Reachability analysis of a class of hybrid gene regulatory networks*

Honglu Sun[1][0000−0002−8265−0984], Maxime Folschette[2][0000−0002−3727−2320], and Morgan Magnin[1][0000−0001−5443−0506]

[1] Nantes Université, École Centrale Nantes, CNRS, LS2N, UMR 6004, F-44000 Nantes, France
[2] Univ. Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France

Abstract. In this work, we study the reachability analysis method of a class of hybrid system called HGRN which is a special case of hybrid automata. The reachability problem concerned in this work is, given a singular state and a region (a set of states), to determine whether the trajectory from this singular state can reach this region. This problem is undecidable for general hybrid automata, and is decidable only for a restricted class of hybrid automata, but this restricted class does not include HGRNs. A priori, reachability in HGRNs is not decidable; however, we show in this paper that it is decidable in certain cases, more precisely if there is no chaos. Based on this fact, the main idea of this work is that if the decidable cases can be determined automatically, then the reachability problem can be solved partially. The two major contributions are the following: firstly, we classify trajectories into different classes and provide theoretical results about decidability; then based on these theoretical results, we propose a reachability analysis algorithm which always stops in finite time and answers the reachability problem partially (meaning that it can stop with the inconclusive result, for example with the presence of chaos).

Keywords: Reachability · Hybrid system · Decidability · Gene regulatory networks · Limit cycle.

1 Introduction

Reachability problem of dynamical system has been investigated on different formalisms, majorly on discrete systems [14,23,6] and hybrid systems [19,2,3,15,8,25]. In this work, we study a reachability analysis method on a class of hybrid system called hybrid gene regulatory network (HGRN) [7,4], which is an extension of Thomas' discrete modeling framework [27,28]. This hybrid system is proposed to model gene regulatory networks, which are networks of genes describing the regulation relations between genes.

HGRNs are similar to piecewise-constant derivative systems (PCD systems) [2] which is a special case of hybrid automata [1]. The major difference between

* Supported by China Scholarship Council.

HGRNs and PCD systems of the works [2,3,25] is the existence of sliding mode, which means that when a trajectory reaches a black wall (a boundary of the discrete state which can be reached but cannot be crossed by trajectories), it is forced to move along the black wall. There exist other methods to define behaviors of trajectories on a black wall [17,24] which are different from the sliding mode in HGRNs.

The reachability problem concerned in this work is to determine whether the trajectory from certain state can reach a certain region (a set of states). We mainly focus on the decidability problem, that is, whether we can find an algorithm to determine the reachability problem such that this algorithm always stops in finite time and gives a correct answer.

The decidability problem among hybrid systems that are close to HGRNs is already studied in the literature. It has been proved that, for PCD systems, it is decidable in 2 dimensions [21] but it is undecidable in 3 dimensions [2]. For general hybrid automata, there exists a restricted class called initialized rectangular automata which is decidable in any dimension [19], but this class does not include HGRNs.

Up to now, there is no theoretical results of the decidability of this problem on HGRNs. A priori, we can expect that it is not decidable because of the existence of chaos. However, if we can show that it is decidable in certain cases, for example, when the trajectory considered in a reachability problem converges asymptotically to a n-dimensional limit cycle, and if these cases can be identified automatically, then the reachability problem can be answered partially, which is the main idea of this work. In order to prove the existence of chaos in HGRNs, we exhibit a HGRN with a chaotic attractor based on a different pre-existing hybrid system [18]. This work has the following contributions:

- We classify trajectories of HGRNs into three classes: trajectories halting in finite time, trajectories attracted by regularly oscillating cycles and chaotic trajectories. For the first two classes, we prove that the reachability problem is decidable and we provide methods to determine automatically their classes. For the third class, a priori, it is undecidable, and we provide a necessary condition for that a trajectory is chaotic.
- Based on the above theoretical results, we propose a reachability analysis algorithm for HGRNs which always stops in finite time and once it stops, it returns whether the set of target states is reached, not reached or if the result is unknown. The unknown result is related to the existence of chaos. To our knowledge, this is the first reachability analysis algorithm for HGRNs and it can be applied to HGRNs in any dimension.

This paper is organized as follows. In Section 2, we introduce basic notions of HGRNs. In Section 3, we present our reachability analysis method, including theoretical results and the reachability analysis algorithm. And finally in Section 4, we make a conclusion by discussing the merits and limits of this method and our future work.

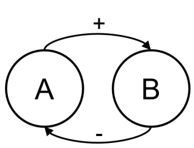
2 Preliminary Definitions

In this section, we present HGRNs and its basic notions. Consider a gene regulatory network with N genes; the set of genes is denoted $G = \{G_1, G_2, ..., G_N\}$. A *discrete state* is an integer vector of length N, noted by d_s, which assigns the discrete level d_s^i to gene G_i, where $i \in \{1, 2, 3, ..., N\}$ and d_s^i is the i^{th} component of d_s. The set of all discrete states is denoted by E_d.

A *hybrid gene regulatory network* (HGRN) is noted $\mathcal{H} = (E_d, c)$. c is a function from E_d to $\mathbb{R}^N$. For each $d_s \in E_d$, $c(s)$, also noted c_s, is called the *celerity* of discrete state d_s and describes the temporal derivative of the system in d_s. A 2-dimensional HGRN is shown in Fig 1. In this system, each of these two genes ($A = G_1, B = G_2$) has two discrete levels: 0 and 1, so there are 4 discrete states: $00, 01, 10, 11$. Black arrows represent the celerities (temporal derivatives) of each discrete state.

In HGRNs, a *state* is also called a *hybrid state*, which is a couple $h = (\pi, d_s)$ containing a *fractional part* π, which is a real vector $[0, 1]^N$, and a discrete state d_s. The set of all hybrid states is denoted by E_h.

A *(hybrid) trajectory* τ of HGRN is a function from a time interval $[0, t_0]$ to $E_\tau = E_h \cup E_{sh}$, where $t_0 \in \mathbb{R}^+ \cup \{\infty\}$, and E_{sh} is the set of all finite or infinite sequences of states: $E_{sh} = \big\{(h_0, h_1, ..., h_m) \in (E_h)^{m+1} \mid m \in \mathbb{N} \cup \{\infty\}\big\}$. A trajectory τ is called a *closed trajectory* if it is defined on $[0, \infty[$ and $\exists T > 0, \forall t \in [0, \infty[, \tau(t) = \tau(t+T)$. In Fig 1, red arrows represent a possible trajectory of this system, which happens, in this particular case, to be a closed trajectory.



A	B	C_A	C_B
0	0	0.6	−0.7
0	1	−0.7	−0.9
1	0	0.7	0.8
1	1	−0.6	0.9

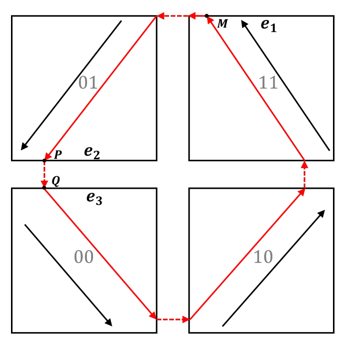


Fig. 1: Example of a HGRN in 2 dimensions. Left: Influence graph (negative feedback loop with 2 genes). Middle: Example of corresponding parameters (celerities). Right: Corresponding example of dynamics; abscissa represents gene A and ordinate represents gene B.

A *boundary* in a discrete state d_s is a set of states defined by $e(G_i, \pi_0, d_s) = \big\{(\pi, d_s) \in E_h \mid \pi^i = \pi_0, \big\}$, where $i \in \{1, 2, ..., N\}, d_s \in E_d$ and $\pi_0 \in \{0, 1\}$. In the rest of this paper, we simply use e to represent a boundary.

In Fig 1, the state $h_M = ((\pi^1_M, 1), (1, 1))$ of point M belongs to $e_1 = (B, 1, (1, 1))$, that is, the upper boundary in the second dimension (the dimension of gene B) of the discrete state 11. Since there is no other discrete state on the other side of e_1, the trajectory from h_M cannot cross e_1 and has to slide along e_1 (e_1 can be called a black wall). The existence of such sliding mode is a speciality of HGRNs. Boundaries like e_1, which can be reached by trajectories but cannot be crossed, are defined as *attractive boundaries*. The state $h_P = ((\pi^1_P, 0), (0, 1))$ of point P belongs to $e_2 = (B, 0, (0, 1))$, the lower boundary in the second dimension of the discrete state 01. The trajectory from h_P reaches instantly $h_Q = ((\pi^1_Q, 1), (0, 0))$, which belongs to $e_3 = (B, 1, (0, 0))$, the upper boundary in the second dimension of discrete state 00, because the celerities on both sides allow this (instant) discrete transition. e_2 is called an *output boundary* of 01 and e_3 is called an *input boundary* of 00.

When a trajectory reaches several output boundaries at the same time (Fig 2 left), it can cross any of them but can only cross one boundary at a time, which causes non-deterministic behaviors. The simulation of HGRNs is presented more formally in the Appendix.

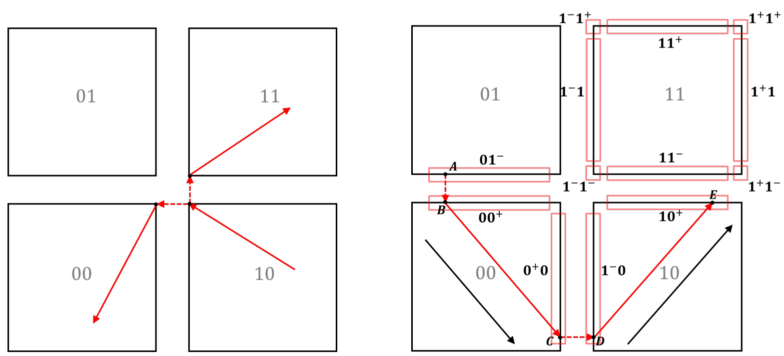


Fig. 2: Left: Illustration of a non-deterministic behavior. Right: Illustration of all discrete domains of state 11, and a sequence of discrete domains in the other states.

In order to analyze dynamical properties of HGRNs, the concepts of *discrete domain*, *transition matrix* and *compatible zone* are introduced in [26]. A *discrete domain* $\mathcal{D}(d_s, S_-, S_+)$ is a set of states inside one discrete state d_s, defined by:

$$\mathcal{D}(d_s, S_-, S_+) = \{(\pi, d_s) \mid \forall i \in \{1, 2, ..., N\}, \pi^i \in \begin{cases} \{1\} & \text{if } i \in S_+ \\ \{0\} & \text{if } i \in S_- \\ ]0, 1[& \text{if } i \notin S_- \cup S_+ \end{cases} \}$$

where S_+ and S_- are power sets of $\{1, 2, ..., N\}$ such that $S_+ \cap S_- = \emptyset$ and $S_+ \cup S_- \neq \emptyset$. In fact, S_+ (S_-) represents the dimensions in which the upper (lower) boundaries are reached by any state $h \in \mathcal{D}(d_s, S_-, S_+)$. In the rest of this paper, we simply use $\mathcal{D}$ to represent a discrete domain when there is no ambiguity.

Some discrete domains are illustrated in Fig 2 right. For example, 11^+ denotes the discrete domain inside discrete state 11 where the upper boundary is reached for the second dimension and no boundary is reached for the first dimension, that is: $\mathcal{D}((1,1), \emptyset, \{2\}) = \{(\pi, (1,1)) \mid \pi^1 \in]0,1[\wedge \pi^2 = 1\}$. The state D in this figure belongs to the discrete domain 1^-0. The discrete state 11 contains 8 discrete domains: 1^-1^-, 11^-, 1^+1^-, 1^-1^+, 11^+, 1^+1^+, 1^-1 and 1^+1, which are depicted in Fig 2 right. Note that, for instance, 1^+1^+ is represented by a small red rectangle for readability, but in fact it only contains one singular hybrid state $((1,1),(1,1))$.

To order to introduce the concepts of *transition matrix* and *compatible zone*, consider a sequence of discrete domains $\mathcal{T} = (\mathcal{D}_i, \mathcal{D}_{i+1}, \mathcal{D}_{i+2}, ..., \mathcal{D}_j)$ in the rest of this section and assume that there is a trajectory τ which starts from $h_i = (\pi_i, d_{s_i}) \in \mathcal{D}_i$, reaches all discrete domains of $\mathcal{T}$ in order without reaching any other discrete domain, and finally reaches $h_j = (\pi_j, d_{s_j}) \in \mathcal{D}_j$. In this case, we say that τ is inside $\mathcal{T}$. For example, in Fig 2 right, the red trajectory is inside the sequence of discrete domains $(01^-, 00^+, 0^+0, 1^-0, 10^+)$.

The relation between π_i and π_j can be described by a *transition matrix* M: $\pi_j = s^{-1}(Ms(\pi_i))$, where s is a function that adds an extra dimension and the value in the extra dimension is always 1: $s((a_1, a_2, ..., a_N)) = (a_1, a_2, ..., a_N, 1)$. The transition matrix M only depends on $\mathcal{T}$. The transition $\pi_j = s^{-1}(Ms(\pi_i))$ can be reformulated by another affine application $x_j = Ax_i + b$, where x_i (resp. x_j) is the short version of π_i (resp. π_j) by only considering the dimensions where boundaries are not reached in $\mathcal{D}_i$ (resp. $\mathcal{D}_j$). The matrix A is called the *reduction matrix* of $\mathcal{T}$, b is called the *constant vector* of $\mathcal{T}$ and the vector x_i is called the *reduction vector* of h_i, which is noted by $x_i = r(h_i)$. For example, for the state $h_M = ((\pi_M^1, 1), (1,1))$ in Fig 1, $r(h_M) = (\pi_M^1)$ which is a 1-dimensional vector.

A priori, not all trajectories from $\mathcal{D}_i$ stay inside $\mathcal{T}$. The maximal subset of $\mathcal{D}_i$ from which the trajectories stay inside $\mathcal{T}$ is called the *compatible zone* of $\mathcal{T}$, noted by $\mathcal{S}$. The compatible zone can also be described by $\mathcal{S} = \{(\pi, d_{s_i}) \in \mathcal{D}_i \mid r(\pi) \in \mathcal{S}_r\}$ where $\mathcal{S}_r$ is a set of reduction vectors of states in $\mathcal{D}_i$ and $\mathcal{S}_r$ is called the *reduction compatible zone.*

3 Reachability analysis method

In this section, we firstly define the reachability problem concerned in this work.

Problem 1 (Reachability). Consider a hybrid state $h_1 = (\pi_1, d_{s_1})$ and a region $R_2 = \{(\pi, d_{s_2}) \mid \pi^i \in [a_i, b_i], i \in \{1,2,...,N\}\}$, where $a_i, b_i \in \mathbb{R}$ and $0 \leq a_i \leq b_i \leq 1, \forall i \in \{1,2,...,N\}$. Does the trajectory τ from h_1 enter the region R_2? In other words, does there exist t_0 such that $\tau(t_0) \in R_2$?

Problem 1 is illustrated in the examples of Fig 3, where the initial state of the trajectory (red arrows) is h_1 and the blue rectangle represents R_2.

The following assumptions are made in this work.

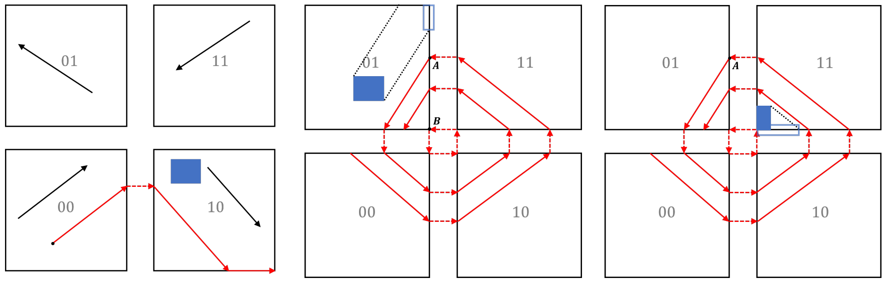


Fig. 3: Left: Illustration of Problem 1 and trajectory halting in finite time. Blue rectangle represents R_2 of Problem 1. Middle and right: Illustration of trajectories attracted by cycles of discrete domains and predecessor in the same discrete state. Blues rectangles represent R_2 of Problem 1 and blue boxes represent their predecessors in the same discrete state.

Assumption 1 *For any sequence of discrete domains $\mathcal{T}$ of which the compatible zone is not empty, we assume that all eigenvalues of the reduction matrix of $\mathcal{T}$ are real.*

For now, we have not found such reduction matrix with complex eigenvalues.

Assumption 2 *The trajectory from h_1 has no non-deterministic behavior.*

Generally, trajectories with non-deterministic behaviors exist, but among state-of-the-art HGRNs of gene regulatory networks, the probability of a randomly chosen initial state that leads to non-deterministic behaviors is almost 0. Therefore, we ignore this kind of trajectory in this work. In fact, the method of this work could also be adapted for non-deterministic trajectories (each time when a non-deterministic state is reached, the current trajectory splits into two or several trajectories, and same method is applied on each of these new trajectories).

Assumption 3 *Any non-instant transition on a limit cycle does not reach more than one new boundary at the same time.*

In real-life systems, it is indeed very unlikely for parameters to be that constrained due to the existence of noise.

3.1 Different classes of hybrid trajectories

In this section, we classify trajectories of HGRNs into three classes: trajectories halting in finite time, trajectories attracted by cycles of discrete domains and chaotic trajectories. And we provide some theoretical results regarding this reachability problem.

Trajectories halting in finite time A trajectory τ is a *trajectory halting in finite time* if $\exists t_0$ such that the derivative of $\tau(t_0)$ is 0 in any dimension, in other words $\tau(t_0)$ is a fixed point. The trajectory in Fig 3 left is a trajectory halting in finite time. We can easily see that Problem 1 is decidable if the trajectory from h_1 is a trajectory halting in finite time, because, in this case, the trajectory is a composition of a finite number of n-dimensional "straight lines"; to verify if this trajectory reaches R_2, we only need to verify if any of these "straight lines" cross R_2, which can be verified in finite time.

Trajectories attracted by cycles of discrete domains A trajectory τ is a *trajectory attracted by a cycle of discrete domains* if $\exists t_0$ such that after t_0, τ always stays inside a cycle of discrete domains $\mathcal{C}_\mathcal{T} = (\mathcal{D}_0, \mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_p, \mathcal{D}_0)$, meaning that τ crosses this cycle an infinite number of times without leaving it. Intuitively, if a trajectory τ is attracted by a cycle of discrete domains, then τ converges to or reaches a limit cycle. In Fig 3 middle and right, both trajectories are attracted by a cycle of discrete domains: indeed, these trajectories converge to the limit cycle in the center of the figure (which only has instant transitions).

To prove the decidability of trajectories attracted by cycles of discrete domains, we introduce the notion of *predecessor in the same discrete state*: for any set of hybrid states in the same discrete state defined by $R = \{(\pi, d_s) \mid \pi \in E\}$ where $E \subseteq [0,1]^N$ is a closed set, the predecessor of R in the same discrete state, noted by $Pre_{d_s}(R)$, is the union of sets of hybrid states: $Pre_{d_s}(R) = \bigcup_{i \in \{1,2,...,q\}} Z_i$, such that: 1) each Z_i belongs to a different discrete domain on an input boundary of d_s, 2) any trajectory from $Pre_{d_s}(R)$ reaches R directly ("reach R directly" means that reach R before reaching a new discrete state), 3) any trajectory from an input boundary of d_S but not from $Pre_{d_s}(R)$ does not reach directly R. For Problem 1, we can see that if the trajectory τ from h_1 has already crossed at least one discrete state (we say τ has already crossed a discrete state at t_0 if there exists $t < t_0$ such that $\tau(t_0)$ and $\tau(t)$ do not belong to the same discrete state) without reaching the region R_2, then Problem 1 is equivalent to "Does τ reach $Pre_{d_{s_2}}(R_2)$?".

Examples of predecessors in the same discrete state are illustrated in Fig 3 middle and right where blues rectangles represent R_2 and blue boxes present their predecessors in the same discrete state.

Theorem 1. *Problem 1 is decidable if the trajectory from h_1 is a trajectory attracted by a cycle of discrete domains.*

The proof of Theorem 1 is given in the Appendix. The idea of this proof can be explained intuitively by 2-dimensional examples in Fig 3 middle and right. In Fig 3 middle, the trajectory which reaches state A, noted by τ, can be considered as two trajectories: the first one is the part of τ before reaching A and the second one is the part of τ after reaching A. This first one can be considered as a trajectory halting in finite time so whether it reaches R_2 is decidable, and in this example it does not reach R_2. For the second one, these two following statements can be verified: 1. The intersection points between this trajectory and the "right" boundary of discrete state 01 must be located in the line segment AB. 2. The line segment AB does not intersect with the predecessor of R_2 in the same discrete state. Based on these two statements, we can prove that this second part cannot reach R_2 either. In this way, we prove theoretically that R_2 is not reached by τ, and since this process can be done automatically in finite time, the problem is decidable. Note that in the general case, this "line segment AB" is a $(n{-}1)$-dimensional region such that the trajectory always returns to this region and this region does not intersect the predecessor of R_2 in the same discrete state. In Fig 3 right, it can be verified automatically in finite time that the limit cycle with only instant transitions (at the center) reaches R_2, and that τ converges to this limit cycle, so we can prove that τ finally reaches R_2, and this case is thus decidable too.

We also develop the following theorem to determine if a trajectory is attracted by a cycle of discrete domains. In order to simplify this theorem, for the cycle of discrete domains $\mathcal{C_T} = (\mathcal{D}_0, \mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_p, \mathcal{D}_0)$ and the hybrid state $h_0 \in \mathcal{D}_0$ considered in this theorem, we note that:

- The reduction matrix and the constant vector of $\mathcal{C_T}$ are A and b respectively.
- The reduction compatible zone of $\mathcal{C_T}$ is described by linear constraints $\{x \mid Wx > c\}$ where c is a vector and W is a matrix. W is of size $n_0 \times n_1$, where n_1 is the number of dimensions of $r(h_0)$. W_i is the i^{th} line of matrix W (W_i is of size $1 \times n_1$) and c_i is the i^{th} component of vector c.
- $r_\infty = \lim_{n\to\infty} f^n(r(h_0))$ where $f(x) = Ax + b$.
- The eigenvalues and eigenvectors of A are $\{\lambda_i \mid i \in \{1, 2, ..., n_1\}\}$ and $\{v_i \mid i \in \{1, 2, ..., n_1\}\}$ respectively. λ_1 is chosen as the eigenvalue with the maximum absolute value among the eigenvalues that differ from 1.
- The decomposition of $r(h_0) - r_\infty$ in the directions of eigenvectors of the reduction matrix A is noted as $r(h_0) - r_\infty = \sum_{i=1}^{n_1} \alpha_i v_i$.

Theorem 2. *A trajectory τ is attracted by a cycle of discrete domains if and only if τ reaches h_0 which belongs to the compatible zone of a cycle of discrete domains $\mathcal{C_T} = (\mathcal{D}_0, \mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_p, \mathcal{D}_0)$ such that $\mathcal{D}_0$ has no free dimension (meaning that, in $\mathcal{D}_0$, boundaries are reached in all dimensions) or the following conditions are satisfied.*

- *$\mathcal{D}_0$ has at least one free dimension.*
- *$\forall i \in \{1, 2, ..., n_1\}, |\lambda_i| \leq 1 \wedge \lambda_i \neq -1$.*
- *$\forall i \in \{1, 2, ..., n_0\}$, we have either $W_i r_\infty = c_i$ or $W_i r_\infty > c_i$. We use I_e to represent the maximum set of integers such that $\forall i \in I_e, W_i r_\infty = c_i$ and we use I_n to represent the maximum set of integers such that $\forall i \in I_n, W_i r_\infty > c_i$.*
- *If $\lambda_1 \neq 0$ (we assume that λ_1 is unique if $\lambda_1 \neq 0$) and I_e is not empty, then λ_1 is positive.*
- *If $\lambda_1 \neq 0$, then $\forall i \in I_e, \forall j \in \{2, ..., n_1\}, |W_i v_1 \alpha_1| > n_1 |W_i v_j \alpha_j|$ (we ignore the case that $\exists i \in I_e, W_i v_1 = 0$).*
- *If $\lambda_1 \neq 0$, then $\forall i \in I_n, \max_{\beta\in\{-1,1\}^{n_1}} \|\sum_{j=1}^{n_1} \beta_j \alpha_j v_j\|_2 < \frac{W_i r_\infty - c_i}{\|W_i\|_2}$.*

The proof of Theorem 2 is given in the Appendix. The main idea of Theorem 2 is illustrated in Fig 4 where the huge rectangle represents a discrete domain $\mathcal{D}$ which has two free dimensions and the zone surrounded by dashed lines represents the compatible zone $\mathcal{S}$ (which is a open set) of a certain cycle of discrete domains $\mathcal{C}_{\mathcal{T}}$. Each dashed line l_{ci} represents a linear constraint of the form $w^T x > c$ where w, x are vectors and c is a real number. The fact that a trajectory τ is attracted by $\mathcal{C}_{\mathcal{T}}$ is equivalent to the fact that the intersection points between τ and $\mathcal{D}$, noted by the sequence $(h_1, h_2, ...)$, always stay inside $\mathcal{S}$ and converge to ($\lambda_1 \neq 0$) or reach ($\lambda_1 = 0$) h_∞, which belongs to the closure of $\mathcal{S}$. Need to mention that this idea of using the intersection points between a trajectory and a hyperplan to study the properties of this trajectory is based on the idea of Poincaré map. Similar ideas have been widely used in the literature to study limit cycles of other hybrid systems [5,12,29,13,11,16,20,10,22] and also have been applied to analyze the stability of limit cycles of HGRNs in [26].

Whether h_∞ belongs to the closure of $\mathcal{S}$ or not can be easily verified by using these linear constraints. A necessary condition for this sequence to always satisfy these linear constraints is that the absolute values of all eigenvalues of the reduction matrix of $\mathcal{C}_{\mathcal{T}}$ are less than or equal to 1. In case that these eigenvalues satisfy this necessary condition, to verify if this sequence always satisfies these linear constraints, we separate these constraints on two classes: the first class contains all constraints which are not reached by h_∞: l_{c2}, l_{c3}, l_{c4}, the second class contains all constraints which are reached by h_∞: l_{c1}, l_{c5}. To verify if l_{c2}, l_{c3}, l_{c4} are always satisfied, we can verify if this sequence enters and stays in a circle centered by h_∞ which only contains states satisfying constraints l_{c2}, l_{c3}, l_{c4} (this is related to the condition: if $\lambda_1 \neq 0$, then $\forall i \in I_n, \max_{\beta \in \{-1,1\}^{n_1}} \|\sum_{j=1}^{n_1} \beta_j \alpha_j v_j\|_2 < \frac{W_i r_\infty - c_i}{\|W_i\|_2}$), such circle can always be found if it is sufficiently small, for example, the circle in Fig 4. To verify if l_{c1}, l_{c5} are always satisfied, we can verify if this sequence is sufficiently "close" to v_1 which is the eigenvector related to the eigenvalue with the maximum absolute value among the eigenvalues that differ from 1 and which also "points into" $\mathcal{S}$ (this is related to the condition: if $\lambda_1 \neq 0$, then $\forall i \in I_e, \forall j \in \{2, ..., n_1\}, |W_i v_1 \alpha_1| > n_1 |W_i v_j \alpha_j|$). Here, sufficiently "close" to v_1 means intuitively that the angle between $\overrightarrow{h_\infty h_i}$ and v_1 is sufficiently small.

Chaotic trajectories In this work, a trajectory of HGRN is called a chaotic trajectory if it does not reach a fixed point and it is not attracted by a cycle of discrete domains. So all trajectories which are not included in the previous two classes are chaotic trajectories. Need to mention that the dynamics of chaotic trajectories, a priori, can be different from the chaotic dynamics of classic nonlinear dynamical systems. The reason why we still use the terminology "chaotic" is that similar concept of chaos has been used in some pre-existing works of other hybrid systems[9,18].

To prove such chaotic trajectories exist, we have constructed a HGRN with chaotic trajectories based on a pre-existing model of circuit with a chaotic attractor [18]. Parameters of this HGRN are given in the code of our work.

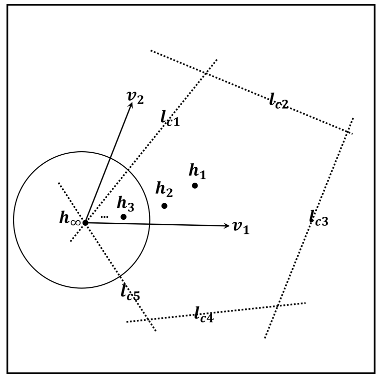


Fig. 4: Illustration of the idea of Theorem 2.

In our work, we have not yet found a method to check reachability for chaotic trajectories, which, a priori, can be undecidable. So, in this subsection, we only introduce a method to predict whether a trajectory is chaotic, based on a necessary condition.

For a chaotic trajectory τ, there exist t_0 and a finite set of discrete domains L_D, such that after t_0, τ cannot reach any discrete domain which does not belong to L_D, and for any discrete domain $\mathcal{D}_0 \in L_D$, $\mathcal{D}_0$ is reached by τ an infinite number of times. This is a result of the fact that the number of discrete domains is finite and the trajectory does not stay in a particular discrete domain.

For any $\mathcal{D}_0 \in L_D$, we can find $t_1 > t_0$ such that, from t_1, τ returns to $\mathcal{D}_0$ an infinite number of times, and each time it stays inside a sequence of discrete domains of the form $(\mathcal{D}_0, ..., \mathcal{D}_0)$. The set of all such sequences of discrete domains is noted by L_T. We assume that L_T is a finite set, which is based on the fact that the number of discrete domains is limited and the dynamics in the discrete states is simple (a constant vector). Based on this, if t_1 is sufficiently big, then we can derive that from t_1, $\forall \mathcal{T} \in L_T$ is crossed by τ an infinite number of times.

Now the sequence of discrete domains crossed by τ from t_1 can be described by the infinite sequence $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, ...)$, where $\forall i \in \mathbb{N}, \mathcal{T}_i \in L_T$. And we can get the following property of chaotic trajectories, which is used in the following section to predict whether a trajectory is chaotic.

Property 1. $\exists i \in \mathbb{N}, \exists k \in \mathbb{N}, k \neq 1$, such that $\mathcal{T}_i \neq \mathcal{T}_{i+1}$ and $\mathcal{T}_i = \mathcal{T}_{i+k}$.

Proof. This can be derived from the two facts: 1) $\exists i \in \mathbb{N}$, such that $\mathcal{T}_i \neq \mathcal{T}_{i+1}$; 2) $\forall i \in \mathbb{N}, \exists k \in \mathbb{N}$, such that $\mathcal{T}_i = \mathcal{T}_{i+k}$. The first one is a direct result of the fact that all elements of L_T must appear in the sequence $(\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, ...)$ and L_T has at least two elements. If the second one is not true, then $\mathcal{T}_i$ is crossed by τ for finite times, which contradicts with the result that $\forall \mathcal{T} \in L_T$ is crossed by τ an infinite number of times. $\square$

3.2 Reachability analysis algorithm

In this section, we present our reachability analysis algorithm, see Algorithm 1, where we call a *transition from h to h'*, noted $h \to h'$, a minimal trajectory from state h that reaches a new boundary in state h'. In other words, $h \to h'$ can be considered an atomic step of simulation, either instant (change of discrete state) or not (with continuous time elapsed).

Algorithm 1 Reachability analysis algorithm

Input 1: A hybrid state $h_1 = (\pi_1, d_{s_1})$
Input 2: A region $R_2 = \left\{(\pi, d_{s_2}) \mid \pi^i \in [a_i, b_i], i \in \{1, 2, ..., N\}\right\}$
Output: "R_2 is reached", "R_2 is not reached" or "unknown result"

1: Current state $h := h_1$
2: **while** h is not a fixed point **do**
3:     $h' :=$ next state so that $h \to h'$ is a transition
4:     **if** Transition $h \to h'$ reaches R_2 **then**
5:         **return** "R_2 is reached"
6:     **else**
7:         $h := h'$
8:         **if** Current simulation is attracted by a cycle of discrete domains **then**
9:             **if** $Stop\_condition(Cycle_h, Cycle_{\mathcal{D}}, R_2)$ returns Yes **then**
10:                 **return** "R_2 is not reached"
11:             **else if** $Stop\_condition(Cycle_h, Cycle_{\mathcal{D}}, R_2)$ returns Reached **then**
12:                 **return** "R_2 is reached"
13:             **end if**
14:         **else if** Current simulation is probably a chaotic trajectory **then**
15:             **return** "unknown result"
16:         **end if**
17:     **end if**
18: **end while**
19: **return** "R_2 is not reached"

To determine if the current simulation is attracted by a cycle of discrete domains (line 8) or if the current simulation is probably a chaotic trajectory (line 14), we use Theorem 2 or Property 1 respectively.

The objective of the function *Stop_condition* is, knowing that this trajectory is attracted by a cycle of discrete domains, to determine if the trajectory can reach R_2 after an infinite number of transitions (see Fig 3 right). If it is the case, the function returns "Reached". Otherwise, if from the current state, there is no more chance to reach R_2 (see Fig 3 middle), then the function returns "Yes". For both cases, this function can give the right answer in finite time, and the result stops the algorithm. However, if both cases do not apply, the function returns "No" and the algorithm continues. The idea of the function *Stop_condition* is similar to the proof of Theorem 1. Details about the function *Stop_condition* are given in the Appendix.

It can be proved that Algorithm 1 always stops in finite time. Firstly, if the trajectory from h_1 is a trajectory halting in finite time, then the algorithm stops after a finite number of transitions. Secondly, if the trajectory is a chaotic trajectory, then Property 1 will be satisfied after a finite number of transitions, and once it is satisfied, the algorithm stops. Thirdly, if the trajectory is attracted by a cycle of discrete domains, then there are three cases: 1. The trajectory reaches R_2 in finite time; 2. The trajectory reaches R_2 after an infinite number of transitions; 3 The trajectory does not reach R_2. We assume here that Property 1 is not satisfied before the trajectory reaching the attractive cycle of discrete domains (the cycle of discrete domains which attracts the trajectory from h_1). For case 1, the algorithm must stop in finite time, as the trajectory will eventually reach R_2. For case 2, the function *Stop_condition* returns "Reached" in finite time. For case 3, the function *Stop_condition* returns "Yes" in finite time. Need to mention that, since Property 1 is a necessary condition for that a trajectory is chaotic, the algorithm might return inconclusive results ("unknown result") even in the cases that are decidable (trajectories are non-chaotic). In fact, among HGRNs of gene regulatory networks, the cases that satisfy this necessary condition are likely to be very rare: there is no identified HGRN of a gene regulatory network with either chaos or non-chaotic trajectory that satisfies this condition. So, for now, this algorithm is sufficient for checking reachability in practice.

4 Conclusion

In this work, we propose a reachability analysis method for HGRNs. In the first part of this work, we classify trajectories of HGRNs into different classes: trajectories halting in finite time, trajectories attracted by cycles of discrete domains and chaotic trajectories, and provide some theoretical results about these trajectories regarding the reachability problem. Then, based on these theoretical results, we provide the first reachability analysis algorithm for HGRNs.

This algorithm always stops, and it returns the correct answer to the reachability problem if it does not stop with the inconclusive result ("unknown result"). In the presence of chaos, the algorithm always stops with this inconclusive result. However, so far, no model with such chaotic behavior has been identified in the model repositories we use from real-life case studies. But the fact that a HGRN with a chaotic trajectory has been identified is a motivation to investigate more.

In our future work, we will try to find other applications of this reachability analysis method and mainly focus on the development of control strategies of gene regulatory networks. Moreover, we are interested in improving the current method to analyze reachability problems in chaotic trajectories.

ADDITIONAL INFORMATION

Link to the code: `https://github.com/Honglu42/Reachability_HGRN/`. Link to the Appendix: `https://hal.science/hal-04182253`.

References

1. Alur, R., Courcoubetis, C., Henzinger, T.A., Ho, P.H.: Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. Tech. rep., Cornell University (1993)
2. Asarin, E., Maler, O., Pnueli, A.: Reachability analysis of dynamical systems having piecewise-constant derivatives. Theoretical computer science **138**(1), 35–65 (1995)
3. Asarin, E., Mysore, V.P., Pnueli, A., Schneider, G.: Low dimensional hybrid systems–decidable, undecidable, don't know. Information and Computation **211**, 138–159 (2012)
4. Behaegel, J., Comet, J.P., Bernot, G., Cornillon, E., Delaunay, F.: A hybrid model of cell cycle in mammals. Journal of bioinformatics and computational biology **14**(01), 1640001 (2016)
5. Belgacem, I., Gouzé, J.L., Edwards, R.: Control of negative feedback loops in genetic networks. In: 2020 59th IEEE Conference on Decision and Control (CDC). pp. 5098–5105. IEEE (2020)
6. Chai, X., Ribeiro, T., Magnin, M., Roux, O., Inoue, K.: Static analysis and stochastic search for reachability problem. Electronic Notes in Theoretical Computer Science **350**, 139–158 (2020)
7. Cornillon, E., Comet, J.P., Bernot, G., Enée, G.: Hybrid gene networks: a new framework and a software environment. advances in Systems and Synthetic Biology (2016)
8. Dang, T., Testylier, R.: Reachability analysis for polynomial dynamical systems using the bernstein expansion. Reliab. Comput. **17**(2), 128–152 (2012)
9. Edwards, R., Glass, L.: A calculus for relating the dynamics and structure of complex biological networks. Advances in chemical physics **132**, 151–178 (2006)
10. Edwards, R.: Analysis of continuous-time switching networks. Physica D: Nonlinear Phenomena **146**(1-4), 165–199 (2000)
11. Edwards, R., Glass, L.: A calculus for relating the dynamics and structure of complex biological networks. Adventures in Chemical Physics: A Special Volume of Advances in Chemical Physics **132**, 151–178 (2005)
12. Firippi, E., Chaves, M.: Topology-induced dynamics in a network of synthetic oscillators with piecewise affine approximation. Chaos: An Interdisciplinary Journal of Nonlinear Science **30**(11), 113128 (2020)
13. Flieller, D., Riedinger, P., Louis, J.P.: Computation and stability of limit cycles in hybrid systems. Nonlinear Analysis: Theory, Methods & Applications **64**(2), 352–367 (2006)
14. Folschette, M., Paulevé, L., Magnin, M., Roux, O.: Sufficient conditions for reachability in automata networks with priorities. Theoretical Computer Science **608**, 66–83 (2015)
15. Frehse, G., Le Guernic, C., Donzé, A., Cotton, S., Ray, R., Lebeltel, O., Ripado, R., Girard, A., Dang, T., Maler, O.: Spaceex: Scalable verification of hybrid systems. In: Computer Aided Verification: 23rd International Conference, CAV 2011, Snowbird, UT, USA, July 14-20, 2011. Proceedings 23. pp. 379–395. Springer (2011)
16. Girard, A.: Computation and stability analysis of limit cycles in piecewise linear hybrid systems. IFAC Proceedings Volumes **36**(6), 181–186 (2003)
17. Gouzé, J.L., Sari, T.: A class of piecewise linear differential equations arising in biological models. Dynamical systems **17**(4), 299–316 (2002)

18. Hamatani, S., Tsubone, T.: Analysis of a 3-dimensional piecewise-constant chaos generator without constraint. IEICE Proceedings Series **48**(A2L-B-3) (2016)
19. Henzinger, T.A., Kopke, P.W., Puri, A., Varaiya, P.: What's decidable about hybrid automata? In: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing. pp. 373–382 (1995)
20. Hiskens, I.A.: Stability of hybrid system limit cycles: Application to the compass gait biped robot. In: Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No. 01CH37228). vol. 1, pp. 774–779. IEEE (2001)
21. Maler, O., Pnueli, A.: Reachability analysis of planar multi-linear systems. In: Computer Aided Verification: 5th International Conference, CAV'93 Elounda, Greece, June 28–July 1, 1993 Proceedings 5. pp. 194–209. Springer (1993)
22. Mestl, T., Lemay, C., Glass, L.: Chaos in high-dimensional neural and gene networks. Physica D: Nonlinear Phenomena **98**(1), 33–52 (1996)
23. Paulevé, L.: Reduction of qualitative models of biological networks for transient dynamics analysis. IEEE/ACM transactions on computational biology and bioinformatics **15**(4), 1167–1179 (2017)
24. Plahte, E., Kjøglum, S.: Analysis and generic properties of gene regulatory networks with graded response functions. Physica D: Nonlinear Phenomena **201**(1-2), 150–176 (2005)
25. Sandler, A., Tveretina, O.: Deciding reachability for piecewise constant derivative systems on orientable manifolds. In: Reachability Problems: 13th International Conference, RP 2019, Brussels, Belgium, September 11–13, 2019, Proceedings 13. pp. 178–192. Springer (2019)
26. Sun, H., Folschette, M., Magnin, M.: Limit cycle analysis of a class of hybrid gene regulatory networks. In: Computational Methods in Systems Biology: 20th International Conference, CMSB 2022, Bucharest, Romania, September 14–16, 2022, Proceedings. pp. 217–236. Springer (2022)
27. Thomas, R.: Boolean formalization of genetic control circuits. Journal of theoretical biology **42**(3), 563–585 (1973)
28. Thomas, R.: Regulatory networks seen as asynchronous automata: a logical description. Journal of theoretical biology **153**(1), 1–23 (1991)
29. Znegui, W., Gritli, H., Belghith, S.: Design of an explicit expression of the poincaré map for the passive dynamic walking of the compass-gait biped model. Chaos, Solitons & Fractals **130**, 109436 (2020)