Trajectory-Directed Discrete State Space Modeling for Formal Verification of Nonlinear Analog Circuits

Sebastian Steinhorst
TUM CREATE
Singapore
sebastian.steinhorst@tum-create.edu.sg

Lars Hedrich
University of Frankfurt/Main
Germany
hedrich@em.cs.uni-frankfurt.de

ABSTRACT

In this paper a novel approach to discrete state space modeling of nonlinear analog circuits is presented, based on the introduction of an underlying discrete analog transition structure (DATS) and the related optimization problem of accurately representing a nonlinear analog circuit with a DATS. Starting from a circuit netlist, a partitioning of the state space to the discrete model is generated parallel and orthogonal to the trajectories of the state space dynamics. Therefore, compared to previous approaches, a significantly higher accuracy of the model is achieved with a lower number of states. The mapping of the partitioning to a DATS enables the application of formal verification algorithms. Experimental validations show the soundness of the approach with an increase in accuracy between a factor of 4 to 10 compared to the state of the art. A model checking case study illustrates the application of the new discretization algorithm to identify a hidden circuit design error.

1. INTRODUCTION

Formal verification approaches, such as model and equivalence checking, have led to a significant increase in design productivity and automation of digital systems. However, in today's mixed-signal embedded systems, analog components can not be designed in an equally efficient manner. Especially in the area of verification, more analog design automation is urgently needed. Analog verification is relying on the expertise and experience of the verification engineer and is mostly performed by analyzing simulation waveforms either manually or using some scripted calculations. While assertion-based verification can be considered as a step into the right direction, formal approaches that consider the circuit's behavior for all possible input signals and all possible internal states are mandatory for closing the analog verification gap.

The value and time continuous characteristics of analog circuits require a special representation for application of formal methods. While in the digital circuit domain finite state machine (FSM) models are a common approach for formal system representation, analog circuits are not easily transferable to such a discrete model. Digital systems have an enumerable finite number of states which can be directly transferred into a FSM representation. Formal verification algorithms can check this set of states completely in order to identify states that violate the specification or to extract globally valid characteristics of the circuit's behavior.

By contrast, the continuous state space of analog systems is not enumerable. Hence, for completely checking their behavior, either an analytical approach reasoning on the continuous model or a discretization to a finite state model is required at the cost of introducing a discretization error. Due to the difficulties in obtaining a solution of nonlinear differential-algebraic equation (DAE) systems representing general analog circuits, analytical approaches are not feasible. Therefore, a discrete modeling approach based on numerical analysis is necessary for formal verification of analog circuits.

This paper will introduce a new discrete modeling algorithm for nonlinear analog circuits, significantly advancing modeling accuracy compared to the state of the art, which is discussed in Section 2. The state space of an analog circuit is defined in Section 3, for which a discrete analog transition structure is introduced in Section 4. In Section 5, the discretization problem for analog circuits is discussed and described by an optimization problem. The new trajectory-directed discretization algorithm is introduced in Section 6 and validated in Section 7, including a model checking case study. Section 8 concludes.

2. RELATED WORK

The first approach to discrete analog state space modeling was developed for checking of the analog behavior of digital circuits against a property specification given as ω-automata [1]. In [2] a verification system applying an extended Computation Tree Logic (CTL) derivative called AnaCTL to a state machine generated from transient responses of analog circuits is proposed. By higher level modeling of analog circuits using labeled hybrid petri nets, verification of analog/mixed signal (AMS) systems but without an automated model generation from the DAE level is introduced in [3]. Generating a strongly abstracted FSM for high-level system simulation directly from the differential equations of the circuit using a learning algorithm on I/O trajectories, the approach in [4] is not intended to represent complex analog properties as it focuses on high abstraction.

With an adaptive state space discretization that controls the size of the axis-parallel hyperbox enclosures of state space regions depending on the level of homogeneity of the state space dynamics, complex properties of nonlinear circuits represented by DAE systems have been verified using CTL specification, extended with an analog operator [5] or with an analog specification language [6]. Although offering the most advanced formal property verification methodology, the structure of the discrete model using hyperboxes to partition the state space introduces a large discretization error and nondeterminism due to not representing the real state space trajectories of the DAEs

This work was financially supported in part by the Singapore National Research Foundation under its Campus for Research Excellence And Technological Enterprise (CREATE) programme.

sufficiently.

In the area of hybrid system modeling for explorative reachability analysis, approaches are available that create conservative approximations of parts of the linear state space dynamics described by ordinary differential equations (ODEs) using polyhedral objects [7] or zonotopes [8]. Modeling using ODEs, however, is a simplification that cannot describe general nonlinear circuits that have to be represented by DAEs.

With these considerations, the goal of this contribution is to develop a discrete modeling approach for nonlinear DAEs that overcomes the modeling inaccuracy of [5, 6] using a more sophisticated state space partitioning with complex geometric objects that was up to now only applied to linear systems. The new modeling can then be used in existing state space-based specification and verification methodologies, thus significantly increasing the modeling accuracy and hence the soundness of such approaches.

3. ANALOG STATE SPACE

We consider an implicit first-order nonlinear DAE system

$$\mathbf{f}(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{u}) = \mathbf{0} \tag{1}$$

obtained by modified nodal analysis from a nonlinear analog circuit network. The state space is spanned by a subset of the DAE system's inner variables. This subset is the vector $\mathbf{z} \subseteq_\zeta \mathbf{x}$ of n_z linear independent state space variables. The subset relation "$\subseteq_\zeta$" is defined by:

$$\mathbf{z} \subseteq_\zeta \mathbf{x} \;\Leftrightarrow\; \forall i \in \{1, ..., n_z\} : z_i = x_{\zeta(i)} \tag{2}$$

A valid assignment of values to the state space variables represents a state of the system. The extended state space $\mathbf{z}^{(e)}$ of an analog system is spanned by the linear independent state space variables $\mathbf{z}$ and the input variables $\mathbf{u}$ with dimension $n_d = n_z + n_u$:

$$\mathbf{z}^{(e)} = \begin{bmatrix} \mathbf{z} \\ \mathbf{u} \end{bmatrix} \tag{3}$$

Candidates $\mathbf{z}^{(all)} \subseteq_\zeta \mathbf{x}$ for state space variables can be identified in the DAE system by their occurrence as first-order time derivatives (FOTDs). This is due to the branch constitutive equations of the circuit elements containing FOTDs such as the current I_{Cap} through a capacitor Cap with capacity C is given by $I_{Cap} = C \cdot \frac{d}{dt} V_{Cap}$. Similarly, inductors introduce inductor currents as a state space variable. Capacitor loops and their dual equivalent of inductor nodes lead to linear dependencies, reducing the number of linear independent state space variables [9]. The linear independent state space variables $\mathbf{z}$ are a subset of $\mathbf{z}^{(all)}$, hence $\mathbf{z} \subseteq_\zeta \mathbf{z}^{(all)} \subseteq_\zeta \mathbf{x}$.

4. DISCRETE ANALOG TRANSITION STRUCTURE

In the domain of discrete state system modeling for formal verification approaches, the Kripke structure [10] is a common model combining an automaton and a labeling of the states with atomic propositions for identifying sets of states where a certain proposition is true. In order to generate a discrete model of an analog circuit, we extend the Kripke structure to a discrete analog transition structure (DATS) which incorporates additional information such as timed transitions that are needed for describing an analog system as follows.

Definition 1. **(Discrete Analog Transition Structure (DATS))**
For a set of atomic state propositions AP and a set of atomic transition propositions TP, the DATS M_{DATS} over AP, TP is a seven-tuple $M_{DATS} = (\Sigma, R, L_A, L_V, L_X, T, L_T)$ where

- *Σ is a finite set of states of the system,*
- *$R \subseteq \Sigma \times \Sigma$ is a total transition relation, hence for every state $\sigma \in \Sigma$ there exists a state σ' such that $(\sigma, \sigma') \in R$,*
- *$L_A : \Sigma \to 2^{AP}$ is a labeling function (LF) that labels each state with the subset of elements from AP that are true in that state,*
- *$L_V : \Sigma \to \mathbb{R}^{n_d}$ is a LF that labels each state with the vector of n_d variables containing the values in this state of the extended state space variables $\mathbf{z}^{(e)}$ of the DAE system,*
- *$L_X : \Sigma \to \mathbb{R}^{n_x}$ is a LF that labels each state with the vector of n_x variables containing the values in this state of the inner variables $\mathbf{x}$ of the DAE system,*
- *$T : R \to \mathbb{R}_0^+$ is a LF that labels each transition from σ to σ' with a real valued positive or zero transition time that represents the time required for the trajectory in the extended state space between these states,*
- *$L_T : R \to 2^{TP}$ is a LF that labels each transition with the set of atomic transition propositions that are true for that transition. This labeling will be used in Section 6.2 for distinguishing transition types.*

Within the structure M_{DATS}, a path π beginning at state σ is a sequence of states $\pi = \sigma_0, \sigma_1, \sigma_2, ..., \sigma_n$ with $\sigma_0 = \sigma$ and $(\sigma_i, \sigma_{i+1}) \in R$ for $0 \leq i < n$.

5. THE DISCRETIZATION PROBLEM FOR ANALOG CIRCUITS

Discrete model generation for analog circuits is the key to applying graph-oriented formal verification algorithms. The task of discrete analog state space modeling is to transfer a continuous analog system represented as a DAE system, discussed in Section 3, into a DATS that was introduced in Section 4:

$$\mathbf{f}(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{u}) = 0 \overset{\text{discrete modeling}}{\longrightarrow} M_{DATS} \tag{4}$$

Therefore, the continuous vector field of the time derivatives of the state space variables that are representing the dynamics of the analog circuit has to be partitioned. As will be detailed in the following, each partition is represented by a state of the DATS model with a transition relation connecting the states corresponding to the dynamic behavior of the circuit. However, the quality of the discretization determines how significant the verification results are. Hence, in the following, discretization criteria are introduced and assembled to an optimization problem.

Consider an infinite point set $\mathcal{Z}$ of points $\mathbf{p}$ in $\mathbb{R}^{n_d}$ of the state space of the circuit that is constrained to user-defined interval boundaries $r = [\underline{r}, \overline{r}]$ for every of the n_d extended state space dimensions:

$$\mathcal{Z} = \{\mathbf{p} \,|\, \underline{r}_i \leq p_i \leq \overline{r}_i\} \text{ for all } 1 \leq i \leq n_d \tag{5}$$

On $\mathcal{Z}$, the continuous vector field $\mathcal{V} : \mathbb{R}^{n_d} \to \mathbb{R}^{n_d}$ is generated by the time derivatives of the state space variables in the vector of extended state space variables

$$\dot{\mathbf{z}}^{(e)} = \begin{bmatrix} \dot{\mathbf{z}} \\ \mathbf{0} \end{bmatrix} \tag{6}$$

of the DAE system describing the circuit:

$$\mathcal{V}(\mathbf{z}^{(e)}) = \{\dot{\mathbf{z}}^{(e)} \,|\, \mathbf{f}(\dot{\mathbf{x}}, \mathbf{x}, \mathbf{u}) = 0\} \tag{7}$$

The vectors $\mathbf{v}_i$ are defining a linearized trajectory in $\mathcal{V}$ from points $\mathbf{p}_i$ to the points $\mathbf{p'}_i$ with $\mathbf{v}_i = \mathbf{p'}_i - \mathbf{p}_i$, calculated by a time step-controlled transient simulation, starting in $\mathbf{p}_i$ with integration time Δt.

The goal is to generate a partitioning of $\mathcal{Z}$ to k non-overlapping partitions $\mathcal{R}_j$ with

$$\bigcup_{1 \leq j \leq k} \mathcal{R}_j = \mathcal{Z} \tag{8}$$

such that the inhomogeneity of the vector field flow within each $\mathcal{R}_j$ is minimal. Each $\mathcal{R}_j$ will represent a state σ_j of the DATS. The concept of inhomogeneity is defined by two criteria which are for a given integration time the difference in direction and the length difference of the infinite set of trajectory vectors in $\mathcal{R}_j$ as follows.

The angle θ_{rs} between any two sampled transition vectors $\mathbf{v}_r$

and $\mathbf{v}_s$ starting in $\mathcal{R}_j$ is defined as:

$$\theta_{rs} = \arccos\left(\frac{\mathbf{v}_r \cdot \mathbf{v}_s}{\|\mathbf{v}_r\|\|\mathbf{v}_s\|}\right) \quad (9)$$

The **maximum direction error** $\epsilon_\theta^{(\mathcal{R}_j)}$ within a $\mathcal{R}_j$ represented by the maximum angle between some $\mathbf{v}_r$ and $\mathbf{v}_s$ is given by:

$$\epsilon_\theta^{(\mathcal{R}_j)} = \max(\theta_{rs} \,|\, \mathbf{p}_r, \mathbf{p}_s \in \mathcal{R}_j) \quad (10)$$

The **overall mean direction error** ϵ_θ over $\mathcal{Z}$ is then defined by:

$$\epsilon_\theta = \frac{1}{k} \sum_{1 \le j \le k} \epsilon_\theta^{(\mathcal{R}_j)} \quad (11)$$

The **length difference ratio** Δ_{rs} between any two sampled transition vectors $\mathbf{v}_r$ and $\mathbf{v}_s$ starting in $\mathcal{R}_j$ is defined as:

$$\Delta_{rs} = \max\left(\frac{\|\mathbf{v}_r\|}{\|\mathbf{v}_s\|}, \frac{\|\mathbf{v}_s\|}{\|\mathbf{v}_r\|}\right) \quad (12)$$

The **maximum length error** $\epsilon_\Delta^{(\mathcal{R}_j)}$ within a $\mathcal{R}_j$ between some $\mathbf{v}_r$ and $\mathbf{v}_s$ is given by:

$$\epsilon_\Delta^{(\mathcal{R}_j)} = \max(\Delta_{rs} \,|\, \mathbf{p}_r, \mathbf{p}_s \in \mathcal{R}_j) \quad (13)$$

The **overall mean length error** over $\mathcal{Z}$ is then defined by:

$$\epsilon_\Delta = \frac{1}{k} \sum_{1 \le j \le k} \epsilon_\Delta^{(\mathcal{R}_j)} \quad (14)$$

Furthermore, increasing the number of partitions and thus decreasing the size of the partitions has negative effects on the efficiency of the verification algorithms. Moreover, in a n-dimensional state space, decreasing the size of every partition to half of its initial size in every dimension increases the number of partitions by the factor 2^n. Therefore, keeping the number of partitions as small as possible is critical for developing feasible discrete modeling algorithms.

Another important criterion is the determinism of the successor relation between adjacent partitions. If the partitions are perfectly enclosing homogeneous state space dynamics, every trajectory starting in a partition ends in a single adjacent partition. Hence, the **out-degree** $\deg(\mathcal{R}_j)$ is 1. We can therefore define the **overall mean out-degree error**

$$\epsilon_{\deg} = \frac{1}{k} \sum_{1 \le j \le k} (\deg(\mathcal{R}_j) - 1) \quad (15)$$

that is 0 if $\deg(\mathcal{R}_j)$ is 1 for $1 \le j \le k$.

Additionally, by transferring the partitioning to a DATS, the successor relation between states representing the partitions determines the paths in the DATS. As the state space variable vectors of the states in the DATS are determined by the centers of the partitions, a sequence of transitions in the DATS corresponds to a trajectory in the state space. If the successor relation of the DATS is determined inaccurately, the behavior of the model does not correspond to the real state space trajectories. Therefore, a successor relation error has to be defined.

Consider two adjacent partitions $\mathcal{R}_i$ and $\mathcal{R}_j$ represented by states σ_i and σ_j, connected by a transition $(\sigma_i, \sigma_j) \in R$ with center points (defined in Eqn. 33) $L_V(\sigma_i) = \mathbf{p}_i^{(c)}$ and $L_V(\sigma_j) = \mathbf{p}_j^{(c)}$ and the vector $\mathbf{v}_i^{(tr)}$ determined by a transient step of length $\|\mathbf{p}_j^{(c)} - \mathbf{p}_i^{(c)}\|$ starting in $\mathbf{p}_i^{(c)}$. The **successor relation error** $\epsilon_{\text{suc}}^{(ij)}$ between two connected adjacent partitions $\mathcal{R}_i$ and $\mathcal{R}_j$ is defined by:

$$\epsilon_{\text{suc}}^{(ij)} = \arccos\left(\frac{(\mathbf{p}_j^{(c)} - \mathbf{p}_i^{(c)}) \cdot \mathbf{v}_i^{(tr)}}{\|\mathbf{p}_j^{(c)} - \mathbf{p}_i^{(c)}\|\|\mathbf{v}_i^{(tr)}\|}\right) \quad (16)$$

The **overall mean successor relation error** for a given partitioning is then defined by:

$$\epsilon_{\text{suc}} = \frac{1}{k} \sum_{1 \le i \le k} \max\{\epsilon_{\text{suc}}^{(ij)} | (\sigma_i, \sigma_j) \in R\} \text{ with } 1 \le j \le k \quad (17)$$

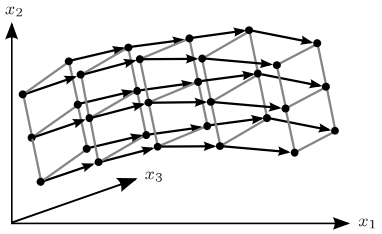


Figure 1: Illustration of a trajectory-directed state space partitioning for a dynamic vector field flow.

With these prerequisites, an optimization problem for the discrete modeling task can now be defined.

DEFINITION 2. **(Optimization Problem for the Discrete Modeling Task)**
Based on the definition of

- *the direction error* $\epsilon_\theta^{(\mathcal{R}_j)}$,
- *the length error* $\epsilon_\Delta^{(\mathcal{R}_j)}$,
- *the number of partitions* k,
- *the overall mean out-degree error* $\epsilon_{\deg}$,
- *and the overall mean successor relation error* ϵ_{suc},

the multi-objective optimization problem connected to partitioning $\mathcal{Z}$ *into* $\mathcal{R}_j$ *with* $1 \le j \le k$ *can now be stated with user defined maximum error bounds* r_θ *and* r_Δ*:*

$$\{\mathcal{R}_1, ..., \mathcal{R}_j, ..., \mathcal{R}_k\} = \arg\min \left\{ \begin{array}{l} k \\ \epsilon_{\deg} \\ \epsilon_{\text{suc}} \end{array} \right\}$$
$$\text{subject to } \left\{ \begin{array}{l} \forall 1 \le j \le k : \epsilon_\theta^{(\mathcal{R}_j)} < r_\theta \\ \forall 1 \le j \le k : \epsilon_\Delta^{(\mathcal{R}_j)} < r_\Delta \end{array} \right\} \quad (18)$$

6. TRAJECTORY-DIRECTED DISCRETE MODELING ALGORITHM

With the optimization problem defined in the previous Section illustrating which criteria have to be considered, a discrete modeling algorithm will be developed in the following that significantly reduces k, $\epsilon_{\deg}$, ϵ_{suc}. The following considerations are required to reach this goal.

The discretization shall be rotation invariant and therefore the state space intersections creating the partitioning cannot be axis-parallel. As an over-approximation of the successor relation of the state space partitions significantly weakens the expressiveness of the verification algorithms, the geometric structure of the state space partitions shall follow the flow of the state space dynamics. Hence, the intersections dividing the state space shall be either parallel or orthogonal to the state space trajectories enclosed by the partitions, thus minimizing ϵ_{suc}. Consequently, $\epsilon_{\deg}$ is also minimized due to the uniqueness of the successor relation, being determined by the real trajectories between the preceding state space partition and its successor. By using time step control algorithms for determining the acceptable trajectory length which can be approximated by a straight line between two points in state space, the homogeneity of the enclosed state space dynamics in the partitions shall be guaranteed. Figure 1 illustrates such a non-axis-parallel trajectory-directed state space partitioning.

6.1 Calculating the State Space Partitioning

The central idea for the new discretization algorithm is that the intersections of the state space are determined by the trajectories of the state space dynamics instead by axis-parallel slicing used by previous approaches. Hence, starting from the infinite point set $\mathcal{Z}$, a slicing structure into k non-overlapping state space regions $\mathcal{R}_j$ of the state space shall be constructed as stated in Equation 8. $\mathcal{Z}$ is constrained to user-defined interval boundaries r for each of the n_d extended state space dimen-

sions, as defined in Equation 5, representing the state space of the analog system.

A linear transformation of $\mathcal{Z}$ is necessary in order to handle size differences of the ranges of the state space variables. All ranges shall be translated and normalized to the interval $[0, 1]$. Therefore, a translation vector $\mathbf{v}^{(t)}$ has to be calculated to move the lower bound of the n_d extended state space variable ranges $\underline{r}_i$ to 0 and to scale them to the interval $[0, 1]$ by factors λ_i:

$$\lambda_i = (\overline{r}_i - \underline{r}_i)^{-1} \tag{19}$$

$$\mathbf{v}^{(t)} = \left[-\underline{r}_1 \cdot \lambda_1 \ \ldots \ -\underline{r}_{n_d} \cdot \lambda_{n_d} \right]^T \tag{20}$$

A transformation matrix $\mathbf{T}$ for the scaling vector λ and the translation vector $\mathbf{v}^{(t)}$ is then generated and the linear transformation for a point $\mathbf{p}^{(orig)}$ to $\mathbf{p}$ with

$$\mathbf{p} = \mathbf{T} \cdot \mathbf{p}^{(orig)} \tag{21}$$

is applied to all following coordinate calculations and reversed for determining the final structure in the untransformed space.

After the preceding preparations, the actual partitioning can be described. The goal of the partitioning algorithm is to determine the vertices and edges of the geometric objects partitioning $\mathcal{Z}$ into the k $\mathcal{R}_j$. Therefore, the algorithm starts from a random starting point that is not a DC-operating-point of the system by appending it to an initially empty waiting list WL. DC-operating-points are detected by a threshold level r_{DC} where the ratio between the norm of the transient step vector and the used integration time falls below this value.

For every point $\mathbf{p}$ in the waiting list, a step-length controlled transition vector $\mathbf{v}$ to the point $\mathbf{p}'$ with $\mathbf{v} = \mathbf{p}' - \mathbf{p}$ is calculated using a transient simulation back-end. In order to identify new starting points for transition vectors, across each vector $\mathbf{v}$ an orthogonal basis vector set $\mathbf{B}$ is constructed:

$$\mathbf{B} = \{\mathbf{b}_1, ..., \mathbf{b}_{n_d} \ : \ \mathbf{b}_i \cdot \mathbf{b}_j = 0\}$$

$$\text{for all } 1 \leq i, j \leq n_d;\ i \neq j;\ \mathbf{v} \in \mathbf{B} \tag{22}$$

$\mathbf{B}$ is constructed using the Gram-Schmidt orthogonalization algorithm [11]. The input to the Gram-Schmidt algorithm

$$\mathbf{B} = \mathbf{GramSchmidt}(\mathbf{M}) \tag{23}$$

is the matrix

$$\mathbf{M} = \begin{bmatrix} \mathbf{v} & \mathbf{i}_1 & \cdots & \mathbf{i}_{j-1} & \mathbf{i}_{j+1} & \cdots & \mathbf{i}_{n_d} \end{bmatrix} \tag{24}$$

consisting of the vector $\mathbf{v}$ and $n_d - 1$ of the n_d column vectors of the unity matrix $\mathbf{I}_{n_d}$ such that the eliminated vector $\mathbf{i}_j$ has its 1-entry in the same dimension j where $\mathbf{v}$ has its maximum absolute magnitude $|v_j|$:

$$j = \underset{1 \leq j \leq n_d}{\arg\max} |v_j| \tag{25}$$

The vectors returned by the algorithm are normalized to length 1. The resulting orthogonal basis set is now scaled to the initial length of $\mathbf{v}$. Additionally, in order to control the discretization error, for each element $\mathbf{b}_i$ of $\mathbf{B}$ a scaling factor $\beta_i^{(a)}$ is calculated. This $\beta_i^{(a)}$ shall assure that the direction and length differences between two transition vectors $\mathbf{v}_r$ and $\mathbf{v}_s$, with $\mathbf{v}_r$ being calculated starting from $\mathbf{p} + \beta_i^{(a)} \cdot \mathbf{b}_i$ and $\mathbf{v}_s$ being the initial transient vector starting from $\mathbf{p}$, are below predefined tolerance levels r_θ and r_Δ:

$$\theta_{rs} < r_\theta \ \wedge \ \Delta_{rs} < r_\Delta \tag{26}$$

Starting with $\beta_i^{(a)} = 1$, the scaling algorithm compares the two transition vectors $\mathbf{v}_r$ and $\mathbf{v}_s$ and decreases the length of $\beta_i^{(a)}$ until the error criteria are satisfied. Therefore, just as the time step control using the local truncation error [12] for controlling the error of transient simulation, a state space step control is applied for controlling $\epsilon_\theta^{(\mathcal{R}_j)}$ and $\epsilon_\Delta^{(\mathcal{R}_j)}$ within each state space partition $\mathcal{R}_j$, determined by the correspondingly scaled $\mathbf{B}$ and the adjacent orthogonal sets.

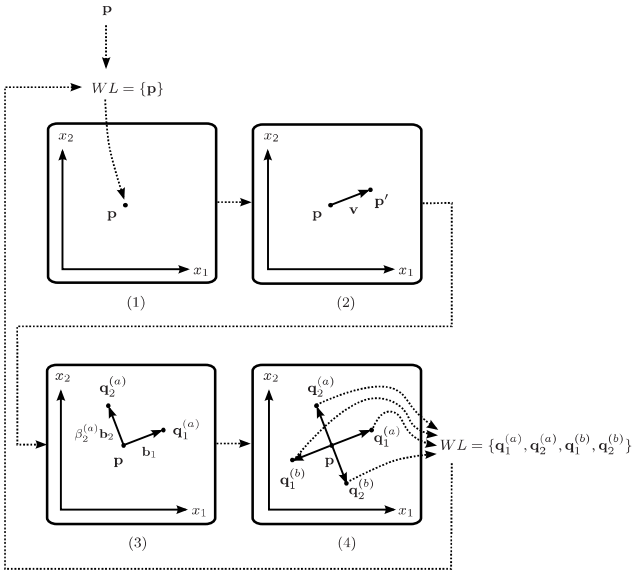


Figure 2: Schematic visualization of the process of determining the trajectory-orthogonal point sets $\mathbf{q}^{(a)}$ and $\mathbf{q}^{(b)}$ in a two-dimensional space.

Satisfying r_θ and r_Δ in the state space area around singularities such as attractors represented by DC-operating-points would cause the length of the vectors $\beta_i^{(a)} \cdot \mathbf{b}_i$ to be decreased infinitely. Therefore, the previously mentioned threshold r_{DC} controls the minimum length of the vectors for obtaining a finite set of partitions.

Each of the scaled orthogonal basis set vectors added to $\mathbf{p}$ describes starting points $\mathbf{q}^{(a)}$ for a new transient step calculation for which in turn the orthogonal basis set is calculated. By an additional point reflection of the vector set $\mathbf{B}$ across $\mathbf{p}$ by vector subtraction, resulting in the point set $\mathbf{q}^{(b)}$, the expansion into all trajectory-orthogonal directions of the state space is obtained with correspondingly calculated error control scaling factors $\beta_i^{(b)}$:

$$\mathbf{q}_i^{(a)} = \mathbf{p} + \beta_i^{(a)} \cdot \mathbf{b}_i \text{ for all } 1 \leq i \leq n_d \tag{27a}$$

$$\mathbf{q}_i^{(b)} = \mathbf{p} - \beta_i^{(b)} \cdot \mathbf{b}_i \text{ for all } 1 \leq i \leq n_d \tag{27b}$$

For every new starting point put into the WL, the inclusion in the defined discretization ranges of the extended state space has to be assured:

$$WL = WL \cup \{\mathbf{q}_i \in (\mathbf{q}^{(a)} \cup \mathbf{q}^{(b)}) | \mathbf{q}_i \in [0, 1]^{n_d}\} \tag{28}$$

Figure 2 outlines the described process of determining the set of trajectory-orthogonal points $\mathbf{q}^{(a)}$ and $\mathbf{q}^{(b)}$ in a two-dimensional space.

The point $\mathbf{q}_1^{(b)}$ generated by the point reflection of the initial transition vector $\mathbf{v}$ across $\mathbf{p}$ is critical. Between $\mathbf{q}_1^{(b)}$ and $\mathbf{p}$ shall be a transition in the direction of the trajectory flow, which is not given by the reflection of $\mathbf{v}$. Hence, the transition vector obtained from a transient step starting in $\mathbf{q}_1^{(b)}$ must not necessarily go through $\mathbf{p}$. Therefore, a correction has to be calculated such that a transient step starting in $\mathbf{q}_1^{(b)}$ goes through $\mathbf{p}$. This can be iteratively resolved by determining the deviation vector

$$\Delta\mathbf{p}_i = \mathbf{p} - \mathbf{q}'_{1_i} \tag{29}$$

between $\mathbf{p}$ and the point $\mathbf{q}'_{1_i}$ with $\mathbf{q}'_{1_i}$ determined by a transient step starting in $\mathbf{q}_{1_i}^{(b)}$. The corrected $\mathbf{q}_{1_{i+1}}^{(b)}$ is then given by:

$$\mathbf{q}_{1_{i+1}}^{(b)} = \mathbf{q}_{1_i}^{(b)} + \Delta\mathbf{p}_i \tag{30}$$

Algorithm 1: Trajectory-Directed Partitioning Algorithm.

Input: Waiting list $WL = \{\mathbf{p}_1\}$
Output: List of accepted partitioning points PL with geometric topology

apply transformation $\mathbf{T}$ to all following steps
foreach $\mathbf{p}_j \in WL$ **do**
  $WL = WL \setminus \mathbf{p}_j$
  calculate transient step vector $\mathbf{v}_j = \mathbf{p}'_j - \mathbf{p}_j$
  generate orthogonal set $\mathbf{B}$ from $\mathbf{v}_j$
  foreach $\mathbf{b}_i \in \mathbf{B}$ **do**
    calculate error control scaling factors $\beta_i^{(a)}$ and $\beta_i^{(b)}$
    calculate points $\mathbf{q}_i^{(a)}$ and $\mathbf{q}_i^{(b)}$
    if *i==1* **then**
      calculate corrected $\mathbf{q}_1^{(b)}$
    end
  end
  foreach $\mathbf{q}_i \in \{\mathbf{q}^{(a)} \cup \mathbf{q}^{(b)}\}$ **do**
    if $\mathbf{q}_i \in [0,1]^{n_d}$ **then**
      if $\neg\exists\, \mathbf{p}^{(ex)} \in (PL \cup WL) : \|\mathbf{p}^{(ex)} - \mathbf{q}_i\| < \gamma \cdot \|\mathbf{p}_j - \mathbf{q}_i\|$ **then**
        $WL = WL \cup \mathbf{q}_i$
      end
    end
  end
  $PL = PL \cup \mathbf{p}_j$
end
reverse transformation $\mathbf{T}$

This correction algorithm is repeated either up to a predefined number i_{max} of times or until $\Delta\mathbf{p}_i$ is below a user-defined error bound. If the algorithm terminates without $\Delta\mathbf{p}$ being acceptable, the length of $\mathbf{v}$ being projected to generate the initial guess for $\mathbf{q}_1^{(b)}$ as well as of all transient step calculations are halved and the process is repeated until the error bound is reached.

Another issue is posed by the set of new starting points to put into the waiting list possibly containing points that are very close to points that have already been processed. Hence, a k-d tree-based proximity criterion has to control the structure of the new starting points in order to avoid overlapping with existing points, giving priority to those points generated by transition vectors over those generated by the orthogonal vectors. If any of the new starting points $\mathbf{q}_i$ from $\mathbf{q}^{(a)}$ or $\mathbf{q}^{(b)}$ is closer to an already calculated existing point than a defined threshold value γ, which is by default 0.75 times the distance between $\mathbf{p}$ and $\mathbf{q}_i$, $\mathbf{q}_i$ is considered as redundant. In this case, $\mathbf{q}_i$ is replaced by the existing point or vice versa, keeping points originating from transient steps. The accepted points from $\mathbf{q}^{(a)}$ and $\mathbf{q}^{(b)}$ are appended to the waiting list WL. Every point in the WL that has been processed is removed from the WL and put into the list PL of accepted partitioning points.

The orthogonal sets to which an accepted point is connected to are stored, in order to later reconstruct the topology of geometric objects from these points. These connections are either represented by transient step vectors or by those from the orthogonal basis set. The coordinates of the accepted points in the untransformed state space can be calculated by inversing the transformation from Equation 21. The trajectory-directed partitioning algorithm is summarized in Algorithm 1.

6.2 Mapping the Trajectory-Directed Partitioning to a DATS

With the vertices and edges of the state space partitioning determined by using the trajectory-directed approach, a mapping onto a DATS has to be generated in order to apply graph-based formal verification algorithms.

Each state σ_i of the DATS is corresponding to one $\mathcal{R}_i$ with $1 \leq i \leq k$. Hence, the cardinality of Σ is k:

$$\Sigma = \{\sigma_1, ..., \sigma_k\} \tag{31}$$

The DATS is then constructed by the following mappings. The parameter vectors of the states are given by the labeling

$$L_V(\sigma_i) = \mathbf{p}_i^{(c)} \tag{32}$$

with $\mathbf{p}_i^{(c)}$ being the center point of $\mathcal{R}_i$, representing the extended state space variables in this point. The center is calculated from the $m = 2^{n_d}$ vertices $\mathbf{p_j}$ that constrain $\mathcal{R}_i$:

$$\mathbf{p}_i^{(c)} = \frac{1}{m} \sum_{1 \leq j \leq m} \mathbf{p}_j \tag{33}$$

A transition exists between those states where the state space trajectories starting in $\mathcal{R}_i$ reach the adjacent $\mathcal{R}_j$. The adjacency is determined by the intersection of the sets of vertices $\mathbf{p}^{(i)}$ spanning $\mathcal{R}_i$ and $\mathbf{p}^{(j)}$ spanning $\mathcal{R}_j$ not being empty:

$$R = \{\bigcup(\sigma_i, \sigma_j) | \forall\, \mathbf{p} \in \mathcal{R}_i \,\exists\, \Delta t : \mathbf{p} + \mathbf{v} \cdot \Delta t \in \mathcal{R}_j\} \tag{34}$$

with $\mathbf{p}^{(i)} \cap \mathbf{p}^{(j)} \neq \varnothing$. The transition times between σ_i and σ_j are determined by the trajectory time Δt computed by a transient step from $\mathbf{p}_i^{(c)}$ to $\mathbf{p}_j^{(c)}$:

$$T(R(\sigma_i, \sigma_j)) = \Delta t(\mathbf{p}_i^{(c)}, \mathbf{p}_j^{(c)}) \tag{35}$$

Each transition from a transient step has an atomic transition proposition of 0, marking it as dynamic transition created by a state space trajectory step:

$$T(R(\sigma_i, \sigma_j)) > 0 \Rightarrow L_T(R(\sigma_i, \sigma_j)) = 0 \tag{36}$$

A trajectory starting in a DC-operating-point ends within this point, hence there is a transition of a state representing a DC-operating-point to itself:

$$(\sigma_i, \sigma_i) \in R \Leftrightarrow \{\forall\, \mathbf{p} \in \mathcal{R}_i : \mathbf{p} + \mathbf{v} \cdot \Delta t \in \mathcal{R}_i\} \text{ for all } \Delta t \geq 0 \tag{37}$$

By definition, the transition time of such transitions shall be zero, as the circuit stays in this loop state until an external excitation makes the circuit leave this state:

$$T(R(\sigma_i, \sigma_i)) = 0 \text{ and } L_T(R(\sigma_i, \sigma_i)) = 0 \tag{38}$$

In order to allow arbitrary input changes for transitions between states in the discrete model, for each state and for each input dimension, input transitions for a state to its successor and predecessor parallel to the axis of the corresponding input dimension have to be created in the DATS. For the s-th input axis, the orthogonalization around the transition vector $\mathbf{v}$ starting in $\mathbf{p}_i$ generates the points $\mathbf{q}_s^{(a)}$ and $\mathbf{q}_s^{(b)}$ with $\mathbf{b}_s$ being parallel to this input axis. This is due to the trajectories being sampled with piecewise constant inputs, and the inputs are only changed between sample steps. Hence, the dynamic transition vectors have zero magnitude in the direction component of the input dimensions.

For a state σ_i, every identified neighboring state σ_j for every input dimension is finally connected to σ_i by an undirected transition in the DATS, meaning that these input edges can be traveled in both directions corresponding to any external input variable change:

$$R = R \cup (\sigma_i, \sigma_j) \cup (\sigma_j, \sigma_i) \tag{39}$$

By definition, the transition time of such transitions shall be zero:

$$T(R(\sigma_i, \sigma_j)) = 0 \text{ and } T(R(\sigma_j, \sigma_i)) = 0 \tag{40}$$

Finally, these transitions have to be identified as input edges in the DATS for offering the possibility of masking these states for verification algorithms such as oscillation detection:

$$L_T(R(\sigma_i, \sigma_j)) = 1 \text{ and } L_T(R(\sigma_j, \sigma_i)) = 1 \tag{41}$$

6.3 Discretization Runtime Complexity

Considering the discretization of an analog circuit to a DATS, the asymptotic worst-case runtime complexity of the

trajectory-directed discretization algorithm is correlated to the number of points n_p sampled in the state space of the modeled circuit. This number of points increases exponentially with the number of dimensions n_d of the extended state space (variables of the energy storing elements and input dimensions). The base κ of the exponential function represents the average number of sampling points needed for covering an one-dimensional range, which is determined by the step length control of the transient steps between the sampled points in the state space:

$$n_p = \kappa^{n_d} \tag{42}$$

For every sampled point, the trajectory-directed discretization algorithm computes the transient simulation step contributing t_{tr}, the Gram-Schmidt orthogonalization contributing t_{gs} and the distance information contributing t_{pr} for the proximity criterion. The remaining parts of the algorithm with subordinate contribution to the complexity of the algorithm are summed up in t_{re}. An exact asymptotic complexity for transient analysis depends on the applied set of algorithms. However, the transient analysis algorithm contains parts with cubic asymptotic worst-case complexity with respect to the matrix of the circuit equations and the number of variables being related to n_d. The Gram-Schmidt algorithm has a complexity of $\mathcal{O}(n_d^3)$ [11]. The proximity neighbor search conducted by a k-d tree consumes $\mathcal{O}(n_p \log n_p)$ to be created and $\mathcal{O}(\log n_p)$ for the query [13]. Hence, the runtime t_d of a discretization run can be estimated by:

$$t_d = \kappa^{n_d} \cdot \underbrace{(t_{tr} + t_{gs} + t_{pr} + t_{re})}_{t_p} \tag{43}$$

The overall computation time of a single sample point t_p is dominated by the large factor of transient analysis t_{tr}, with t_{pr} only becoming dominant for high state numbers. Anyhow, with respect to n_p, the asymptotic runtime complexity of the discretization algorithm is dominated by the proximity computation as it is the only component with direct dependency on n_p:

$$\mathcal{C}_{n_p} = \mathcal{O}(n_p \log n_p) \tag{44}$$

However, changing the perspective of complexity considerations to be relative to the number of state space dimensions, the asymptotic worst-case runtime complexity of the trajectory-directed discretization algorithm is dominated by the exponential growth of sample points with respect to the number of state space dimensions:

$$\mathcal{C}_{n_d} = \mathcal{O}(\kappa^{n_d}) \tag{45}$$

While this exponential runtime complexity in the number of extended state space dimensions is common to all discrete modeling approaches for analog circuits, relevant analog circuit blocks usually do not exceed a system order of eight, which can be handled well by this approach.

7. EXPERIMENTAL VALIDATION

7.1 Modeling Accuracy Analysis

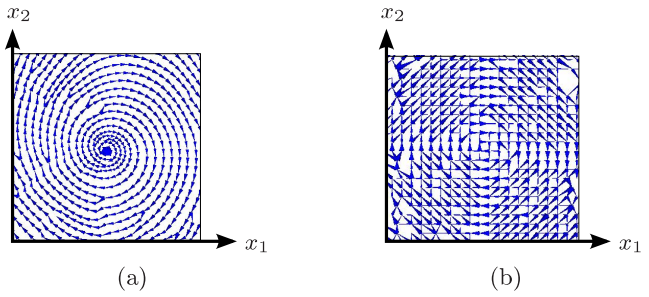
In this section, the modeling accuracy of a prototypic C++ and GNU Octave implementation of the new trajectory-directed discretization (TDD) approach is validated on example circuits, comparing k, ϵ_{suc}, and ϵ_{deg}, as defined in Section 5, to the results of the hyperbox discretization (HBD) using the tool from [5].

As an initial illustrative example, on the generated transition structure of the DATS, a comparison between the new TDD and HBD is made for a damped oscillator described by the differential equation system

$$\dot{\mathbf{x}}_\mathbf{1} - \mathbf{x}_\mathbf{2} = \mathbf{0} \tag{46a}$$

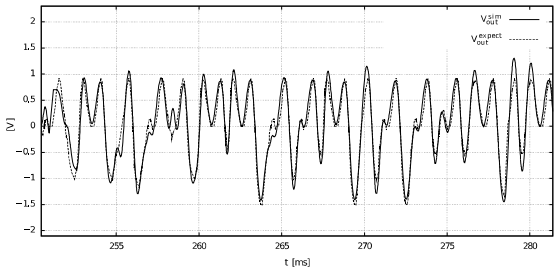
$$-\dot{\mathbf{x}}_\mathbf{2} - \sin(\mathbf{x}_\mathbf{1}) - \alpha \cdot \mathbf{x}_\mathbf{2} = \mathbf{0} \tag{46b}$$

with $\alpha = 0.05$. The results depicted in Figure 3 show that the TDD captures the correct state space trajectories with a decreasing amplitude towards the center. In contrast, the HBD generates a massive overapproximation of the transitions, not capturing the damped oscillation. Paths in the HBD model exist that either allow oscillation at fixed amplitudes forever or shortcutting towards the center fixpoint immediately, massively limiting expressiveness of the model.

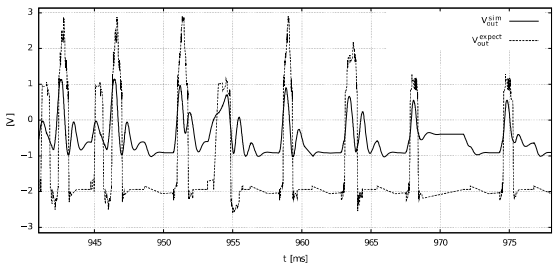


(a) (b)

Figure 3: Transition structure of the damped oscillator for the new TDD modeling (a) and HBD (b).



(a)



(b)

Figure 4: Transient response V_{out}^{sim} of the 2nd-order biquad lowpass filter compared to V_{out}^{expect} calculated on the DATS model generated by TDD (a) and HBD (b).

For another comparison of the discretization quality, a DATS model has been generated for an active 2nd-order biquad lowpass filter circuit [5] using the TDD approach and using the HBD approach. For a piecewise-linear input signal, a representative excerpt of the transient response V_{out}^{sim} and the expected behavior V_{out}^{expect}, calculated as a sequence of states on the DATS, are shown in Figure 4. In comparison to the matching waveforms computed by the TDD, the results obtained by the HBD show a massive deviation between transient simulation and the behavior of the discrete model. A comparison of ϵ_{suc} supports the conclusion of a large improvement of the TDD (5.11°) over the HBD (29.58°). The results for the presented damped oscillator and the 2nd-order biquad lowpass filter, as well as for a charge pump circuit [6] and a tunnel diode oscillator [5], are summarized in Table 1. All runtimes are computed on an Intel Core i5 with 2.6 GHz clock frequency and 4 GB of RAM. The higher runtimes of TDD compared to HBD are due to the prototypic implementation and the proximity criterion evaluation.

Table 1: Comparison of the number of dimensions n_d, number of states k, error criteria ϵ_{suc}, ϵ_{deg}, and the runtimes τ between TDD and HBD modeling for different circuits.

Circuit	n_d	Algo.	k	ϵ_{suc}	ϵ_{deg}	τ
Damped Oscillator	2	TDD	480	3.89°	0	6.2s
		HBD	838	39.39°	2.9	2.7s
Biquad Lowpass	3	TDD	1546	5.11°	0	138s
		HBD	2060	29.58°	2.23	91s
Charge Pump	4	TDD	2482	9.37°	0	152s
		HBD	2878	38.28°	1.38	108s
T-Diode Oscillator	2	TDD	982	4.46°	0	6.7s
		HBD	1871	22.93°	1.91	3.1s

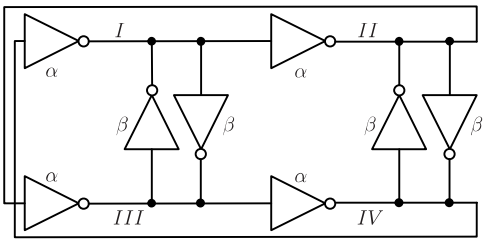


Figure 5: Modified ring oscillator with an even number of inverter stages and cross-coupling.

Finally, the ASL model checking tool from [6] has been modified to operate on the new TDD instead of the HBD. The obtained model checking results for the discussed circuits correspond to the increased modeling quality with the reported property values from model checking being significantly closer to those from transient analysis. In the next section, a model checking case study will illustrate the capabilities of the new discretization approach.

7.2 Model Checking Case Study

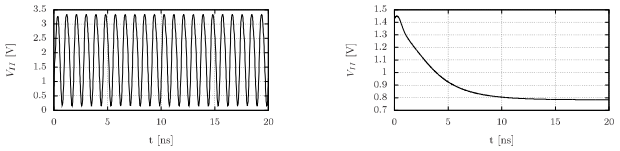
The following case study will illustrate that conventional analog circuit simulation within a test bench setup can lead to wrong verification assumptions and how, in contrast, a model checking approach using trajectory-directed discretization can identify hidden design errors.

When a finite number of simulations with input stimuli or initial conditions is conducted and the expected behavior of the circuit is validated by the simulations, in today's industrial verification applications the circuit is assumed to be successfully verified due to the lack of formal verification tools. However, not finding a specification violation with simulation runs cannot prove that the specification is satisfied under all circumstances.

The example circuit illustrated in Figure 5 is a modified ring oscillator with an even number of inverter stages and cross-coupling [14]. Due to the bridges β, the circuit oscillates if there is a ratio α/β of the transistor sizes in the feedback chain to those of the bridges within the interval [0.4, 2.0].

This circuit has in fact been considered as successfully verified by transient simulation with a set of predefined initial conditions and went into production. What was discovered only after the tapeout of the circuit is its crucial property of being prone to certain initial conditions that prevent it from oscillating when the α/β ratio reaches or exceeds the interval boundaries. These particular initial conditions have not been covered by the simulation runs during verification. For two different initial conditions, the simulation runs are illustrated in Figure 6.

The critical behavior with certain initial conditions could have been detected by applying model checking. In order to



(a) (b)

Figure 6: Transient responses for the node voltage V_{II} of the ring oscillator for transistor ratio $\alpha/\beta = 1.95$. With initial conditions $V_I, V_{II}, V_{III} = 0$ V and $V_{IV} = 0.5$ V, the circuit oscillates (a). With the initial conditions $V_I = 3.33444$ V, $V_{II} = 1.49605$ V, $V_{III} = 3.16195$ V, $V_{IV} = 0.207917$ V detected by formal model checking, the circuit does not oscillate (b).

demonstrate this, a formal property verification of the circuit was performed using the ASL verification methodology [6] on a DATS generated with the new trajectory-directed approach.

The properties to verify are the existence of oscillation and proving that the circuit oscillates for every possible initial condition. Hence, the ASL specification shown in Listing 1 was developed.

Listing 1: ASL specification for oscillator verification.

```
# Assert that the circuit oscillates
osci_set = on all select oscillation;
for is_empty(osci_set) assert false;

# Assert that circuit has no non−periodic steady states
for is_empty(DCpoints) assert true;

# Which initial conditions lead into DCpoints?
bad_initial_conditions = reach DCpoints;
```

The simple property specification in Listing 1 first checks if there is a periodic oscillation trajectory in the state space. The next assertion requires the oscillator's state space not to contain any non-periodic steady states. If those exist, a set `bad_initial_conditions` is assigned with the states that can reach these steady states. Every state from this set represents an initial condition that causes the circuit to run into a steady state instead into the oscillation trajectory. The transient simulation in Figure 6(b), started from such a bad initial condition identified by the model checking algorithms, shows that the circuit in fact does not oscillate.

Furthermore, model checking runs have been conducted for the α/β-ratios 1.05, 1.35, 1.65 and 1.95 in order to systematically check the circuit properties and to prove the consistency of the model checking on the DATS model generated by the trajectory-directed discretization algorithm.

Table 2 summarizes the verification results including the oscillation periods reported by the ASL model checking algorithms as well as by transient analysis for $V_{DD} = 3.3$ V. If bad initial conditions were detected by model checking, a transient analysis run was conducted with these conditions in order to prove that the circuit shows the expected behavior. Additionally, information about modeling and model checking runtimes are denoted in the table.

Figure 7 shows the transition vectors between states of the detected oscillation set in the DATS and the non-periodic steady states identified by ASL model checking for an α/β-ratio of 1.95 projected to the state space dimensions V_I, V_{II} and V_{III}. Figure 8 illustrates the state space trajectories leading into the non-periodic steady states.

Table 2: Verification results for the modified ring oscillator with results obtained from model checking (MC) on the TDD model and transient analysis (TRA).

α/β-ratio	1.05	1.35	1.65	1.95
MC reports bad init. cond.	-	-	-	✓
# States DATS	16036	15810	14680	13288
Discretization Runtime	13 : 22 m	12 : 42 m	11 : 24 m	10 : 30 m
MC Runtime	7.5 s	7.1 s	6.0 s	5.4 s
Oscillation period MC	1.494 μs	1.224 μs	1.055 μs	0.961 μs
Oscillation period TRA	1.415 μs	1.159 μs	1.014 μs	0.936 μs

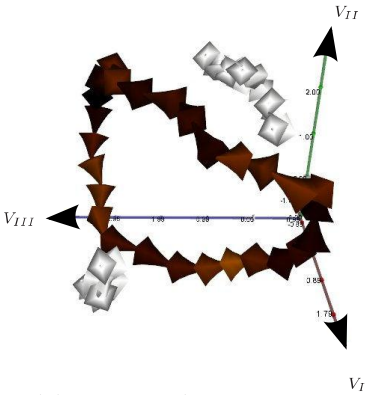


Figure 7: Transition vectors between states of the detected oscillation set and non-periodic steady states of the modified ring oscillator detected by model checking for α/β-ratio 1.95.

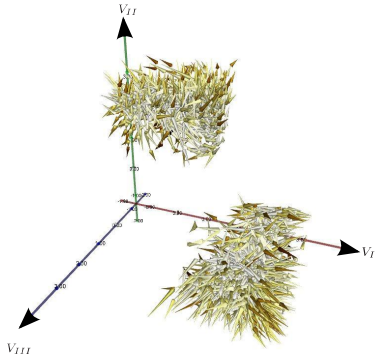


Figure 8: State space trajectories of initial conditions leading into the non-oscillating steady states for transistor ratio $\alpha/\beta = 1.95$.

8. CONCLUSIONS

In this paper, a trajectory-directed state space modeling was presented that significantly improves modeling accuracy, for the first time allowing to capture the dynamics of the nonlinear analog state space in a discrete model with an accuracy that is close to transient analysis. Moreover, an improvement in accuracy of a factor of 4 to 10 compared to the previous state-of-the-art approach can be observed from the experimental results. Due to the higher modeling accuracy, this is achieved with a lower number of states at a slightly longer runtime.

The new approach allows to represent circuits more efficiently, attenuating the exponential relation between the generated states and the number of state space dimensions that is common for all complete discrete state space modeling approaches. The application to a model checking case study shows the capability of the modeling approach to capture critical circuit properties. Future work will focus on an on-demand discretization controlled at runtime by the verification algorithms, further reducing the number of sampled states by continuously evaluating which regions of the state space have to be explored by the modeling algorithm.

9. REFERENCES

[1] R. P. Kurshan and K. L. McMillan. Analysis of digital circuits through symbolic reduction. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 10(11):1356–1371, 1991.

[2] T. R. Dastidar and P. P. Chakrabarti. A verification system for transient response of analog circuits. *ACM Trans. Des. Autom. Electron. Syst.*, 12(3):1–39, 2007.

[3] D. Walter, S. Little, C. Myers, N. Seegmiller, and T. Yoneda. Verification of analog/mixed-signal circuits using symbolic methods. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 27(12):2223 –2235, 2008.

[4] C. Gu and J. Roychowdhury. FSM model abstraction for analog/mixed-signal circuits by learning from I/O trajectories. In *Proc. of the 16th Asia and South Pacific Design Automation Conference*, ASPDAC '11, pages 7–12, 2011.

[5] W. Hartong, L. Hedrich, and E. Barke. Model checking algorithms for analog verification. In *Proc. of the 39th conference on Design automation (DAC '02)*, pages 542–547, 2002.

[6] S. Steinhorst and L. Hedrich. Model Checking of Analog Systems using an Analog Specification Language. In *Proc. of the Conference on Design, Automation and Test in Europe 2008 (DATE'08)*, pages 324–329, 2008.

[7] T. Dang, A. Donzé, and O. Maler. Verification of analog and mixed-signal circuits using hybrid system techniques. In A. J. Hu and A. K. Martin, editors, *FMCAD*, volume 3312 of *Lecture Notes in Computer Science*, pages 21–36. Springer, 2004.

[8] M. Althoff, A. Rajhans, B. H. Krogh, S. Yaldiz, X. Li, and L. Pileggi. Formal verification of digital phase-locked loops using reachability analysis and continuization. In *ICCAD '11: Proc. of the 2011 International Conference on Computer-Aided Design*, pages 659–666, 2011.

[9] D. E. Schwarz and C. Tischendorf. Structural analysis of electric circuits and consequences for MNA. *Int. Journal of Circuit Theory and Applications*, 28(2):131–162, 2000.

[10] E.M. Clarke, O. Grumberg, and D.A. Peled. *Model checking*. Springer, 1999.

[11] G.H. Golub and C.F. Van Loan. *Matrix computations*. Johns Hopkins Univ Pr, 1996.

[12] A. Vladimirescu. *The SPICE book*. John Wiley & Sons, Inc. New York, NY, USA, 1994.

[13] J. L. Bentley. K-d trees for semidynamic point sets. In *SCG '90: Proceedings of the sixth annual symposium on Computational geometry*, pages 187–197, New York, NY, USA, 1990. ACM.

[14] K. D. Jones, J. Kim, and V. Konrad. Some 'real world' problems in the analog and mixed signal domains. In G. J. Pace and S. Singh, editors, *Seventh International Workshop on Designing Correct Circuits, Budapest*, pages 15–29. ETAPS 2008, March 2008.