

[3] T. Kam, T. Villa, R. Brayton, and A. Sangiovanni-Vincentelli. *Synthesis of FSMs: functional optimization*. Kluwer Academic Publishers, Boston, 1997.

[4] S. Khatri, A. Narayan, S. Krishnan, K. McMillan, R. Brayton, and A. Sangiovanni-Vincentelli. Engineering change in a non-deterministic FSM setting. In *The Proceedings of the Design Automation Conference*, pages 451–456, June 1996.

[5] S.C. Krishnan. *ω-Automata, Games and Synthesis*. PhD thesis, EECS Department, University of California, Berkeley, 1998. Tech. Report No. UCB/ERL M98/30.

[6] Yiu-Chung (Freddy) Mang. *Games in Open Systems Verification and Synthesis*. PhD thesis, University of California, Berkeley, May 2002.

[7] A. Mishchenko, R. Brayton, J.-H. Jiang, T. Villa, and N. Yevtushenko. Efficient solution of language equations using partitioned representations. In *The Proceedings of the Design, Automation and Test in Europe Conference*, volume 01, pages 418–423, March 2005.

[8] A. Petrenko, N. Yevtushenko, and R. Dssouli. Testing strategies for communicating finite state machines. In T. Mizuno, T. Higashino, and N. Shiratori, editors, *IFIP WG 6.1 International Workshop on Protocol Test Systems (7th : 1994 : Tokyo, Japan)*, pages 193–208. Chapman & Hall, 1995.

[9] A. Petrenko, N. Yevtushenko, A. Lebedev, and A. Das. Non-deterministic state machines in protocol conformance testing. In O. Rafiq, editor, *IFIP TC6/WG6.1 International Workshop on Protocol Test Systems (6th : 1993 : Pau, France)*, pages 363–378. North-Holland, 1994.

[10] Maria Vetrova. *Designing and Testing FSM compensators*. PhD thesis, Tomsk State University, Russia, 2004. (in Russian).

[11] T. Villa, T. Kam, R. Brayton, and A. Sangiovanni-Vincentelli. *Synthesis of FSMs: logic optimization*. Kluwer Academic Publishers, Boston, 1997.

[12] T. Villa, N. Yevtushenko, and S. Zharikova. Characterization of progressive solutions of a synchronous FSM equation. In *Vestnik, 278, Series Physics*, pages 129–133, September 2003. (In Russian).

[13] Y. Watanabe and R.K. Brayton. The maximum set of permissible behaviors for FSM networks. In *IEEE International Conference on Computer-Aided Design*, pages 316–320, November 1993.

[14] Elizabeth Wolf. *Hierarchical Models of Synchronous Circuits for Formal Verification and Substitution*. PhD thesis, Stanford University, September 1995. Tech. Report No. CS-TR-95-1557.

[15] N. Yevtushenko, T. Villa, R. Brayton, A. Petrenko, and A. Sangiovanni-Vincentelli. Solution of synchronous language equations for logic synthesis. In *The Biannual 4th Russian Conference with Foreign Participation on Computer-Aided Technologies in Applied Mathematics*, September 2002, http://www.parades.rm.cnr.it/ villa/articoli/ps/tomsk2002.ps.

[16] N. Yevtushenko, T. Villa, R. Brayton, A. Petrenko, and A. Sangiovanni-Vincentelli. Equisolvability of series vs. controller's topology in synchronous language equations. In *The Proceedings of the Design, Automation and Test in Europe Conference*, pages 1154–1155, March 2003. Extended abstract.

[17] N. Yevtushenko, T. Villa, R. Brayton, A. Petrenko, and A. Sangiovanni-Vincentelli. Sequential synthesis by language equation solving. Technical report, Tech. Rep. No. UCB/ERL M03/9, Berkeley, CA, April 2003, http://www.parades.rm.cnr.it/ villa/articoli/ps/TR-UCB_ERL-M03_9.ps.

[18] N. Yevtushenko, T. Villa, R. Brayton, A. Petrenko, and A. Sangiovanni-Vincentelli. Compositionally progressive solutions of synchronous FSM equations. *Discrete Event Dynamic Systems*, 18(1):51–89, March 2008.

[19] N.V. Yevtushenko and A.Y. Matrosova. Design of testable automaton networks. *Automatic and Remote Control*, 52(3, pt.2):416–423, March 1991.

[20] N.V. Yevtushenko and A.Y. Matrosova. Synthesis of checking sequences for automaton networks. *Automatic Control and Computer Sciences*, 25(2):1–4, 1991.

[21] S. Zharikova. Digital circuits optimization through solving a system of FSM equations. In *Vestnik TSU, Tomsk, N. 1*, pages 255–259, 2002. (In Russian).