Unifying Control and Verification of Cyber-Physical Systems (UnCoVerCPS)

WP3 Online Verification for Control (Task 3.1)

D3.1 – Report on Reachability Analysis of Nonlinear Systems and Compositional Verification

WP3	**D3.1 – Report on Reachability Analysis of Nonlinear Systems and Compositional Verification**
Authors	Matthias Althoff - Technische Universität München Ahmed El-Guindy - Technische Universität München Bastian Schürmann - Technische Universität München Goran Frehse - Université Joseph Fourier Grenoble 1
Short Description	We present scalabe techniques for the reachability analysis of continuous nonlinear systems. Two over-approximative abstractions of the nonlinear dynamics are proposed: linear and polynomial differential inclusions. Scalability of the approach is further increased by compositional verification techniques. The methods are demonstrated by smart grid use cases.
Keywords	Reachability analysis, nonlinear continuous systems, abstraction, formal verification, compositional verification, smart grids.
Deliverable Type	Report
Dissemination level	Public
Delivery Date	31 Dec 2015
Contributions by	—
Internal review by	Maria Prandini - Politecnico di Milano Olaf Stursberg - Universität Kassel
External review by	—
Internally accepted by	Matthias Althoff
Date of acceptance	17 Dec 2015

Document history:

Version	**Date**	**Author/Reviewer**	**Description**
1.0	01 Oct 2015	Althoff et al.	Internal review version
1.1	17 Dec 2015	Althoff et al.	Final version

Contents

1 Introduction

Formal verification is a key aspect in UnCoVerCPS since this project aims at combining novel controller synthesis techniques with novel formal verification techniques for systems with a mixed discrete and continuous dynamics, which we refer to as hybrid dynamics in this deliverable. Before a system can be verified, one has to specify the properties to be checked as described in deliverable D 1.1. As demonstrated in D 1.1, most specifications can be formulated as a monitor automaton. In essence, the verification problem using monitor automata can be boiled down into a reach-avoid problem, where one should reach a goal region, while avoiding a set of unsafe states. For instance, a set of unsafe states in a chemical plant may contain all temperatures above the boiling point of a liquid. The verification task would be to guarantee that it is impossible to reach the unsafe set containing temperatures above the boiling point. The challenge of guaranteeing the avoidance of unsafe sets lies in the infinitely many possible trajectories that a system can evolve with when the initial state, the input, or the parameters may take values within a continuous set.

An obvious technique to test the correct behavior of a system, is by simulation. The big advantage of a simulation is that it might produce a counter-example, i.e. a trajectory that hits a set of unsafe states. In this case, one can show that a system is unsafe. However, one cannot prove that the system is safe if no counter-example is produced, since there exist infinitely many possible trajectories due to uncertain initial states, inputs, and parameters. Thus, testing exemplary trajectories is not sufficient since the trajectory that hits the unsafe set may have been missed; see Fig. 1 for two continuous state variables x_1 and x_2.

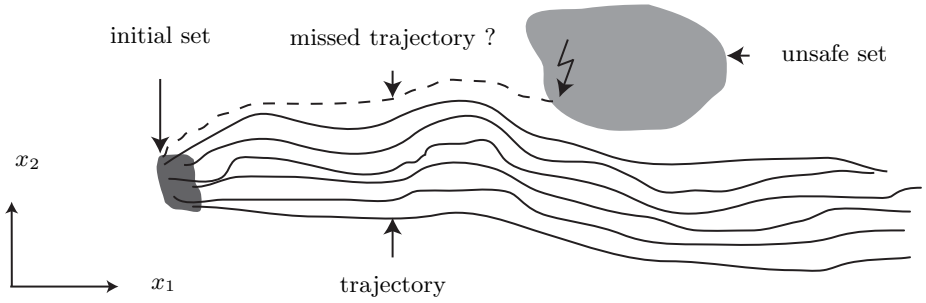


Figure 1: Searching for counter-examples by simulation.

In UnCoVerCPS, safety verification is conducted via reachability analysis. Loosely speaking, reachability analysis determines the set of states that a system can possibly visit. A more precise description of a reachable set is the union of all possible trajectories that a system can evolve within finite or infinite time, when starting from a bounded set of initial states, subject to a set of possible input and parameter values. An example of a reachable set is

presented in Fig. 2. If the reachable set does not intersect any set of unsafe states, one can guarantee the safety of the system.

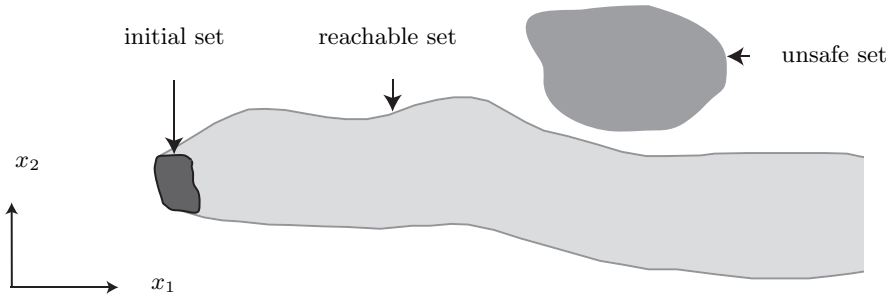


Figure 2: Verification using reachable sets.

However, one can only compute the exact reachable set for special cases [1, 2, 3]. A possibility to still prove the safety of a system is to over-approximate the set of reachable states, as shown in Fig. 3. Clearly, if the over-approximated set of reachable states does not intersect the set of unsafe states, the original system is safe, too. The downside is that if the over-approximation intersects the unsafe set, one cannot decide if the system is unsafe since the exact reachable set might not intersect the unsafe set. Thus, the goal is to minimize the over-approximation of reachable sets along with a moderate increase in computational costs. If this goal is accomplished, there is much hope that reachability analysis will become a tool that is frequently used by a huge variety of engineers – much the same as today's use of simulations.

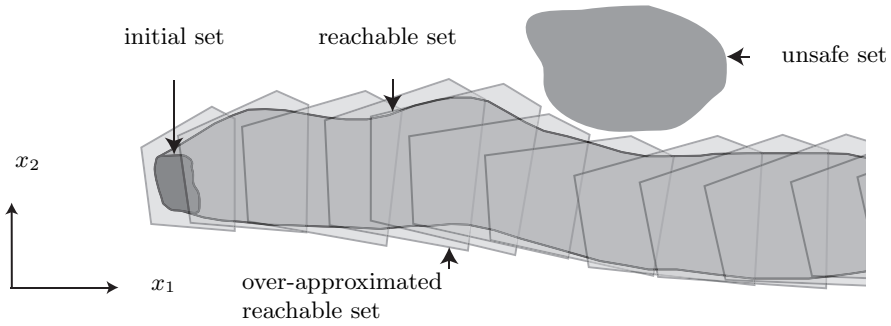


Figure 3: Verification using over-approximated reachable sets.

Our goal in UnCoVerCPS is to significantly reduce the computation time for formal verification to make it applicable during the operation of the system, which we refer to as online verification. The online capability of our verification approaches allows one to consider the current situation and thus tackle the problem of verifying systems in unknown and changing environments. It should be mentioned that the proposed advances in online verification will also benefit classical offline verification techniques. The proposed new techniques for reachability analysis are implemented in the tools *State Space Explorer* (*SpaceEx*) [4] and

Continuous Reachability Analyzer (*CORA*) [5], which are continuously improved during the course of the project. The applicability of the developed techniques are tested by applying them to our demonstrators as described in deliverable D 5.1.

The deliverable is organized as follows: Previous work is reviewed in Sec. 2. In Sec. 3, we present techniques for abstracting complicated nonlinear dynamics to simpler dynamics. Abstraction is understood as the process to simplify the mathematical description while not loosing any rigor by adding uncertainty to the abstract model. Thus, the abstract models are no longer deterministic, but are still analyzable to reachability analysis – or even become feasible for reachability analysis due to the abstraction. Two abstraction techniques are presented. The first one abstracts to linear systems (see Sec. 4) and the second one abstracts to polynomial systems (see Sec. 5). Finally, we present a technique for compositional verification in Sec. 6, i.e., a divide-and-conquer technique to verify larger systems by composing verification results of smaller subcomponents. Examples in the area of smart grids, illustrate the usefulness of the presented techniques. We have also implemented the presented techniques in the verification tool *CORA*. Later in the project, the techniques for nonlinear continuous dynamics will also be transferred to *SpaceEx*.

2 Previous Work

Formal verification has pioneered for purely discrete systems and was later extended to timed automata and continuous as well as hybrid systems involving discrete and continuous dynamics. By definition, cyber-physical systems have a hybrid dynamics. Similar to discrete systems, a variety of formal verification techniques have been developed: automatic theorem proving [6], constraint propagation [7], barrier certificates [8], and reachability analysis [9]. Contrary to discrete dynamics, the most common property to be checked is whether the continuous state variable enters a set of forbidden states.

The aforementioned techniques are not applicable to on-the-fly verification or have not yet been adapted such that they can be appropriately used during the operation of systems. Theorem proving typically requires human intervention so that it does not qualify for on-the-fly verification (e.g. 656 user interventions in [10, p.3577]). Further, the runtime of theorem proving for hybrid systems is unknown and thus does not qualify for real-time applications. Although constraint propagation and barrier certificates have upper bounds on the runtime, the computational complexity makes on-the-fly computation infeasible considering the current state-of-the-art. For both techniques, the maximum number of considered continuous state variables is around 5, see e.g. [7, 8].

Due to the potential applicability of reachability analysis to on-the-fly verification of cyber-physical systems, the remaining literature review focuses on this technique. For most systems, the continuous rather than the discrete dynamics is the most challenging aspect of the verification process. Reachability analysis can be performed via simulation techniques when approximate bisimulation properties can be shown [11], by Eulerian schemes, and by Lagrangian schemes. Eulerian schemes translate the reachability problem into an optimization problem of Hamilton-Jacobi equations, requiring to discretize the state space [12]. This causes exponential complexity in the number of continuous state variables limiting the applicability to systems with no more than 5 continuous state variables. Lagrangian schemes propagate the reachable set for consecutive points in time or time intervals. Especially for linear continuous dynamics, large state spaces with potentially more than 100 continuous state variables[1] can be efficiently computed. Typical set representations are: polytopes [13], zonotopes [14], ellipsoids [15], support functions [16], and oriented hyper-rectangles [17].

In this paragraph we present own previous work in the area of reachability analysis of nonlinear systems. We have developed novel algorithms that can handle ordinary differential equations and differential-algebraic equations with more than 100 continuous state variables since the algorithms have a polynomial complexity with respect to the number n of continuous state variables [18]. The work in [18] is an extension of the earlier work [19]. Both approaches are based on a conservative linearization, where the over-approximation holds for a region moving along the reachable set, and has been adopted by other researchers, see e.g. [20]. While the work in [18] and [19] uses linear abstractions, the work in [21] uses polynomial abstractions. In this deliverable, we present new and better abstraction techniques and a first analysis on the efficiency of compositional verification techniques with respect to different partitions of a complete system into components that are individually analyzed under consideration of interactions with other components.

3 Abstraction Techniques for Nonlinear Systems

Although a fairly large group of dynamic systems can be described by linear continuous systems, the extension to nonlinear continuous systems is an important step for the analysis of more complex systems. The analysis of nonlinear systems is much more complicated since many valuable properties are no longer valid. One of them is the superposition principle, which allows the homogeneous and the inhomogeneous solution to be obtained separately. Another aspect is that reachable sets of linear systems can be computed by a linear map.

[1] depending on the set representation and whether uncertain inputs are considered

This makes it possible to exploit that geometric representations such as ellipsoids, zonotopes, and polytopes are closed under linear transformations, i.e. they are again mapped to ellipsoids, zonotopes and polytopes, respectively. In CORA, reachability analysis of nonlinear systems is based on abstraction. We present abstraction to linear systems and to polynomial systems. Since the abstraction causes additional errors, the abstraction errors are determined in an over-approximative way and added as an additional uncertain input so that an over-approximative computation is ensured.

General nonlinear continuous systems with uncertain parameters and Lipschitz continuity are considered. The initial state $x(0)$ can take values from a set $\mathcal{X}_O \subset \mathbb{R}^n$ and the input u takes values from a set $\mathcal{U} \subset \mathbb{R}^m$. The evolution of the state x is defined by the following differential equation:

$$\dot{x}(t) = f(x(t), u(t)), \qquad x(0) \in \mathcal{X}_O \subset \mathbb{R}^n, \quad u(t) \in \mathcal{U} \subset \mathbb{R}^m,$$

where $u(t)$ and $f(x(t), u(t))$ are assumed to be globally Lipschitz continuous so that the Taylor expansion for the state and the input can always be computed, a condition required for the abstraction. The solution of (3) for $x(0) = x_0$, $t \in [0, t_f]$, and input trajectory $u(\cdot)$ is denoted by $\chi(t, x_0, u(\cdot))$. Note that $u(\cdot)$ refers to a trajectory, whereas $u(t)$ refers to the value of the trajectory at time t. The exact reachable set for an uncertain initial set $\mathcal{X}_O$ and a set of possible inputs $\mathcal{U}$ is

$$\mathcal{R}^e([0, t_f]) = \Big\{ \chi(t, x_0, u(\cdot)) \Big| t \in [0, t_f],\, x_0 \in \mathcal{X}_O,\, \forall t :\, u(t) \in \mathcal{U} \Big\}. \tag{1}$$

Since, as previously mentioned, the set of reachable states cannot be computed exactly, we compute over-approximations $\mathcal{R}([0, t_f]) \supseteq \mathcal{R}^e([0, t_f])$. The iterative computation of reachable sets for linear systems requires set-based addition (*Minkowski addition*) and set-based multiplication:

$$\mathcal{X} \oplus \mathcal{Y} := \{x + y | x \in \mathcal{X}, y \in \mathcal{Y}\},$$
$$\mathcal{X} \otimes \mathcal{Y} := \{x\, y | x \in \mathcal{X}, y \in \mathcal{Y}\}.$$

Note that in the remainder of this deliverable, the symbol for set-based multiplication is often omitted for simplicity of notation, and that one or both operands can be singletons. For addition and subtraction of a set with a singleton, we use the classical symbols $+$ and $-$. In order to avoid parentheses, it is agreed that operations of fixed values have precedence over corresponding set-based operations, e.g. $a + b \oplus \mathcal{C} = (a + b) \oplus \mathcal{C}$, and that set-based multiplication has precedence over Minkowski addition.

A schematic visualization of the overall concept for computing the reachable set is shown in Fig. 4. The reachable set in this deliverable is computed iteratively for time intervals

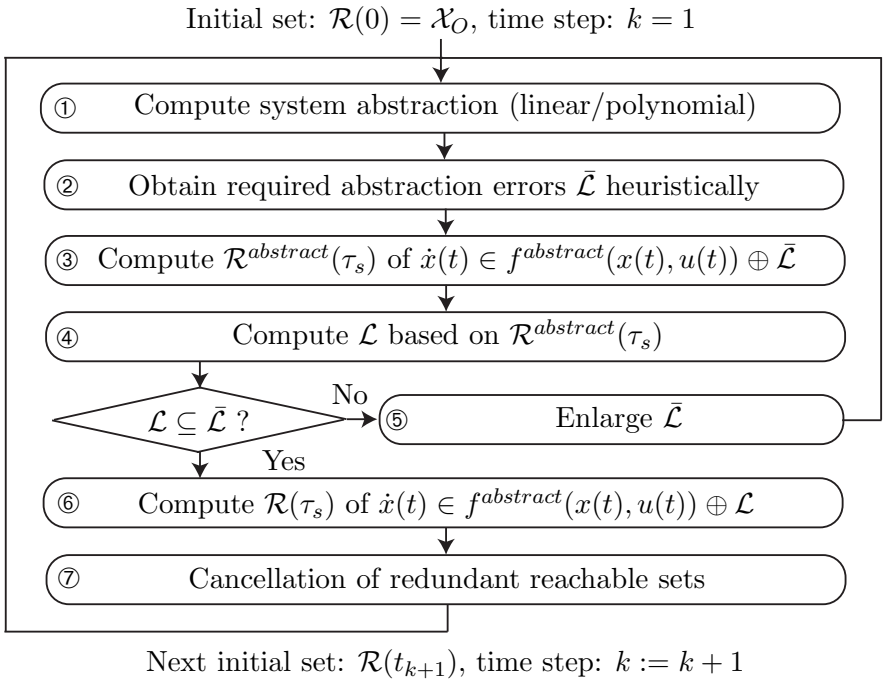


Figure 4: Computation of reachable sets – overview.

$t \in \tau_s = [k\,r, (k+1)r]$ where $k \in \mathbb{N}^+$ and $r \in \mathbb{R}^+$ is the time step. The procedure for computing the reachable sets of the consecutive time intervals is as follows:

① The nonlinear system $\dot{x}(t) = f(x(t), u(t))$ is either abstracted to a linear system or a polynomial system. After introducing $z = [x^T,\, u^T]^T \in \mathbb{R}^o$ the more general polynomial form can be formalized for the i^{th} dimension of the system as

$$\begin{aligned} \dot{x}_i = f^{abstract}(x,u) = & w_i + \frac{1}{1!}\sum_{j=1}^{o} C_{ij} z_j(t) + \frac{1}{2!}\sum_{j=1}^{o}\sum_{k=1}^{o} D_{ijk} z_j(t) z_k(t) \\ & + \frac{1}{3!}\sum_{j=1}^{o}\sum_{k=1}^{o}\sum_{l=1}^{o} E_{ijkl} z_j(t) z_k(t) z_l(t) + \dots \end{aligned} \tag{2}$$

We introduce the set of abstraction errors $\mathcal{L}$ to ensure that $f(x,u) \in f^{abstract}(x,u) \oplus \mathcal{L}$, which allows the reachable set to be computed in an over-approximative way.

② Since the set of abstraction errors is initially unknown for each time interval, we guess an over-approximation of the set of abstraction errors, denoted by $\bar{\mathcal{L}}$.

③ The reachable set $\mathcal{R}^{abstract}(\tau_s)$ of $\dot{x}(t) \in f^{abstract}(x(t), u(t)) \oplus \bar{\mathcal{L}}$ is computed.

④ The set of abstraction errors $\mathcal{L}$ is computed based on the reachable set $\mathcal{R}^{abstract}(\tau_s)$. Details on how to obtain the set of abstraction errors are later described.

⑤ When $\mathcal{L} \not\subseteq \bar{\mathcal{L}}$, the abstraction error is not admissible, since the guessed set of abstraction errors $\bar{\mathcal{L}}$ is not an over-approximation. This requires the assumption $\bar{\mathcal{L}}$ to be enlarged.

If several enlargements are not successful, one has to split the reachable set and continue with one more partial reachable set from then on.

⑥ If $\mathcal{L} \subseteq \bar{\mathcal{L}}$, the abstraction error is accepted and the reachable set is obtained by using the tighter abstraction error: $\dot{x}(t) \in f^{abstract}(x(t), u(t)) \oplus \mathcal{L}$.

⑦ It remains to increase the time step ($k := k + 1$) and cancel redundant reachable sets that are already covered by previously computed reachable sets. This decreases the number of reachable sets that have to be considered in the next time interval.

4 On-The-Fly Linearization

In this section, we present how to abstract the original nonlinear dynamics in (3) to linear dynamics on-the-fly. The abstracton is on-the-fly since we abstract the dynamics based on the current state of the system and not statically for a fixed linearization point. For a concise notation, we use the linearization point $z^* := [x^{*T}, u^{*T}]^T$, and $\mathcal{R}^z := \mathcal{R}(\tau_s) \times \mathcal{U}$. The linearization point is chosen differently for each iteration so that it is close to the center of the next reachable set $\mathcal{R}(\tau_s)$, which is a good heuristic for minimizing the linearization error. The Euler integration method is used for the time increment $0.5r$ to approximate this point by $x^* = c^d + 0.5r \cdot f(c^d, c^u)$, where c^d and c^u are the respective centers of the sets $\mathcal{R}(t_s)$ and $\mathcal{U}$. For the input linearization point we choose $u^* = c^u$, using the same argument that the center is a good heuristics.

The linearization of (3) is performed using a first-order Taylor expansion with Lagrangian remainder:

$$\begin{aligned} \dot{x}_i = f_i(z(t)) \in f_i(z^*) + \frac{\partial f_i(z)}{\partial z}\Big|_{z=z^*}(z(t) - z^*) \\ \oplus \underbrace{\left\{ \frac{1}{2}(z(t) - z^*)^T \frac{\partial^2 f_i(z)}{\partial z^2}\Big|_{z=\xi}(z(t) - z^*) \Big| \xi,\, z(t) \in \mathcal{R}^z \right\}}_{=:\mathcal{L}_i}, \end{aligned} \tag{3}$$

where $\mathcal{L}_i$ denotes the projection of $\mathcal{L}$ onto the i^{th} coordinate. The Lagrangian remainder $\mathcal{L}$ encloses all higher-order terms if ξ can take any value of the linear combination of z and z^*, i.e. $\xi \in \{\alpha z + (1 - \alpha)z^* | \alpha \in [0, 1]\}$, which follows from the mean value theorem [22, p. 87]. Since for the time interval τ_s, (i) $z(t)$ can take any values from $\mathcal{R}^z$, (ii) $\mathcal{R}^z$ will be represented by a convex set for the linear abstraction, and (iii) z^* is chosen as an interior point of this set, it follows that for $\xi \in \mathcal{R}^z$ the set of Lagrangian remainders is captured in (3). In order to obtain the standard notation of the linearized system, the z vector is separated into the

state vector x and the input vector u.

$$\begin{aligned}\dot{x} &\in f(z^*) + \frac{\partial f(z)}{\partial z}\Big|_{z=z^*}(z - z^*) \oplus \mathcal{L} \\ &= A\Delta x + B\Delta u + f(x^*, u^*) \oplus \mathcal{L}\end{aligned} \tag{4}$$

with

$$\begin{aligned}&\Delta x = x - x^*, \quad \Delta u = u - u^* \\ &A = \frac{\partial f(x,u)}{\partial x}\Big|_{x=x^*}, \quad B = \frac{\partial f(x,u)}{\partial u}\Big|_{u=u^*}\end{aligned}$$

We start with the easiest way of obtaining the set of Lagrangian remainders, followed by a more sophisticated quadratic and even more sophisticated cubic technique. In order to apply techniques for computing the Lagrangian remainder, we first have to introduce zonotopes with which we over-approximate reachable sets in this deliverable.

4.1 Zonotopes

We first recall the representation of a zonotope, followed by three interpretations. Throughout this deliverable, we index elements of vectors and matrices by subscripts and enumerate vectors or matrices by superscripts in parentheses to avoid confusion with the exponentiation of a variable. For instance $A_{ij}^{(k)}$ is the element of the i^{th} row and j^{th} column of the k^{th} matrix $A^{(k)}$.

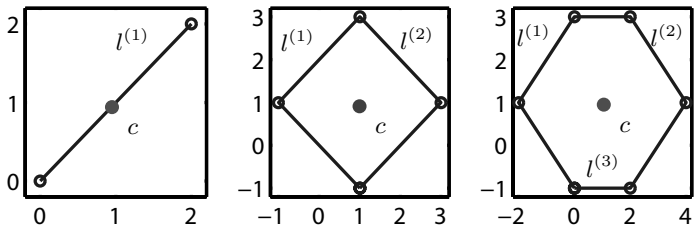
Definition 4.1 (Zonotope (G-Representation)) *Zonotopes are parameterized by a center $c \in \mathbb{R}^n$ and generators $g^{(i)} \in \mathbb{R}^n$ and defined for $c \in \mathbb{R}^n$, $g^{(i)} \in \mathbb{R}^n$ as*

$$\mathcal{Z} = \Big\{ c + \sum_{i=1}^{p} \beta_i \, g^{(i)} \Big| \beta_i \in [-1, 1] \Big\}. \tag{5}$$

The order of a zonotope is defined as $\varrho = \frac{p}{n}$.

We write in short $\mathcal{Z} = (c, g^{(1)}, \ldots, g^{(p)})$. Zonotopes are a compact way of representing sets in high-dimensional spaces. A zonotope can be interpreted as the Minkowski addition of line segments $l^{(i)} = [-1, 1]\, g^{(i)}$, and is visualized step-by-step in $\mathbb{R}^2$ in Fig. 5. Another interpretation of a zontope is the projection of a p-dimensional unit hypercube $\mathcal{C} = [-1, 1]^p$ onto the n-dimensional space by the matrix of generators $G = \Big[g^{(1)}, \ \ldots, \ g^{(p)} \Big]$, which is then translated to the center c: $\mathcal{Z} = c \oplus G \otimes \mathcal{C}$. We write in short $\mathcal{Z} = (c, G)$.

More important than an efficient set representation of zonotopes is that linear maps $M \otimes \mathcal{Z}$ ($M \in \mathbb{R}^{q \times n}$) and Minkowski addition $\mathcal{Z}_1 \oplus \mathcal{Z}_2$, as required for reachability analysis, can be computed efficiently and exactly [23]. Given $\mathcal{Z}_1 = (c, g^{(1)}, \ldots, g^{(p_1)})$ and $\mathcal{Z}_2 = (d, h^{(1)}, \ldots, h^{(p_2)})$



(a) $c \oplus l^{(1)}$ (b) $c \oplus l^{(1)} \oplus l^{(2)}$ (c) $c \oplus \ldots \oplus l^{(3)}$

Figure 5: Step-by-step construction of a zonotope.

one can efficiently compute

$$\begin{aligned} \mathcal{Z}_1 \oplus \mathcal{Z}_2 &= (c + d, g^{(1)}, \ldots, g^{(p_1)}, h^{(1)}, \ldots, h^{(p_2)}), \\ M \otimes \mathcal{Z}_1 &= (M\,c, M\,g^{(1)}, \ldots, M\,g^{(p_1)}). \end{aligned} \tag{6}$$

For the multiplication with an interval matrix $\mathcal{M}$, we split $\mathcal{M}$ into a real-valued matrix $M \in \mathcal{R}^{n\times n}$ and an interval matrix with radius $S \in \mathcal{R}^{n\times n}$, such that $\mathcal{M} = M \oplus [-S, S]$. After introducing S_j as the j^{th} row of S, the result is over-approximated as shown in [24, Theorem 3.3] by

$$\mathcal{M}\mathcal{Z}_1 \subseteq (M\mathcal{Z}_1 \oplus [-S, S]\mathcal{Z}_1) \subseteq (Mc_1, Mg^{(1)}, \ldots, Mg^{(p_1)}, h^{(1)}, \ldots, h^{(n)})$$

$$h_j^{(i)} = \begin{cases} S_j(|c| + \sum_{k=1}^{p_1} |g|^{(k)}), \text{ for } i = j \\ 0, \text{ for } i \neq j \end{cases}.$$

Using the alternative notation of a zonotope $\mathcal{Z} = (c, G)$, the Cartesian product of two zonotopes $\mathcal{Z}_1 = (c,\, G)$ and $\mathcal{Z}_2 = (d,\, H)$ is

$$\mathcal{Z}_1 \times \mathcal{Z}_2 = \left(\begin{bmatrix} c \\ d \end{bmatrix}, \begin{bmatrix} G & \mathbf{0} \\ \mathbf{0} & H \end{bmatrix} \right),$$

where $\mathbf{0}$ is a matrix of zeros of proper dimension. We will also need the enclosure of a zonotope by a multidimensional box [24, Prop. 2.2] and its absolute value:

$$\begin{aligned} \text{box}(\mathcal{Z}_1) :=& [c_1 - \Delta g, c_1 + \Delta g], \qquad \Delta g := \sum_{i=1}^{p_1} |g^{(i)}|, \\ |\mathcal{Z}_1| :=& |c_1| + \Delta g. \end{aligned} \tag{7}$$

The representation of reachable sets with zonotopes allows an efficient computation as presented later.

4.2 Lagrangian Remainder

In this section, we provide a simple but efficient technique to over-approximate the Lagrange remainder in (3). After defining $H^{(i)}(\xi) := \frac{\partial^2 f_i(\xi)}{\partial z^2}$, where i is the system dimension of f, one

can write the Lagrange remainder in (3) as

$$\mathcal{L} = \left\{ \frac{1}{2}(z - z^*)^T H^{(i)}(\xi)(z - z^*) \middle| \xi, z \in \mathcal{R}^z \right\}, \tag{8}$$

where $\xi \in \mathcal{R}^z$ since we assume that $z^* \in \mathcal{R}^z$. In order to determine the set $\mathcal{L}$ for a time interval τ_s, one has to consider the possible values of $z \in \mathcal{R}^z$ within this time interval. In order to determine the maximum absolute values of $\mathcal{L}$ for each dimension $\mathcal{L}_i$ in an efficient way, the following over-approximation is computed:

Proposition 4.1 *The absolute values of the Lagrange remainder for each dimension can be over-approximated for $z \in \mathcal{R}^z$ by*

$$|\mathcal{L}_i| \subseteq [0, \hat{L}_i]$$

$$\text{with } \hat{L}_i = \frac{1}{2}\gamma^T \max_{\xi \in \mathcal{R}^z}(|\mathcal{H}^{(i)}(\xi)|)\gamma, \ \gamma = |c - z^*| + \sum_{i=1}^{p} |g^{(i)}|$$

where c is the center and $g^{(i)}$ are the generators of the zonotope $\mathcal{R}^z$. The max*-operator and the absolute values are applied elementwise.*

Proof 1 *The following over-approximations apply for the absolute value of L_i:*

$$\begin{aligned} |\mathcal{L}_i| &= \left\{ \frac{1}{2}|(z - z^*)^T H^{(i)}(\xi)(z - z^*)| \, \middle| \xi, z \in \mathcal{R}^z \right\} \\ &\subseteq \frac{1}{2}[0, \max_{\xi, z \in \mathcal{R}^z} \left(|(z - z^*)^T H^{(i)}(\xi)(z - z^*)| \right)] \\ &\subseteq \frac{1}{2}[0, \max_{z \in \mathcal{R}^z}(|z - z^*|)^T \max_{\xi \in \mathcal{R}^z}(|H^{(i)}(\xi)|) \max_{z \in \mathcal{R}^z}(|z - z^*|)] \end{aligned}$$

The expression $\max_{z \in \mathcal{R}^z}(|z - z^*|)$ can be further rewritten since $z \in \mathcal{R}^z$ is within a zonotope with center c and generators $g^{(i)}$:

$$z \in \mathcal{R}^z, \beta_i \in [-1, 1] : \max_{z \in \mathcal{R}^z}(|z - z^*|) = \max_{\beta_i \in [-1,1]}(|c - z^* + \sum_{i=1}^{p} \beta_i g^{(i)}|) \leq |c - z^*| + \sum_{i=1}^{p} |g^{(i)}| = \gamma$$

such that the expression of proposition 4.1 is obtained.

The expression $\max(|\mathcal{H}^{(i)}(\xi)|)$ in proposition 4.1 is computed via interval arithmetics [25]. To do so, the values of z have to be over-approximated by an interval vector as shown in (7): $z \in \text{box}(\mathcal{R}^z)$. From this follows that $\xi \in \mathcal{R}^z$ also becomes an interval vector. The result of proposition 4.1 also allows us to find a linearization point z^* that minimizes the values $\hat{L}_i$ and thus the set of Lagrange remainders.

Proposition 4.2 *The bound of the Lagrange remainder $\hat{L}$ is minimized by choosing $z^* = c$ as the linearization point.*

Proof 2 *The value of γ is minimized by $z^* = c$ which can be directly checked from its computation in proposition 4.1. By choosing $z^* = c$, it follows that $\max_{z\in\mathcal{R}^z}(|z - z^*|)$ is minimized and thus $\hat{L}_i$ is minimized since $\max_{\xi\in\mathcal{R}^z}(|H^{(i)}(\xi)|)$ is not affected by z^*.*

4.3 Quadratic Evaluation Using Zonotopes

The previous computation of the set of Lagrange remainders in Proposition 4.1 does not consider correlations between the generators $g^{(i)}$ of $\mathcal{R}^z$. In order to consider this correlation for a tighter over-approximation of the linearization error we introduce the over-approximation of a quadratic map:

Lemma 4.1 (Quadratic Map) *Given a zonotope $\mathcal{Z} = (c, g^{(1)}, \dots, g^{(p)})$ and a discrete set of matrices $Q^{(i)} \in \mathbb{R}^{n\times n}$, $i = 1 \dots n$, the set*

$$\mathcal{Z}_Q = \{\varphi | \varphi_i = x^T Q^{(i)} x,\, x \in \mathcal{Z}\}$$

is over-approximated by a zonotope

$$\text{quad}(Q, \mathcal{Z}) := (d, h^{(1)}, \dots, h^{(\sigma)})$$

with $\sigma = \binom{p+2}{2} - 1$ generators, where the center is

$$d_i = c^T Q^{(i)} c + 0.5 \sum_{s=1}^{p} {g^{(s)}}^T Q^{(i)} g^{(s)},$$

and the generators are computed as

$$\begin{aligned}
j = 1 \dots p: \quad & h_i^{(j)} = c^T Q^{(i)} g^{(j)} + {g^{(j)}}^T Q^{(i)} c \\
j = 1 \dots p: \quad & h_i^{(p+j)} = 0.5 {g^{(j)}}^T Q^{(i)} g^{(j)} \\
l = \sum_{j=1}^{p-1} \sum_{k=j+1}^{p} 1: \quad & h_i^{(2p+l)} = {g^{(j)}}^T Q^{(i)} g^{(k)} + {g^{(k)}}^T Q^{(i)} g^{(j)}
\end{aligned}$$

Proof 3 *Inserting the definition of a zonotope into the set $\mathcal{Z}_Q = \{\varphi | \varphi_i = x^T Q^{(i)} x,\, x \in \mathcal{Z}\}$ yields*

$$\Big\{\varphi \Big| \varphi_i = (c + \sum_{j=1}^{p} \beta_j g^{(j)})^T Q^{(i)} (c + \sum_{j=1}^{p} \beta_j g^{(j)}),\, \beta_j \in [-1, 1]\Big\},$$

which can be rearranged to

$$\mathcal{Z}_Q = \Big\{\varphi \Big| \varphi_i = \underbrace{c^T Q^{(i)} c + \sum_{j=1}^{p} 0.5 {g^{(j)}}^T Q^{(i)} g^{(j)}}_{d_i} + \sum_{j=1}^{p} \beta_j \underbrace{(c^T Q^{(i)} g^{(j)} + {g^{(j)}}^T Q^{(i)} c)}_{h_i^{(j)}}$$
$$+ \sum_{j=1}^{p} (2\beta_j^2 - 1) \underbrace{0.5 {g^{(j)}}^T Q^{(i)} g^{(j)}}_{h_i^{(p+j)}} + \sum_{j=1}^{p-1} \sum_{k=j+1}^{p} \beta_j \beta_k \underbrace{({g^{(j)}}^T Q^{(i)} g^{(k)} + {g^{(k)}}^T Q^{(i)} g^{(j)})}_{h_i^{(2p+l)}},$$
$$\beta_i \in [-1, 1] \Big\} \subseteq \Big(d, h^{(1)}, \ldots, h^{(\sigma)}\Big).$$

The obtained zonotope is an over-approximation since $\beta_j \in [-1, 1]$, $(2\beta_j^2 - 1) \in [-1, 1]$, *and* $\beta_j \beta_k \in [-1, 1]$ *for* $j \neq k$. *The number of new generators is obtained from the fact that the new generators* $h^{(j)}$ *are computed by picking two elements from the set containing all generators and the center, where replacement is allowed and order does not matter. By subtracting the possibility that one can choose two centers, one obtains* $\sigma = \binom{p+2}{2} - 1$ *generators.*

The above Lemma is useful for the following abstraction of (8):

$$\begin{aligned} \mathcal{L} &= \{\frac{1}{2}(z - z^*)^T H^{(i)}(\xi)(z - z^*) | \xi, z \in \mathcal{R}^z\} \\ &\subseteq \{\frac{1}{2} z^T \mathcal{H}^{(i)} z | z \in \mathcal{R}^z_\Delta\}, \quad \mathcal{R}^z_\Delta = \mathcal{R}^z \oplus (-z^*), \quad \mathcal{H}^{(i)} = \{H^{(i)}(\xi) | \xi \in \mathcal{R}^z\} \end{aligned} \tag{9}$$

Lemma 4.1 is used in the following Theorem to over-approximate $\mathcal{L}$ in (9).

Theorem 4.1 (Linearization Error) *Let each* $\mathcal{H}^{(i)}$ *in* (9) *be bounded by an interval matrix, the linearization error* $\mathcal{L}$ *is over-approximated by*

$$\mathcal{L} \subseteq \frac{1}{2}\Big\{z^T \mathcal{H}^{(i)} z \Big| z \in \mathcal{R}^z_\Delta\Big\} \subseteq \frac{1}{2}\mathrm{quad}^{\mathrm{int}}(\mathcal{H}, \mathcal{R}^z_\Delta),$$

where

$$\mathrm{quad}^{\mathrm{int}}(\mathcal{H}, \mathcal{R}^z_\Delta) = \mathrm{quad}(H_c, \mathcal{R}^z_\Delta) \oplus [-\eta, \eta], \quad \eta := |\mathcal{R}^z_\Delta|^T H_\Delta |\mathcal{R}^z_\Delta|$$

and $H_c, H_\Delta \in \mathbb{R}^{n \times n}$ *such that* $\mathcal{H} = H_c \oplus [-H_\Delta, H_\Delta]$.

Proof 4 *For simplicity of notation, we first introduce the vectors* $\lambda^{(i)}$, *where* $\lambda^{(1)} = c$ *represents the center, and* $\lambda^{(2)}, \ldots, \lambda^{(\sigma+1)}$ *the generators of* $\mathcal{R}^z_\Delta$. *In Lemma 4.1, the operation* $\mathrm{quad}(\mathcal{H}, \mathcal{R}^z_\Delta)$ *is broken down into expressions of the form*

$${\lambda^{(i)}}^T (H_c \oplus [-H_\Delta, H_\Delta]) \lambda^{(j)} = \underbrace{{\lambda^{(i)}}^T H_c \lambda^{(j)}}_{\textit{fixed value}} \oplus \underbrace{{\lambda^{(i)}}^T [-H_\Delta, H_\Delta] \lambda^{(j)}}_{\textit{symmetric set}},$$

where the equality follows from the fact that the set-valued components have only a single occurrence. Due to the equality and the fact that each result has a fixed and symmetric part, we can conclude that

$$\text{quad}^{\text{int}}(\mathcal{H}, \mathcal{R}^z_\Delta) = \text{quad}(H_c, \mathcal{R}^z_\Delta) \oplus \text{quad}([-H_\Delta, H_\Delta], \mathcal{R}^z_\Delta)$$

One can further simplify $\text{quad}([-H_\Delta, H_\Delta], \mathcal{R}^z_\Delta)$ *to*

$$\text{quad}([-H_\Delta, H_\Delta], \mathcal{R}^z_\Delta) = \bigoplus_{i=1}^{p+1} \bigoplus_{j=1}^{p+1} \left([-1,1] \lambda^{(i)} [-H_\Delta, H_\Delta] \lambda^{(j)} \right) =$$

$$[-1,1] \sum_{i=1}^{p+1} \sum_{j=1}^{p+1} \left(|\lambda^{(i)}| H_\Delta |\lambda^{(j)}| \right) = [-1,1] \underbrace{\left(\sum_{i=1}^{p+1} |\lambda^{(i)}| \right)}_{= |\mathcal{R}^z_\Delta|, \text{ see (7)}} H_\Delta \underbrace{\left(\sum_{j=1}^{p+1} |\lambda^{(j)}| \right)}_{= |\mathcal{R}^z_\Delta|, \text{ see (7)}}$$

which concludes the proof.

In the next subsection, we provide a novel and an even tighter over-approximation of the Lagrange remainder.

4.4 Cubic Evaluation Using a Combination of Zonotopes and Intervals

The over-approximation of the Lagrange remainder can be further improved by not only considering second order terms of the multivariate Taylor expansion of (3), but also higher order terms. This makes it possible to consider the fixed linearization point z^* for the second order terms instead of $\xi \in \mathcal{R}^z$:

$$\dot{x}_i \in w_i + \frac{1}{1!} \sum_{j=1}^{o} C_{ij} z_j + \frac{1}{2!} \sum_{j=1}^{o} \sum_{k=1}^{o} D_{ijk} z_j z_k \oplus \left\{ \frac{1}{3!} \sum_{j=1}^{o} \sum_{k=1}^{o} \sum_{l=1}^{o} E_{ijkl}(\xi) z_j z_k z_l \Big| \xi \in \mathcal{R}^z \right\}. \quad (10)$$

Thus, given that $z \in \mathcal{R}^z$, the set of Lagrangian remainders using cubic Taylor terms is obtained as

$$\begin{aligned} \mathcal{L}_i &= \left\{ \frac{1}{2!} \sum_{j=1}^{o} \sum_{k=1}^{o} D_{ijk} z_j z_k + \frac{1}{3!} \sum_{j=1}^{o} \sum_{k=1}^{o} \sum_{l=1}^{o} E_{ijkl}(\xi) z_j z_k z_l \Big| z, \xi \in \mathcal{R}^z \right\} \\ &\subseteq \left\{ \frac{1}{2!} \sum_{j=1}^{o} \sum_{k=1}^{o} D_{ijk} z_j z_k \Big| z \in \mathcal{R}^z \right\} \oplus \left\{ \frac{1}{3!} \sum_{j=1}^{o} \sum_{k=1}^{o} \sum_{l=1}^{o} E_{ijkl}(\xi) z_j z_k z_l \Big| z, \xi \in \mathcal{R}^z \right\} \end{aligned} \quad (11)$$

Lemma 4.2 (Cubic Map) *Given a zonotope* $\mathcal{Z} = (g^{(0)}, \ldots, g^{(p)})$ *and a tensor* $C \in \mathbb{R}^{n \times n \times n \times n}$, $i = 1 \ldots n$, *the set*

$$\mathcal{Z}_C = \left\{ \lambda \Big| \lambda_i = \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} C_{ijkl} x_j x_k x_l,\ x \in \mathcal{Z} \right\}$$

is over-approximated by a zonotope

$$\text{cubic}(C, \mathcal{Z}) := (l^{(0)}, \ldots, l^{(\omega)}),$$

where ω is the number of obtained generators. The center is computed as

$$l_i^{(0)} = \sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} C_{ijkl} g_j^{(0)} g_k^{(0)} g_l^{(0)} + 0.5\sum_{s=1}^{p}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} C_{ijkl}\Big(g_j^{(s)} g_k^{(s)} g_l^{(0)} + g_j^{(s)} g_k^{(0)} g_l^{(s)} + g_j^{(0)} g_k^{(s)} g_l^{(s)}\Big),$$

and the generators are computed as

$$s = 1 \dots p: \quad h_i^{(s)} = \sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} C_{ijkl}\Big(g_j^{(s)} g_k^{(s)} g_l^{(s)}\Big),$$

$$s = 1 \dots p: \quad h_i^{(p+s)} = 0.5\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} C_{ijkl}\Big(g_j^{(s)} g_k^{(s)} g_l^{(0)} + g_j^{(s)} g_k^{(0)} g_l^{(s)} + g_j^{(0)} g_k^{(s)} g_l^{(s)}\Big),$$

$$l = \sum_{s=0}^{p-1}\sum_{t=s+1}^{p} 1: \quad h_i^{(2p+l)} = \sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} C_{ijkl}\Big(g_j^{(s)} g_k^{(s)} g_l^{(t)} + g_j^{(s)} g_k^{(t)} g_l^{(s)} + g_j^{(t)} g_k^{(s)} g_l^{(s)}\Big),$$

$$l = \sum_{s=0}^{p-2}\sum_{t=s+1}^{p-1}\sum_{u=t+1}^{p} 1: \quad h_i^{(0.5p^2+2.5p+l)} = \sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} C_{ijkl}\Big(g_j^{(s)} g_k^{(t)} g_l^{(u)} + g_j^{(s)} g_k^{(u)} g_l^{(t)} + g_j^{(t)} g_k^{(s)} g_l^{(u)} + g_j^{(t)} g_k^{(u)} g_l^{(s)} + g_j^{(u)} g_k^{(s)} g_l^{(t)} + g_j^{(u)} g_k^{(t)} g_l^{(s)}\Big).$$

The complexity of constructing this zonotope over-approximation with respect to the dimension n is $\mathcal{O}(n^5)$.

Proof 5 *Inserting the definition of a zonotope into the set*

$$\mathcal{Z}_C = \Big\{\varphi \Big| \varphi_i = \sum_{j=1}^{o}\sum_{k=1}^{o}\sum_{l=1}^{o} C_{ijkl} z_j z_k z_l,\ z \in \mathcal{Z}\Big\}$$

yields

$$\Big\{\varphi \Big| \varphi_i = \sum_{j=1}^{o}\sum_{k=1}^{o}\sum_{l=1}^{o} C_{ijkl}(c_j + \sum_{s=1}^{p}\beta_s g_j^{(s)})(c_k + \sum_{s=1}^{p}\beta_s g_k^{(s)})(c_l + \sum_{s=1}^{p}\beta_s g_l^{(s)}),\ \beta_j \in [-1,1]\Big\},$$

which can be rearranged to

$$\begin{aligned}\Big\{\varphi \Big| \varphi_i = \sum_{j=1}^{o}\sum_{k=1}^{o}\sum_{l=1}^{o} C_{ijkl}\Big(& c_j c_k c_l \\ &+ c_j c_k \sum_{s=1}^{p}\beta_s g_l^{(s)} + c_j \sum_{s=1}^{p}\beta_s g_k^{(s)} c_l + \sum_{s=1}^{p}\beta_s g_j^{(s)} c_k c_l \\ &+ c_j (\sum_{s=1}^{p}\beta_s g_k^{(s)})(\sum_{s=1}^{p}\beta_s g_l^{(s)}) + (\sum_{s=1}^{p}\beta_s g_j^{(s)}) c_k (\sum_{s=1}^{p}\beta_s g_l^{(s)}) \\ &+ (\sum_{s=1}^{p}\beta_s g_j^{(s)})(\sum_{s=1}^{p}\beta_s g_k^{(s)}) c_l \\ &+ (\sum_{s=1}^{p}\beta_s g_j^{(s)})(\sum_{s=1}^{p}\beta_s g_k^{(s)})(\sum_{s=1}^{p}\beta_s g_l^{(s)})\Big),\ \beta_j \in [-1,1]\Big\}\end{aligned}$$

This can be further simplified by factoring out the multiplication factors β_i:

$$\begin{aligned}\left\{\varphi \middle| \varphi_i = \sum_{j=1}^{o}\sum_{k=1}^{o}\sum_{l=1}^{o} C_{ijkl}\Big(& c_j c_k c_l \\ & + \sum_{s=1}^{p} \beta_s \Big(c_j c_k g_l^{(s)} + c_j g_k^{(s)} c_l + g_j^{(s)} c_k c_l \Big) \\ & + \sum_{s=1}^{p}\sum_{t=1}^{p} \beta_s \beta_t \Big(c_j g_k^{(s)} g_l^{(t)} + g_j^{(s)} c_k g_l^{(t)} + \beta_s g_j^{(s)} g_k^{(t)} c_l \Big) \\ & + \sum_{s=1}^{p}\sum_{t=1}^{p}\sum_{u=1}^{p} \beta_s \beta_t \beta_u \Big(g_j^{(s)} g_k^{(t)} g_l^{(u)} \Big) \Big),\ \beta_j \in [-1,1] \right\}\end{aligned}$$

Since the order of generator multiplications does not matter, we further group them since this reduces the number of combinations of β values, which reduces the number of obtained generators to over-approximate the cubic map. Thus, we obtain

$$\begin{aligned}\left\{\varphi \middle| \varphi_i = \sum_{j=1}^{o}\sum_{k=1}^{o}\sum_{l=1}^{o} C_{ijkl}\Big(& c_j c_k c_l \\ & + \sum_{s=1}^{p} \beta_s \Big(c_j c_k g_l^{(s)} + c_j g_k^{(s)} c_l + g_j^{(s)} c_k c_l \Big) \\ & + \sum_{s=1}^{p} \beta_s^2 \Big(c_j g_k^{(s)} g_l^{(s)} + g_j^{(s)} c_k g_l^{(s)} + \beta_s g_j^{(s)} g_k^{(s)} c_l \Big) \\ & + \sum_{s=1}^{p-1}\sum_{t=s+1}^{p} \beta_s \beta_t \Big(c_j g_k^{(s)} g_l^{(t)} + g_j^{(s)} c_k g_l^{(t)} + \beta_s g_j^{(s)} g_k^{(t)} c_l + c_j g_k^{(t)} g_l^{(s)} + g_j^{(t)} c_k g_l^{(s)} + \beta_s g_j^{(t)} g_k^{(s)} c_l \Big) \\ & + \sum_{s=1}^{p} \beta_s^3 \Big(g_j^{(s)} g_k^{(s)} g_l^{(s)} \Big) \\ & + \sum_{s=1}^{p-1}\sum_{t=s+1}^{p} \beta_s^2 \beta_t \Big(g_j^{(s)} g_k^{(s)} g_l^{(t)} + g_j^{(s)} g_k^{(t)} g_l^{(s)} + g_j^{(t)} g_k^{(s)} g_l^{(s)} \Big) \\ & + \sum_{s=1}^{p-2}\sum_{t=s+1}^{p-1}\sum_{u=t+1}^{p} \beta_s \beta_t \beta_u \Big(g_j^{(s)} g_k^{(t)} g_l^{(u)} + g_j^{(s)} g_k^{(u)} g_l^{(t)} + g_j^{(t)} g_k^{(s)} g_l^{(u)} \\ & \qquad + g_j^{(t)} g_k^{(u)} g_l^{(s)} + g_j^{(u)} g_k^{(s)} g_l^{(t)} + g_j^{(u)} g_k^{(t)} g_l^{(s)} \Big) \Big),\ \beta_j \in [-1,1] \right\}\end{aligned}$$

Given that $\beta_i \in [-1,1]$ we have that $\beta_s, \beta_s\beta_t, \beta_s^3, \beta_s^2\beta_t, \beta_s\beta_t\beta_u \in [-1,1]$ and that $\beta_s^2 \in [0,1]$. Since β_s^2 is always positive, we split the term multiplied by β_s^2 into a center part and its

remainder:

$$\Big\{\varphi\Big|\varphi_i = \sum_{j=1}^{o}\sum_{k=1}^{o}\sum_{l=1}^{o} C_{ijkl}\Big(c_j c_k c_l + 0.5\sum_{s=1}^{p}\Big(c_j g_k^{(s)} g_l^{(s)} + g_j^{(s)} c_k g_l^{(s)} + \beta_s g_j^{(s)} g_k^{(s)} c_l\Big)$$
$$+ \sum_{s=1}^{p}\beta_s\Big(c_j c_k g_l^{(s)} + c_j g_k^{(s)} c_l + g_j^{(s)} c_k c_l\Big)$$
$$+ \sum_{s=1}^{p}(2\beta_s^2-1)0.5\Big(c_j g_k^{(s)} g_l^{(s)} + g_j^{(s)} c_k g_l^{(s)} + \beta_s g_j^{(s)} g_k^{(s)} c_l\Big)$$
$$+ \sum_{s=1}^{p-1}\sum_{t=s+1}^{p}\beta_s\beta_t\Big(c_j g_k^{(s)} g_l^{(t)} + g_j^{(s)} c_k g_l^{(t)} + \beta_s g_j^{(s)} g_k^{(t)} c_l + c_j g_k^{(t)} g_l^{(s)} + g_j^{(t)} c_k g_l^{(s)} + \beta_s g_j^{(t)} g_k^{(s)} c_l\Big)$$
$$+ \sum_{s=1}^{p}\beta_s^3\Big(g_j^{(s)} g_k^{(s)} g_l^{(s)}\Big)$$
$$+ \sum_{s=1}^{p-1}\sum_{t=s+1}^{p}\beta_s^2\beta_t\Big(g_j^{(s)} g_k^{(s)} g_l^{(t)} + g_j^{(s)} g_k^{(t)} g_l^{(s)} + g_j^{(t)} g_k^{(s)} g_l^{(s)}\Big)$$
$$+ \sum_{s=1}^{p-2}\sum_{t=s+1}^{p-1}\sum_{u=t+1}^{p}\beta_s\beta_t\beta_u\Big(g_j^{(s)} g_k^{(t)} g_l^{(u)} + g_j^{(s)} g_k^{(u)} g_l^{(t)} + g_j^{(t)} g_k^{(s)} g_l^{(u)}$$
$$+ g_j^{(t)} g_k^{(u)} g_l^{(s)} + g_j^{(u)} g_k^{(s)} g_l^{(t)} + g_j^{(u)} g_k^{(t)} g_l^{(s)}\Big)\Big),\ \beta_j \in [-1,1]\Big\}$$

By using $c_j = g_j^{(0)}$ *one can integrate* $\Big(c_j c_k g_l^{(s)} + c_j g_k^{(s)} c_l + g_j^{(s)} c_k c_l\Big)$ *into* $\Big(g_j^{(s)} g_k^{(s)} g_l^{(t)} + g_j^{(s)} g_k^{(t)} g_l^{(s)} + g_j^{(t)} g_k^{(s)} g_l^{(s)}\Big)$ *and* $\Big(c_j g_k^{(s)} g_l^{(t)} + g_j^{(s)} c_k g_l^{(t)} + \beta_s g_j^{(s)} g_k^{(t)} c_l + c_j g_k^{(t)} g_l^{(s)} + g_j^{(t)} c_k g_l^{(s)} + \beta_s g_j^{(t)} g_k^{(s)} c_l\Big)$ *into* $\Big(g_j^{(s)} g_k^{(t)} g_l^{(u)} + g_j^{(s)} g_k^{(u)} g_l^{(t)} + g_j^{(t)} g_k^{(s)} g_l^{(u)} + g_j^{(t)} g_k^{(u)} g_l^{(s)} + g_j^{(u)} g_k^{(s)} g_l^{(t)} + g_j^{(u)} g_k^{(t)} g_l^{(s)}\Big)$. *Considering that* $\beta_s, (2\beta_s^2-1), \beta_s\beta_t, \beta_s^3, \beta_s^2\beta_t, \beta_s\beta_t\beta_u \in [-1,1]$ *we obtain the result of the lemma.*

Now that we can compute cubic maps of zonotopes, we can over-approximate the computation in (11). Given the set of possible values of $E_{ijkl}(\xi)$ as $\mathcal{E} = \{E_{ijkl}(\xi)|\xi \in \mathcal{R}^z\}$ we define the center and radius of this set as $E_c \oplus [-E_\Delta, E_\Delta] \supseteq \mathcal{E}$. Using the center and the radius, we can over-approximate the cubic map as

$$\begin{aligned}\mathcal{L}_i &\subseteq \Big\{\frac{1}{2!}\sum_{j=1}^{o}\sum_{k=1}^{o} D_{ijk} z_j z_k \Big| z \in \mathcal{R}^z\Big\} \oplus \Big\{\frac{1}{3!}\sum_{j=1}^{o}\sum_{k=1}^{o}\sum_{l=1}^{o} E_{ijkl}(\xi) z_j z_k z_l \Big| z, \xi \in \mathcal{R}^z\Big\} \\ &\subseteq \Big\{\frac{1}{2!}\sum_{j=1}^{o}\sum_{k=1}^{o} D_{ijk} z_j z_k \Big| z \in \mathcal{R}^z\Big\} \oplus \Big\{\frac{1}{3!}(\sum_{j=1}^{o}\sum_{k=1}^{o}\sum_{l=1}^{o} E_c z_j z_k z_l \Big| z, \xi \in \mathcal{R}^z\Big\} \oplus [-\omega_i, \omega_i]) \\ &\subseteq \frac{1}{2!}\mathrm{quad}(D, \mathcal{R}^z) \oplus \frac{1}{3!}\Big(\mathrm{cubic}(E, \mathcal{R}^z) \oplus [-\omega_i, \omega_i]\Big)\end{aligned} \tag{12}$$

where

$$\omega_i = \sum_{j=1}^{o}\sum_{k=1}^{o}\sum_{l=1}^{o} E_{\Delta,ijkl} |\mathcal{R}^z|_j |\mathcal{R}^z|_k |\mathcal{R}^z|_l.$$

4.5 Application to a Smart Grid Use Case

The load-following capabilities of power plants became increasingly important in the recent years in order to ensure a reliable operation of future power systems. We use reachability analysis techniques developed in UnCoVerCPS to rigorously verify safety of critical components that often pose limitations on the flexibility of power plants to perform fast load changes. The previously proposed reachability algorithms makes it possible to compute the bounds of all possible trajectories for a range of operating conditions, while simultaneously meeting the practical requirements of a real power plant. As an example, we consider the verification of the water level inside a drum unit. In contrast to any previous work, our results are based on measurement data of a realistic configuration of a boiler system located within a 450 MW combined cycle plant in Germany. Furthermore, we use an abstract model which considers the modeling errors, thus ensuring that all dynamical behaviors of the process are replicated by the abstraction. As a result, we formally guarantee that the water level inside the drum always remains within safe limits for load changes equivalent to 40 MW, thus maximally exploiting the power plant adaptability and its load following capabilities.

Formal verification of the steam-drum unit was initially proposed in [26] as a benchmark problem for formal analysis and controller synthesis. It attracted considerable attention as an interesting theoretical problem and became a classical case study for testing and comparing formal methods, see [27, 28, 29]. The main drawback of the benchmark problem is that the modeling is based on elementary assumptions and abstract decisions, which do not capture any of the complicated dynamics of the process. Hence, it does not hold for formal analysis when considering real practical problems. In contrast to the benchmark problem, we consider a realistic configuration of a steam-drum unit using a well-developed nonlinear model, which is approximated using a polynomial function and validated against measurements data. The unit is located within the 450 MW combined cycle power plant (*München Süd GuD*), owned by SWM Services GmbH (Munich City Utilities).

The on-the-fly reachability algorithm developed in UnCoVerCPS is efficient enough to meet the real-time requirements of the considered power plant for secondary frequency control (5 *min*) or tertiary control (15 *min*). It can establish in advance, whether a requested load change imposed by the transmission system operator will trigger the water level safety limits. By doing so, the plant operator can accept or reject the requested load dispatch, thus preventing unnecessary shutdowns of the facility. On the one hand, the power plant avoids drastic economical losses, and on the other hand, the transmission system operator evades a sudden loss of one of its generating units, which jeopardizes the stability of its balancing

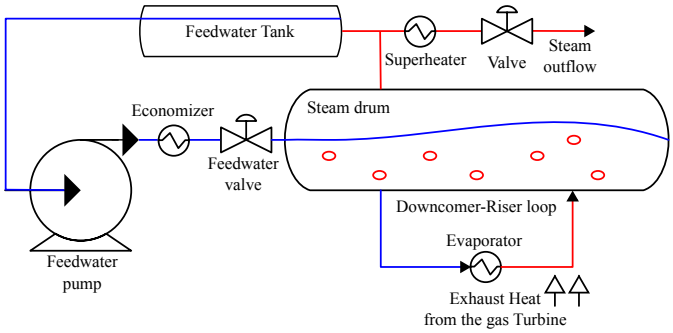


Figure 6: Simplified description of the steam generation process using a steam drum unit. Red line indicates hot steam and the blue line indicates cold water.

area.

4.5.1 Process Modeling

The simplified diagram of the steam generation process is shown in Fig. 6. Cold water inside the feedwater tank is pumped and heated at the economizer stage before going through the drum inlet. Due to the gravitational force, feedwater flows through the naturally circulated downcomer riser loop, where it is converted into steam at the evaporator stage. Different riser tubes collect the steam and supply it back into the drum. In the final stage, the saturated steam is taken from the drum outlet to the superheater. The model consists of four components, i.e. the drum unit, the regulating valves, the exhaust heat, and the controller. We refer the reader to previous works by the authors [30, 31] for further details with regards to the modeling assumptions and controller design.

Steam-Drum Model We consider the well-developed Åström - Bell model [32] for the drum unit. To easily represent the equations, let $V\,[\mathrm{m}^3]$ represent the volume, $Q\,[\mathrm{J/s}]$ and $q\,[\mathrm{kg/s}]$ denote heat and mass flow rates, respectively, $r\,[\%]$ is the valve opening percentage, $P_D\,[\mathrm{MW}]$ is the electrical power, $\rho[\mathrm{m}^3/\mathrm{kg}]$ is the density, $h\,[\mathrm{J/kg}]$ is the specific enthalpy, and $T\,[\mathrm{s}]$ represents the residence time. Additionally, v and τ are the gain and time constant for a first-order lag element. The subscripts s, wt, f, c, dc, $\circ$ and D refer to steam, water, feedwater, condensation, downcomer, hypothetical situation and current demand, respectively.

The drum state variables are the pressure $\bar{P}\,[\mathrm{Pascal}]$, the water volume V_{wt}, the steam-mass quality $\alpha_r\,[\%]$ in the riser tubes and the steam bubbles volume V_{sd} under the water level. The inputs are the heat flow rate Q, the feedwater q_f and steam q_s flow rates. Using

the nonlinear variables e_{nm} found in [32], the nonlinear drum model is expressed by:

$$\begin{aligned}
\dot{\bar{P}} &= \frac{e_{11}Q + q_f(e_{11}h_f - e_{21}) - q_s(e_{21} - e_{11}h_s)}{e_{11}e_{22} - e_{12}e_{21}} \\
\dot{V}_{wt} &= \frac{Q + q_f h_f - q_s h_s - e_{22}\dot{\bar{P}}}{e_{21}} \\
\dot{\alpha}_r &= \frac{Q - \alpha_r h_c q_{dc} - e_{31}\dot{\bar{P}}}{e_{33}} \\
\dot{V}_{sd} &= \frac{1}{T}(V_\circ - V_{sd}) - q_f\frac{h_f - h_{wt}}{e_{44}h_c} - \frac{e_{41}\dot{\bar{P}} + e_{43}\dot{\alpha}_r}{e_{44}}.
\end{aligned} \tag{13}$$

Controller Model The drum is controlled using an observer-based state feedback controller (see [31]) with a controller to ensure that the drum pressure P [bar] and water level l [mm] track their corresponding reference signals w_p and w_l. The integrated errors $h := [\eta, \psi]^T$ are treated as artificial state variables:

$$\dot{\eta} = w_P - P, \quad \dot{\psi} = w_l - l. \tag{14}$$

The state-feedback controller generates the control action for the feedwater flow $\hat{q}_f$ and the steam valve opening percentage $\hat{r}_s$ to preserve the water level and pressure at the desired reference values according to

$$\begin{bmatrix} \hat{q}_f \\ \hat{r}_s \end{bmatrix} = \underbrace{\begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix}}_{K} \begin{bmatrix} x \\ h \end{bmatrix}, \tag{15}$$

where K is the state feedback matrix. The feedwater control action $\hat{q}_f$ is compared to the actual flow rate, and the deviation between both generates the regulating valve control action $\hat{r}_f$ using a PI-controller

$$\hat{r}_f = v_{pf}\left(1 + v_{if}\int_0^t (\hat{q}_f - q_f)dt\right), \tag{16}$$

where v_{pf}, v_{if} are the proportional and integrator scalar gains of the PI-controller.

Model Extension The heat flow rate Q is regarded as an additional state and not treated as an input variable due to the combined-cycle nature of the process. The heat is directly associated with the gas turbine exhaust temperature, which corresponds to the electrical power demand P_D established by the transmission system operator. The heat flow rate is modeled as:

$$\dot{Q} = \frac{v_D P_D - Q}{\tau_D}, \tag{17}$$

and the regulation through a control valve for feedwater and steam flow rates is

$$q_f = r_f \frac{k_{vf}\rho_w\sqrt{\Delta P}}{3600}, \qquad \dot{r}_f = \frac{v_f\hat{r}_f - r_f}{\tau_f}, \tag{18}$$

$$q_s = r_s \frac{13.6\, k_{vs}\sqrt{\rho_s P}}{3600}, \qquad \dot{r}_s = \frac{v_s\hat{r}_s - r_s}{\tau_s}, \tag{19}$$

where ΔP [bar] is the pressure drop across the valve, and k_v [kg/hr] is the valve sizing coefficient.

Combining the results (13)-(19), one obtains an 11-th order model with the input signal $u = [P_D]$, the state variables $x = [\bar{P},\, V_{wt},\, \alpha_r,\, V_{sd},\, \eta,\, \psi,\, Q,\, \hat{r}_f,\, r_f,\, r_s,\, \hat{q_f}]^T$, and the output vector $y = [P,\, l]^T$.

4.5.2 Model Abstraction

The model derived in Sec. 4.5.1 is a mathematical model of the real process and thus suffers from imperfections in modeling of actual physical phenomena. Furthermore, derived models often contain complicated nonlinear expressions that are challenging for a formal analysis. To solve the aforementioned problems, an abstraction described by a polynomial differential inclusion $f(x,u) \in f^{abstract}(x,u) \oplus \mathcal{L}$ based on measurements data is proposed, where $f^{abstract}(x,u)$ is chosen as in (2). The measurement data is used to obtain $\mathcal{L}$ using a state observer. The reachable set of this model is next computed based on on-the-fly linearization.

4.5.3 Discussion of Results

First we present the validation of the polynomial model of Sec. 4.5.2 against measurement data to show its ability to capture the dynamical behavior of the real process. The validation data covers almost the entirety of the gas turbine operational range, i.e. from 70 MW to 120 MW in both directions (increased/decreased generation). The data is collected over a period of one year during engagement of one of the authors with the plant *München Süd*. We investigate the safety of the water level against high-load transitions (≤ 40 MW) by computing the over-approximative reachable set of the system. The computations are performed on a standard computer with an Intel Core i7-4810MQ CPU.

Validation of the Polynomial Model The model (13) is initially realized in MATLAB R2014b using the Symbolic toolbox. It is approximated by a polynomial function using the Taylor expansion at $P_D = 95$ MW (center of the gas turbine operational range). The validation procedure is carried out by simulating the models (1-st, 2-nd and 3-rd order polynomial functions) and comparing the simulation results to the experimental data. All simulations are performed using the `ODE45` solver.

It is shown in Fig. 7 and Fig. 8 that the polynomial models replicate the dynamical behavior of the real process and most importantly the shrink and swell physical phenomena.

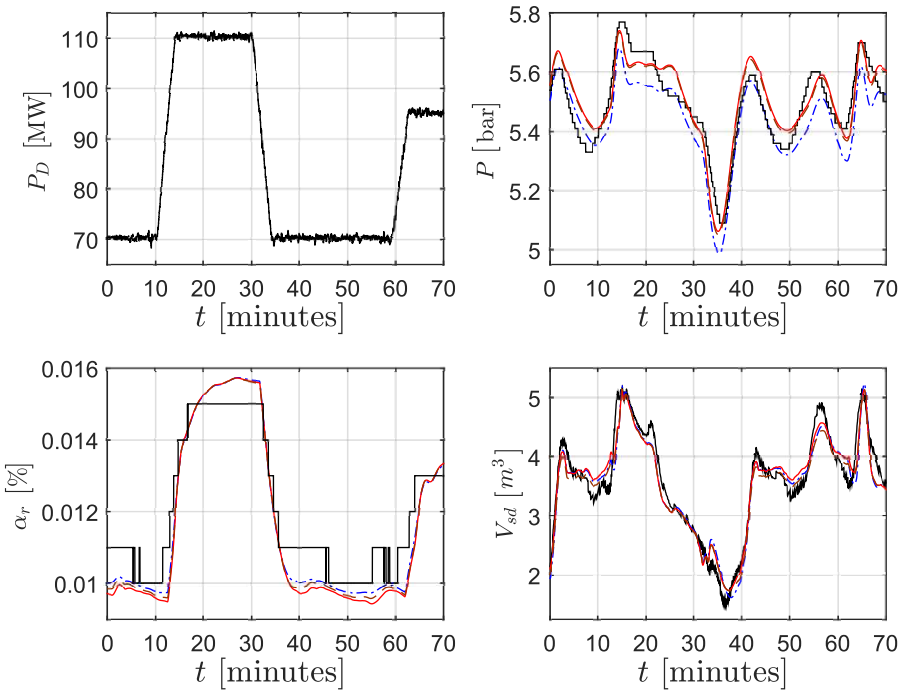


Figure 7: Comparison between data measurements (black solid), 1-st (blue dashed dotted), 2-nd (red solid) and 3-rd (brown dotted) order polynomial model for perturbations of the gas turbine power.

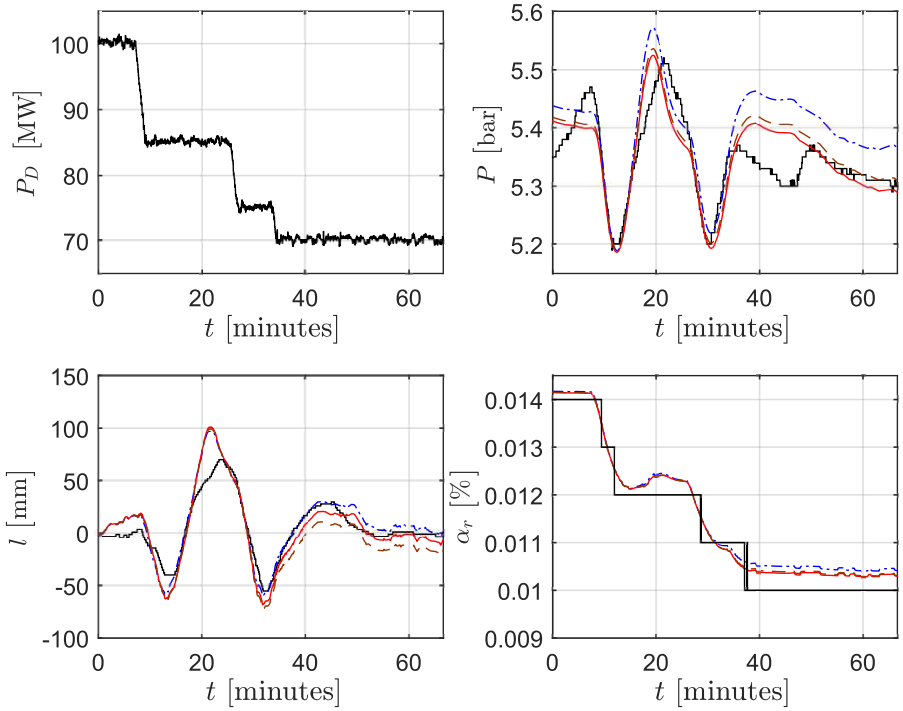


Figure 8: Comparison between data measurements (black solid), 1-st (blue dashed dotted), 2-nd (red solid) and 3-rd (brown dotted) order polynomial model for decrease of the gas turbine power from 100 MW to 70 MW.

It should be made clear that the model inaccuracy is acknowledged and identifiable. It is caused due to the nature of the proposed polynomial approximation, in addition to the simplification of certain components during the modeling procedure. The remaining inaccuracy is systematically obtained and added as additional uncertain input. Details of this procedure will be shown in a future deliverable.

Load-following Safety Verification The reachable set is computed for a time horizon $t_f = 5\,\text{min}$ with a time step of 1 sec using *CORA*. The time horizon is chosen according to the practical requirements of the power plant; when *München Süd* is subjected to secondary frequency control, it is notified by the transmission system operator 5 min earlier to meet a load change equivalent to 40 MW.

The task is to guarantee that the water level l inside the drum does not surpass $\pm 300\,\text{mm}$. If the limits are triggered, it leads to tripping of the boiler as a safety precautions to protect

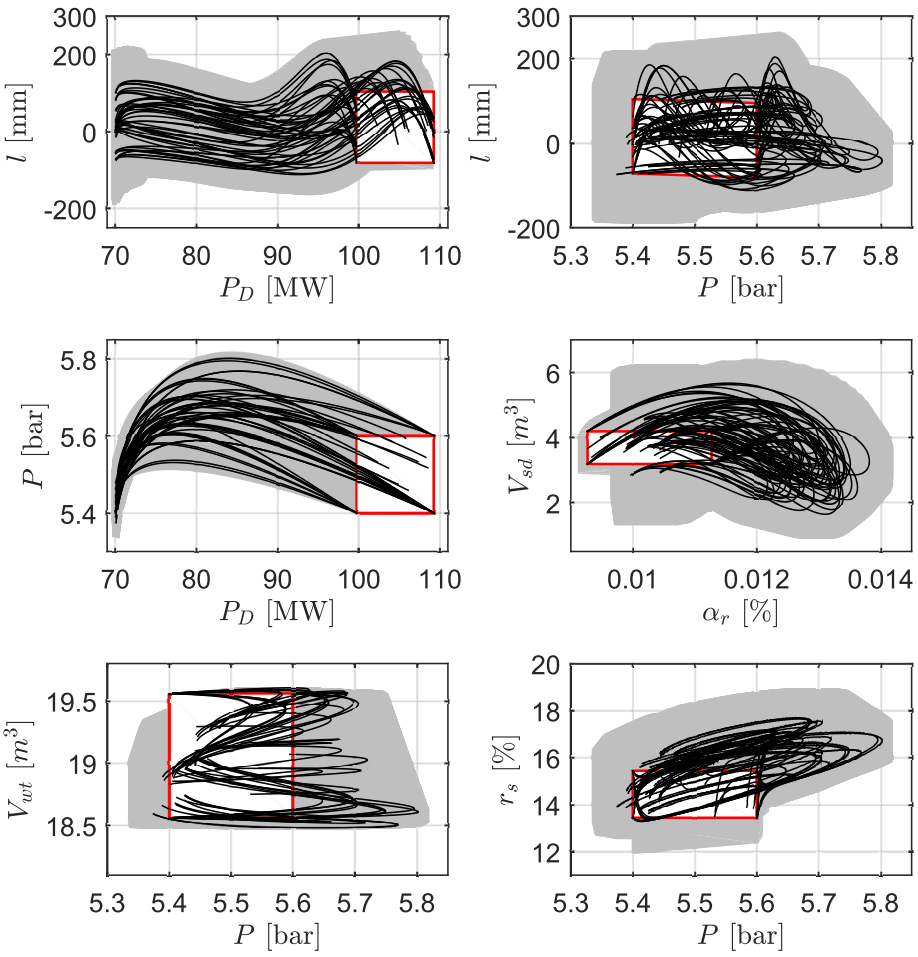


Figure 9: Selected projections of the reachable set for a load-change of the gas turbine from 80 MW to 120 MW. Black lines represents random simulation results ($n = 50$), the gray area shows the reachable set, the red box is the initial set of state variables $\mathcal{R}(0)$.

critical components, e.g., the superheater and the steam-turbine. During that time, the power plant is no longer operational and unable to meet load requirements of the transmission system operator.

Table 1 shows a comparison between the computational time of reachability analysis for different models. Using the 3-rd order polynomial model, It takes 206.373 sec to compute the reachable set, where the calculation of the Lagrangian remainder consumes 96 % of the time. The computational time meets practical requirements of the plant, as it allows on-the-fly verification of the process safety for high-load transitions in the requested time ($t \leq 5$ min). To speed up the computational time, one may use the 2-nd order polynomial model with different modeling errors.

Using the model (13) [32] without a polynomial approximation, it takes 57.2 min to compute the reachable set for a time-horizon of 10 sec. The huge difference in the computational time is due to the complicated nonlinear expression, which were simplified with a polynomial function, thus substantially reducing the computational complexity and making the algorithm feasible for practical problems.

The reachable set projections of chosen state variables for the 2-nd order polynomial model are shown in Fig. 9 and Fig. 10. The load change is equivalent to 40 MW which is the maximum load that can be requested by the transmission system operator. It is clear that the water level does not hit the safety limits (the same also holds for the 3-rd order polynomial model), hence the safety of the steam drum is formally verified under the assumption that the set $\mathcal{E}$ contains all modelling errors.

Table 1: Comparison between the computational time of reachability analysis and numerical simulations

Model	t_f	$\mathcal{R}([0, t_f])$	$\mathcal{L}([0, t_f])$
2-nd	5 min	41.4183 s	34.064 s
3-rd	5 min	206.373 s	198.534 s
[32]	10 s	3420 s	3078 s

The final results show that it is computationally feasible to implement the proposed reachability algorithm, while meeting the practical requirements of a real power plant. The reachability algorithm can be easily integrated into a Distributed Control System (DCS), in parallel to the existing control structure, and operates automatically without any interaction from the operator's side. The operators have the opportunity to accept or reject the requested load transitions, since reachability analysis establishes in advance whether the demanded load

change shall trigger the water level safe limit, thus avoiding unnecessary shutdown of the facility.

5 On-The-Fly Polynomialization

The previous section shows that abstracting nonlinear continuous dynamics to linear differential inclusions results in a scalable approach for reachability analysis. However, when the abstraction becomes inaccurate, linearization techniques require splitting of reachable sets, resulting in an exponential growth of required linearizations. In this section, the dynamics is more accurately abstracted to polynomial difference inclusions. Thus, it is no longer guaranteed that reachable sets of consecutive time steps are mapped to convex sets, requiring a non-convex set representation, resulting in no or less splitting.

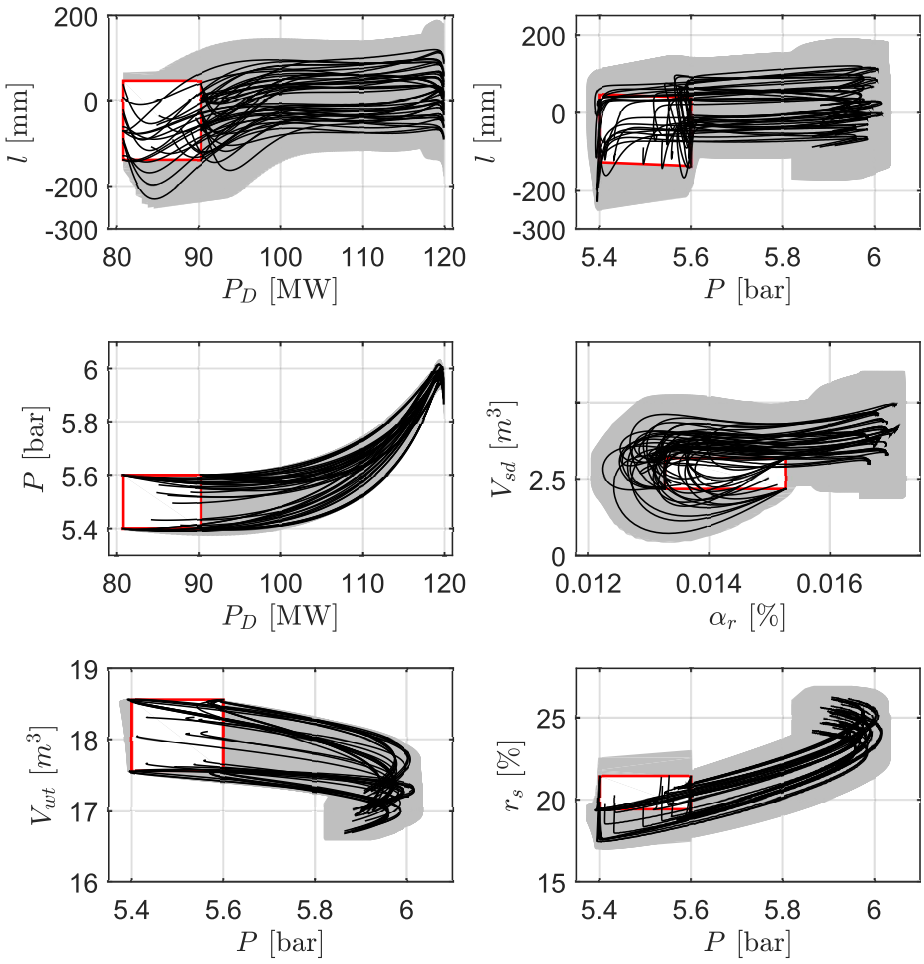


Figure 10: Selected projections of the reachable set for a load-change of the gas turbine from 110 MW to 70 MW. Black lines represents random simulation results ($n = 50$), the gray area shows the reachable set, the red box is the initial set of state variables $\mathcal{R}(0)$.

5.1 Abstraction to Difference Inclusions

As a first step, all higher order terms in (2) are interpreted as an input v to a linear system:

$$\begin{aligned} \dot{x} &\in Ax(t) + v(z(t), u(t)) \oplus \mathcal{L}(t) \\ v_i(z, u) &= w_i + \sum_{j=1}^{m} B_{ij} u_j + \frac{1}{2!} \sum_{j=1}^{o} \sum_{k=1}^{o} D_{ijk} z_j z_k + \dots \end{aligned} \tag{20}$$

In order to obtain a tight over-approximation, the auxiliary variables $u^{\Delta}(t) = u(t) - u^c$, $z^{\Delta}(t) = z(t) - z(t_s)$ are introduced to split the input $v(z(t), u(t))$ for $t \in \tau_s$ into a constant part $v(z(t_s), u^c)$ fixed at the specific point in time t_s and a time-varying part $v^{\Delta}(z^{\Delta}(t), z(t_s), u^{\Delta}(t))$:

$$\begin{aligned} v_i(z(t), u(t)) &= w_i + \sum_{j=1}^{m} B_{ij}(u_j^c + u_j^{\Delta}(t)) + \frac{1}{2!} \sum_{j=1}^{o} \sum_{k=1}^{o} D_{ijk}(z_j(t_s) + z_j^{\Delta}(t))(z_k(t_s) + z_k^{\Delta}(t)) + \dots \\ &= v(z(t_s), u^c) + v^{\Delta}(z^{\Delta}(t), z(t_s), u^{\Delta}(t)) \end{aligned}$$

where

$$\begin{aligned} v(z(t_s), u^c) &= w_i + \sum_{j=1}^{m} B_{ij} u_j^c + \frac{1}{2!} \sum_{j=1}^{o} \sum_{k=1}^{o} D_{ijk} z_j(t_s) z_k(t_s) + \dots \\ v^{\Delta}(z^{\Delta}(t), z(t_s), u^{\Delta}(t)) &= \sum_{j=1}^{m} B_{ij} u_j^{\Delta}(t) + \frac{1}{2!} \sum_{j=1}^{o} \sum_{k=1}^{o} D_{ijk} \\ &\quad \Big(z_j(t_s) z_k^{\Delta}(t) + z_j^{\Delta}(t) z_k(t_s) + z_j^{\Delta}(t) z_k^{\Delta}(t) \Big) + \dots \end{aligned} \tag{21}$$

After defining $\mathcal{U}^{\Delta} := \mathcal{U} \oplus (-u^c)$ and assuming that the reachable set $\mathcal{R}(\tau_s)$ and

$$\mathcal{R}^{\Delta}(\tau_s) := \Big\{ \chi(t; x(t_s), u(\cdot)) - x(t_s) \Big| t \in \tau_s,\, x(t_s) \in \mathcal{R}(t_s), \forall t \in \tau_s :\, u(t) \in \mathcal{U} \Big\} \tag{22}$$

are already known, where $\chi()$ is the solution of (3), the set of possible values of $v^{\Delta}(z^{\Delta}(t), z(t_s), u^{\Delta}(t))$ is bounded by

$$\mathcal{V}^{\Delta}(\tau_s) := \Big\{ v^{\Delta}(z^{\Delta}, z, u^{\Delta}) \Big| z^{\Delta} \in \mathcal{R}^{\Delta}(\tau_s) \times \mathcal{U}^{\Delta}, z \in \mathcal{R}(\tau_s) \times \mathcal{U},\, u^{\Delta} \in \mathcal{U}^{\Delta} \Big\}. \tag{23}$$

Using (20) - (23), the linear differential inclusion

$$\dot{x} \in Ax(t) + v(z(t_s), u^c) \oplus \big(\mathcal{V}^{\Delta}(\tau_s) \oplus \mathcal{L}(\tau_s) \big)$$

is obtained for $t \in \tau_s$. Due to the superposition principle of linear systems, the solution is obtained by adding the solution of the homogeneous solution $x^h(t_{s+1})$, the input solution due to constant input $x^{p,c}(r)$, where $r = t_{s+1} - t_s$, and the input solution set due to time-varying inputs $\mathcal{R}^{p,\Delta}(\mathcal{V}^{\Delta}(\tau_s) \oplus \mathcal{L}(\tau_s), r)$ to

$$x(t_{s+1}) \in x^h(t_{s+1}) + x^{p,c}(t) \oplus \mathcal{R}^{p,\Delta}\big(\mathcal{V}^{\Delta}(\tau_s) \oplus \mathcal{L}(\tau_s), r\big). \tag{24}$$

The well-known homogeneous solution is $x^h(t_{s+1}) = e^{Ar}x(t_s)$, the input solution due to constant input is

$$x^{p,c}(r) = \Gamma(r)\, v(z(t_s), u^c), \quad \Gamma(r) := \int_0^r e^{A(r-t)}dt,$$

where $\Gamma(r) = A^{-1}(e^{Ar} - I)$ (I is the identity matrix) and when A is not invertible, the approach in [33] is used. The reachable set $\mathcal{R}^{p,\Delta}(\tilde{\mathcal{V}}(\tau_s), r)$ due to the set of uncertain time-varying inputs within $\tilde{\mathcal{V}}(\tau_s) := \mathcal{V}^{\Delta}(\tau_s) \oplus \mathcal{L}(\tau_s)$ is computed as in [33].

The difference to previous approaches (e.g. [19, 20]) is that due to the separation in a constant and a time-varying input, nonlinear terms are saved from linearization at times t_s, while within τ_s, an abstracting linear differential inclusion is used. Inserting $v(z(t_s), u^c)$ from (21) into the overall solution (24) results in a nonlinear difference equation that encloses the exact solution:

$$\begin{aligned} x_i(t_{s+1}) \in & \sum_{j=1}^{n}(e^{Ar})_{ij}x_j(t_s) + \sum_{j=1}^{n}\Gamma_{ij}(r)\Big(w_j + \sum_{k=1}^{m}B_{jk}u_k^c + \frac{1}{2!}\sum_{k=1}^{o}\sum_{l=1}^{o}D_{jkl}z_k(t_s)z_l(t_s) + \dots\Big) \\ & \oplus \mathcal{R}_i^{p,\Delta}(\mathcal{V}^{\Delta}(\tau_s) \oplus \mathcal{L}(\tau_s), r) \end{aligned} \tag{25}$$

The benefits of the above difference inclusion for reachability analysis do not immediately show. Note that for small time increments r, as typically used in reachability analysis, the set $\mathcal{R}_i^{p,\Delta}$ becomes small, no matter how large the set of possible values of $z(t_s)$ becomes during the reachability analysis. Thus, for large sets of $z(t_s)$, the nonlinearity is well captured by all other terms, while the abstractions in $\mathcal{R}_i^{p,\Delta}$ are not dominant. By replacing exact values with sets in (25), we obtain the reachable set at the next point in time:

$$\mathcal{R}(t_{s+1}) = \underbrace{e^{Ar}\mathcal{R}(t_s)}_{=:\mathcal{PZ}_1} \oplus \mathcal{R}^{p,\Delta}(\mathcal{V}^{\Delta}(\tau_s) \oplus \mathcal{L}(\tau_s), r) \oplus \underbrace{\Gamma(r)\Big(w \oplus Bu^c \oplus \frac{1}{2!}\mathrm{sq}(D, \mathcal{R}_z(t_s))\Big)}_{=:\mathcal{PZ}_2} \tag{26}$$

The new approach includes nonlinear mappings so that in general convex sets are no longer mapped to convex sets as in other works, requiring a new non-convex set representation as presented in the following subsection.

5.2 Polynomial Zonotopes

Set representations in most previous works are convex since they are easy to represent and manipulate (see e.g. [13, 14, 15, 16, 17, 9, 34, 19]). However, the convexity property makes the efforts in capturing the nonlinear dynamics obsolete, since convex sets only work well for linear maps. A non-convex set representation is presented, which can be efficiently stored

and manipulated. The proposed representation shares many similarities with Taylor models [35] (as shortly discussed later) and is a generalization of zonotopes, which have shown great performance for linear and nonlinear reachability analysis [14, 19].

Definition 5.1 (Polynomial Zonotope) *Given a starting point $c \in \mathbb{R}^n$, multi-indexed generators $f^{([i],j,k,\dots,m)} \in \mathbb{R}^n$, and single-indexed generators $g^{(i)} \in \mathbb{R}^n$, a polynomial zonotope is defined as*

$$\mathcal{PZ} = \Big\{ c + \sum_{j=1}^{p} \beta_j f^{([1],j)} + \sum_{j=1}^{p}\sum_{k=j}^{p} \beta_j \beta_k f^{([2],j,k)} + \dots + \sum_{j=1}^{p}\sum_{k=j}^{p} \dots \sum_{m=l}^{p} \underbrace{\beta_j \beta_k \dots \beta_m}_{\eta \text{ factors}} f^{([\eta],j,k,\dots,m)} + \sum_{i=1}^{q} \gamma_i g^{(i)} \Big| \beta_i, \gamma_i \in [-1,1] \Big\}. \tag{27}$$

The scalars β_i are called dependent factors, since changing their value does not only affect the multiplication with one generator, but other generators, too. On the other hand, the scalars γ_i only affect the multiplication with one generator, so that they are called independent factors. The number of dependent factors is p, the number of independent factors is q, and the polynomial order η is the maximum power of the scalar factors β_i. The order of a polynomial zonotope is defined as the number of generators ξ divided by the dimension, which is $\rho = \frac{\xi}{n}$. For a concise notation and later derivations, we introduce the matrices

$$\begin{aligned} E^{[i]} &= [\underbrace{f^{([i],1,1,\dots,1)}}_{=:e^{([i],1)}} \dots \underbrace{f^{([i],p,p,\dots,p)}}_{=:e^{([i],p)}}] \text{ (equal indices)}, \\ F^{[i]} &= [f^{([i],1,1,\dots,1,2)}\ f^{([i],1,1,\dots,1,3)}\ \dots\ f^{([i],1,1,\dots,1,p)} \\ &\quad\ f^{([i],1,1,\dots,2,2)}\ f^{([i],1,1,\dots,2,3)}\ \dots\ f^{([i],1,1,\dots,2,p)} \\ &\quad\ f^{([i],1,1,\dots,3,3)}\ \dots] \text{ (unequal indices)}, \\ G &= [g^{(1)}\ \dots\ g^{(q)}], \end{aligned}$$

and $E = \begin{bmatrix} E^{[1]} & \dots & E^{[\eta]} \end{bmatrix}$, $F = \begin{bmatrix} F^{[2]} & \dots & F^{[\eta]} \end{bmatrix}$ ($F^{[i]}$ is only defined for $i \geq 2$). Note that the indices in $F^{[i]}$ are ascending due to the nested summations in (5.1)*. In short form, a polynomial zonotope is written as $\mathcal{PZ} = (c, E, F, G)$.*

For a given polynomial order i, the total number of generators in $E^{[i]}$ and $F^{[i]}$ is derived using the number $\binom{p+i-1}{i}$ of combinations of the scalar factors β with replacement (i.e. the same factor can be used again). Adding the numbers for all polynomial orders and adding the number of independent generators q, results in $\xi = \sum_{i=1}^{\eta} \binom{p+i-1}{i} + q$ generators, which is in $\mathcal{O}(p^\eta)$ with respect to p. The non-convex shape of a polynomial zonotope with polynomial

order 2 is shown in Fig. 11 for

$$E^{[1]} = \begin{bmatrix} -1 & 1 \\ 0.5 & 1.5 \end{bmatrix}, \quad E^{[2]} = \begin{bmatrix} 1 & 0.6 \\ -1.5 & 0.1 \end{bmatrix}, \quad F^{[2]} = \begin{bmatrix} -1.5 \\ 1.1 \end{bmatrix}, \quad G = \begin{bmatrix} 0.4 \\ 0.4 \end{bmatrix}$$

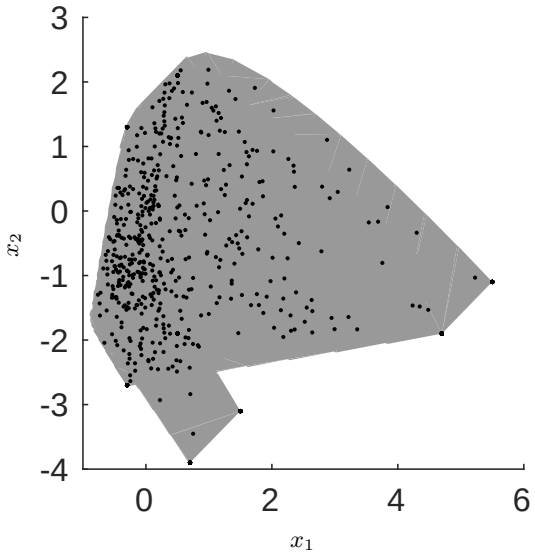


Figure 11: Over-approximative plot of a polynomial zonotope. Random samples of possible values demonstrate the accuracy of the over-approximative plot.

A zonotope $\mathcal{Z}$ is a special case of a polynomial zonotope that has only generators $g^{(i)}$, which is denoted by $\mathcal{Z} = (c, G)$. Due to the absence of $E^{[i]}, F^{[i]}$ a zonotope is centrally symmetric to c so that for zonotopes, c is referred to as the *center* and not the *starting point*.

Although a Taylor model [35] is not a set, but a multidimensional polynomial plus a multidimensional interval, they can represent exactly the same sets than polynomial zonotopes when the input of the Taylor models is a multidimensional interval. The different organization of polynomial zonotopes, separating dependent from independent variables, makes it easier to over-approximate them by zonotopes or perform the order reduction techniques presented subsequently.

5.3 Operations on Polynomial Zonotopes

It is often required to over-approximate a polynomial zonotope by a zonotope:

Proposition 5.1 (Over-approximation by a Zonotope) *A polynomial zonotope* $\mathcal{PZ} = (c, E, F, G)$ *can be over-approximated by a zonotope* $\mathcal{Z} = \texttt{zonotope}(\mathcal{PZ}) = (\tilde{c}, \tilde{G})$ *so that*

$\mathcal{PZ} \subseteq \mathcal{Z}$, by choosing

$$\tilde{c} = c + \frac{1}{2}\sum_{i=1}^{\lfloor \eta/2 \rfloor}\sum_{j=1}^{p} e^{([2i],j)},$$
$$\tilde{G} = \left[\tfrac{1}{2}E^{[2]} \quad \tfrac{1}{2}E^{[4]} \dots \quad E^{[1]} \quad E^{[3]} \dots \quad F^{[1]} \quad F^{[2]} \dots \quad G\right],$$

where $\lfloor \eta/2 \rfloor$ returns the lowest integer of $\eta/2$. The computational complexity for a given polynomial zonotope order ρ with respect to n is $\mathcal{O}(n^2)$.

A sketch of the proof is as follows: Generators with dependent factors are made independent by moving them into the generator matrix G, which always results in an over-approximation. Dependent factors β_i with even powers are within $[0, 1]$ (e.g. $\beta_1^2 \in [0, 1]$) instead of $[-1, 1]$ so that $E^{[2]}$, $E^{[4]}$, ... can be multiplied by 0.5 and their mean is added to c.

The multiplication of a matrix $M \in \mathbb{R}^{o \times n}$ with a polynomial zonotope $\mathcal{PZ} = (c, E, F, G)$ and the Minkowski addition of a zonotope $\mathcal{Z} = (\tilde{c}, \tilde{G})$ with $\mathcal{PZ}$ follow directly from the definition of polynomial zonotopes:

$$\begin{aligned} M \otimes \mathcal{PZ} &= (Mc, ME, MF, MG), \\ \mathcal{PZ} \oplus \mathcal{Z} &= (c + \tilde{c}, E, F, [G,\ \tilde{G}]). \end{aligned} \tag{28}$$

For a given polynomial zonotope order ρ, the computational complexity with respect to n is $\mathcal{O}(n^3)$ for the multiplication and $\mathcal{O}(n)$ for the addition. Note that the Minkowski addition of two polynomial zonotopes is never required since $\mathcal{R}^{\Delta}$ and $\mathcal{R}^{p,\Delta}$ are represented by zonotopes. The only addition between two polynomial zonotopes is between $\mathcal{PZ}_1 = (c_1, E_1, F_1, G_1)$ and $\mathcal{PZ}_2 = (c_2, E_2, F_2, G_2)$ in (26). Since both summands have the same dependent factors (proof omitted), one can apply an exact set addition, where the resulting polynomial zonotope is $(c_1 + c_2, E_1 + E_2, F_1 + F_2, [G_1, G_2])$. The generators with independent factors in G_1 and G_2 are added by concatenation as for the Minkowski addition in (28).

5.4 Order Reduction of Polynomial Zonotopes

Many of the previously presented operations increase the order of polynomial zonotopes due to added generators. As a consequence, an order reduction technique has to be applied to limit the representation size and the computational costs. Most techniques for classical zonotopes remove generators and add new, but fewer ones that capture the spanned set of the removed generators (see e.g. [36]). This results in a reordering of the generators, which is no problem for zonotopes since the ordering of generators is irrelevant. However, generators of polynomial zonotopes can only be reordered within G, where generators are multiplied by

independent factors γ_i. For this reason, a new order reduction technique is developed that does not change the ordering of generators in E and F.

The size of E and F is fixed and only G grows after performing the required operations presented in this work. Thus, it is required to remove generators from G and stretch the generators in E and F such that the ones removed from G are compensated in an over-approximative way. As for most applied order reduction techniques of zonotopes, heuristics are used rather than strict optimization techniques due to their favorable ratio of computational costs to obtained over-approximation.

Proposition 5.2 (Over-approximative Generator Removal)
Given is $\mathcal{PZ} = (c, E, F, G)$ of which n linearly independent generators with indices $\mathrm{ind}_1, \dots \mathrm{ind}_n$ are picked from $E^{[1]}$ and stored in $P = [e^{([1],\mathrm{ind}_1)} \dots e^{([1],\mathrm{ind}_n)}]$ ($\det(P) \neq 0$). The over-approximating polynomial zonotope $\widehat{\mathcal{PZ}} = (c, \hat{E}, F, \hat{G})$ from which the generator $g^{(i)}$ is removed, is computed as

$$\hat{G} = \Big[g^{(1)} \dots g^{(i-1)}, g^{(i+1)}, \dots g^{(q)}\Big],$$
$$\hat{E} = \Big[\hat{E}^{[1]}\, E^{[2]}\, \dots\, E^{[\eta]}\Big],$$
$$\hat{e}^{[1],j} = \begin{cases} (1 + (P^{-1}g^{(i)})_j)e^{([1],j)} \text{ for } j \in \{\mathrm{ind}_1, \dots, \mathrm{ind}_n\}, \\ e^{([1],j)} \text{ otherwise.} \end{cases}$$

The computational complexity is $\mathcal{O}(n^3)$ due to the matrix inversion when using the Gauss-Jordan elimination.

Proof 6 *The generator $g^{(i)}$ can be composed from the generators in P:*

$$g^{(i)} = e^{([1],\mathrm{ind}_1)}\phi_1 + \dots + e^{([1],\mathrm{ind}_n)}\phi_n = P\phi \text{ so that } \phi = P^{-1}g^{(i)}.$$

Note that P^{-1} can always be computed since $\det(P) \neq 0$. Thus, $g^{(i)}$ can be replaced by n new generators $e^{([1],\mathrm{ind}_1)}\phi_1$, …, $e^{([1],\mathrm{ind}_n)}\phi_n$, which causes an over-approximation since each generator has an independent factor γ_{q+j}:

$$\Big\{\gamma_i g^{(i)} \Big| \gamma_i \in [-1, 1]\Big\} \subseteq \Big\{\sum_{j=1}^{n} \gamma_{q+j} e^{([1],\mathrm{ind}_j)}\phi_j \Big| \gamma_{q+j} \in [-1, 1]\Big\}$$

The n new generators are aligned with the corresponding generators in $E^{[1]}$ and can be removed by stretching each $e^{([1],\mathrm{ind}_j)}$ by the factor

$$\frac{\|e^{([1],\mathrm{ind}_j)}\|_2 + \|e^{([1],\mathrm{ind}_j)}\phi_j\|_2}{\|e^{([1],\mathrm{ind}_j)}\|_2} = 1 + \phi_j = 1 + (P^{-1}g^{(i)})_j.$$

We remove the longest generators $g^{(i)}$ (maximum 2-norm) from G so that the generators in E are more dominant than the ones in G, which is required since only the generators in E are used for the exact computation of quadratic maps. The heuristic for choosing the set of picked generators P in Prop. 5.2 is as follows: The first generator is the one in $E^{[1]}$ that is best aligned with the removed generator $g^{(i)}$, and the other $n-1$ generators are the ones which have the least alignment with $g^{(i)}$ and among each other. The alignment is measured by the normalized scalar product $\frac{|g^{(i)^T} e^{([1],\mathrm{ind}_i)}|}{\|g^{(i)}\|_2 \|e^{([1],\mathrm{ind}_i)}\|_2}$, where a value of 1 occurs when the vectors are aligned and 0 for perpendicular vectors. The generators $g^{(i)}$ are removed until the order is less than a user-defined order. Without giving the proof, the complexity of the order reduction heuristic is $\mathcal{O}(n^3)$.

6 Compositional Verification

In order to tackle the curse of dimensionality in the verification of cyber-physical systems, it is unavoidable to use a divide-and-conquer approach that makes it possible to verify a large system in a modular way by verifying smaller subsystems. In assume-guarantee reasoning (see Fig. 12), assumptions on system inputs are made, which are used to verify the specifications of subsystems. The partial verification results are utilized to check the correctness of the assumptions. For instance, we can assume a set of possible inputs $\mathcal{U}_1$ and $\mathcal{U}_2$ of subsystems M_1 and M_2 as shown in Fig. 12. Using reachability analysis, we can determine the possible set of outputs $\mathcal{Y}_1$ and $\mathcal{Y}_2$. If those output sets are enclosed by the corresponding input sets, the reachable set of the entire system is over-approximated. The problem with assume-guarantee reasoning is that the verification result quickly becomes too conservative since dependencies between subsystems are not considered. We address this problem by considering hierarchies in the system modeling and in the computation of reachable sets. As initial solutions, the verification of nonlinear systems is performed by (1) dividing the original system into subsystems, and (2) only dividing the system into subsystems for the abstraction error computation. A new investigation regarding the robustness to varying selections of possible partitions is also provided.

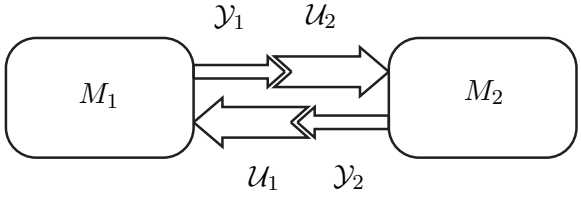


Figure 12: Assume-Guarantee Reasoning: The input sets enclose the output sets ($\mathcal{Y}_1 \subseteq \mathcal{U}_2, \mathcal{Y}_2 \subseteq \mathcal{U}_1$).

6.1 Compositional Reachable Set Computation

This subsection describes the first option investigated for compositional analysis by splitting the system into subsystems. The reachable set of each subsystem is computed as previously presented. The input sets representing inputs to the complete system are known, however, the possible inputs from the interfaces of the subsystems are unknown. They depend on the reachable set of neighboring subsystems (i.e., subsystems where an output of one subsystem is an input to the other one, or the other way round), which in turn depend on the reachable sets of other subsystems. This mutual dependence is broken apart by assuming a set obtained from enlarging the inputs in between subsystems, which we refer to as interface inputs, by a factor λ_U, where i refers to the i^{th} subsystem:

$$\hat{\mathcal{U}}^{(i)} = \hat{c}_U^{(i)} \oplus ((\lambda_U - 1)\Lambda^{(i)} + I)(\mathcal{U}^{(i)} \oplus (-\hat{c}_U^{(i)})), \tag{29}$$

where Λ is a diagonal matrix that contains ones for indices corresponding to interface inputs and zeros otherwise. Based on the system input $\hat{\mathcal{U}}^{(i)}$, the reachable set of the corresponding subsystem is computed as previously presented in Sec. 3. For aggregating the reachable sets of the complete system by partial reachable sets $\mathcal{R}^{(i)}(\tau_s)$ of the i^{th} subsystem, matrices $\Phi^{(i)}$ are introduced, which map the local states of the i^{th} subsystem to the states of the full system. The matrices $\Phi^{(i)}$ contain ones when states are correlated, and zeros otherwise, so that the complete reachable set is obtained using the Cartesian product:

$$\mathcal{R}(\tau_s) = \Phi^{(1)}\mathcal{R}^{(1)}(\tau_s) \times \Phi^{(2)}\mathcal{R}^{(2)}(\tau_s) \times \ldots \times \Phi^{(n_s)}\mathcal{R}^{(n_s)}(\tau_s), \tag{30}$$

where n_s is the number of subsystems and the computation of linear maps and the Cartesian products of zonotopes is performed as presented in Sec. 4.1. In order to check if the assumption on the set of interface inputs is correct for all subsystems, further matrices $\Upsilon^{(i)}$ are introduced, which map the states of the complete system to the interface inputs of the i^{th} subsystem. Again, the matrix contains ones for corresponding states and interface inputs and zeros otherwise. If

$$\forall i : \Upsilon^{(i)}\mathcal{R}(\tau_s) \subseteq \Lambda^{(i)}\hat{\mathcal{U}}^{(i)}$$

the assumption is over-approximative and thus valid. Otherwise, one has to re-apply the enlargement in (29) for the subsystems that violate the assumption. This procedure is summarized in Alg. 1 under the assumption that no split of the reachable set is required.

Algorithm 1 `reachNextCompositional`($\mathcal{R}^{(i),d}(t_s), \lambda_U, \mathcal{U}^{(i)}$, `otherInputsToReachNext`)

Require: Previous set $\mathcal{R}^{(i),d}(t_s)$ and input set $\mathcal{U}^{(i)}$ for each subsystem, factor λ_U, `otherInputsToReachNext`).

Ensure: $\mathcal{R}^{(i),d}(t_{k+1}), \mathcal{R}(\tau_s)$

$\forall i : inputEnclosure^{(i)} =$ `false`
repeat
  for $i = 1 \dots n_s$ **do**
    if $inputEnclosure^{(i)} ==$ `false` **then**
      obtain $\hat{\mathcal{U}}^{(i)}$ based on $\mathcal{U}^{(i)}$ using (29)
      Standard reachability computation based on $\hat{\mathcal{U}}^{(i)} \rightarrow \mathcal{R}^{(i),d}(t_{k+1}), \mathcal{R}^{(i)}(\tau_s)$
    end if
  end for
  $\mathcal{R}(\tau_s) = \Phi^{(1)}\mathcal{R}^{(1)}(\tau_s) \times \ldots \times \Phi^{(n_s)}\mathcal{R}^{(n_s)}(\tau_s)$
  for $i = 1 \dots n_s$ **do**
    if $\Upsilon^{(i)}\mathcal{R}(\tau_s) \subseteq \Lambda^{(i)}\hat{\mathcal{U}}^{(i)}$ **then**
      $inputEnclosure^{(i)} =$ `true`
    else
      $inputEnclosure^{(i)} =$ `false`
    end if
    $\mathcal{U}^{(i)} = \Lambda^{(i)}\hat{\mathcal{U}}^{(i)} \oplus (I - \Lambda^{(i)})\mathcal{U}^{(i)}$
  end for
until $\forall i : inputEnclosure^{(i)} ==$ `true`

6.2 Compositional Linearization Error Computation

In some systems, the dynamics of subsystems might be strongly correlated, resulting in unsatisfactory over-approximations of the compositional algorithm in Alg. 1. Since most of the computation time is spent on evaluating the linearization error, one could only compute the linearization error compositionally, while maintaining all the correlations for the reachable set computation as presented in Sec. 3.

Using a decomposition of the full system into subsystems as in the previous subsection, the Lagrangian remainder in Sec. 4.2 is evaluated compositionally. For this purpose, the set of input values from subsystem interfaces has to be considered resulting in the set of inputs for each subsystem as proposed in (29). The partial Lagrange remainders of the i^{th} subsystem denoted by $\mathcal{L}^{(i)}$ are combined to the complete Lagrange remainder as for the reachable set in (30):

$$\mathcal{L}(\tau_s) = \Phi^{(1)}\mathcal{L}^{(1)}(\tau_s) \times \Phi^{(2)}\mathcal{L}^{(2)}(\tau_s) \times \ldots \times \Phi^{(n_s)}\mathcal{L}^{(n_s)}(\tau_s).$$

The computation of the Lagrange remainder has complexity $\mathcal{O}(n^5)$, where n is the number of state variables, whereas all other operations have complexity $\mathcal{O}(n^3)$ when using zonotopes as the set representation. Thus, the compositional computation of the linearization error has similar computational savings than the completely compositional computation as presented in the previous subsection.

6.3 Application to a Smart Grid Use Case

The introduced methods are applied to the transient stability analysis of a smart grid use case. Other than the power plant example in Sec. 4.5, we consider the interconnection of several power plants through the electric grid. From now on we refer to a power plant as a generator and a consumer as a power demand.

6.3.1 Power System Modeling

The mathematical models used for the case studies are standard models. However, those models are huge and tedious to implement manually. For this reason, we have integrated an automatic method that generates the required differential equations and constraints in CORA. The dynamic variables of the i^{th} generator are the generator phase angle $\tilde{\delta}_i$ [rad], the angular velocity ω_i [rad/s], and the torque $T_{m,i}$ [p.u.] (p.u.: per unit). The commanded power production $P_{c,i}$ [p.u.] is a system input. The generator phase angles $\delta_i = \tilde{\delta}_i - \Theta_s$ are chosen relative to the slack bus angle Θ_s which has a constant angular velocity ω_s. The

dynamic equations of the chosen generator model are according to [37]:

$$\begin{aligned}\dot{\delta}_i &= \omega_i - \omega_s \\ \dot{\omega}_i &= -\frac{D_i}{M_i}(\omega_i - \omega_s) + \frac{1}{M_i}T_{m,i} - \frac{1}{M_i}P_{g,i} \\ \dot{T}_{m,i} &= -\frac{1}{T_{SV,i}R_{D,i}\omega_s}(\omega_i - \omega_s) - \frac{1}{T_{SV,i}}T_{m,i} + \frac{1}{T_{SV,i}}P_{c,i}.\end{aligned} \tag{31}$$

The parameter values are chosen identical to [37] for each generator and synchronous condensers and are listed in Tab. 2.

The power flow equations are obtained using standard methods, see e.g. [38, p.174]. The algebraic variables of the i^{th} bus are the absolute value of the bus voltage V_i [p.u], the phase angle of the bus voltage $\tilde{\Theta}_i$ [rad], the active power P_i [p.u.], the reactive power Q_i [p.u.], and the generator voltage E_i [p.u.] if the bus is connected to a generator. The bus phase angles with respect to the slack bus are denoted by $\Theta_i = \tilde{\Theta}_i - \Theta_s$. The buses are connected via admittances $Y_{ij} = Y_{ji}$, where i and j are the indices of the connected buses. The admittance from the generator to the i^{th} generator bus is $Y_{g,i}$, where $|Y_{g,i}|$ [p.u.], $\Psi_{g,i} = \angle Y_{g,i}$ [rad] are the absolute values and phase angles, respectively. The absolute value and the angle of the admittances are denoted by $|Y_{ij}|$ and $\Psi_{ij} = \angle Y_{ij}$, respectively. The active and reactive power of each bus results from the generator production $P_{g,i}, Q_{g,i}$ and a demand of that node $P_{d,i}, Q_{d,i}$. The parameters of the power grid are chosen according to the corresponding IEEE benchmark problem and can be found in [39].

In order to obtain the constraints on the continuous variables, we classify nodes in the power network as *slack bus* (reference bus), *generator bus* (bus to which a generator is attached), and *load bus* (bus to which typically a consumer is attached). We introduce N_g as the number of generators and N_l as the number of load buses. Since we are dealing with an alternating current network, buses produce active and reactive power:

$$\begin{aligned}P_{g,i} &= E_i V_i |Y_{g,i}| \cos(\Psi_{g,i} + \delta_i - \Theta_i) - V_i^2 |Y_{g,i}| \cos(\Psi_{g,i}), \\ Q_{g,i} &= -E_i V_i |Y_{g,i}| \sin(\Psi_{g,i} + \delta_i - \Theta_i) + V_i^2 |Y_{g,i}| \sin(\Psi_{g,i}).\end{aligned}$$

The power flow equations as in [38, p.174] of each bus are

$$\begin{aligned}P_i = P_{g,i} + P_{g,i}^d + P_{d,i} &= \sum_{j=1}^{N_g+N_l} V_i V_j |Y_{ij}| \cos(\Psi_{ij} + \Theta_j - \Theta_i), \\ Q_i = Q_{g,i} + Q_{g,i}^d + Q_{d,i} &= -\sum_{j=1}^{N_g+N_l} V_i V_j |Y_{ij}| \sin(\Psi_{ij} + \Theta_j - \Theta_i),\end{aligned} \tag{32}$$

where $P_{g,i}$ and $Q_{g,i}$ are the active and reactive power produced by generators with the dynamics according to (31), while $P_{g,i}^d$ and $Q_{g,i}^d$ are directly injected active and reactive powers

from renewable energy sources. In order to write the power system in the standard form of time-invariant, semi-explicit, index-1 differential-algebraic equations, the dynamic, algebraic, and input variables are renamed. Additionally, the number of cut transmission lines N_i for the considered subsystem is introduced. It is also required to consider variables of neighboring subsystems. The j^{th} voltage of the k^{th} subsystem is denoted by $\hat{V}_{k,j}$ and an analogous notation is used for $\hat{\Theta}_{k,j}$. The function $[k, j] = h(i)$ returns the subsystem number k of which the bus with number j is connected to the considered subsystem and i takes integers up to the number of cut transmission lines ($i = 1 \dots N_i$), thus providing the first, second, and further input sources. The algebraic variables are assigned as follows:

$$\begin{aligned} i = 1 \dots N_g : &\quad y_i = E_i, \\ i = 1 \dots N_l : &\quad y_{N_g+i} = V_{N_g+i}, \\ i = 2 \dots (N_g + N_l) : &\quad y_{N_g+N_l+i-1} = \Theta_i. \end{aligned}$$

Note that Θ_1 is not considered in the above assignment since it is the phase of the slack bus and thus always 0. The dynamic variables are

$$\begin{aligned} i = 1 \dots N_g : &\quad x_i = \delta_i, \\ i = 1 \dots N_g : &\quad x_{N_g+i} = \omega_i, \\ i = 1 \dots N_g : &\quad x_{2N_g+i} = T_{m,i}, \end{aligned}$$

and the inputs are assigned as follows:

$$\begin{aligned} i = 1 \dots N_g : &\quad u_i = P_{c,i}, \\ i = 1 \dots (N_g + N_l) : &\quad u_{N_g+i} = P^d_{g,i}, \\ i = 1 \dots (N_g + N_l) : &\quad u_{2N_g+N_l+i} = Q^d_{g,i}, \\ i = 1 \dots N_i, [k, j] = h(i) : &\quad u_{3N_g+2N_l+i} = \hat{V}_{k,j}, \\ i = 1 \dots N_i, [k, j] = h(i) : &\quad u_{3N_g+2N_l+N_i+i} = \hat{\Theta}_{k,j}. \end{aligned}$$

Table 2: Parameters of the generators.

$\forall i$:	M_i	D_i	$\lvert Y_{g,i}\rvert$	$\Psi_{g,i}$	$T_{SV,i}$	$R_{D,i}$	ω_s
	$\frac{1}{15\pi}$	0.04	5	$-\frac{\pi}{2}$	1	0.05	120π

6.3.2 Scenario

The transient stability analysis is performed as follows. After a pre-fault phase of 0.1 s, the power plant producing the most power is taken off the grid (e.g. caused by a short circuit)

for 0.03 s and afterwards reconnected. In the post-fault phase, the dynamics is computed until all continuous state variables reach the set of initial states. In all case studies, the center of the initial set is the steady state solution denoted by a superscripted zero. For all power generators the initial phase is $\delta_i(0) \in \delta_i^0 \oplus 0.005 \cdot [-1, 1]$, the initial rotational speed is $\omega_i(0) \in \omega_i^0 \oplus 0.1 \cdot [-1, 1]$, and the initial torque is $T_{m,i}(0) \in T_{m,i}^0 \oplus 0.001 \cdot [-1, 1]$.

The case study is based on the IEEE 14-bus benchmark system, enhanced by the generator dynamics as introduced in Sec. 6.3.1. We compute the linearization error compositionally as described in Sec. 6. For that purpose, the power system is split into different subsystems as described next in Sec. 6.3.3.

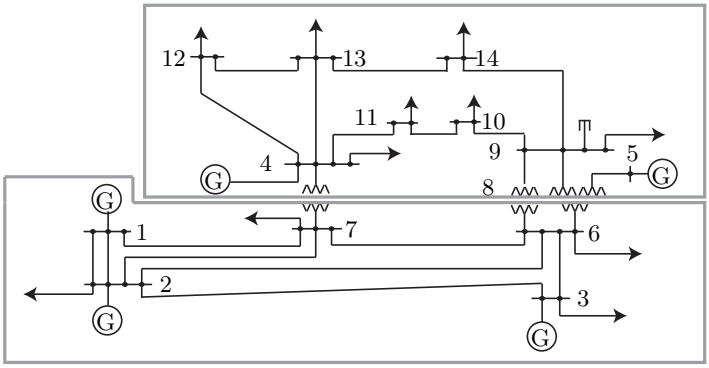
6.3.3 Sensitivity Analysis of Partitioning

In this subsection we investigate whether the proposed approach is sensitive with respect to the partitioning into subsystems. We consider three partitions as illustrated in Fig. 13. It is expected that the sensitivity is manageable since correlations between states are preserved using our technique. This is demonstrated by comparing the reachable sets for selected projections of different partitions in Fig. 14. It can be observed that the result is almost identical although the partitions are quite different. The accuracy of the reachable sets in Fig. 14 is indicated by simulations of system trajectories from randomly chosen initial states, which are plotted as black lines. Note that the results for the algebraic variables jump after the pre-fault and fault-on phase since the system model switches.

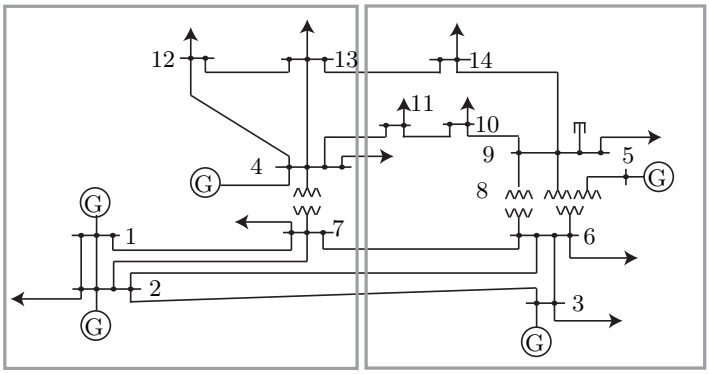
Computation times for different partitions are listed in Tab. 3 when the linearization errors of subsystems are computed in parallel using different cores. All computations are performed in MATLAB on an Intel i7-3520M processor with 2.9 Ghz. Partition 1 and 2 in Fig. 13 result in similar computation times since the size of each subsystem is comparable and thus, the computational load is well balanced between each processor. Since partition 3 is unbalanced, the resulting computational time is much closer to the one of the full system. Although the over-approximation of the compositional computation of the linearization error is small, the savings in computation time are significant. This is because the linearization error computation consumes around 90% of the overall computation time.

Table 3: Computation times for different partitions.

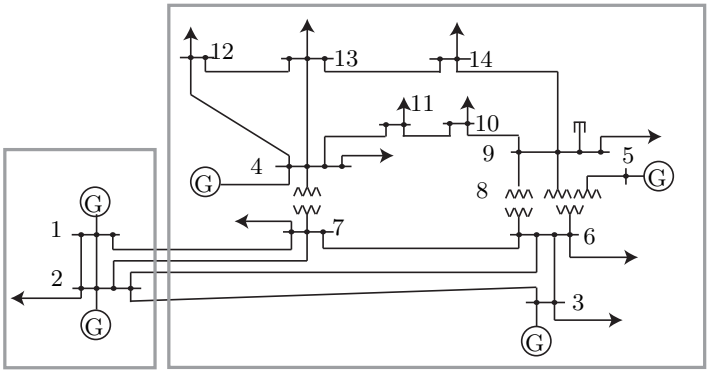
partition:	full	1	2	3
computation time in [s]	1634	797.4	769.8	1038



(a) Partition 1.



(b) Partition 2.

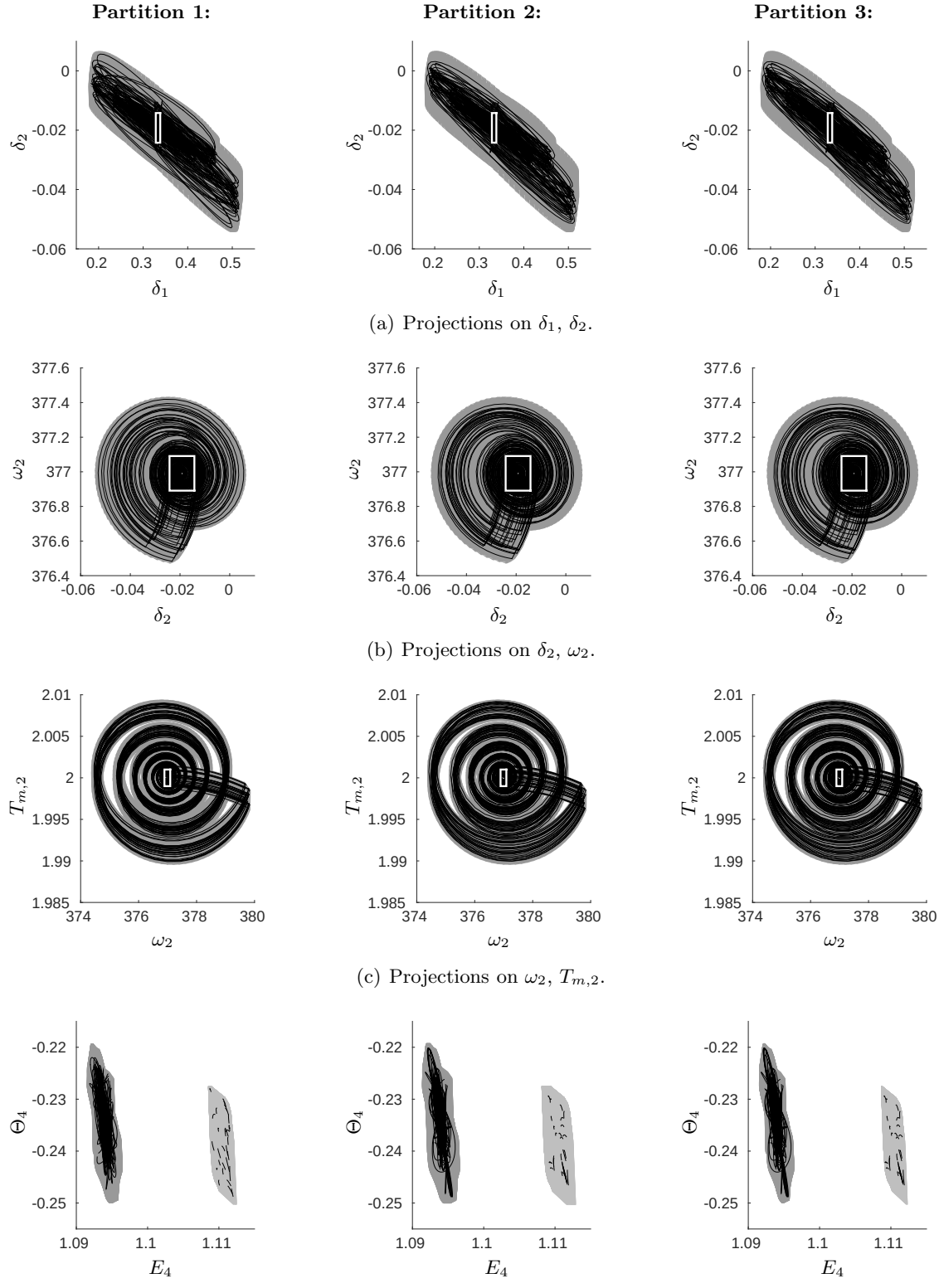


(c) Partition 3.

Figure 13: Considered partitions of the IEEE 14-bus benchmark system. Gray lines show subsystem borders.

7 Conclusion

This deliverable summarizes new abstraction techniques for the reachability analysis of nonlinear systems and their compositional verification. For nonlinear systems, two abstractions are presented. First, on-the-fly linearization is presented, which abstracts arbitrary nonlinear systems to linear ones on-the-fly by adapting the linear system parameters based on the current center of the reachable set. The uncertain input is over-approximative to guarantee the



(a) Projections on δ_1, δ_2.

(b) Projections on δ_2, ω_2.

(c) Projections on ω_2, $T_{m,2}$.

(d) Projections on E_4, Θ_4.

Figure 14: Selected projections of reachable sets for transient stability analysis. Partitions 1-3 are shown from left to right. Black lines show random simulations, gray areas show reachable sets, and the white box the initial set. For algebraic variables, dark gray represents pre-fault and post-fault sets, light gray represents fault-on sets, and medium gray represents sets for all fault phases.

conservativeness of the approach. We present a novel cubic evaluation of the linearization error that is more accurate compared to a quadratic one. The effectiveness of linear system abstractions is demonstrated by a smart grid example, which uses data from an actual power plant in Munich. This made it possible to formally verify the correctness of a steam generation process for the very first time.

Second, on-the-fly abstraction to polynomial systems is presented, which is more accurate compared to linear abstractions. The disadvantage, however, is that sets are no longer propagated by linear maps, such that the convexity property of reachable sets is lost. To this end, we apply so-called polynomial zonotopes, which can represent non-convex sets. Polynomial zonotopes have been integrated into CORA for further use in UnCoVerCPS.

Finally, we show a novel concept for the compositional verification of nonlinear systems. In order to preserve the dependencies between variables, we only compose the problem of computing abstraction errors into smaller subproblems. As a result, we obtain significant savings in computational time as shown in Tab. 3 since the abstraction error computation is the most time-consuming task. However, the accuracy is almost as good as for a monolithic verification since the linear part of the dynamics is not partitioned, see Fig. 14. The robustness of this approach is demonstrated by a new study on smart grids that shows that the results are insensitive with respect to the partitioning into subproblems for the abstraction error computation.

The results of this deliverable will be used in all tasks related to constraint generation for controller synthesis and on-the-fly verification of our use cases.

References

[1] R. Alur and D. L. Dill, "A theory of timed automata," *Theoretical Computer Science*, vol. 126, pp. 183–235, 1994.

[2] T. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya, "What's decidable about hybrid automata?" *Journal of Computer and System Sciences*, vol. 57, pp. 94–124, 1998.

[3] G. Lafferriere, G. J. Pappas, and S. Yovine, "Symbolic reachability computation for families of linear vector fields," *Symbolic Computation*, vol. 32, pp. 231–253, 2001.

[4] G. Frehse, C. L. Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler, "SpaceEx: Scalable verification of hybrid systems," in *Proc. of the 23rd International Conference on Computer Aided Verification*, ser. LNCS 6806. Springer, 2011, pp. 379–395.

[5] M. Althoff, "An introduction to CORA 2015," in *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems*, 2015, pp. 120–151.

[6] A. Platzer, *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Springer, 2010.

[7] S. Ratschan and Z. She, "Constraints for continuous reachability in the verification of hybrid systems," in *Proc. of Artificial Intelligence and Symbolic Computation*, ser. LNCS 4120. Springer, 2006, pp. 196–210.

[8] S. Prajna, "Barrier certificates for nonlinear model validation," *Automatica*, vol. 42, no. 1, pp. 117–126, 2006.

[9] E. Asarin, T. Dang, G. Frehse, A. Girard, C. Le Guernic, and O. Maler, "Recent progress in continuous and hybrid reachability analysis," in *Proc. of the 2006 IEEE Conference on Computer Aided Control Systems Design*, 2006, pp. 1582–1587.

[10] N. Aréchiga, S. M. Loos, A. Platzer, and B. H. Krogh, "Using theorem provers to guarantee closed-loop system properties," in *In Proc. of the American Control Conference*, 2012, pp. 3573–3580.

[11] A. Girard and G. J. Pappas, "Verification using simulation," in *Hybrid Systems: Computation and Control*, ser. LNCS 3927. Springer, 2006, pp. 272–286.

[12] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependent Hamilton–Jacobi formulation of reachable sets for continuous dynamic games," *IEEE Transactions on Automatic Control*, vol. 50, pp. 947–957, 2005.

[13] A. Chutinan and B. H. Krogh, "Computational techniques for hybrid system verification," *IEEE Transactions on Automatic Control*, vol. 48, no. 1, pp. 64–75, 2003.

[14] A. Girard, C. Le Guernic, and O. Maler, "Efficient computation of reachable sets of linear time-invariant systems with inputs," in *Hybrid Systems: Computation and Control*, ser. LNCS 3927. Springer, 2006, pp. 257–271.

[15] A. B. Kurzhanski and P. Varaiya, "Ellipsoidal techniques for reachability analysis," in *Hybrid Systems: Computation and Control*, ser. LNCS 1790. Springer, 2000, pp. 202–214.

[16] A. Girard and C. Le Guernic, "Efficient reachability analysis for linear systems using support functions," in *Proc. of the 17th IFAC World Congress*, 2008, pp. 8966–8971.

[17] O. Stursberg and B. H. Krogh, "Efficient representation and computation of reachable sets for hybrid systems," in *Hybrid Systems: Computation and Control*, ser. LNCS 2623. Springer, 2003, pp. 482–497.

[18] M. Althoff and B. H. Krogh, "Reachability analysis of nonlinear differential-algebraic systems," *IEEE Transactions on Automatic Control*, vol. 59, no. 2, pp. 371–383, 2014.

[19] M. Althoff, O. Stursberg, and M. Buss, "Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization," in *Proc. of the 47th IEEE Conference on Decision and Control*, 2008, pp. 4042–4048.

[20] T. Dang, O. Maler, and R. Testylier, "Accurate hybridization of nonlinear systems," in *Hybrid Systems: Computation and Control*, 2010, pp. 11–19.

[21] M. Althoff, "Reachability analysis of nonlinear systems using conservative polynomialization and non-convex sets," in *Hybrid Systems: Computation and Control*, 2013, pp. 173–182.

[22] M. Berz and G. Hoffstätter, "Computation and application of Taylor polynomials with interval remainder bounds," *Reliable Computing*, vol. 4, pp. 83–97, 1998.

[23] W. Kühn, *Mathematical Visualization*. Springer, 1998, ch. Zonotope Dynamics in Numerical Quality Control, pp. 125–134.

[24] M. Althoff, "Reachability analysis and its application to the safety assessment of autonomous cars," Dissertation, Technische Universität München, 2010, http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:bvb:91-diss-20100715-963752-1-4.

[25] L. Jaulin, M. Kieffer, and O. Didrit, *Applied Interval Analysis*. Springer, 2006.

[26] J.-R. Abrial, "Steam-boiler control specification problem," in *Formal Methods for Industrial Applications*. Springer, 1996, pp. 500–509.

[27] T. A. Henzinger and H. Wong-Toi, *Using HyTech to synthesize control parameters for a steam boiler*. Springer, 1996.

[28] H. Wong-Toi, "The synthesis of controllers for linear hybrid automata," in *Proceedings of the 36th IEEE Conference on Decision and Control*, vol. 5, 1997, pp. 4607–4612.

[29] J. Lygeros, C. Tomlin, and S. Sastry, "Multiobjective hybrid controller synthesis," in *Hybrid and Real-Time Systems*. Springer, 1997, pp. 109–123.

[30] A. El-Guindy, S. Runzi, and K. Michels, "Optimizing drum-boiler water level control performance: A practical approach," in *Proc. of the IEEE Conference on Control Applications*, 2014, pp. 1675–1680.

[31] A. El-Guindy, F. Nickel, and K. Michels, "Centralised multivariable feedback control of steam drums in combined cycle power plants," *VGB PowerTech*, vol. 4, pp. 73–78, 2015.

[32] K. J. Åström and R. D. Bell, "Drum boiler dynamics," *Automatica*, vol. 36, pp. 363–378, 2000.

[33] M. Althoff, C. Le Guernic, and B. H. Krogh, "Reachable set computation for uncertain time-varying linear systems," in *Hybrid Systems: Computation and Control*, 2011, pp. 93–102.

[34] E. Asarin, T. Dang, and A. Girard, "Hybridization methods for the analysis of nonlinear systems," *Acta Informatica*, vol. 43, pp. 451–476, 2007.

[35] J. Hoefkens, M. Berz, and K. Makino, *Scientific Computing, Validated Numerics, Interval Methods*. Springer, 2001, ch. Verified High-Order Integration of DAEs and Higher-Order ODEs, pp. 281–292.

[36] A. Girard, "Reachability of uncertain linear systems using zonotopes," in *Hybrid Systems: Computation and Control*, ser. LNCS 3414. Springer, 2005, pp. 291–305.

[37] Y. C. Chen and A. D. Domínguez-García, "Assessing the impact of wind variability on power system small-signal reachability," in *Proc. of the International Conference on System Sciences*, 2011, pp. 1–8.

[38] P. Schavemaker and L. van der Sluis, *Electrical Power System Essentials*. Wiley, 2008.

[39] University of Washington, "Power systems test case archive," http://www.ee.washington.edu/research/pstca/, University of Washington.