EQUATIONS over AUTOMATA/FSMs
SOLVING A SYNCHRONOUS AUTOMATA/FSM EQUATION

Nina Yevtushenko
Tomsk State University, Russia
May, 2, 2011
Outline

- Equations over automata and FSMs
- Applications of automata/FSM equation solving
- Solving a synchronous equation over FSMs
  - solving an equation for a tail component of series composition
  - solving an equation for a head component of series composition
  - language approach for solving FSM equation
  - deriving a largest solution over the largest alphabet
  - solving FSM equation based on DNC input sequences
- Particular solutions to a synchronous FSM equation
  - largest complete solution
  - largest Moore solution
  - compositionally progressive solution
- Optimization of logic circuits and systems of FSM equations
Problem statement

- What is the mathematical model for $X$, Context and Spec?

- What is the composition operator $@$?

- What is the conformance relation $\sim$?

Solving the equation

Context $@$ $X \sim$ Specification
Mathematical machines

- Automata
- Finite State machines
- Petri Nets
- $\omega$-automata
- ???

All mathematical machines are counterparts for corresponding languages

SOLUTION Solve the equation over languages
Composition operators

- Synchronous composition (•) corresponds to instantaneous communication of systems
- Parallel composition (◊) corresponds to asynchronous communication allowing arbitrary delays between communication events
Conformance relations

Language is a set of strings

\( \subseteq \) Language containment

\( L(\text{Context}) \bullet X \subseteq L(\text{Spec}) \)

\( L(\text{Context}) \triangledown X \subseteq L(\text{Spec}) \)

\( = \) Language equality

\( L(\text{Context}) \bullet X = L(\text{Spec}) \)

\( L(\text{Context}) \triangledown X = L(\text{Spec}) \)
Application to discrete event system optimization

1. A system cannot be designed in optimal way
2. A system has the modular structure and is optimized iteratively in order to:
   a) simplify modules
   b) fault tolerance
   c) decrease the number of communication lines
   d) ???

The only solution for complex systems: 
**iterative optimization**
Discrete event system optimization

- The system is the composition of automata (or FSMs)
- Optimize module $A_j$
- Collapse all other modules into automaton/FSM Context
- Solve the equation Context @ $X = Spec$
- Determine an optimal solution
Application to the design of a plant controller (model matching)

How design a controller such that its composition with the plant meets the specification?

Context @ X = Spec

Diagram: Controller to Traffic lights with Smart traffic Lights
Application to the design of a protocol converter

Two protocol implementations cannot communicate according to non-matching input and output sets

Decision: to derive a converter
Application to testing modules of a discrete event system

**Problem** How to test a component FSM $A_j$ when no access to its inputs and outputs is granted

**Solution** To derive tests for an isolated FSM $A_j$ and transmit them to external inputs and outputs

**Another problem** Not each test derived for an isolated $A_j$ can be transmitted to external inputs and outputs

**Reason** Equivalence relation is too strong when deriving tests for a component FSM $A_j$

**Question** What is the relation that can be tested for FSM $A_j$ when the latter is tested in context?
Solve the equation

\[ \text{Context} \at \ X = \text{Spec} \]

A **general solution** to the equation defines the **external equivalence** relation for \( A_j \) and shows how precisely \( A_j \) can be tested in context.

All the tests that are derived based on a general solution can be transmitted to external inputs and outputs.
Application to the design of winning strategies for logic games

- Player_1 plays in the best way
- Winning states are labeled with 1
- **Question** What is a strategy of Player_2 in order to reach a winning state first and vice versa?
Application to cryptography with a public key

Ciphering

Alice and Bob

Verona University
Solving an FSM equation for a binary serial synchronous composition

Questions
1) Which FSM can replace the tail component FSM?

2) Which FSM can replace the head component FSM?

Binary synchronous composition

How to describe the complete flexibility of each component FSM?
Optimizing the tail component FSM of series composition

- Erase input labels in the flow diagram of the head component FSM and derive an automaton that represents all possible output sequences of the head component (determinize if necessary)
- For each transition, add all outputs as output symbols
- Derive the intersection of the obtained FSM with the tail component: obtain FSM $B'$ that generally is a partial deterministic FSM

**Theorem** An FSM $C$ can replace the tail FSM in the serial composition iff $C$ is quasi-equivalent to FSM $B'$

! If the automaton representing output sequences of the head component is non-deterministic then $B'$ can be non-deterministic but is equivalent to a partial deterministic FSM
Example of optimizing the tail component FSM

Component FSMs

The head component FSM $A$

The tail component FSM $B$

Non-deterministic automaton representing output sequences of $A$

When determinizing there are states \{1\}, \{2\}, \{2,3\} and \{4,5\}

States \{1\} and \{4,5\} are equivalent

a) Observable FSM $A^{out}$ with input sequences representing output sequences of FSM $A$

b) $B' = A^{out} \cap B$

States $a1$, $a2$, $a3$ and states $b2$, $b3$, $c1$ of $B'$ are compatible and can be merged

FSM $C$ s.t.

$A \circ B \equiv A \circ C$
Optimizing the head component FSM of series composition

Two input sequences of the tail component FSM $\mathcal{B}$ are $\mathcal{B}$-equivalent if FSM $\mathcal{B}$ has the same output response to both input sequences.

Let $L(A)$ be the largest solution to FSM equation

Binary synchronous composition

How to describe the complete flexibility of the head component FSM?
The set of states of FSM $\mathcal{B}^{ext}$ is a subset of the set of pairs $(b, b')$ with the initial state $(b_1, b_1)$, input and output alphabets coincide with the input alphabet of the FSM $\mathcal{B}$. There is a transition $((b_j, b_k), u, u', (b_j', b_k'))$ iff there exists an output $o$ s.t. FSM $\mathcal{B}$ has transitions $(b_j, u, o, b_j')$ and $(b_k, u', o, b_k')$. 
Example of optimizing the head component FSM

Component FSMs

FSM A

FSM B

FSM $\mathcal{B}$ has the same output response to input sequences $u_1 u_1 u_1$ and $u_2 u_1 u_2$, and thus, $u_2 u_1 u_2$ is in the set of output responses of FSM $\mathcal{B}^{ext}$ to $u_1 u_1 u_1$ and vice versa.

$L(A) = A \cdot \mathcal{B}^{ext}$
Question How to solve the equation \( \chi \cdot C \equiv S \)?

Solution Since the behavior of an FSM is the set of finite strings the equation can be solved as the language equation

\[ \chi \cdot L_C \equiv L_S \]
Synchronous composition of languages

\[ io \in A \cdot B \Leftrightarrow \exists u \in U \left[ iu \in A \land uo \in B \right] \]

The language \( L_A \cdot L_B = \left[ \left( L_A \right)^{\uparrow O} \cap \left( L_B \right)^{\uparrow I} \right]^{\downarrow I \times O} \) is called the synchronous composition of languages \( L_A \) and \( L_B \).
Solving synchronous language inequality *Context* • *X ⊆ Spec*

- Language *B* is a *solution* to the inequality if *Context* • *B ⊆ Spec*
- A solution *Largest* is the *largest* solution to the inequality if each solution is a subset of *Largest*
- Language *B* is a *trivial* solution if *B* = ∅

Inequality *Context* • *X ⊆ Spec* is *solvable* and has the *largest solution*

Any subset of the largest solution is a solution to inequality
The largest solution to the synchronous language inequality

\[ \alpha \in (O \times U)^* \text{ is a solution to } Context \cdot X \subseteq Spec \]

\[ \Leftrightarrow \{\alpha\} \cdot Context \subseteq Spec \]
\[ \Leftrightarrow [\{\alpha\} \uparrow_I \cap Context \uparrow_O] \downarrow_{I \times O} \cap Spec = \emptyset \]
\[ \Leftrightarrow \{\alpha\} \uparrow_I \cap Context \uparrow_O \cap Spec \uparrow_U = \emptyset \]
\[ \Leftrightarrow \alpha \notin (Context \uparrow_O \cap Spec \uparrow_U) \downarrow_{O \times U} \]
\[ \Leftrightarrow \alpha \notin (Context \cdot \lceil Spec \rceil) \]
\[ \Leftrightarrow \alpha \in \lceil (Context \cdot \lceil Spec \rceil) \rceil \]

\[ \lceil (Context \cdot \lceil Spec \rceil) \rceil \text{ – the largest solution} \]

\[ \lceil Spec \rceil \text{ is the complement of the language } Spec \]
Solving synchronous language equation $\textit{Context} \bullet X = \textit{Spec}$

Let $\textit{Largest} = \lceil (\textit{Context} \bullet \lfloor \textit{Spec} \rfloor) \rceil$ be the largest solution to the inequality

$\textit{Context} \bullet X \subseteq \textit{Spec}$

**Theorem** The equation $\textit{Context} \bullet X = \textit{Spec}$

is solvable if and only if $\textit{Context} \bullet \textit{Largest} = \textit{Spec}$

Regular language equations can be
Finite automata

\[ S = \langle S, A, \delta_s, s_0, F_S \rangle \]

**finite automaton**

- \( S \) - finite nonempty set of *states* with the initial state \( s_0 \)
- \( F_S \) - subset of *final* (or *accepting*) states
- \( A \) - alphabet of *actions*
- \( \delta_s \subseteq S \times A \times S \) - *transition relation*

Flow diagram:

- A flow diagram showing a transition from state \( s_0 \) to state \( s_1 \) on input \( a \).
Projection and lifting operators over automata

- **Projection** ($\downarrow$)
  Given FA $F$ that accepts language $L$ over $I \times U$, FA $F_{\downarrow I}$ that accepts language $L_{\downarrow I}$ over $I$ is obtained by replacing each edge $(s, iu, s')$ in $F$ by the edge $(s, i, s')$

- **Lifting** ($\uparrow$)
  Given FA $F$ that accepts language $L$ over $I \times U$, FA $F_{\uparrow I}$ that accepts language $L_{\uparrow I}$ over $I$ is obtained by replacing each edge $(s, iu, s')$ in $F$ by the edge $(s, i, s')$

---

**Equation:**

$$\text{Context} \cdot X \subseteq \text{Spec}$$

**Largest solution:**

$$\left\lfloor \left( \text{Context} \cdot \downarrow \text{Spec} \right) \right\rfloor = \left\lfloor \left( \text{Context}_{\uparrow O} \cap \left( \downarrow \text{Spec}_{\uparrow U} \right) \right) \downarrow U \times O \right\rfloor$$
Synchronous composition of FSMs (definition)

The synchronous composition \( A \cdot \beta \) of two initialized FSMs \( A \) and \( \beta \) is the initialized connected reduced observable FSM with the language

\[
L_A \cdot L_\beta = \left((L_A)^\uparrow_{\mathcal{R} \times \mathcal{O}_2} \cap (L_\beta)^\uparrow_{\mathcal{I} \times \mathcal{O}_1}\downarrow_{\mathcal{I} \times \mathcal{O}_1 \times \mathcal{R} \times \mathcal{O}_2}\right)
\]

**Proposition** If \( L_A \) and \( L_\beta \) are FSM languages then

\[
\left((L_A)^\uparrow_{\mathcal{R} \times \mathcal{O}_2} \cap (L_\beta)^\uparrow_{\mathcal{I} \times \mathcal{O}_1}\downarrow_{\mathcal{I} \times \mathcal{O}_1 \times \mathcal{R} \times \mathcal{O}_2}\right)
\]

is an FSM language.
Synchronous composition of FSMs (example)

Structural FSM

Verona University
Solving synchronous automata/FSM equations

FSM inequality: $\mathcal{C} \cdot X \leq \mathcal{L}$ ($\leq$ - reduction relation)

FSM equation: $\mathcal{C} \cdot X \equiv \mathcal{L}$ ($\equiv$ - trace equivalence relation)

To solve an FSM equation the corresponding equation over regular languages is solved. All operators are performed over finite automata.

The resulting automaton is converted to an FSM.
Solving FSM equations (cont-d)

Solve an FSM equation as a language equation
Given the largest solution, extract the largest FSM language

An observable reduced FSM with this language is the largest solution to an FSM inequality
All and only the reductions of the largest solution to an FSM inequality are FSM solutions to the inequality

! Generally, the largest solution is a partial nondeterministic FSM
Solving a synchronous FSM equation

If for the largest solution $M$ to the FSM inequality
\[ C \cdot X \leq S \]
it holds
\[ C \cdot M \equiv S \]
then FSM $M$ is a largest solution to the FSM equation
\[ C \cdot X \equiv S \]

Each solution to the FSM equation is a reduction of the largest solution

! Not each complete reduction of a largest solution is a solution to the FSM equation

! The complexity is exponential in general case
Other approaches to FSM equation solving

**Question** How to simplify the construction?

**Solution**
- Consider special topologies (controller topology), special classes of FSMs (needs additional research) etc
- To find subset of the largest solution
- More?
Solving a synchronous FSM equation over the largest alphabet

FSM $\beta$ is complete and deterministic

Suppose that we do not care about the input alphabets of the FSM $X$

The equation $C \bullet X \equiv S$ can be solved over input alphabet $I_1 \times I_2 \times O_2 \times V$
Solving the equation over the largest alphabet

**Input:** Complete deterministic FSMs \( C = \langle C, I_1 \times V, U \times O_1, T, c_0 \rangle \) and \( S = \langle S, I_1 \times I_2, O_1 \times O_2, T, s_0 \rangle \)

**Output:** The largest solution \( l_{\text{max}} \) of the FSM inequality \( C \cdot X \leq S \) with input alphabet \( I_1 \times I_2 \times O_2 \times V \) and output alphabet \( O_1 \times U \)

**Step 1.** Lift FSM \( S \) to \( V \times U \) and FSM \( C \) to \( I_1 \times O_1 \)

Derive the intersection of the lifted FSMs

**Step 2.** FSM \( l_{\text{max}} \) has the same set of states as the above intersection with the designated state \( DNC \)

- There is a transition \( (c_s, i_1, o_2, v, o_1, u, c' s') \) in the FSM \( l_{\text{max}} \) iff there is such a transition in the intersection of the lifted FSMs
- There is a transition \( (c_s, i_1, o_2, v, o_1, u, DNC) \) in the FSM \( l_{\text{max}} \) iff at state \( c \) of FSM \( C \) there is no transition under Input/Output pair \( i_2 u / o_2 v \)
- There is a self-loop at state \( DNC \) labeled with all Input/Output pairs in the FSM \( l_{\text{max}} \)

**Theorem** Let the FSM \( l_{\text{max}} \) be the largest solution to the inequality \( C \cdot X \leq S \) over the input alphabet \( I_1 \times I_2 \times O_2 \times V \)

If the composition \( C \cdot l_{\text{max}} \) is equivalent to FSM \( S \) then \( l_{\text{max}} \) is the largest solution to the equation \( C \cdot X \equiv S \) over the largest alphabet (blue dashed lines)

Otherwise, the equation \( C \cdot X \equiv S \) is not solvable over any input alphabet
Example of solving an FSM equation over the largest alphabet

The largest solution to the equation $\mathcal{C} \cdot \chi \cong \mathcal{S}$
(Missing transitions go to DNC state)
Comparison of two algorithms

Comparing two algorithms for equation solving

- When deriving a solution over the largest alphabet the projection operator is not used
  - However
  - The algorithm can be applied only for deterministic complete FSMs
  - When deriving a solution over a smaller input alphabet the projection of the largest solution over the largest alphabet should be taken (exponential complexity)

- Language based algorithm for solving equations can be used for all kinds of FSMs
  - However
  - Has exponential complexity
Why equation solving is not widely used

There are two problems
Generally
- Equation solving algorithms have exponential complexity
- Even a complete submachine of the largest solution is not necessarily a solution to the corresponding FSM equation (we still lack such characterization of all possible solutions)

One of possible solutions To use a part of the largest solution that can be derived based on don’t care (DNC) input sequences
Solving an equation based on input DNCs

**Input:** Complete deterministic FSMs $A$, $B$, $S = A \cdot B$

**Output:** Partial FSM $B'$

**Step 1.** For each state $b \in B$ s.t. there exists $(a_j, b) \in S$ for some $a_j \in A$ determine the subset of DNC symbols of $U$ which cannot appear as outputs of $A$ at state $a_j$.

The set of DNC inputs at state $b \in B$ is the intersection of such subsets over all $(a_j, b) \in S$.

**Step 2.** When deriving partial FSM $B'$ for each state $b \in B$ delete transitions under all DNC inputs.

**Theorem** Each complete FSM that is quasi-equivalent to $B'$ can replace FSM $B$ in the composition without changing the composition behavior.

Cannot be done based of FSM $A$ only, since FSM $B$ also influences the set of DNC inputs. If the behavior of $B$ is changed the set of DNC inputs is also changed.

Verona University
Solving an equation based on input DNCs (example)

FSMs $A$, $B$, $S = A \cdot B$

Partial FSM $B'$ and an FSM with a single state that is quasi-equivalent to $B'$, i.e., can replace the component FSM $B$ without changing the composition behavior.

$u_3$ is a DNC input for $b_0$
$u_1$ is a DNC input for $b_1$
Restricted solutions to FSM equation

- Complete solutions
- Moore solutions
- Compositionally progressive solutions
- ???

\[ C \cdot X \cong S \]
Complete solutions

**Equation** \( C \cdot x \cong S \)
- Extract from the largest solution \( M \) its maximal complete submachine \( \text{Com}_M \)

**Proposition** Each complete solution to the equation \( C \cdot x \cong S \) is a reduction of \( \text{Com}_M \)

! In general, not each complete reduction of \( \text{Com}_M \) is a solution to the equation
Largest complete submachine

How to derive the largest complete submachine $COM_\mathcal{S}$ of initialized FSM $\mathcal{S}$

Iteratively delete states where at least one input is undefined

If the initial state is deleted then $\mathcal{S}$ has no complete submachine

Otherwise, what is left is the largest complete submachine of $\mathcal{S}$
Moore solutions

**Equation** $C \cdot X \cong S$

**Theorem**
- If $C$ and $S$ are initialized complete deterministic FSMs and $C$ is a Moore FSM then each complete reduction of the largest solution to the equation $C \cdot X \cong S$ is a solution to the equation
- If $C$ and $S$ are initialized complete deterministic FSMs then each complete deterministic Moore reduction of the largest solution to the equation $C \cdot X \cong S$ is a solution to the equation

A solution to the equation can replace a corresponding component FSM without changing the composition behavior
Deriving the largest Moore solution

How to derive the largest Moore submachine $Moore_S$ of initialized FSM $S$?

Iteratively

- For each state, define a subset of outputs which are outputs for each input $(i, u) \in I_1 \times V$ at the state
- Delete states where such a subset is empty or a transition under some input is undefined

If the initial state is deleted then $S$ has no Moore submachine

Otherwise, what is left is the largest complete submachine $Moore_S$ of $S$
A solution is *compositionally progressive* if when combined with the context there are no deadlocks.

In order to derive a largest compositionally progressive solution two options can be used:

- Delete solution traces that induce deadlocks when combined with the context.

**Problem** the number of sequences is infinite.

Does the procedure terminate?

- Delete ‘bad’ states.

**Problem** state splitting.
Global optimization for component FSMs

The system is the composition of FSMs

Optimize component FSM $A_j$

- Collapse all other modules into FSM Context
- Solve the equation $\text{Context} \cdot X \equiv S$
- Extract an optimal solution from the largest solution to the equation

Optimizing a component
Problem when using the global optimization

**Problem** When collapsing component FSMs into a single FSM, the latter becomes huge.

**Solution** to use a local optimization (optimization only w.r.t. to neighbor component FSM.)
Optimizing a component FSM using a collection of equations

Consider a collection of equations w.r.t. neighbor FSMs

\[ X \cdot A_4 \equiv A_j \cdot A_4 \]

\[ \ldots \]

**Proposition** A solution to any equation of the collection can replace \( A_j \) without changing the collection of equations.
Systems of synchronous equations

Sometimes a component FSM cannot be optimized when using a collection of equations

**Solution** to use two circuits for representation
Using systems of FSM equations for the optimization

System of FSM equations

\[
\begin{align*}
X \cdot B & \equiv S_1 \\
X \cdot C & \equiv S_2
\end{align*}
\]

\[S_1 = A \cdot B \quad \text{and} \quad S_2 = A \cdot C\]

**Theorem** FSM \( l \) is a solution to the equation \( X \cdot (B \cdot C) \equiv S \) iff \( l \) is a solution to the system of equations.
Solving a system of FSM equations

System of FSM equations

\[
\begin{align*}
A_1 \cdot X & \equiv S_1 \\
A_2 \cdot X & \equiv S_2 \\
\vdots & \\
A_n \cdot X & \equiv S_n
\end{align*}
\]

Steps
- Derive the largest solution to each equation
- Intersect largest solutions

Complete solutions, Moore solutions, compositionally progressive solutions etc can be considered for a system of FSM equations.
Example

Largest solution to $X \cdot B \cong A \cdot B$

Largest solution to $X \cdot C \cong A \cdot C$
Example (cont-d)

Largest solutions to $X \cdot B \cong A \cdot B$ and $X \cdot C \cong A \cdot C$

The intersection of largest solutions

An FSM that can replace the component FSM $A$
Conclusions

An algebra for language/automata/FSM equations is proposed.

The complexity of solving automata/FSM equations for various composition topologies has been studied.

Some simplifications of the general procedure are proposed.

Restricted solutions (to FSM equations/systems of FSM equations) which are of theoretical/practical use are considered.
Open problems

- The complexity of deriving general solution is exponential

- We lack the complete characterization for the set of all solutions to an automata/FSM equation. It is shown that language features are insufficient for the complete solution characterization, save for restricted topologies

- Not each solution is of practical use

- FSM equation solving over other composition operators

- How to solve an equation $\text{Context} \bullet X \geq \text{Spec}$

- We lack an efficient method for solving synchronous equations over other mathematical machines such as Petri Nets, etc
References


Thanks for attention!