Minimally Constrained Stable Switched Systems and Application to Co-simulation

Cláudio Gomes[1], Raphaël M. Jungers[2], Benoît Legat[3], and Hans Vangheluwe[4]

***Abstract*—We propose an algorithm to restrict the switching signals of a constrained switched system in order to guarantee its stability, while at the same time attempting to keep the largest possible set of allowed switching signals.**

Our work is motivated by co-simulation of complex dynamical systems by multiple cores. There, numerical stability is a hard constraint, but should be attained by restricting as few as possible the allowed behaviours of the simulators.

We apply our results to certify the stability of an adaptive co-simulation orchestration algorithm, which selects the optimal switching signal at run-time, as a function of (varying) performance and accuracy requirements.

I. Introduction

A switched system is defined as

$$x_{k+1} = A_{\sigma_k} x_k : \quad \sigma_k \in \{1, \dots, m\}, \ A_{\sigma_k} \in \mathcal{A} \tag{1}$$

where $\{1, \dots, m\}$ is the set of modes, σ_k is the mode active at time k, and $\mathcal{A} = \{A_1, \dots, A_m\} \subseteq \mathbb{R}^{n \times n}$ is a set of real matrices. We denote the sequence $\sigma_0, \sigma_1, \dots, \sigma_k$ as the *switching signal*.

Switched systems are used to model many systems [1]. In this paper, we are motivated by a new application [2] in the field of co-simulation. It is a numerical technique to couple multiple simulators, each simulating a part of a coupled system, in order to compute the overall behavior more efficiently [3].

As illustrated in Figure 1, in order to run a co-simulation, each simulator approximates the solution of a differential equation, exchanging values with other simulators at agreed-upon communication times. Between communication time instants, the unknown inputs are approximated with extrapolation functions.

[1]C. G. is a FWO Research Fellow, at the University of Antwerp, supported by the Research Foundation - Flanders (File Number 1S06316N). claudio.gomes@uantwerp.be

[2]R. M. J. is a F.R.S.-FNRS Research Associate, supported by the French Community of Belgium, the Walloon Region and the Innoviris Foundation.

[3]B. L. is a F.R.S.-FNRS Research Fellow, at Université Catholique de Louvain.

[4]H. V. is a Professor at the University of Antwerp. His work is partially supported by Flanders Make vzw, the strategic research centre for the manufacturing industry.

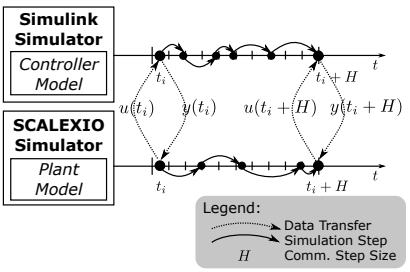

Fig. 1: Co-simulation.

To improve performance, during a co-simulation each simulator can adapt its policy: (i) the discretization step size and/or the numerical algorithm as a result of error estimates; (ii) the input approximation function as a result of the dynamics of the inputs; and (iii) the order and the values exchanged, as a result of varying the structure of the coupled system being simulated. We will call *policy sequence* to the sequence of policies taken by all simulators over time.

For other applications of adaptive methods, see

In the context of (co-) simulation, it is fundamental to ensure that a (co-) simulation method preserves the stability of the system being (co-) simulated. The error dynamics of a (co-) simulation can be modelled as a switched system (Section IV), hence this question becomes one of deciding stability of System (1). Furthermore, as the choice of future policies is influenced by the past policies, we consider *constrained switched systems*.

If there exist one or more policy sequences that can make the (co-) simulation method unstable, then the simulators have to be forbidden from following these. In the context of constrained switched systems, there are many ways of forbidding a policy sequence, and each way also forbids sequences that do not make the (co-) simulation unstable.

We tackle the problem of how to best forbid policy sequences that make the constrained switched system unstable. We propose that the best solution is to maximize the *entropy* of the stabilized constrained switched system in order to maximize the adaptability of the resulting (co-) simulation method.

In the next section, we formulate the problem. Then, in Section III we propose an algorithm that approximates the solution, and we prove that the it terminates and that the resulting system is stable. Furthermore, we provide a lifting technique that

yields better solutions. Section IV describes applications. Section V presents related work and Section VI concludes. An extended version of this work is presented in [4].

II. Problem Formulation

In this section, we formulate the research problem, and introduce the necessary concepts.

In practice, some switching signals of System (1) may not be relevant, and a way to represent the sensible ones is required.

Definition 1: Given a bounded set of matrices $\mathcal{A} = \{A_1, \dots, A_m\}$, we define an *automaton* as a directed and labelled graph $\mathbf{G} = (V, E)$, with nodes V and edges E such that no node has zero ingoing or outgoing degree. Each edge $(v, w, \sigma) \in E$ represents a transition from node $v \in V$ to node $w \in V$, where $\sigma \in \{1, \dots, m\}$ is the *label*, corresponding to A_σ.

We say that the switching signal, or word, $s = \sigma_0\sigma_1 \dots \sigma_{k-1}$ is *accepted* by an automaton $\mathbf{G}$ if it corresponds to a path in $\mathbf{G}$, that is, if there exists $v_0, v_1, \dots, v_k \in V$, such that $(v_j, v_{j+1}, \sigma_j) \in E$ for all $j = 0, \dots, k-1$. An accepted word induces an accepted matrix product $A_s = A_{\sigma_{k-1}} \cdots A_{\sigma_1} A_{\sigma_0} \in \mathcal{A}^k$.

We denote the set of accepted words of length k as $\mathbf{G}_k$, and the set of all words accepted by the automaton as $\mathbf{G}^* = \bigcup_{k=1}^{\infty} \mathbf{G}_k$. Moreover, $\mathbf{G}_k^\circ$ denotes the set of accepted cycles of length k.

Definition 2 (CSS): Given a set of matrices $\mathcal{A} = \{A_1, \dots, A_m\}$, and an automaton $\mathbf{G} = (V, E)$, we define a *constrained switched system* (CSS) $S = \langle \mathcal{A}, \mathbf{G} \rangle$ as a system where the variable x_k satisfies:

$$x_{k+1} = A_{\sigma_k} x_k : \quad \sigma_0 \dots \sigma_{k-1} \in \mathbf{G}_k. \tag{2}$$

We say that System (2) is *stable* iff

$$\lim_{k\to\infty} \|x_k\| = \lim_{k\to\infty} \left\| A_{\sigma_{k-1}} \cdots A_{\sigma_0} x_0 \right\| = 0,$$

for any word $\sigma_0 \dots \sigma_{k-1} \in \mathbf{G}_k$ and any $x_0 \in \mathbb{R}^n$.

To determine the stability of a CSS, we introduce the *constrained joint spectral radius*.

Definition 3 ([5, Definition 1.2]): The *constrained joint spectral radius* is defined as

$$\hat{\rho}(S) = \lim_{k\to\infty} \hat{\rho}_k(S) \text{ where } \hat{\rho}_k(S) = \sup_{w \in \mathbf{G}_k} \|A_w\|^{\frac{1}{k}},$$

and $\|\cdot\|$ is any matrix norm that satisfies the submultiplicative property.

Proposition 1 ([5, Lemma 3.1]): If $\hat{\rho}(S) < 1$, then the CSS is stable.

A way to prove that a CSS admits an unstable switching signal is to find a $c \in \mathbf{G}_k^\circ$ with $\rho(A_c) \geq 1$.

Proposition 2 ([6, Theorem 2.3]): If $\hat{\rho}(S) > 1$, then there exists a cycle $c \in \mathbf{G}_k^\circ$ of length k that satisfies $\rho(A_c) \geq 1$.

The problem of finding such cycles is outside the scope of our work (see [7] for the algorithm we used). As such, we introduce the following definition, which represents any algorithm available for this purpose.

Definition 4 (Oracle): Given $\epsilon > 0$, we define a stability oracle $\mathcal{O}_\epsilon : S \to \{\texttt{Stable}\} \cup \bigcup_{k=1}^{\infty} \mathbf{G}_k^\circ$, where S is a CSS. The oracle $\mathcal{O}_\epsilon$ returns either `Stable` certifying that $\hat{\rho}(S) < 1$ or a cycle $c \in \mathbf{G}_k^\circ$ such that $\rho(A_c)^{1/k} > 1 - \epsilon$.

We emphasize that the oracle has a (slightly) imperfect behaviour: in case $1 - \epsilon < \hat{\rho}(S) < 1$, one cannot guarantee what the outcome of the oracle will be. This imperfection is intentional, as it models the state of the art [8]. Proposition 2 ensures that if $\hat{\rho}(S) > 1 - \epsilon$, there exists a k and a cycle $c \in \mathbf{G}_k^\circ$ such that $\rho(A_c)^{1/k} > 1 - \epsilon$.

We now proceed to define the set of possible different switching signals that are admissible.

Definition 5 (Admissible Regular Language): We say that $\mathcal{L} = \mathbf{G}^*$ is the language *recognized* by the automaton $\mathbf{G}$. A language is *regular* if it is recognized by a finite automaton. A language $\mathcal{L}$ recognized by an automaton $\mathbf{G}$ is *admissible* for $\mathcal{A}$ if the constrained switched system $S = \langle \mathcal{A}, \mathbf{G} \rangle$ satisfies $\hat{\rho}(S) < 1$.

Let $\mathcal{L}_0$ denote the language recognized by the automaton $\mathbf{G}_0$ of a given $S = \langle \mathcal{A}, \mathbf{G}_0 \rangle$. Informally, our goal is to find the "largest" regular language $\mathcal{L}^\star \subseteq \mathcal{L}_0$ that is admissible. A widely used notion to describe the size of a regular language is that of Entropy.

Definition 6 (Entropy [9, Definition 4.1.1]): Given a regular language $\mathcal{L}$ recognized by an automaton $\mathbf{G}$, we define the *entropy* as $h(\mathcal{L}) = \lim_{k\to\infty} \frac{1}{k} \log_2 |\mathbf{G}_k|$. We denote the entropy of the language $\mathbf{G}^*$ recognized by an automaton $\mathbf{G}$ as $h^*(\mathbf{G})$. This quantity can be computed as shown in [4].

If $\mathcal{L} \subseteq \mathcal{L}'$, then $\mathbf{G}_k \subseteq \mathbf{G}'_k$ for any k, and so $h(\mathcal{L}) \leq h(\mathcal{L}')$. Our problem can now be formulated.

Problem 1: Given a CSS $\langle \mathcal{A}, \mathbf{G}_0 \rangle$, find the language $\mathcal{L}^\star$ solution of the following optimization problem:

$$\begin{aligned} \mathcal{L}^\star = \sup_{\mathcal{L} \text{ regular}} \; & h(\mathcal{L}) \text{ s.t.} \\ & \mathcal{L} \subseteq \mathcal{L}_0, \\ & \mathcal{L} \text{ is admissible for } \mathcal{A}. \end{aligned} \tag{3}$$

where $\mathcal{L}_0$ is the language recognized by $\mathbf{G}_0$.

Remark 1: In Problem 1, we restrict our attention to regular languages. While there are examples that highlight the benefit of using non-regular languages (see Example 1), in practice, one needs an *efficient* way of generating accepted switching signals. Automata allow the decision procedure to be fast, with little memory. In addition, as hinted in Example 2, regular languages may be constructed to approximate an admissible language with entropy arbitrarily close

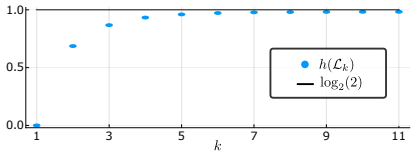

Fig. 2: Evolution of $h(\mathcal{L}_k)$ of Example 2 in terms of k.

to the entropy of the optimal solution, even if that optimal solution is a non-regular language.

Example 1: Consider $\mathcal{A} = \{A_1, A_2\}$, with $A_1 = 2$ and $A_2 = \frac{1}{2}$, and $\mathbf{G} = (V, E)$, where $V = \{v_1\}$ and $E = \{(v_1, v_1, 1), (v_1, v_1, 2)\}$. The optimal solution $\mathcal{L}^\star$ of (relaxed) Problem 1 should include every word that has more 1s than 2s. As shown in [10, Example 1.73], no automaton can be built that accepts this language.

Example 2: Consider $\mathcal{A} = \{A_1, A_2\}$, with $A_1 = 1$ and $A_2 = \frac{1}{2}$. A language is admissible if it does not contain the infinite repetition of the symbol 1. Let $\mathcal{L}_k$ be language of all words that do not contain k consecutive 1's. Figure 2 suggests that that $h(\mathcal{L}_k)$ tends to $\log_2(2)$ when k tends to infinity.

III. Lift-and-Constrain Stabilization

A. Constraining for more stability

Algorithm 1 details an iterative procedure that stabilizes a given CSS $S = \langle \mathcal{A}, \mathbf{G} \rangle$, using the oracle in Definition 4. At each iteration, if the oracle returns a cycle $c = \sigma_k \dots \sigma_k$, then c is eliminated from $\mathbf{G}$. The removal of a cycle can be accomplished by removing an edge of $\mathbf{G}$, thus potentially decreasing $\hat{\rho}(S)$. After removing the cycle c, any infinite sequence in $\mathbf{G}^*$ for which c is a subsequence will be eliminated too. The algorithm can produce an empty CSS, which does not imply that the original CSS is impossible to stabilize.

Example 3: Consider the automaton in Figure 3, and suppose the oracle has returned the cycle 234. Any of the edges in red can be removed to forbid the unstable sequence. If edge $v_1 \xrightarrow{2} v_2$ is removed, the infinite sequences accepted by the resulting automaton end with either an infinite sequence of 2's, or an infinite sequence of 3's. If edge $v_2 \xrightarrow{3} v_3$ is removed instead, the resulting automaton accepts infinite sequences comprised of repeating subsequences which include 2, or 3, or 12.

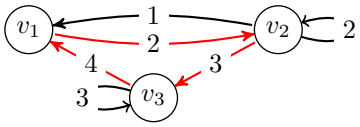

Fig. 3: Automaton of Example 3.

As Example 3 shows, the choice of different edges to be removed has a different impact in the entropy of the resulting automaton.

Data: A CSS $S = \langle \mathcal{A}, \mathbf{G} \rangle$.
Result: A stable CSS $S = \langle \mathcal{A}, \mathbf{G} \rangle$.
while $\mathcal{O}_\epsilon(S) \neq$ `Stable` **do**
1) Find $e \in \arg\max\{ h^*(\mathbf{G} - e) \mid e \in E,\ e$ is an edge of the cycle $\mathcal{O}_\epsilon(S) \}$;
2) Set $\mathbf{G} := \mathbf{G} - e$;
end

Algorithm 1: Stabilization algorithm for a constrained switched system. $h^*(\mathbf{G})$ denotes the entropy of the language recognized by $\mathbf{G}$. The difference $\mathbf{G} - e$ denotes the automaton obtained by removing the edge e from $\mathbf{G}$.

Theorem 1: Given a CSS $S = \langle \mathcal{A}, \mathbf{G} \rangle$ and an oracle satisfying Definition 4, Algorithm 1 terminates in finite time and the resulting CSS is stable.

B. Lifting for less conservativeness

The CSS returned by Algorithm 1 is not necessarily the optimal solution to Problem 1. To maximize the entropy of the stabilized CSS's, we propose to take an M*-Path-Dependent lift* of the automaton representing the input language $\mathcal{L}_0$.

Definition 7 ([11, Definition 3]): Given an automaton $\mathbf{G}$, we define the *lifted* automaton $\mathbf{G}^{[k]}$ of degree k as follows. For each path $v_0, \sigma_0, v_1, \sigma_1, \dots, \sigma_k, v_{k+1}$ of $\mathbf{G}$, $\mathbf{G}^{[k]}$ has a node $u^- = v_0\sigma_0 v_1 \sigma_1 \dots \sigma_{k-1} v_k$, a node $u^+ = v_1\sigma_1 v_2 \sigma_2 \dots \sigma_k v_{k+1}$ and an edge (u^-, u^+, σ_k).

The lifted automaton represents the same language, as shown by Proposition 3, but, as suggested by Theorem 2 and illustrated by Example 2, lifting the automaton before applying Algorithm 1 allows to obtain an admissible language with higher entropy.

Proposition 3: Let $\mathcal{L}$ be the language recognized by $\mathbf{G}$ and $\mathcal{L}^{[k]}$ the language recognized by $\mathbf{G}^{[k]}$, where $\mathbf{G}^{[k]}$ is the lift of degree k of $\mathbf{G}$. Then $\mathcal{L} = \mathcal{L}^{[k]}$.

Theorem 2: Consider Algorithm 1 with input $\mathcal{A}, \mathbf{G}_0^{[k]}$ (resp. $\mathcal{A}, \mathbf{G}_0^{[k+1]}$) where $\mathbf{G}_0^{[k]}$ (resp. $\mathbf{G}_0^{[k+1]}$) is the lift of degree k (resp. $k+1$) of a given automaton $\mathbf{G}$. If $\mathcal{O}_\epsilon(\mathcal{A}, \mathbf{G}_0^{[k]})$ and $\mathcal{O}_\epsilon(\mathcal{A}, \mathbf{G}_0^{[k+1]})$ are cycles corresponding to the same word, then $h^*(\mathbf{G}_1^{[k]}) \leq h^*(\mathbf{G}_1^{[k+1]})$.

In Section IV we show results corroborating Theorem 2.

IV. Application

We first sketch the application of our algorithm to simulation, and then detail the treatment to co-simulation in Section IV-B.

A. Simulation

Consider the problem of approximating the solution $x(t)$ of the system,

$$\dot{x}(t) = \bar{A}x(t), \text{ with } x(0) = x_0, \tag{4}$$

using an adaptive simulation algorithm.

Example 4: The approximation $\tilde{x}(t)$ of the solution to System (4), computed by a simulation algorithm that uses different step sizes, and different numerical methods, can be modelled as an unconstrained switched system (System (1)), with

$$\begin{aligned}\mathcal{A} &= \left\{ \tilde{A}_{fe,h}, \tilde{A}_{md,h}, \tilde{A}_{rg,h} \middle| h \in \{0.001, 0.002\} \right\} \\ \tilde{A}_{fe,h} &\triangleq \mathbf{I} + \bar{A}h \\ \tilde{A}_{md,h} &\triangleq \mathbf{I} + \bar{A}h + (\bar{A}h)^2/2 \\ \tilde{A}_{rg,h} &\triangleq \mathbf{I} + \bar{A}h + (\bar{A}h)^2/12 + (\bar{A}h)^3/6 + (\bar{A}h)^4/24.\end{aligned}$$

In the above set $\mathcal{A}$, the matrices correspond respectively to the Forward Euler method, the Midpoint method and the Runge-Kutta method.

In Example 4, if one assumes that System (4) is stable, that is, $\lim_{t\to\infty} \|x(t)\| = 0$ for any $x(0)$, then we need to ensure that the error made by discrete approximation $\tilde{x}(t)$ is dissipated. For this purpose, we introduce the *error switched system*, whose state variable is $e_t = \tilde{x}(t) - x(t)$, and which can be written as

$$\begin{aligned} e_{t+h} &= A_{\sigma(t)} e_t + L_{\sigma(t)}, \text{ with} \\ L_{\sigma(t)} &\triangleq \big(A_{\sigma(t)} - \exp(\bar{A}h)\big) x(t), \end{aligned} \tag{5}$$

where $L_{\sigma(t)}$ is the local error corresponding to $A_{\sigma(t)} \in \mathcal{A}$, and $\mathcal{A}$ is defined in Example 4. Neglecting $L_{\sigma(t)}$ yields a switched system, the stability of which determines the dissipation of error.

As an example, the error of the adaptive simulation algorithm introduced in Example 4 may not be stable for a switching signal $111\ldots$, but may be stable for $2121\ldots$ As a result, the adaptive simulation method may opt for a policy sequence that requires fewer model evaluations (compared to a non-adaptative algorithm), whilst preserving the stability.

B. Co-simulation

We apply our algorithm to certify an adaptive orchestration algorithm for the co-simulation of a controlled inverted pendulum. We will consider two simulators.

In the context of co-simulation, time is discretized into a countable set $T = \{t_0, t_1, t_2, \ldots\} \subset \mathbb{R}$, where $t_{i+1} = t_i + H_i$ is the time at step i and H_i is the communication step size at step i, with $i = 0, 1, \ldots$. From time $t_i \to t_{i+1}$, the simulator S_j, with $j = 1, 2$, is a mapping,

$$\begin{aligned} \tilde{x}_j(t_{i+1}) &= F_j(t_i, \tilde{x}_j(t_i), u_j(t_i)) \\ y_j(t_i) &= G_j(t_i, \tilde{x}_j(t_i), u_j(t_i)) \end{aligned} \tag{6}$$

with state vector $\tilde{x}_j$, input vector u_j and output vector y_j.

Simulators exchange outputs only at times $t \in T$. We assume without loss of generality that the two simulators are coupled in a feedback loop, that is,

$$u_1 = y_2 \text{ and } u_2 = y_1. \tag{7}$$

In the interval $t \in [t_i, t_{i+1}]$, each simulator S_j approximates the solution to a linear ODE,

$$\begin{aligned} \dot{x}_j &= A_j x_j + B_j u_j \\ y_j &= C_j x_j + D_j u_j \end{aligned} \tag{8}$$

where A_j, B_j, C_j, D_j are matrices, and the initial state $x_j(t_i)$ is computed in the most recent co-simulation step. To avoid algebraic loops and keep the exposition short, we assume that either D_1 or D_2 is the zero matrix. Let $D_2 = \mathbf{0}$.

Since the simulators only exchange outputs at times $t_i, t_{i+1} \in T$, the input u_j has to be extrapolated in the interval $[t_i, t_{i+1})$. In the simplest co-simulation strategy[1], this extrapolation is often implemented as a zero-order hold: $\tilde{u}_j(t) = u_j(t_i)$, for $t \in [t_i, t_{i+1})$. Then, Equation (8) can be re-written to represent the unforced system being integrated by each simulator:

$$\begin{bmatrix} \dot{x}_j \\ \dot{\tilde{u}}_j \end{bmatrix} = \begin{bmatrix} A_j & B_j \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} x_j \\ \tilde{u}_j \end{bmatrix} \tag{9}$$

We can represent the multiple internal integration steps of System (9), performed by the simulator S_j in the interval $t \in [t_i, t_{i+1}]$, as

$$\begin{bmatrix} \tilde{x}_j(t_{i+1}) \\ \tilde{u}_j(t_{i+1}) \end{bmatrix} = \tilde{A}_j^{k_j} \begin{bmatrix} \tilde{x}_j(t_i) \\ \tilde{u}_j \end{bmatrix} \tag{10}$$

where, e.g., $\tilde{A}_j = \mathbf{I} + h_j \begin{bmatrix} A_j & B_j \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$ for the Forward Euler method, $k_j = (t_{i+1} - t_i)/h_j$ is the number of internal steps, and $0 < h_j \leq H_i$ is the internal fixed step size that divides H_i.

At the beginning of the co-simulation step i, $u_1(t_i) = y_2(t_i)$ and $u_2(t_i) = y_1(t_i)$, as in Equation (7). This, together with Equation (8), gives,

$$\begin{aligned} u_1(t_i) &= C_2 \tilde{x}_2(t_i) \\ u_2(t_i) &= C_1 \tilde{x}_1(t_i) + D_1 C_2 \tilde{x}_2(t_i). \end{aligned} \tag{11}$$

Equations (9), (10), and (11) can be used to represent each co-simulation step by a linear mapping

$$\begin{bmatrix} \tilde{x}_1(t_{i+1}) \\ \tilde{x}_2(t_{i+1}) \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \tilde{A}_1^{k_1} & \mathbf{0} \\ \mathbf{0} & \tilde{A}_2^{k_2} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & C_2 \\ \mathbf{0} & \mathbf{I} \\ C_1 & D_1 C_2 \end{bmatrix}}_{\tilde{A}} \begin{bmatrix} \tilde{x}_1(t_i) \\ \tilde{x}_2(t_i) \end{bmatrix}$$

whose state transition matrix $\tilde{A}$ depends on the following items [13]: coupling approach, input approximations, numerical methods, step size h_j, and

[1]The derivation presented can be applied to more sophisticated input extrapolation techniques, see [12, Equation (9)].

communication step size H_i. Each configuration of these items affects the stability differently. Moreover, as hinted in [2], an adaptive algorithm that changes the configuration at runtime, based on varying tolerance and performance requirements, is beneficial *as long as it does not make the co-simulation unstable* (recall Example 4). Therefore, with the present work we generate a stabilized CSS, that encodes the set of all possible sequences of configurations that make the co-simulation stable, which can then be consulted during the co-simulation, with little computational cost [2].

In order to represent an adaptive co-simulation, let $\mathcal{A}$ be a set of co-simulation state transition matrices, each representing a particular configuration of the above items. An adaptive orchestration algorithm will, at the beginning of each co-simulation step i, inspect the state variables, and/or local error estimators [14], and decide which configuration should be used to proceed to step $i+1$.

Consider the co-simulation of an inverted pendulum that is kept at the equilibrium point using a state feedback controller. Simulator S_1 represents the controller, and simulator S_2 represents the pendulum.

Around the equilibrium point, the pendulum can be approximated as a system of the form of Equation (8), with

$$A_2 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -(I+ml^2)(b/p) & (m^2gl^2)/p & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -(mlb)/p & mgl(M+m)/p & 0 \end{bmatrix}$$

$$B_2 = \begin{bmatrix} 0 & (I+ml^2)/p & 0 & ml/p \end{bmatrix}^\top$$

$$C_2 = \mathbf{I} \qquad D_2 = \mathbf{0}$$

and parameters $M = 0.5, m = 0.2, b = 0.1, I = 0.006, g = 9.8, l = 0.3$.

The controller is a linear quadratic regulator, which, put in the form of Equation (8), is

$$A_1 = \mathbf{0} \qquad B_1 = \mathbf{0}$$

$$C_1 = \mathbf{0} \qquad D_1 = \begin{bmatrix} 1.0000 & 1.6567 & -18.6854 & -3.4594 \end{bmatrix}.$$

Assume we can use the Forward Euler and Midpoint methods, with internal fixed step sizes in the set $\{0.01, 0.02, 0.1, 0.2\}$. Furthermore, $H = 0.1$ or $H = 0.2$. Then, applying Equations (9), (10), and (11), we get a switched system over 8 matrices.

The matrices A_2 (corresponding to $H = 0.2$, $h_1 = 0.2$, Forward Euler), A_3 (corresponding to $H = 0.2$, $h_1 = 0.02$, Midpoint) and A_4 (corresponding to $H = 0.2$, $h_1 = 0.02$, Forward Euler) have a spectral radius larger than one. This means that the switching signals $222...$, $333...$ and $444...$ should be forbidden.

Applying Algorithm 1 directly to the unconstrained switched system , leads to removal of the edges with labels $2, 3$ and 4. This completely disallows the use of the matrices A_2, A_3 and A_4. The resulting language turns out to be admissible, its entropy is $\log_2(5)$.

Applying Algorithm 1 to a lift of degree 1, we get a constrained switched system with the automaton shown in Figure 4, where the edges in red were removed by the algorithm. We can see that the matrices A_2, A_3 and A_4 are now allowed by the algorithm (only the cyclic application of each one of these matrices is still disallowed). This solution is less conservative than the one with degree 0. One allowed cycle is 32645, where the symbols 5 and 6 seem to play the role of stabilizing the cycle.

We applied Algorithm 1 to the lifts of degree 0, 1 and 2. At each application of the algorithm, a stable constrained switched system was produced, with an entropy that increased with the degree of the lift. These results, summarized in Table I, corroborate Theorem 2.

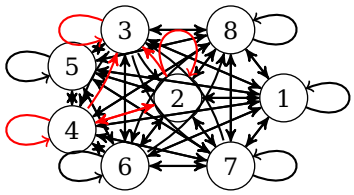

Fig. 4: Solution with entropy $\log_2(7.2568898)$.

TABLE I: Entropy achieved per lift degree.

k	Entropy [bit]	CPU time [s]
0	$\log_2(5)$	0.13
1	$\log_2(7.2568898)$	1.8
2	$\log_2(7.7083039)$	280

In the implementation, we first find all unstable cycles of length from 1 to 3 by iterating over all cycles of these lengths using brute force and selecting the ones that have a spectral radius above one. Note that several cycles can be disallowed at the same time using the same edge. Therefore we remove in priority the edges breaking the most cycles and use the entropy to break ties. We then use the method of [15] to determine whether the system is stable or whether it can provide an unstable cycle.

This application to co-simulation illustrates an important advantage of the method presented in [15]: it is capable of finding large unstable cycles. For the lift of degree 2 for example, it found the cycle $224533542245335422453354224523254$ of length 33.

V. Related Work

There has been a huge effort to understand how to ensure that a (discrete time) switched system is stable [16], [6], [8], [17]. To the best of our knowledge, the problem we introduce herehas never been studied. In the broader field of stabilization of switched systems, we can highlight the works in [18], [19], [20], [21],

[22], [23]. The key difference with our work is the goal: we are not satisfied with a single stable switching signal; we want to provide the maximum flexibility to the stabilized CSS, which can make use of this flexibility to choose the most appropriate switching signal. In the context of co-simulation, the reader can find stability analysis of traditional orchestration algorithms in [12], [24] and references thereof.

VI. Conclusion

We introduce a new problem in the context of constrained switched systems: 1) to restrict the switching possibilities of the system, so as to ensure its stability, and 2) to leave as many switching policies as possible (provided that the system becomes stable).

The motivation for leaving as many switching policies as possible lies in the fact that, in adaptive co-simulation, the orchestration algorithm will make the best possible choice as a function of information obtained during the simulation.

The problem is interesting in that it transforms a control problem into the problem of building an optimal language, that is, optimizing the construction of an automaton. By combining classical control concepts for switched systems , with classical automata-theoretic concepts , one can design algorithms to solve this problem. Our algorithm takes the form of a hierarchy of sufficient conditions, where increasingly better solutions are found by lifting the automaton (see Figure 2 and Table I). Essentially, this allows one to control the optimality of the solution, at the cost of processing power and memory.

This work is aimed to be a proof of concept, and we leave many research questions open. We want to investigate the conservativeness of restricting ourselves to regular languages, and to modify Algorithm 1 so that stronger theoretical results can be proven.

References

[1] R. M. Jungers, V. Protasov, and V. D. Blondel, "Efficient algorithms for deciding the type of growth of products of integer matrices," *Linear Algebra and its Applications*, vol. 428, no. 10, pp. 2296–2311, 2008.

[2] C. Gomes, B. Legat, R. M. Jungers, and H. Vangheluwe, "Stable Adaptive Co-simulation : A Switched Systems Approach," in *IUTAM Symposium on Co-Simulation and Solver Coupling*, Darmstadt, Germany, 2017, p. to appear.

[3] C. Gomes, C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe, "Co-simulation: A Survey," *ACM Computing Surveys*, vol. 51, no. 3, p. Article 49, Apr. 2018.

[4] C. Gomes, R. Jungers, B. Legat, and H. Vangheluwe, "Minimally Constrained Stable Switched Systems and Application to Co-simulation," Tech. Rep. arXiv:1809.02648, 2018.

[5] X. Dai, "A Gel'fand-type spectral radius formula and stability of linear constrained switching systems," *Linear Algebra and its Applications*, vol. 436, no. 5, pp. 1099–1113, Mar. 2012.

[6] R. Jungers, *The Joint Spectral Radius: Theory and Applications*. Springer Science & Business Media, 2009, vol. 385.

[7] B. Legat, R. M. Jungers, and P. A. Parrilo, "Generating Unstable Trajectories for Switched Systems via Dual Sum-Of-Squares Techniques," in *19th International Conference on Hybrid Systems: Computation and Control*. New York, New York, USA: ACM Press, 2016, pp. 51–60.

[8] P. A. Parrilo and A. Jadbabaie, "Approximation of the joint spectral radius of a set of matrices using sum of squares," in *HSCC*. Springer, 2007, pp. 444–458.

[9] D. Lind and B. Marcus, *An Introduction to Symbolic Dynamics and Coding*. Cambridge university press, 1995.

[10] M. Sipser, *Introduction to the Theory of Computation*, 2013.

[11] M. Philippe, R. Essick, G. E. Dullerud, and R. M. Jungers, "Stability of discrete-time switching systems with constrained switching sequences," *Automatica*, vol. 72, pp. 242–250, Oct. 2016.

[12] M. Busch, "Continuous approximation techniques for co-simulation methods: Analysis of numerical stability and local error," *Journal of Applied Mathematics and Mechanics*, vol. 96, no. 9, pp. 1061–1081, Sep. 2016.

[13] C. Gomes, C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe, "Co-simulation: State of the art," Tech. Rep., Feb. 2017.

[14] M. Arnold, C. Clauß, and T. Schierz, "Error Analysis and Error Estimates for Co-simulation in FMI for Model Exchange and Co-Simulation v2.0," in *Progress in Differential-Algebraic Equations*, S. Schöps, A. Bartel, M. Günther, W. E. J. ter Maten, and C. P. Müller, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 107–125.

[15] B. Legat, P. A. Parrilo, and R. M. Jungers, "Certifying unstability of Switched Systems using Sum of Squares Programming," Tech. Rep., Oct. 2017.

[16] H. Lin and P. J. Antsaklis, "Stability and Stabilizability of Switched Linear Systems: A Survey of Recent Results," *IEEE Transactions on Automatic Control*, vol. 54, no. 2, pp. 308–322, Feb. 2009.

[17] A. A. Ahmadi, R. Jungers, P. A. Parrilo, and M. Roozbehani, "Analysis of the joint spectral radius via lyapunov functions on path-complete graphs," in *Proceedings of the 14th international conference on Hybrid systems: computation and control*. ACM, 2011, pp. 13–22.

[18] X. Xu and P. Antsaklis, "Optimal Control of Switched Systems Based on Parameterization of the Switching Instants," *Automatic Control*, vol. 49, no. 1, pp. 2–16, Jan. 2004.

[19] L. Hetel, J. Daafouz, and C. Iung, "Stabilization of Arbitrary Switched Linear Systems With Unknown Time-Varying Delays," *IEEE Transactions on Automatic Control*, vol. 51, no. 10, pp. 1668–1674, Oct. 2006.

[20] W. Zhang, A. Abate, J. Hu, and M. P. Vitus, "Exponential stabilization of discrete-time switched linear systems," *Automatica*, vol. 45, no. 11, pp. 2526–2536, Nov. 2009.

[21] X. Zhao, L. Zhang, P. Shi, and M. Liu, "Stability and Stabilization of Switched Linear Systems With Mode-Dependent Average Dwell Time," *Automatic Control*, vol. 57, no. 7, pp. 1809–1815, Jul. 2012.

[22] A. Kundu and D. Chatterjee, "Stabilizing switching signals: A transition from point-wise to asymptotic conditions," *Systems & Control Letters*, vol. 106, pp. 16–23, 2017.

[23] P. Prabhakar and M. García Soto, "Formal synthesis of stabilizing controllers for switched systems," in *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*. ACM, 2017, pp. 111–120.

[24] C. Gomes, P. Karalis, E. M. Navarro-López, and H. Vangheluwe, "Approximated Stability Analysis of Bimodal Hybrid Co-simulation Scenarios," in *1st Workshop on Formal Co-Simulation of Cyber-Physical Systems*. Trento, Italy: Springer, Cham, 2017, pp. 345–360.