Design of self-synchronizing stream ciphers: A new control-theoretical paradigm

Brandon Dravie, Philippe Guillot, Gilles Millérioux

Design of Self-Synchronizing Stream Ciphers: a New Control-Theoretical Paradigm ⋆

B. Dravie * **P. Guillot** ** **G. Millérioux** *

* *Université de Lorraine, CRAN, UMR 7039, France, CNRS, CRAN, UMR 7039, France (e-mail: {brandon.dravie,gilles.millerioux}@univ-lorraine.fr).*
** *Université Paris 8, LAGA, France, (e-mail: philippe.guillot@univ-paris8.fr)*

Abstract: This paper deals with the use of control theoretical concepts in the context of private communication. It is proposed a new and systematic methodology to design a cryptographic architecture belonging to the special class of ciphers called Self Synchronizing Stream Ciphers (SSSC). Till now, the constructions of SSSC were based on automata with finite input memory involving shifts or triangular functions (T–functions) as state transition functions. Besides, only ad-hoc approaches were available in the literature. The contribution of this paper is to propose not only a general framework to design SSSC, but with potentially more general state transition functions as well. Two control-theoretical issues are treated to this end: as a new paradigm, the construction of flat dynamical systems, and on the other hand, the notion of mortality of a set of matrices, a problem which is in general not decidable.

Keywords: Self-Synchronizing Stream Ciphers, flatness, LPV systems, structural analysis

1. INTRODUCTION

Synchronization has been an important topic in automatic control for years. Roughly speaking, by synchronization, it is meant correlated (according to given criteria) behaviors of at least two or more interconnected entities in virtual or physical networks. Throughout the past centuries, scientists have attempted to explain the emergence of order through the concept of synchronization. C. Huygens in 1665 can be considered as a pioneer. There is an outstanding number of examples of synchronized phenomena (see Strogatz (2003)) borrowed from nature, biology, neuroscience, physiology and more recently social networks.

Synchronization can be a very efficient way of tackling engineering issues as well. For example, there is a growing interest in cooperative control problems. Such problems involve several autonomous entities (also called agents) which try to collectively reach a global objective by a suitable connectivity or couplings. The related applications are mobile robots, unmanned and autonomous vehicles, satellites, air traffic control. Synchronization is also central in communication: video, broadcasting, Phase Lock Loop-based equipments. When synchronization must occur in a peer-to-peer communication setup without any external control, it is called self-synchronization. It turns out that self-synchronization is central in cryptography, more specifically, in symmetric cryptography involving the so-called Self-Synchronizing Stream Ciphers (SSSC) (see Menezes et al. (1996)). Such ciphers are based on generators which can take the form of dynamical systems operating on finite fields and must deliver complex sequences. Those sequences are used to scramble the information to be safely transmitted. For proper decryption, those sequences must be self-synchronized.

In this paper, we aim at addressing SSSC-based cryptography in terms of control theoretical concepts. The main result is that the concept of flatness together with graph-theory allows to provide a convenient and systematic way to construct general classes of SSSC, the ciphers being viewed as dynamical systems. The flatness on one hand and the SSSC on the other hand are detailed a little bit more below.

Differential flatness is a property of some continuous-time controlled dynamical systems introduced in Fliess et al. (1995). The counterpart for discrete-time systems is called difference flatness. For a flat discrete-time system, the state variables as well as the input are written as a function of the flat output (including forward and backward shifts in the output). Difference flatness has been first reported in Fliess and Marquez (2000); Sira-Ramirez and Agrawal (2004) although we should mention that a closely related notion, namely the dynamic feedback linearization, was addressed earlier in Aranda-Bricaire et al. (1996). Flatness-based control has been involved in many applications and has an outstanding interest in, for instance, trajectory planning as reported in Meurer (2011); Chamseddine et al. (2012), predictive control and constraint handling as detailed in De Donà et al. (2009); Kandler et al. (2012). The reader can also refer to the book of Sira-Ramirez and Agrawal (2004) for further applications. Since in this paper, only discrete-time finite

⋆ This work was supported by Research Grants ANR-13-INSE-0005-01 from the Agence Nationale de la Recherche in France

state dynamical systems are considered, throughout this paper, the word flatness will be often used for short without confusion.

Self-Synchronizing Stream Ciphers were patented in 1946. This self-synchronization property has many advantages and is especially relevant to group communications. Since 1960, specific SSSC have been designed and are still used to provide bulk encryption (for hertzian line, RNIS link, ...) in military applications or governmental radio mobile networks. In the early 90s, studies have been performed in Maurer (1991); Daemen et al. (1992) to propose secure designs of SSSC. These works have been followed by effective constructions (Daemen et al. (1992); Sarkar (2003); Daemen and Kitsos (2008)), but till now, all of these SSSC have been broken, which motivates the search of new constructions of SSSC. It is this issue that is investigated in this paper. It is shown here that the consideration of the special class of flat Linear Parameter Varying (LPV) systems, with the help of graph theory, allow for a convenient and systematic design of such ciphers.

The paper is organized as follows. Section 2 is devoted to the problem statement. The architecture of Self-Synchronizing Stream Ciphers is presented in terms of dynamical systems. In Section 3, the special class of LPV dynamical systems is presented, the definition of difference flatness is introduced and is particularized for LPV systems. An algebraic characterization in terms of state space matrix representation is provided. The connection between flatness and SSSC is made. In Section 4, an effective construction of SSSC is proposed. It is based on the interpretation of flatness in terms of the structure of a graph associated to the LPV system. A basic example is given to make a clear understanding of the method.

2. SELF-SYNCHRONIZING STREAM CIPHERS AND DYNAMICAL SYSTEMS

2.1 Generalities on stream ciphers

For a stream cipher, it must be given an alphabet A, that is, a finite set of basic elements named symbols. Hereafter, the index k will stand for the discrete-time. On the *transmitter* part, a plaintext (also called information or message) $u \in \mathcal{M}$ ($\mathcal{M}$ is the message space) consisting of a string of symbols $m_k \in A$ is encrypted according to an encryption function e which depends on a so-called running key (also called keystream) z_k which is invertible for any prescribed z_k. Hence, the ciphering is performed with

$$c_{k+b} = e(z_{k+b}, m_k) \tag{1}$$

where e is the ciphering function, m_k is the plaintext symbol and $c_k \in B$ is the ciphertext symbol which belongs to an alphabet B usually (and assumed hereafter) identical to A. The integer $b \geq 0$ stands for a potential delay between the plaintext m_k and the corresponding ciphertext c_{k+b}. This is explained by computational reasons, for instance pipelining (see Daemen and Kitsos (2005) for instance). Consequently, for stream ciphers, the way how to encrypt each plaintext symbol changes on each iteration. The resulting ciphertext $c \in \mathcal{C}$ ($\mathcal{C}$ is called the ciphertext space), a string of symbols c_k, is conveyed through a public channel to the *receiver*.
At the receiver side, the ciphertext c is decrypted according to a decryption function d which depends on the running key $\hat{z}_k$. The decryption function d obeys the following rule. For any two keystream symbols $\widehat{z}_{k+b}, z_{k+b}$, it holds that

$$\hat{m}_{k+b} := d(c_{k+b}, \widehat{z}_{k+b}) = m_k \text{ whenever } \hat{z}_{k+b} = z_{k+b}. \tag{2}$$

From (2), it is clear that, beyond the equality of the secret keys, the running keys z_k and $\hat{z}_k$ must be synchronized for a proper decryption. The distinct classes of stream ciphers differ each other by the way on how the keystreams are generated and synchronized. The generators delivering the keystreams will be parametrized by a secret key denoted in the sequel by θ. Next, we detail the special class of stream ciphers called Self-Synchronizing Stream Ciphers.

2.2 Keystream generators for Self-Synchronizing Stream Ciphers

A well-admitted approach to generate the keystreams has been first suggest in Maurer (1991). It is based on the use of state automata with finite input memory. This is typically the case in the cipher Moustique Kasper et al. (2004). At the ciphering side, the automaton delivering the keystream takes the form:

$$\begin{cases} q_{k+1} = g_\theta(q_k, c_{k+b}) \\ z_{k+b} = h_\theta(q_k) \end{cases} \tag{3}$$

where q_k is the internal state. As previously stressed, the delay b is due to the fact that the output (also called filtering) function h is pipelined with an architecture involving b layers. If such an automaton has a finite input memory, it means that by iterating (3) a finite number of times, there exists a function l_θ and a finite integer M such that

$$q_k = l_\theta(c_{k+b-1}, \ldots, c_{k+b-M}) \tag{4}$$

and thus

$$z_{k+b} = h_\theta(l_\theta(c_{k+b-1}, \ldots, c_{k+b-M})) \tag{5}$$

Actually, the fact that the keystream symbol can be written in the general form involving a function f_θ

$$z_{k+b} = f_\theta(c_{k-\ell}, \ldots, c_{k-\ell'}) \tag{6}$$

is a common feature of all the SSSC. The quantities ℓ and ℓ' are integers in $\mathbb{Z}$. Equation (6) is called canonical equation.

Remark 1. The outcome of implementing the recursive form (3) instead of directly implementing the canonical form (6) is that we can obtain complex nonlinear functions f_θ by implementing simpler nonlinear functions g_θ. The complexity results from the successive iterations which act as composition operations.

At the deciphering side, the automaton takes the form:

$$\begin{cases} \widehat{q}_{k+1} = g_\theta(\widehat{q}_k, c_{k+b}) \\ \widehat{z}_{k+b} = h_\theta(\widehat{q}_k) \end{cases} \tag{7}$$

where $\widehat{q}_k$ is the internal state. Following the same reasoning, since g_θ corresponds to the state transition function of an automatom with finite input memory, it is clear that after a transient time of maximal lenghth equal to M, it holds that, for $k \geq M$,

$$\widehat{q}_k = q_k \text{ and } \widehat{z}_{k+b} = z_{k+b} \tag{8}$$

Hence, since the generators synchronize automatically after at most M iterations, the decryption is automatically and properly achieved after at most M iterations too. No specific synchronizing protocol between the cipher and decipher is needed. This explains the terminology Self-Synchronizing Stream Ciphers. The quantity M is called the synchronization delay.

To obtain a finite input memory feature, the solutions proposed in the open literature (see Daemen (1995) for example), call for state transition functions g_θ in the form of shifts or T–functions (T for Triangle), which are functions that propagate dependencies in one direction only. The aim of the paper is twofold: showing that it is possible to construct automata with finite input memory with more general state transition functions than T–functions and giving a systematic methodology of construction. The construction is based on the property of flatness, a structural property borrowed from control theory. This property is considered for the class of Linear Parameter-Varying (LPV) systems as motivated in next section.

3. LPV SYSTEMS AND FLATNESS

The aim of this section is to illustrate the potential interest of LPV flat dynamical systems for cryptographic issues.

3.1 Definition of difference flatness for LPV systems

A Single Input Single Output (SISO) Linear Parameter-Varying (LPV) system denoted by Σ_ρ, defined over a field $\mathbb{F}$, is described by the following state space representation:

$$\Sigma_\rho : \left\{ \begin{array}{l} x_{k+1} = A_{\rho(k)} x_k + B_{\rho(k)} u_k \\ y_k = C_{\rho(k)} x_k + D_{\rho(k)} u_k \end{array} \right. \quad (9)$$

where $k \in \mathbb{N}$ stands for the discrete time, $x_k \in \mathbb{F}^n$ is the state vector, $u_k \in \mathbb{F}$ is the input, $y_k \in \mathbb{F}$ is the output. The matrices $A \in \mathbb{F}^{n\times n}$, $B \in \mathbb{F}^{n\times 1}$, $C \in \mathbb{F}^{1\times n}$ and $D \in \mathbb{F}^{1\times 1}$ are respectively the dynamical matrix, the input matrix, the output matrix and the direct transfer matrix. Such a system is called Linear Parameter-Varying because it is written with a linear dependency with respect to the state vector. The set of all the varying parameters of A, B, C and D are collected on a vector denoted by $\rho(k) = \left[\, \rho^1(k), \rho^2(k), ..., \rho^{L_\rho}(k) \,\right] \in \mathbb{F}^{L_\rho}$ where L_ρ is the total number of non zero (possibly varying) entries. The matrices A, B, C and D do not necessarily depend on all the parameters $\rho^i(k)$. Such systems can exhibit nonlinear dynamics. Indeed, the nonlinearity is obtained by defining the varying parameters $\rho^i(k)$ as nonlinear functions φ_i of the output y_k (or a finite number of shifts) $\rho^i(k) = \varphi_i(y_k, y_{k-1}, \cdots)$. Let us notice that the notation $\rho^i(k)$ (usual in the literature for LPV systems) is somehow abusive because it does not reflect an explicit dependency with respect to the time k but on quantities indexed with k.

Now, let us focus on the property of flatness. For a non negative integer k_0, a sequence $\{\rho(k_0), \rho(k_0+1), \ldots\}$ will be called a realization and denoted with ρ. The definition of a flat LPV system is given below in a generic sense, that is for almost every realization ρ.

Definition 1. The system (9) is said to be generically flat if, for almost every realization ρ, there exists a variable y_k, called a flat output, such that all system variables can be expressed as a function of the flat outputs and a finite number of its backward and forward shifts. In other words, there exist two functions $\mathcal{F}_\rho$ and $\mathcal{G}_\rho$ parametrized by ρ such that

$$\left\{ \begin{array}{l} x_k = \mathcal{F}_\rho\big(y_{k+k_\mathcal{F}}, \ldots, y_{k+k'_\mathcal{F}}\big) \\ u_k = \mathcal{G}_\rho\big(y_{k+k_\mathcal{G}}, \ldots, y_{k+k'_\mathcal{G}}\big) \end{array} \right. \quad (10)$$

where $k_\mathcal{F}$, $k'_\mathcal{F}$, $k_\mathcal{G}$ and $k'_\mathcal{G}$ are $\mathbb{Z}$-valued integers.

Remark 2. Let us consider the input u_k as the plaintext and the output y_k as the ciphertext. The first equality in (10) shows that if the output y_k of (9) is flat, an interesting correspondence with (4) can be made. Indeed, x_k can be used as a quantity from which, through a filtering function, the keystream symbol z_k of an SSSC can be derived. The second equation shows that it is possible to recover the input u_k from a finite number of shifted outputs y_k. This is typically the central notion behind the decryption part of an SSSC.

The next subsection is devoted to the conditions which must be fulfilled by (9) to guarantee its flatness property. Then, it will be shown that from flat LPV systems, automata with finite input memory like (3), and thus, having the self synchronizing property, can be naturally derived, taking into account Remark 2. Furthermore, it will be showhn that more general architectures of SSSC can be obtained. By more general, it is understood involving state transitions functions not restricted to T–functions.

3.2 Conditions for flat outputs

First, it must be pointed out that for a given dynamics (first equation of (9)), y_k is not necessarily a flat output. It depends on the matrices C and D. Besides, for some specific matrices A and B, it may happen that none of the matrices C and D yield to a flat output. As a result, the conditions which guarantee that an output of (9) is flat must be expressed in terms of properties verified by the 4-uple of state matrices A, B, C and D. Before proceeding further, it is necessary to deal with the relative degree of (9). We recall below a usual general definition.

Definition 2. The relative degree of a discrete-time dynamical system is the minimal number r of iterations such that its output y_{k+r} at time $k+r$ is sensitive to its input u_k.

Now, let us particularize this general definition to the system defined by equation (9). To this end, let us denote, for $k_2 \geq k_1$, by $\prod_{l=k_2}^{k_1} A_{\rho(l)}$ the product of the matrices $A_{\rho(i)}$, from k_2 down to k_1, and $\prod_{l=k_2}^{k_1} A_{\rho(l)} = \mathbf{1}_n$ if $k_2 < k_1$, and introduce the quantity $\mathcal{T}^{i,j}_{\rho(k)}$ defined for $j \leq i$ as

$$\begin{array}{l} \mathcal{T}^{i,j}_{\rho(k)} = C_{\rho(k+i)} \prod_{l=k+i-1}^{k+j+1} A_{\rho(l)} B_{\rho(k+j)} \text{ if } j \leq i-1 \text{ and} \\ \mathcal{T}^{i,i}_{\rho(k)} = D_{\rho(k+i)} \end{array} \quad (11)$$

From (9), it holds that $y_{k+1} = C_{\rho(k+1)} A_{\rho(k)} x_k + \mathcal{T}^{1,0}_{\rho(k)} u_k + \mathcal{T}^{0,0}_{\rho(k+1)} u_{k+1}$. Then, by iterating the output y_k for $i \geq 1$ and noticing that $\mathcal{T}^{i,j}_{\rho(k+1)} = \mathcal{T}^{i+1,j+1}_{\rho(k)}$, it follows that

$$y_{k+i} = C_{\rho(k+i)} \prod_{l=k+i-1}^{k} A_{\rho(l)} x_k + \sum_{j=0}^{i} \mathcal{T}^{i,j}_{\rho(k)} u_{k+j} \quad (12)$$

Hence, if (9) admits a finite relative degree r, it follows from Definition 2 that:

(1) $r = 0$ if $\mathcal{T}^{0,0}_{\rho(k)} \neq 0$ for all k
(2) $r < \infty$ is the least integer s such that for all k

$$\begin{array}{l} \mathcal{T}^{i,j}_{\rho(k)} = 0 \quad \text{for } i = 0, \ldots, s-1 \text{ and } j = 0, \ldots, \ i, \\ \mathcal{T}^{s,0}_{\rho(k)} \neq 0 \end{array} \tag{13}$$

Hence, it holds that

$$y_{k+r} = C_{\rho(k+r)} \prod_{l=k+r-1}^{k} A_{\rho(l)} x_k + \mathcal{T}^{r,0}_{\rho(k)} u_k \tag{14}$$

Proposition 1. If the LPV system (9) has a finite relative degree r, the following condition that must hold for a positive integer K

$$P_{\rho(k+K-1:k+K-1+r)} P_{\rho(k+K-2:k+K-2+r)} \cdots P_{\rho(k:k+r)} = 0 \tag{15}$$

with

$$P_{\rho(k:k+r)} = A_{\rho(k)} - B_{\rho(k)} (\mathcal{T}^{r,0}_{\rho(k)})^{-1} C_{\rho(k+r)} \prod_{l=k+r-1}^{k} A_{\rho(l)} \tag{16}$$

is equivalent to that y_k is a flat output.

Proof. If the LPV system (9) has a finite relative degree r, the following dynamical system can always be defined since $\mathcal{T}^{r,0}_{\rho(k)}$ is invertible.

$$\left\{ \begin{array}{l} \widehat{x}_{k+1} = P_{\rho(k:k+r)} \widehat{x}_k + B_{\rho(k)} (\mathcal{T}^{r,0}_{\rho(k)})^{-1} y_{k+r} \\ \widehat{u}_{k+r} = (\mathcal{T}^{r,0}_{\rho(k)})^{-1} (C_{\rho(k+r)} \prod_{l=k+r-1}^{k} A_{\rho(l)} \widehat{x}_k - y_{k+r}) \end{array} \right. \tag{17}$$

Iterating $K-1$ times the first equation of (17) yields

$$x_k = P_{\rho(k-1:k-1+r)} \cdots P_{\rho(k-K:k-K+r)} x_{k-K} + \sum_{i=1}^{K-1} \Big[\prod_{j=0}^{i-1} P_{\rho(k-j-1:k+r-j-1)} \Big] (\mathcal{T}^{r,0}_{\rho(k-1-i)})^{-1} B_{\rho(k-i-1)} y_{k-i-1+r} \tag{18}$$

Let us define $\varepsilon_k = x_k - \widehat{x}_k$. By simple manipulations, it can be shown that ε_k verifies $\varepsilon_{k+1} = P_{\rho(k:k+r)} \varepsilon_k$. Hence, after a finite transient time equal to K, since (15) holds, one has $\widehat{x}_k = x_k$. Hence, the state vector x_k exclusively depends on shifted outputs if and only if (15) holds and x_k reads

$$x_k = \sum_{i=0}^{K-1} \Big[\prod_{j=0}^{i-1} P_{\rho(k-j-1:k+r-j-1)} \Big] (\mathcal{T}^{r,0}_{\rho(k-1-i)})^{-1} B_{\rho(k-i-1)} y_{k-i-1+r} \tag{19}$$

which gives the explicit function $\mathcal{F}_\rho$ involved in (10). Letting $\omega_k = u_k - \widehat{u}_{k+r}$ and following the same reasoning, it is shown that u_k exclusively depends on shifted outputs. The explicit form of $\mathcal{G}_\rho$ involved in (10) is obtained by substituting (19) into the second equation of (17). It shows that (9) is flat with y_k as flat output.

3.3 Connection between flatness and SSSC

In this subsection, the connection between flat LPV systems and SSSC is established.

Proposition 2. If the LPV system (9) has relative degree r and is flat, then the finite state automata given by

$$\left\{ \begin{array}{l} q_{k+1} = P_{\rho(k:k+r)} q_k + B_{\rho(k)} (\mathcal{T}^{r,0}_{\rho(k)})^{-1} y_{k+r} \\ z_{k+r} = C_{\rho(k+r)} \prod_{l=k+r-1}^{k} A_{\rho(l)} q_k \end{array} \right. \tag{20}$$

along with

$$y_{k+r} = z_{k+r} + \mathcal{T}^{r,0}_{\rho(k)} u_k \tag{21}$$

and

$$\left\{ \begin{array}{l} \widehat{q}_{k+1} = P_{\rho(k:k+r)} \widehat{q}_k + B_{\rho(k)} (\mathcal{T}^{r,0}_{\rho(k)})^{-1} y_{k+r} \\ \widehat{z}_{k+r} = C_{\rho(k+r)} \prod_{l=k+r-1}^{k} A_{\rho(l)} \widehat{q}_k \end{array} \right. \tag{22}$$

along with

$$\widehat{u}_{k+r} = (\mathcal{T}^{r,0}_{\rho(k)})^{-1} (y_{k+r} - \widehat{z}_{k+r}) \tag{23}$$

define an SSSC.

Proof. If (9) has relative degree r and is flat, (15) holds and thus, (20) is well defined and can be identified with (3) while (22) is also well defined and can be identified with (7). The property (15), that is flatness, ensures that (20) and (22) are automata with finite input memory. The identification of (21) with (1) and the identification of (23) with (2) gives respectively the encryption and decryption functions.

The following correspondences hold:

- u_k plays the role of m_k (plaintext symbol)
- y_k plays the role of c_k (ciphertext symbol)
- $z_{k+r} = C_{\rho(k+r)} \prod_{l=k+r-1}^{k} A_{\rho(l)} q_k$ is the keystream symbol of the cipher
- $\widehat{z}_{k+r} = C_{\rho(k+r)} \prod_{l=k+r-1}^{k} A_{\rho(l)} \widehat{q}_k$ is the keystream symbol of the decipher
- r plays the role of b (delay)
- the function $(z_{k+r}, u_k) \mapsto z_{k+r} + \mathcal{T}^{r,0}_{\rho(k)} u_k$ plays the role of e (encryption function)
- the function $(\widehat{z}_{k+r}, y_{k+r}) \mapsto (\mathcal{T}^{r,0}_{\rho(k)})^{-1} (y_{k+r} - \widehat{z}_{k+r})$ plays the role of d (decryption function)
- K plays the role of M (synchronization delay)

The nonlinearity is obtained by defining the values of the varying parameters $\rho^i(k)$ as nonlinear functions of the output y_k (or a finite number of shifts), implemented in the form of so-called S-boxes φ_i: $\rho^i(k) = \varphi_i(y_k, y_{k-1}, \cdots)$

Remark 3. The encryption and decryption functions e and d are quite simple. This is a common feature for SSSC. For example in the Boolean case, those functions are often nothing but the exclusive or. Actually, the security is essentially based on the properties of the keystream sequences.

Let us comment on the state transitions matrices $P_{\rho(k:k+r)}$. It is worth pointing out that $P_{\rho(k:k+r)}$ is not imposed to be triangular, that is may represent a larger class than strict T–functions. However, choosing non triangular matrices fulfilling (20) is an intricate problem. Indeed, it is closely related to the notion of *mortality*. It is said that a set of matrices is mortal if the zero matrix can be expressed as product of finite length of matrices. And yet, from the papers Paterson (1970); Blondel and Tsitsiklis (1997); Bournez and Branicky (2002), except from some very special cases, the problem of checking mortality of a set of matrices is unsolvable. Furthermore, the problem under consideration here is more intricate since the matrices $P_{\rho(k:k+r)}$ are not given but should be chosen. In other

words, we are not concerned with analysis but with synthesis. The method proposed here, that is the design a flat LPV system (9) followed by deriving the automaton (20) (which will be, because of its flatness property, with finite input memory and so self-synchronizing) constitutes an efficient alternative to a very challenging direct design of the automaton (20). This approach is new in cryptography and provides a general framework. However, from a control theory point of view, the issue of designing a flat LPV system is a new paradigm. Indeed, in automatic control, we are usually given a system and we have to check whether a system is flat or not and to check for the flat outputs. The aim of the next section is to show that a graph-oriented approach can be a solution to that issue.

4. CONSTRUCTION OF SSSC: STRUCTURAL CONSIDERATIONS

4.1 Structured systems and digraphs

The core idea is that the LPV system (9) can be considered as an admissible realization of a corresponding structured linear system. A structured linear system is a system only defined by the sparsity pattern of the state space realization matrices. In other word, in a structured linear system, we distinguish between the entries that are fixed and known to be zero and the other ones that can take any value in $\mathbb{F}$, including the ones which are varying. As a simple example, let us consider the setting

$$A_{\rho(k)} = \begin{pmatrix} a & 0 \\ 1 & \rho^1(k) \end{pmatrix} \text{ and } B_{\rho(k)} = \begin{pmatrix} \rho^2(k) \\ 0 \end{pmatrix}$$

where a is a constant element in $\mathbb{F}$, $\rho^1(k)$ and $\rho^2(k)$ are varying parameters in $\mathbb{F}$. The dynamical matrix and the input matrix, denoted respectively by I_A and I_B, of the corresponding structured linear system read:

$$I_A = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix},\ I_B = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

As a consequence, if the structural linear system corresponding to (9) is flat, the flatness will hold for any realization ρ. Hence, the challenge is to define a methodology to construct flat linear structural systems. A graph-based approach is suggested below to that purpose. Let us recall some basic on graphs. A digraph $\mathcal{G}(\Sigma_\rho)$ describing the structured linear system associated to the state equations (9), is the combination of a vertex set $\mathcal{V}$ and an edge set $\mathcal{E}$. The vertices represent the state and the input components of Σ_ρ while the edges model the dynamic relations between these variables. One has $\mathcal{V} = \mathbf{X} \cup \{\mathbf{u}\}$ where $\mathbf{X}$ is the set of state vertices defined as $\mathbf{X} = \{\mathbf{x^1}, \ldots, \mathbf{x^n}\}$ and $\mathbf{u}$ is the input vertex. The edge set is $\mathcal{E} = \mathcal{E}_A \cup \mathcal{E}_B$, with $\mathcal{E}_A = \{(\mathbf{x^j}, \mathbf{x^i}) \,|A(i,j) \neq 0\}$ and $\mathcal{E}_B = \{(\mathbf{u}, \mathbf{x^i}) \,|B(i) \neq 0\}$ where $A(i,j)$ and $B(i)$ denote the $(i,j)th$ element and the ith element of the matrix A and B respectively.

- A directed path P is a sequence of successive edges directed in the same direction which connect a sequence of vertices. It is said that the path P covers a vertex if this vertex is the begin or the end vertex of one of the edges of P;
- In a directed path from a vertex $\mathbf{v^i}$ to a vertex $\mathbf{v^j}$, it is said that $\mathbf{v^j}$ is a successor of $\mathbf{v^i}$ and conversely, $\mathbf{v^i}$ is a predecessor of $\mathbf{v^j}$;

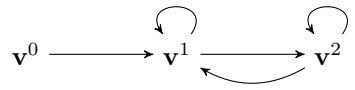

Fig. 1. Digraph obtained for $n = 2$, $r = 2$, $n_a = n_M = 5$. The flat output is $y^F = x^2$ and corresponds to the vertex $\mathbf{v^2}$

- A directed path is simple when every vertex occurs only once in this path;
- The length of a directed path P is equal to the number of the edges involved in P. We denote by $\ell(\mathbf{v^i}, \mathbf{v^j})$ the minimal length of a path linking $\mathbf{v^i}$ to $\mathbf{v^j}$;
- $V_{ess}(\mathbf{v^i}, \mathbf{v^j})$ is the set of vertices, called essential vertices from $\mathbf{v^i}$ to $\mathbf{v^j}$, which are common to all the paths linking $\mathbf{v^i}$ to $\mathbf{v^j}$ whenever at least one path exists.

We recall from Millérioux and Boukhobza (2015) the necessary and sufficient conditions which must be satisfied by a vertex $\mathbf{v^i}$ ($i \in \{1, \ldots, n\}$) of the digraph $\mathcal{G}(\Sigma_\rho)$ to be associated to a flat output $y^F = x^i$ ($i \in \{1, \ldots, n\}$) of (9). In such a case, the output state space matrix $C_{\rho(k)} = C$ is constant and has zero entries except the entry located at the column number i which is equal to one. The direct transfer matrix $D_{\rho(k)} = D$ is the zero matrix.

Theorem 3. The output y^F of a structured linear system associated to the vertex $\mathbf{v^F} \in \mathbf{X}$ is generically a flat output iff, in the associated digraph $\mathcal{G}(\Sigma_\rho)$, all the three following conditions hold:
C0. $\mathbf{v^F}$ is a successor of $\mathbf{u}$;
C1. All the simple paths from $\mathbf{u}$ to $\mathbf{v^F}$ have the same length equal to $\ell(\mathbf{u}, \mathbf{v^F})$;
C2. All the cycles cover at least an element of $V_{ess}(\mathbf{u}, \mathbf{v^F})$.

4.2 Construction of SSSC based on the digraph

Conditions **C0**-**C2** are instrumental for our purpose. Indeed, systematic constructions of digraphs fulfilling **C0**-**C2** can be easily proposed. An example of construction is given in an external document [1]. The digraph is parametrized by the triplet (n, r, n_a). The dimension n of the system (9) corresponds to the number of vertices of the graph minus one (the vertex assigned to the input). The relative degree r fulfilling (13) gives the number of edges in the main direct path. The integer n_a defines the number of edges in the digraph $\mathcal{G}(\Sigma_\rho)$ and can be chosen freely chosen provided it is less than the maximal number n_M of edges (see the external document).
Let us consider the graph depicted in Figure 1 which results from the construction (see the external document) for $n = 2$ and $n_a = n_M = 5$ (maximal number of edges) and $r = 2$. The relative degree r being equal to 2, it means that the component x^2 of the state vector of the corresponding LPV system (9) will be the flat output y_k, that is the ciphertext. Hence, $C_{\rho(k)} = C = [0\ 1]$ and $D_{\rho(k)} = 0$. Then, given a digraph $\mathcal{G}(\Sigma_\rho)$, characterized by the triplet (n, r, n_a), we can derive the matrices I_A and I_B of the structured linear system from the adjacency matrix. The adjacency matrix, denoted with $\mathcal{I}$, of the digraph $\mathcal{G}(\Sigma_\rho)$, is the $(n+1) \times (n+1)$ matrix whose each entry $(a)_{ij}$ is defined as follows for $1 \leq i, j \leq n$

[1] see the website `https://sites.google.com/site/autocrypt/`

$$(a)_{ij} = \begin{cases} 1 \text{ if there exists an edge from } \mathbf{v^j} \text{ to } \mathbf{v^i} \\ 0 \text{ otherwise} \end{cases} \quad (24)$$

Hence, the adjacency matrix associated to $\mathcal{G}(\Sigma_\rho)$ is given by:

$$\mathcal{I} = \left(\begin{array}{c|c} 0 & I_B^t \\ \hline 0 & \\ \vdots & I_A^t \\ 0 & \end{array}\right) \quad (25)$$

where I_A^t and I_B^t stand respectively for the transpose of I_A and I_B. For our example, the adjacency matrix $\mathcal{I}$ and the structured matrices I_A and I_B as defined by (25) are

$$\mathcal{I} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix},\ I_A = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix},\ I_B = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad (26)$$

Finally, each of the non zero entries of I_A and I_B can be potentially replaced by a nonlinear function $\rho^i(k)$ to construct the matrices $A_{\rho(k)}$ and $B_{\rho(k)}$ of (9). Let us choose, for example, the varying parameters $\rho^i(k)$ for $i = 1, 2, 3$ of the first three entries of A as constant and equal to 1. This finally leads to the following matrices $A_{\rho(k)}$ and $B_{\rho(k)}$:

$$A_{\rho(k)} = \begin{pmatrix} 1 & 1 \\ 1 & \rho^4(k) \end{pmatrix} \text{ and } B_{\rho(k)} = B = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

The varying parameter $\rho^4(k)$ is in practice implemented in the form of a so-called S-box of which entry is the flat output y_k and possibly a finite number of backwards iterates. By construction, any nonlinearity would lead to a flat LPV system and so to an SSSC. The approach proposed here gives thereby a family of SSSC. The design of a specific SSSC is completed by deriving the equations of Proposition 2. Actually, it can be shown that even for this simple example, the state transition matrix $P_{\rho(k:k+2)}$ is non triangular, which corroborates that this method allows to provide a novel class of SSSC.

5. CONCLUSION

A systematic and general construction of Self Synchronizing Stream Ciphers based on flat Linear Parameter Varying (LPV) dynamical systems has been proposed. It is based on algebraic conditions guaranteeing flatness of the LPV system and so the self-synchronizing property, those conditions being interpreted in terms of the structure of the graph associated to the LPV dynamical system. It has been shown that such an approach allows to enlarge the existing classes of SSSC, more precisely, to obtain non triangular SSSC. Two control-theoretical issues have been treated to this end: as a new paradigm, the construction of flat dynamimal systems and the notion of mortality, which is in general not decidable.

From a practical point of view, a large dimension must be chosen to allow a sufficiently complex architecture for the sake of security. The proposed graph-based approach is well suited to deal with high dimensions. The dimension $n = 40$, which is compatible with security issues (not treated in this paper because out of the scope), has been tested with success on a real-world equipement (see the website `https://sites.google.com/site/autocrypt/` for technical aspects).

Acknowledgments This work was supported by Research Grants ANR-13-INSE-0005-01 from the Agence Nationale de la Recherche.

REFERENCES

Aranda-Bricaire, E., Kotta, U., and Moog, C.H. (1996). Linearization of discrete-time systems. *SIAM J. Control Optim.*, 34(6).

Blondel, V.D. and Tsitsiklis, J.N. (1997). When is a pair of matrices mortal? *Information Processing Letters*, 63, 283–286.

Bournez, O. and Branicky, M. (2002). The mortality problem for matrices of low dimensions. *Theory of Computing Systems*, 35(4), 433–448.

Chamseddine, A., Zhang, Y., Rabbath, C., Join, C., and Theillol, D. (2012). Flatness-based trajectory planning/replanning for a quadrator unmanned aerial vehicle. *IEEE Trans. on Aerospace and electronic systems*, 35(5), 1667–1671.

Daemen, J. (1995). *Cipher and Hash function design, strategies based on linear and differential cryptanalysis.* PhD Thesis, Katholieke Universiteit Leuven.

Daemen, J., Govaerts, R., and Vandewalle, J. (1992). On the design of high speed self-synchronizing stream ciphers. In *Proc. of the ICCS/ISITA'92 conference*, volume 1, 279–283. Singapore.

Daemen, J. and Kitsos, P. (2005). The self-synchronizing stream cipher mosquito: eSTREAM documentation, version 2. Technical report, e-Stream Project. Available at: www.ecrypt.eu.org/stream/p3ciphers/mosquito/mosquito.pdf.

Daemen, J. and Kitsos, P. (2008). The self-synchronizing stream cipher moustique. In *New Stream Cipher Designs - The eSTREAM Finalists*, 210–223. Springer-Verlag Berlin Heidelberg.

De Donà, J., Suryawan, F., Seron, M., and Lévine, J. (2009). A flatness-based iterative method for reference trajectory generation in constrained nmpc. In L. Magni, D. Raimondo, and F. Allgöwer (eds.), *Nonlinear Model Predictive Control*, volume 384 of *Lecture Notes in Control and Information Sciences*, 325–333. Springer Berlin Heidelberg.

Fliess, M., Levine, J., Martin, P., and Rouchon, P. (1995). Flatness and defect of non-linear systems: introductory theory and examples. *Int. Jour. of Control*, 61(6), 1327–1361.

Fliess, M. and Marquez, R. (2000). Toward a module theoretic approach to discrete-time linear predictive control. *International Journal of Control*, 73, 606–623.

Kandler, C., Ding, S., Koenings, T., and Weihold, N. (2012). A differential flatness based model predictive control approach. In *Proc. of IEEE International Conference on Control Applications (CCA)*. Dubrovnik.

Kasper, E., Rijmen, V., E.Bjorstad, Rechberger, C., Robshaw, M., and Sekar, G. (2004). Correlated keystreams in moustique. Technical report, ESTREAM Project.

Maurer, U.M. (1991). New approaches to the design of self-synchronizing stream cipher. *Advance in Cryptography, In Proc. Eurocrypt '91, Lecture Notes in Computer Science*, 548–471.

Menezes, A.J., Oorschot, P.C., and Vanstone, S.A. (1996). *Handbook of Applied Cryptography.* CRC Press.

Meurer, T. (2011). Flatness-based trajectory planning for diffusion-reaction systems in a parallelepipedon - a spectral approach. *Automatica*, 47(5), 935 – 949.

Millérioux, G. and Boukhobza, T. (2015). Characterization of flat outputs for lpv discrete-time systems: a graph-oriented approach. In *54th Conference on Decision and Control (CDC 2015)*. Osaka, Japan.

Paterson, M.S. (1970). Unsolvability in 3×3 matrices. *Studies in Applied Mathematics*, 49(1), 105–107.

Sarkar, P. (2003). Hiji-bij-bij: A new stream cipher with a self-synchronizing mode of operation. *IACR Cryptology ePrint Archive*, 2003, 14.

Sira-Ramirez, H. and Agrawal, S.K. (2004). *Differentially Flat Systems.* Marcel Dekker, New York.

Strogatz, S.H. (2003). *SYNC: The Emerging Science of Spontaneous Order.* Penguin Group.