2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Distributed Safety Verification for Multi-Agent Systems

Han Wang, Antonis Papachristodoulou, Kostas Margellos

Abstract— **The control barrier function (CBF) framework is a powerful tool for safe controller design and safety analysis. Given a dynamical system and a CBF, the system is safe if the CBF-induced constraints are satisfied for every state inside an invariant set, which is a subset of the safe set. In this paper we propose a safety verification algorithm for networked nonlinear multi-agent systems. In our proposed algorithm, we independently sample scenarios from the invariant set, and subsequently quantify safety for the multi-agent system by solving a scenario program in a distributed manner. Both the scenario sampling and safety verification algorithms are fully distributed. The efficacy of our algorithm is demonstrated by an example on multi-robot collision avoidance.**

I. Introduction

Safety for dynamical systems is a property that requires the system state to remain within a safe set for all time. This property is of importance in various applications, including collision avoidance [2], [3], vehicle platooning [4], [5], and vehicle merging control [6]. For a designed controller, validating safety for the dynamical system is naturally important. In a single-agent system, the agent can evaluate the system's behaviour to assess the risk of being unsafe under the employed control input. For a multi-agent safety verification problem, cooperation among agents is essential as safety involves multiple agents.

Over the past decade, there has been considerable attention focused on safety verification. Given a dynamical system and a safe set, the goal is to verify safety with a well-posed algorithm. There are several ways to formulate and solve this problem. A reach-avoid game can be formulated and solved to determine the set of initial conditions resulting in trajectories that reach the target set without entering the unsafe set [7]–[9]. Thus, safety verification is a part of reachability analysis. The challenge lies in solving the Hamilton-Jacobi Partial Differential Equation underlying the problem.

To address this issue, the barrier certificates method was proposed within a convex programming framework [10], [11]. This method identifies an invariant set inside the safe set that system trajectories cannot escape from, thus ensuring safety. Numerical methods using barrier certificates propose sum-of-squares (SOS) programs, which are equivalent to semi-definite programs [12], [13]. However, in real-world applications, the system model and control input may not be precisely known or may be unknown. Handling uncertainties is arduous for these methods.

The authors are with the Department of Engineering Science, University of Oxford, Oxford, United Kingdom. E-mails: {han.wang, antonis kostas.margellos}@eng.ox.ac.uk

An extended version to this paper can be found in [1].

Another emerging technique for safety in control systems is the Control Barrier Function (CBF) approach [14]. By regulating the inner product of the CBF derivative and the vector field of the controlled system, safety is rigorously guaranteed at any time. In other words, if the CBF condition holds for every state in the invariant set, the system is safe. Safety can be verified by checking the CBF condition for states sampled from the invariant set to obtain a more tractable computation. This methodology lies in the scope of the scenario approach, [15]–[20].

The main contribution of this paper is that it proposes a distributed safety verification algorithm, which addresses the problem of certifying safety for a multi-agent system. Safety is quantified by means of CBFs, which are used to detect violations of safety constraints. A scenario-based verification algorithm is proposed to provide a probabilistic quantification of safety. We also propose a sequential sampling algorithm to sample scenarios efficiently in a distributed fashion. We extend [19, Theorem 1] to a multi-agent setting and establish a tighter upper bound on the probability of being unsafe. The safety verification program is shown to be amenable to parallelization.

Section II presents necessary preliminaries for the paper. Section III presents the main results, which include a scenario-based safety verification algorithm, a sequential scenario sampling algorithm and the probabilistic result. Section IV demonstrates the safety verification algorithm on a multi-robot system collision avoidance case study using a standard distributed safe controller design algorithm. Section V concludes the paper and provides some directions for future research.

II. Preliminaries

A. Notations

$\mathbb{R}$, $\mathbb{R}^N$, $\mathbb{R}_+$ denote the sets of one-dimensional, N-dimensional and nonnegative real numbers, respectively. $\mathbb{N}$ is the set of natural numbers. A continuous function $\alpha(\cdot) : (-b, a) \to (-\infty, +\infty)$ is said to be an extended class-$\mathcal{K}$ function for positive a and b, if it is strictly increasing and $\alpha(0) = 0$. $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denotes a graph with a nodes set $\mathcal{V}$ and an edge set $\mathcal{E}$. Throughout the paper $\mathcal{S}$ is used for a safe set, $\mathcal{U}$ is denotes an input constraint set, and $\mathcal{B}$ is used for an invariant set. Boldface symbols are used as stacked vectors for vector elements, e.g., $\boldsymbol{x} = [x_1^\top, \dots, x_N^\top]^\top$. Specifically, $\mathbf{0}$ is vector whose elements are all zero, and I is an identity matrix, with their dimensions being clear from the context. For a set $\mathcal{K}$, $|\mathcal{K}|$ denotes its cardinality. For a function $s(x) : \mathbb{R}^n \to \mathbb{R}$, we use the calligraphic font to represent the corresponding zero-super level set, i.e., $\mathcal{S} := \{x | s(x) \geq 0\}$.

B. Control Barrier Functions

Consider a nonlinear control-affine system

$$\dot{x} = f(x) + g(x)u, \tag{1}$$

with $x(t) \in \mathbb{R}^n$, $u(t) \in \mathcal{U} \subset \mathbb{R}^m$, $f(x) : \mathbb{R}^n \to \mathbb{R}^m$, and $g(x) : \mathbb{R}^n \to \mathbb{R}^{n\times m}$. Both f and g are further assumed to be Lipschitz continuous. Defining the solution of (1) to be $x(t, x_0)$, where $x_0 = x(0)$ represents the initial condition and t denotes time, our goal is to verify safety for the closed-loop system $\dot{x} = f(x) + g(x)u(x)$. For a trajectory starting from x_0, safety requires $x(t, x_0) \in \mathcal{S}, \forall t \geq 0$, where $\mathcal{S}$ is a predefined safe set.

Control barrier functions constitute an extension to barrier certificates [10] for safety verification and controller design. It has been revealed in these papers that safety is closely related to the notion of *invariance*.

Definition 1. *A set $\mathcal{B} \subset \mathbb{R}^n$ is said to be invariant with respect to vector field* (1)*, if for any $x_0 \in \mathcal{B}$, there exists $u \in \mathcal{U}$ such that $x(t, x_0) \in \mathcal{B}$.*

The relationship between safety and invariance is demonstrated in the following lemma.

Lemma 1. *If there exists an invariant $\mathcal{B} \subseteq \mathcal{S}$, system* (1) *is safe for any $x_0 \in \mathcal{B}$.*

Definition 2. *For the control-affine dynamical system* (1)*, a continuously differentiable function $b(\cdot) : \mathbb{R}^n \to \mathbb{R}$ is said to be a control barrier function for the set $\mathcal{B}$, if there exists an extended class-$\mathcal{K}$ function $\alpha(\cdot)$ and a set $\mathcal{C}$, where $\mathcal{B} \subseteq \mathcal{C} \subset \mathbb{R}^n$, such that for any $x \in \mathcal{C}$,*

$$\sup_{u\in\mathcal{U}} [\mathcal{L}_f b(x) + \mathcal{L}_g b(x)u + \alpha(b(x))] \geq 0. \tag{2}$$

Here $\mathcal{L}_f b(x)$ and $\mathcal{L}_g b(x)$ are Lie derivatives, which are defined by $\mathcal{L}_f b(x) := \frac{\partial b(x)}{\partial x} f(x)$ and $\mathcal{L}_g b(x) := \frac{\partial b(x)}{\partial x} g(x)$, respectively.

Lemma 2. *[21] Consider a control barrier function $b(x)$ defined on $\mathcal{C}$. Then for any $x \in \mathcal{C}$, any $u(x) \in K_{cbf}(x)$ will render the set $\mathcal{B}$ invariant.*

We note that the control barrier function $b(x)$ does not generally need to be the same function as $s(x)$. The synthesis problem is beyond the scope of this paper; readers are referred to [22] for details.

C. Multi-agent Optimisation Problem

A distributed optimisation problem is the problem of finding a global optimum for an optimisation problem over a network of agents, where every agent has access to a local decision variable, and agents are coupled through the constraints.

Consider a networked system of N agents communicating over a connected and undirected graph $\mathcal{G}$, with nodes set $\mathcal{V} = \{1, \ldots, N\}$, and edge set $\mathcal{E}$ such that $\{i, j\} \in \mathcal{E}$ if agent j communicates with agent i. In addition to the pair-wise interaction relationship, agents are also grouped in E sub-networks. For each sub-network $\mathcal{G}_e$, $e = 1, \ldots, E$, the set of agents inside it is denoted by $\mathcal{V}_e \subseteq \mathcal{V}$. We denote the set of constraints that agent i participates in as $\mathcal{C}_i$, with $\mathcal{V}_e = \{i | e \in \mathcal{C}_i\}$. $\mathcal{N}_i$ is the set of neighboring agents for agent i. The multi-agent distributed optimisation problem that needs to be solved in a distributed manner can then be defined as

$$\begin{aligned} &\min_{\boldsymbol{x}} \sum_{i=1}^{N} J_i(x_i) \\ &\text{subject to } x_i \in \mathcal{X}_i, \forall i = 1, \ldots, N, \\ &\qquad \sum_{k\in\mathcal{V}_e} h_{ke}(x_k) \leq 0, \forall e = 1, \ldots, E, \end{aligned} \tag{3}$$

where for all $i = 1, \ldots, N$, $x_i \in \mathbb{R}^{d_i}$ for some $d_i \in \mathbb{N}$. Also, $J_i(x_i) : \mathbb{R}^{d_i} \to \mathbb{R}$. For all $k \in \mathcal{V}_e$, $h_{ke} : \mathbb{R}^{d_k} \to \mathbb{R}$ stand for the group constraints that agent k needs to satisfy. The function $h_{ke}(x_k)$ is not necessarily the same for every k in the same sub-network. An efficient distributed algorithm to solve problem (3) is [23, Algorithm RSDD].

Algorithm 1 Distributed Optimisation Algorithm for agent i [23, adopted from Algorithm RSDD]

```
Initialization Predefined $\lambda_{il}^0, \forall l \in \mathcal{N}_i \cap \mathcal{V}_e, \forall e \in \mathcal{C}_i$.
Output: Optimal solution $x_i^*$.
1: while not reaching convergence do
2:     Receive $\lambda_{il}^k$ from $\forall l \in \mathcal{N}_i \cap \mathcal{V}_e, \forall e \in \mathcal{C}_i$.
3:     Compute $((x_i^{k+1}, \boldsymbol{\rho}_i^{k+1}), \boldsymbol{\mu}_i^{k+1})$ as a primal-dual solution of the following optimisation problem.
           $$\begin{aligned} &\min_{x_i, \boldsymbol{\rho}_i} \; J_i(x_i) \\ &\text{subject to } x_i \in \mathcal{X}_i, \boldsymbol{\rho}_i \geq 0, \\ &\qquad h_{ie}(x_i) + \sum_{l\in\mathcal{N}_i\cap\mathcal{V}_e} (\lambda_{il}^k - \lambda_{li}^k) \leq 0, \forall e \in \mathcal{C}_i. \end{aligned} \tag{4}$$
4:     Receive $\mu_{le}^{k+1}$ from agent $l \in \mathcal{N}_i \cap \mathcal{V}_e$.
5:     Update $\lambda_{il}$ by
           $$\lambda_{il}^{k+1} = \lambda_{il}^k - \gamma^k(\mu_{ie}^{k+1} - \mu_{le}^{k+1}). \tag{5}$$
6: end while
```

D. Scenario Optimisation

Scenario optimisation is a data-driven robust optimisation methodology where one aims at searching for an optimum over uncertain sets. For the case where the uncertain sets are continuous, and possibly non-convex or unknown, this kind of optimisation problem is usually hard to solve with guaranteed robustness. The scenario approach, on the other hand, proposes to solve the problem over finite empirical records, named *scenarios* for a certain confidence of feasibility of the optimal solution. An uncertain optimisation problem is formulated as

$$\begin{aligned} &\min_{x\in\mathcal{H}} \; c^\top x \\ &\text{subject to } x \in \mathcal{X}_\delta, \text{ for all } \delta \in \Delta, \end{aligned} \tag{6}$$

where $x \in \mathbb{R}^n$ is a decision variable constrained by a convex set $\mathcal{H} \subset \mathbb{R}^n$, $c \in \mathbb{R}^n$ is a constant vector. The convex uncertain constraint set $\mathcal{X}_\delta$ is parameterized by an uncertain parameter δ, which is a random variable defined on a probability space $(\Delta, \mathcal{F}, \mathbb{P})$, where Δ is a sample space, $\mathcal{F}$ is an event space and $\mathbb{P}$ is a probability measure. As discussed above, directly solving problem (6) involves an infinite number of constraints for a continuous set Δ. Alternatively, the scenario approach proposes to solve the problem by enforcing the constraints only on scenarios.

The corresponding prototype convex scenario optimisation problem is formulated as

$$\begin{aligned} &\min_{x \in \mathcal{X}} \; c^\top x \\ &\text{subject to } x \in \bigcap_{i=1,\dots,N} \mathcal{X}_{\delta_i}, \end{aligned} \tag{7}$$

where δ_i, $i = 1, \dots, N$ are *scenarios* sampled independently from the uncertain set Δ. The scenario optimisation problem (7) is a convex optimisation problem which can be solved efficiently. Clearly, the optimal solution satisfies $x_N^* \in \bigcap_{i=1,\dots,N} \mathcal{X}_{\delta_i}$, but is not necessarily within $\mathcal{X}_\delta$ for an arbitrary new $\delta \in \Delta$.

III. Distributed Safety Verification

This section presents a method to verify safety for a multi-agent system in a distributed manner. We employ the violation of the CBF constraints as a metric of the possibility that the underlying system becomes unsafe. This verification method is applicable to any type of time-invariant control inputs and the CBF is considered only as a verification criterion and not necessarily a control design principle.

Consider an N-agent system with the dynamics of the i-th agent described by

$$\dot{x}_i = f_i(x_i) + g_i(x_i)u_i, \tag{8}$$

where $x_i(t) \in \mathcal{X}_i \subset \mathbb{R}^{n_i}$ denotes its state, $u_i \in \mathcal{U}_i \subset \mathbb{R}^{m_i}$ denotes its control input, while $\mathcal{X}_i$ and $\mathcal{U}_i$ are the state space and control admissible set, respectively. The dynamics $f_i(x_i) : \mathbb{R}^{n_i} \to \mathbb{R}^{n_i}$ and $g_i(x_i) : \mathbb{R}^{n_i} \to \mathbb{R}^{n_i} \times \mathbb{R}^{m_i}$ are both locally Lipchistz-continuous. Vector $\boldsymbol{x} = [x_1^\top, \dots, x_N^\top]^\top$ stacks the states of all systems, $\boldsymbol{u} = [u_1^\top, \dots, u_N^\top]^\top$ stacks the control inputs, while $f(\boldsymbol{x})$, $g(\boldsymbol{x})$ stack the dynamics for each agent. In this way, the system dynamics of the whole multi-agent system can be described by $\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + g(\boldsymbol{x})\boldsymbol{u}$.

To maintain safety within each sub-network e, each agent i can communicate and collaborate with its neighbors $j \in \mathcal{N}_i$ to ensure $s_e(\boldsymbol{x}_e) \geq 0$, where $s_e(\cdot) \in \mathbb{R}$. The safe set is $\mathcal{S}_e := \{\boldsymbol{x} : s_e(\boldsymbol{x}_e) \geq 0\}$. Correspondingly, CBFs $b_e(\boldsymbol{x}_e)$ are synthesized in advance for safety verification, with the invariant set defined by $\mathcal{B}_e := \{\boldsymbol{x}_e : b_e(\boldsymbol{x}_e) \geq 0\}$. The overall invariant (safe) set is defined by the Cartesian intersection of all the individual safe (invariant) sets, i.e. $\mathcal{B} = \bigcap_{e=1}^E \mathcal{B}_e$, $\mathcal{S} = \bigcap_{e=1}^E \mathcal{S}_e$.

Based on these CBFs, the CBF conditions for safety at state $\boldsymbol{x}$ are

$$\sum_{i \in \mathcal{V}_e} -h_{ie}(\boldsymbol{x}) \leq 0, \forall e = 1, \dots, E, \tag{9}$$

where $h_{ie}(\boldsymbol{x}) = \frac{\partial b_e}{\partial x_i}(f_i(x_i) + g_i(x_i)u_i(\boldsymbol{x})) + \alpha_{ie}(b_e)$. If (9) holds for every $\boldsymbol{x} \in \mathcal{B}$, then we conclude that the system is safe using Lemma 2. However, this is usually arduous to validate if $\mathcal{B}$ is a set of non-zero measures, which contains *infinite* number of states to be verified. To address this problem, we propose to sample *finite* states for a tractable verification. The CBF conditions (9) will be checked over these realizations in a distributed manner.

A. Scenario Based Safety Verification

Consider an N-agent system (8) and an invariant set $\mathcal{B}$. Our goal is to verify whether $\boldsymbol{x}(t) \in \mathcal{B}, \forall \boldsymbol{x}(0) \in \mathcal{B}, \forall t \geq 0$ for system (1), using a feedback controller $\boldsymbol{u}(\boldsymbol{x})$. The form of $\boldsymbol{u}(\boldsymbol{x})$ is not necessarily known for our approach.

The scenario-based safety verification program is shown as follows:

$$\begin{aligned} \min_{\boldsymbol{z} \leq 0, \boldsymbol{\zeta} \geq 0} \; & \sum_{i=1}^N \sum_{e \in \mathcal{C}_i} \left(z_{ie}^2 + M \sum_{r=1}^{\bar{N}} \zeta_{ie}^{(r)} \right) \qquad \text{(SC-Verification)} \\ \text{s.t. } & \sum_{i \in \mathcal{V}_e} h_{ie}(u_i(\boldsymbol{x}^{(r)})) \leq \sum_{i \in \mathcal{V}_e} (z_{ie} + \zeta_{ie}^{(r)}), \\ & \forall e = 1, \dots, E, \forall r = 1, \dots, \bar{N}, \end{aligned} \tag{10}$$

where scenarios $\boldsymbol{x}^{(r)} \in \mathcal{B}$, $r = 1, \dots, \bar{N}$ are extracted according to some probability distribution to be defined in the sequel. Throughout the section $\bar{X} = \{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(\bar{N})}\}$ denotes the set of scenarios, where $\boldsymbol{x}^{(r)} = [(x_1^{(r)})^\top, \dots, (x_N^{(r)})^\top]^\top \in \mathbb{R}^{\sum_{i=1}^N n_i}$, for $r = 1, \dots, \bar{N}$. Relaxation variables $\boldsymbol{\zeta}$ are introduced to ensure feasibility, while $M > 0$ is a large enough penalty coefficient.

Program (SC-Verification) is a quadratic program, where all the constraints are linear based on the samples. If for any $\boldsymbol{x} \in \mathcal{B}$ and corresponding control input $\boldsymbol{u}(\boldsymbol{x})$, all the CBF constraints are satisfied, then $\boldsymbol{\zeta}^* = 0$. Conversely, $\boldsymbol{\zeta}^* \neq 0$ represents a CBF constraint violation, and indicates that the system is under a *risk* of being unsafe. A new set $\mathcal{Z}(\mathcal{B})$ for optimal solution $\boldsymbol{z}^*$ is defined as follows

$$\begin{aligned} &\boldsymbol{z}^* \in \mathcal{Z}(\mathcal{B}) \iff \\ &\sum_{i \in \mathcal{V}_e} h_{ie}(u_i(\boldsymbol{x})) \leq \sum_{i \in \mathcal{V}_e} z_{ie}^*, \forall e = 1, \dots, E, \forall \boldsymbol{x} \in \mathcal{B}. \end{aligned} \tag{11}$$

Then $\mathcal{Z}(\mathcal{B})$ is constituted of N individual set $\mathcal{Z}_i(\mathcal{B})$ as

$$\mathcal{Z}(\mathcal{B}) = \bigcap_{i=1}^N \mathcal{Z}_i(\mathcal{B}). \tag{12}$$

The argument of $\mathcal{Z}$ and $\mathcal{Z}_i$ is dropped in the sequel for simplicity.

B. Sampling the Scenarios

The scenarios are sampled independently from the invariant set $\mathcal{B}$. For sampling we define a probability density function $\pi(\boldsymbol{x})$ associated with set $\mathcal{B}$ that satisfies:

$$\int_{\mathcal{B}} \pi(\boldsymbol{x}) d\boldsymbol{x} = 1.$$

One typical choice of $\pi(\boldsymbol{x})$ is to set it according to the density of a uniform distribution, i.e., $\pi(x) = \pi^{\text{uni}}(\boldsymbol{x}) = \frac{1}{\int_{\mathcal{B}} d\boldsymbol{x}}$. For the existence of $\pi^{\text{uni}}(\boldsymbol{x})$, we make an assumption on $\mathcal{B}$.

Assumption 1. *$\mathcal{B}$ is bounded with non-zero Lebesgue measure.*

Then, $\boldsymbol{x}$ can be sampled for $\bar{N}$ times independently from the distribution $\pi^{\text{uni}}(\boldsymbol{x})$. We note here that different probability distributions have minor impact on the final violation results, interested readers are referred to [19, Section 3.1]. Although the uniform distribution here is well-defined, the uncertain set $\mathcal{B}$ is defined implicitly as an intersection of multiple individual invariant sets $\mathcal{B}_e$. Sampling a point from the proposed uniform distribution can be difficult in practice since each individual agent lacks a global sense to $\mathcal{B}$. Here, we provide a sequential algorithm to sample scenarios $\boldsymbol{x}^{(r)}$, $r = 1, \ldots, \bar{N}$.

Algorithm 2 Scenarios Sampling Algorithm

```
   **Initialization** Uncertain set $\mathcal{B}$, failure times $F = 0$.
   **Output:** Scenario $\boldsymbol{x}^{(r)}$.
1: Sample $x_1^{(r)}$ from $\pi_1(x_1)$.
2: **for** $i = 2, \ldots, N$ **do**
3:     Construct a distribution $\pi_i(\{x_1^{(r)}, \ldots, x_{i-1}^{(r)}\}; x_i)$ following (13) (14).
4:     **if** $\pi_i = 0$ **then**
5:         $F \leftarrow F + 1$.
6:         go to $i = i - F$ ($i = 1$ is step 1).
7:     **end if**
8:     Sample $x_i^r$ from $\pi_i(\{x_1^{(r)}, \ldots, x_{i-1}^{(r)}\}; x_i)$.
9: **end for**
```

The algorithm constructs the densities from which samples are extracted sequentially for each agent. We first show the construction of $\pi_i(\{x_1^{(r)}, \ldots, x_{i-1}^{(r)}\}; x_i)$. Here $\{x_1^{(r)}, \ldots, x_{i-1}^{(r)}\}$ are sampled according to the i-th density, π_i. To determine this, we first define the uncertain sets from which samples are extracted for agent i with part of the states of agents in the same sub-network $\mathcal{G}_e$ fixed.

$$\tilde{\mathcal{B}}_{ie} = \begin{cases} \mathcal{X}_i, \text{ if } \exists l \in \mathcal{V}_e, \text{such that } l > i, \\ \{x_i \in \mathbb{R}^{n_i} | b_{ie}(x_i, \{x_l^{(r)}\}) \geq 0)\}, \text{ otherwise.} \end{cases} \tag{13}$$

We then have that $\tilde{\mathcal{B}}_i = \bigcap\limits_{e \in \mathcal{C}_i} \tilde{\mathcal{B}}_{ie}$. The parameters in (13) can all be collected by local communication, since only states of agents in the same sub-network are required. Note here $\tilde{\mathcal{B}}_i$ is possibly empty with some parameters $\{x_1^{(r)}, \ldots, x_{i-1}^{(r)}\}$. The distribution $\pi_i(\{x_1^{(r)}, \ldots, x_{i-1}^{(r)}\}; x_i)$ used in the step 3 of Algorithm 2 is a uniform distribution over $\tilde{\mathcal{B}}_i$, i.e.,

$$\pi_i(\{x_1^{(r)}, \ldots, x_{i-1}^{(r)}\}; x_i) = \begin{cases} \frac{1}{\int_{\tilde{\mathcal{B}}_i} dx_i}, \text{ if } \tilde{\mathcal{B}}_i \neq \emptyset, \\ 0, \text{ otherwise.} \end{cases} \tag{14}$$

In step 1, the first scenario $x_1^{(r)}$ associated with Agent 1 is sampled from distribution $\pi_1(x_1) = \frac{1}{\int_{\mathcal{X}_1} dx}$, since now there are no other agents involved to restrict the uncertain set for agent 1. Then, the sampling-construction procedures repeat sequentially from agent 2 to agent N. One case needs additional attention, where $\pi_i = 0$. This implies that $\tilde{\mathcal{B}}_i = \emptyset$. There exists $\{x_1^{(r)}, \ldots, x_{i-1}^{(r)}\}$ such that $\tilde{\mathcal{B}}_i \neq \emptyset$. Therefore, if $\pi_i = 0$ (Step 4), then go back to the sampling-construction of agent $i - F$, $F \neq 1$ is to avoid a deadlock on step i. The deadlock happens when for given scenarios $x_1^{(r)}, \ldots, x_{i-2}^{(r)}$, the uncertain set $\tilde{\mathcal{B}}_{i-1}$ and distribution π_{i-1} is such that for any $x_{i-1}^{(r)} \in \tilde{\mathcal{B}}_{i-1}$, $\tilde{\mathcal{B}}_i = \emptyset$. It is guaranteed that $F \leq i - 1$ for $i \geq 2$, since $\tilde{\mathcal{B}}_1 = \mathcal{X}_1 \neq \emptyset$.

Proposition 1. *Suppose Assumption 1 hold. The scenarios $\boldsymbol{x}^{(r)}$, $r = 1, \ldots, \bar{N}$, are feasible, i.e. $\boldsymbol{x}^{(r)} \in \mathcal{B}$, and independent.*

Proof. The feasibility result holds directly from the definition of every uncertain set $\tilde{\mathcal{B}}_i$ in (13) from which $x_i^{(r)}$ is sampled. As a result, we have $b_{ie}(x_i^{(r)}, \{x_k^{(r)}\}) \geq 0$ for any $i = 1, \ldots, N$, $e \in \mathcal{C}_i$, and $k \in \mathcal{V}_e$. Therefore, $\boldsymbol{x}^{(r)} \in \mathcal{B}$. $\boldsymbol{x}^{(r)}$ for $r = 1, \ldots, \bar{N}$ are independent since for $r = 1, \ldots, \bar{N}$, $x_1^{(r)}$ are independently sampled from distribution π_1. □

We note here that the elements in $\boldsymbol{x}^{(r)}$ are correlated, but this will not influence the independence results in Proposition 1 since we seek independence across r.

C. Distributed Safety Verification

After sampling scenarios $\boldsymbol{x}^{(r)}$, $r = 1, \ldots, \bar{N}$ using Algorithm 2, we are at the stage of solving the safety verification program (SC-Verification).

Suppose the local cost function $J_i(\boldsymbol{z}_i, \boldsymbol{\zeta}_i)$, and constraint function $\hat{h}_{ie}(\boldsymbol{z}_i, \boldsymbol{\zeta}_i)$ are

$$\begin{aligned} J_i(\boldsymbol{z}_i, \boldsymbol{\zeta}_i) \quad &= \sum_{e \in \mathcal{C}_i} \left(z_{ie}^2 + M \sum_{r=1}^{\bar{N}} \zeta_{ie}^{(r)} \right), \\ \hat{h}_{ie}^{(r)}(\boldsymbol{z}_i, \boldsymbol{\zeta}_i) \quad &= h_{ie}(u_i(\boldsymbol{x}^{(r)})) - z_{ie} - \zeta_{ie}^{(r)}, r = 1, \ldots, \bar{N}. \end{aligned} \tag{15}$$

Then Algorithm 1 can be applied to solve the distributed scenario optimisation problem (SC-Verification). In the sequel, we use $\boldsymbol{z}^*$ and $\boldsymbol{\zeta}^*$ to represent the optimal solution to (SC-Verification), with scenarios $\boldsymbol{x}^{(r)}$, $r = 1, \ldots, \bar{N}$.

We then have the following theorem as the main result on probabilistic safety.

Theorem 1. *Suppose Assumption 1 hold. Choose $\beta_i \in (0, 1), i = 1, \ldots, N$, and set $\beta = \sum_{i=1}^{N} \beta_i$. For $i = 1, \ldots, N$, and $0 \leq s_i^* \leq \bar{N} - 1$, consider the polynomial* *equation in t_i*

$$\begin{aligned}&\begin{pmatrix}\bar{N}\\ s_i^*\end{pmatrix} t_i^{\bar{N}-s_i^*}-\frac{\beta_i}{2\bar{N}}\sum_{j=s_i^*}^{\bar{N}-1}\begin{pmatrix}j\\ s_i^*\end{pmatrix} t_i^{j-s_i^*}\\ &-\frac{\beta_i}{6\bar{N}}\sum_{j=\bar{N}+1}^{4\bar{N}}\begin{pmatrix}j\\ s_i^*\end{pmatrix} t_i^{j-s_i^*}=0,\end{aligned} \tag{16}$$

while for $s_i^*=\bar{N}$ consider the polynomial equation

$$1-\frac{\beta}{6N}\sum_{j=\bar{N}+1}^{4\bar{N}}\begin{pmatrix}j\\ s_i^*\end{pmatrix} t_i^{j-\bar{N}}=0. \tag{17}$$

For any $i=1,\dots,N$, this equation has exactly two solutions in $[0,+\infty)$ denoted by $\underline{t}_i(s_i^*)$ and $\bar{t}_i(s_i^*)$, where $\underline{t}_i(s_i^*)\le\bar{t}(s_i^*)$. Let $\bar{\epsilon}_i(s_i^*):=1-\underline{t}_i(s_i^*)$, $\bar{\epsilon}(s^*)=\min\{\sum_{i=1}^N\bar{\epsilon}_i(s_i^*),1\}$. We then have that

$$\mathbb{P}^{\bar{N}}\left\{\mathbb{P}\left\{\boldsymbol{x}\in\mathcal{B}:0\notin\mathcal{Z}\right\}\le\bar{\epsilon}(s^*)\right\}\ge 1-\beta, \tag{18}$$

where s_i^* is the number of non-zero $\zeta_{ie}^{(r)*}$, $e\in\mathcal{C}_i$.

Proof. See [1, Theorem 5]. □

The safety verification algorithm is shown in Algorithm 3. For initialization, agents need to sample $\bar{N}$ independent scenarios using Algorithm 2, and select a confidence level. After collecting all the scenarios, agents solve the scenario program (SC-Verification) locally using Algorithm 1. For each agent, solving the verification program is to count the number of violated CBF constraints over the scenarios. The number of constraint violations is denoted by s_i^*. The local probabilistic result for the upper bound $\epsilon_i(s_i^*)$ is calculated by every agent using Theorem 3. In the final step, agents receive the upper bound $\bar{\epsilon}_j(s_j^*)$ and confidence parameter β_j from every $j\in\mathcal{N}_i$, in order to obtain the safety verification result for the entire system.

Algorithm 3 Distributed Safety Verification Algorithm

```
Initialization Sample $\bar{N}$ scenarios $x^{(r)}$ using Algorithm 2, every agent selects a confidence $\beta_i$
Output: The probabilistic result (18)
1: for every agent $i$ do
2:     Solve the scenario program (SC-Verification) locally using Algorithm 1.
3:     Calculate $s_i^*$, which is the number of non-zero $\zeta_{ie}^*$, for $e\in\mathcal{C}_i$.
4:     Calculate $\bar{\epsilon}_i(s_i^*)$ using Theorem 1.
5:     Receive $\bar{\epsilon}_j(s_j^*)$ and $\beta_j$ from $j\in\mathcal{N}_i$, compute $\bar{\epsilon}(s^*)=\sum_{i=1}^N(\bar{\epsilon}_i(s_i^*))$, $\beta=\sum_{i=1}^N\beta_i$.
6: end for
```

IV. Simulation Results

We evaluated the performance of our distributed safety verification algorithm on a standard distributed safe controller design algorithm [24], which is based on solving parallel CBF-QP problems. The main idea of this algorithm is to split the CBF constraints for every agent to acquire a local computation. However, this direct split may result in feasibility issues, especially for heterogeneous systems and large-scale networked systems. We consider the same multi-robot position swapping problem as in [24].

We consider a group of four robots with double integrator dynamics

$$\begin{bmatrix}\dot{\boldsymbol{p}}_i\\ \dot{\boldsymbol{v}}_i\end{bmatrix}=\underbrace{\begin{bmatrix}0 & I_{2\times2}\\ 0 & 0\end{bmatrix}}_{A_i}\begin{bmatrix}\boldsymbol{p}_i\\ \boldsymbol{v}_i\end{bmatrix}+\underbrace{\begin{bmatrix}0\\ I_{2\times2}\end{bmatrix}}_{B_i}\boldsymbol{a}_i, \tag{19}$$

where $\boldsymbol{p}_i\in\mathbb{R}^2$, $\boldsymbol{v}_i\in\mathbb{R}^2$ represent positions and velocities, and $\boldsymbol{a}_i\in\mathbb{R}^2$ is the control input, representing accelerations. The working space is restricted to be $||\boldsymbol{p}_i||_\infty\le x^{\max}$, $||\boldsymbol{v}_i||_\infty\le v_i^{\max}$, $||\boldsymbol{a}_i||_\infty\le a_i^{\max}$. Each robot is regarded as a disk centered at $\boldsymbol{p}_i$ with radius $D_i\in\mathbb{R}_+$. The safety certificate $s_{ij}(\boldsymbol{p}_{ij},\boldsymbol{v}_{ij})$ for collision avoidance between robot i and j is defined by

$$s_{ij}(\boldsymbol{p}_{ij})=||\Delta\boldsymbol{p}_{ij}||_2^2-D_{ij}, \tag{20}$$

where $\Delta\boldsymbol{p}_{ij}=\boldsymbol{p}_i-\boldsymbol{p}_j$, $D_{ij}=D_i+D_j$. Note here that the system is heterogeneous as different robots have different mobility. The control barrier function for invariance certificates is then defined pair-wisely, as

$$\begin{aligned}b_{ij}(\boldsymbol{p}_{ij},\boldsymbol{v}_{ij})=&\sqrt{2(a_i^{\max}+a_j^{\max})(||\Delta\boldsymbol{p}_{ij}||_2^2-D_{ij})}\\ &+\frac{\Delta\boldsymbol{p}_{ij}^\top}{||\Delta\boldsymbol{p}_{ij}||_2^2}\Delta\boldsymbol{v}_{ij},\end{aligned} \tag{21}$$

where $\Delta\boldsymbol{v}_{ij}=\boldsymbol{v}_i-\boldsymbol{v}_j$. The function $b_{ij}(\boldsymbol{p}_{ij},\boldsymbol{v}_{ij})$ is guaranteed to be a CBF since when $b_{ij}(\boldsymbol{p}_{ij},\boldsymbol{v}_{ij})>0$, collision can be avoided with a maximum braking acceleration $\boldsymbol{a}_i^{\max}+\boldsymbol{a}_j^{\max}$ applied to robots i and j. For $i=1,2$, $\boldsymbol{a}_i^{\max}=1$, while for $i=3,4$, $\boldsymbol{a}_i^{\max}=10$, $D_{ij}=0.3$. At every given state $\boldsymbol{p},\boldsymbol{v}$, for agent i and j, the CBF-QP for controller design is formulated by

$$\begin{aligned}\min_{\boldsymbol{a}_i,\boldsymbol{a}_j}\ & ||\boldsymbol{a}_i||^2+||\boldsymbol{a}_j||^2\\ \text{s.t. } & \dot{b}_{ij}(\boldsymbol{p}_{ij},\boldsymbol{v}_{ij})+\alpha_{ij}(b_{ij}(\boldsymbol{p}_{ij},\boldsymbol{v}_{ij}))\ge0,\\ & ||\boldsymbol{a}_i||_\infty\le a_i^{\max},||\boldsymbol{a}_j||_\infty\le a_j^{\max}.\end{aligned} \tag{22}$$

The centralized QP problem (22) is split into two problems

$$\begin{aligned}\min_{\boldsymbol{a}_i,\rho}& ||\boldsymbol{a}_i||^2+M\rho^2\\ \text{s.t.}&\frac{\partial b_{ij}}{\partial(\boldsymbol{p}_{ij},\boldsymbol{v}_{ij})}A_i\begin{bmatrix}\boldsymbol{p}_i\\ \boldsymbol{v}_i\end{bmatrix}+B_i\boldsymbol{a}_i+\frac{\alpha_{ij}(b_{ij}(\boldsymbol{p}_{ij},\boldsymbol{v}_{ij}))}{2}\ge\rho,\\ & ||\boldsymbol{a}_i||_\infty\le a_i^{\max},\rho\le0.\end{aligned} \tag{23}$$

$$\begin{aligned}\min_{\boldsymbol{a}_j,\rho}& ||\boldsymbol{a}_j||^2+M\rho^2\\ \text{s.t.}&\frac{\partial b_{ij}}{\partial(\boldsymbol{p}_{ij},\boldsymbol{v}_{ij})}A_j\begin{bmatrix}\boldsymbol{p}_j\\ \boldsymbol{v}_j\end{bmatrix}+B_j\boldsymbol{a}_j+\frac{\alpha_{ij}(b_{ij}(\boldsymbol{p}_{ij},\boldsymbol{v}_{ij}))}{2}\ge\rho,\\ & ||\boldsymbol{a}_j||_\infty\le a_j^{\max},\rho\le0,\end{aligned} \tag{24}$$

where ρ are non-positive relaxation variables, M is a large enough penalty coefficient. Hopefully ρ is zero, such that (22) holds with the controllers $\boldsymbol{a}_i$ and $\boldsymbol{a}_j$ solved by (23) and (24). However, in our case $\boldsymbol{a}_i^{\max} \neq \boldsymbol{a}_j^{\max}$, which may raise a feasibility issue.

To verify the safety for this approach, we independently sample 200 states $[\boldsymbol{p}, \boldsymbol{v}]$ from the invariant set, which is $\bigcap_{i,j}\{\boldsymbol{p}_{ij}, \boldsymbol{v}_{ij} \ : \ b_{ij}(\boldsymbol{p}_{ij}, \boldsymbol{v}_{ij}) \ \geq \ 0\}$. We also restrict the working space to be $x^{\max} = 4$, $v^{\max} = 3$ to avoid the trivial result that the robots are initialized to be far away. We sample 200 scenarios for safety verification, with every agent holds a confidence parameter $\beta_i = 0.025$, for $i = 1, \ldots, 4$. $s_1^* = 7$, $s_2^* = 8$, $s_3^* = 7$ and $s_4^* = 6$. Based on these information, we obtain $\bar{\epsilon}(s^*) = 0.3811$ and $1-\beta = 0.9$. Therefore, the safety verification result is

$$\mathbb{P}^{\bar{N}}\{\mathbb{P}\{\boldsymbol{x} \in \mathcal{B} : 0 \notin \mathcal{Z}\} \leq 0.3811\} \geq 0.9. \qquad (25)$$

We then sample 100 groups of 200 scenarios to validate the obtained result. The empirical cumulative distribution function for the probability of safety violation is plotted in Figure 1 by Monte-Carlo experiments. For each group, we can calculate the number of violations for the violation probability. Collecting these statistics for all the groups, we can then calculate the empirical cumulative distribution function for the violation probability. It can be observed that the probability of $\mathbb{P}\{\boldsymbol{x} \in \mathcal{B} : 0 \notin \mathcal{Z}\} \leq 0.3811$ is close to 1, which validates our obtained result.

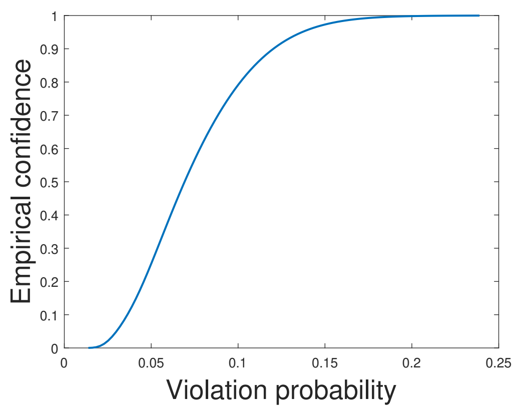

Fig. 1. Empirical cumulative distribution function for safety violation.

V. Conclusion

In this paper, we address the problem of safety verification for a multi-agent networked system using control barrier functions. We propose a scenario-based verification program that can be solved by each agent. Scenarios are independently sampled from the invariant set to evaluate safety. Our distributed safety verification program computes an upper bound on the probability of being unsafe. We conduct simulations to verify the safety of a classical distributed CBF-based controller design algorithm. In future work, we will focus on obtaining a tighter bound for the probability of being unsafe, as well as studying how the choice of the class-$\mathcal{K}$ function influences the result.

References

[1] H. Wang, A. Papachristodoulou, and K. Margellos, "Distributed control design and safety verification for multi-agent systems," *arXiv preprint arXiv:2303.12610*, 2023.

[2] F. Ding, J. He, Y. Ren, H. Wang, and Y. Zheng, "Configuration-aware safe control for mobile robotic arm with control barrier functions," *arXiv preprint arXiv:2204.08265*, 2022.

[3] H. Wang, Y. Li, W. Yu, J. He, and X. Guan, "Moving obstacle avoidance and topology recovery for multi-agent systems," in *American Control Conference (ACC)*, pp. 2696–2701, IEEE, 2019.

[4] J. Axelsson, "Safety in vehicle platooning: A systematic literature review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1033–1045, 2016.

[5] A. Alam, A. Gattami, K. H. Johansson, and C. J. Tomlin, "Guaranteeing safety for heavy duty vehicle platooning: Safe set computations and experimental evaluations," *Control Engineering Practice*, vol. 24, pp. 33–41, 2014.

[6] W. Xiao and C. G. Cassandras, "Decentralized optimal merging control for connected and automated vehicles with safety constraint guarantees," *Automatica*, vol. 123, p. 109333, 2021.

[7] K. Margellos and J. Lygeros, "Hamilton–jacobi formulation for reach–avoid differential games," *IEEE Transactions on automatic control*, vol. 56, no. 8, pp. 1849–1861, 2011.

[8] J. Lygeros, "On reachability and minimum cost optimal control," *Automatica*, vol. 40, no. 6, pp. 917–927, 2004.

[9] C. J. Tomlin, J. Lygeros, and S. S. Sastry, "A game theoretic approach to controller design for hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 949–970, 2000.

[10] S. Prajna and A. Jadbabaie, "Safety verification of hybrid systems using barrier certificates," in *International Workshop on Hybrid Systems: Computation and Control*, pp. 477–492, Springer, 2004.

[11] S. Prajna, A. Jadbabaie, and G. J. Pappas, "A framework for worst-case and stochastic safety verification using barrier certificates," *IEEE Transactions on Automatic Control*, vol. 52, no. 8, pp. 1415–1428, 2007.

[12] S. Prajna, A. Papachristodoulou, and P. A. Parrilo, "Introducing sostools: A general purpose sum of squares programming solver," in *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.*, vol. 1, pp. 741–746, IEEE, 2002.

[13] S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo, "Sostools: Sum of squares optimization toolbox for matlab," 2004.

[14] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2016.

[15] M. C. Campi and S. Garatti, "The exact feasibility of randomized solutions of uncertain convex programs," *SIAM Journal on Optimization*, vol. 19, no. 3, pp. 1211–1230, 2008.

[16] M. C. Campi and S. Garatti, "Wait-and-judge scenario optimization," *Mathematical Programming*, vol. 167, no. 1, pp. 155–189, 2018.

[17] G. Calafiore and M. C. Campi, "Uncertain convex programs: randomized solutions and confidence levels," *Mathematical Programming*, vol. 102, no. 1, pp. 25–46, 2005.

[18] G. C. Calafiore and M. C. Campi, "The scenario approach to robust control design," *IEEE Transactions on automatic control*, vol. 51, no. 5, pp. 742–753, 2006.

[19] S. Garatti and M. C. Campi, "Risk and complexity in scenario optimization," *Mathematical Programming*, pp. 1–37, 2019.

[20] P. Akella and A. D. Ames, "A barrier-based scenario approach to verifying safety-critical systems," *IEEE Robotics and Automation Letters*, 2022.

[21] A. D. Ames, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to adaptive cruise control," in *53rd IEEE Conference on Decision and Control*, pp. 6271–6278, IEEE, 2014.

[22] H. Wang, K. Margellos, and A. Papachristodoulou, "Safety verification and controller synthesis for systems with input constraints," *arXiv preprint arXiv:2204.09386*, 2022.

[23] I. Notarnicola and G. Notarstefano, "Constraint-coupled distributed optimization: A relaxation and duality approach," *IEEE Transactions on Control of Network Systems*, vol. 7, no. 1, pp. 483–492, 2019.

[24] L. Wang, A. D. Ames, and M. Egerstedt, "Safety barrier certificates for collisions-free multirobot systems," *IEEE Transactions on Robotics*, vol. 33, no. 3, pp. 661–674, 2017.