

# On CAST.FSM Computation of Hierarchical Multi-Layer Networks of Automata

Michael Affenzeller, Franz Pichler, and Rudolf Mittelmann

Institute of Systems Science  
Systems Theory and Information Technology  
Johannes Kepler University  
Altenbergerstrasse 69  
A-4040 Linz - Austria

**Abstract.** CAST.FSM denotes a CAST tool which has been developed at the Institute of Systems Science at the University of Linz during the years 1986-1993. The first version of CAST.FSM was implemented in INTERLISP-D and LOOPS for the Siemens-Xerox workstation 5815 ("Dandelion"). CAST.FSM supports the application of the theory of finite state machines for hardware design tasks between the architecture level and the level of gate circuits. The application domain, to get practical experience for CAST.FSM, was the field of VLSI design of ASICs where the theory of finite state machines can be applied to improve the testability of such circuits ("design for testability") and to optimise the required silicon area of the circuit ("floor planning"). An overview of CAST as a whole and of CAST.FSM as a CAST tool is given in [11]. In our presentation we want to report on the re-engineering of CAST.FSM and on new types of applications of CAST.FSM which are currently under investigation. In this context we will distinguish between three different problems:

1. the implementation of CAST.FSM in ANSI Common Lisp and the design of a new user interface by Rudolf Mittelmann [5].
2. the search for systemstheoretical concepts in modelling intelligent hierarchical systems based on the past work of Arthur Koestler [3] following the concepts presented by Franz Pichler in [10].
3. the construction of hierarchical formal models (of multi-layer type) to study attributes which are assumed for SOHO-structures (SOHO = Self Organizing Hierarchical Order) of A. Koestler.

The latter problem will deserve the main attention in our presentation. In the present paper we will build such a hierarchical model following the concepts of parallel decomposition of finite state machines (FSMs) and interpret it as a multi-layer type of model.

## 1 Implementation of CAST.FSM in ANSI Common Lisp

CAST.FSM was implemented during the year 1986 in INTERLISP-D and LOOPS on Siemens 5815 work stations [12]. However, already in 1992 those workstations had to be put out of order. For the continuation of our research tasks in VLSI

design it was necessary to implement parts of CAST.FSM in Common Lisp and Flavors [7]. In addition, new algorithms were developed to speed up computation and to deal with new areas of finite state machine problems. On the basis of this implementation of CAST.FSM (internally called CAST 2), Rudolf Mittelmann made a portation from Flavors to CLOS for Apple MacIntosh computers using Procyon Common Lisp. This version of CAST.FSM, called macCASTfsm [6], has been in use as a CAST tool at our institute until the year 2000. In addition to the previous CAST.FSM implementations, macCASTfsm also offers new implemented methods as the inversion of finite state machines, the representation of finite memory machines in canonical form, and the shift register representation of finite state machines by the method of Böhling as done by Josef Scharinger in [13]. Due to the incompatibility of macCASTfsm with the latter operating systems of the Apple MacIntosh the use of macCASTfsm on appleMacIntosh computers got obsolete. Therefore, after the introduction of the ANSI Common Lisp standard and since associated Common Lisp environments became available, we had the task to realize a portation of macCASTfsm. Rudolf Mittelmann chose Allegro Common Lisp from Franz, Inc. and also did the implementation. This new version of CAST.FSM for Windows PCs, called winCASTfsm, presently replaces macCASTfsm as a rather powerful and complex CAST tool since 2000.

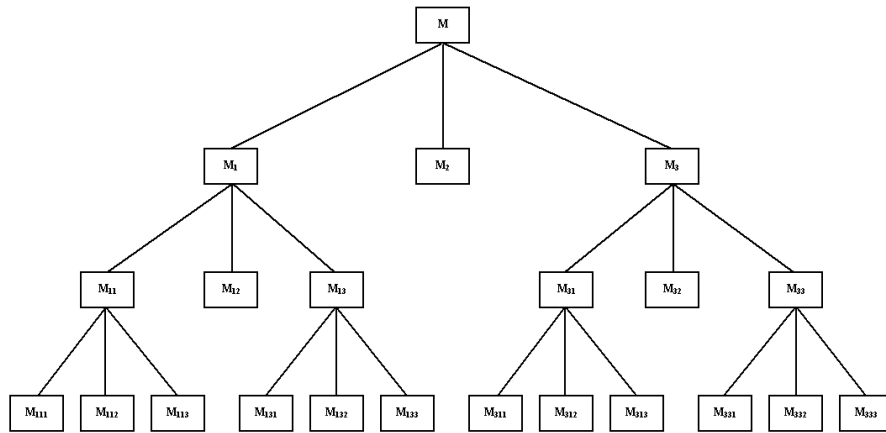
## **2 Hierarchical Models of A. Koestler (Holarchies)**

Arthur Koestler, a well known writer but also an author of science-oriented books, proposed a concept for modeling complex intelligent systems of hierarchical order by his "holarchical nets" [3]. The nodes (called "holons" by Koestler) of such a tree-structured system are assumed to model intelligent components and are equipped with an individual rule base for an autonomous realization of strategies. The introduction of the concept of a "holarchy" was mainly motivated by his studies of the human brain and the organisational structure of a company. Nowadays, motivated by the concept of agent systems in computer science, it is appropriate to consider a SOHO-structure in the sense of A. Koestler as a special kind of an organized multi-agent system (OMAS) as considered by Ferber [1]. Our ultimate goal is to formalize and to refine A. Koestler's concept of a "holarchy" (SOHO-structure) in order to achieve a system-theoretical model which can be classified as "operational". Desired properties of such models are "self-regulation" and "self-repairment". For a mathematical approach to holarchies and for biographical notes to A. Koestler we refer to [4].

## **3 The Multi-Strata and the Multi-Layer Representation of a FSM's Parallel Decomposition**

Generally speaking, decomposition of finite state machines (FSMs) deals with the problem of how a machine can be replaced by more than one simpler machines, i.e. a single FSM is transformed into a network of FSMs. In the case of

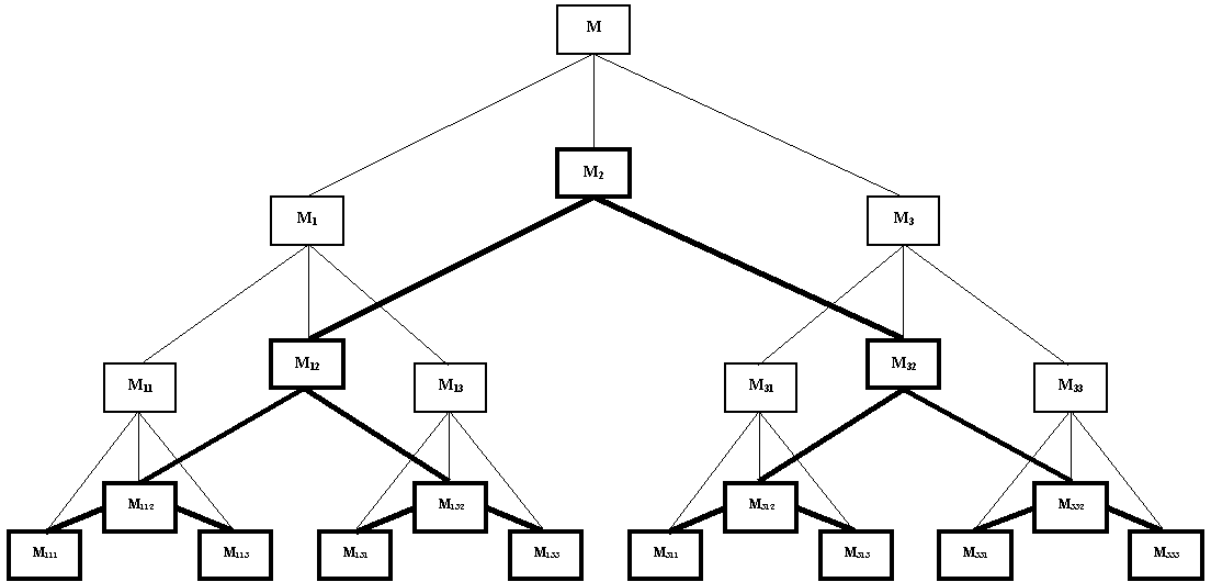
a parallel decomposition of a FSM, a single machine  $M$  is divided into a certain number of submachines whereby the parallel connection of those smaller submachines  $M_i$  ( $2 \leq i \leq n$ ) simulates the original finite state machine  $M$ . Consequently for each of the so obtained submachines  $M_i$  a parallel decomposition can be obtained by splitting  $M_i$  into  $M_{ij}$  ( $2 \leq j \leq n$ ) and so on. The various possibilities of potential parallel decompositions are given by the lattice of the machine or submachine that is actually been taken into account. For further theoretical details the reader may be referred to the well known book of Hartmanis and Sterns [2].



**Fig. 1.** Multi-strata representation of a FSM parallel decomposition.

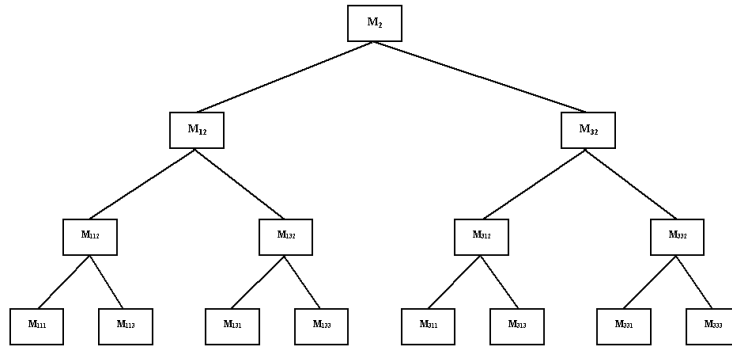
Fig. 1 gives an example of a parallel decomposition of a FSM  $M$  into three submachines  $M_1$ ,  $M_2$  and  $M_3$  whereby  $M_1$  and  $M_3$  further decomposed into  $M_{11}$ ,  $M_{12}$ , and  $M_{13}$  respectively into  $M_{31}$ ,  $M_{32}$ , and  $M_{33}$  and so on. In this multi-strata kind of representation, that is commonly used in the description of FSM parallel decompositions, each level represents the whole system with the restriction that the leaves of each level have to be thought as members of the subjacent levels in order to do so. IN [10] it has been shown, how in general a hierarchical structure (which is required for a holarchy in the sense of A. Koestler) can be derived from a multi-layer representation. In our case of FSMs we have to choose a node, i.e. a submachine, at each layer that will be considered as the direct superior of the other nodes (submachines) being situated in the same layer.

Fig. 2 shows the adaption of the exemplary representation of the FSM's parallel decomposition given in Fig. 1 to the desired (holarchic) interpretation as an intermediate step where the multi-strata representation is fully included and the holarchic model is represented in the bold highlighted part of the figure.



**Fig. 2.** Intermediate step between the multi-strata and the multi-layer (holarchic) representation of a FSM's parallel decomposition.

Fig. 3 shows the achieved holarchic multi-layer representation of the introduced



**Fig. 3.** Multi-layer representation of a FSM parallel decomposition.

multiple parallel decomposition. This is a pure multi-layer representation, i.e. only the components of all layers together model the whole system. As it can be seen in our context, the nodes of the multi-layer model are exactly the leaves of the multi-strata model. Therefore, it is an obvious specification for the design

of such a holarchic model that each subtree emerging by multiple parallel decomposition has to be built up in a way that exactly one of the newly emerging submachines will become the direct superior of the other nodes. As a rule for the destination of this superior-machine it is reasonable to choose that submachine that has the least number of states on order to perform leader tasks. Furthermore, the structure given in Fig. 3 will exactly describe the example of the parallel decomposition of the MCNC benchmark machine which will be shown in the following chapter using CAST.FSM.

#### 4 Realization of a Hierarchical Representation of a FSM by winCASTfsm

In the following we will describe the formation of a finite state machine hierarchy on the example of the bbsse-e MCNC benchmark machine [7] as an application of the latest version of winCASTfsm.

In order to perform a parallel decomposition of a finite state machine it is necessary to compute the lattice of the machine. winCASTfsm supports this feature and the computed lattice is finally displayed as a Hasse-diagram in a new window, a so called lattice browser. Within such a lattice browser, groups of nodes may be selected in order to perform certain lattice browser commands like 'Add Partitions' or 'Multiply Partitions' for showing the supremum respectively the infimum of a selected collection of partitions. According to the theory of FSM decomposition we have to select a group of nodes whose infimum is 0 in order to be able to perform a pure parallel decomposition.

Fig. 4 shows the lattice of the bbsse-e benchmark machine whereby those par-

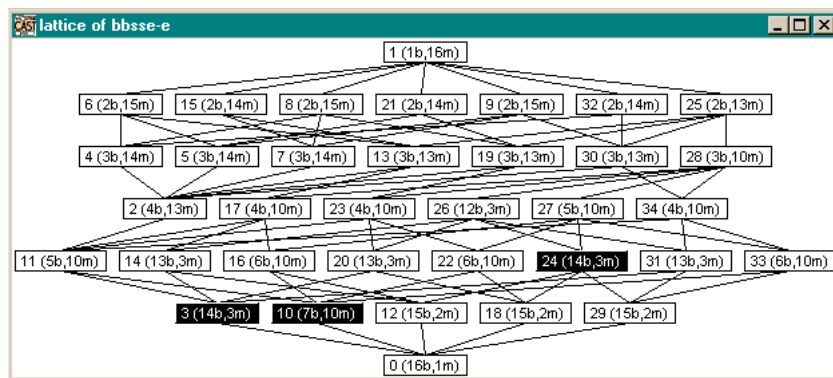
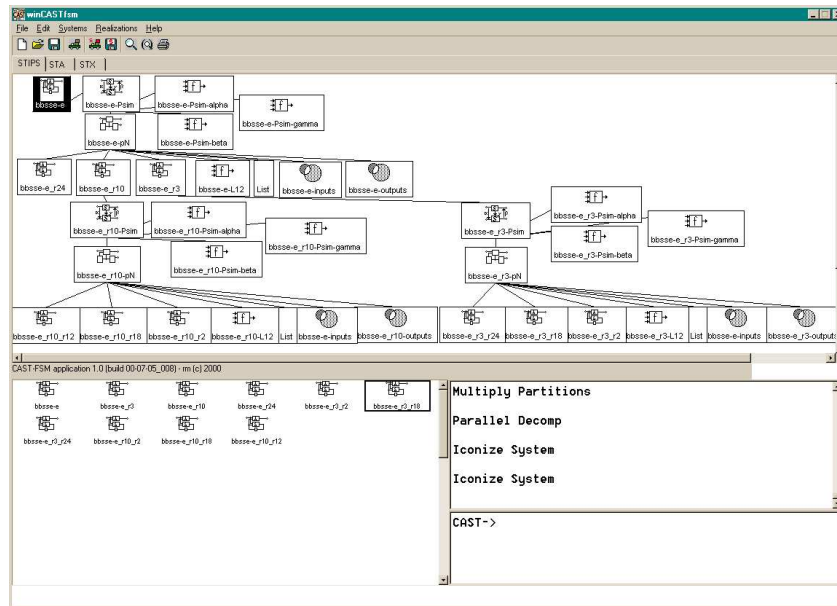


Fig. 4. Lattice of the bbsse-e benchmark FSM.

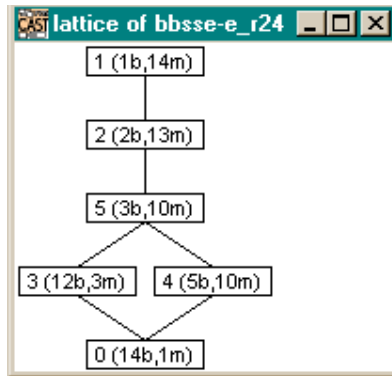
titions which were selected for the parallel decomposition are displayed highlighted. The bracket term includes the number of blocks, i.e. the number of states of the corresponding submachine, and the number of states united by the

biggest block. For example the partition with number 28 has three blocks, and the biggest block unites ten states (hence the label '28(3b,10m)'). According to Fig. 3 the nodes 3, 24 and 14 correspond to  $M_1$ ,  $M_2$  and to  $M_3$  respectively. Having performed the synthesis operation 'Parallel Decomposition' on the selected partitions, winCASTfsm shows the result as an expansion of the realization graph allowing a successive implementation of the described methodology. As indicated in Fig. 5, the realization graph of the bbsse-e benchmark FSM,

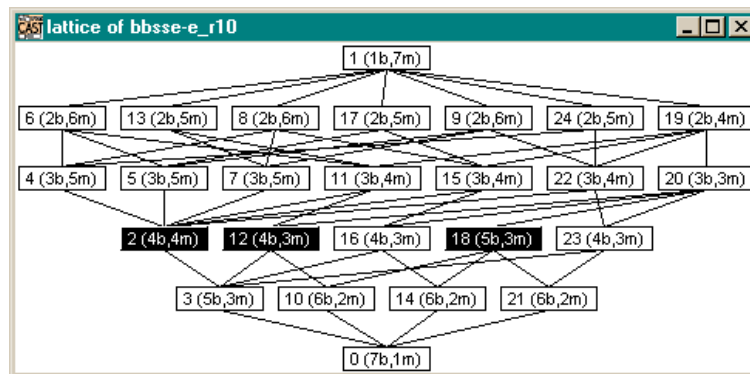


**Fig. 5.** Realization graph for the parallel decomposition of the bbsse-e benchmark FSM.

the machine is divided into three submachines each named by its engendering partition (bbsse-e-r24, bbsse-e-r10 and bbsse-e-r3). By an analysis of the lattices (Fig. 6, Fig. 7, Fig. 8) of the submachines it seems reasonable to select the bbsse-e-r24 as the superior of the remaining submachines. Those remaining submachines will recursively be taken into account for further parallel decompositions as illustrated in the realization graph (Fig. 6). According to the holarchic multi-layer representation given in Fig. 3 the lattice given in Fig. 6 denotes the lattice of  $M_2$  and the lattices of Fig. 7 and Fig. 8 denote the lattices of  $M_1$  and  $M_3$  respectively. The latter both are each split (parallel decomposition) into three submachines whereby one of each triple ( $M_{12}$  and  $M_{32}$  in the notation of Fig. 2 and Fig. 3) is selected as the superior of the others. Subsequently the remaining submachines ( $M_{11}$ ,  $M_{13}$ ,  $M_{31}$ , and  $M_{33}$  in the notation of Fig. 2) are each split into three 'sub-sub-submachines' ending in the final holarchic multi-layer model



**Fig. 6.** Lattice of the submachine derived from partition 24. This submachine is the selected superior and will therefore not be taken into account for further decompositions



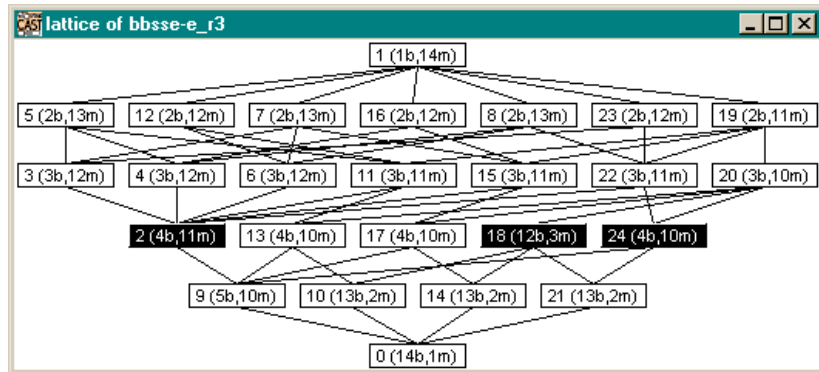
**Fig. 7.** Lattice of the submachine derived from partition 10. The highlighted partitions indicate the partitions that are selected for a further parallel decomposition

given in Fig. 3.

Depending on the task and on the FSM actually taken into account we would achieve different structures with regard to the width and the depth of the final multi-layer hierarchy or the holarity when speaking in the words of Arthur Koestler.

## 5 Conclusion

The present paper briefly introduces the new version winCASTfsm as the latest version of the CAST tool CAST.FSM which has been developed at the department "Systems Theory and Information Technology" at the "Institute of Systems Science" of the University of Linz. In order to demonstrate the power of winCASTfsm we have performed a multiple parallel decomposition of the bbsse-e



**Fig. 8.** Lattice of the submachine derived from partition 3. The highlighted partitions indicate the partitions that are selected for a further parallel decomposition

MCNC benchmark FSM into a hierarchical (multi-layer) representation including the definition of a decision-order.

CAST.FSM is currently in use at the University of Linz as a teaching aid. Furthermore, it is provided as a CAST tool for layout- and the analysis-tasks in application areas such as micro electronics, signal-processing, and coding (development of cryptographic algorithms).

## References

1. Ferber J.: Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence. Addison Wesley Longman (1999)
2. Hartmanis J., Stearns R. E.: Algebraic Structure Theory of Sequential Machines. Prentice Hall Series in Applied Mathematics (1966)
3. Koestler A.: The Ghost in the Machine. Hutchinson&Co.Ltd. London (1967)
4. Meine S.: A Holarchical Organized Design Assistant for Hierarchical Desompositions. to be published in Eurocast'2001. Lecture Notes in Computer Science. Springer-Verlag, Berlin Heidelberg New York (2001)
5. Mittelmann, R.: CAST-FSM in ANSI Common Lisp und Neimplementierung der Benutzeroberfläche. Technical Report, 14 pages. (2000)
6. Mittelmann, R.: Systems Theory on the Apple Macintosh: macCASTfsm Unser's Guide. Technical Report, 50 pages. (1992)
7. Müller-Wipperfürth T., Geiger M.: Algebraic Decomposition of MCNC Benchmark FSMs for Logic Synthesis. In: IEEE Proceedings EURO ASIC'91, Paris (1991) 146–151
8. Müller-Wipperfürth T., Pichler F.: FSM State Assignment for Improved Testability. In: International Workshop on Logic Synthesis. Tahoe City (1993) P5/1–P5/6
9. Müller-Wipperfürth T., Scharinger J., Pichler F.: FSM Shift Register Realization for Improved Testability. In: Pichler, F., Moreno-Diaz, R. (eds.): Eurocast'93. Lecture Notes in Computer Science, Vol. 763. Springer-Verlag, Berlin Heidelberg New York (1993) 254–267

10. Pichler F.: Modeling Complex Systems by Multi-agent Hierarchy. In: Pichler, F., Moreno-Diaz, R. (eds.): Eurocast'99. Lecture Notes in Computer Science, Vol. 1798. Springer-Verlag, Berlin Heidelberg New York (2000) 154–168
11. Pichler F., Schwärzel H.: CAST: Computerunterstützte Systemtheorie. Springer-Verlag, Berlin Heidelberg New York (1990)
12. Prähofer H.: Ein Programmsystem zur Unterstützung von automatentheoretischen Verfahren. In: Pernul, A Min Tjoa (eds.): 10. Fachtagung, Informatik-Forschungsinstitut. Oldenbourg, München (1987)
13. Scharinger J.: Implementierung der Schieberegister-Realisierung für CAST-Systeme. Master Thesis, University of Linz (1991)