Web-based Tools for Finite State Machine Decomposition with Analysis of Information Flows

E. Fomina, A. Keevallik, M. Kruus, and A. Sudnitson

Department of Computer Engineering, TTU, Raja 15, 12618 Tallinn, Estonia, E-mail: alsu@cc.ttu.ee

ABSTRACT: Finite state machine is a convenient model for specification, analysis and synthesis of control part of electronic systems. This work focuses on a particular but a comprehensive problem of decomposition of FSMs. We are concerned with solving complex combinatorial tasks arising from the process of design. An approach to modelling of information flows in networks of FSMs is proposed. The educational aim of this work is to provide a basic theoretical background for design of discrete systems using opportunities of asynchronous mode of education via the Internet.

1 Introduction

FSMs have been widely used to express algorithms, communication protocols, digital systems, sequential logic circuits, and sequential logic cells. Decomposition of FSMs is a well-known and important problem in sequential circuit synthesis [1-3].

Driven by remarkable theorems of Claude E. Shannon, which motivate entropy as the measure of information content, we have been examining the entropy measures of search for approximate or indirect methods of evaluation information and information dependencies in FSMs. Our goal is to originate a quantitative theory of decomposition for FSMs based on the structural decomposition theory [4].

A substantial part of this work is the development of a user-friendly interactive system developed for WWW that assists designers to deepen basic concepts and notions in digital design and helps to design complex control devises. The system uses Java technology that represents a powerful tool for the development of platform-independent interactive software, which can be used on the WWW through Java-enabled Web browser.

This system under design provides remote distance interactive learning and supports various phases of the learning process. The modern information technologies have enabled education using synchronous and asynchronous tools [5]. In asynchronous mode students can have access to instructional material at any time and from any convenient location. Asynchronous learning networks provide in addition a network of people who can interact with each other using electronic connectivity tools to simulate the interactivity of physical presence. The architecture of the system under development allows using both modes of education.

2 Motivation and Preliminaries

Theoretical background of our system is "decomposition synthesis" approach. This is based on the FSM decomposition theory, which uses partition pair algebra proposed in [4]. The importance of this theory lies in the fact that it provides a direct link between algebraic relationships and physical realizations of machines. The mathematical foundation of this theory rest on an algebraization of the concept of "information" in a machine and supply the algebraic formalism necessary to study problems about the flow of this information in machines when they operate. The formal techniques are very closely related to modern algebra. It has, we believe after Hartmanis and Stearns [4], an abstract beauty combined with the challenge of physical interpretation and application. It falls squarely in the interdisciplinary area of applied algebra, which is a part of engineering mathematics.

This paper targets the very first step in the synthesis flow where the FSM characterizing the control part of the high-level representation is described in the form of a State Transition Graph (STG) and each state is represented in a symbolic form.

While we investigate mainly input-state dependencies, FSM may be treated generally as triplet $\langle S, C, d \rangle$, where $S=\{s_1, \dots, s_m\}$ is a set of states;
$C=\{x_1, \dots, x_l\}$ is a set of primary input symbolic variables;
d: $D(d) \rightarrow S$ is a next state function with domain $D(d)=D_1 \times \dots \times D_l \times S$ and codomain S. Here, D_i represents a set of values each x_i may assume.

We define a network of FSM as an abstract algebraic system. FSM network is a system $N=\langle X_N, B_N, R_N \rangle$, where C_N is a set of network input variables;
$B_N = \{A_i \mid i \in I = \{1, \dots, n\}\}$ is a set of state machines referred as component machines;
$R_N \subseteq B \times B$ is a relation of connection.

To describe the network more thoroughly, we use the set of internal symbolic variables of net $Z=\{z_i \mid z_i \in S_i, i \in I, i=\{1,\ldots,n\}\}$ and representation of relation of connection R_N as incidence matrix $|| r_{ij} ||$. r_{ij} =1 means *i*-th component FSM receives information from *j*-th component FSM.

Network state machines could be defined as

$A_i = < X_i , S_i , d_i >$, where

$X_i = Z_i \cup E_i$, $Z_i \subseteq Z$, $E_i \subseteq X_N$;

$Z_i = \{z_{ij} \mid r_{ij}=1$ in incidence matrix of relation $R_N\}$ is a set of internal inputs;

E_i is a set of external inputs, $E_i \subseteq X$;

d_i: $D(d_i) \rightarrow S_i$, is a transition function.

The decomposition proceeds from the set of partitions on the set of states, *S*, which are induced on FSM by a network that realizes prototype machine (a partition on the set of states, *S*, is a collection of disjoint subsets of states whose set union is *S*). The individual machines that make up the overall realization are referred to as sub-machines. Each sub-machine corresponds to a partition from the basic set.

A block of states in a partition effectively corresponds to a state in the sub-machine associated with that partition. All the states belonging to a single block in a submachine are given the same code in that submachine. Therefore, there is no way of distinguishing between two states belonging to a single block in a sub-machine without recourse to information from other sub-machines.

The concept of partition pairs is introduced to study how "ignorance spreads" or "information flows" runs through a sequential machine when it operates. The notion of partition pairs is based on the idea that the first partition of pair $\langle p_1, p_2 \rangle$ has enough information to calculate the second one. It is natural that for any partition p we can determine the $M(p)$ partition. The operator $M(p)$ gives the maximum front partition of partition pair. Informally speaking, for a given partition$p$, the partition $M(p)$ describes the least amount of information we must have about the present state of *A* to the next state (i.e., the block of p which contains the next state of *A*). If partition $M(p_i)$ is less or equal to multiplication of partitions from $\{p_j \mid r_{ij} = 1\}$ than it means that *i*-th sub-FSM receives enough information from sub-FSMs with which it is connected accordingly relation of connection to compute the next state.

3 Decomposition Software System

To implement the software system's architecture we should follow four main requirements [1] :

- Possibility to ran under various operating systems;
- Implementation of new modules without changing the rest of the system;
- Realizing a client server architecture;
- Using the same source to generate the printed and interactive worksheets to prevent inconsistency after modifications.

These requirements cause the use the applet concept of Java language. Java is the natural programming language of choice on the client side because of its flexibility of Graphic User Interface (GUI) design, convenient network programming, and platform independence. The last property is especially significant since it allows the same applet program to run on client computers of different platform.

Developed system includes building tutorial of FSM synthesis theory and additional useful information for working with client software. The advantage of the tutorial is interconnectedness among different topics and with other related tutorials, which is easy to implement on the WWW using the hypertext mark-up language. The tutorial contains many examples for students to study and compare.

The sequence of applets to understand the essence of decompositions is developed. Next, we discuss main of them from "informational" point of view.

In our reasoning, we proceed from information theoretic concepts, which are rationalized on the basis of algebraic structure theory of sequential machines [2]. In the following, we assume that the state lines of the FSM are modeled as Markov chain characterized by the stochastic matrix $(q_{ij})_{1 \le i, j \le m}$, where q_{ij} is the conditional probability of the FSM being in *j*-th state given that it was previously in *i*-th state. These probabilities, along with the steady state probability vector $(p_i)_{1 \le i \le m}$ (we suppose that all states are reachable) can be found using standard techniques for probabilistic analysis of FSMs [5].

Let $E = \{e_1, e_2, \ldots e_g\}$ be a complete set of events which may occur with the probabilities $p_1, p_2, \ldots , p_g$ In order to quantify the content of information, Shannon introduces the concept of entropy.

Entropy of *E* (denoted by *H*(*E*)) is given by:

$$H(E) = -\sum_{e \in E} p(e) \cdot \log_2 p(e) \qquad (1)$$

Depending on the specified sense of event, we can define several entropy measures, e.g. the entropy of FSM based on the state of occupation probabilities or based on the state transition probabilities. Reasoning similarly, we define the entropy of partition p as:

$$H(p) = -\sum_{B \in p} p(B) \cdot \log_2 p(B) \qquad (2)$$

where the probability of the block $B \subseteq S$ is defined as the cumulative occupation probability of the states in *B*.

Entropy of FSM network corresponding to the set of partitions, $N = \{p_1, p_2, \ldots , p_n\}$, is equal to

$$H(N) = \sum_{p \in N} H(p) \qquad (3)$$

The sequence of applets to understand the essence of decompositions is developed.

Applet 1 describes how partitions on a set can be "*multiplied*" and "*added*". These operations on partitions play a central role in the structure theory of FSM and form a basic link between machine concepts and algebra. The sum of two partitions p_1 and p_2 is the largest partition (the one with the most blocks) that is refined by both p_1 and p_2. The product of p_1 and p_2 is the smallest partition (the one with the fewest blocks) that refines both p_1 and p_2. A partition on the set of states of the FSM can be considered as a measure of information about the FSM. That it is why the multiplication of all partitions in the set must be zero partition in order to preserve all the information about the source FSM behavior in the network of FSMs defined by the partition set. The functionality of the prototype machine is maintained in the decomposed machine if the partitions associated with the decomposition are such that their product is the zero-partition on S (every block of partition consists exactly of one state).

Applet 2 exhibits formal correspondence to intuitive concept of a "subcomputation". We consider the concept of a homomorphism. Since a machine A can be used to realize its homomorphic image A', we can say informally that A' does a part or a subcomputation of the computation performed by A. From partition algebra point of view the concept of homomorphism relates to partitions with substitution property. We recall that if a partition p on the set of states of a machine A has the substitution property, than as long as we know the block of p which contains a given state of A, we can compute the block of p to which that state is transformed by any given input sequence. Intuitively we say that the "ignorance" about the given state (as specified by the partition p) does not spread as the machine operates [5].

Applet 3. The concept of partition pairs is more general than substitution property and is introduced to study how "ignorance spreads" or "information flows" through a sequential machine when it operates. If (p,p') is a *partition pair* on the FSM A than blocks of p are mapped into the blocks ofp' by A In other words, if we only know the block of p which contains the state of A, then we can compute for every input the block ofp' to which this state is transferred by A.

For partition pair $\langle p_i, p_j \rangle$ the conditional entropy is

$$H(p_j,p_i) = H(p_i \cdot p_j) - H(p_i) \qquad (3)$$

It is natural that for any partition p we can determine the $M(p)$ partition. The operator $M(p)$ gives the maximum front partition of partition pair. Informally speaking, for a given partition$p$, the partition $M(p)$ describes the least amount of information we must have about the present state of A to the next state (i.e., the block of p which contains the next state of A). Thus this partition gives precise meaning to our intuitive concept "how much do we have to know about the present state to compute … about the next state".

To calculate this partition we need to find the symbolic cover of the discrete function F_i: $D(d) \rightarrow p_i$. Given a FSM, we first assign one-hot codes to all states. Then symbolic minimization is applied to the one-hot coded machine using multi-valued logic minimization. The result is a symbolic cover, K_i, of the F_i. Each element of the symbolic cover is a symbolic prime implicant, that is a triplet $\langle b, B', B \rangle$ where B' is the set of states (block of partition $M(p)$) which transit to the next state contained in the same block B of partition p under input condition$b$. The number of prime implicates, $|K_i|$, is proportional to number of rows in the transition table of corresponding sub-machine.

Applet 4 performs construction of FSM network that realizes the prototype FSM. We consider FSM network as algebraic system $N = \langle B, I, O, g, F \rangle$ where B is a set of component FSMs, I is a set of inputs and O is a set of outputs, g is output function of network, $F \subseteq B \times B$ is a relation of connection of component FSMs of the network. We represent F as incidence matrix $|| r_{ij} ||$. $r_{ij} = 1$ means i-th component FSM receives information from j-th component FSM. Every FSM from B is in correspondence with chosen partition p_i. If partition $t = M(p_i)$ is less or equal to multiplication of partitions from $\{p_j \mid r_{ij} = 1\}$ than it means that i-th component FSM receives enough information from component FSMs with which it is connected accordingly F to compute the next state.

Our work proceeds from the fact that the principal NP-complete problem of FSM decomposition is searching of a set of partitions on the set of states of prototype FSM. As it was shown in [4], only such a set of partitions may be used for FSM decomposition. The lack of a methodology of searching of these partitions is substantial limitation of application of powerful algebraic decomposition theory in practice. We attempt to surmount this obstacle. Implementation of FSM in a device with the lack of external terminals appears very often in practice and has always been a problem for designers.

Applet 5 is devoted to choice of decomposition partition on the set of states of prototype FSM to meet a requirement on the number of inputs. Here we should emphasize the fact that the machine decomposition problem and the reduction of variable dependence are virtually identical concepts. This is NP-hard problem, and amounts to solving a face hypercube-embedding problem [1]. In spite of recent advances, computing a decision of this task remains prohibitive for FSM of practical complexity. In this applet we show how the input-state dependencies can be used to decrease the number of inputs. Theoretical foundation of our approach

is based on the new notion of partition with don't care's and its relation to pair algebra.

A Partition with Don't Care's (PDC) r of a set S is a collection of disjoint nonempty subsets of S. The disjoint subsets are called blocks of r and their set union is equal to $S_d \subseteq S$. The set difference $S \setminus S_d$ is Don't Care's area of the PDC and we can consider it as some distinguished (special) block b_c, which may be empty. The PDC ρ in reality defines a set of conventional partitions, denoted by $G(\tilde{n})$, generated by distributing the elements of distinguished block over the other blocks of the PDC and over new created blocks in all possible ways. The set of all PDC pairs on A is pair algebra on $\acute{A} \times \acute{A}$, where $\acute{A}$ is lattice of PDC of S. Thus, all the results that are derived about pair algebra [5] hold for PDC pairs on A.

The idea of the next two applets is to introduce additional "idle" states into the FSM in the hope to meet design constraints. The network of FSMs consists of components working alternatively in time, i.e. all components except one are suspended in one of extra state (the "wait" state). In [1] similar approach is called factorisation of the sequential state machines. This property gives opportunity to apply sleep mode operation (dynamic power management) for saving power consumption.

Applet 6 enables to decompose a prototype FSM into a set connected component FSMs with given constraints on the complexity of component FSMs (a number of inputs, outputs, states and rows in their transition tables) on the base of one partition on the set of states. The number of states of component FSM is equal to the number of states in corresponding block of partition π plus 1 (waits or idle state).

Applet 7 implements a method for FSM decomposition with outputs distributed among the component FSMs. A partition on the set of prototype FSM outputs is taken as primary design requirement.

4 Concluding Remarks

This work focuses on how new Web-based frameworks can enable research and development in synthesis of control-dominated discrete systems. The pedagogical basis on which our system is built is that students learn most quickly when they are presented with material that they can quickly use to solve design problems. Students can decompose a given FSM into a set of connected FSMs with given constraints on the complexity of component FSMs (a number of inputs, outputs, states and rows in their transition tables). The system assists to fulfil projects on design of digital devices.

The idea of using entropy based informational measures can be extended to other phases of logic synthesis also. Here we should emphasize the fact that the machine decomposition and the reduction of variable dependence are virtually identical concepts. To ensure that partition entropy is a good indicator of implementation complexity, experiments have been carried on hundreds of FSMs. They proved that the correlation between of decomposition partition p_i and the complexity of sub-FSM A_i (area) are very high (more than 0,95).

In future we plan to update the system to carry on an interactive Web-based decomposition synthesis at a higher level that considers both the controller and data path simultaneously. The synthesis system under development should not be only design automation software but it should be a research tool and educational system we will be able to use for further development of synthesis theory.

Acknowledgements

This work has been supported partially by the Estonian Science Foundation (under Grant 4876) and by the Ministry of Education in Thüringen, Germany (DILDIS project).

References

[1] P. Ashar, S.Devadas, and A.R.Newton, *Sequential Logic Synthesis.* Kluwer Academic Publishers, Boston, 1992.

[2] S. Baranov, *Logic Synthesis for Control Automata*, Kluwer Academic Publishers, Boston, 1994.

[3] G. De Micheli, *Synthesis and Optimization of Digital Circuits*. McGraw-Hill, Inc., New York, 1994.

[4] M. Dubin, T. Neese, and I. Moon, *Action Based Learning for Switching and Automata Theory*. Available: http://vlsi.colorado.edu/~mooni/N_ABLE/N_ABLE.html

[5] J. Hartmanis, R. E. Stearns, *Algebraic Structure Theory of Sequential Machines*, Prentice-Hall, Englewood Cliffs, New York, 1966.

[6] A Sudnitson, S. Devadze, A. Levenko. *Finite State Machine Decomposition*. Available: http://www.pld.ttu.ee/dildis/automata/applets.

[7] H-D. Wuttke, K. Henke, R. Peukert, "*Interned Based Education – An Experimental Environment for Various Educational Purposes"*. In Proc. *The IASTED International Conference Computers and advanced Technology in Education*, (Philadelphia, PA USA, May 1999), pp. 50-54.