Available online at www.sciencedirect.com

ScienceDirect

IFAC-PapersOnLine 49-9 (2016) 019–024

Graphical characterization of the set of all flat outputs for structured linear discrete-time systems

Taha Boukhobza *,** **Gilles Millérioux** *,**

* *Université de Lorraine, Centre de Recherche en Automatique de Nancy, UMR 7039, Vandoeuvre-lès-Nancy, F-54506, France (e-mail: {taha.boukhobza, gilles.millerioux}@univ-lorraine.fr.*
** *CNRS, Centre de Recherche en Automatique de Nancy, UMR 7039, Vandoeuvre-lès-Nancy, F-54506, France*

Abstract: This paper deals with the characterization of flat outputs for structured linear discrete-time systems. More precisely, considering a structured discrete-time linear system, we provide a complete set of the flat outputs using constructive polynomial complexity order algorithms. The proposed method is simple to implement. It is based on usual algorithms dedicated to the computation of successors and predecessors of vertex subsets and to the computation and the ordering of strongly connected components in a digraph.

© 2016, IFAC (International Federation of Automatic Control) Hosting by Elsevier Ltd. All rights reserved.

Keywords: Flatness, Structured Linear Discrete-Time Systems, Structural Analysis, Graphs.

1. INTRODUCTION

Flatness is a control-theoretical concept introduced in Fliess et al. (1995). For a flat continuous-time system, flatness is equivalent to a complete parametrization of all system variables (inputs and states) in terms of a finite number of independent variables and a finite number of their time derivatives. Those variables are called flat outputs. For discrete-time systems, a specific treatment must be considered insofar as derivatives are replaced by shifted outputs. Hence, for a flat system, the state variable as well as the input of discrete-time systems can be written as some function of the output (including forward and backward shifts in the output). Flatness-based control has been involved in many applications and has an outstanding interest in trajectory planning (Meurer (20114), Chamseddine et al. (2012)), predictive control and constraint handling (Doná et al. (2009),Kandler et al. (2012)). For discrete-time systems, it has been investigated in a much lesser extent in comparison with continuous-time systems. As some exceptions, we can quote Millérioux et al. (2009),Parriaux et al. (2013), Sato (2012) and Kaldmae et al. (2013).

The point is that flatness-based control requires the knowledge of the flat outputs of the system. There exist substantial results for checking whether or not a given output is a flat output. On the other hand, the characterization of the set of all flat outputs for a given system is a challenging problem. A brute approach consists in attempting to directly meet the definition, that is expressing the input and the state vector as a function exclusively involving derivatives of the output in the continuous case or shifts of the output in the discrete-time case. However, this combinatorial method is clearly heavy and not well adapted for large scale systems. A second approach is to test a posteriori whether a given output is flat or not. As a more advanced method, the work in Lvine et al. (2003) proposed an approach to parametrize all the flat outputs by resorting to some so-called defining matrices. However, this work deals with continuous-time systems. In Sira-Ramirez et al. (2004), the authors also characterize flat outputs for linear discrete SISO systems starting mainly from the transfer function or from a controllable realization of a given linear discrete system. One of the main differences with our proposed result is that we work with structured and so numerically non-fixed systems which ar not necessarily in some given canonical form. Moreover, contrary to Sira-Ramirez et al. (2004), we are not concerned with causality because of the kind of application we address does not require such constraint.

In this paper, we propose to use a graph-oriented approach to characterize all the flat outputs for linear structured SISO discrete-time systems. It is shown that it is computationally more efficient than an exhaustive approach. Let us point out that graph-oriented approaches have been used over the years with success to characterize many structural properties of linear systems (Dion et al. (2003)), bilinear systems (Boukhobza et al. (2007)) and switching systems (Boukhobza (2012)).

The paper is organized as follows. Section 2 is devoted to the problem statement. The digraph representation of a structured linear system and some corresponding definitions are recalled in Section 3. The main result is provided in Section 4. In a first step, the graphical conditions given in Millérioux et al. (2015) are recalled to check whether or not a given output is flat. Then, in a second step and as the main result, an exact characterization of all the flat outputs is provided. This characterization is constructive and uses well-known algorithms, commonly used for finding successors and predecessors of vertex subsets or for computing and ordering strongly connected components in a digraph. It is shown that the approach is simple to implement, has exponential complexity and is less complex than the exhaustive search. Section 5 is devoted to an illustrative example. Finally, some concluding remarks and perspectives are given in Section 6.

2405-8963 © 2016, IFAC (International Federation of Automatic Control) Hosting by Elsevier Ltd. All rights reserved.
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2016.07.481

2. PROBLEM STATEMENT

Consider the following structured SISO linear discrete-time system

$$\Sigma_\Lambda : x(k+1) = Ax(k) + Bu(k) \tag{1}$$

where $x \in \mathbb{R}^n$ is the state vector and $u \in \mathbb{R}$ the input. The matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times 1}$ are respectively the dynamical and input matrices.

By structured system, it is meant a system where it is assumed only the sparsity pattern of the matrices A and B. Hence, we distinguish between the entries that are fixed and known to be zero and the other ones that can take any value in $\mathbb{R}$. The latter ones are denoted with λ_i and collected on a vector $\Lambda = (\lambda_1, \lambda_2, \ldots, \lambda_h)^T$ belonging to the parameter space $\Omega_\Lambda \subseteq \mathbb{R}^h$. We obtain a so-called admissible realization of the structured system Σ_Λ. A property is true generically (van der Woude (1999)) if it is true for almost all the realizations of the structured system Σ_Λ. Hereafter, "for almost all the realizations" will be understood (see Dion et al. (2003); van der Woude (1999)) as "for all parameters $(\lambda_1, \lambda_2, \ldots, \lambda_h)^T \in \Omega_\Lambda$ except for those lying in a some proper algebraic variety in Ω_Λ" . The proper algebraic variety for which the property is not true is the zero set of some non-trivial polynomial with real coefficients in the h system parameters $\lambda_1, \lambda_2, \ldots, \lambda_h$ or equivalently an algebraic variety which has a Lebesgue measure which is zero.

Let us give the definition of flatness for Σ_Λ.

Definition 1. The structured linear discrete-time system Σ_Λ is said to be generically flat if, for almost all its realizations, there exists a set of p independent variables y referred to as flat outputs, such that all system variables can be expressed as a function of the flat output and a finite number of its backward and/or forward shifts. In particular, there exist two functions $\mathcal{F}$ and $\mathcal{G}$ such that

$$\begin{cases} x(k) = \mathcal{F}\big(y(k+k_\mathcal{F}), \ldots, y(k+k'_\mathcal{F})\big) \\ u(k) = \mathcal{G}\big(y(k+k_\mathcal{G}), \ldots, y(k+k'_\mathcal{G})\big) \end{cases} \tag{2}$$

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

The objective of the paper is to provide all the flat outputs in the specific form $y(k) = x_i(k)$ or $y(k) = u(k)$ for the structured linear discrete-time system Σ_Λ. The method is based on a graph-oriented approach, to be known as well suited and efficient, when dealing with structured systems.

3. DIGRAPH REPRESENTATION AND DEFINITIONS

Digraph $\mathcal{G}(\Sigma_\Lambda)$

A digraph $\mathcal{G}(\Sigma_\Lambda)$, associated to the state equations (1) of the system Σ_Λ, is the combination of a vertex set $\mathcal{V}$ and an edge set $\mathcal{E}$. The vertices represent the state and the control input components of Σ_Λ whereas the edges model the static or 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 set of control input vertices. In the case of SISO systems under consideration here, $\mathbf{U}$ is a singleton, that is $\mathbf{U} = \{\mathbf{u}\}$. The edge set is $\mathcal{E} = \{A - edges\} \cup \{B - edges\}$, with $\{A - edges\} = \{(\mathbf{x_j}, \mathbf{x_i}) \,|A(i,j) \neq 0\}$ and $\{B - edges\} = \{(\mathbf{u}, \mathbf{x_i}) \,|B(i) \neq 0\}$ where $A(i,j)$ and $B(i,j)$ denote the $(i,j)th$ element of the matrix A and B respectively. In the sequel, we will denote by $\mathbf{v}$ or by $\mathbf{v_j}$, $(j = 0, \ldots, n)$ a vertex of the digraph $\mathcal{G}(\Sigma_\Lambda)$ that involves $n+1$ vertices, regardless whether it is the input or a state vertex.

Path and related definitions

- A finite sequence of successive edges $(\mathbf{v^{r_j}}, \mathbf{v^{r_{j+1}}}) \in \mathcal{E}$ is called a path. A path of length $m+1$ is denoted by $P = \mathbf{v^{r_0}} \to \mathbf{v^{r_1}} \to \ldots \to \mathbf{v^{r_m}}$. We say that P covers or involves the vertices $\mathbf{v^{r_0}}, \mathbf{v^{r_1}}, \ldots, \mathbf{v^{r_m}}$;
- In a digraph $\mathcal{G}(\Sigma_\Lambda)$, if there exists 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 $\mathbf{v^j}$;
- $Succ(\mathcal{V})$ denotes the subset of all the successors of the vertices belonging to $\mathcal{V}_1$ that is $Succ(\mathcal{V}_1) \stackrel{def}{=} \{\mathbf{v} \in \mathcal{V}$, there exists a path from $\mathcal{V}_1$ to $\mathbf{v}\}$, $Pred(\mathcal{V})$ denotes the subset of all the predecessors of the vertices belonging to $\mathcal{V}_1$ that is $Pred(\mathcal{V}_1) \stackrel{def}{=} \{\mathbf{v} \in \mathcal{V}$, there exists a path from $\mathbf{v}$ to $\mathcal{V}_1\}$.
- A path is simple when every vertex occurs only once in this path;
- Paths are disjoint if they have no common vertex;
- The length of a 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 which are common to all the paths linking $\mathbf{v^i}$ to $\mathbf{v^j}$.
- A cycle is a simple path linking a vertex $\mathbf{v_i}$ to $\mathbf{v_i}$ with length $\ell(\mathbf{v_i}, \mathbf{v_i}) > 0$
- It is said that a set of cycles are "aligned" if there exists a simple path covering at least one of the vertex of all the cycles.

4. MAIN RESULT

In Subsection 4.1, we recall in Proposition 1 the necessary and sufficient conditions which must be satisfied by each vertex of $\mathcal{G}(\Sigma_\Lambda)$ to be associated to a flat output. Then, in Subsection 4.2, we provide a constructive method to obtain all the flat outputs and acts as the main result.

4.1 Preliminaries : recall of necessary and sufficient conditions for a flat output

The following result is borrowed from Millérioux et al. (2015).

Proposition 1. Consider the structured linear discrete-time system Σ_Λ described by (1). The output denoted y_F associated to a specific vertex $\mathbf{v_F} \in \mathbf{X} \cup \mathbf{U}$ is generically a flat output for the structured discrete linear system Σ_Λ if and only if, in the associated digraph $\mathcal{G}(\Sigma_\Lambda)$, the following three conditions hold:

C0. $\mathbf{v_F}$ is a successor of $\mathbf{u}$;

C1. The length of all the $\{\mathbf{u}\}$-$\{\mathbf{v_F}\}$ simple paths is equal to $\ell(\mathbf{u}, \mathbf{v_F})$;

C2. All the cycles cover at least an element of $V_{ess}(\{\mathbf{u}\}, \{\mathbf{v_F}\})$.

Proof: The proof has been established in Millérioux et al. (2015) and recalled in Appendix A for a self-contained and clear understanding. △

Download English Version:

https://daneshyari.com/en/article/710293

Download Persian Version:

https://daneshyari.com/article/710293

Daneshyari.com