Robust Reachable Set: Accounting for Uncertainties in Linear Dynamical Systems

BINEET GHOSH, University of North Carolina at Chapel Hill
PARASARA SRIDHAR DUGGIRALA, University of North Carolina at Chapel Hill

Reachable set computation is one of the primary techniques for safety verification of linear dynamical systems. In reality the underlying dynamics have uncertainties like parameter variations or modeling uncertainties. Therefore, the reachable set computation must consider the uncertainties in the dynamics to be useful *i.e.* the computed reachable set should be over or under approximation if not exact. This paper presents a technique to compute reachable set of linear dynamical systems with uncertainties. First, we introduce a construct called support of a matrix. Using this construct, we present a set of sufficient conditions for which reachable set for uncertain linear system can be computed efficiently; and safety verification can be performed using bi-linear programming. Finally, given a linear dynamical system, we compute robust reachable set, which accounts for all possible uncertainties that can be handled by the sufficient conditions presented. Experimental evaluation on benchmarks reveal that our algorithm is computationally very efficient.

Additional Key Words and Phrases: Formal Methods, Linear Uncertain Systems, Safety Verification, Model Checking, Reachable Set Computation, Robust Reachable Set.

1 INTRODUCTION

Controllers deployed in safety critical scenarios such as autonomous vehicles, drones, or medical devices, should always adhere to safety specification. One of the widely used techniques for verification of safety specification for controllers is to compute the reachable set. Given a set of initial configurations, the reachable set represents the set of states that are visited by any one of the trajectories starting from the initial set. Typical techniques for reachable set computation assume that the underlying dynamics is known exactly and does not have any parameter variations.

A closer look at the life cycle of control design process reveals a different picture. The system model provided to the control engineer is obtained after performing system identification. Noisy measurements during the system identification might result in a different dynamics. Additionally, sometimes these models are derived from first principles and use constants such as *gravity*, *weight of a component*, etc. The control engineer designs a state feedback control assuming that the model obtained during system identification is correct. This closed loop system is then verified for safety properties. If one discovers an error in the model, or change in value of one of the constants, the computation performed for obtaining the reachable set are invalid and the designer needs to re-compute the reachable set from scratch.

This paper addresses the following question: Are there a class of uncertainties, for which, computing the reachable set while accounting for uncertainties is computationally inexpensive? If so, how to characterize such uncertainties? Our answer to the above questions is an affirmative. We present a class of uncertainties for which reachable set can be computed efficiently and safety verification can be performed using bi-linear optimization. We present sufficient conditions for which our algorithm is applicable.

Our approach uses *generalized star* [10] representation of the set of states encountered during the process of computation. Our algorithm relies on the observation that matrix multiplication is the key operation for computing reachable set using generalized stars (Lemma 2.7). In this paper,

we identify a sub-class of linear systems with uncertainties for which matrix multiplication can be performed using symbolic techniques. We illustrate this in Examples 1.1 and 1.2.

Example 1.1. Consider the discrete linear dynamical system $x^+ = Ax$ where A is $\begin{bmatrix} 1 & \alpha \\ 0 & 2 \end{bmatrix}$ and α represents either the modeling uncertainty or a parameter. Observe that $A^2 = \begin{bmatrix} 1 & 3\alpha \\ 0 & 4 \end{bmatrix}$ and does not contain any quadratic terms or higher order terms of α. Additionally, for all k, A^k does not contain any quadratic or higher order terms of α.

Example 1.2. Consider the linear dynamical system $x^+ = Ax$ where A is an $n \times n$ matrix given as $A = \begin{bmatrix} Q & R \\ \mathbf{0} & T \end{bmatrix}$ where Q ($\in \mathbb{R}^{m \times m}$) is a block matrix of size $m \times m$ (where $m < n$), R is a block matrix of size $m \times (n-m)$, $\mathbf{0}$, is a block matrix of size $(m-n) \times m$ containing only 0s, and T $\left(\in \mathbb{R}^{(n-m)\times(n-m)}\right)$ is a block matrix of size $(n-m) \times (n-m)$. Observe that A^2 is $\begin{bmatrix} Q \times Q & Q \times R + R \times T \\ \mathbf{0} & T \times T \end{bmatrix}$. If all the uncertainties are in block matrix R, then, similar to Example 1.1, A^2 does not have any nonlinear terms in $Q \times R + R \times T$. Additionally, this property holds for all A^k, $k \geq 2$. Example 1.1 is a special case of this structure.

The properties of matrices given in Example 1.1 and 1.2 are structural in nature. Consider Example 1.2; irrespective of the values of elements in matrices Q ($\in \mathbb{R}^{m \times m}$) and T $\left(\in \mathbb{R}^{(n-m)\times(n-m)}\right)$, the matrix A^k will not have any nonlinear terms. We demonstrate in Section 4 that, reachable set for such systems can be represented using bi-linear constraints. Hence, checking for safety specification can be performed using bi-linear programming. In this paper, we introduce a framework for checking if the uncertainties satisfy such structural properties. We provide sufficient conditions under which such structural properties hold and present an algorithm for safety verification using bi-linear optimization.

Using these sufficient conditions, we construct an artifact called *robust reachable set* that accounts for uncertainties. Given a dynamical system, we identify all the uncertainties for which the sufficient condition is applicable and introduce uncertainties as suggested by the user. The resulting reachable set also accounts for the effect of uncertainties over the reachable set. As a result of the sufficient conditions, the robust reachable set can be computed symbolically and is not an over-approximation. Experimental evaluation on several benchmarks demonstrates that our approach is indeed computationally feasible.

The contributions of this paper are the following:

(1) Provide a framework for inferring structural properties of matrix products.
(2) Compute *robust reachable set*, that represents the effects of model uncertainties on the reachable set, symbolically.
(3) Present an algorithm for verification of uncertain linear systems using bi-linear optimization.

2 NOTATIONS

Dynamical systems evolve in a state space. In this paper, our domain of state space is $\mathbb{R}^n$. The state of the system is denoted as x and vectors in $\mathbb{R}^n$ are denoted by v. Given a matrix $M \in \mathbb{R}^{m \times n}$,

the $(i,j)^{th}$ element is denoted as $M[i,j]$. The domain of Boolean matrices of dimension $m \times n$ are denoted with $\mathbb{B}^{m \times n}$

Definition 2.1 (Discrete Linear Dynamical Systems). Given a matrix $A \in \mathbb{R}^{n \times n}$, a discrete linear dynamical system is denoted as:

$$x^+ = Ax \tag{1}$$

Definition 2.2 (Trajectories). A trajectory of the discrete linear dynamical system, denoted as $\xi_A : \mathbb{R}^n \times \mathbb{N} \rightarrow \mathbb{R}^n$, describes the evolution of the system in time. Given an initial state $x_0 \in \mathbb{R}^n$, the trajectory is defined as

$$\xi_A(x_0, 0) = x_0. \tag{2}$$
$$\xi_A(x_0, t) = A \times \xi(x_0, t-1). \tag{3}$$

Therefore, $\xi_A(x_0, t) = A^t x_0$ where $A^t = \underbrace{A \times A \times \ldots \times A}_{\text{t-times}}$.

We drop A from the subscript of ξ when it is clear from the context.

Definition 2.3 (Reachable Set). Given a linear dynamical system $x^+ = Ax$, initial set of states Θ, and a time step $t \in \mathbb{N}$, the reachable set of states

$$RS(A, \Theta, t) = \{\ x \mid \exists x_0 \in \Theta, x = \xi_A(x_0, t)\} \tag{4}$$

Definition 2.4 (Uncertain Linear Systems and Reachable Set). An uncertain linear dynamical system is denoted as

$$x^+ = \Lambda x \tag{5}$$

where $\Lambda \subseteq \mathbb{R}^{n \times n}$. Given an initial set Θ and time step $t \in \mathbb{N}$, the reachable set of an uncertain linear system is defined as:

$$RS(\Lambda, \Theta, t) = \{\ x \mid \exists x_0 \in \Theta, \exists A \in \Lambda, x = \xi_A(x_0, t)\}. \tag{6}$$

An alternative definition is:

$$RS(\Lambda, \Theta, t) = \bigcup_{A \in \Lambda} RS(A, \Theta, t). \tag{7}$$

It is trivial to observe that Equations (6) and (7) represent the same set. Informally, the user does not know the exact dynamics, however, can determine the range of uncertainty associated with the dynamics.

Definition 2.5 (Safety Specification). An uncertain linear dynamical system $x^+ = \Lambda x$ with initial set $\Theta \in \mathbb{R}^n$ and time bound $T_b \in \mathbb{N}$ is said to be safe with respect to an unsafe set $U \subseteq \mathbb{R}^n$ if and only if

$$\forall t \in \mathbb{N},\ 0 \leq t \leq T_b,\ RS(\Lambda, \Theta, t) \cap U = \emptyset. \tag{8}$$

One of the widely used techniques for proving that safety specification is satisfied is by computing a symbolic representation of reachable set (or its over-approximation) and check if the representation overlaps with the unsafe set. In this paper, we use the symbolic representation of *generalized stars* [10].

Definition 2.6. A generalized star S is defined as a tuple $\langle c, V, P \rangle$ where $c \in \mathbb{R}^n$ is called the *center*, $V = \{v_1, v_2, \ldots, v_m\}$ where $\forall i, 1 \leq i \leq m, v_i \in \mathbb{R}^n$ are called a set of *basis vectors* (that span $\mathbb{R}^n$),

and $P : \mathbb{R}^m \rightarrow \{\top, \bot\}$ is called the *predicate*. The set of states represented by a generalized star is defined as:

$$[\![S]\!] = \{\ x \mid \exists \alpha_1, \alpha_2, \ldots, \alpha_m \text{ such that } \\ x = c + \Sigma_{i=1}^{m} \alpha_i v_i \text{ and } P(\alpha_1, \alpha_2, \ldots, \alpha_m) = \top\} \tag{9}$$

We abuse notation and use S to refer to both $[\![S]\!]$ and S.

Generalized star representation is very efficient for performing linear transformations on sets. Naturally, they are suitable for computing the representations of reachable set for linear dynamical systems.

Lemma 2.7 (Reachable Set Computation Using Generalized Stars). *Given an initial set Θ in a generalized star representation as $\langle c, V, P\rangle$, the reachable set, $RS(A, \Theta, t) = \langle c', V', P\rangle$ where $c' = A^t c$ and $V' = \{v'_1, v'_2, \ldots, v'_m\}$ and $\forall i, 1 \leq i \leq m$, $v'_i = A^t v_i$.*

Notice that, in the star representation of reachable set, the center and the basis vectors change and the predicate is same as that of initial set. Hence, this representation has the advantage computing representations of reachable set by performing only matrix-matrix multiplications and matrix-vector multiplication operations. In this paper, we propose a method to compute *robust reachable set*, the reachable set after inducing uncertainties into coefficients of a given linear system. Next we present an algorithm for performing safety verification of such uncertain systems using bi-linear optimization.

3 PRELIMINARIES

Definition 3.1 (Linear Matrix Expressions). Let, $Vars = \{y_1, y_2, \cdots, y_k\}$ denote a set of variables and $\mathsf{LE} = \{\sum_{i=1}^{k} \beta_i y_i, \beta_i \in \mathbb{R}\}$ denote the set of all linear expressions over variables in $Vars$. A matrix M is called a *Linear Matrix Expression* (LME) if $M \in \{\mathbb{R} \cup \mathsf{LE}\}^{n \times n}$.

An LME M over $Vars = \{y_1, y_2, \ldots, y_k\}$ can also be written as $M = N_0 + N_1 y_1 + \ldots + N_k y_k$ where $\forall i, 0 \leq i \leq k, N_i \in \mathbb{R}^{n \times n}$ and N_i represents the coefficients of the variable y_i. We represent the LME M as the tuple $\langle N_0, N_1, \ldots, N_k\rangle$.

If the perturbations are time varying we represent the variables as $Vars_T = \{y_1(t), y_2(t), \cdots, y_k(t)\}$. $y_i(t+1)$ denotes the valuation of (perturbation) variable y_i at time $t+1$.

Informally, the elements of the matrix M can be either real numbers or linear expressions of variables in $Vars$.

Definition 3.2 (Uncertain Linear Systems). An uncertain linear system Λ is defined as a pair (M, Γ) where M is a linear matrix expression over $Vars$ and $\Gamma : \mathbb{R}^k \rightarrow \{\top, \bot\}$ is a predicate. The set of matrices denoted by Λ are given as

$$\Lambda = \{\ A \mid \exists \gamma_1, \gamma_2, \ldots, \gamma_k, \text{ such that } \Gamma(\gamma_1, \gamma_2, \ldots, \gamma_k) = \top \\ \text{and } A = \underbrace{N_0 + N_1\gamma_1 + \ldots + N_k\gamma_k}_{M}\ \} \tag{10}$$

Informally, Λ represents the set of matrices obtained by assigning different valuations to the variables $Vars$ that satisfy the predicate Γ. While in general, Γ can be any semi-algebraic set, in this paper, we restrict Γ to represent polytopes. In this paper we restrict Γ to represent a bounded polytope.

As seen in Section 2, matrix-matrix multiplications is an important operation for computing reachable set using generalized stars. To compute reachable set of uncertain linear systems represented as LMEs, one need to compute products of two LMEs.

Definition 3.3. Given two LMEs $A = \langle N_0, N_1, \ldots, N_k \rangle$ and $B = \langle M_0, M_1, \ldots, M_k \rangle$ over the same set of variables $Vars = \{y_1, y_2, \cdots, y_k\}$, the product of two LMEs (similar to product of two linear expressions) is defined as

$$\begin{aligned} A \times B &= (N_0 + N_1 y_1 + \ldots + N_k y_k) \times \\ &(M_0 + M_1 y_1 + \ldots + M_k y_k) \\ &= N_0 M_0 + N_0 M_1 y_1 + \ldots + N_0 M_k y_k \\ &+ N_1 M_0 y_1 + N_1 M_1 y_1^2 + \ldots + N_1 M_k y_1 y_k \\ &+ \vdots \\ &+ N_k M_0 y_k + N_k M_1 y_1 y_k + \ldots + N_k M_k y_k^2. \end{aligned} \tag{11}$$

(12)

Observe that product of two LMEs can have higher degree terms. However, in this paper, we focus our attention on LMEs for which the product is closed *i.e.* does not have any non-linear terms.

Definition 3.4. LMEs $A = \langle N_0, \ldots, N_k \rangle$ and $B = \langle M_0, \ldots, M_k \rangle$ are said to be closed under multiplication if the following two conditions are satisfied:

(1) $\forall_{1 \le i \le k}, N_i \times M_i = 0$
(2) $\forall_{1 \le i,j \le k}\ (N_i \times M_j) + (N_j \times M_i) = 0$

Therefore, $A \times B = \langle L_0, \ldots, L_k \rangle$ where,

(1) $L_0 = N_0 \times M_0$.
(2) $\forall_i\ 1 \le i \le k, L_i = (N_i \times M_0) + (N_0 \times M_i)$.

If the uncertain linear system is represented as an LME, such that A^{t-1} and A are both LME and closed under multiplication, then the representation of the reachable set at time t using generalized stars will have only second degree terms. Our goal in this paper is to identify a criterion over LMEs such that their product is closed under multiplication. In this paper, we provide a sufficient condition, and show that this sufficient condition is not necessary. We introduce a notion of *support* of a matrix.

3.1 Matrix Support

Definition 3.5 (Matrix Support). Given a matrix $M \in \mathbb{R}^{m \times n}$, $\text{supp}(M) = B$ where $B \in \mathbb{B}^{m \times n}$ such that for all $i, 1 \le i \le m, 1 \le j \le n$, $B[i,j] = 0$ if and only if $M[i,j] = 0$.

Informally, the support of a matrix identifies the elements of a matrix that are zero and the elements that are non-zero.

Definition 3.6 (Block Boolean Matrix). A matrix $B \in \mathbb{B}^{n \times n}$ is said to be a block Boolean matrix, denoted as $block((r_1, c_1), (r_2, c_2))$ if and only if for all i, j where $r_1 \le i \le r_2$ and $c_1 \le j \le c_2$, $B[i,j] = 1$ and $B[i,j] = 0$ otherwise.

Figure 1 shows a pictorial representation of a Block Boolean Matrix.

Definition 3.7 (Addition and Multiplication of Boolean Matrices). Given Boolean matrices $B_1, B_2 \in \mathbb{B}^{m \times n}$, we define the addition operation on Boolean matrices as $B_1 \oplus B_2 = B_3$ where $B_3 \in \mathbb{B}^{m \times n}$ and

$$\forall i, j, 1 \le i \le m, 1 \le j \le n, B_3[i,j] = B_1[i,j] \vee B_2[i,j].$$

Given Boolean matrices $B_1 \in \mathbb{B}^{m \times k}$ and $B_2 \in \mathbb{B}^{k \times n}$, we define the multiplication operation on Boolean matrices as $B_1 \otimes B_2 = B_3$ where $B_3 \in \mathbb{B}^{m \times n}$ and

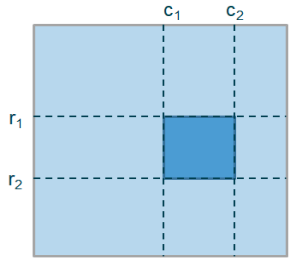

Fig. 1. Pictorial representation of a Block Boolean Matrix $block((r_1, c_1), (r_2, c_2))$. Light Blue represents 0, and Dark Blue represents 1

$$\forall i, j, 1 \leq i \leq m, 1 \leq j \leq n, B_3[i, j] = \bigvee_{l=1}^{k} B_1[i, l] \wedge B_2[l, j].$$

Informally, the addition and multiplication operations on
Boolean matrices are similar to that of matrices with real numbers. We also extend the matrix difference operation as follows.

Definition 3.8. Given Boolean matrices B and B_1, $[\![B_1]\!]B$ is the difference matrix where $[\![B_1]\!]B[i, j] = 1$ if and only if $B[i, j] = 1$ and $B_1[i, j] = 0$.

Informally, in $[\![B_1]\!]B$, we remove all the 1s from B that are also 1s in B_1. Given a set of matrices $B_1, B_2, \ldots, B_k$, $[\![B_1, B_2, \ldots, B_k]\!]B = [\![B_1 \oplus \ldots \oplus B_k]\!]B$.

Definition 3.9 (Sub-Support and Super-Support). Given Boolean matrices $B_1, B_2 \in \mathbb{B}^{m \times n}$, we say that B_1 is a *sub-support* of B_2, denoted as $B_1 \leq B_2$ if and only if for all i, j, if $B_1[i, j] = 1$ then $B_2[i, j] = 1$. An equivalent formulation is for all i, j, if $B_2[i, j] = 0$ then $B_1[i, j] = 0$. We also say that B_2 is a *super-support* of B_1.

Definition 3.10 (Intersection of Support Matrices). Given Boolean matrices $B_1, B_2 \in \mathbb{B}^{m \times n}$, we denote intersection $B_1 \cap B_2 = B$, where $B \in \mathbb{B}^{m \times n}$ and $B[i, j] = B_1[i, j] \wedge B_2[i, j]$

Properties of Supports: We now present various properties of supports of matrices.

Lemma 3.11. *Given* $M_1, M_2 \in \mathbb{R}^{m \times n}$, $\text{supp}(M_1 + M_2) \leq \text{supp}(M_1) \oplus \text{supp}(M_2)$.

Proof. Let us denote $\text{supp}(M_1 + M_2)$ as B_1 and $\text{supp}(M_1) \oplus \text{supp}(M_2)$ as B_2. Consider an i, j such that $B_2[i, j] = 0$. From Definition 3.7, it follows that $\text{supp}(M_1)[i, j] = 0$ and $\text{supp}(M_2)[i, j] = 0$. Therefore, $M_1[i, j] = 0$ and $M_2[i, j] = 0$. Therefore $(M_1 + M_2)[i, j] = 0$. Therefore $\text{supp}(M_1 + M_2)[i, j] = 0$. Therefore, for all i, j if $B_2[i, j] = 0$ then $B_1[i, j] = 0$. Therefore $B_1 \leq B_2$. □

Lemma 3.12. *Given* $M_1 \in \mathbb{R}^{m \times n}$ *and* $M_2 \in \mathbb{R}^{n \times l}$, $\text{supp}(M_1 \times M_2) \leq \text{supp}(M_1) \otimes \text{supp}(M_2)$.

Proof. Let us denote $\text{supp}(M_1 \times M_2)$ as B_1 and $\text{supp}(M_1) \otimes \text{supp}(M_2)$ as B_2. Consider an i, j such that $B_2[i, j] = 0$. From Definition 3.7, it follows that for all k, either $\text{supp}(M_1)[i, k] = 0$ or $\text{supp}(M_2)[k, j] = 0$ or both. Hence, for all k, either $M_1[i, k] = 0$ or $M_2[k, j] = 0$ or both. Therefore $(M_1 \times M_2)[i, j] = 0$. Therefore $\text{supp}(M_1 \times M_2)[i, j] = 0$, that is, $B_1[i, j] = 0$. Therefore $B_1 \leq B_2$. □

Corollary 3.13. *Given matrices* M_1 *and* M_2, *if* $\text{supp}(M_1) \otimes \text{supp}(M_2) = \boldsymbol{0}$, *then* $M_1 \times M_2 = \boldsymbol{0}$

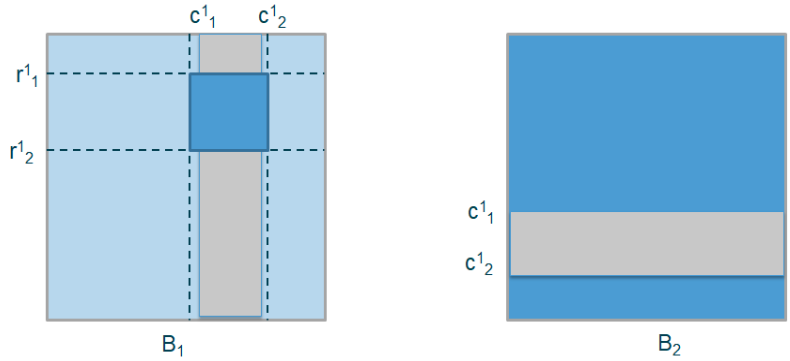

Fig. 2. If all the cells contained in the grey zone of B_2 contain 0 then $B_1 \times B_2 = 0$. Light Blue represents 0, and Dark Blue represents 1

.

Given two Block Boolean Matrices $B_1, B_2 \in \mathbb{B}^{n\times n}$, we can check if $B_1 \times B_2 = 0$ in $O(1)$. Let, $B_1 = block((r_1^1, c_1^1), (r_2^1, c_2^1))$ and $B_2 = block((r_1^2, c_1^2), (r_2^2, c_2^2))$. If $\{c_1^1, c_1^1 + 1, \cdots c_2^1\} \cap \{r_1^2, r_1^2 + 1, \cdots r_2^2\} = \emptyset$, then $B_1 \times B_2 = 0$. Checking this interval intersection operation can be performed in $O(1)$ time. The condition that elements in B_2 from rows $[c_1^1, c_2^1]$ are zeros is illustrated in Figure 2. Since matrix multiplication is not commutative, Corollary 3.13 does not a guarantee that $M_2 \times M_1 = 0$.

Lemma 3.14. *Given $B_1, B_2 \in \mathbb{B}^{m\times n}$ and $B_3 \in \mathbb{B}^{n\times l}$. If $B_1 \leq B_2$, then $B_1 \otimes B_3 \leq B_2 \otimes B_3$.*

Proof. Consider an i, j such that $(B_2 \otimes B_3)[i, j] = 0$, then for all k, either $B_2[i, k] = 0$ or $B_3[k, j] = 0$ or both. From Definition 3.7, it follows that, for all k, either $B_1[i, k] = 0$ or $B_3[k, j] = 0$ or both. Therefore $(B_1 \otimes B_3)[i, j] = 0$. Hence, $B_1 \otimes B_3 \leq B_2 \otimes B_3$. □

Corollary 3.15. *Given $B_1, B_2 \in \mathbb{B}^{n\times m}$ and $B_3 \in \mathbb{B}^{l\times n}$. If $B_1 \leq B_2$, then $B_3 \otimes B_1 \leq B_3 \otimes B_2$.*

4 SUFFICIENT CONDITIONS FOR REPRESENTING REACHABLE SET OF UNCERTAIN LINEAR SYSTEMS USING BI-LINEAR INEQUALITIES

In this section, we present sufficient conditions for product of LMEs to be closed under product (Definition 3.4). This allows us to compute the symbolic representation of the reachable set of uncertain linear systems using bi-linear inequalities.

Lemma 4.1. *Given LMEs $A = \langle N_0, N_1, \ldots, N_k\rangle$ and $B = \langle M_0, M_1, \ldots, M_k\rangle$ over the same set of variables Vars. If $\forall i, j$, $1 \leq i \leq k$ and $1 \leq j \leq k$, $\text{supp}(N_i) \otimes \text{supp}(M_j) = \mathbf{0}$, then $A \times B$ results in an LME.*

Proof. From Corollary 3.13, if for all i, j, $\text{supp}(N_i) \otimes \text{supp}(M_j) = \mathbf{0}$, then for all i, $N_i \times M_i = \mathbf{0}$ and for all i, j, with $i \neq j$, $N_i \times M_j = \mathbf{0}$ and $N_j \times M_i = \mathbf{0}$. Therefore $N_i \times M_j + N_j \times M_i = 0$.

Therefore, both the conditions specified in Definition 3.4 for closure of products over LMEs are satisfied. Hence $A \times B$ results in an LME. □

Lemma 4.1 provides a sufficient condition for product of two LMEs to result in an LME. We now present sufficient conditions for A^k to be an LME.

Theorem 4.2. *Given an LME $A = \langle N_0, N_1, \ldots, N_k \rangle$ if*

$$\forall i, j, 1 \leq i, j \leq k, \text{supp}(N_i) \otimes \text{supp}(N_j) = \boldsymbol{0} \tag{13}$$

$$\forall i, 0 \leq i \leq k, \text{supp}(N_0) \otimes \text{supp}(N_i) \leq \text{supp}(N_i),$$
$$\textit{and } \text{supp}(N_i) \otimes \text{supp}(N_0) \leq \text{supp}(N_i). \tag{14}$$

then for all $m \geq 2$, A^m is an LME.

Proof. The proof is by induction. We strengthen the inductive hypothesis to prove the above property.

Inductive Hypothesis: Given an LME A that satisfies Equations (13) and (14), for all $m \geq 2$, A^m is an LME $\langle L_0^m, L_1^m, \ldots, L_k^m \rangle$, and additionally, for all $i, 0 \leq i \leq k$, $\text{supp}(L_i^m) \leq \text{supp}(N_i)$.

Base Case (m = 2): Given Equations 13 and 14, using Lemma 4.1, we know that A^2 is an LME. Let us denote $A^2 = \langle L_0^2, L_1^2, \ldots, L_k^2 \rangle$.

Consider L_i^2 where $i > 0$. From Definition 3.4, we know that $L_i^2 = N_0 \times N_i + N_i \times N_0$. Therefore

$$\begin{aligned}
\text{supp}(L_i^2) &= \text{supp}(N_0 \times N_i + N_i \times N_0) \\
&\leq \text{supp}(N_0 \times N_i) \oplus \text{supp}(N_i \times N_0) \\
&\quad \text{(from Lemma 3.11)} \\
&\leq \big(\text{supp}(N_0) \otimes \text{supp}(N_i)\big) \oplus \big(\text{supp}(N_i) \otimes \text{supp}(N_0)\big) \\
&\quad \text{(from Lemma 3.12)} \\
&\leq \text{supp}(N_i) \oplus \text{supp}(N_i) \text{ (from Condition 14)} \\
&\leq \text{supp}(N_i)
\end{aligned}$$

Hence, $\forall i > 0$ $\text{supp}(L_i^2) \leq \text{supp}(N_i)$.

The reasoning for $\text{supp}(L_0^2) \leq \text{supp}(N_0)$ follows similarly.

Induction Step: Suppose that for an $m > 2$ the induction hypothesis is satisfied. We now prove that the inductive hypothesis holds for $m + 1$.

Let us denote A^m as $\langle L_0^m, L_1^m, \ldots, L_k^m \rangle$. From inductive hypothesis, we know that, for all i, $0 \leq i \leq k$, $\text{supp}(L_i^m) \leq \text{supp}(N_i)$. Therefore, from Equation (13) and Corollary 3.13, we know that $\forall i, j$ $i > 0, j > 0$, $N_j \times L_i^m = \mathbf{0}$. Therefore, from Definition 3.4, $A \times A^m$ results in an LME. Let us denote $A^{m+1} = \langle L_0^{m+1}, L_1^{m+1}, \ldots, L_k^{m+1} \rangle$.

To prove the inductive hypothesis, consider L_i^{m+1} where $i > 0$. From Definition 3.4, we know that $L_i^{m+1} = N_0 \times L_i^m + N_i \times L_0^m$. Therefore

$$\begin{aligned}
\text{supp}(L_1^{m+1}) &= \text{supp}(N_0 \times L_i^m + N_i \times L_0^m) \\
&\leq \text{supp}(N_0 \times L_i^m) \oplus \text{supp}(N_i \times L_0^m) \\
&\quad \text{(from Lemma 3.11)} \\
&\leq \big(\text{supp}(N_0) \otimes \text{supp}(L_i^m)\big) \oplus \\
&\quad \big(\text{supp}(N_i) \otimes \text{supp}(L_0^m)\big) \text{ (from Lemma 3.12)} \\
&\leq \big(\text{supp}(N_0) \otimes \text{supp}(N_i)\big) \oplus \\
&\quad \big(\text{supp}(N_i) \otimes \text{supp}(N_0)\big) \text{ (from Induction)} \\
&\leq \text{supp}(N_i) \oplus \text{supp}(N_i) \text{ from Condition 14.} \\
&\leq \text{supp}(N_i)
\end{aligned}$$

Hence, $\forall i > 0$ $\text{supp}(L_i^{m+1}) \leq \text{supp}(N_i)$. The reasoning for $\text{supp}(L_0^{m+1}) \leq \text{supp}(N_0)$ follows similarly. □

Discussion: Theorem 4.2 provides a sufficient condition for A^k to be an LME for all $k \geq 2$ using the notion of supports. In our opinion, support is an abstract domain over matrices. This abstract domain is very useful for inferring properties of LMEs, such as closure under multiplication. In future, we would like to explore similar abstract domains and explore more properties of LMEs.

THEOREM 4.3. *Let $\Lambda = (A, \Gamma)$ be an uncertain system where $A = \langle N_0, N_1, \ldots, N_k \rangle$ is an LME satisfying Equations (13) and (14), and $\Gamma : \mathbb{R}^k \to \{\top, \bot\}$. Since A satisfies Equations (13) and (14), from Theorem 4.2, it follows that, given t, A^t is also an LME, represented as $\langle L_0, L_1, \ldots, L_k \rangle$. Given an initial set $\Theta = \langle c, V, P \rangle$; the reachable set is given as:*

$$
\begin{aligned}
RS(\Lambda, \Theta, t) \quad = \quad & \{\, x \mid \exists y_1, \ldots, \exists y_k, \exists \alpha_1, \ldots, \exists \alpha_m, && (15)\\
& x = c' + \Sigma_{i=1}^{m} \alpha_i v_i' && (16)\\
& c' = (L_0 + L_1 y_1 + \ldots L_k y_k) \times c,\\
& \forall i, 1 \leq i \leq m, v_i' = (L_0 + L_1 y_1 + \ldots L_k y_k) v_i,\\
& \textit{and } \Gamma(y_1, \ldots, y_k) = \top \wedge P(\alpha_1, \ldots, \alpha_m) = \top \}
\end{aligned}
$$

PROOF. Consider an $M \in \Lambda$, therefore, $M = N_0 + N_1\gamma_1 + \ldots + N_k\gamma_k$ where $\Gamma(\gamma_1, \ldots, \gamma_k) = \top$. Since A^t is also an LME, we have $M^t = L_0 + L_1\gamma_1 + \ldots + L_k\gamma_k$. Hence, $RS(M, \Theta, t)$ is given as

$$
\begin{aligned}
RS(M, \Theta, t) \quad = \quad & \{\, x \mid \exists \alpha_1, \ldots, \exists \alpha_m, x = c' + \Sigma_{i=1}^{m} \alpha_i v_i' && (17)\\
& c' = (L_0 + L_1\gamma_1 + \ldots + L_k\gamma_k) \times c,\\
& \forall 1 \leq i \leq m, v_i' = (L_0 + L_1\gamma_1 + \ldots + L_k\gamma_k) v_i,\\
& \text{and } P(\alpha_1, \ldots, \alpha_m) = \top \}
\end{aligned}
$$

Since $RS(\Lambda, \Theta, k) = \bigcup_{M \in \Lambda} RS(M, \Theta, t)$. Hence, adding existential quantifiers for Equation (17) will yield Equation (16). □

Observe that $RS(\Lambda, \Theta, t)$ can be formulated as a semi-algebraic set with bi-linear constraints. The safety verification algorithm for uncertain linear systems first checks if Conditions 13 and 14 are satisfied by the uncertain linear system. If the conditions are satisfied, then A^k is computed as an LME. For checking $RS(\Lambda, \Theta, t) \cap U$ where the unsafe set U is given as a polyhedral or quadratic constraints, one can employ quadratically constrained quadratic programming tools [13, 16].

Algorithm 1 for performing safety verification of uncertain linear systems relies on Theorem 4.3. Lines 1-2 in the algorithm checks if the condition 13 and 14 are satisfied. At line 3, A^t is calculated. From Theorem 4.2 we know that A^t is also an LME as it satisfies conditions condition 13 and 14. At line 4 the reachable set is computed. Finally, lines 5-7 checks if the reachable set intersects with the unsafe set. This check can be formulated as a bi-linear programming. An example of an uncertain linear system and formulation of bi-liniear inequalities is given in Section 5.1

There are several advantages of computing the reachable set of an uncertain linear system in this manner. First, this representation is exact and is not an over-approximation. Second, computing this symbolic representation requires only matrix-matrix multiplications. Standard libraries that perform these computations are very efficient. Third, one need not recompute the reachable set for changing the uncertainty. One can simply change the set of constraints Γ and re-use the results of reachable set computation. Finally, this technique can provide counterexamples. That is, if the safety property is violated, one can diagnose the problem and provide the values corresponding to the uncertainty that lead to the safety violation. This ability to generate counterexamples is generally not possible while computing over-approximations.

input : Uncertain linear system $\Lambda = (A, \Gamma)$ where $A = \langle N_0, N_1, \ldots, N_k \rangle$; Initial set $\Theta = \langle c, V, P \rangle$; Time step t; Unsafe Set U
output : Safe or Unsafe or Invalid

1 **if** *Condition 13 or 14 is not satisfied by A* **then**
2 | **return** Invalid;
3 Compute A^t as $\langle L_0, L_1, \ldots, L_k \rangle$;
4 Compute $RS(\Lambda, \Theta, t)$ according to Equation (16);
5 **if** $RS(\Lambda, \Theta, t) \cap U \neq \emptyset$ **then**
6 | **return** Unsafe;
7 **else**
8 | **return** Safe;

Algorithm 1: Algorithm for performing safety verification of uncertain linear system when A satisfies Conditions 13 and 14.

Example 4.4. Conditions in Equations (13) and (14) are only sufficient conditions over an LME A such that A^k is also an LME. We provide an example to demonstrate that these conditions are not necessary.

Consider $A = \begin{bmatrix} 1+x & x \\ -x & 1-x \end{bmatrix}$. That is, $A = N_0 + N_1 x$ where $N_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ and $N_1 = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}$. It is easy to infer that not only A^2 is also an LME but for all $t > 2, A^t$ is an LME. Furthermore, $\text{supp}(N_1) \otimes \text{supp}(N_1) \neq \mathbf{0}$ and hence Condition 13 is not satisfied.

Observe that $A^t = N_0 + tN_1x$. Hence, even small perturbations over the model can grow over time and significantly change the reachable set.

Note, if a given linear dynamics is already known to satisfy conditions 13 and 14, one need not check for the conditions (lines 1-2, Algorithm 1). Computing the power (line 3, Algorithm 1) will automatically preserve the required structural properties (*i.e.* we can execute lines 3-8 by skipping 1-2 of Algorithm 1). This paper have introduced the notion of structures for two reasons. First, to understand the general principle behind the empirical observations, and second, to determine the instances to introduce uncertainties while ensuring that the robust reachable set can still be represented using quadratic constraints.

4.1 Generalizing Sufficient Conditions for Closure of LME products

Conditions 13 and 14 are restrictive and require
$\text{supp}(N_0) \otimes \text{supp}(N_0) \leq \text{supp}(N_0)$. We present a less restrictive conditions where given an LME A, A^k is also an LME.

Lemma 4.5. *Given LME $A = \langle N_0, N_1, \ldots, N_k \rangle$, if $\exists E_0, \exists E_1, \ldots, \exists E_k$ where, for all $i, E_i \in \mathbb{B}^{n \times n}$ such that*

$$\forall i, \text{supp}(N_i) \leq E_i. \tag{18}$$

$$\forall i, j, 1 \leq i, j \leq k, E_i \otimes E_j = \mathbf{0}. \tag{19}$$

$$\forall i, 0 \leq i \leq k, E_0 \otimes E_i \leq E_i \text{ and } E_i \otimes E_0 \leq E_i. \tag{20}$$

Then for all $n \geq 2$, A^n is also an LME.

Proof. This proof, similar to the proof of Theorem 4.2, is a proof by induction where inductive hypothesis is strengthened.

Inductive Hypothesis: Given an LME A that satisfies Equations (18), (19) and (20), for all $m \geq 2$, A^m is an LME $\langle L_0^m, L_1^m, \ldots, L_k^m \rangle$, and additionally, for all $i, 0 \leq i \leq k$, $\text{supp}(L_i^m) \leq \text{supp}(E_i)$.

The proof for the base case and the inductive step are very similar to that of proof of Theorem 4.2. The proof is given as a part of Appendix. □

Conditions 19 and 20 are less restrictive versions of conditions 13 and 14 respectively, because of condition 18. Informally, any support matrix U_i, which is a super support of N_i and satisfies conditions 19 and 20 will also satisfy conditions 13 and 14.

According to Lemma 4.5, if conditions 18, 19 and 20 are satisfied, then 13 can be generalized to condition 19 and condition 14 can be generalized to $\forall i, 0 \leq i \leq k$, if $\text{supp}(N_0) \otimes \text{supp}(N_i) \leq \text{supp}(U_i)$, and $\text{supp}(N_i) \otimes \text{supp}(N_0) \leq \text{supp}(U_i)$

4.2 Sufficient Conditions for Time-Varying Perturbations

In this section, we extend the sufficient conditions to time-varying uncertainties. Let, the given dynamics be represented by the *LME* $A = \langle N_0, N_1, \ldots, N_k \rangle$ and the time-varying uncertainties are denoted as $Vars_T = \{y_1(t), y_2(t), \cdots, y_k(t)\}$. That is, at time $t = 1$, the LME is given as $N_0 + N_1 y_1(1) + \cdots + N_k y_k(1)$. At $t = 2$, the LME is given as $N_0 + N_1 y_1(2) + \cdots + N_k y_k(2)$.

Theorem 4.6. *Given an LME* $A = \langle N_0, N_1, \ldots, N_k \rangle$ *over time-varying variables* $Vars_T = \{y_1(t), y_2(t), \cdots, y_k(t)\}$, *if*

$$\forall i, j, 1 \leq i, j \leq k, \text{supp}(N_i) \otimes \text{supp}(N_j) = \boldsymbol{0} \tag{21}$$

$$\forall i, 0 \leq i \leq k, \text{supp}(N_0) \otimes \text{supp}(N_i) \leq \text{supp}(N_i),$$

$$\textit{and } \text{supp}(N_i) \otimes \text{supp}(N_0) \leq \text{supp}(N_i). \tag{22}$$

then for all $m \geq 2$, A^m *is an LME.*

Proof. The proof is by induction. We prove a stronger inductive property of the LMEs.

Base Case (m = 2): Due to conditions 21 and 22, A^2 will also be an *LME* (Similar proof as lemma 4.1). And the corresponding *LME* is: $N_0^2 + \big(N_0 N_1 y_1(2) + N_1 N_0 y_1(1)\big) + \big(N_0 N_2 y_2(2) + N_2 N_0 y_2(1)\big) + \cdots + \big(N_0 N_k y_k(2) + N_k N_0 y_k(1)\big)$.

Rearranging the terms we get: $N_0^2 + \big(M_1^1 y_1(1) + M_2^1 y_1(2)\big) + \big(M_1^2 y_2(1) + M_2^2 y_2(2)\big) + \cdots + \big(M_1^k y_k(1) + M_2^k y_k(2)\big)$. From Equation 22, it follows that $\forall 1 \leq i \leq k, \forall 1 \leq j \leq 2$, $\text{supp}(M_j^i) \leq \text{supp}(N_i)$.

Inductive Hypothesis: Let, the LME representation be as follows: $N_0^m + \big(O_1^1 y_1(1) + O_2^1 y_1(2) + \cdots + O_m^1 y_1(m)\big) + \big(O_1^2 y_2(1) + O_2^2 y_2(2) + \cdots + O_m^2 y_2(m)\big) + \cdots + \big(O_1^k y_k(1) + O_2^k y_k(2) + \cdots + O_m^k y_k(m)\big)$. From stronger inductive hypothesis, we also have, $\forall i, 1 \leq i \leq k, \forall j, 1 \leq j \leq m$, $\text{supp}(O_j^i) \leq \text{supp}(N_i)$ and $\text{supp}(N_0^m) \leq \text{supp}(N_0)$).

Induction Step: At $m + 1$ step, we will have the following representation: $N_0^{m+1} + N_0\big(O_1^1 y_1(1) + O_2^1 y_1(2) + \cdots + O_m^1 y_1(m)\big) + N_1 N_0^m y_1(m+1) + N_0\big(O_1^2 y_2(1) + O_2^2 y_2(2) + \cdots + O_m^2 y_2(m)\big) + \cdots + N_0\big(O_1^k y_k(1) + O_2^k y_k(2) + \cdots + O_m^k y_k(m)\big) + N_k N_0^m y_k(m+1)$ Therefore, this is also an *LME*. Additionally, $\forall i, i \leq i \leq k \forall j, 1 \leq j \leq m$ $\text{supp}(N_0 \times O_j^i) \leq \text{supp}(N_i)$ $[\text{supp}(O_j^i) \leq \text{supp}(N_i)]$; and, $\forall i, 1 \leq i \leq k$ $\text{supp}(N_i N_0^m) \leq \text{supp}(N_i)$. □

4.3 Comparing with Interval Arithmetic

In this section, we provide an example where calculating reachable set using naive interval arithmetic results in an over approximation, whereas symbolic reachable set is exact. It should also be noted, in case of interval arithmetic, if the uncertainty changes (even for a single variable), the

reachable set needs to be recomputed from scratch, whereas in case of symbolic reachable set computation no re-computation is required.

Following example illustrates the over-approximation caused by interval arithmetic in reachable set computation.

Example 4.7. Consider the matrix $A = \begin{bmatrix} 2 & 4\alpha \\ 0 & -1 \end{bmatrix}$, where $\alpha \in [2, 4]$ corresponds to a perturbation. The uncertain matrix after performing matrix multiplication using interval arithmetic will be $A^2 = \begin{bmatrix} 2 & 4[2,4] \\ 0 & -1 \end{bmatrix} \times \begin{bmatrix} 2 & 4[2,4] \\ 0 & -1 \end{bmatrix} = \begin{bmatrix} 4 & [0,24] \\ 0 & 1 \end{bmatrix}$ Using symbolic computation, it will be $A^2 = \begin{bmatrix} 4 & 4\alpha \\ 0 & 1 \end{bmatrix}$

It is easy to verify that example 4.7 satisfies the sufficient conditions 13 and 14.

5 ROBUST REACHABLE SET: INTRODUCING PERTURBATIONS IN THE LINEAR DYNAMICS

In Section 4, we presented a set of sufficient conditions for which the reachable set of uncertain system can be represented using bi-linear inequalities. In this section, we will apply these sufficient conditions to compute the *robust reachable set*. Our procedure is as follows: Given a linear dynamical system $x^+ = Ax$, we identify the set Ψ of all indices $[i, j]$ such that, the LME obtained after replacing the indices in Ψ with a variable, satisfies the conditions presented in Section 4.

Once the set of indices Ψ is identified, we construct the uncertain linear system by introducing perturbations in the numerical values in the matrix A at indices in Ψ by a value determined by the user. That is, the user would be interested to check whether the safety property is satisfied after changing the numerical value at index i, j by $\pm 5\%$ or $\pm 10\%$.

To discover the set Ψ, we search for all Block Boolean matrices $H \in \mathbb{B}^{n \times n}$ such that the generalized sufficient conditions are satisfied. Given H, we first construct a matrix U_H such that, for all matrices G where $\text{supp}(G) \leq U_H$, we have $\big(\text{supp}(G) \otimes H\big) \oplus \big(H \otimes \text{supp}(G)\big) \leq H$.

Lemma 5.1. *Given a block matrix $N = block((r_1, c_1), (r_2, c_2))$, the corresponding $U_N \in \mathbb{B}^{n \times n}$ is given as*

$$U_N[i,j] = \begin{cases} 0 & \text{if } ((i < r_1 \vee i > r_2) \wedge (r_1 \leq j \leq r_2)) \\ & \text{or } ((j < c_1 \vee j > c_2) \wedge (c_1 \leq i \leq c_2)) \\ 1 & \text{otherwise.} \end{cases}$$

Proof. This proof has two parts. First, for the above U_N, we prove that $(U_N \otimes N) \oplus (N \otimes U_N) \leq N$. Second, for any U'_N such that $U_N \leq U'_N$ and $U_N \neq U'_N$, we show that $(U'_N \otimes N) \oplus (N \otimes U'_N) \not\leq N$.

Part 1: Given the above U_N, we first prove that $U_N \otimes N \leq N$. The proof for $N \otimes U_N \leq N$ follows similarly. To prove that $U_N \otimes N \leq N$, it suffices to prove the two parts. First, $\forall j, j < c_1$ or $j > c_2$, $(U_N \otimes N)[i, j] = 0$. Second, when $c_1 \leq j \leq c_2$, and $i < r_1$ or $i > r_2$, $(U_N \otimes N)[i, j] = 0$. Consider the element at $[i, j]$ of $U_N \otimes N$.

$$(U_N \otimes N)[i,j] = \bigvee_{k=1}^{n} U_N[i,k] \wedge N[k,j].$$

1) Since N is a block matrix with corners (r_1, c_1) and (r_2, c_2), all the elements in columns less than c_1 and greater than c_2 in N are zeros. Hence, for all $j < c_1$ or $j > c_2$, $(U_N \otimes N)[i, j] = 0$.

2) Consider $c_1 \leq j \leq c_2$ and $i < r_1$ or $i > r_2$. If $k < r_1$ or $k > r_2$ then $N[k, j] = 0$ and if $r_1 \leq k \leq r_2$, $U_N[i, k] = 0$. Therefore, the disjunction $\bigvee_{k=1}^{n} U_N[i,k] \wedge N[k,j]$ will result in 0. That is, when $i < r_1$ or $i > r_2$ and $c_1 \leq j \leq c_2$, $(U_N \otimes N)[i, j] = 0$.

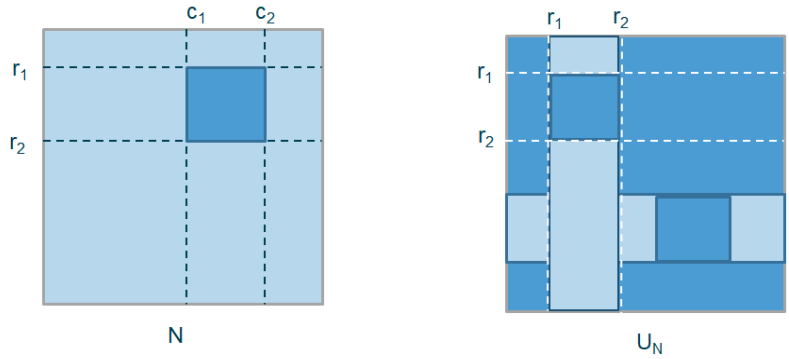

Fig. 3. Pictorial representation of a block matrix N and its corresponding U_N. Light Blue represents 0, and Dark Blue represents 1

Therefore, $U_N \otimes N \leq N$.
The proof for $N \otimes U_N \leq N$ follows similarly.

Part 2: Consider U'_N (different from U_N) such that $U_N \leq U'_N$, then there exists at least one i, j such that $U'_N[i,j] = 1$ and $U_N[i,j] = 0$. Let us consider the case where $(j < c_1 \vee j > c_2) \wedge (c_1 \leq i \leq c_2)$. For such an i, j, we have that $(U'_N \otimes N)[i,j] = \bigvee_{k=1}^{n} U'_N[i,k] \wedge N[k,j]$. By definition, it follows that $(U'_N \otimes N)[i,j] = 1$. However $N[i,j] = 0$. Therefore $U'_N \otimes N \not\leq N$. The proof for the case where $(i < r_1 \vee i > r_2) \wedge (r_1 \leq j \leq r_2)$ follows similarly. □

Figure 3 shows a pictorial view of U_N, given N.
Given a Boolean matrix N that represents the support of the system dynamics and a set of Block Boolean matrices $\{N_1, N_2, \cdots, N_k\}$ representing the support of uncertainties, we construct $[\![N_i]\!]N$ and check if $[\![N_i]\!]N \leq U_{N_i}$. If not, we can infer that introducing uncertainties at N_i might not preserve the closure under multiplication of the resulting LMEs.

Theorem 5.2. *Let, $N_1, N_2, \cdots, N_k$ be a set of Block Boolean Matrices, representing uncertainties in a given dynamics. If $\exists_{1 \leq i \leq k} N_i$, such that, it violates either of the following conditions:*

$$N_i \otimes N_j = \boldsymbol{0} \tag{23}$$

$$[\![N_i]\!]N_0 \otimes N_i \leq N_i,$$

$$\text{and } N_i \otimes [\![N_i]\!]N_0 \leq N_i. \tag{24}$$

then $[\![N_1, N_2, \cdots, N_k]\!]N_0$ with the uncertainties $N_1, \cdots, N_k$ violates either Condition 13 or 14, or both.

Proof. Case 1: $\exists_{1 \leq p,r \leq k}\ N_p \times N_r \neq 0$. Trivial.
Case 2: Let $N_i = block((r_1, c_1), (r_2, c_2))$. and it violates Condition 24 with $[\![N_i]\!]N_0$. It implies columns r_1 to r_2 excluding the rows r_1 to r_2 has a 1 in $[\![N_i]\!]N_0$, at-least in one cell; OR rows c_1 to c_2 excluding columns c_1 to c_2 has 1 in $[\![N_i]\!]N_0$, at-least in one cell (from Lemma 5.1).
Let us assume $[\![N_1, N_2, \cdots, N_k]\!]N_0$ along with the set of uncertainties $N_1, N_2, \cdots, N_i, \cdots, N_k$ satisfies Conditions 13 and 14.
To satisfy Condition 14, the 1 in columns r_1 to r_2 excluding the rows r_1 to r_2 OR rows c_1 to c_2 excluding columns c_1 to c_2 must not be present in $[\![N_1, N_2, \cdots, N_k]\!]N_0$. The only way to achieve that is, if $\exists_{j \neq i}\ 1 \leq j \leq k\ N_j$ intersects with columns r_1 to r_2 excluding the rows r_1 to r_2 OR rows c_1 to c_2 excluding columns c_1 to c_2 where there is a 1 in $[\![N_1, N_2, \cdots, N_k]\!]N_0$; in such a case $N_i \times N_j \neq 0$, violating condition 13.

Therefore, $[\![N_1, N_2, \cdots, N_k]\!]N_0$ along with the set of uncertainties $N_1, N_2, \cdots, N_i, \cdots, N_k$ does not satisfy Conditions 13 or 14 □

Corollary 5.3. *If $[\![N_1, N_2, \cdots, N_k]\!]N_0$ along with the set of uncertainties $N_1, N_2, \cdots, N_k$ satisfy Conditions 13 and 14 then $\forall_{1 \le i \le k} N_i$ will satisfy Conditions 23 and 24 with $[\![N_i]\!]N_0$*

It is easy to observe that converse of Theorem 5.2 does not hold true *i.e.* if $N_1, N_2, \cdots, N_k$ are a set of Block Boolean Matrix, representing uncertainties in a given dynamics and $\forall_i\ N_i$ satisfies Conditions 23 and 24 with $[\![N_i]\!]N_0$ then it does not imply $[\![N_1, N_2, \cdots, N_k]\!]N_0$ along with the set of uncertainties $N_1, N_2,$
$\cdots, N_k$ will satisfy Conditions 13 and 14.

We leverage Theorem 5.2 to search for block uncertainties in a given dynamics, such that Conditions 13 and 14 are satisfied. Given a block size of $p \times r$, we look for all block uncertainties N_i of size $p \times r$ such that they satisfy Conditions 23 and 24 with $[\![N_i]\!]N_0$ and put it in a set κ. The maximal subset of κ is the maximum number of uncertain blocks of size $p \times r$ that can be induced in A so that conditions 13 and 14 are satisfied. From Theorem 5.2 we know if a block N_i of size $m \times n$ is not in κ, it can never be in any of the sets of block uncertainties that satisfies condition 13 and 14.

We use Lemma 5.1 and Theorem 5.2 for introducing uncertainties in the given linear dynamics $x^+ = Ax$ such that it satisfies the sufficient conditions in Lemma 4.5. We search for all block matrices M such that $N \times N = \mathbf{0}$. Let, $[\![N]\!]N_0$ be a matrix representing the constants coefficients after excluding all the block matrices N. Then we check if $\text{supp}([\![N]\!]N_0) \le U_N$ and add them to a set κ. We consider all possible subsets of κ and check if the sufficient conditions in Lemma 4.5 are satisfied. We collect the subset with the maximum number of uncertainties that can be introduced and return the corresponding LME. The algorithm is formally given in Algorithm 2.

Description of Algorithm 2: Lines 4-6, searches for all the blocks N in the given dynamics A such that it satisfies Conditions 23 and 24 with $[\![N]\!]N_0$.
Lines 7-8: If there are no blocks found in the previous step (Lines 4-6) that satisfies Conditions 23 and 24 then, no fault introduction is possible which satisfies Conditions 13 and 14; this follows from Theorem 5.2.
Lines 9-16: We look for the largest subset of κ that satisfies Conditions 13 and 14. From Theorem 5.2 this set is guaranteed to be the largest set which satisfies Conditions 13 and 14. The conditions checked in line 13 if satisfied implies the conditions mentioned in Lemma 4.5. Therefore, the subset S satisfies all the sufficient conditions.
Line 17: Returns the maximal subset $\{N_1, N_2, \cdots, N_k\}$ that satisfies the Conditions 13 and 14.
For a particular subset S: line 12, has a run time of $O(n^2 \cdot k)$, where A is of size $n \times n$; the check in line 13 takes $O(k^2)$, as checking $H_i \times H_j\ O(1)$, and there are $O(k^2)$ of them; checks in line 14 takes $O(n^2)$.
This shows that if the uncertain blocks are given by the user *i.e.* uncertain blocks are already known, we can check if they satisfy Conditions 13 and 14 very efficiently. Moreover this check needs to be performed only once at the start. And then the safety verification can be performed using bi-linear optimization as discussed.

Observations: Notice that the Algorithm 2 searches through all possible block matrices (potentially n^4) to search for matrices that satisfy sufficient conditions and then searches for all possible subsets of the set κ (potentially 2^{n^4}). For each subset $s = \{N_1, N_2, \cdots, N_k\}$, we check if $\text{supp}(N_0) \le U_0$. Where N_0 is the coefficient matrix, excluding all N_i; $U_0 = \bigcap U_{N_i}$ and U_{N_i} are obtained according to Lemma 5.1. However, due to the restriction that $N \times N = 0$ and the condition that $\text{supp}(N_0) \le U_0$, we observe that, in practice, the set κ often contains the subsets of the order $O(n)$. Hence, we found that our approach to be useful in several benchmarks with dimensions ranging from 5 to 20.

input : Linear dynamical system $x^+ = Ax$
output : Linear Matrix Expression $A' = \langle N_0, N_1, \ldots, N_k \rangle$ that satisfies conditions in Lemma 4.5.

$\kappa \leftarrow \emptyset$;
maxPerturbations $\leftarrow 0$;
maxS $\leftarrow \emptyset$;
$N_0 \leftarrow supp(A)$;
for *all blocks* $N = block((r_1, c_1), (r_2, c_2))$ *of size less than* $n \times n$ *in the matrix* A **do**
  if $\text{supp}([\![N]\!]N_0) \leq U_N$ *and* $N \times N = \mathbf{0}$ *and* $U_N \times U_N \leq U_N$ **then**
    $\kappa \leftarrow \kappa \cup \{N\}$;
if $\kappa = \emptyset$ **then**
  return Cannot introduce uncertainties;
for *all subsets* S *of* κ **do**
  $S = \{N_1, N_2, \ldots, N_k\}$;
  $U_0 = \bigcap U_{N_i}$;
  if $\forall i, j, 1 \leq i, j \leq k, H_i \times H_j = \mathbf{0}$ **then**
    if $U_0 \times U_0 \leq U_0, \wedge\ \text{supp}(N_0) \leq U_0$ **then**
      if size(S) > maxPerturbations **then**
        maxPerturbations $\leftarrow$ size(S);
        maxS $\leftarrow S$;
return $\langle A, N_1, \ldots, N_k \rangle$ where maxS $= \{N_1, N_2, \ldots, N_k\}$.

Algorithm 2: Algorithm that searches for all possible LMEs that satisfy the sufficient conditions in Lemma 4.5 and returns the LME with maximum number of uncertainties.

5.1 Illustration

In this section we show how the experiments were performed with the help of a toy example. In the rest of the section we evaluate the applicability of our approach on several benchmarks. In each benchmark we provide all the set of inputs required to test the applicability of our approach. Let us consider the following example:

$$\begin{bmatrix} x_1^+ \\ x_2^+ \\ x_3^+ \\ x_4^+ \end{bmatrix} = \begin{bmatrix} 3 & 2.9 & 3.9 & 2 \\ 0 & 7 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

In the above example, the values in magenta (2.9) and brown (3.9) are uncertain. The different colors symbolize that these uncertainties are independent. Let, the initial set $\Theta \triangleq [1, 2] \times [2, 2] \times [3, 3] \times [1, 1]$. Suppose that the uncertainty associated with the value in magenta be $\pm 10\%$ and the value in brown be $\pm 20\%$. The system is considered to be safe if at every step the value of $x_1 < 100$. For performing safety verification of the uncertain linear system, the following steps are performed.

- A is represented as the following *LME*; the variables y and z correspond to the uncertainties.

$$\underbrace{\begin{bmatrix} 3 & 0 & 0 & 2 \\ 0 & 7 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{N_0} + \underbrace{\begin{bmatrix} 0 & 2.9 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{N_1} y + \underbrace{\begin{bmatrix} 0 & 0 & 3.9 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{N_2} z$$

- We now check the sufficient conditions in Equations (13) and (14). That is, the following conditions are checked:
  - Check $\text{supp}(N_1) \times \text{supp}(N_2) = 0$, $\text{supp}(N_2) \times \text{supp}(N_1) = 0$, $\text{supp}(N_1) \times \text{supp}(N_1) = 0$, and $\text{supp}(N_2) \times \text{supp}(N_2) = 0$.
  - Compute $U_N = U_{N_1} \cap U_{N_2}$. U_{N_1} and U_{N_2} are obtained based on Lemma 5.1. Check if $\text{supp}(N_0) \leq U_N$ and $U_N \times U_N \leq U_N$

  All the above conditions are satisfied for this example. Hence, from Theorem 4.2, we know that for all t, A^t is an *LME*.
- We now compute A^2 as

$$\begin{bmatrix} 9 & 0 & 0 & 8 \\ 0 & 49 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 29 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} y + \begin{bmatrix} 0 & 0 & 19.5 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} z$$

- For checking the safety property of $x_1 < 100$, we give the following constraints from the reachable set to Gurobi.

$$\begin{aligned} x_1 - (9\alpha_1 + 29y\alpha_2 + 19.5z\alpha_3 + 8\alpha_4) &= 0 \\ x_1 &\geq 100 \\ 1 \leq \alpha_1 \leq 2,\ 2 \leq \alpha_2 \leq 2,\ 3 \leq \alpha_3 \leq 3,\ 1 \leq \alpha_4 \leq 1 \\ 0.9 \leq y \leq 1.1,\ 0.8 \leq z \leq 1.2 \end{aligned}$$

  where α_i represents the constraints on x_i in the initial set and y and z represents the uncertainties.
- If Gurobi returns that the above set of equations are feasible, then we report that the system is unsafe. Else, the system is safe. Since most of these conditions are linear and contain only a few quadratic constraints (that too only product of variable terms), Gurobi can quickly solve such instances.

6 EVALUATION

To evaluate the applicability of our approach, we have implemented the algorithm in a python based tool that uses `numpy` and `scipy` for matrix multiplications and Gurobi engine for solving bi-linear inequalities. The implementation along with the documentation is in a public GitHub repository [1]. All the experiments were performed on a Lenovo ThinkPad Mobile Workstation with i7-8750H CPU with 2.20 GHz and 32GiB memory on Ubuntu 18.04 operating system (64 bit).
Let us consider the following toy example:

[1] https://github.com/bineet-coderep/Robust_Reach_Set_Computation

Benchmark	Dim	Fault Search Time	Discretization Parameter	Steps	Violated at Step	Faults	Verif. Res.	Time
Quadcopter (Without Faults)	16	-	0.01	-	108	-	Unsafe	0.69 s
Quadcopter (With Faults)	16	8.87 s	0.01	-	99	60	Unsafe	1.35 s
Quadcopter (Time-varying Faults)	16	8.87 s	0.01	-	99	60	Unsafe	2.15 s
Platoon (Without Faults)	10	-	0.01	2000	-	-	Safe	7.8 s
Platoon (With Faults)	10	0.49 s	0.01	-	231	9	Unsafe	1.76 s
Platoon (Time-varying Faults)	10	0.49 s	0.01	-	231	9	Unsafe	1.91 s
Anesthesia (Without Faults)	5	-	0.01	2000	-	-	Safe	3.48 s
Anesthesia (With Faults)	5	0.01 s	0.01	-	623	4	Unsafe	2.78 s
Anesthesia (Time-varying Faults)	5	0.01 s	0.01	-	623	4	Unsafe	5.02 s
Motor (Without Faults)	7	-	0.01	2000	-	-	Safe	5.04 s
Motor (With Faults)	7	0.01 s	0.01	-	19	12	Unsafe	0.08 s
Motor (Time-varying Faults)	7	0.01 s	0.01	-	19	12	Unsafe	0.10 s

Table 1. Table describing the results of verification. **Dim**: Dimensions of the system, **Search Time**: Time taken to search for all possible faults, **Discretization Parameter**: The time step to discretize the model, **Steps**: Time bound on the reachable set computation (2000), **Violated at Step**: The step at which the condition was violated, **Faults:** Number of uncertainties that could be introduced using our method, **Verif. Res.:** Verification result, **Time:** Time taken to verify of safety property using Gurobi.

For the last two benchmarks, the counter example was taking the same extreme value from the given perturbation range for both time-varying and invariant case, therefore both the cases violated at the same step. Similar behavior was observed for the other two benchmarks as well.

To evaluate the applicability of our approach, we have implemented the algorithm in a python based tool that uses `numpy` and `scipy` for matrix multiplications and Gurobi [2] engine for solving bilinear inequalities. All the experiments were performed on a Lenovo ThinkPad Mobile Workstation with i7-8750H CPU with 2.20 GHz and 32GiB memory on Ubuntu 18.04 operating system (64 bit).

We collected a set of linear dynamics benchmarks from the proceedings of ARCH workshop [3] and introduced uncertainties in the linear dynamical system and checked if the safety property is satisfied after introducing the uncertainties. Table 1 provides the time taken by our algorithm to introduce uncertainties, compute the symbolic representation of the reachable set, and check for violation of safety specification.

Flight Envelope Protection in Autonomous Quadrotors: In [18], the authors presented a linear approximation of the dynamics of a quadcopter and an open loop controller that preserves safety property. This model assumes the value of gravity to be $9.8m/sec^2$. The dynamics was discretized with a parameter $h = 0.01$. We inferred that the $block((0, 6), (6, 16))$ satisfies our generalized conditions. Note that this block contains g. We discovered that introducing an uncertainty of ±13%, caused a violation of the safety property earlier than usual. However, introducing an uncertainty of ±5% did not cause violation any earlier. Without any perturbation $\dot{x} \leq -15$ is violated at step 108, taking 0.69 seconds. When the perturbation was introduced, the condition was violated at step 99, taking 1.35 seconds. The initial set chosen for this experiment was $[3, 4] \times [3, 4] \times [3, 4] \times [3, 4] \times [3, 4] \times [3, 5] \times [-0.5, 0.5] \times [1, 1] \times [1, 1] \times [1, 1] \times [1, 1] \times [1, 1] \times [1, 1] \times [1, 1] \times [1, 1]$. With the same $block$, h and initial set, the same safety condition was checked with time-varying faults. Instead of a constant perturbation, the perturbation varied after every 6 steps. The range of perturbation was kept same as the time-invariant case. With these setup, the same safety condition was violated at step same step, 99, taking 2.15 s

Network Cooperative Platoon of Vehicles for Testing Methods and Verification Tools: In [23] the authors presented a model of vehicle platoon formation. The dynamics was discretized with a parameter of 0.01. We discovered that introducing uncertainty in $block((0, 9), (9, 10))$ had effect on the safety of the system. Without any perturbation $e_1 \leq 1$ is not violated up-to time 20, taking 7.8 seconds. But changing values from 0.01 to [0.001,0.019] caused it to violate the condition at step 231, taking 1.76 seconds. But no violation was seen at ±20%, showing high robustness of the system. This experiment was performed with an initial set of $[1, 1] \times [1, 1] \times [1, 1] \times [1, 1] \times [1, 1] \times [1, 1] \times [1, 1] \times [1, 1] \times [1, 1] \times [1, 1]$. We checked the same safety condition with time-varying perturbation as well. All the other parameters like initial set, faulty block, discretization parameters were kept same. The perturbation varied after every 20 steps. At each step, the perturbation range was same as the time invariant case. The safety condition was violated at step 231 (same step number as the time-invariant case), taking 1.91 s

PK/PD Model for Anesthesia Delivery: In [14], the authors presented continuous linear dynamics model of the anesthesia delivery for a child of weight 25kg. We discretize the model with time step 0.01 and run our algorithm that automatically generates the LMEs and introduces perturbations in the dynamics. The maximum number of uncertainties that we could introduce using our method was 4. In our experiments we discovered that introducing uncertainty in $block((0, 4), (4, 5))$ had effect in the safety of the system. We took initial set as $[1, 6] \times [0, 10] \times [0, 10] \times [1, 8] \times [1, 1]$. We verified $c_p \leq 0$ is not violated up-to time 20 without any fault, taking time 3.48 seconds. But perturbing the value 8.72×10^{-7} to $[-8.72 \times 10^{-4}, 8.72 \times 10^{-4}]$ violated the condition at step 623, taking time 2.78 seconds. No violation was observed at perturbations $[-8.72 \times 10^{-5}, 8.72 \times 10^{-5}]$. This signifies that the safety property is robust.

[2] http://www.gurobi.com/

[3] https://cps-vo.org/group/ARCH

Similarly, we checked the same safety condition, keeping all other parameters same with time varying perturbation as well. The perturbation varied after every 30 steps. The factor of variation at each step was same as the time varying one. The safety condition was violated at step 623, taking 5.02 s

Motor-Transmission Drive System: The model of transmission system presented by authors in [4] used several constants as a part of their model. The model was discretized with a time step of 0.01 and $[0, 0] \times [-0.08, 0.08] \times [-0.0165, -0.0165] \times [-0.01, 0.01]$ $[0, 0] \times [70, 70] \times [1, 1]$ was taken as the initial set. Fault was introduced at $block((0, 4), (4, 7))$, and the condition $p_x \leq -0.02$ was checked. Without any fault, this condition was not violated up-to time 20, taking time 5.04 seconds. When a perturbation of -110% to 0% was introduced, the condition violated after 19 steps taking time 0.08 seconds.
Keeping all the parameters like initial set, discretization parameter, faulty blocks, the same safety condition was checked against time-varying perturbation. The perturbation varied every 2 steps. At each step the range of perturbation was same as the time-varying case. The safety condition was violated at 19-th step taking 0.10 s.

During the experiments, we observed that by increasing the uncertainty, the given safety condition was violating at earlier steps. For some benchmarks we saw that introducing small perturbations in the model had no effect on the given safety condition. We gradually increased the uncertainty until it violates. Benchmarks which tolerated high uncertainty are considered robust. We reported verification result with both fault and without fault. When there is no uncertainty in the system, safety verification was done using linear programming, so the time taken is less. But, when uncertainties are introduced, to perform safety verification we need to solve bi-linear optimization at every step. We have reported the safety condition, the initial set, cells where uncertainty was induced, and amount of uncertainty induced. We used Algorithm 2 to find out the blocks which satisfies Conditions 13 and 14, and induced fault in those blocks.

As it can be observed from Table 1, the time taken for solving a couple of thousand bi-linear inequalities is in the order of seconds, this demonstrates that our approach is promising and can potentially scale to higher dimensions.

7 RELATED WORK

Reachable set computation of linear systems has been a well studied problem where several symbolic representation are used [6, 10–12, 15, 21]. However, most of these techniques cannot be extended easily to uncertain linear systems.

Control of linear systems with uncertainties (often called robust control) has had several distinguishing results [17, 19, 27]. However, most of these works focus on stability of the control design and not safety. One of the first works that investigated reachable set computation of uncertain systems was [2]. In this paper, the uncertainties in the continuous linear dynamics are converted into a discrete linear dynamics using taylor expansion. The reachable set for this discrete time uncertain system is computed using interval arithmetic. In [3], the same approach is extended to nonlinear systems. These nonlinear systems locally linearized and the difference between the nonlinear and local linear dynamics is accounted using Lagrange remainders.
In [7] computation of an outer approximation of the finite-time reachable sets related to a set (or family) of discrete time linear dynamical systems is discussed. A particular class of polytopes, zonotopes, is used to implicitly represent the computed sets. It extends existing algorithm based on zonotopes so that it can efficiently propagate structured parametric uncertainties.

Two other works that explicitly considered uncertain linear dynamics were [1, 22]. In [1], the uncertainties are accounted using interval matrices [20, 25] and reachable set computation is performed using zonotopes. In [22], a piecewise bilinear function approximation of the reachable

set is computed by constructing reachable sets of several linear dynamics sampled from the uncertain linear system.

There are several differences between the approaches presented in our paper and [1–3, 22]. First, our paper considers discrete dynamics systems whereas they consider continuous dynamics. Second, our paper presents a sufficient condition under which reachable set for uncertain linear systems can be represented using bilinear inequalities. This condition enables us to explore the space of uncertain systems for which reachable set can be computed without spending too much of computation resources. Third, the approach presented in this paper is the *exact* symbolic representation of reachable set and does not contain any over-approximation. Finally, since the predicate Γ in the uncertain linear system does not change, one need not recompute the reachable set representation. Instead, if the user decides to change the uncertain dynamics, she can add a new set of constraints Γ'. We have also shown that interval arithmetic that is used in [2] results in a strictly conservative over-approximation.

While one can compute the over-approximation of an uncertain or parametric linear system by formulating it as a nonlinear dynamical system and use a nonlinear dynamics reachability tool [5, 8, 9], the result would be an over approximation of the reachable set. Additionally, if the uncertainty needs to be changed, one has to recompute the reachable set. Given that it is an over-approximation, the user is still unsure whether the safety violation is because of over-approximation by the symbolic representation used in the tool or because of the uncertainty.

Some relevant recent results include [24, 26]. In both of these approaches, the reachable set of the *nominal* dynamics is computed using reachable set computation techniques. Then, the effect of the uncertainties on the reachable set is computed. In [24], the effect of uncertainties is computed using the Jacobian and in [26], this is computed using Lipschitz constant of the perturbed dynamics. Naturally, these techniques rely on numerical computations and interval arithmetic. The techniques in [24, 26] are complementary to the techniques discussed in this paper. We believe that merging these numerical techniques with the symbolic techniques presented in this paper would yield in more accurate reachable set computation for uncertain systems.

8 CONCLUSION AND FUTURE WORK

In this paper, we presented a notion of uncertain linear dynamics represented using linear matrix expressions and presented sufficient conditions under which the reachable set of uncertain system can be expressed using bi-linear inequalities. We then introduced a notion of robust reachable set and presented an algorithm for computing the robust reachable set. We also demonstrated our algorithm on several benchmarks where robust reachable set can be computed within seconds and the user can gain insights about the robustness properties of some of the constants in the model.

As a part of future work, we would like to extend this work to continuous dynamics and remove the restrictions presented in Lemma 4.5. Removing these restrictions poses interesting theoretical questions in terms of computability (is robust reachable set computable?) and complexity (complexity of safety verification). The authors' eventual goal is to develop a technique that when given as input a linear dynamical system $x^+ = Ax$ and an unsafe set U, computes a robustness matrix R corresponding to the uncertainties that can be introduced without violating the safety property.

Acknowledgements: The authors want to thank Mahesh Viswanathan for helpful discussions. The work done in this paper is based upon work supported by the National Science Foundation (NSF) under grant numbers CNS 1739936, CNS 1935724. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of NSF.

REFERENCES

[1] Matthias Althoff, Bruce H Krogh, and Olaf Stursberg. Analyzing reachability of linear dynamic systems with parametric uncertainties. In *Modeling, Design, and Simulation of Systems with Uncertainties*, pages 69–94. Springer, 2011.

[2] Matthias Althoff, Olaf Stursberg, and Martin Buss. Reachability analysis of linear systems with uncertain parameters and inputs. In *2007 46th IEEE Conference on Decision and Control*, pages 726–732. IEEE, 2007.

[3] Matthias Althoff, Olaf Stursberg, and Martin Buss. Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization. In *Proc. of the 47th IEEE Conference on Decision and Control*, 2008.

[4] Hongxu Chen, Sayan Mitra, and Guangyu Tian. Motor-transmission drive system: a benchmark example for safety verification. In *ARCH@ CPSWeek*, pages 9–18, 2014.

[5] Xin Chen, Erika Ábrahám, and Sriram Sankaranarayanan. Flow*: An analyzer for non-linear hybrid systems. In *Computer Aided Verification - 25th International Conference, CAV 2013. Proceedings*, 2013.

[6] Alongkrit Chutinan and Bruce H Krogh. Computational techniques for hybrid system verification. *IEEE transactions on automatic control*, 48(1):64–75, 2003.

[7] C. Combastel and S.A. Raka. On computing envelopes for discrete-time linear systems with affine parametric uncertainties and bounded inputs. *IFAC Proceedings Volumes*, 44(1):4525 – 4533, 2011. 18th IFAC World Congress.

[8] Tommaso Dreossi. Sapo: Reachability computation and parameter synthesis of polynomial dynamical systems. In *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*, HSCC '17, 2017.

[9] Parasara Sridhar Duggirala, Sayan Mitra, Mahesh Viswanathan, and Matthew Potok. C2e2: A verification tool for stateflow models. In *21st International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2015)*, 2015.

[10] Parasara Sridhar Duggirala and Mahesh Viswanathan. Parsimonious, simulation based verification of linear systems. In *International Conference on Computer Aided Verification*, pages 477–494. Springer, 2016.

[11] Goran Frehse. Phaver: Algorithmic verification of hybrid systems past hytech. In *International workshop on hybrid systems: computation and control*, pages 258–273. Springer, 2005.

[12] Goran Frehse, Colas Le Guernic, Alexandre Donzé, Scott Cotton, Rajarshi Ray, Olivier Lebeltel, Rodolfo Ripado, Antoine Girard, Thao Dang, and Oded Maler. Spaceex: Scalable verification of hybrid systems. In *International Conference on Computer Aided Verification*, pages 379–395. Springer, 2011.

[13] Mituhiro Fukuda and Masakazu Kojima. Branch-and-cut algorithms for the bilinear matrix inequality eigenvalue problem. *Computational Optimization and Applications*, 19(1):79–105, 2001.

[14] Victor Gan, Guy A Dumont, and Ian Mitchell. Benchmark problem: A pk/pd model and safety constraints for anesthesia delivery. In *ARCH@ CPSWeek*, pages 1–8, 2014.

[15] Antoine Girard. Reachability of uncertain linear systems using zonotopes. In *International Workshop on Hybrid Systems: Computation and Control*, pages 291–305. Springer, 2005.

[16] Keat-Choon Goh, Michael G Safonov, and George P Papavassilopoulos. Global optimization for the biaffine matrix inequality problem. *Journal of global optimization*, 7(4):365–380, 1995.

[17] Michael Green and David JN Limebeer. *Linear robust control*. Courier Corporation, 2012.

[18] Shahab Kaynama and C Tomlin. Benchmark: Flight envelope protection in autonomous quadrotors. In *Workshop on Applied Verification for Continuous and Hybrid Systems*, 2014.

[19] Pramod P Khargonekar, Ian R Petersen, and Kemin Zhou. Robust stabilization of uncertain linear systems: quadratic stabilizability and h/sup infinity/control theory. *IEEE Transactions on Automatic Control*, 35(3):356–361, 1990.

[20] Olaf Knüppel. Profil/bias—a fast interval library. *Computing*, 53(3-4):277–287, 1994.

[21] Alexander B Kurzhanski and Pravin Varaiya. Ellipsoidal techniques for reachability analysis: internal approximation. *Systems & control letters*, 41(3):201–211, 2000.

[22] Ratan Lal and Pavithra Prabhakar. Bounded error flowpipe computation of parameterized linear systems. In *Proceedings of the 12th International Conference on Embedded Software*, pages 237–246. IEEE Press, 2015.

[23] Ibtissem Ben Makhlouf and Stefan Kowalewski. Networked cooperative platoon of vehicles for testing methods and verification tools. In *ARCH@ CPSWeek*, pages 37–42, 2014.

[24] Pierre-Jean Meyer, Alex Devonport, and Murat Arcak. Tira: Toolbox for interval reachability analysis. *arXiv preprint arXiv:1902.05204*, 2019.

[25] Ramon E Moore. *Methods and applications of interval analysis*, volume 2. Siam, 1979.

[26] Mohamed Serry and Gunther Reissig. Hyper-rectangular over-approximations of reachable sets for linear uncertain systems. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 6275–6282. IEEE, 2018.

[27] Kemin Zhou, John Comstock Doyle, Keith Glover, et al. *Robust and optimal control*, volume 40. Prentice hall New Jersey, 1996.

A PROOF OF LEMMA 4.5

Proof. **Base Case (m = 2):** Given Equations 19, Lemma 3.12, Corollary 3.13, and Lemma 4.1, we know that A^2 is an LME. Let us denote $A^2 = \langle L_0^2, L_1^2, \ldots, L_k^2 \rangle$.

Consider L_i^2 where $i > 0$. From Definition 3.3, we know that $L_i^2 = (N_0 \times N_i) + (N_i \times N_0)$. Therefore

$$\begin{aligned}
\mathrm{supp}(L_i^2) &= \mathrm{supp}(N_0 \times N_i + N_i \times N_0) \\
&\leq \big(\mathrm{supp}(N_0) \otimes \mathrm{supp}(N_i)\big) \oplus \big(\mathrm{supp}(N_i) \otimes \mathrm{supp}(N_0)\big) \\
&\quad \text{(from Lemmas 3.11 and 3.12).} \\
&\leq \big(\mathrm{supp}(E_0) \otimes \mathrm{supp}(E_i)\big) \oplus \big(\mathrm{supp}(E_i) \otimes \mathrm{supp}(E_0)\big) \\
&\quad \text{(from Condition 18).} \\
&\leq \mathrm{supp}(E_i) \oplus \mathrm{supp}(E_i) \text{ (from Condition 20).} \\
&\leq \mathrm{supp}(E_i)
\end{aligned}$$

Hence, $\forall i > 0\ \mathrm{supp}(L_i^2) \leq \mathrm{supp}(E_i)$.
The reasoning for $\mathrm{supp}(L_0^2) \leq \mathrm{supp}(E_0)$ follows similarly.

Induction Step: Suppose that for an $m > 2$ the induction hypothesis is satisfied. We now prove that the inductive hypothesis holds for $m + 1$.

Let us denote A^m as $\langle L_0^m, L_1^m, \ldots, L_k^m \rangle$. From inductive hypothesis, we know that, for all i, $0 \leq i \leq k$, $\mathrm{supp}(L_i^m) \leq \mathrm{supp}(E_i)$. Therefore, from Equation 18, 19, and Corollary 3.13, we know that $\forall i, j\ i > 0, j > 0$, $N_j \times L_i^m = \mathbf{0}$. Therefore, from Definition 3.4, $A \times A^m$ results in an LME. Let us denote $A^{m+1} = \langle L_0^{m+1}, L_1^{m+1}, \ldots, L_k^{m+1} \rangle$.

To prove the inductive hypothesis, consider L_i^{m+1} where $i > 0$. From Definition 3.3, we know that $L_i^{m+1} = N_0 \times L_i^m + N_i \times L_0^m$. Therefore

$$\begin{aligned}
\mathrm{supp}(L_i^{m+1}) &= \mathrm{supp}(N_0 \times L_i^m + N_i \times L_0^m) \\
&\leq \big(\mathrm{supp}(N_0) \otimes \mathrm{supp}(L_i^m)\big) \oplus \\
&\quad \big(\mathrm{supp}(N_i) \otimes \mathrm{supp}(L_0^m)\big) \\
&\quad \text{(from Lemmas 3.12 and 3.11).} \\
&\leq \big(\mathrm{supp}(E_0) \otimes \mathrm{supp}(E_i)\big) \oplus \\
&\quad \big(\mathrm{supp}(E_i) \otimes \mathrm{supp}(E_0)\big) \text{ (from Induction).} \\
&\leq \mathrm{supp}(E_i) \oplus \mathrm{supp}(E_i) \text{ (from Condition 20).} \\
&\leq \mathrm{supp}(E_i)
\end{aligned}$$

Hence, $\forall i > 0\ \mathrm{supp}(L_i^{m+1}) \leq \mathrm{supp}(E_i)$.
The reasoning for $\mathrm{supp}(L_0^{m+1}) \leq \mathrm{supp}(E_0)$ follows similarly.

□