Formal Specification and Parametric Verification of the ICAROUS Distributed Merging Protocol for Autonomous Aircraft Systems

William Schultz[1], Swee Balachandran[2],

[1] Formal Methods Group
Northeastern University
schultz.w@northeastern.edu

[2] National Institute of Aerospace
NASA Langley
sweewarman.balachandran@nasa.gov

Aug 2021 @ NASA Langley
Intern exit presentation

Outline

schultz.w@northeastern.edu

Outline

schultz.w@northeastern.edu

The ICAROUS System

- *ICAROUS (**I**ndependent **C**onfigurable **A**rchitecture for **R**eliable **O**perations of **U**nmanned **S**ystems)* is a software architecture for unmanned aircraft systems (UAS)[1]

[1] Consiglio, María and Muñoz, César and Hagen, George and Narkawicz, Anthony and Balachandran, Swee. ICAROUS: Integrated configurable algorithms for reliable operations of unmanned systems. In: *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*. IEEE. 2016, pp. 1–5.

The ICAROUS System

- *ICAROUS (**I**ndependent **C**onfigurable **A**rchitecture for **R**eliable **O**perations of **U**nmanned **S**ystems)* is a software architecture for unmanned aircraft systems (UAS)[1]
- Includes several software modules for high assurance operation and collision avoidance

[1] Consiglio, María and Muñoz, César and Hagen, George and Narkawicz, Anthony and Balachandran, Swee, "ICAROUS: Integrated configurable algorithms for reliable operations of unmanned systems".

The ICAROUS System

- *ICAROUS (**I**ndependent **C**onfigurable **A**rchitecture for **R**eliable **O**perations of **U**nmanned **S**ystems)* is a software architecture for unmanned aircraft systems (UAS)[1]
- Includes several software modules for high assurance operation and collision avoidance
- Has a distributed algorithm for merging a set of aircraft through an intersection in a decentralized fashion

[1] Consiglio, María and Muñoz, César and Hagen, George and Narkawicz, Anthony and Balachandran, Swee, "ICAROUS: Integrated configurable algorithms for reliable operations of unmanned systems".

Overview of the ICAROUS Merging Protocol

- Set of n aircraft $\{a_1, \ldots, a_n\}$ that want to merge through a designated intersection, specified by a point in the airspace

schultz.w@northeastern.edu

Overview of the ICAROUS Merging Protocol

- Set of n aircraft $\{a_1, \ldots, a_n\}$ that want to merge through a designated intersection, specified by a point in the airspace
- Coordination occurs between the aircraft so that a schedule for when aircraft leave the intersection can be computed

schultz.w@northeastern.edu

Overview of the ICAROUS Merging Protocol

- Set of n aircraft $\{a_1, \ldots, a_n\}$ that want to merge through a designated intersection, specified by a point in the airspace
- Coordination occurs between the aircraft so that a schedule for when aircraft leave the intersection can be computed
- Each aircraft has an earliest and latest arrival time, $R_i \in \mathbb{R}^+$ and $D_i \in \mathbb{R}^+$, respectively

schultz.w@northeastern.edu

Overview of the ICAROUS Merging Protocol

- Set of n aircraft $\{a_1, \ldots, a_n\}$ that want to merge through a designated intersection, specified by a point in the airspace
- Coordination occurs between the aircraft so that a schedule for when aircraft leave the intersection can be computed
- Each aircraft has an earliest and latest arrival time, $R_i \in \mathbb{R}^+$ and $D_i \in \mathbb{R}^+$, respectively
- Must compute schedule of arrival times $T = (T_1, \ldots, T_n)$ such that

$$\forall i \in \{1, \ldots, n\} : R_i \leq T_i \leq D_i - P$$

for some separation time P.

schultz.w@northeastern.edu

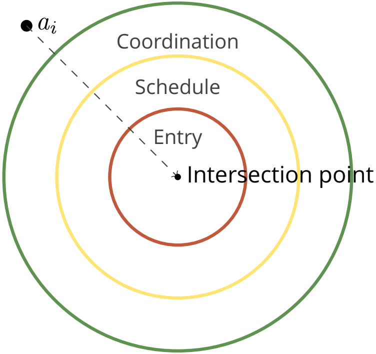
Overview of the ICAROUS Merging Protocol

- Newest version of the protocol uses a simplified consensus mechanism coordinating the merging and schedule computation

schultz.w@northeastern.edu

Overview of the ICAROUS Merging Protocol

- Newest version of the protocol uses a simplified consensus mechanism coordinating the merging and schedule computation
- Designated radial zones expanding outward from the intersection point for aircraft to execute various behaviors needed to achieve the necessary goals



schultz.w@northeastern.edu

Outline

schultz.w@northeastern.edu

Formalizing the ICAROUS Merging Protocol

- The merging protocol is a real-time system with both continuous and discrete dynamics, and its behavior depends on several environmental parameters

schultz.w@northeastern.edu

Formalizing the ICAROUS Merging Protocol

- The merging protocol is a real-time system with both continuous and discrete dynamics, and its behavior depends on several environmental parameters
- **Goal**: formalize an abstract model of the protocol that allows us to understand under what environmental parameters the system satisfies some given property.

schultz.w@northeastern.edu

Formalizing the Merging Protocol

- The protocol can be viewed as a *hybrid automata*[2]

[2] Thomas A Henzinger. The theory of hybrid automata. In: *Verification of digital and hybrid systems*. Springer, 2000, pp. 265–292.

[3] Rajeev Alur and David L Dill. A theory of timed automata. In: *Theoretical computer science* 126.2 (1994), pp. 183–235.

schultz.w@northeastern.edu

Formalizing the Merging Protocol

- The protocol can be viewed as a *hybrid automata*[2]
- With some simplifying assumptions about aircraft speeds, however, we can consider it more similar to a *timed automata*[3], a special case of the former

[2] Henzinger, “The theory of hybrid automata”.
[3] Alur and Dill, “A theory of timed automata”.

schultz.w@northeastern.edu

Formalizing the Merging Protocol

- The protocol can be viewed as a *hybrid automata*[2]
- With some simplifying assumptions about aircraft speeds, however, we can consider it more similar to a *timed automata*[3], a special case of the former
- Avoids the need to model dynamics using differential equations

[2] Henzinger, "The theory of hybrid automata".

[3] Alur and Dill, "A theory of timed automata".

schultz.w@northeastern.edu

Modeling the Merging Protocol in TLA+

- TLA+ (Temporal Logic of Actions) is a high level specification language built primarily for specifying concurrent/distributed protocols, created by Leslie Lamport

[4] Leslie Lamport. Real time is really simple. In: *Microsoft Research* (2005), pp. 2005–30.

schultz.w@northeastern.edu

Modeling the Merging Protocol in TLA+

- TLA+ (Temporal Logic of Actions) is a high level specification language built primarily for specifying concurrent/distributed protocols, created by Leslie Lamport
- Not designed for real time verification, but can be extended in a straightforward manner to model real time clocks[4]

[4] Lamport, "Real time is really simple".

schultz.w@northeastern.edu

Modeling the Merging Protocol in TLA+

- TLA+ (Temporal Logic of Actions) is a high level specification language built primarily for specifying concurrent/distributed protocols, created by Leslie Lamport
- Not designed for real time verification, but can be extended in a straightforward manner to model real time clocks[4]
- Has an associated explicit state model checker, TLC, for finite state verification of temporal properties

[4] Lamport, "Real time is really simple".

schultz.w@northeastern.edu

Modeling the Merging Protocol in TLA+

- TLA+ (Temporal Logic of Actions) is a high level specification language built primarily for specifying concurrent/distributed protocols, created by Leslie Lamport
- Not designed for real time verification, but can be extended in a straightforward manner to model real time clocks[4]
- Has an associated explicit state model checker, TLC, for finite state verification of temporal properties
- Choice of TLA+ primarily influenced by its high degree of expressivity, our familiarity with it, and its automated verification tools.

[4] Lamport, "Real time is really simple".

schultz.w@northeastern.edu

Modeling the Protocol in TLA+

- Defining a system in TLA+ requires the definition of an *initial state predicate* and *next state relation*

schultz.w@northeastern.edu

Modeling the Protocol in TLA+

- Defining a system in TLA+ requires the definition of an *initial state predicate* and *next state relation*
- For example:

schultz.w@northeastern.edu

Modeling the Protocol in TLA+

- Defining a system in TLA+ requires the definition of an *initial state predicate* and *next state relation*
- For example:

VARIABLE x

$$Init \triangleq x \in \{0, 1, 2\}$$

$$Next \triangleq \exists inc \in \{1, 2\} : x' = x + inc$$

$$Spec \triangleq Init \land \Box[Next]_x$$

schultz.w@northeastern.edu

Components of the TLA+ Merging Specification

schultz.w@northeastern.edu

Components of the TLA+ Merging Specification

1. *Aircraft dynamics*: positions and velocities of aircraft, when they enter and exit zones

schultz.w@northeastern.edu

Components of the TLA+ Merging Specification

1. *Aircraft dynamics*: positions and velocities of aircraft, when they enter and exit zones
2. *Consensus mechanism*: logic for election of a leader aircraft and arrival time info propagation

schultz.w@northeastern.edu

Components of the TLA+ Merging Specification

1. *Aircraft dynamics*: positions and velocities of aircraft, when they enter and exit zones
2. *Consensus mechanism*: logic for election of a leader aircraft and arrival time info propagation
3. *Schedule computation*: Local computation of arrival schedules based on known information

schultz.w@northeastern.edu

Components of the TLA+ Merging Specification

1. *Aircraft dynamics*: positions and velocities of aircraft, when they enter and exit zones
2. *Consensus mechanism*: logic for election of a leader aircraft and arrival time info propagation
3. *Schedule computation*: Local computation of arrival schedules based on known information
4. *Real time clock*: tracking current time and outstanding timers/deadlines

schultz.w@northeastern.edu

State Variables

Aircraft Dynamics

$speed \in Node \to \mathbb{N}:$ aircraft's initial speed

$coordEntryTime \in Node \to \mathbb{N}:$ coordination zone entry time

$coordLeaveAt \in Node \to \mathbb{N}:$ coordination zone exit time

$schedLeaveAt \in Node \to \mathbb{N}:$ schedule zone exit time

$entryLeaveAt \in Node \to \mathbb{N}:$ entry zone exit time

$zoneStatus \in Node \to Zone:$ current zone

schultz.w@northeastern.edu

State Variables

Aircraft Dynamics

$speed \in Node \rightarrow \mathbb{N}:$ aircraft's initial speed

$coordEntryTime \in Node \rightarrow \mathbb{N}:$ coordination zone entry time

$coordLeaveAt \in Node \rightarrow \mathbb{N}:$ coordination zone exit time

$schedLeaveAt \in Node \rightarrow \mathbb{N}:$ schedule zone exit time

$entryLeaveAt \in Node \rightarrow \mathbb{N}:$ entry zone exit time

$zoneStatus \in Node \rightarrow Zone:$ current zone

Consensus Mechanism

$leader \in Node \rightarrow \{True, False\}:$ leader status

$term \in Node \rightarrow \mathbb{N}:$ term number

$arrivalTimes \in Node \rightarrow (Node \rightarrow \mathbb{N}):$ arrival time info known by each aircraft

$zoneStatusInfo \in Node \rightarrow (Node \rightarrow Zone):$ zone status info known by each aircraft

$hbTimeout \in Node \rightarrow (\mathbb{N} \cup \{None\}):$ when next heartbeat from leader should occur

$leaderTimeout \in Node \rightarrow (\mathbb{N} \cup \{None\}):$ when next election should occur

schultz.w@northeastern.edu

State Variables

Aircraft Dynamics

$speed \in Node \to \mathbb{N}:$ aircraft's initial speed

$coordEntryTime \in Node \to \mathbb{N}:$ coordination zone entry time

$coordLeaveAt \in Node \to \mathbb{N}:$ coordination zone exit time

$schedLeaveAt \in Node \to \mathbb{N}:$ schedule zone exit time

$entryLeaveAt \in Node \to \mathbb{N}:$ entry zone exit time

$zoneStatus \in Node \to Zone:$ current zone

Consensus Mechanism

$leader \in Node \to \{True, False\}:$ leader status

$term \in Node \to \mathbb{N}:$ term number

$arrivalTimes \in Node \to (Node \to \mathbb{N}):$ arrival time info known by each aircraft

$zoneStatusInfo \in Node \to (Node \to Zone):$ zone status info known by each aircraft

$hbTimeout \in Node \to (\mathbb{N} \cup \{None\}):$ when next heartbeat from leader should occur

$leaderTimeout \in Node \to (\mathbb{N} \cup \{None\}):$ when next election should occur

Real Time Clock

$now \in \mathbb{N}:$ current time

schultz.w@northeastern.edu

Environmental Parameters of Interest

$HBInterval \in \mathbb{N}:$ time between heartbeat messages sent by a primary

$LeaderTimeout \in \mathbb{N}:$ time an aircraft waits before running an election

$coordDist \in \mathbb{N}:$ coordination zone length

$schedDist \in \mathbb{N}:$ schedule zone length

$entryDist \in \mathbb{N}:$ entry zone length

schultz.w@northeastern.edu

Initial States

$$Init \triangleq$$

Aircraft Dynamics

$$\begin{cases} \land zoneStatus = [n \in Node \mapsto \text{“None”}] \\ \land coordLeaveAt = [n \in Node \mapsto 0] \\ \land schedLeaveAt = [n \in Node \mapsto 0] \\ \land entryLeaveAt = [n \in Node \mapsto 0] \\ \land speed \in [Node \rightarrow MinInitSpeed..MaxInitSpeed] \\ \land schedTime = [n \in Node \mapsto 0] \\ \land schedUpdate = [n \in Node \mapsto FALSE] \\ \land coordEntryTime \in [Node \rightarrow 0, CoordEntrySepTime] \end{cases}$$

Consensus Mechanism

$$\begin{cases} \land leader = [n \in Node \mapsto FALSE] \\ \land arrivalTimes = [n \in Node \mapsto [i \in Node \mapsto None]] \\ \land zoneStatusInfo = [n \in Node \mapsto [i \in Node \mapsto \text{“None”}]] \\ \land hbTimeout = [n \in Node \mapsto None] \\ \land leaderTimeout = [n \in Node \mapsto None] \\ \land term = [n \in Node \mapsto 0] \end{cases}$$

Real Time Clock

$$\{\ \land now = 0$$

schultz.w@northeastern.edu

Transition Relation

$$Next \triangleq$$

$$\text{Aircraft Dynamics} \begin{cases} \lor \exists i \in Node : EnterCoordZone(i) \\ \lor \exists i \in Node : EnterSchedZone(i) \\ \lor \exists i \in Node : EnterEntryZone(i) \\ \lor \exists i \in Node : Exit(i) \end{cases}$$

$$\text{Consensus Mechanism} \begin{cases} \lor \exists i \in Node : BecomeLeader(i) \\ \lor \exists i \in Node : IncTerm(i) \\ \lor \exists i \in Node, sub \in \text{SUBSET } Node : BroadcastHB(i, sub) \\ \lor \exists i \in Node : ComputeSchedule(i) \end{cases}$$

$$\text{Real Time Clock} \begin{cases} \lor Tick \end{cases}$$

schultz.w@northeastern.edu

Modeling Real Time

- Current time is modeled with an explicit variable, now

schultz.w@northeastern.edu

Modeling Real Time

- Current time is modeled with an explicit variable, now
- The $Tick$ action advances the clock by some discrete increment, subject to preconditions

schultz.w@northeastern.edu

Modeling Real Time

- Current time is modeled with an explicit variable, now
- The $Tick$ action advances the clock by some discrete increment, subject to preconditions
- The general form of the $Tick$ action is as follows:

$$
\begin{aligned}
Tick \triangleq \\
& \exists d \in DiscreteTime : \\
& \land\ TimerConds \\
& \land\ now' = now + d
\end{aligned}
$$

schultz.w@northeastern.edu

Modeling Real Time

- Current time is modeled with an explicit variable, now
- The $Tick$ action advances the clock by some discrete increment, subject to preconditions
- The general form of the $Tick$ action is as follows:

$$
\begin{aligned}
Tick \triangleq \\
& \exists d \in DiscreteTime : \\
& \land\ TimerConds \\
& \land\ now' = now + d
\end{aligned}
$$

- $DiscreteTime$ is the set of possible clock increment values the clock can take, and $TimerConds$ are preconditions that prevent the clock from ticking past a deadline e.g.

$$\forall i \in Node : (hbTimeout[i] \neq None) \Rightarrow (now + d \leq hbTimeout[i])$$

schultz.w@northeastern.edu

Outline

schultz.w@northeastern.edu

Parametric Verification with the TLC Model Checker

- **Goal**: semi-automated way to discover parameter values for which protocol satisfies some property

schultz.w@northeastern.edu

Parametric Verification with the TLC Model Checker

- **Goal**: semi-automated way to discover parameter values for which protocol satisfies some property
  - Idea is to use the model checker to verify discretized parameter regions

schultz.w@northeastern.edu

Parametric Verification with the TLC Model Checker

- **Goal**: semi-automated way to discover parameter values for which protocol satisfies some property
  - Idea is to use the model checker to verify discretized parameter regions
  - Visualize the safe and unsafe regions of the parameter space

schultz.w@northeastern.edu

Methodology

- Consider the system as a parameterized specification $S(k_1, \ldots, k_n)$ for parameters $k_i \in \mathbb{N}$

schultz.w@northeastern.edu

Methodology

- Consider the system as a parameterized specification $S(k_1, \ldots, k_n)$ for parameters $k_i \in \mathbb{N}$
- For a given property P, check

$$S(k_1, \ldots, k_n) \vDash P$$

over a range of numeric parameter values $(k_1, \ldots, k_n) \in K_1 \times \cdots \times K_n$ for finite domains $K_i \subseteq \mathbb{N}$.

schultz.w@northeastern.edu

Methodology

- Consider the system as a parameterized specification $S(k_1, \ldots, k_n)$ for parameters $k_i \in \mathbb{N}$
- For a given property P, check

$$S(k_1, \ldots, k_n) \models P$$

  over a range of numeric parameter values $(k_1, \ldots, k_n) \in K_1 \times \cdots \times K_n$ for finite domains $K_i \subseteq \mathbb{N}$.
- Initially focused on examining 2D parameter spaces, with bounded time

schultz.w@northeastern.edu

Methodology

- Consider the system as a parameterized specification $S(k_1, \ldots, k_n)$ for parameters $k_i \in \mathbb{N}$
- For a given property P, check

  $$S(k_1, \ldots, k_n) \models P$$

  over a range of numeric parameter values $(k_1, \ldots, k_n) \in K_1 \times \cdots \times K_n$ for finite domains $K_i \subseteq \mathbb{N}$.
- Initially focused on examining 2D parameter spaces, with bounded time
  - For all parameters $k_1, \ldots, k_n$, vary two distinct parameters i and j and fix the rest

schultz.w@northeastern.edu

Methodology

- Consider the system as a parameterized specification $S(k_1, \ldots, k_n)$ for parameters $k_i \in \mathbb{N}$
- For a given property P, check

$$S(k_1, \ldots, k_n) \vDash P$$

  over a range of numeric parameter values $(k_1, \ldots, k_n) \in K_1 \times \cdots \times K_n$ for finite domains $K_i \subseteq \mathbb{N}$.
- Initially focused on examining 2D parameter spaces, with bounded time
  - For all parameters $k_1, \ldots, k_n$, vary two distinct parameters i and j and fix the rest
  - Place an upper bound on the clock value now

schultz.w@northeastern.edu

Preliminary Verification Results

- Focused on checking $LeaderTimeout$ vs. $HBInterval$ parameter region:

schultz.w@northeastern.edu

Preliminary Verification Results

- Focused on checking $LeaderTimeout$ vs. $HBInterval$ parameter region:
  - $LeaderTimeout$: $[200, 1200]$, $step = 20$
  - $HBInterval$: $[200, 900]$, $step = 20$

schultz.w@northeastern.edu

Preliminary Verification Results

- Focused on checking $LeaderTimeout$ vs. $HBInterval$ parameter region:
  - $LeaderTimeout$: $[200, 1200]$, $step = 20$
  - $HBInterval$: $[200, 900]$, $step = 20$
  - $coordDist$: 1000
  - $schedDist$: 1000
  - $entryDist$: 1000
  - $CoordEntrySepTime$: 0
  - $MaxNow$: 4000
  - $Node$: $\{a_1, a_2, a_3\}$

schultz.w@northeastern.edu

Preliminary Verification Results

- Focused on checking $\mathit{LeaderTimeout}$ vs. $\mathit{HBInterval}$ parameter region:
  - $\mathit{LeaderTimeout}$: $[200, 1200]$, $\mathit{step} = 20$
  - $\mathit{HBInterval}$: $[200, 900]$, $\mathit{step} = 20$
  - $\mathit{coordDist}$: 1000
  - $\mathit{schedDist}$: 1000
  - $\mathit{entryDist}$: 1000
  - $\mathit{CoordEntrySepTime}$: 0
  - MaxNow: 4000
  - Node: $\{a_1, a_2, a_3\}$
- Invariant checked:

$$
\begin{aligned}
\mathit{NoCollisions} \triangleq\ & \forall i, j \in \mathit{Node} : \\
& \neg(\land \mathit{zoneStatus}[i] = \text{“}\mathit{Entry}\text{”} \\
& \quad \land \mathit{zoneStatus}[j] = \text{“}\mathit{Entry}\text{”} \\
& \quad \land \mathit{entryLeaveAt}[i] = \mathit{entryLeaveAt}[j] \\
& \quad \land i \neq j)
\end{aligned}
$$

schultz.w@northeastern.edu

Preliminary Verification Results

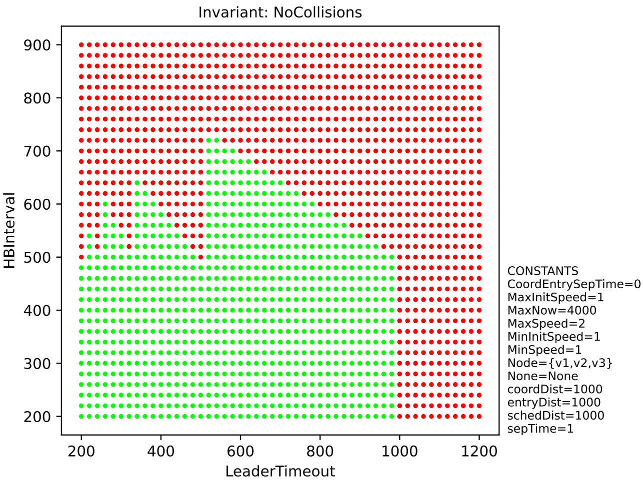
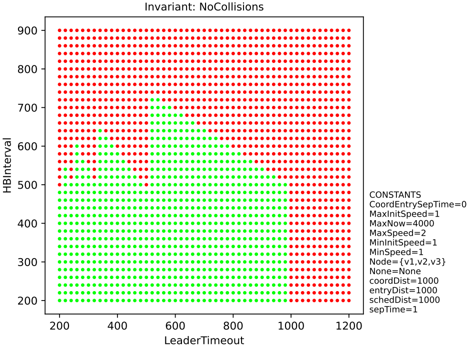


Figure: Verification results for *LeaderTimeout* vs. *HBInterval*

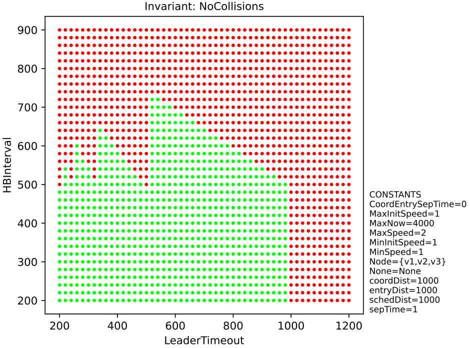
schultz.w@northeastern.edu

Preliminary Verification Results



- How to understand this plot?

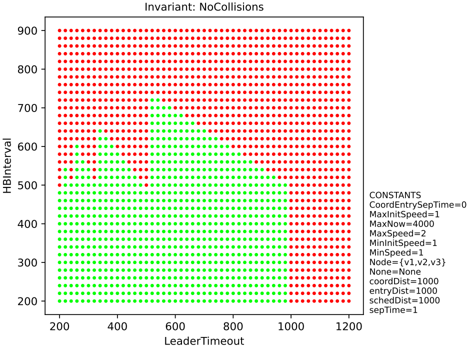
Preliminary Verification Results



- Aircraft are only elected in the coordination zone, so they have a limited window for election i.e. $coordDist/initSpeed = 1000$

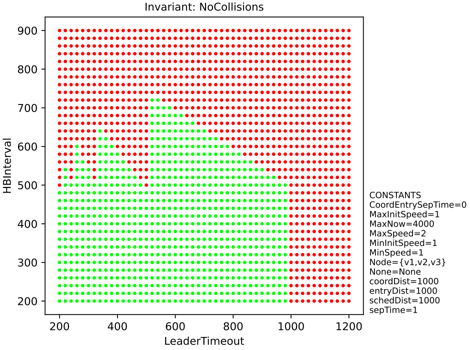
schultz.w@northeastern.edu

Preliminary Verification Results



- Moreover, if more than one election occurs in the coordination zone, this leader takeover pushes back when the first round of heartbeats are sent

Preliminary Verification Results



- Moreover, if more than one election occurs in the coordination zone, this leader takeover pushes back when the first round of heartbeats are sent
- If a leader cannot complete two rounds of heartbeats before aircraft enter the entry zone, may lead to inconsistent schedules

schultz.w@northeastern.edu

Fitting a Model

Parametric inequality for avoiding collisions:

$$2 \cdot H + N_L \cdot L \leq \frac{coordDist + schedDist}{initSpeed}$$

schultz.w@northeastern.edu

Fitting a Model

Parametric inequality for avoiding collisions:

$$2 \cdot H + N_L \cdot L \leq \frac{coordDist + schedDist}{initSpeed}$$

where

$$\begin{aligned} L &= LeaderTimeout \\ H &= HBInterval \\ N_L &= \left\lfloor \frac{T_{coord}}{L} \right\rfloor \end{aligned}$$

and

$$T_{coord} = \frac{coordDist}{initSpeed}$$

schultz.w@northeastern.edu

Fitting a Model

Parametric inequality for avoiding collisions: (*after plugging in*)

$$2 \cdot H + N_L \cdot L \leq \frac{coordDist + schedDist}{initSpeed}$$

where

$$\begin{aligned} L &= LeaderTimeout \\ H &= HBInterval \\ N_L &= \left\lfloor \frac{coordDist}{initSpeed \cdot L} \right\rfloor \end{aligned}$$

schultz.w@northeastern.edu

Fitting a Model

Parametric inequality for avoiding collisions: (*after plugging in again*)

$$2 \cdot H + \left\lfloor \frac{coordDist}{initSpeed \cdot L} \right\rfloor \cdot L \leq \frac{coordDist + schedDist}{initSpeed}$$

where

$$\begin{aligned} L &= LeaderTimeout \\ H &= HBInterval \end{aligned}$$

schultz.w@northeastern.edu

Fitting a Model

Parametric inequality for avoiding collisions: (*after plugging in again*)

$$2 \cdot H + \left\lfloor \frac{coordDist}{initSpeed \cdot L} \right\rfloor \cdot L \leq \frac{coordDist + schedDist}{initSpeed}$$

We can plot this function for some simple parameters.

schultz.w@northeastern.edu

Fitting a Model

$$2 \cdot H + \left\lfloor \frac{coordDist}{initSpeed \cdot L} \right\rfloor \cdot L \leq \frac{coordDist + schedDist}{initSpeed}$$

schultz.w@northeastern.edu

Fitting a Model

$$2 \cdot H + \left\lfloor \frac{coordDist}{initSpeed \cdot L} \right\rfloor \cdot L \leq \frac{coordDist + schedDist}{initSpeed}$$

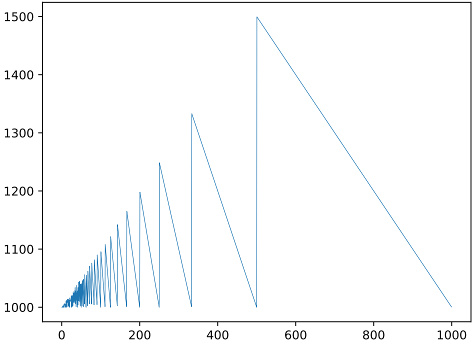


Figure: Sawtooth boundary function $f(x) = 2000 - \left\lfloor \frac{1000}{x} \right\rfloor \cdot x$

schultz.w@northeastern.edu

Fitting a Model

- Overlaying a portion of this function onto the original plot, scaled appropriately:

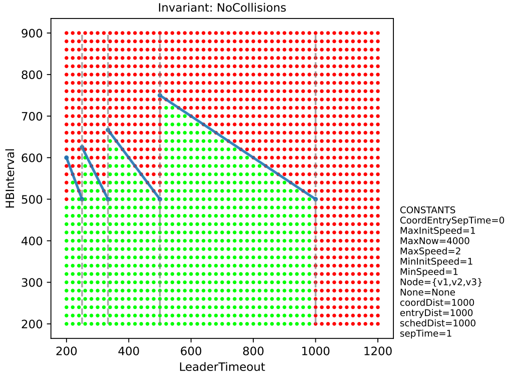


Figure: Annotated verification results for $LeaderTimeout$ vs. $HBInterval$

schultz.w@northeastern.edu

Fitting a Model

- Early demonstration that this approach can provide useful insights about system behavior

schultz.w@northeastern.edu

Fitting a Model

- Early demonstration that this approach can provide useful insights about system behavior
  - e.g. derive symbolic constraints from the discretized verification data

schultz.w@northeastern.edu

Fitting a Model

- Early demonstration that this approach can provide useful insights about system behavior
  - e.g. derive symbolic constraints from the discretized verification data
- Further verification results for more parameter ranges were generated, but not yet analyzed in depth

schultz.w@northeastern.edu

Outline

schultz.w@northeastern.edu

Limitations

- Current method is approximate

schultz.w@northeastern.edu

Limitations

- Current method is approximate
  - Only provides hints at what the safe regions are

schultz.w@northeastern.edu

Limitations

- Current method is approximate
  - Only provides hints at what the safe regions are
  - Could serve as an initial step before more complete verification attempts e.g. using automated theorem prover

schultz.w@northeastern.edu

Limitations

- Current method is approximate
  - Only provides hints at what the safe regions are
  - Could serve as an initial step before more complete verification attempts e.g. using automated theorem prover
- Model checking can be expensive

schultz.w@northeastern.edu

Limitations

- Current method is approximate
  - Only provides hints at what the safe regions are
  - Could serve as an initial step before more complete verification attempts e.g. using automated theorem prover
- Model checking can be expensive
  - Several minutes, up to hours, to generate large, fine-grained parameter ranges

schultz.w@northeastern.edu

Limitations

- Current method is approximate
  - Only provides hints at what the safe regions are
  - Could serve as an initial step before more complete verification attempts e.g. using automated theorem prover
- Model checking can be expensive
  - Several minutes, up to hours, to generate large, fine-grained parameter ranges
  - To generate the results shown in Figure 21, checked 1836 parameter configurations in 5 min. 42 seconds with 8 TLC worker threads on 6-core 2.6GHz Intel Core i7 Macbook Pro.

schultz.w@northeastern.edu

Future Directions

- Explore symbolic techniques implemented by tools like IMITATOR 3[5] (similar to HyTech[6])

---

5 Étienne André. IMITATOR 3: Synthesis of Timing Parameters Beyond Decidability. In: *International Conference on Computer Aided Verification*. Springer. 2021, pp. 552–565.

6 Thomas A Henzinger, Pei-Hsin Ho, and Howard Wong-Toi. HyTech: A model checker for hybrid systems. In: *International Journal on Software Tools for Technology Transfer* 1.1-2 (1997), pp. 110–122.

schultz.w@northeastern.edu

Future Directions

- Explore symbolic techniques implemented by tools like IMITATOR 3[5] (similar to HyTech[6])
    - Unclear if they are able to infer the class of parameter constraints that arise in the merging protocol

[5] André, "IMITATOR 3: Synthesis of Timing Parameters Beyond Decidability".

[6] Henzinger, Ho, and Wong-Toi, "HyTech: A model checker for hybrid systems".

schultz.w@northeastern.edu

Future Directions

- Explore symbolic techniques implemented by tools like IMITATOR 3[5] (similar to HyTech[6])
    - Unclear if they are able to infer the class of parameter constraints that arise in the merging protocol
- Automatic inference of parameter constraints from verification data

[5] André, "IMITATOR 3: Synthesis of Timing Parameters Beyond Decidability".

[6] Henzinger, Ho, and Wong-Toi, "HyTech: A model checker for hybrid systems".

schultz.w@northeastern.edu

Future Directions

- Explore symbolic techniques implemented by tools like IMITATOR 3[5] (similar to HyTech[6])
  - Unclear if they are able to infer the class of parameter constraints that arise in the merging protocol
- Automatic inference of parameter constraints from verification data
- Model checking optimizations:
  - Binary edge search
  - Boundary refinement
  - Improved TLC support for these specific types of parameterized verification tasks

5 André, "IMITATOR 3: Synthesis of Timing Parameters Beyond Decidability".

6 Henzinger, Ho, and Wong-Toi, "HyTech: A model checker for hybrid systems".

schultz.w@northeastern.edu

Outline

schultz.w@northeastern.edu

Related Work

- Uppaal[7] and Kronos[8], tools for standard timed automata verification

[7] Gerd Behrmann, Alexandre David, and Kim G Larsen. A tutorial on uppaal. In: *Formal methods for the design of real-time systems* (2004), pp. 200–236.

[8] Marius Bozga, Conrado Daws, Oded Maler, Alfredo Olivero, Stavros Tripakis, and Sergio Yovine. Kronos: A model-checking tool for real-time systems. In: *International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems*. Springer. 1998, pp. 298–302.

[9] Alur and Dill, "A theory of timed automata".

[10] Henzinger, Ho, and Wong-Toi, "HyTech: A model checker for hybrid systems".

[11] Thomas Hune, Judi Romijn, Mariëlle Stoelinga, and Frits Vaandrager. Linear parametric model checking of timed automata. In: *The Journal of Logic and Algebraic Programming* 52 (2002), pp. 183–220.

[12] André, "IMITATOR 3: Synthesis of Timing Parameters Beyond Decidability".

[13] Mikael Asplund. Automatically proving the correctness of vehicle coordination. In: *ICT Express* 4.1 (2018), pp. 51–54.

schultz.w@northeastern.edu

Related Work

- Uppaal[7] and Kronos[8], tools for standard timed automata verification
- Verification techniques for *parametric timed automata*[9]
  - HyTech model checker[10], developed in 1997, but no longer maintained
  - Extensions of Uppaal to do parameter synthesis[11]
  - IMITATOR[12] is a more recent tool developed over the last decade or so

7 Behrmann, David, and Larsen, "A tutorial on uppaal".

8 Bozga, Daws, Maler, Olivero, Tripakis, and Yovine, "Kronos: A model-checking tool for real-time systems".

9 Alur and Dill, "A theory of timed automata".

10 Henzinger, Ho, and Wong-Toi, "HyTech: A model checker for hybrid systems".

11 Hune, Romijn, Stoelinga, and Vaandrager, "Linear parametric model checking of timed automata".

12 André, "IMITATOR 3: Synthesis of Timing Parameters Beyond Decidability".

13 Asplund, "Automatically proving the correctness of vehicle coordination".

schultz.w@northeastern.edu

Related Work

- Uppaal[7] and Kronos[8], tools for standard timed automata verification
- Verification techniques for *parametric timed automata*[9]
  - HyTech model checker[10], developed in 1997, but no longer maintained
  - Extensions of Uppaal to do parameter synthesis[11]
  - IMITATOR[12] is a more recent tool developed over the last decade or so
- Using SMT solvers to verify autonomous vehicle coordination protocols[13]

---

7 Behrmann, David, and Larsen, "A tutorial on uppaal".

8 Bozga, Daws, Maler, Olivero, Tripakis, and Yovine, "Kronos: A model-checking tool for real-time systems".

9 Alur and Dill, "A theory of timed automata".

10 Henzinger, Ho, and Wong-Toi, "HyTech: A model checker for hybrid systems".

11 Hune, Romijn, Stoelinga, and Vaandrager, "Linear parametric model checking of timed automata".

12 André, "IMITATOR 3: Synthesis of Timing Parameters Beyond Decidability".

13 Asplund, "Automatically proving the correctness of vehicle coordination".

schultz.w@northeastern.edu

Questions?

References I

- Alur, Rajeev and David L Dill. A theory of timed automata. In: *Theoretical computer science* 126.2 (1994), pp. 183–235.
- André, Étienne. IMITATOR 3: Synthesis of Timing Parameters Beyond Decidability. In: *International Conference on Computer Aided Verification*. Springer. 2021, pp. 552–565.
- Asplund, Mikael. Automatically proving the correctness of vehicle coordination. In: *ICT Express* 4.1 (2018), pp. 51–54.
- Behrmann, Gerd, Alexandre David, and Kim G Larsen. A tutorial on uppaal. In: *Formal methods for the design of real-time systems* (2004), pp. 200–236.
- Bozga, Marius, Conrado Daws, Oded Maler, Alfredo Olivero, Stavros Tripakis, and Sergio Yovine. Kronos: A model-checking tool for real-time systems. In: *International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems*. Springer. 1998, pp. 298–302.

References II

- Consiglio, María and Muñoz, César and Hagen, George and Narkawicz, Anthony and Balachandran, Swee. ICAROUS: Integrated configurable algorithms for reliable operations of unmanned systems. In: *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*. IEEE. 2016, pp. 1–5.
- Henzinger, Thomas A. The theory of hybrid automata. In: *Verification of digital and hybrid systems*. Springer, 2000, pp. 265–292.
- Henzinger, Thomas A, Pei-Hsin Ho, and Howard Wong-Toi. HyTech: A model checker for hybrid systems. In: *International Journal on Software Tools for Technology Transfer* 1.1-2 (1997), pp. 110–122.
- Hune, Thomas, Judi Romijn, Mariëlle Stoelinga, and Frits Vaandrager. Linear parametric model checking of timed automata. In: *The Journal of Logic and Algebraic Programming* 52 (2002), pp. 183–220.
- Lamport, Leslie. Real time is really simple. In: *Microsoft Research* (2005), pp. 2005–30.