CS 498: Shield Synthesis of Cyber Physical Systems

Synthesis of Minimum-Cost Shields for Multi-agent Systems

Presented by:
Samay Varshney (180101097)
Siddhartha Jain (180101078)

Introduction

The use of multi-agent systems such as teams of unmanned aerial vehicles (UAVs) has been predicted to grow significantly in many areas of our life.

In the private sector, drones are being used in applications from package delivery to inventory management in warehouses and can be considered as a multi-agent system.

Individual agents have to not only fulfill their local objectives and meet their local requirements, but also abide by systemwide or global safety requirements such as avoiding collision with other agents.

Need for Shielding

There has been a large body of work in designing algorithms to perform agent coordination and task assignment for a wide array of applications.

For example, software frameworks such as UxAS provide mission-level autonomy for multi-agent systems and include capabilities from high level task assignment to path planning for unmanned systems. Such frameworks often allow for dynamic task reallocation as missions change, but in doing so, cannot necessarily account for potential violations of global safety specifications.

This necessitates shielding the agents at runtime from a possible task assignment that can cause a violation of a global safety specification.

Shield Synthesis

Shield synthesis is a general method to automatically derive runtime enforcement implementations, called shields, from temporal logical specifications.

A shield is attached to a reactive system, monitors the behaviour of the system (i.e., its inputs and outputs), and corrects erroneous outputs instantaneously, but only if necessary and as infrequently as possible.

Shield Synthesis for multi-agent systems

A shield monitors, and if needed corrects the output of one or more agents in the system, such that a given global safety specification is always satisfied.

In order to explore the design space of possible shields for multi-agent systems, we categorize shields based on three criteria:

- **Interference of the shield processes with the individual agents:** One shield might be prefered over another, which can be decided on basis of interference cost in order to quantify the quality of a shield. Aim will be to synthesize cost optimal shields that minimize the interference cost for the worst-case behavior of the multi-agent system.

Shield Synthesis for multi-agent systems (cont..)

- **Assumptions on the multi-agent system:** A shield has to guarantee safety for any possible implementation of the multi-agent system or specifications of each of the agents. However, it is often realistic to make assumptions on the worst-case behavior of the system and synthesize optimal shields w.r.t. the chosen interference cost under these assumptions.

- **The fairness of the shield w.r.t. the individual agents:** It is often important that a shield treats all agents fairly: in case of an error, a fair shield does not always interfere with the same agent repeatedly. Fairness notion for shields is defined to tackle it.

Related Work

In the context of enforcement, in case of an error the previous approaches either stop system execution or suppress, insert, or delay actions.

Approach that generates monitors that detect and prevent violations by one-step lookahead has a limitation to cause deadlocks.

An alternative approach circumvents this problem by using model checking to determine for each event whether it should be blocked, which is done online. In contrast, shields are synthesized offline, accounting for the effect of the shield on the future execution.

None of these mentioned, nor previous works on shield synthesis, considers quantitative specifications about how violations should be mitigated.

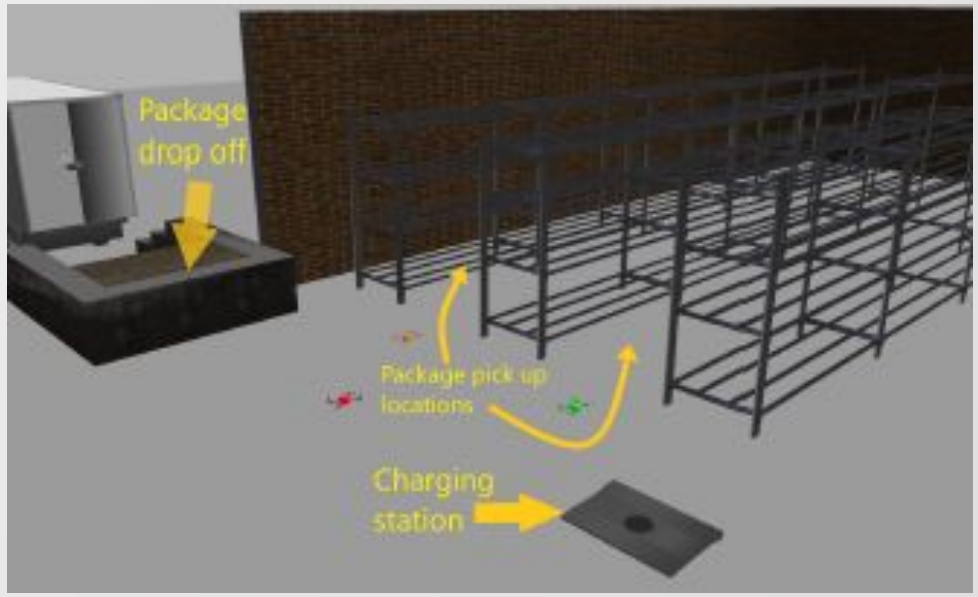
Case Study

Input: The commands sent to move package(s) from the shelves to the loading dock which will generate task assignments for the UAVs.

Global Requirements: Not allowing more than one UAV in a row at same time, not allowing them to fly too close to each other, and not allowing them to charge or drop of packages at the same time.

Shield takes task assignment as input and overwrites it with new task assignment when necessary.

The figure is a snapshot of a warehouse environment where packages need to be moved from shelves to a loading area to be sent for delivery. UAVs also need to periodically visit a charging station.



Preliminaries: Basic notations

Input: I, Input Alphabet: $\Sigma_I = 2^I$
Output: O, Output Alphabet: $\Sigma_O = 2^O$
Alphabet: $\Sigma = \Sigma_I \times \Sigma_O$

Set of finite (infinite) words (also k/a execution traces) over Σ is denoted by Σ^* (Σ^w).

Length: $|\bar{\sigma}|$ for the trace $\bar{\sigma} \in \Sigma$.

For $\bar{\sigma}_I = x_0x_1.... \in \Sigma_I^\infty$ and $\bar{\sigma}_O = y_0y_1... \in \Sigma_O^\infty$, $(\bar{\sigma}_I \| \bar{\sigma}_O)$ is the composition $(x_0, y_0)(x_1, y_1)... \in \Sigma^\infty$

For $i \in N$ and a word $\bar{\sigma} = \sigma_0\sigma_1...$

- $\bar{\sigma}[i] = \sigma_i$
- $\bar{\sigma}[i; j) = \sigma_i\sigma_{i+1}...\sigma_{j-1}$ if $j \in N$ and $\sigma[i; j) = \sigma_i\sigma_{i+1}...$ if $j = \infty$

Language: Set $L \subseteq \Sigma^\infty$ of words.

Preliminaries: Reactive System

Each agent in the multi-agent system, as well as the shield, is a reactive system which is defined by a 6-tuple $P = (Q, q_0, \Sigma_I, \Sigma_O, \delta, \lambda)$, where

- Q is a finite set of states
- $q_0 \in Q$ is the initial state
- Σ_I is the input alphabet
- Σ_O is the output alphabet
- $\delta : Q \times \Sigma_I \rightarrow Q$ is the complete transition function
- $\lambda : Q \times \Sigma_I \rightarrow \Sigma_O$ is the output function

Given an input trace $\bar{\sigma}_I = x_0 x_1 ... \in \Sigma_I^\infty$ a reactive system P produces an output trace $\bar{\sigma}_O = P(\bar{\sigma}_I) = \lambda(q_0, x_0)\lambda(q_1, x_1)... \in \Sigma_O^\infty$ with $q_{i+1} = \delta(q_i, x_i)$ for all $i \geq 0$.

The set of words produced by P is denoted $L(P) = \{ \bar{\sigma}_I \,||\, \bar{\sigma}_O \in \Sigma^\infty \mid P(\bar{\sigma}_I) = \bar{\sigma}_O \}$.

Preliminaries: Multi-agent reactive systems

A multi-agent reactive system D is a tuple (P, Σ_I, Σ_O), where $P = \{P_1, \ldots, P_n\}$ is a set of agents, where each $P_i = (Q_i, q_{0,i}, I, \Sigma_{I,i}, \Sigma_{O,i}, \delta_i, \lambda_i)$ is a reactive system.

The outputs of the multiagent system D are $O = \bigcup^n_{i=1} O_i$ and its inputs are $I = \bigcup^n_{i=1} I_i$.

The multiagent system is a reactive system $D = (Q, q_0, \Sigma_I, \Sigma_O, \delta, \lambda)$ defined as follows:

- The set $Q = \bigotimes_i Q_i$ of states is formed by the product of the states of all agents $P_i \in P$
- The initial state q_0 is formed by the initial states $q_{0,i}$ of all $P_i \in P$
- The transition function δ updates, for each agent $P_i \in P$, the Q_i part of the state in accordance with the transition function δ_i, using the projection $\sigma(I_i)$ as input.
  $\delta(q_0, \sigma) = (\delta_0(q_{0,0}, \sigma(I_0)), \delta_1(q_{0,1}, \sigma(I_1)), \delta_2(q_{0,2}, \sigma(I_2)) \ldots . \delta_n(q_{0,n}, \sigma(I_n)))$
- The output function λ labels each state with the union of the outputs of all $P_i \in P$ according to λ_i.
  $\lambda(q_0, \sigma) = \bigcup^n_{i=1} \lambda_i(q_{0,i}, \sigma(I_i))$

Preliminaries: Specification

A specification φ defines a set $L(\varphi) \subseteq \Sigma^{\infty}$ of allowed traces.

A reactive system D realizes φ, denoted by $D \models \varphi$, iff $L(D) \subseteq L(\varphi)$.

φ is called a safety specification if every trace $\bar{\sigma}$ that is not in L(φ) has a prefix τ such that all words starting with τ are also not in the language L(φ).

Safety specification φ is represented by a safety automaton $\varphi = (Q, q_0, \Sigma, \delta, F)$, where $F \subseteq Q$ is a set of safe states.

Preliminaries: Games

A game is a tuple $\mathcal{G} = (G, g_0, \Sigma, \delta, Acc, Val)$, where

- G is a finite set of states
- $g_0 \in G$ is the initial state
- $\delta: G \times \Sigma \rightarrow G$ is a complete transition function
- qualitative objective of the game: $Acc: (G \times \Sigma \times G)^w \rightarrow B$ is a winning condition
- quantitative objective of the game: $Val: (G \times \Sigma \times G)^w \rightarrow R$ is a value

The game is played by two players: the system and the environment.

In every state $g \in G$ (starting with g_0), the environment chooses an input $\sigma_I \in \Sigma_I$ and then the system chooses some output $\sigma_O \in \Sigma_O$.

These choices define the next state $g' = \delta(g, (\sigma_I, \sigma_O))$.
The resulting(infinite) sequence $\bar{\pi} = (g_0, \sigma_I, \sigma_O, g_1)(g_1, \sigma_I, \sigma_O, g_2)...$ is called a play.

A play $\bar{\pi}$ is won by the system iff $Acc(\bar{\pi}) = \top$.

Preliminaries: Games (continue..)

A deterministic strategy for the environment is a function $p_e : G^* \rightarrow \Sigma_I$.
A nondeterministic (deterministic) strategy for the system is a relation $p_s : G^* \times \Sigma_I \rightarrow 2^{\Sigma_O}$ (function $p_s : G^* \times \Sigma_I \rightarrow \Sigma_O$).

A strategy is winning for the system if all plays $\bar{\pi}$ that can be constructed when defining the outputs using the strategy result in $Acc(\bar{\pi}) = \top$.

The winning region **Win** is the set of states from which a winning strategy exists.

A permissive winning strategy $p_s : G^* \times \Sigma_I \rightarrow 2^{\Sigma_O}$ is a strategy that is not only winning for the system, but also contains all deterministic winning strategies.

A safety game defines Acc via a set $F \subseteq G$ of safe states: $Acc(\bar{\pi}) = \top$ iff $g_i \in F$ for all $i \geq 0$, i.e., if only safe states are visited in the play $\bar{\pi}$. Otherwise, $Acc(\bar{\pi}) = \bot$.

The quantitative objective of the system is to minimize $Val(\bar{\pi})$, while the environment tries to maximize it.

Preliminaries: Properties of Traces

A finite trace $\bar{\sigma} \in \Sigma^*$ is wrong w.r.t. a specification φ, if the corresponding play cannot be won, i.e., if there is no way for the system to guarantee that any extension of $\bar{\sigma}$ satisfies φ.

An output σ_O is called wrong for a trace $\bar{\sigma}$ and input σ_I, if it makes the trace wrong, i.e. when $\bar{\sigma}$ is not wrong, but $\bar{\sigma}.(\sigma_I, \sigma_O)$ is.

Given a sequence $(\bar{\sigma_I} \,||\, \bar{\sigma_O} \,||\, \bar{\sigma_O}') \in (\Sigma_I \times \Sigma_O \times \Sigma_O)^\infty$, we denote with $\mathrm{WIdx}(\bar{\sigma_I} \,||\, \bar{\sigma_O} \,||\, \bar{\sigma_O}')$, the positions of occurrences of wrong outputs in $\bar{\sigma_O}$.

Formally, $i \in \mathrm{WIdx}(\bar{\sigma_I} \,||\, \bar{\sigma_O} \,||\, \bar{\sigma_O}')$ iff $\bar{\sigma_O}[i]$ is wrong for the trace $(\bar{\sigma_I}[0; i) \,||\, \bar{\sigma_O}[0; i))$ and the input $\bar{\sigma_I}[i]$.

We denote with $\mathrm{Agts} = \{1, \ldots, n\}$ the set of agent ids of a multi-agent system D. For a set $\Pi \subseteq \mathrm{Agts}$, we define $O_\Pi = \bigcup_{i \in \Pi} O_i$ and $\Sigma_{O\Pi} = 2^{O\Pi}$. For $\sigma_O \in \Sigma_O$ and $i \in \mathrm{Agts}$, we denote with $\sigma_O(O_i)$ the projection of σ_O on O_i. For $\Pi \subseteq \mathrm{Agts}$, we define $\sigma_O(O_\Pi)$ similarly.

Preliminaries: Properties of Traces (cont..)

For $\sigma_O, \sigma_O' \in \Sigma_O$, the set $Diff(\sigma_O, \sigma_O') = \{ i \in Agts \mid \sigma_O(O_i) \neq \sigma_O'(O_i) \}$ gives the set of agents whose outputs in σ_O differ from those in σ_O'.

Let $(\sigma_O, \sigma_O') \in (\Sigma_O, \Sigma_O)^\infty$ be a sequence of output pairs.

Then (σ_O, σ_O') is called a deviation period if

- $\bar{\sigma_O}[i] \neq \bar{\sigma_O}'[i]$ for every $i < |\bar{\sigma_O}|$ and
- if $|\bar{\sigma_O}| < \infty$, then $\bar{\sigma_O}[|\bar{\sigma_O}|] = \bar{\sigma_O}'[|\bar{\sigma_O}|]$

Multi Agent Systems: Attaching the Shield

A shield $S = (Q, q_0, \Sigma_I \times \Sigma_O, \Sigma_O, \delta, \lambda)$ is a reactive system that is attached to a multi-agent system $D = (\{ P_1,...,P_n \}, \Sigma_I, \Sigma_O)$.

Since S has to enforce a global specification that can refer to all inputs and outputs of D, S is attached to the whole multi-agent system D using serial composition: the shield S monitors the outputs of all the agents and corrects them if necessary.

Given $D = (\{ P_1,...,P_n \}, \Sigma_I, \Sigma_O)$, with $P_i = (Qi, q_{0,i}, \Sigma_{I,i}, \Sigma_{O,i}, \delta_i, \lambda_i)$, the serial composition of D and S is a reactive system $DoS = (Q^{\wedge}, q_0^{\wedge}, \Sigma_I, \Sigma_O, \delta^{\wedge}, \lambda^{\wedge})$, with

states $Q^{\wedge} = \bigotimes_i Q_i \times Q$

$q_0^{\wedge} = (q_{0,1}, \ldots, q_{0,n}, q_0)$

transition function $\delta^{\wedge}((q_1, \ldots, q_n, q), \sigma_I) = (\delta_1(q_1, \sigma_{I1}), \ldots, \delta_n(q_n, \sigma_{I,n}), \delta(q, (\sigma_I, \sigma_O))$
where σ_O is the output of all agents $\sigma_O = (\lambda_1(q_1, \sigma_{I1}), \ldots, \lambda_n(q_n, \sigma_{In}))$

output function $\lambda^{\wedge}((q_1, \ldots, q_n, q), \sigma_I) = \lambda(q, (\sigma_I, \sigma_O))$.

Multi Agent Systems: Shield Definition

A shield must satisfy the following basic requirements:

- **Correctness:** A reactive system $S = (Q, q_0, \Sigma_I \times \Sigma_O, \Sigma_O, \delta, \lambda)$ ensures correctness with respect to a safety specification φ if for any multi-agent system $D = (\{ P_1,...,P_n \}, \Sigma_I, \Sigma_O)$ it holds that $(D \circ S) \models \varphi$.

- **No unnecessary interference:** A shield is only allowed to interfere when the output of the multi-agent system is wrong. Formally, given a safety specification φ, a reactive system $S = (Q, q_0, \Sigma_I \times \Sigma_O, \Sigma_O, \delta, \lambda)$ does not interfere unnecessarily if for any multi-agent system $D = (\{ P_1,...,P_n \}, \Sigma_I, \Sigma_O)$ and any trace $(\vec{\sigma_I} \,||\, \vec{\sigma_O}) \in (\Sigma_I \times \Sigma_O)^\infty$ of D that is not wrong, we have that $S(\vec{\sigma_I} \,||\, \vec{\sigma_O}) = \vec{\sigma_O}$.

Multi-agent systems: Interference cost functions

An ***interference cost function c*** : $\Sigma_O \times \Sigma_O \rightarrow \mathbb{N}$ assigns a cost $c(\sigma_O, \sigma_O')$ to each pair of system output $\sigma_O \in \Sigma_O$ and shield output $\sigma_O' \in \Sigma_O$, such that $c(\sigma_O, \sigma_O') = 0$ if $\sigma_O = \sigma_O'$ and $c(\sigma_O, \sigma_O') = f(\sigma_O, \sigma_O') > 0$ if $\sigma_O = \sigma_O'$ where f is a function chosen by the system designer.

The higher the cost, the more undesirable is the corresponding way of interference by the shield.

We propose two concrete cost functions:

- The ***boolean cost function*** c_B : $\Sigma_O \times \Sigma_O \rightarrow \{0, 1\}$ considers the multi-agent system as a monolithic system, in which $c_B(\sigma_O, \sigma_O') = 0$ if $\sigma_O = \sigma_O'$ and $c_B(\sigma_O, \sigma_O') = 1$ if $\sigma_O = \sigma_O'$.

- The ***counting cost function*** $c_\#$: $\Sigma_O \times \Sigma_O \rightarrow \mathbb{N}$, takes into account the number of agents whose output is modified in which $c_\#(\sigma_O, \sigma_O') = |\{p \in Agts \mid \sigma_O(O_p) \neq \sigma_O'(O_p)\}|$.

Let $c : \Sigma_O \times \Sigma_O \rightarrow \mathbb{N}$ be a cost function. We define the ***accumulated interference cost function*** $c_{acc} : \Sigma^\infty_O \times \Sigma^\infty_O \rightarrow \mathbb{N}$ based on the given cost function c by $c_{acc}(\bar{\sigma_O}, \bar{\sigma_O}') = \sum_{i=0}^{|\sigma o|} c(\bar{\sigma_O}[i], \bar{\sigma_O}'[i])$

Multi-agent systems: Shield optimization objective

In this objective, the task of the shield is to minimize the worst-case accumulated cost for ending the deviation period.

Let S be a shield for a safety specification φ, and let $\bar{\sigma} \in \Sigma^*$ be a finite trace.
We define Dev($\bar{\sigma}$, S) (set of maximal deviation periods) to be the set of all deviation periods that extend the trace ($\bar{\sigma}$ || S($\bar{\sigma}$)), and result from outputs of S.

The set Dev($\bar{\sigma}$, S) consists of all finite or infinite deviation periods resulting from possible future behaviours of the multi-agent system and the environment.

Let $c : \Sigma_O \times \Sigma_O \rightarrow \mathbb{N}$ be a cost function. A shield S is ***locally optimal*** w.r.t. c, if for every shield S' and every trace $\bar{\sigma} = (\bar{\sigma}_I \,||\, \bar{\sigma}_O) \in \Sigma^*$ it holds that

$$\sup_{(\bar{\sigma}o \,||\, \bar{\sigma}o') \in Dev(\bar{\sigma}, S)} c_{acc}(\bar{\sigma}_O, \sigma_O') \leq \sup_{(\bar{\sigma}o \,||\, \bar{\sigma}o') \in Dev(\bar{\sigma}, S')} c_{acc}(\bar{\sigma}_O, \sigma_O')$$

Multi Agent Systems: Occurrences of Faults

Mostly, we have some knowledge about the system which can be used to make assumptions about its worst-case behavior and synthesize cost optimal shields under these assumptions.

Since each agent has to satisfy its own objective, the design of a single agent often neglects some global safety requirements. However, it is often realistic to assume that the length of all sequences of wrong outputs is bounded. This assumption can be made e.g. if the agents interact only rarely, and when they do, they interact only for a bounded period of time.

Assumption(b) $\subseteq (\Sigma \times \Sigma_O \times \Sigma_O)^\infty$: Set of traces in which the length of each sequence of wrong outputs does not exceed a given bound b.

We incorporate assumptions on the system in the definition of cost-optimal shields. We modify definition of locally-optimal shields by restricting the sets of deviation periods to those corresponding to traces in Assumption $\subseteq (\Sigma \times \Sigma_O \times \Sigma_O)^\infty$.

Multi Agent Systems: Fair Shielding

The definition of fair shields uses the notion of minimal correcting sets: a minimal correcting set for a wrong trace is the minimal set of agents such that modifying some of the outputs of these agents results in correct output.

Let $\bar{\sigma} = (\bar{\sigma_I} \,||\, \bar{\sigma_O}).(\sigma_I, \sigma_O) \in (\Sigma_I \times \Sigma_O)^*$ be a wrong trace.
A set $\pi \subseteq Agts$ is a **minimal correcting set** for $\bar{\sigma}$ if and only if the following conditions are satisfied:

- there exists $\sigma_O' \in \Sigma_O$ such that $Diff(\sigma_O, \sigma_O') = \pi$ and the trace $(\bar{\sigma_I} \,||\, \bar{\sigma_O}).(\sigma_I, \sigma_O')$ is not wrong.
- for all $\sigma_O' \in \Sigma_O$ such that $Diff(\sigma_O, \sigma_O') \subsetneq \pi$ and the trace $(\bar{\sigma_I} \,||\, \bar{\sigma_O}).(\sigma_I, \sigma_O')$ is wrong.

$Mcs(\bar{\sigma})$ is the set of all minimal correcting sets for $\bar{\sigma}$.

Multi Agent Systems: Fair Shielding

Given a trace $\bar{\sigma} = (\bar{\sigma_I} \,||\, \bar{\sigma_O} \,||\, \bar{\sigma_O}') \in (\Sigma_I \times \Sigma_O \times \Sigma_O)^*$ and an agent p,

Alt($\bar{\sigma}$, p) is the set of positions in $\bar{\sigma}$ where there exist both a minimal correcting set containing p and a minimal correcting set that does not contain p.

A shield S is **fair** if for every agent $p \in$ Agts and trace $\bar{\sigma} = (\bar{\sigma_I} \,||\, \bar{\sigma_O} \,||\, \bar{\sigma_O}') \in$ L(S) it holds that,

- if the set Alt($\bar{\sigma}$, p) is infinite, then there exist infinitely many indices $i \in$ Alt($\bar{\sigma}$, p) in which $p \notin$ Diff($\bar{\sigma_O}[i]$, $\bar{\sigma_O}'[i]$). In simple terms, the output of p is not altered by S in step i.

Synthesis of Shields for Multi-Agent Systems

The synthesis procedure consists of the following three steps:

- Construct a safety game G^s and compute its permissive winning strategy ρ^s, such that any shield S that implements ρ^s ensures correctness ($D \circ S \models \varphi$) and S does not interfere with D unnecessarily.
- We augment the game graph with the assumptions.
- Compute deterministic strategy that implements ρ^s and satisfies the interference constraints.

Constructing and Solving the Safety Game

Let φ be a safety specification represented as a safety automaton $\varphi = (Q, q_0, \Sigma, \delta, F)$. Let $W \subseteq F$ be the winning region of φ when considered as a safety game.

We construct a safety game $\mathcal{G}^s$ such that its most permissive strategy subsumes all possible shields that are correct w.r.t. φ and that do not interfere unnecessarily.

We construct a safety game $\mathcal{G}^s = (G^s, g^s, \Sigma, \Sigma_O, \delta^s, F^s)$ such that
$G^s = \{(g, u, t) \mid g \in Q; u; t \in \{\tau, \bot\}\}$ is the state space where

- the variable u is a flag that indicates wrong outputs by the system
- the variable t tracks deviations between the outputs of the system and the shield

$g^s_0 = (g_0, \bot, \bot)$ is the initial state,
δ^s is the next state function $\delta^s((g, u, t), (\sigma_I, \sigma_O), \sigma_O') = (\delta(g, \sigma_I, \sigma_O'), u', t')$ with

- $u' = \tau$ if $\delta(g, \sigma_I, \sigma_O) \notin W$, and $u' = \bot$ otherwise,
- $t' = \tau$ if $\sigma_O \neq \sigma_O'$ and $t' = \bot$ otherwise;

Fs is the set of safe states, such that $Fs = \{(g, u, t) \in G^s \mid (g \in W) \wedge (u = \bot \rightarrow t = \bot)\}$.

We use standard algorithms for safety games to compute the winning region W^s and the most permissive winning strategy $p_s : G^* \times \Sigma_I \rightarrow 2^{\Sigma o}$.

Synthesis with Assumptions on the Faults

To fulfil the goal of synthesizing cost-optimal shields under the assumption that the length of sequences of wrong outputs is bounded by some constant b, a new game graph G^a is created.

Let a safety game $\mathcal{G}^s = (G^s, g_0{}^s, \Sigma, \Sigma_O, \delta_s, F_s)$ with winning region W^s and permissive winning strategy ρ_s and a bound $b \in \mathbb{N}$ on the maximal length of sequences of wrong outputs.

We construct a new game $\mathcal{G}^a = (G^a, g_0{}^a, \Sigma, \Sigma_O, \delta_a, Acc^a)$ where

- $G^a = W^s \times \{ 0,..., b+1 \}$ is the set of states,
- $g_0{}^a = (g_0{}^s, 0)$ is the initial state,
- δ^a is the next-state function: $\delta^a((g^s, v), \sigma, \sigma_O') = (\delta^s(g^s, \sigma, \sigma_O') \cap \rho_s(g^s, \sigma), v')$ such that
  - if $v \leq b$ and $u' = \tau$, then $v' = v + 1$,
  - if $v \leq b$ and $u' = \bot$, then $v' = 0$, and
  - if $v = b + 1$, then $v' = b + 1$
- Acc^a is such that $Acc^a(\vec{\pi}) = \tau$ iff
  - $\exists i \geq 0.\ g_i{}^a = (g_i{}^s, v_i)$ with $v_i = b + 1$, or
  - there are inf. many $g_i{}^a = (g_i, u_i, t_i, v_i)$ with $t_i = \bot$.

Counter v tracks the length of the current sequence of wrong outputs by the system, and is reset to 0 when the output is correct. If v exceeds the bound b, it remains b+1 forever.

Synthesis of Locally-Optimal Shields

We propose a procedure to synthesize shields that minimize the cost per deviation period assuming that all sequences of wrong outputs are bounded.

We start with the augmented game graph $\mathcal{G}^a = (G^a, g_0{}^a, \Sigma, \Sigma_O, \delta_a, Acc^a)$ and construct a new game $\mathcal{G}^{opt} = (G^a, g_0{}^a, \Sigma, \Sigma_O, \delta_a, Acc^a, Val^{opt})$ with value function $Val^{opt}(\bar{\pi})$ which is an **accumulated cost objective** using c as edge labeling: $cost^{opt}(g^a, (\sigma_I, \sigma_O), \sigma_O') = c(\sigma_O, \sigma_O')$.

We synthesize shields, that are winning according to Acc^a (i.e., either the assumption on the system that any sequence of wrong outputs has a length of at most b is violated, or infinitely often the shield does not interfere) and optimize Val^{opt} (i.e., the worst-case accumulated cost for reaching the end of the deviation per deviation period).

Synthesis of Fair Shields

Augment the states of $\mathcal{G}^s$ with Boolean variables that track information about the minimal correcting sets for each transition.

Let $Mcs(g, \sigma_I, \sigma_O, W)$ be the set of all minimal sets of agents such that correcting the output of these agents results in a successor state of g that is in W.

For $\pi \subseteq Agts$, it holds that $\pi \in Mcs(g, \sigma_I, \sigma_O, W)$ iff

- there exists σ_O' such that $\delta(g, \sigma_I, \sigma'_O) \in W$ and $Diff(\sigma_O, \sigma_O') = \pi$,
- π is minimal

Given $\mathcal{G}^s = (G^s, g_0^s, \Sigma, \Sigma_O, \delta_s, F_s)$ with W^s and ρ^s, we construct a game $G^f = (G^f, g_0^f, \Sigma, \Sigma_O, \delta^f, Acc^f)$ with set of states $G^f = W^s \times \{\bot, \top\}^n \times \{\bot, \top\}^n$, where $n = |Agts|$ is the number of agents. Initial state is $g_0^f = (g_0^s, \bot^n, \bot^n)$, and the transition relation δ^f.

Synthesis of Fair Shields (cont…)

Transition relation δ^f is such that $\delta^a((g, u, t, m_1,...,m_n, c_1,...,c_n), \sigma_O, \sigma_O') = (\delta^s((g, u, t), \sigma, \sigma_O') \cap \rho^s((g, u, t), \sigma), m_1',..., m_n', c_1',..., c_n')$ where for each agent $p \in$ Agts it holds that

- $m_p' = \tau$ iff there exist minimal correcting sets $\pi, \pi' \in Mcs(g, \sigma_I, \sigma_O, W)$ with $p \in \pi$ and $p \notin \pi'$
- $c_p' = \tau$ iff $p \in Diff(\sigma_O, \sigma_O')$.

The acceptance condition Acc^f encodes the fairness requirement on the shield for agent p using the m_p and c_p.

It states for each agent $p \in$ Agts that if a play contains infinitely many occurrences of states in which $m_p = \tau$, then it should contain infinitely many occurrences of states in which $c_p = \bot$ and $m_p = \tau$.

Experiment Evaluation: Gridworld

Reactive synthesis tool Slugs was used to compute locally-optimal shields under the assumption, that sequences of wrong outputs are bounded by a constant b.

Consider a gridworld with two agents that can move in one of the four cardinal directions at each time step. One grid cell is designated as a charging station.

global safety property $\varphi = \varphi_{collision} \wedge \varphi_{charge}$, where
$\varphi_{collision}$ requires collision avoidance and no simultaneous charging and
φ_{charge} describes when and how the agents should use the charging station.
The formula φ_{charge} is of the form $\varphi_{charge,1} \wedge \varphi_{charge,2}$.

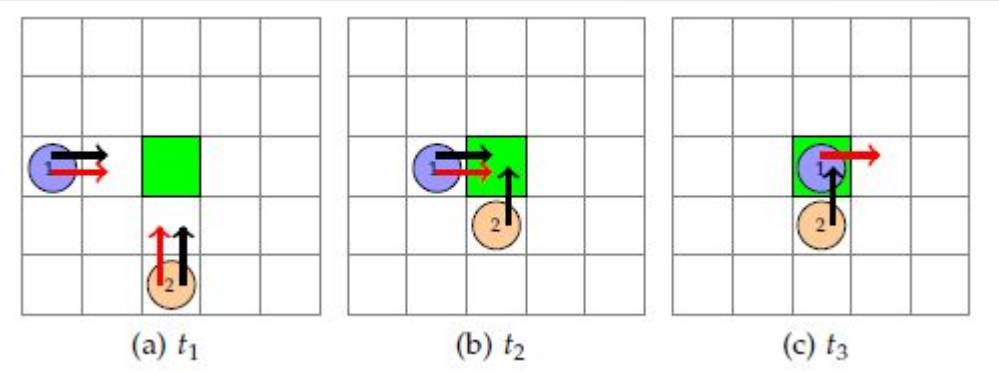
The formula $\varphi_{charge,i}$ requires that one agent cannot enter the charging area right after the other one has left.
$\varphi_{collision}$ requires that the agents do not occupy the same position.

Gridworld (cont..)

Locally-optimal shields is synthesized using an interference cost function c that assigns higher costs for any interferences with the first agent than with the second one.

The first column gives the bound b.
The second and the third column state the number of input and output bits of φ.
The fourth column states the total number of variables of the constructed game (including the variables that augment the state space).
The fifth column gives the number of reachable states in the game.
The last two columns gives the synthesis time (in seconds) to construct the permissive strategy and the locally-optimal strategy.

Case	b	Inp vars	Out vars	Game vars	Game states	time perm	time cost-opt
(1)	2	16	6	28	887	245	67
	5	16	6	30	3456	245	830
(2)	2	24	12	68	41544	3545	3019
	5	24	12	70	7.5×10^5	3545	9822



Experiment Evaluation: UAV Mission Planning

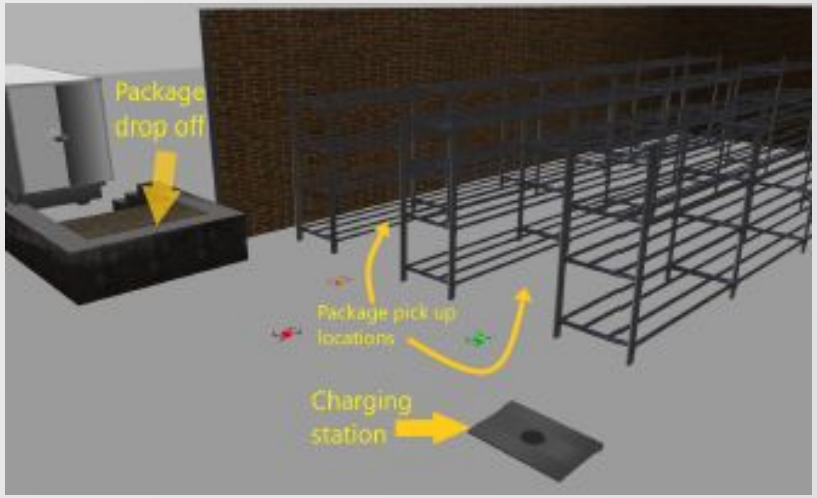
The environment consists of 2 rows of shelves. We assume there are 12 discrete package pick up points in each row of shelves along with the drop off location and charging station.
The input of each UAV controller is its location in (x, y, z).
The possible outputs of each UAV consist of 17 trajectories precomputed using a minimum-snap trajectory generator that moves the UAV from one discrete state to another.

A safety specification φ_{dist} which captures the requirement that the controllers should not choose trajectories that bring them within distance less than a given threshold r.

φ_{shelf} which states that no more than one UAV can move into the row of shelves at the same time.

Only one UAV can be at the charging station or drop-off point at any given time.

UAVs cannot be allowed to run out of power when not in a charging station.



Conclusion and Future Work

Key contribution is the study of quantitative objectives in the shield synthesis setting.

This is also the first work to consider fairness requirements on the shields.

It also introduced the notion of interference cost, and discussed several costs and synthesis objectives that are of interest when considering multi-agent systems.

For finding optimal shield, the shield plans optimally in the short term, without considering the possibility of further wrong outputs once the deviation period ends. Therefore, this optimality criterion is useful when wrong outputs of the system are expected to be rare.

A promising avenue for future work is to investigate bounded synthesis with quantitative objectives, in order to synthesize distributed shields, which will enhance the efficiency of shields for distributed systems.

Questions?