Franz Pichler
Johannes Kepler University Linz
A-4040 Linz, Austria
franz.pichler@jku.at

Mathematical Systems Theory: Conceptual Framework and Application Examples

1. Introduction

In scientific problem solving we use models to explore the properties of a real existing system. A special kind of models are mathematical models. Generally speaking the purpose of models is to provide means for the development of algorithm to solve problems which exist for a considered real system. For many reasons it is necessary to provide means for an effective top down development of such algorithms. This requires that the models are based on a solid mathematical basis such as given by the fields of Mathematical Logic and Set Theory, Algebra, Analysis. There exist different classes of mathematical models and also, within a class different model types. It is important to know the different possible existing relations between such classes and between the different possible model types. This enables the problem solver in the process of model construction to select for a given problem the relevant class and to chose the most suited model type. An overview in mathematics comparable to the style of Bourbaki is here most helpful.

A basic framework for model design and for the development of problem solving algorithm can be provided by the field of Mathematical Systems Theory which includes topics such as Automata, Petri Nets, Continuous Time and Discrete Time Dynamical Systems and others. There are a number of monographs and textbooks from the past available, which deal with this topics in detail. It can not be the goal

of this paper to try to repeat the results which can be found there. Our paper has rather the goal to make the reader aware of the already existing results of Mathematical Systems Theory with the hope that curricula in Science and Engeneering, especially also the field of Computer Engineering and Computing Science, will also in the future include such topics.

2. Mathematical Modeling

Based on the kind of architecture we distinguish as model classes between the following three classes:

1. *Chaos Models* are characterized by irregular coupling relations between the different components, no ordering principle of the couplings can "directly" be discovered
2. *Cosmos Models* have regular, mathematical known (e.g. linear) coupling relations with a well defined ordering structure (e.g. a serial-, parallel-, star-, bus-, or a hierarchical order)
3. *Bios Models* consists of a large number of multi -facetted components of different classes and types which are hierarchical ordered by different layers

Chaos models are used to model by mathematical means "real chaos" as it appears in different fields of science but also in engineering. The phenomenon of brownean motion and thermal noise are well known examples in physics. Complex systems which have been designed by couplings of components which have been "glued on" without the knowledge of its resulting effects are examples in engineering. By mathematical analysis it is a chance to represent real chaos as a mathematical object which allows problem solving by mathematical means, [1], [2].

Cosmos model denote in our terminology models which are mathematical objects. Their architecture allows the investigation by mathematical means. Cosmos models are the common class of models in science and engineering. The most important chaos models in physics and in engineering models are such which are constructed by differential equation systems. Other models of this class which are in common use in physics and engineering, to mention additional important

examples, use difference equation systems, finite state machines or petri nets. All Cosmos models of this type become by their evolution in time or space (in mathematical terms, by their "integration") the model type "dynamical system", a type which is central for mathematical systems theory, [8], [9].

Bios models in our definition consists of many components of different kind which are hierarchically grouped into different layers. They are typical for models of living systems as they exist in nature. Their mathematical analysis is difficult and can often only be done by computer simulation. Essential features of bios models are their ability to demonstrate essential biological properties such as self-repair and replication, [4], [5]. [6].

To support the "art of mathematical modeling" it is of help to distinguish between five different model types as follows:

1. Model type *Black Box*: A black box is modeling the interface of a system and its environment by the determination of input/output ports and their associated signals together with the determination of possible parameter ports and the values which can be applied there.
2. Model type *Generator* : A generator model determines the states and local state transition. Furthermore initial states and relations of states to possible input/output signals are determined.
3. Model type *Dynamical System*: Dynamical systems determine, similar to a generator model, the states of a system but allow the determination of global state transition, or in other words, the computation of state trajectories.
4. Model type *Algorithm*: An algorithm determines a computational process starting from given initial data, followed by a sequence of operations to be executed until a set of final data is reached.
5. Model type *Network*: A network model consists of a set of different components together with the different relations which realize the interaction to form a system.

The model types above are well known in science and engineering. It is easy to give mathematical examples. The process of practical problem solving starts, de-

pending on the a priori given knowledge on a certain model type. Here the types "black box", "generator" and "network" have a favourite role. For further analysis the goal is often to reach a model of type "dynamical system" or, if a computation is the goal, the type "algorithm" is desired. To support the realization of such processes, the consideration of transformation between the different model types are required.

Transformations of this kind have a long tradition in mathematics. Examples are the "determination of a solution of a differential equation" (a transformation from a generator model into a dynamical system) or the "decomposition of a given binary relation into a composition of (smaller) binary relations" (a transformation of a black box model into a network –of black box models–).

3. Mathematical Systems Theory

Mathematical Systems Theory can be seen as a part of Applied Mathematics with an emphasis on "systems problems" as they appear in science and engineering. In history the term "systems theory" was used first by the german engineer and professor Karl K¨upfm¨ uller in his book "*Die Systemtheorie der Elektrischen Nachrichtenübertragung*" (Systems Theory for Electrical Information Transmission) which appeared in the year 1949. Another pioneer in systems theory was the Austrian biologist Ludwig von Bertalanffy. His book "*General Systems Theory*" of 1969 adressed systems problems in biological systems. The field of "Mathematical Systems Theory" was established in the USA by the work of Rudolf Kalman and others from the year 1960 on. Its possible applications were in that time period mainly oriented towards automation and control. However also fundamental research on automata theory, information transmission and coding, general dynamical systems was done in that years and can be seen also as part of Mathematical Systems Theory. Even the field of "Cybernetics" as introduced by Norbert Wiener in 1948 influenced Mathematical Systems Theory very strongly. Today Mathematical Systems Theory deals with different difficult problems in the design and analysis of the complex engineering systems, including computer hardware and computer software. Its applications are besides of the field of automation and control also in the field of computer engineering, signal processing and coding, software engineering and others.

In order to give an idea what kind of parts are available in Mathematical Systems Theory for the different model classes which we introduced above, we give in the following an overview and also citations of existing literature.

- **Chaos Models:** Theoretical insights can be derived by the results of the now existing theory of chaotic processes ("deterministic chaos" generated by nonlinear difference-and differential equation xsystems) and by the theory of fractals, [1], [2].
- **Cosmos Models:** The field of dynamical systems (finite state machines, cellular automata, petri nets, theory of difference equations systems, theory of differential equation systems, general dynamical systems) offers means for the construction (design) and the mathematical treatment of models of the class of "Cosmos models" [8], [9].
- **Bios Models:** Bios models require, to name a few, a mathematical treatment of living systems (for example founded in its architecture on the concept of holons as proposed by Arthur Koestler). In Mathematical Systems Theory the theory of morphogenetic processes, of learning systems, of intelligent multi-agent systems and of systems with the ability of reproduction and self-repair deal with aspects of living systems. John Casti has shown how by natural extension of a discrete time linear dynamical system the construction of an abstract metabolism-repair system in the sense of Robert Rosen is possible. This offers a chance to apply the existing results of (engineering oriented) Mathematical Systems Theory also for the modelling of living systems. The future inclusion of the approach of Casti in university courses in Mathematical Systems Theory would be very desirable and would help to build a bridge between the existing engineering and physics oriented approaches in modelling and the already existing framework for modelling living systems ([3], [5], [6].

4. CAST Methods in Problem Solving

We know that the development of computing power has changed dramatically the means of problem solving. The tools for the design and analysis of models of different kind are today highly developed. In many cases, however, their design has been done by piecewise improvements and improvements seem to be possible. Further-

more the documentation of such tools is in many cases only rudimentary available for the user. Tools are often for proprietary reason closed to changes and redesign is very difficult. In this situation it seems to be desirable to develop a conceptual framework which allows the development of tools of better quality which are supported by the available existing methods of mathematical systems theory. The such improved tools should be able to apply the results of the existing different fields of mathematical systems theory such as "linear systems theory" (the theory of ordinary linear differentialand difference equation systems with constant coefficients together with the theory of linear automata) or the theory of finite state machines for practical design and analysis tasks. In the past such tools have been called "CAST-tools", where CAST stands for "Computer Aided Systems Theory". The term "CAST" was first introduced by the author together with Roberto Moreno Diaz in 1989 (EUROCAST'89, Las Palmas, Gran Canaria, Spain). CAST tools complement in microelectronic applications the tools for CAD (Computer Aided Design) and CAM (Computer Aided Manufacturing).

A CAST tool can for short be characterized by the availability of model types of different kind together with the known existing transformations to relate such types to each other. Both have to be available in implemented form as components of a user friendly data base system. The problem solver is made able to use a CAST tool to develop in problem solving CAST algorithms by starting from a initial model of certain type followed by applying stepwise known model transformations until a solution is reached. A important feature of a CAST tool is that the different model types are hierarchically ordered such that the construction of CAST algorithms can be done top down.

For a description of the framework recommended for the implementation of a CAST tool the reader is advised to consult the existing literature [7].

Since systems theory is already for many years an accepted topic in the fields of Electrical Engineering, Communication Engineering and Control Engineering it is naturally that different tools have been implemented there in the past to support design and analysis (e.g. design and analysis of control systems or the design of electrical filters) [21]. Similar developments can be observed in the field of Computer Engineering and Computer Science. Although the tools

developed there make usually no reference to CAST they can be considered as CAST tools in our sense. An example of a CAST tool which reflects its main features is the tool CAST.FSM which was implemented in the year 1986 at the Department of Systems Theory of the Johannes Kepler University Linz, Austria, by supervision of the author. CAST.FSM is able to support design and analysis tasks for finite state machines. Originally it was implemented by the language Interlisp-D for Xerox Park Dandelion work stations. The possible applications of CAST.FSM were mainly seen in the field of VLSI design. There the decomposition of a finite state machine to get an improvement in its testability was considered as an important topic. Other applications concerned the field of cryptography, which will reported in a later chapter. To give support to the "CAST movement" in the past special international meetings have been organized and published (Proceedings of the International Symposia EUROCAST in 1989 (Las Palmas),1991(Krems),1993 (Las Palmas),1995 (Innsbruck),1997 (Las Palmas),1999 (Vienna), 2001 (Las Palmas), 2003 (Las Palmas), 2005 (Las Palmas), 2007 (Las Palmas) edited by Roberto Moreno Diaz, Franz Pichler and others, Lecture Notes in Computer Science, Springer-Verlag Berlin- Heidelberg).

As a result of the done activities in research and teaching concerning the "CAST framework" and the development of CAST tools it can be stated that the done activities improved the awareness of scientists and engineers, here mainly in the field of computer science and computer engineering, of the value of the use of mathematical models and of taking a mathematical approach, based on mathematical models, in problem solving.

5. Applications in Cryptography

In the following we will try to demonstrate how a systems theoretical oriented approach can be applied to practical problems. We use results which have been achieved during the former occupation of the author as chairman and professor at the Department of Systems Theory and Simulation at the Johannes Kepler University Linz. As field of application we choose in the following the field of cryptography, which can be considered as a part of the field of signal processing and coding.

5.1 Design of Pseudo Random Generators

Pseudo Random Generators (for short PRG's) are needed in stream ciphering (secret key systems). Stream cipher systems are besides of block cipher systems a part of the so called "secret key system" where both the keys, the transmitter-key and the receiver-key, have to be kept secret. Such cryptographic systems allow very fast encryption o data and are therefore strong in use. The most simple stream cipher system, as seen from the hardware point of view, which is nevertheless highly secure is given by Vernam encryption, where a true random data stream, which is available on both ends of the transmission line, is mixed with the plain text data. A practical approximation of a Vernam system uses a pseudo random generator on both ends of the line to produce the "key stream" to be mixed with the plain text data. The design of pseudo random generators of high cryptographic quality requires a carefully mathematical approach. The following criteria (described here in a first form) give more detail on the kind of requirements on a pseudo random generator for stream ciphering applications. The following properties (1)-(6) are required:

1. long primitive period
2. large linear complexity
3. ideal k-gram distribution of key stream
4. if K and K' differ only in one digit (for the Hamming distance –dH– between K and K', we have $dH(K,K') = 1$) then for almost all pairs (w,w') of length n of input/output words which are generated $dH(w,w') = \frac{n}{2}$ should be valid (*confusion property*).
5. redundancies in the key stream must dissipate in long term statistics (*diffusion property*)
6. boolean functions which are used must fulfil certain criteria of non-linearity, correlation immunity, bentness and others to guarantee the non-leaking property

The requirement (2), to assure a large linear complexity, should be explained here in more detail. (2) is a measure to prove resistance against linear algebra attacks, this means that the simulation of a pseudo random generator by a linear machine is difficult to realize. The linear complexity of data stream which is generated by a pseudo random generator is defined as the dimension of the state space which

is needed for the equivalent linear machine. It is therefore a special case of the Kolmogoroff-Chaitin measure of a data stream. The linear complexity of a scalar binary data stream can be effectively be determined by the Massey -Berlekamp algorithm. For vector-valued (n-Byte) streams of data we introduce in a later chapter of this paper the Rissanen algorithm to determine the associated linear complexity.

The cryptologically essential part of any practical stream cipher system is the pseudo random generator (PRG) which can be modelled by autonomous finite state machine PRG=(Q, B, δ, β, K) with (complex) state transition function δ, output function b and initial state K. The PRG generates the output sequence $k= k0, k1, k2, k3$... which is the key stream. As key $\boldsymbol{K}$ of the PRG the triple $\boldsymbol{K} = (\delta, \beta, K)$ can be taken. The component K of the key may not be identical to the initial state $q(0)$ of the PRG. A randomization variable X such that the initial state $q(0)$ is given by $q(0)=(K,X)$ may be additionally introduced to make a key attack more difficult.

We repeat again some of the design requirements of above taking the fact that the PRG is modelled by a finite state machine into account.

The following properties of a PRG are desired :

1. the key space (the set of all keys $\boldsymbol{K}$) has to be large enough
2. the key stream has to have a very long period
3. the key stream does not differ in the statistical properties with respect to a number of statistical tests from true random streams
4. it is provable computational difficult to compute from an arbitrary number of digits of k the key which is in operation
5. a effective digital electronic realization of PRG is possible

In cryptology the following modes of operation of a PRG are distinguished:

- **Counter mode:** The key K is reduced to K=ß. The PRG is designed such that a the state trajectories are long cycles (a MLFSR or a CCMLFSR may serve for the realization of the state transition). The key stream is determined by the individually chosen output function
- **Internal feedback mode:** The key K is given by K=K or also K=(δ ,K)

In the past at the Institute of Systems Theory and Simulation, JKU Linz, several research projects on the design of stream cipher systems have been pursued. In the following we discuss by three examples the key problems, as seen from a systems theoretical point of view, which had to be solved. The examples were

1. Design of a one-way function by 2-D cellular automata
2. Design of a combiner generator
3. PRG based on cascades of baker register machines

Example (1): Design of a PRG with cellular state transition function. The state transition function is realized by a cellular 8 x 8 systolic array of finite state machines. It is assumed that the individual machines have 5 bit register memory. The key $\boldsymbol{K}$ of the PRG is defined by the initial state of the different machines. Key length is therefore 320 bit, which is high enough to meet requirement (1) of above. The coupling of the cellular array has to be locally homogenous. The following coupling has been chosen: each FSM is coupled with its 4 neighbours by 2 input ports from left and from the above neighbour, 2 output ports to the right and to the below neighbour. In order that the one-way property of the state transition function is fulfilled the individual finite state machines FSM (i, j),$(i, j = 1,2,...,8)$ have to be properly chosen. The input/output function of the cellular array is defined as follows: The first row receives a constant input $k0$ at time 0 (which can be taken as a private user key), the last row produces the output of the function $f : \boldsymbol{B}^8 \to \boldsymbol{B}^8$. The state equation $k(t+1)=f(k(t))$, with $k(0)=k0$ and $t=0,1,2,...$ defines the key stream k of the PRG.

For the design of the finite state machines we followed empirical given knowledge as follows. Two different FSM's have been designed, which were randomly distributed on the cellular array. The following properties of the FSM have been required:

1. not invertible
2. complex linearisation
3. state reduced
4. no finite memory
5. not decomposible
6. no shift register realization possible

To find valid FSM's which fulfil (1)-(6) the tool CAST.FSM, our method base system for the application of finite state machine theory, was applied. Furthermore the cellular automata array was simulated with our tool CAST.LISA (Linz Systolic Array Simulator). Finally a prototype for microelectronic realization was designed by using the tool VENUS of the company Siemens AG, Munich.

By simulation the following I/O properties of the state transition function of the cellular array by computing the hamming distance *dH* have been investigated:

1. *dH*(input,output)
2. *dH*(output, output(i)), where output(i) defines the output which results if the original input is changed by 1 bit at coordinate i (i=,1,2,...,8)
3. *dH*(output,next output)

The received results have been proven satifactory. In the final step characteristic features of the microelectronic design was evaluated. The tool VENUS provided the following results for the designed application specific integrated circuit (ASIC):

- number of cernel cells: 5.942
- number of cell types: 38
- number of pad cells: 36
- number of gates: 51.402
- floor space for cells: 16,886.888 qum
- floor space needed: 61,026.572 qum
- clock frequency: 50 Mhz

Cell library which was used: LIBM (Siemens AG Munich)

The necessary floor space proved to be too big to get a satisfactory chip production. A redesign was not taken in this example. This example of a cryptographic design task was the content of a master thesis (in german) [22].

Example (2) : Design of a combiner generator A PRG is called a combiner generator if its architecture is of the kind as shown in figure 1.

A combiner generator consists of a parallel circuit of m PRG's which are "combined" by their outputs with the combiner C. C is usually realized by a (boolean) function, however, also finite state machine combiners are possible. From a systems theory point of view there a two important properties required such that a combiner generator is "strong". (a) It is to prove that the architecture as shown in figure 1 can not be simplified by decomposition. (b) the combiner C has to be designed such that the output stream y becomes "much stronger" in a cryptanalytic sense with regard to the individual input streams *x1,x2,. . . ,xm*. For the design of boolean combiners its spectral properties in the Walsh-Fourier domain are important. The mathematical theory of Walsh functions play here an important role, [20]. The design of finite state machines can be in the case of finite memory machines reduced to the design of boolean function combiners. Another method to design finite state machine designers uses the irregular switching of Boolean function combiners as a principle, [18].

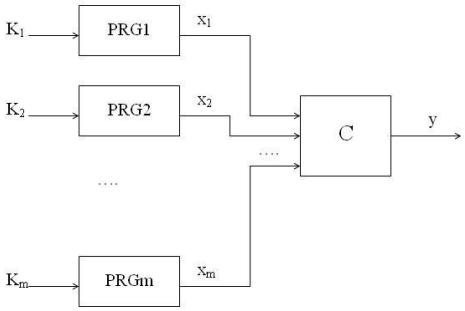

Figure 1. Architecture of a combiner generator

Example (3): Design of a Pseudo Random Generator based on cascades of baker-permutation register machines A baker permutation register machine is defined as clock controlled register machines with baker permutation as transition function. A baker permutation is a discrete finite realization of the generalized baker transform which is known in mathematics. Generalized baker transforms generate by iteration a chaotic process It has been shown that such a permutation is a random permutation in the sense of Feller. This gives the motivation to use it for the state transition of register machines (the state transition is defined by register

reading and writing operations). It has been shown that Gollmanncascades of clock controlled baker permutation register machines (CCBRM's) give pseudo random generators of good cryptanalytic quality. A PRG constructed by a parallel circuit of CCBRM's with an architecture as shown in figure 1 has been tested for stream cipher applications [19].

5.2 Linear Complexity Measures for Stream Ciphers

Cryptographic devices which use the concept of stream ciphering are based on pseudo random sequences which are used to be mixed with plaintext data streams. The generation of such pseudo random sequences is done by a "digital machinery", in hardware or in software, such that cryptanalysis (determination of the key in use or recovering the plaintext hidden in the observed data stream) is a hard mathematical problem. This gives the necessary requirement to prove that the generation of the pseudo random sequences cannot be done by a "linear digital machinery". From the point of view of the theory of finite state machines we have in this context the following problem statement: Given a data stream $a = a(0), a(1), a(2),....$ determine a autonomous linear finite state machine ALFSM=(Q, B, δ, β) of minimal dimension n which generates from a certain initial state $x(0)$ the data stream a. The existence of a solution to this given problem depends on the kind of the data stream a. In case that there exists a solution ALFSM we call the dimension n of the ALFSM (which is equivalent to the dimension of its state space Q) the linear complexity of a.

The restricted problem to find a ALFSM which generates a data stream of finite length M, $aM = a(0), a(1), a(2),...., a(M-1)$ has always a solution with linear complexity $n(M)$, with $n(M) \leq M/2$. The sequence $(n(M))$, $M = 1, 2, 3,...$ of values $n(M)$ which we can compute stepwise by the given data stream a is called the linear complexity profile of a. The values $n(M)$ are increasing. If from a certain length M^* a constant value $n(M^*)$ is reached, we have computed the linear complexity $n=n(M^*)$ of the data stream a. In case that for a data stream a this possible, a is not suitable for the use in a cryptographic strong stream cipher.

Together with the determination of $n(M)$ for aM we also are interested to determine in addition the associated ALFSM(M) and also the associated initial state $x(M)(0)$ of it. The sequence LCM$(a)=(n(M)$, ALFSM(M), $x(M)\ (0))M=1,2,3,...$, is called the desired linear complexity measures of a data stream a. For the application in cryptanalysis effective methods to compute linear complexity measures of a data stream a are necessary. For scalar-valued data streams such a method is given by the well known algorithm of Massey-Berlekamp. Only recently a method which allows the effective computation of linear complexity measure also for vector-valued data streams has been found. It has been called the algorithm of Rissanen [14], [15]. The essential method to develop the Rissanen algorithm is provided by the theory of linear systems realization. In the following we give a short overview on the different steps which are necessary for the development of the Rissanen algorithm.

The determination of a minimal linear system (F, G, H) from a given impulse response A is called in Mathematical Systems Theory the linear systems realization problem. Rudolf Kalman developed in the 1960's a general theory to solve the linear systems realiziation problem based on the mathematical framework of R-modules. The textbook on "Systems Theory" of Padulo-Arbib provides a good introduction to linear realization. The essential data for the computation of the state space of (F, G, H) are provided by the Hankel-matrix H derived from the impulse response A of (F, G, H). The restricted problem to determine $(F(M), G(M), H(M))$ from a impulse response $AM=A(0), A(1), A(2), ..., A(M-1)$ of finite length M is called the linear partial realization problem ([8] chapter 6, [9]).

The algebraic theory of linear systems realizations deals with determination of a minimal linear system $\Sigma=(F, G, H)$ for a given (observed) impulse response A: T$\rightarrow M$ (p x m) . T denotes the time scale and M(p x m) is the set of matrices of size pm over a field **K** (the impulse response A can be considered as a multiple I/O experiment on Σ):

if T=**R** (continuous time) and **K**=**R**, Σ is a linear differential system
if T=**Z** (discrete time) and **K**=**R**, Σ is a linear difference system
if T=**Z** (discrete time) and **K**=GF(q), Σ is a linear finite state machine LFSM

Our interest is the case of linear finite state machines LFSM=*(F, G, H)*.In this case the impulse response A of Σ is given by *A=A(0),A(1),...* where *A(k)* are matrices with elements in *GF(q)*. It can be shown that the impulse response A of a discrete time linear system Σ *=(F, G, H)* can generally be expressed by

$$A=(HG, HFG, HF^2 G, HF^3 G,...) \quad (1)$$

The linear system realization problem has the goal to determine *(F, G, H)* which meets equation (1). To reach a solution of the linear realization problem we have to determine the state space Q of Σ. Let $f:U \rightarrow Y$ denote the "zero state" I/O function of Σ which assigns to each input function u with finite support ("input word") the associated output function $y=f(u)$ ("output word").Then the state space Q of Σ can be constructed by the quotient space

$$Q=U/ker(f) \quad (2)$$

For a impulse response *AM=A(0),A(1),A(2),...,A(M-1)* of finite length the associated Hankel matrix *H(N)* is given by the $N \times N$ matrix

$$H(N) = \begin{bmatrix} A(0) & A(1) & A(2) & \dots & A(N-1) \\ A(1) & A(2) & A(3) & \dots & A(N) \\ \dots & \dots & \dots & \dots & \dots \\ A(N-1) & A(N) & A(N+1) & \dots & A(M-1) \end{bmatrix}$$

The rows of *H(N)* show the responses of the impulse inputs *e(i)* and their shifts *e(i,k), k=1,2,...,N-1.*

The dimension of the associated linear system *(F(M),G(M),H(M))* is given by the rank of *H(N)*.

Let *e1*, *e2*, *e3*,...,en denote unit input impulse functions which generate as response linearly independent rows of the associated Hankel matrix *H(N)*. With [*e1*],[*e2*],[*e3*],...,[*en*] we denote the basis of the state space Q of the linear system *(F(M),G(M),H(M))* which is generated by the set {e1, e2, e3,...,en}. Then in case

that rank $H(N)$ = rank $H(N+1)$ the matrices $F(M)$, $G(M)$, $H(M)$ can be computed by

$$\begin{aligned} F(M) &= [[e10], [e20], [e30],..., [en0] \\ G(M) &= [[e(1)],[e(2)],[e(3)],...,[e(m)]] \\ H(M) &= [f(e1)(0)jf(e2\ (0))jf(e3(0))j...jf(en(0))] \end{aligned} \tag{3}$$

here $[u]$ denotes the state which is generated by the input word u in the chosen basis, $ei0$ $(i=1,2,3,...,n)$ denotes the 1-step shifted function ei, $f(ei)(0)$, $i=1,2,...,n$ is the output of the system at time 0 as response to ei.

Jorma Rissanen (IBM Research CA and Stanford University, now with MDL-Research, Tampere, Finland, expert in "Statistical Modeling) developed in the late 1960's for the computation of partial realizations an effective method to compute recursively stepwise by the length M of an observed impulse response AM the associated linear partial realization $SM=(F(M),G(M),H(M))$ and so also its dimension $n(M)$.By the method of Rissanen the computation of $SM+1$ depends only on the result SM and the next value $A(M)$ of $AM+1$, [10], [11].

The solution of the linear realization problem allows a solution of the stated cryptanalytic problem in the following way: For $m=1$ (scalar input) the finite length impulse response AM of LFSM is a vectorvalued sequence $AM=(A(0), A(1),..., A(M-1))$ with $A(k) \in GF(q)^p$ which can be interpreted as a part of a pseudo random sequence generated in a stream cipher device. Then the Rissanen method of linear partial realization allows the determination of the linear complexity measures of A.

If $S=(F,G,H)$ is for AM the solution of the linear partial realization problem then (F,H) generates from the initial state $x(0)=G$ (since $G=Ge(0)$, e denotes the scalar unit impulse function) the finite length sequence AM as output.
Remark : This result seems to be have been overlooked so far by the professional cryptologists.

To get it, one has to know the result of Kalman's realization theory including the method of Rissanen of 1971 and to consider the fact that the impulse response AM is defined from time 1 on. By timeinvariance of a linear system S and by the fact that

S=(F,G,H) operates with zero-input just as the corresponding autonomous linear System *(F,H)* the above solution is rather trivial.

The solution of the stated cryptanlytic problem for the determination of the linear complexity profile of a vector-valued pseudo random sequence *AM* of (usually very long) length *M* by means of the Rissanen method for the solution of the linear partial realization problem will be called Rissanen algorithm. The Rissanen algorithm generalizes the Massey-Berlekamp algorithm. The Massey-Berlekamp algorithm computes for scalar sequences s with values in *GF(q)* the linear complexity profile *L*. The computation is effective by the recursive procedure to compute *L(M+1)* from *L(M)* and *s(M)*.The method is based on polynomial presentation of sequences by using the D-transform (Laurent expansion of sequences) which is common in shift register theory. The realizing ALFSM is in this case a autonomous linear 76 feedback shift register ALFSR of minimal length *n(M)*. The Massey- Berlekamp algorithm is standard and since a long time known in the field of cryptography [16].

Modern cryptographic devices for (fast) stream cipher systems need pseudo random generators which generate vector-valued (Byte-oriented) sequences to be $\oplus$-mixed with the plain text data to get the cipher text. For the determination of the linear complexity profile of the generated sequences by quality control the Rissanen method for partial linear systems realization can be applied. The Rissanen algorithm (we use this term to point to the Massey-Berlekamp algorithm) as discussed above, computes to a given vector-valued digital signal for "windows" of length M (overlapping or non-overlapping) an associated set of parameters which are given by the matrices *F,G,H*, which allow a reconstruction of the signal. Application in data compression, signal classification and data coding seem to be possible.
Remark: Australian outback-patents by Pichler-Kookaburra are pending [23].

The Rissanen method for the computation of linear partial realizations needs an effective implementationby software. According to information received from Jorma Rissanen no modern implementation of his method of 1971 is known. My former PhD student Dr.Dominik Jochinger, provided recently an implementation for it. The Rissanen algorithm together with the already implemented Massey-Berlekamp algorithm will become a part of the "Crypto Workbench" [17].

Of practical, but also of theoretical interest, is the determination of a basis for the LFSM=*(F,G,H)* which is computed by the Rissanen method such that the matrix *F* becomes rational canonical form, which means that the LFSM is a parallel composition of autonomous LFSR's.

Our results show that rather "classic" topics of mathematical systems theory, such as the "Algebraic Theory of Linear Systems" as developed by Rudolf Kalman and others nearly 50 years ago have still the "power" to lead to new applications of current interest. The "linear realization method", which is a part of the theory, was developed originally mainly for applications in control theory. Today this method seems to be rather forgotten in academic curricula. I myself was envolved in teaching this topic some 30 years ago. However only recently the here reported application has been found (together with the support of L.Kookaburra of Forresters Beach, Australia), [14], [15].

The paper of Jonckheere,E. and Chingwo Ma: A Simple Hankel Interpretation of the Massey-Berlekamp algorithm, which appeared in "Linear Algebra and ist Applications", 1989, 65-76, makes reference to realization theory however the authors seem to have not seen the possibility of a generalization to vector-valued sequences [13].

6. Final Words

Mathematical Systems Theory has its origin in the goal to provide mathematical methods for model construction and model analysis in the field of control (design and analysis of controllers for dynamical processes in engineering and science) and in communication engineering (design and analysis of electrical networks for signal transmission, design and analysis of coding devices, electrical wave filters etc). Its results have been proven to be of general interest in science and engineering. This is especially true in the case of the field of Computer Science and Computer Engineering. Today the field of Mathematical Systems Theory can be seen as the kind of Applied Mathematics which is needed in this new fields of engineering. The solution of problems in current topics in computer science and computer engineering require a precisely mathematical treatment of models and algorithm development. Mathematical Systems Theory can give support here and is able to provide a

solid basis for it. In Mathematical Systems Theory the theory of dynamical system (with input and output, as pioneered by the work of Rudolf Kalman and others) play a central part. They are important instruments for the mathematical modelling of "Cosmos Models" in Science and Engineering and can give also directions in dealing with "Chaos Models". However, as recent work by John Casti and others in Mathematical Systems Theory shows, they can also be imbedded into a more general framework of dynamic systems which can serve in the class of "Bios Models". This gives the hope that Mathematical Systems Theory will continue to be an important topic in teaching, research and in applications.

References

1. Pierre Bergé, Yves Pomeau, Christian Vidal. Order within Chaos. Towards a deterministic approach to turbulence. Hermann, Paris 1984.

2. Benoit B. Mandelbrot. The Fractal Geometry of Nature. W.H. Freeman and Company, New York 1977.

3. John Casti, Anders Karlqvist. Newton to Aristotle. Toward a Theory of Models for Living Systems. Birkhäuser, Boston 1989.

4. John Casti: Newton, Aristotle, and the Modeling of Living Systems. *In Casti-Karlquist: Newton to Aristotle*. Birkäuser, Boston 1989, pp.47- 89.

5. Roberto Moreno-Diaz and José Mira-Mira. Brain Processes, Theories and Models. The MIT Press, Cambridge, Mass. USA,1996.

6. Arthur Koestler. The Ghost in the Machine. Random House, New York, 1967.

7. Franz Pichler, Heinz Schw¨ artzel.CAST Methods in Modelling. Springer-Verlag Berlin-Heidelberg 1992.

8. Kalman, R.E., P.L. Falb, M.A. Arbib. Topics in Mathematical Systems Theory. McGraw Hill, New York 1969.

9. Padulo, L. and M. Arbib. System Theory. An Unified Approach to Continuous and Discrete Systems. Hemisphere Publishing Corporation, Washington D.C. 1974.

10. Rissanen, J. Recursive Identification of Linear Systems, *In SIAM Journal on Control*, Vol 9, No 3, August 1971,420-430.

11. Rissanen, J. and T. Kailath. Partial Realization of Random Systems. In *Automatica*, Vol. 8, Pergamon Press 1972, 389-396.

12. Pichler, F. General Dynamical Systems: Construction and Realization *Lecture Notes in Economics and Mathematical Systems*, Springer 1976, 393-408.

13. Jonckheere, E. and Chingwo Ma. A Simple Hankel Interpretation of the Massey-Berlekamp Algorithm. *In Linear Algebra and its Applications*, 1989, 65-76.

14. Pichler, F. Effective Computation of Cryptanalytic Measures for Stream Cipher Data by the Rissanen Algorithmus. In *Revista de la Academia Canaria de Ciencias*, XIX (N′ums.1-2), 2007,pp.9-22.

15. Pichler, F.and L. Kookaburra. Forresters Beach Notes Forresters Beach, Central Coast, NSW, Australia, February 2008.

16. Massey, J. Shift register synthesis and BCH decoding. In *IEEE Trans on Information Theory* IT-15, 1967, 122-127.

17. Jochinger, D. A Software Implementation of the Rissanen Algorithm Lecture at EUROCAST'09, Las Palmas, February 2009.

18. Pichler, F. A highly nonlinear cellular FSM-combiner for stream ciphers. Lecture presented at EUROCAST 07, Las Palmas, February 2007.

19. Pichler, F. and D. Jochinger. A new Pseudo Random Generator based on Gollmann Cascades of Baker Register Machines. Lecture presented at EUROCAST 05, Las Palmas, February 1905.

20. Pichler, F. Walsh-Fourier Analysis of Boolean Combiners in Cryptography. In *Walsh and Dyadic Analysis, Proceedings of the Workshop*, October 18-19,2007, Faculty of Electronics Niˇs, Serbia, pp.171-181.

21. Jamshidi, M., Herget,C.J. Computer-Aided Control Systems Engineering. North-Holland, Amsterdam 1985.

22. Gassner,F. Entwurf und Implementierung eines komplexen 168 Schaltwerkes mit dem VLSI- Entwurfssystem VENUS, Master Thesis, Johannes Kepler University, Dept. of Systems Science,1989.

23. Pichler,F. and L. Kookaburra. Research Emeritus Notes. Forresters Beach, NSW, Australia, February 2008.