Scalable Forward Reachability Analysis of Multi-Agent Systems with Neural Network Controllers

Oliver Gates, Matthew Newton and Konstantinos Gatsis

Abstract— **Neural networks (NNs) have been shown to learn complex control laws successfully, often with performance advantages or decreased computational cost compared to alternative methods. Neural network controllers (NNCs) are, however, highly sensitive to disturbances and uncertainty, meaning that it can be challenging to make satisfactory robustness guarantees for systems with these controllers. This problem is exacerbated when considering multi-agent NN-controlled systems, as existing reachability methods often scale poorly for large systems. This paper addresses the problem of finding overapproximations of forward reachable sets for discrete-time uncertain multi-agent systems with distributed NNC architectures. We first reformulate the dynamics, making the system more amenable to reachablility analysis. Next, we take advantage of the distributed architecture to split the overall reachability problem into smaller problems, significantly reducing computation time. We use quadratic constraints, along with a convex representation of uncertainty in each agent's model, to form semidefinite programs, the solutions of which give overapproximations of forward reachable sets for each agent. Finally, the methodology is tested on two realistic examples: a platoon of vehicles and a power network system.**

I. INTRODUCTION

There has been recent interest in the use of neural networks (NNs) for control in closed-loop feedback systems. Neural network controllers (NNCs) can be used to imitate traditional control policies, such as model predictive control (MPC), with reduced computational cost [1], or to implement deep reinforcement learning (RL) policies [2]. Even for simple linear systems, NNCs can be used to implement complex non-linear control laws (which may not be easy to achieve with existing methods). NNs are, however, highly sensitive to input perturbations, so disturbances in the closed-loop system can have adverse effects [3]. This is problematic when NNCs are used in safety-critical systems, and recent work has focused on reachability analysis for systems with NNCs; if we can overapproximate the forward reachable sets, then it can be verified that certain regions of the state space are avoided over a given horizon.

The problem of computing forward reachable sets becomes more challenging when considering a multi-agent system in which each agent is controlled by an NN (or a series of NNs); the effects of small perturbations to the input

The authors are with the Department of Engineering Science, University of Oxford, UK. Email: oliver.gates@st-hildas.ox.ac.uk, matthew.newton@worc.ox.ac.uk, konstantinos.gatsis@eng.ox.ac.uk

of one NNC are propagated through the system. Control of multi-agent systems is well-studied, and common goals include consensus, formation control and flocking/swarming [4]. Multi-agent control architectures can be categorised according to the dependence of each agent's control input on other agents' states: (a) centralised control, in which each agent's control input is a function of all agents' states; (b) distributed control, in which each agent's control input is a function of a subset of the other agents' states; (c) decentralised control, in which each agent's control input is a function of only its own state [5]. In distributed control, common approaches include state feedback [6] and distributed model predictive control (DMPC) [7].

A number of methods have been proposed for the reachability analysis of systems with NNCs [8]–[16]. In general, the problem of reachability for discrete-time LTI systems is undecidable [17], so methods for safety verification of closed-loop systems often aim to find tight overapproximations of the forward reachable sets. Alternatively, additional restrictions can be placed on the problem to allow the exact reachable sets to be computed. Generally, there is a tradeoff between scalability and tightness of the bounds [3].

In [8], semidefinite programming (SDP) is used to compute overapproximations of the forward reachable sets, and [9] builds on this work by considering parameter-varying systems. In [10] and [11], the input set is partitioned into smaller sets, and a linear programming (LP) approach is used to overapproximate the reachable sets; in [10], the input set partitioning approach is also applied to the method in [8], and a comparison is made between the LP and SDP approaches, demonstrating that the former is faster but the latter results in tighter bounds. The work in [12] restricts the input sets to be constrained zonotopes, allowing for the exact computation of reachable sets, and the work in [13] represents the input sets as hybrid zonotopes, allowing for a class of non-convex input sets. In [14], polynomial zonotopes are used to abstract the closed-loop dynamics, providing tight overapproximations. Other approaches include the use of polynomial optimisation [15] and Bernstein polynomials [16].

Existing reachability methods for NN-controlled systems do not explicitly consider multi-agent systems with distributed control architectures. Similarly to the single-agent case, in which NNCs can be used to learn complex control laws, a distributed NNC architecture can be used to learn complex distributed control laws [18]. An example is DMPC, in which each agent's controller solves an optimisation problem based on its own state and those of its neighbours. An overview of DMPC is given in [7], and its applications

include vehicle platooning [19], frequency regulation in power systems [20] and formation control of UAVs [21].

In this paper, we present a scalable method to compute overapproximations of the forward reachable sets for uncertain multi-agent systems with distributed NNC architectures. To the best of our knowledge, this is the first work which explicitly deals with the multi-agent NNC reachability problem. The main contributions are as follows:

- we reformulate the dynamics, making the system more amenable to reachability analysis;
- we take advantage of the distributed architecture to split the overall reachability problem into smaller problems, using an existing SDP-based approach to overapproximate the forward reachable sets for each agent, and further extend this result to incorporate model uncertainty in the agents' dynamics;
- we demonstrate the effectiveness of this methodology on two realistic multi-agent systems with different structures: a platoon of vehicles and a power network;
- we compare our approach to the approach of treating the multi-agent system as one overall system, and we demonstrate that our approach outperforms the alternative approach.

In Section II, we describe the dynamics, give some remarks about the form of the controller and describe the forward reachability problem. In Section III, we provide a simplification of the dynamics and control input, and in Section IV, we present the reachability method. In Section V, we introduce model uncertainty. In Section VI, we present experiments to demonstrate the method.

A. Notation

The set of real $n \times m$ matrices is denoted by $\mathbb{R}^{n \times m}$, the set of real n-length vectors by $\mathbb{R}^n$ and the set of real numbers by $\mathbb{R}$. The set of symmetric $n \times n$ matrices is denoted by $\mathbb{S}^n$. The set of diagonal $n \times n$ matrices is denoted by $\mathbb{D}^n$. The set of positive integers is denoted by $\mathbb{Z}^+$. The cardinality of a set $\mathcal{S}$ is denoted by $|\mathcal{S}|$. The $n \times n$ identity matrix is denoted by I_n. The symbols $\geq$ and $\leq$ apply elementwise to vectors and matrices. $A \preceq 0$ implies that matrix A is negative semidefinite. The number 0 is used to represent the scalar, vector or matrix of appropriate size; the size will be clear from the context.

For clarity, whenever the letters i and j are used in this paper, they refer to the index of an agent.

II. PROBLEM STATEMENT

A. Multi-agent dynamics

We consider a discrete-time multi-agent system of M agents, where each agent i has linear time-invariant dynamics

$$x_{k+1}^{[i]} = A_{ii} x_k^{[i]} + \sum_{j \in \mathcal{N}_i} A_{ij} x_k^{[j]} + B_i u_k^{[i]} + w_k^{[i]}, \quad (1)$$

where for each agent $i \in \{1, \ldots, M\} = \mathcal{I}$, $x_k^{[i]} \in \mathbb{R}^{n_x}$ is the local state, $x_k^{[j]} \in \mathbb{R}^{n_x}$ is the jth neighbouring state, $u_k^{[i]} \in \mathbb{R}^{n_u}$ is the control input, $w_k^{[i]} \in \mathbb{R}^{n_x}$ is a known

external input, $A_{ii} \in \mathbb{R}^{n_x \times n_x}$ is the state matrix, $A_{ij} \in \mathbb{R}^{n_x \times n_x}$ is the matrix describing the effect of state $x_k^{[j]}$ on agent i, $B_i \in \mathbb{R}^{n_x \times n_u}$ is the input matrix, and $\mathcal{N}_i$ is the set of neighbours. Note that for simplicity, we have assumed that all agents have the same dimensions (although this assumption could be relaxed). We also assume that $|\mathcal{N}_i| > 0 \; \forall i$.

In this paper, without loss of generality, we focus on distributed control architectures, in which each agent's control input is a function of a subset $\mathcal{N}_i$ of all other agents' states $\mathcal{I}$. The methods presented in this paper can be easily extended to the other two cases by setting $\mathcal{N}_i = \mathcal{I}$ (centralised) or $\mathcal{N}_i = \{i\}$ (decentralised).

B. Control input

In a traditional distributed control scheme with proportional feedback, the control input $u_{\text{trad},k}^{[i]}$ might be given by

$$u_{\text{trad},k}^{[i]} = \sum_{j \in \mathcal{N}_i} K_{ij} \left(x_k^{[j]} - x_k^{[i]} \right),$$

where $K_{ij} \in \mathbb{R}^{n_u \times n_x}$ is some gain matrix (which could represent multiple gain matrices) [22]. Similarly, in a DMPC scheme, the ith control input is generated by solving an optimisation problem based on the agent's state $x_k^{[i]}$ and the neighbours' states $x_k^{[j]} \; \forall j \in \mathcal{N}_i$. In this paper, we consider the extension of the traditional distributed control schemes to NN-based control, in which the ith control input is generated by some non-linear function of the agent's state $x_k^{[i]}$ and the neighbours' states $x_k^{[j]} \; \forall j \in \mathcal{N}_i$. We also consider the possibility of controller saturation. Hence, the ith control input $u_k^{[i]}$ is given by

$$u_k^{[i]} = \text{sat}_{\mathcal{U}_i} \left[\sum_{j \in \mathcal{N}_i} \pi_{ij} \left(\begin{bmatrix} x_k^{[i]} \\ x_k^{[j]} \end{bmatrix} \right) \right], \quad (2)$$

where $\text{sat}_{\mathcal{U}_i}$ is a projection onto the set $\mathcal{U}_i = \{u \in \mathbb{R}^{n_u} \mid \underline{u}_i \leq u \leq \overline{u}_i\}$, where $\underline{u}_i$ and $\overline{u}_i$ are the lower and upper limits, respectively, for the ith controller, and $\pi_{ij} : \mathbb{R}^{2n_x} \to \mathbb{R}^{n_u}$ is a function representing the mapping of the input through a multi-layer perceptron (MLP).

Remark 1: In (2), we consider a separate NN π_{ij} for each neighbouring agent, as this preserves the ability to consider the effects of each neighbour individually, by isolating the effect of a particular neighbour's contribution to the control input. However, the control input could also be generated by feeding the state of the agent and the states of its neighbours into a single MLP $\Pi_i : \mathbb{R}^{(1+|\mathcal{N}_i|)n_x} \to \mathbb{R}^{n_u}$. A transformation between the two architectures is given in Section III-B.

C. Multi-layer perceptron

The mapping $s \mapsto \pi_{ij}(s)$ for an L-layer MLP is

$$z_{ij}^0 = s, \quad (3a)$$
$$z_{ij}^{\ell+1} = \sigma^\ell \left(W_{ij}^\ell z_{ij}^\ell + b_{ij}^\ell \right), \quad \ell = 0, \ldots, L-1, \quad (3b)$$
$$\pi_{ij}(s) = W_{ij}^L z_{ij}^L + b_{ij}^L, \quad (3c)$$

where $z_{ij}^\ell \in \mathbb{R}^{n_\ell}$ is the ℓth vector of activation values (note that $n_0 = 2n_x$), $W_{ij}^\ell \in \mathbb{R}^{n_{\ell+1} \times n_\ell}$ is the ℓth weight matrix, $b_{ij}^\ell \in \mathbb{R}^{n_{\ell+1}}$ is the ℓth bias vector, and $\sigma^\ell : \mathbb{R}^{n_{\ell+1}} \to \mathbb{R}^{n_{\ell+1}}$ is the ℓth ReLU activation function, i.e. $\sigma^\ell(y) = \max(y, 0)$, applied elementwise, such that $\sigma^\ell(y) = \begin{bmatrix} \max(y_1, 0) & \cdots & \max(y_{n_{\ell+1}}, 0) \end{bmatrix}^\top$, where $y = \begin{bmatrix} y_1 & \ldots & y_{n_{\ell+1}} \end{bmatrix}^\top$ is the vector of pre-activation values. For simplicity, we assume that all MLPs have the same structure (size and number of hidden layers) for all i, j.

D. Overapproximation of forward reachable sets

We denote the set of all possible states of the ith agent at time k as $\mathcal{X}_k^{[i]}$, such that $x_k^{[i]} \in \mathcal{X}_k^{[i]}$. Given $\mathcal{X}_k^{[i]}$ and the sets of neighbouring states at time k, i.e. $\mathcal{X}_k^{[j]} \ \forall j \in \mathcal{N}_i$, we aim to find an overapproximation $\widehat{\mathcal{X}}_{k+1}^{[i]}$ of the reachable set $\mathcal{X}_{k+1}^{[i]}$ at the next time step. Specifically, we aim to find the tightest polytopic overapproximation of the reachable set

$$\min \quad \text{vol}\left(\widehat{\mathcal{X}}_{k+1}^{[i]}\right) \tag{4a}$$

$$\text{subject to} \quad \widehat{\mathcal{X}}_{k+1}^{[i]} \supseteq \mathcal{X}_{k+1}^{[i]}, \tag{4b}$$

$$\widehat{\mathcal{X}}_{k+1}^{[i]} \text{ is a polytope,} \tag{4c}$$

for each agent i, where vol is the n_x-dimensional volume. This could then be applied recursively to overapproximate the next forward reachable sets $\mathcal{X}_{k+2}^{[i]}, \ldots, \mathcal{X}_{k+N}^{[i]}$ over a finite horizon N, and used to verify that certain unsafe regions of the state space are avoided over this horizon.

III. REFORMULATION OF DYNAMICS

An obvious approach to the forward reachability problem is to augment the agents' states into one 'overall' state

$$x_k = \begin{bmatrix} x_k^{[1]\top} & \cdots & x_k^{[M]\top} \end{bmatrix}^\top,$$

then form a recursion for the overall dynamics and use existing methods to perform reachability analysis on this system. The main issue with this approach is that it ignores the distributed architecture of the system – the dynamics of each agent depend only on its neighbours, not on all other agents. As a result, we reformulate the dynamics given by (1) and (2) to allow reachability analysis to be performed for each agent. In Section VI, we show that there is a significant computational advantage to solving M smaller reachability problems over one large reachability problem for the SDP-based method. We note that the approach of decomposing the reachability problem into smaller problems has been used more generally in work on reachability analysis [23], [24].

A. Simplification of dynamics

Let $g(i, j)$ be a function which returns the jth neighbour of the ith agent. For example, if agent 3 has $\mathcal{N}_3 = \{2, 5, 6\}$, then $g(3, 1) = 2$, $g(3, 2) = 5$ and $g(3, 3) = 6$. Then we can write $\mathcal{N}_i = \{g(i, 1), \ldots, g(i, q_i)\}$, where $q_i = |\mathcal{N}_i|$, and let

$$\tilde{A}_i = \begin{bmatrix} A_{ii} & A_{ig(i,1)} & \cdots & A_{ig(i,q_i)} \end{bmatrix}, \tag{5}$$

$$\tilde{x}_k^{[i]} = \begin{bmatrix} x_k^{[i]\top} & x_k^{[g(i,1)]\top} & \cdots & x_k^{[g(i,q_i)]\top} \end{bmatrix}^\top, \tag{6}$$

such that $\tilde{x}_k^{[i]}$ is the concatenation of the state of agent i and the states of the neighbours of agent i at time k, then (1) can be written as

$$x_{k+1}^{[i]} = \tilde{A}_i \tilde{x}_k^{[i]} + B_i u_k^{[i]} + w_k^{[i]}.$$

B. Simplification of control input

To simplify (2), we aim to represent it as the mapping of $\tilde{x}_k^{[i]}$ through a single MLP $\Pi_i : \mathbb{R}^{(1+q_i)n_x} \to \mathbb{R}^{n_u}$. Note that the layer sizes can differ from agent to agent, depending on $|\mathcal{N}_i|$. The mapping from $\tilde{x}_k^{[i]}$ to $\Pi_i(\tilde{x}_k^{[i]})$ is then

$$z_{i,k}^0 = \tilde{x}_k^{[i]}, \tag{7a}$$

$$z_{i,k}^{\ell+1} = \sigma_i^\ell \left(W_i^\ell z_{i,k}^\ell + b_i^\ell \right), \quad \ell = 0, \ldots, L-1, \tag{7b}$$

$$\Pi_i\left(\tilde{x}_k^{[i]}\right) = W_i^L z_{i,k}^L + b_i^L, \tag{7c}$$

where $z_{i,k}^\ell \in \mathbb{R}^{\tilde{n}_\ell^i}$ is the ℓth vector of activation values (note that $\tilde{n}_0^i = (1+q_i)n_x$), $\sigma_i^\ell : \mathbb{R}^{\tilde{n}_{\ell+1}^i} \to \mathbb{R}^{\tilde{n}_{\ell+1}^i}$ is the ℓth ReLU activation function, and the weight matrices and bias vectors are given by $W_i^0 = \mathfrak{W}_i^0 \Lambda_i$, $b_i^0 = \mathfrak{b}_i^0$, $W_i^\ell = \mathfrak{W}_i^\ell$, $b_i^\ell = \mathfrak{b}_i^\ell$ for $\ell = 1, \ldots, L-1$, and $W_i^L = \Omega_i \mathfrak{W}_i^L$, $b_i^L = \Omega_i \mathfrak{b}_i^L$, where

$$\mathfrak{W}_i^\ell = \text{blkdiag}\left(W_{ig(i,1)}^\ell, \ldots, W_{ig(i,q_i)}^\ell \right),$$

$$\mathfrak{b}_i^\ell = \begin{bmatrix} b_{ig(i,1)}^{\ell\top} & \cdots & b_{ig(i,q_i)}^{\ell\top} \end{bmatrix}^\top,$$

for $\ell = 0, \ldots, L$, where $\Lambda_i \in \mathbb{R}^{2q_i n_x \times \tilde{n}_0^i}$ and $\Omega_i \in \mathbb{R}^{n_u \times q_i n_u}$ are given by

$$\Lambda_i = \begin{bmatrix} I_{n_x} & 0 & 0 & 0 & \cdots & 0 \\ 0 & I_{n_x} & 0 & 0 & \cdots & 0 \\ I_{n_x} & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & I_{n_x} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ I_{n_x} & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & I_{n_x} \end{bmatrix}, \quad \Omega_i = \begin{bmatrix} I_{n_u} \\ \vdots \\ I_{n_u} \end{bmatrix}^\top.$$

Finally, we can write

$$\sum_{j \in \mathcal{N}_i} \pi_{ij}\left(\begin{bmatrix} x_k^{[i]} \\ x_k^{[j]} \end{bmatrix} \right) \equiv \Pi_i\left(\tilde{x}_k^{[i]}\right),$$
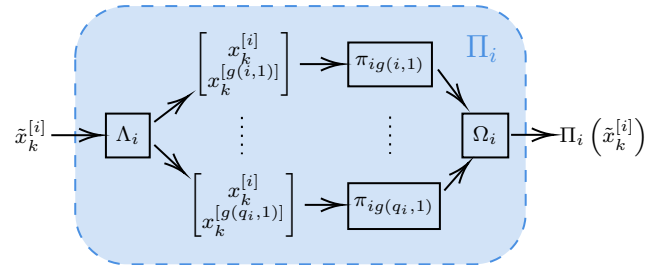
which is illustrated in Figure 1.

Figure 1. Reformulation of individual MLPs into one larger MLP

C. Summary of reformulation

Lemma 1: The dynamics given by (1) and (2) can be written as

$$x_{k+1}^{[i]} = \tilde{A}_i \tilde{x}_k^{[i]} + B_i \mathrm{sat}_{\mathcal{U}_i}\left[\Pi_i\left(\tilde{x}_k^{[i]}\right)\right] + w_k^{[i]}, \qquad (8)$$

where $\tilde{A}_i$ and $\tilde{x}_k^{[i]}$ are defined in (5) and (6), respectively, and $\Pi_i(\tilde{x}_k^{[i]})$ is given by (7a)–(7c).

IV. FORWARD REACHABILITY ANALYSIS

To overapproximate the forward reachable sets, we extend the reachability method outlined in [3] and [8] for the multi-agent case. The method in [8], called Reach-SDP, uses quadratic constraints (QCs) and semidefinite programming; the input set, NN and reachable set are abstracted using QCs. This involves forming quadratic inequalities for sets and functions by pre- and post-multiplying a matrix by a 'basis vector'. Using 'change-of-basis' matrices allows the same basis vector to be used for all inequalities, which allows linear matrix inequalities (LMIs) to be formed for forward reachability analysis of the closed-loop system.

A. Incorporation of control limits into MLP

We first add two layers to the MLP in (7a)–(7c) to account for the control limits [8]. Recall that $\tilde{x}_k^{[i]}$ is the concatenation of the state of agent i and the states of the neighbours of agent i at time k. The mapping from $\tilde{x}_k^{[i]}$ to $u_k^{[i]}$ is now defined by (7a), (7b) and

$$z_{i,k}^{L+1} = \sigma_i^L\left(W_i^L z_{i,k}^L + b_i^L - \underline{u}_i\right), \qquad (9a)$$

$$z_{i,k}^{L+2} = \sigma_i^{L+1}\left(-z_{i,k}^{L+1} + \overline{u}_i - \underline{u}_i\right), \qquad (9b)$$

$$u_k^{[i]} = -z_{i,k}^{L+2} + \overline{u}_i. \qquad (9c)$$

We then define the overall basis vector for the QCs as $\left[\mathbf{z}_{i,k}^\top \quad 1\right]^\top$, where $\mathbf{z}_{i,k}^\top = \left[z_{i,k}^{0\top} \quad z_{i,k}^{1\top} \quad \cdots \quad z_{i,k}^{L+2\top}\right]$.

B. Input set

We describe the input set $\widetilde{\mathcal{X}}_k^{[i]}$, such that $\tilde{x}_k^{[i]} \in \widetilde{\mathcal{X}}_k^{[i]}$, by a hyper-rectangle, i.e.

$$\widetilde{\mathcal{X}}_k^{[i]} = \left\{x \in \mathbb{R}^{\tilde{n}_0^i} \mid \underline{\tilde{x}}_k^{[i]} \leq x \leq \overline{\tilde{x}}_k^{[i]}\right\}, \qquad (10)$$

where $\underline{\tilde{x}}_k^{[i]}, \overline{\tilde{x}}_k^{[i]} \in \mathbb{R}^{\tilde{n}_0^i}$ are known bounds on the state of the agent and the states of its neighbours at time k. Using [3, Definition 1] and [3, Proposition 1], and noting that the first activation value $z_{i,k}^0$ is equal to $\tilde{x}_k^{[i]}$, we can write (10) as

$$\begin{bmatrix} z_{i,k}^0 \\ 1 \end{bmatrix}^\top P_k^i(\Gamma) \begin{bmatrix} z_{i,k}^0 \\ 1 \end{bmatrix} \geq 0, \qquad (11)$$

where

$$P_k^i(\Gamma) = \begin{bmatrix} -2\Gamma & \Gamma\left(\underline{\tilde{x}}_k^{[i]} + \overline{\tilde{x}}_k^{[i]}\right) \\ \left(\underline{\tilde{x}}_k^{[i]} + \overline{\tilde{x}}_k^{[i]}\right)^\top \Gamma & -2\underline{\tilde{x}}_k^{[i]\top} \Gamma \overline{\tilde{x}}_k^{[i]} \end{bmatrix},$$

where $\Gamma \in \mathbb{D}^{\tilde{n}_0^i}$ and $\Gamma \geq 0$. By using a change-of-basis matrix [8]

$$E_{\mathrm{in}}^i = \begin{bmatrix} I_{\tilde{n}_0^i} & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix},$$

we can write (11) as

$$\begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix}^\top \Delta_k^i(\Gamma) \begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix} \geq 0, \qquad (12)$$

where $\Delta_k^i(\Gamma) = E_{\mathrm{in}}^{i\top} P_k^i(\Gamma) E_{\mathrm{in}}^i$. This is summarised in the following lemma.

Lemma 2: Consider the state of agent i and the states of the neighbours of agent i at time k. If these are are bounded by a hyper-rectangular input set, i.e. $\underline{\tilde{x}}_k^{[i]} \leq \tilde{x}_k^{[i]} \leq \overline{\tilde{x}}_k^{[i]}$, as in (10), then (12) holds.

C. ReLU activation functions

First, we define $\hat{z}_{i,k}^{\ell+1} = W_i^\ell z_{i,k}^\ell + b_i^\ell$ for $\ell = 0, \ldots, L-1$, $\hat{z}_{i,k}^{L+1} = W_i^L z_{i,k}^L + b_i^L - \underline{u}_i$, and $\hat{z}_{i,k}^{L+2} = -z_{i,k}^{L+1} + \overline{u}_i - \underline{u}_i$. Then, we can write $z_{i,k}^\ell = \sigma_i^{\ell-1}(\hat{z}_{i,k}^\ell)$ for $\ell = 1, \ldots, L+2$, and write the concatenation of activation functions as

$$\boldsymbol{\nu}_{i,k} = \sigma_i\left(\hat{\boldsymbol{\nu}}_{i,k}\right), \qquad (13)$$

where $\boldsymbol{\nu}_{i,k}^\top = \begin{bmatrix} z_{i,k}^{1\top} & \cdots & z_{i,k}^{L+2\top} \end{bmatrix}$, $\hat{\boldsymbol{\nu}}_{i,k}^\top = \begin{bmatrix} \hat{z}_{i,k}^{1\top} & \cdots & \hat{z}_{i,k}^{L+2\top} \end{bmatrix}$, and $\sigma_i : \mathbb{R}^{n_n^i+2n_u} \to \mathbb{R}^{n_n^i+2n_u}$ is applied elementwise, where $n_n^i = \sum_{\ell=1}^L \tilde{n}_\ell^i$. Using [3, Definition 2] and [3, Lemma 3], we can relax (13) as

$$\begin{bmatrix} \hat{\boldsymbol{\nu}}_{i,k} \\ \boldsymbol{\nu}_{i,k} \\ 1 \end{bmatrix}^\top Q^i(\lambda, \nu, \eta) \begin{bmatrix} \hat{\boldsymbol{\nu}}_{i,k} \\ \boldsymbol{\nu}_{i,k} \\ 1 \end{bmatrix} \geq 0, \qquad (14)$$

where $Q^i(\lambda, \nu, \eta) \in \mathbb{S}^{2(n_n^i+2n_u)+1}$ is defined in [3, Lemma 3] as Q. By using a change-of-base matrix [8]

$$E_{\mathrm{mid}}^i = \left[\begin{array}{cccccc|c} W_i^0 & 0 & \cdots & 0 & 0 & 0 & b_i^0 \\ 0 & W_i^1 & \cdots & 0 & 0 & 0 & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & b_i^{L-1} \\ 0 & 0 & \cdots & W_i^L & 0 & 0 & b_i^L - \underline{u}_i \\ 0 & 0 & \cdots & 0 & -I_{n_u} & 0 & \overline{u}_i - \underline{u}_i \\ \hline 0 & I_{\tilde{n}_1^i} & \cdots & 0 & 0 & 0 & \\ \vdots & \vdots & \ddots & \vdots & \vdots & 0 & \\ 0 & 0 & \cdots & I_{\tilde{n}_L^i} & 0 & 0 & \mathbf{0} \\ 0 & 0 & \cdots & 0 & I_{n_u} & 0 & \\ 0 & 0 & \cdots & 0 & 0 & I_{n_u} & \\ \hline & & \mathbf{0} & & & & 1 \end{array}\right],$$

we can write (14) as

$$\begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix}^\top \Theta^i(\lambda, \nu, \eta) \begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix} \geq 0, \qquad (15)$$

where $\Theta^i(\lambda, \nu, \eta) = E_{\mathrm{mid}}^{i\top} Q^i(\lambda, \nu, \eta) E_{\mathrm{mid}}^i$. This is summarised in the following lemma.

Lemma 3: Given that the activation values for the extended NN satisfy $z_{i,k}^\ell = \sigma_i^{\ell-1}(\hat{z}_{i,k}^\ell)$ for $\ell = 1, \ldots, L+2$, as in (13), then (15) holds.

D. Reachable set

We parameterise the overapproximation $\widehat{\mathcal{X}}_{k+1}^{[i]}$ of the reachable set $\mathcal{X}_{k+1}^{[i]}$ using a polytope, i.e. the intersection of m halfspaces

$$\widehat{\mathcal{X}}_{k+1}^{[i]} = \left\{ x \in \mathbb{R}^{n_x} \mid H_1^\top x \le h_1, \ldots, H_m^\top x \le h_m \right\}, \quad (16)$$

where $H_1, \ldots, H_m \in \mathbb{R}^{n_x}$ and $h_1, \ldots, h_m \in \mathbb{R}$, which can be written as [8]

$$\widehat{\mathcal{X}}_{k+1}^{[i]} = \bigcap_{p=1}^m \left\{ x \in \mathbb{R}^{n_x} \mid \begin{bmatrix} x \\ 1 \end{bmatrix}^\top S_p(h_p) \begin{bmatrix} x \\ 1 \end{bmatrix} \le 0 \right\},$$

where

$$S_p(h_p) = \begin{bmatrix} 0 & H_p \\ H_p^\top & -2h_p \end{bmatrix},$$

so each halfspace can be equivalently written as

$$\begin{bmatrix} x_{k+1}^{[i]} \\ 1 \end{bmatrix}^\top S_p(h_p) \begin{bmatrix} x_{k+1}^{[i]} \\ 1 \end{bmatrix} \le 0. \quad (17)$$

By using a change-of-basis matrix

$$E_{\mathrm{out},k}^i = \begin{bmatrix} \tilde{A}_i & 0 & \cdots & 0 & -B_i & B_i \overline{u}_i + w_k^{[i]} \\ 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix},$$

we can write (17) as

$$\begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix}^\top \Psi_k^i(h_p) \begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix} \le 0, \quad (18)$$

where $\Psi_k^i(h_p) = {E_{\mathrm{out},k}^i}^\top S_p(h_p) E_{\mathrm{out},k}^i$. Note that the structure of the change-of-basis matrix differs to that in [8], as $\tilde{A}_i$ is not square, to account for the multi-agent dynamics. The result is summarised in the following lemma.

Lemma 4: If (18) holds for $p = 1, \ldots, m$, then $\widehat{\mathcal{X}}_{k+1}^{[i]}$ is a polytope defined by $H_1, \ldots, H_m$ and $h_1, \ldots, h_m$, as in (16).

E. Reachability algorithm

By combining the results in Lemmas 1–4, we arrive at the following result for the overapproximation of the forward reachable set of agent i.

Theorem 1: Consider a discrete-time LTI system of the form in (1) and (2), where the structure of each MLP is given by (3a)–(3c). At time k, let the state of agent i and the states of its neighbours be bounded by a hyper-rectangular input set, as in (10). If there exists a solution to

$$\min_{\Gamma, \lambda, \nu, \eta, h_p} \quad h_p$$
$$\text{subject to} \quad \Delta_k^i(\Gamma) + \Theta^i(\lambda, \nu, \eta) + \Psi_k^i(h_p) \preceq 0,$$

for $p = 1, \ldots, m$, where $H_1, \ldots, H_m$ are specified by the user, then the resulting polytope is the solution to (4a)–(4c).

Proof: Note that we are performing reachability analysis on the system given in (8). However, from Lemma 1, the original dynamics of the form given in (1) and (2) are equivalent to those given in (8), so performing reachability analysis on the system given in (8) is identical to performing reachability analysis on the system given in (1) and (2). The remainder of the proof is similar to that in [8, Theorem 1]. First, we pre-multiply each of the three terms in the LMI

by $\begin{bmatrix} \mathbf{z}_{i,k}^\top & 1 \end{bmatrix}$ and post-multiply each term by $\begin{bmatrix} \mathbf{z}_{i,k}^\top & 1 \end{bmatrix}^\top$, resulting the scalar inequality

$$\begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix}^\top \Delta_k^i(\Gamma) \begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix} + \begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix}^\top \Theta^i(\lambda, \nu, \eta) \begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix}$$
$$+ \begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix}^\top \Psi_k^i(h_p) \begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix} \le 0.$$

From Lemmas 2 and 3 and our definitions of the input set and MLP, we know that the first two terms are non-negative, and as the overall expression is non-positive, then the last term is non-positive, i.e.

$$\begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix}^\top \Psi_k^i(h_p) \begin{bmatrix} \mathbf{z}_{i,k} \\ 1 \end{bmatrix} \le 0.$$

Hence, from Lemma 4, we can conclude that $\widehat{\mathcal{X}}_{k+1}^{[i]}$ is a polytope defined by $H_1, \ldots, H_m$ and $h_1, \ldots, h_m$, where $\widehat{\mathcal{X}}_{k+1}^{[i]} \supseteq \mathcal{X}_{k+1}^{[i]}$. ∎

This result is implemented in Algorithm 1, which presents a method for computing hyper-rectangular output sets (a special case of the polytopic sets), given hyper-rectangular input sets. $\mathrm{Reach}(\widetilde{\mathcal{X}}_k^{[i]})$ represents the solution to the semidefinite program in Theorem 1, given $\widetilde{\mathcal{X}}_k^{[i]}$, and $\delta_{\cdot,\cdot}$ is the Kronecker delta. This algorithm can be used iteratively to find approximate reachable sets for $k+2$, $k+3$, etc. Note also that this approach is parallelisable across agents.

Algorithm 1 One-step forward reachability analysis with hyper-rectangular constraints

---

Input: input sets $\mathcal{X}_k^{[1]}, \ldots, \mathcal{X}_k^{[M]}$
1: **for** $i = 1, \ldots, M$ **do**
2:    $\widetilde{\mathcal{X}}_k^{[i]} \leftarrow \mathcal{X}_k^{[i]} \times \mathcal{X}_k^{[g(i,1)]} \times \ldots \times \mathcal{X}_k^{[g(i,q_i)]}$
3:    **for** $p = 1, \ldots, 2n_x$ **do**
4:       **if** $p \le n_x$ **then**
5:          $H_p \leftarrow \begin{bmatrix} \delta_{1,p} & \cdots & \delta_{n_x,p} \end{bmatrix}^\top$
6:       **else**
7:          $H_p \leftarrow -\begin{bmatrix} \delta_{n_x+1,p} & \cdots & \delta_{2n_x,p} \end{bmatrix}^\top$
8:       **end if**
9:       $h_{p,k+1}^{[i]} \leftarrow \mathrm{Reach}\left(\widetilde{\mathcal{X}}_k^{[i]}\right)$
10:    **end for**
11:    $\overline{\hat{x}}_{k+1}^{[i]} \leftarrow \begin{bmatrix} h_{1,k+1}^{[i]} & \cdots & h_{n_x,k+1}^{[i]} \end{bmatrix}^\top$
12:    $\underline{\hat{x}}_{k+1}^{[i]} \leftarrow -\begin{bmatrix} h_{n_x+1,k+1}^{[i]} & \cdots & h_{2n_x,k+1}^{[i]} \end{bmatrix}^\top$
13:    $\widehat{\mathcal{X}}_{k+1}^{[i]} \leftarrow \left\{ x \in \mathbb{R}^{n_x} \mid \underline{\hat{x}}_{k+1}^{[i]} \le x \le \overline{\hat{x}}_{k+1}^{[i]} \right\}$
14: **end for**
Output: approximate reachable sets $\widehat{\mathcal{X}}_{k+1}^{[1]}, \ldots, \widehat{\mathcal{X}}_{k+1}^{[M]}$

V. MODEL UNCERTAINTY

In this section, we introduce model uncertainty into the dynamics and extend Theorem 1 to account for this. Consider the dynamics in (1). Instead of assuming direct knowledge of the state and input matrices, we now assume that they lie

in the convex hull of a finite number of matrices, i.e.

$$A_{ii} \in \mathrm{co}\left\{A_{ii}^1, \ldots, A_{ii}^{C_i}\right\}, \quad B_i \in \mathrm{co}\left\{B_i^1, \ldots, B_i^{D_i}\right\}, \tag{19}$$

where $C_i, D_i \in \mathbb{Z}^+$, for $i = 1, \ldots, M$. Note that Ψ_k^i in (18) depends on A_{ii} and B_i, so we can write $\Psi_k^i(h_p; A_{ii}, B_i)$.

Theorem 2: Consider the formulation in Theorem 1, but with the addition of model uncertainty in the state and input matrices described by (19). If there exists a solution to

$$\min_{\Gamma, \lambda, \nu, \eta, h_p} \quad h_p$$

$$\text{subject to} \quad \Delta_k^i(\Gamma) + \Theta^i(\lambda, \nu, \eta) + \Psi_k^i(h_p; A_{ii}^c, B_i^d) \preceq 0,$$
$$\forall\, c \in \{1, \ldots, C_i\}, \; d \in \{1, \ldots, D_i\},$$

for $p = 1, \ldots, m$, where $H_1, \ldots, H_m$ are specified by the user, then the resulting polytope is the solution to (4a)–(4c).

Proof: The proof relies on the fact that $\Psi_k^i(h_p; A_{ii}^c, B_i^d)$ depends affinely on A_{ii}^c and B_i^d. The proof has some similarities to that in [9], but the forms of the matrices and dynamics differ, so we include the proof for completeness. First, note that we can write (19) as

$$A_{ii} = \sum_{c=1}^{C_i} \alpha_c A_{ii}^c, \quad B_i = \sum_{d=1}^{D_i} \beta_d B_i^d, \tag{20}$$

where $\sum_{c=1}^{C_i} \alpha_c = 1$, $\alpha_c \geq 0 \;\forall c \in \{1, \ldots, C_i\}$ and $\sum_{d=1}^{D_i} \beta_d = 1$, $\beta_d \geq 0 \;\forall d \in \{1, \ldots, D_i\}$. From the definitions of Ψ_k^i and $\tilde{A}_i$, we can write

$$\Psi_k^i(h_p; A_{ii}^c, B_i^d) = \begin{bmatrix} 0 & \Phi_k^i(A_{ii}^c, B_i^d) \\ \Phi_k^i(A_{ii}^c, B_i^d)^\top & \Xi_k^i(h_p; B_i^d) \end{bmatrix},$$

where

$$\Phi_k^i(A_{ii}^c, B_i^d) = \begin{bmatrix} \begin{bmatrix} A_{ii}^c & A_{ig(i,1)} & \cdots & A_{ig(i,q_i)} \end{bmatrix}^\top H_p \\ 0 \\ -B_i^{d^\top} H_p \end{bmatrix},$$

and

$$\Xi_k^i(h_p; B_i^d) = 2 H_p^\top \left(B_i^d \overline{u}_i + w_k^{[i]} \right) - 2 h_p,$$

so it can be seen that $\Psi_k^i(h_p; A_{ii}^c, B_i^d)$ depends affinely on A_{ii}^c and B_i^d. Hence, multiplying by α_c and β_d, summing over $c = 1, \ldots, C_i$ and $d = 1, \ldots, D_i$, and using $\sum_{c=1}^{C_i} \alpha_c A_{ii}^c = A_{ii}$, $\sum_{c=1}^{C_i} \alpha_c = 1$, $\sum_{d=1}^{D_i} \beta_d B_i^d = B_i$ and $\sum_{d=1}^{D_i} \beta_d = 1$ from (20) gives

$$\sum_{c=1}^{C_i} \sum_{d=1}^{D_i} \alpha_c \beta_d \Psi_k^i(h_p; A_{ii}^c, B_i^d) = \Psi_k^i(h_p; A_{ii}, B_i). \tag{21}$$

Finally, multiplying the LMIs in the semidefinite program by α_c and β_d and summing over $c = 1, \ldots, C_i$ and $d = 1, \ldots, D_i$, such that

$$\sum_{c=1}^{C_i} \sum_{d=1}^{D_i} \alpha_c \beta_d \left[\Delta_k^i(\Gamma) + \Theta^i(\lambda, \nu, \eta) \right]$$

$$+ \sum_{c=1}^{C_i} \sum_{d=1}^{D_i} \alpha_c \beta_d \Psi_k^i(h_p; A_{ii}^c, B_i^d) \preceq 0,$$

results in

$$\Delta_k^i(\Gamma) + \Theta^i(\lambda, \nu, \eta) + \Psi_k^i(h_p; A_{ii}, B_i) \preceq 0,$$

which follows from the fact that $\sum_{c=1}^{C_i} \sum_{d=1}^{D_i} \alpha_c \beta_d = 1$ and from (21). ∎

This result is useful, as we only have to solve a finite number of semidefinite programs to find an overapproximation of the forward reachable set for all convex combinations of $A_{ii}^1, \ldots, A_{ii}^{C_i}$ and $B_i^1, \ldots, B_i^{D_i}$.

VI. EXPERIMENTS

In this section, we use two realistic examples of multi-agent systems to demonstrate our results. We also give a comparison to the approach proposed at the start of Section III, in which the states of each agent are augmented into one overall state and existing reachability methods are applied on the overall system. We also introduce model uncertainty into one of the systems and analyse this case. Simulations were performed on MATLAB, and CVX with MOSEK was used to solve the semidefinite programs. The NNs were trained to approximate distributed MPC schemes with a given horizon; the systems were simulated with MPC, and the resulting input-output data pairs were used to train the NNs.

A. Vehicle platooning

In the first example, we consider the example of control of a platoon of vehicles. There are several forms of this problem, and control of a vehicular platoon has a number of benefits, including improved safety, higher road capacity, lower emissions, and/or reduced congestion [25]–[28]. In this example, we consider the adaptive cruise control (ACC) problem, in which each vehicle aims to maintain a fixed distance from the vehicle in front, whilst travelling at a given velocity. The continuous-time longitudinal dynamics of each vehicle are given by [29]

$$\dot{x}^{[i]}(t) = \bar{A}_{ii} x^{[i]}(t) + \bar{A}_{ii-1} x^{[i-1]}(t) + \bar{B}_i u^{[i]}(t),$$

(note that $\mathcal{N}_i = \{i-1\}$) where the state vector is

$$x^{[i]}(t) = \begin{bmatrix} e^{[i]}(t) & v^{[i]}(t) & a^{[i]}(t) \end{bmatrix}^\top,$$

where $e^{[i]}(t)$ is the distance error between vehicle i and vehicle $i-1$ (i.e. if the desired distance between vehicle i and the vehicle in front, $i-1$, is $\bar{d}^{[i]}$ and the actual distance is $d^{[i]}(t)$, then $e^{[i]}(t) = d^{[i]}(t) - \bar{d}_i$), $v^{[i]}(t)$ is the velocity of the ith vehicle, $a^{[i]}(t)$ is the acceleration of the ith vehicle, and

$$\bar{A}_{ii} = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -\frac{1}{\tau} \end{bmatrix}, \quad \bar{A}_{ii-1} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$\bar{B}_i = \begin{bmatrix} 0 & 0 & \frac{1}{\tau} \end{bmatrix}^\top,$$

where τ is the engine time constant [30] and $u^{[i]}(t)$ is the ith acceleration input. Note that the lead vehicle ($i = 1$) has no physical neighbour, but this can be resolved by imagining a virtual vehicle [29] with state $x^{[0]} = \begin{bmatrix} 0 & \bar{v} & 0 \end{bmatrix}^\top$, where $\bar{v}$ is the reference velocity (to be maintained by the platoon). This
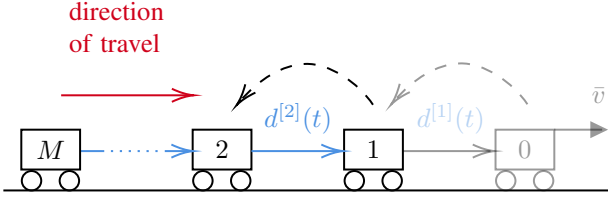
Figure 2. Platoon of M vehicles, where each vehicle i only receives information from vehicle $i-1$; the 'virtual' vehicle is shown with increased opacity

is shown in Figure 2. The dynamics are discretised assuming zero-order hold (ZOH) with a sample period $T = 0.1$ s, treating $x^{[i-1]}(t)$ and $u^{[i]}(t)$ as exogenous inputs, such that the discrete-time dynamics are in the form in (1), where $w_k^{[i]} = 0 \ \forall i \in \mathcal{I}$. The NNs have 2 hidden layers, both with 15 neurons. The ith control input is

$$u_k^{[i]} = \text{sat}_{\mathcal{U}_i}\left[\pi_{ii-1}\left(\begin{bmatrix} e_k^{[i]} \\ v_k^{[i-1]} - v_k^{[i]} \\ a_k^{[i-1]} - a_k^{[i]} \end{bmatrix}\right)\right],$$

where $e_k^{[i]} \equiv e^{[i]}(kT)$, $v_k^{[i]} \equiv v^{[i]}(kT)$ and $a_k^{[i]} \equiv a^{[i]}(kT)$. The forward reachable sets were computed for five time steps, $M = 9$ agents, initial conditions given by $\mathcal{X}_0^{[i]} = \{x \in \mathbb{R}^3 \mid \underline{x} \le x \le \overline{x}\} \ \forall i \in \mathcal{I}$, where $\underline{x}^\top = \begin{bmatrix} -0.1 & 19.95 & -0.01 \end{bmatrix}$ and $\overline{x}^\top = \begin{bmatrix} 0.1 & 20.05 & 0.01 \end{bmatrix}$, and controller limits given by $\overline{u}_i = -\underline{u}_i = 5 \ \forall i \in \mathcal{I}$. A step change of -2 was applied to the reference velocity at $k = 0$ (from $\overline{v} = 20$ to $\overline{v} = 18$). The results are shown in Figure 3 for the first three vehicles. Note that the step change in $\overline{v}$
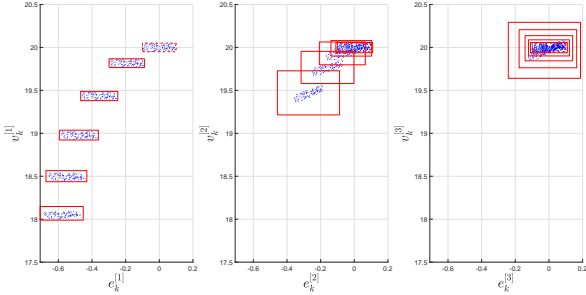

Figure 3. Plots of the reachable sets in red (solid) for distance error and velocity, and simulated trajectories (blue) for the vehicle platooning example; the initial set is shown in red (dashed) – only agents 1 to 3 are shown

takes some time to propagate down the platoon, hence the difference in range between the plots. A comparison between the computation time for this approach (Reach-SDP-MA) and the existing method (Reach-SDP) is shown in Table 1 for different values of M.

Table 1. Comparison of methods (times in s)

M	1	2	3	4	5
Reach-SDP-MA	3.85	7.04	11.60	14.17	20.82
Reach-SDP [8]	4.01	75.39	562.94	3636.62	41025.05

We then extend the vehicle platooning example to account for model uncertainty. Consider the case in which $A_{ii} \in \text{co}\{(1-\delta)A^0, (1+\delta)A^0\}$ and $B_i \in \text{co}\{(1-\delta)B^0, (1+\delta)B^0\}$, where $A^0 \in \mathbb{R}^{3\times3}$ and $B^0 \in \mathbb{R}^3$ are the nominal values of A_{ii} and B_i, respectively, and $\delta = 0.01$. The results are shown in Figure 4 for the first three vehicles. Because of the uncertainty, the size of the
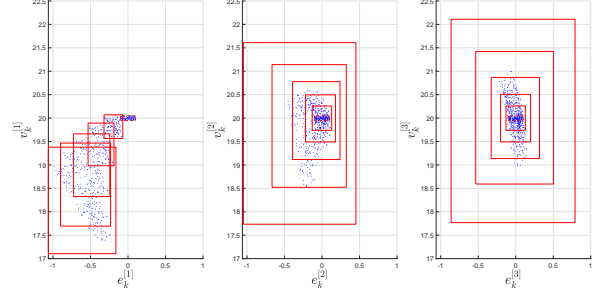

Figure 4. Plots of the reachable sets in red (solid) for distance error and velocity, and simulated trajectories (blue) for the vehicle platooning example with model uncertainty; the initial set is shown in red (dashed) – only agents 1 to 3 are shown

exact reachable sets increases, so the overapproximations are larger (compared to Figure 3).

B. Power network system

For the second example, we consider automatic generation control (AGC) of a power network system. Unlike the previous example, the dynamics are not identical across agents, and some agents have more than one neighbour. There is also an exogenous input term. This system consists of M generation areas, and the aim is to reduce the frequency deviation in each area, in spite of load changes. Common approaches to this control problem include decentralised and distributed MPC schemes [20], [31], [32]. The continuous-time dynamics of each area are given by [31], [33]

$$\dot{x}^{[i]}(t) = \bar{A}_{ii}x^{[i]}(t) + \sum_{j\in\mathcal{N}_i}\bar{A}_{ij}x^{[j]}(t) + \bar{B}_i u^{[i]}(t) + \bar{L}_i\Delta P_L^{[i]}(t),$$

where the state vector for area i is

$$x^{[i]}(t) = \begin{bmatrix} \Delta\theta^{[i]}(t) & \Delta\omega^{[i]}(t) & \Delta P_m^{[i]}(t) & \Delta P_v^{[i]}(t) \end{bmatrix}^\top,$$

where $\Delta\theta^{[i]}(t)$, $\Delta\omega^{[i]}(t)$, $\Delta P_m^{[i]}(t)$ and $\Delta P_v^{[i]}(t)$ are the deviations in rotor angle, frequency, mechanical power and steam valve position, respectively, from the nominal values [34], $u^{[i]}(t)$ is the reference power, $\Delta P_L^{[i]}(t)$ is the local power load, and

$$\bar{A}_{ii} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{\sum_{j\in\mathcal{N}_i}P_{ij}}{2H_i} & -\frac{D_i}{2H_i} & \frac{1}{2H_i} & 0 \\ 0 & 0 & -\frac{1}{T_{t_i}} & \frac{1}{T_{t_i}} \\ 0 & -\frac{1}{R_iT_{g_i}} & 0 & -\frac{1}{T_{g_i}} \end{bmatrix},$$

$$\bar{A}_{ij} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{P_{ij}}{2H_i} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \bar{B}_i = \begin{bmatrix} 0 & 0 & 0 & T_{g_i} \end{bmatrix}^\top,$$

$$\bar{L}_i = \begin{bmatrix} 0 & -\frac{1}{2H_i} & 0 & 0 \end{bmatrix}^\top,$$

where P_{ij}, H_i, D_i, T_{t_i}, R_i and T_{g_i} are defined in [31]. In this example, we consider $M = 4$ generation areas (Scenario 1 in [34]), where $\mathcal{N}_1 = \{2\}$, $\mathcal{N}_2 = \{1, 3\}$, $\mathcal{N}_3 = \{2, 4\}$ and $\mathcal{N}_4 = \{3\}$, as shown in Figure 5. The dynamics are
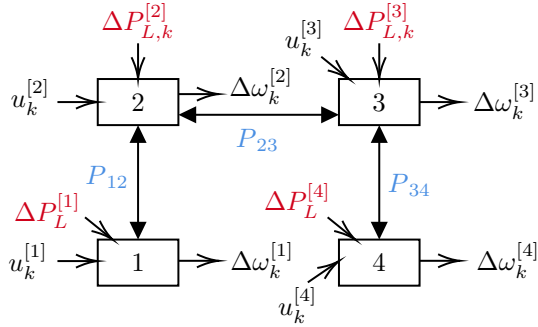

Figure 5. Power network system with 4 areas (adapted from [31])

discretised assuming ZOH with a sample period $T = 1$ s, treating $x^{[j]}(t) \ \forall j \in \mathcal{N}_i$, $u^{[i]}(t)$ and $\Delta P_L^{[i]}(t)$ as exogenous inputs, such that the discrete-time dynamics are in the form in (1). The NNs have 2 hidden layers, both with 10 neurons. The ith control input is

$$u_k^{[i]} = \text{sat}_{\mathcal{U}_i} \left[\sum_{j \in \mathcal{N}_i} \pi_{ij} \left(\begin{bmatrix} x_k^{[i]} \\ x_k^{[j]} \end{bmatrix} - \begin{bmatrix} x_{\text{ref},k}^{[i]} \\ x_{\text{ref},k}^{[j]} \end{bmatrix} \right) \right],$$

where $x_{\text{ref},k}^{[i]}$ and $x_{\text{ref},k}^{[j]}$ are the state and neighbour reference values, respectively. These are incorporated into the reachability analysis by modifying the first weight matrices and bias vectors. The forward reachable sets were computed for three time steps with initial conditions given by $\mathcal{X}_0^{[i]} = \{x \in \mathbb{R}^4 \mid \underline{x} \leq x \leq \overline{x}\} \ \forall i \in \mathcal{I}$, where $\overline{x}^\top = -\underline{x}^\top = \begin{bmatrix} 10^{-4} & 10^{-7} & 10^{-3} & 10^{-3} \end{bmatrix}$, and controller limits given in [34]. A step change of -0.15 was applied to $\Delta P_{L,k}^{[i]} \equiv \Delta P_L^{[i]}(kT) \ \forall i \in \mathcal{I}$, and the results are shown in Figure 6.
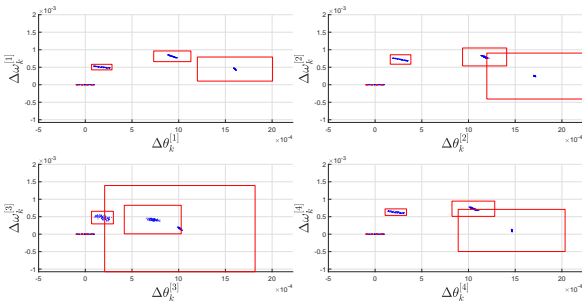

Figure 6. Plots of the reachable sets in red (solid) for angle and frequency deviation and simulated trajectories (blue) for the power network example; the initial set is shown in red (dashed); here, $\Delta \theta_k^{[i]} \equiv \Delta \theta^{[i]}(kT) \ \forall i \in \mathcal{I}$ and $\Delta \omega_k^{[i]} \equiv \Delta \omega^{[i]}(kT) \ \forall i \in \mathcal{I}$

VII. CONCLUSION AND FUTURE WORK

In this paper, we presented a scalable method to overapproximate the forward reachable sets of multi-agent systems with distributed NNC architectures. After simplifying the dynamics, we presented a method to split the overall reachability problem into multiple smaller reachability problems. We then extended this approach to account for model uncertainty. The effectiveness of this method was demonstrated on realistic examples, and it was shown to be significantly faster than the overall reachability method whilst producing the same bounds. It should also be noted that the method presented in this paper can be applied to any system that can be decomposed into the form in (1); it is not necessarily specific to multi-agent systems.

Opportunities for future work include using the general framework presented in this paper to improve the efficiency of other reachability methods, such as LP-based methods. Also, further consideration could be given to synthesis of the NNCs, and reachability methods could be incorporated into the training process. Other types of activation functions and sets could also be considered.

REFERENCES

[1] B. Karg and S. Lucia, "Efficient representation and approximation of model predictive control laws via deep learning," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3866–3878, 2020.

[2] H. Li, T. Wei, A. Ren, Q. Zhu, and Y. Wang, "Deep reinforcement learning: Framework, applications, and embedded implementations: Invited paper," in *2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2017, pp. 847–854.

[3] M. Fazlyab, M. Morari, and G. J. Pappas, "Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming," *IEEE Transactions on Automatic Control*, vol. 67, no. 1, pp. 1–15, 2022.

[4] P. Shi and B. Yan, "A survey on intelligent control for multiagent systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 161–175, 2021.

[5] M. Andreasson, "Control of multi-agent systems with applications to distributed frequency control power systems," Ph.D. dissertation, KTH Royal Institute of Technology, 2013.

[6] M. Andreasson, D. V. Dimarogonas, H. Sandberg, and K. H. Johansson, "Distributed control of networked dynamical systems: Static feedback, integral action and consensus," *IEEE Transactions on Automatic Control*, vol. 59, no. 7, pp. 1750–1764, 2014.

[7] R. Negenborn and J. Maestre, "Distributed model predictive control: An overview and roadmap of future research opportunities," *IEEE Control Systems Magazine*, vol. 34, no. 4, pp. 87–97, 2014.

[8] H. Hu, M. Fazlyab, M. Morari, and G. J. Pappas, "Reach-SDP: Reachability analysis of closed-loop systems with neural network controllers via semidefinite programming," in *2020 59th IEEE Conference on Decision and Control (CDC)*, 2020, pp. 5929–5934.

[9] A. Sadeghzadeh and P.-L. Garoche, "Reachability analysis of linear parameter-varying systems with neural network controllers," in *2022 IEEE Conference on Control Technology and Applications (CCTA)*, 2022, pp. 1372–1377.

[10] M. Everett, G. Habibi, C. Sun, and J. P. How, "Reachability analysis of neural feedback loops," *IEEE Access*, vol. 9, pp. 163 938–163 953, 2021.

[11] W. Xiang and T. T. Johnson, "Reachability analysis and safety verification for neural network control systems," *arXiv preprint arXiv:1805.09944*, 2018.

[12] Y. Zhang and X. Xu, "Reachability analysis and safety verification of neural feedback systems via hybrid zonotopes," *arXiv preprint arXiv:2210.03244*, 2022.

[13] ——, "Safety verification of neural feedback systems based on constrained zonotopes," in *2022 IEEE 61st Conference on Decision and Control (CDC)*. IEEE, 2022, pp. 2737–2744.

[14] N. Kochdumper, C. Schilling, M. Althoff, and S. Bak, "Open-and closed-loop neural network verification using polynomial zonotopes," *arXiv preprint arXiv:2207.02715*, 2022.

[15] M. Newton and A. Papachristodoulou, "Reachability analysis of neural feedback loops using sparse polynomial optimisation," in *2022 IEEE 61st Conference on Decision and Control (CDC)*, 2022, pp. 2745–2750.

[16] J. Fan, C. Huang, X. Chen, W. Li, and Q. Zhu, "ReachNN*: A tool for reachability analysis of neural-network controlled systems," *Automated Technology for Verification and Analysis*, p. 537.

[17] N. Fijalkow, J. Ouaknine, A. Pouly, J. Sousa-Pinto, and J. Worrell, "On the decidability of reachability in linear time-invariant systems," in *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*. ACM, 2019.

[18] K. Muandet, K. Fukumizu, B. Sriperumbudur, and B. Schölkopf, "Kernel mean embedding of distributions: A review and beyond," *Foundations and Trends® in Machine Learning*, vol. 10, no. 1-2, pp. 1–141, 2017.

[19] P. Wang, H. Deng, J. Zhang, L. Wang, M. Zhang, and Y. Li, "Model predictive control for connected vehicle platoon under switching communication topology," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 7817–7830, 2022.

[20] H. Hu, K. Gatsis, M. Morari, and G. J. Pappas, "Non-cooperative distributed mpc with iterative learning," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 5225–5232, 2020, 21st IFAC World Congress.

[21] Q. Yuan and X. Li, "Distributed model predictive formation control for a group of uavs with newton-euler dynamics," in *2020 European Control Conference (ECC)*, 2020, pp. 1484–1489.

[22] F. Lewis, H. Zhang, K. Hengster-Movric, and A. Das, *Cooperative Control of Multi-Agent Systems: Optimal and Adaptive Design Approaches*, ser. Communications and Control Engineering. Springer London, 2014.

[23] X. Chen and S. Sankaranarayanan, "Decomposed reachability analysis for nonlinear systems," in *2016 IEEE Real-Time Systems Symposium (RTSS)*, 2016, pp. 13–24.

[24] M. Chen, S. L. Herbert, M. S. Vashishtha, S. Bansal, and C. J. Tomlin, "Decomposition of reachable sets and tubes for a class of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 63, no. 11, pp. 3675–3688, 2018.

[25] J. C. Zegers, E. Semsar-Kazerooni, J. Ploeg, N. van de Wouw, and H. Nijmeijer, "Consensus control for vehicular platooning with velocity constraints," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 5, pp. 1592–1605, 2018.

[26] Y. Zheng, S. Eben Li, J. Wang, D. Cao, and K. Li, "Stability and scalability of homogeneous vehicular platoon: Study on the influence of information flow topologies," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 1, pp. 14–26, 2016.

[27] Y. Zheng, S. E. Li, K. Li, and L.-Y. Wang, "Stability margin improvement of vehicular platoon considering undirected topology and asymmetric control," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 4, pp. 1253–1265, 2016.

[28] J. Ploeg, B. T. M. Scheepers, E. van Nunen, N. van de Wouw, and H. Nijmeijer, "Design and experimental evaluation of cooperative adaptive cruise control," in *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2011, pp. 260–265.

[29] S. Öncü, J. Ploeg, N. van de Wouw, and H. Nijmeijer, "Cooperative adaptive cruise control: Network-aware analysis of string stability," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1527–1537, 2014.

[30] V. Jain, D. Liu, and S. Baldi, "Adaptive strategies to platoon merging with vehicle engine uncertainty," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15065–15070, 2020, 21st IFAC World Congress.

[31] S. Riverso, M. Farina, and G. Ferrari-Trecate, "Plug-and-play decentralized model predictive control for linear systems," *IEEE Transactions on Automatic Control*, vol. 58, no. 10, pp. 2608–2614, 2013.

[32] M. Zeilinger, Y. Pu, S. Riverso, G. Ferrari-Trecate, and C. Jones, "Plug and play distributed model predictive control based on distributed invariance and optimization," in *52nd IEEE Conference on Decision and Control*, 2013, pp. 5770–5776.

[33] H. Saadat, *Power System Analysis*, ser. Electrical and Computer Engineering Series. WCB/McGraw-Hill, 1999.

[34] S. Riverso and G. Ferrari-Trecate, "Hycon2 benchmark: Power network system," *arXiv preprint arXiv:1207.2000*, 2012.