RESEARCH REVIEW 2023

Carnegie Mellon University
Software Engineering Institute

Formal Arguments for Large-Scale Assurance (FALSA)

Dr. Gabriel Moreno
Principal Researcher

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM23-1074

Overview

DoD needs to deploy new capability with speed and confidence to adapt to changing missions and environments.

However, assuring systems to deploy them with confidence is a bottleneck to achieving the desired speed.

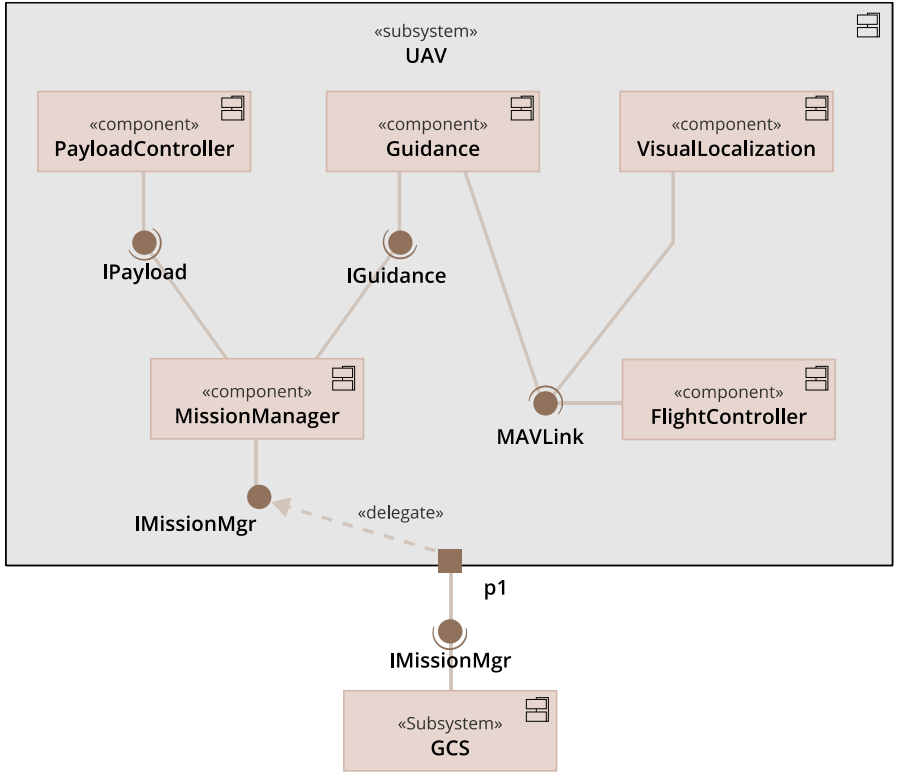
This work aims to speed up the assurance of evolving large-scale systems through

- reuse and sound integration of diverse assurance analyses
- rapid detection of non-conformance between the system behavior and its assurance argument

Model Problem

- Preserves the technical difficulties that this project addresses
- Based on a humanitarian mission that must be carried out autonomously by a UAV to deliver life-saving supplies in a disaster zone
- Implementation in simulation

Model Problem Architecture



Main characteristics:

- Multiple quality attribute concerns: safety, timing, control, security
- Mix of existing COTS and new components to be integrated
- Mixed assurance results

All these characteristics are what make assurance with speed difficult.

Barriers to Assurance with Speed and Confidence

Multiple factors hinder assurance speed:

- lack of effective reuse of assurance results
- inability to integrate multiple type of assurance analyses with different levels of formality
  - consequently, the inability to automatically and formally evaluate such assurance arguments
- lack of flexibility to adapt to changing mission/environment

Solution

Combine analyses and results that have different levels of formality in a formal assurance argument that

- ensures that the combination has a single interpretation that can be automated
- can be developed incrementally
- is modular to allow reuse of assurance results

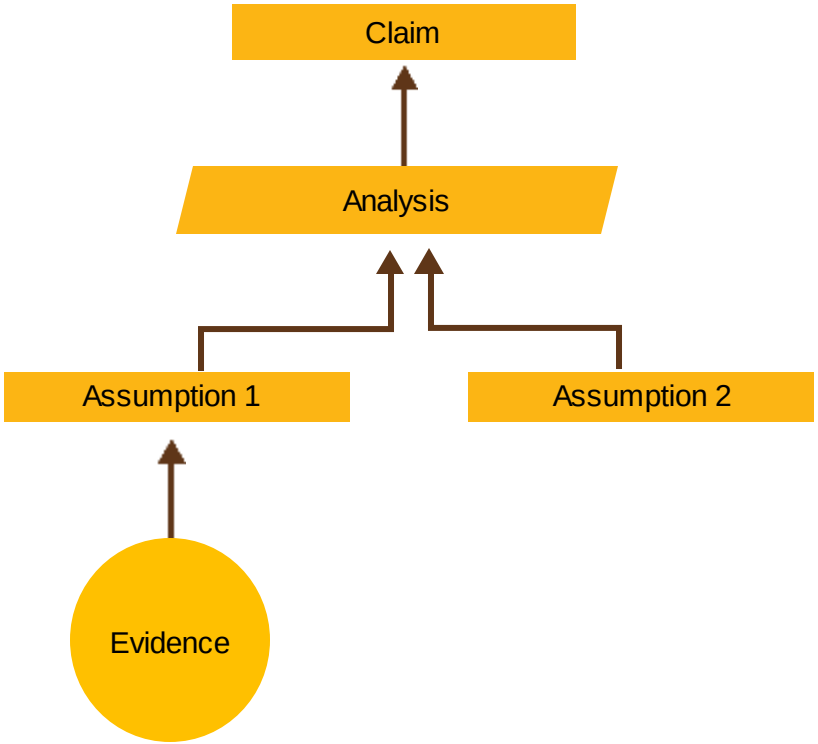
Use runtime monitoring and data to

- detect whether system behavior conforms to the assurance argument
- deal with incorrect or changing environment or system assumptions
- provide the foundation for runtime enforcement and adaptation in future work

Carnegie Mellon University
Software Engineering Institute

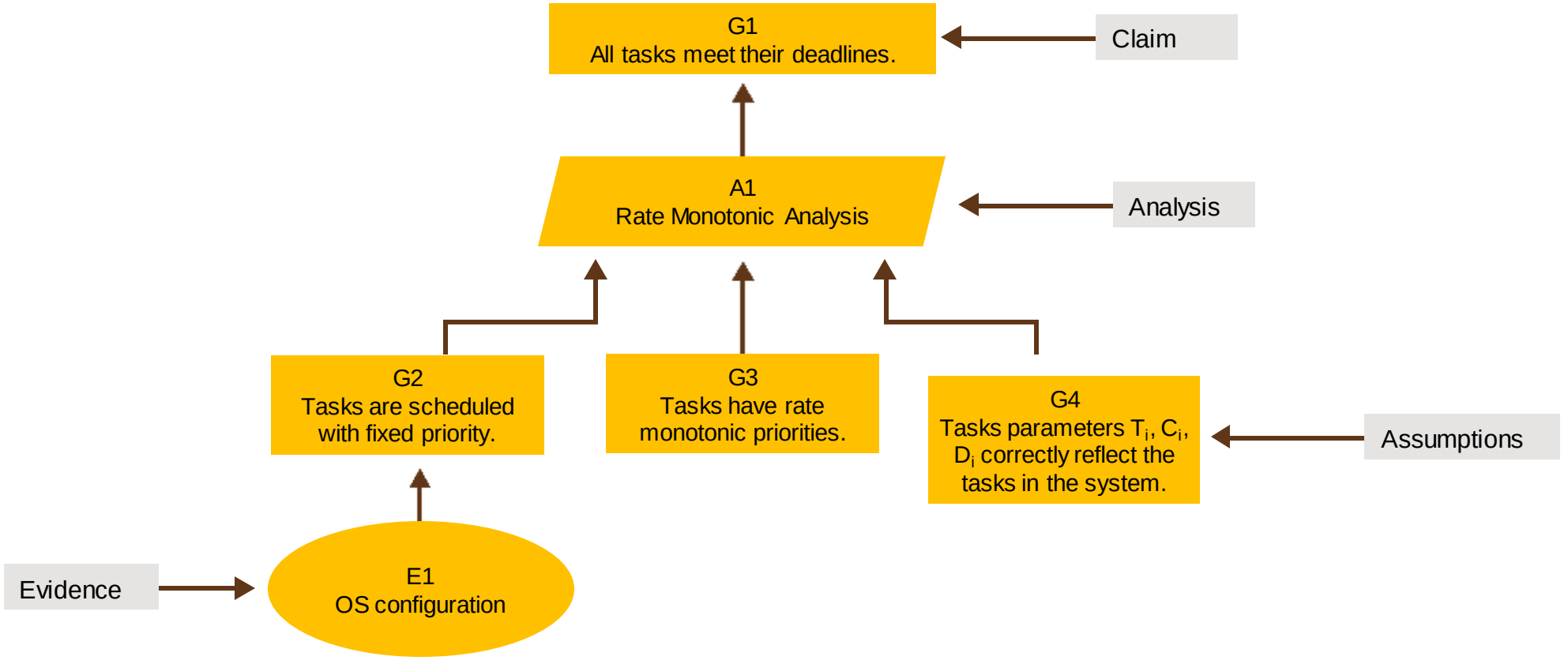
Assurance Reasoning with Mixed Formality Levels

Argument Elements

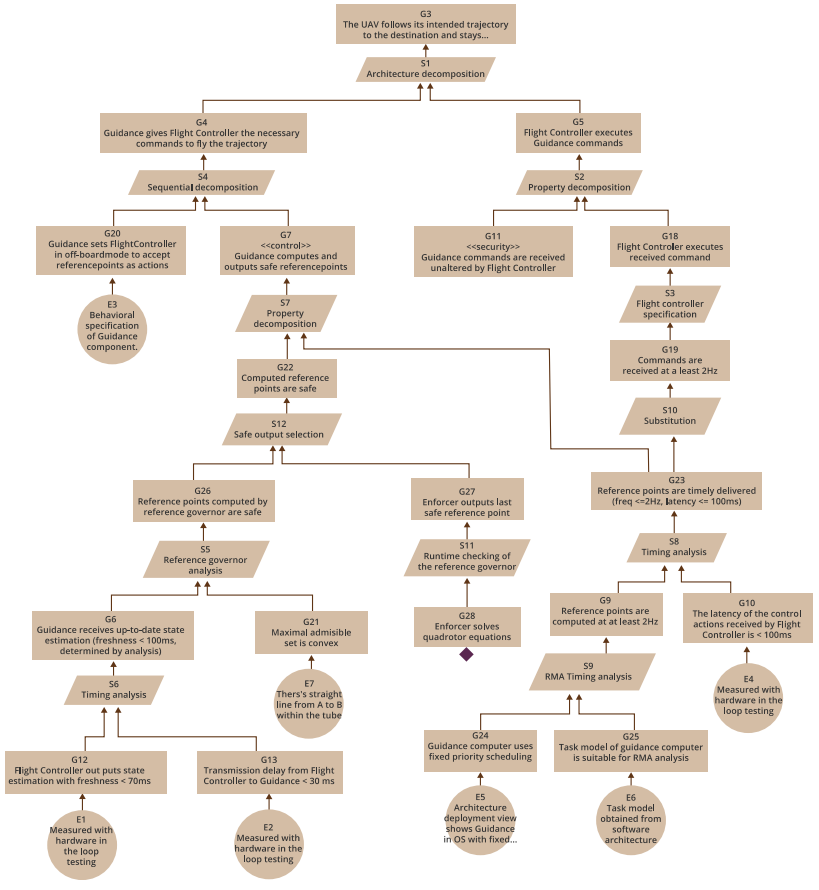


- *Claim:* property or assertion about the system
- *Analysis:* a procedure that, given that its assumptions hold, proves or supports a claim
- *Assumption:* property that must hold for the analysis results to be valid
- *Evidence:* data or facts about the system and/or its environment relevant to an assumption

Argument Elements: Example



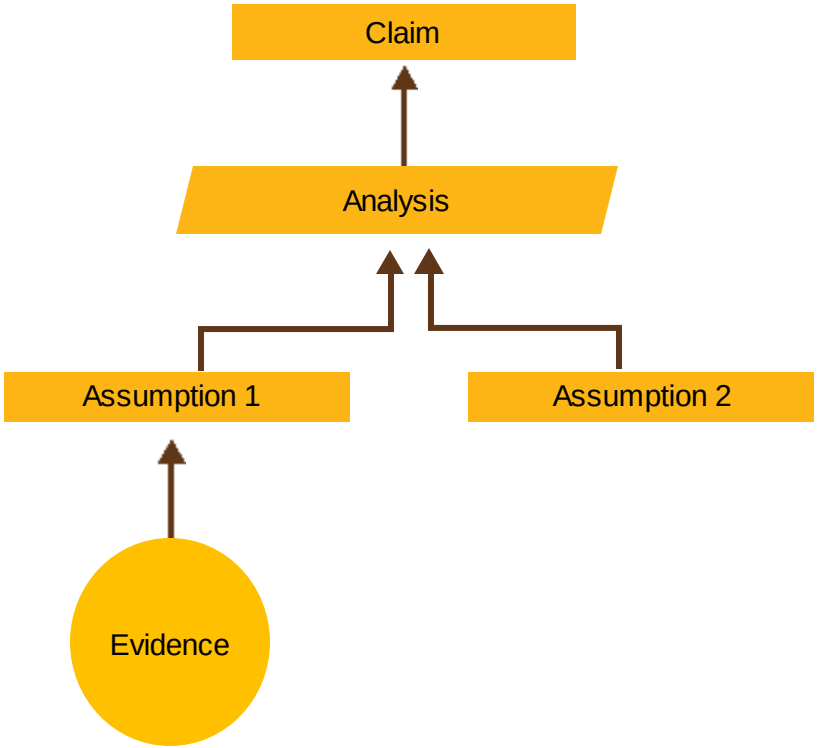
Building Formal Arguments



Multiple analyses are combined to guarantee high-level assurance claims about a system (e.g., the drone flies safe trajectories).

- Assumptions are discharged with other analyses or with evidence.
- The argument can be built incrementally (e.g., replacing partial evidence with the result of a formal analysis).

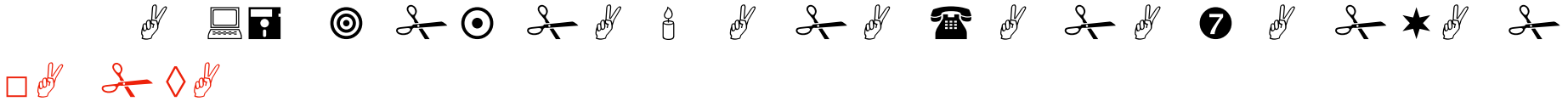
Different Levels of Strength in Assurance Arguments



- Both analyses and evidence can have different levels of strength, depending on
  - comprehensiveness: how many cases they cover
  - rigor: method of obtaining the result, such as theorem-based analysis, model checking, simulation, testing, or inspection
- Both analysis and evidence strength levels impact the strength of a claim.
- We need to "compute" the strength of a claim based on the analyses and evidence that support it.

Handling Levels of Strength with Modal Logic

Modal logic: *multiple worlds* and ***modes in which something can be true***



□ is true in ALL worlds (i.e., cases).

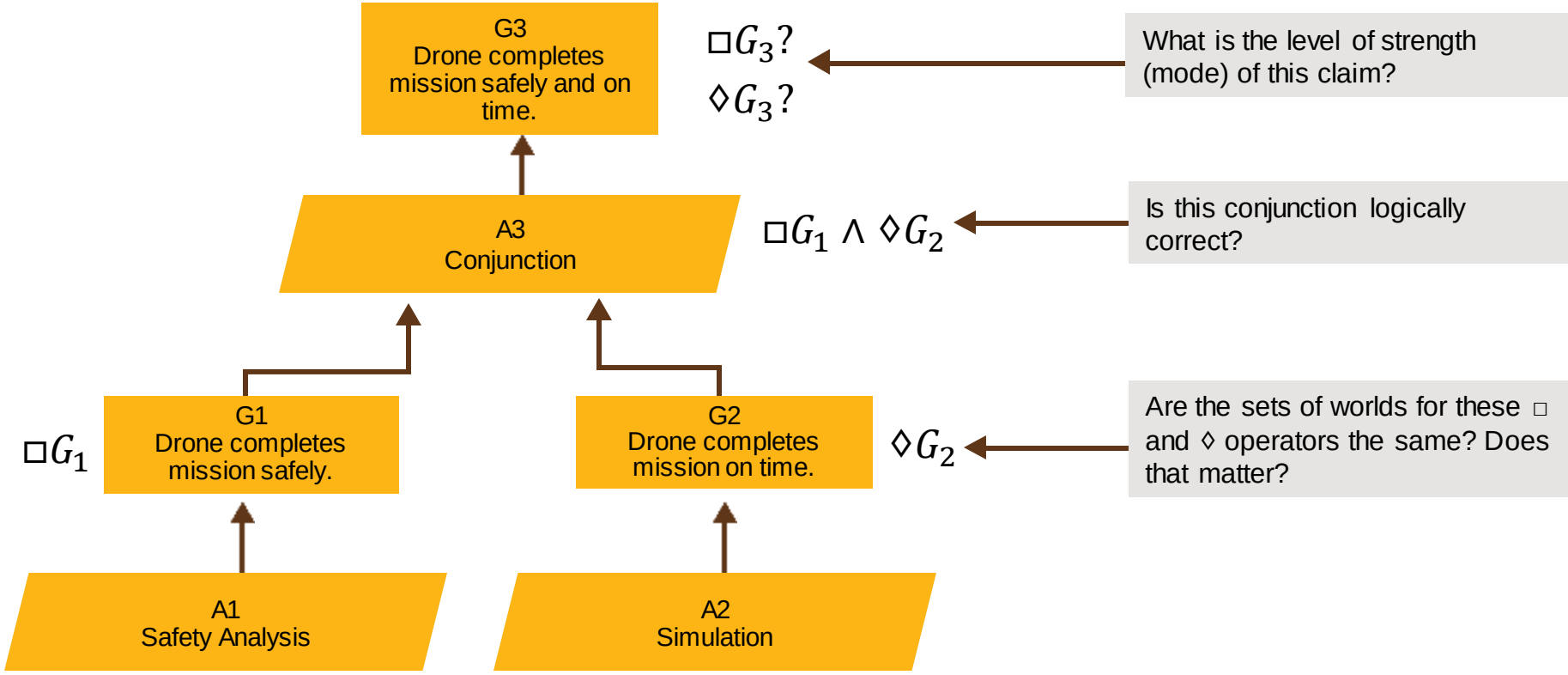
◊ is true in at least ONE world.

Example of modal logic: *temporal logic*

- "World" expresses the state of a system at an instant in time.
- "Box" expresses that a property will always be true.

One goal of our research is to describe how to instantiate "worlds," "box," and "diamond" to represent different types of analyses and claims within a single logic system.

Combining Elements with Different Levels of Strength



Describing “Worlds” Through “Resources”

Carnegie Mellon University Software Engineering Institute

Just as time is the central concept in temporal logic, system analysis is the central concept in our logic.

Analysis domains include

- real-time performance
- logic correctness
- safety
- physical correctness

Each analysis domain draws on information from the system, the platform on which it executes, and the environment in which it is situated.

Collectively, we refer to this information as *resources*.

Modal Logic in the Context of Analyses

Carnegie Mellon University Software Engineering Institute

Worlds, boxes, and diamonds need to account for the following observations:

- Different resources are needed for different analyses.
- Analyses for different domains are performed separately.
- Different analyses can implicitly or explicitly affect one another.
- All analyses must eventually be composed.

We are proposing world types as vehicles for separating the universe of worlds into subsets based on resources.

- World types are defined in terms of the resources they include.
- Modal operators apply to specific world types.

□❄ ✌ is true in ALL worlds of type ❄.

◊❄ ✌ is true in at least ONE world of type ❄📯

World Type Example

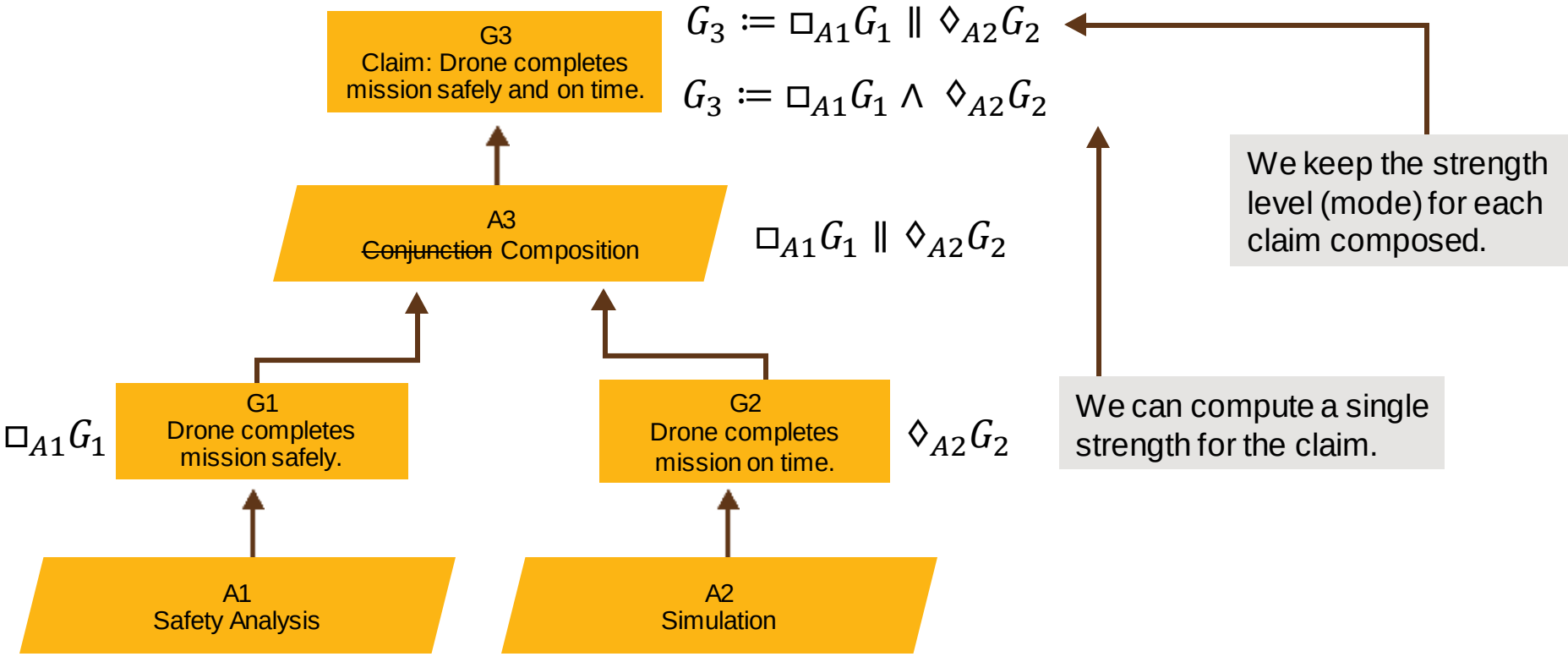
Consider a type of real-time system analysis known as rate monotonic analysis (RMA). One result in RMA expressed in predicate logic is

- If all tasks have rate monotonic priorities: $\forall i, j : T_i < T_j \rightarrow p_i > p_j$
- And total utilization is less than a known bound (UB): $\sum_i C_i/T_i \leq UB$
- Then all tasks meet their deadlines: $\forall i : R_i \leq D_i$

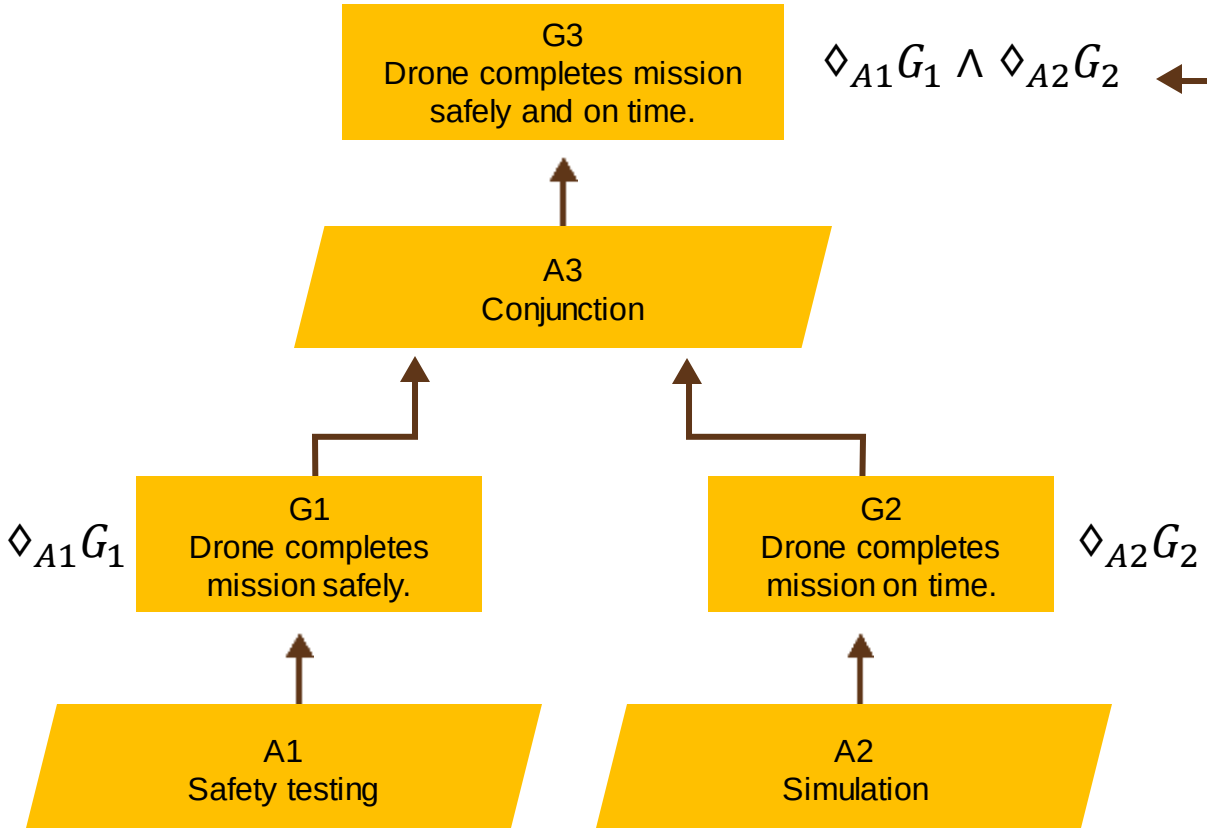
Implicit in this representation is that this is true regardless of the state of all other resources associated with all other analyses.

World types explicitly define the resources of concern to a particular analysis. Using world types, a more accurate representation is □ $R_i \leq D_i$.

Combining Elements with Different Levels of Strength



Combining Elements with the Same Level of Strength



Whether this is sound depends on the world types A1 and A2.

The stronger the claims, the easier to compose them:

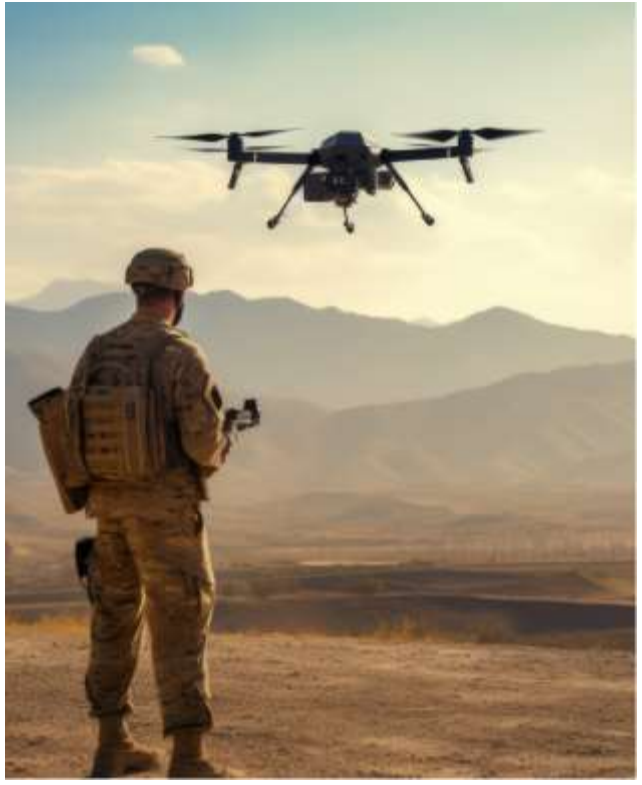
- Conjunction of two claims that are necessary (□) is always feasible.
- Conjunction of two claims that are possible (◊) is feasible only if
  - the analyses producing the claims agree on the resources their world types share and
  - all their other resources are disjoint

Proofs for modal logic rules highlighted this result.

Carnegie Mellon University
Software Engineering Institute

Assuring Physical Correctness

Physical Correctness



(Photo by Staff Sgt. Andrew H. Owen, Virginia Guard Public Affairs)

Many important military systems are cyber-physical systems (CPS).

Assurance arguments for CPS must include quality attributes that are important for a cyber system that interacts with the physical world. These include

- safety
- real-time performance
- logical correctness
- **physical correctness**—The dynamics of how a system moves through the physical world is obviously important.

Scientific machine learning (ML) holds potential for contributing to assurance of physical correctness.

Scientific Machine Learning

Carnegie Mellon University Software Engineering Institute

Scientific ML involves incorporating scientific principles (such as those codified in differential equations) into deep learning models.

Can be used as a numerical method where training the **neural network is constrained** to finding a function that approximates the solution to a differential equation (PINN).

Can also use data to determine differential equation parameters. This is referred to as the "the inverse problem" (inverse PINN).

"Higher-level techniques" – **operator learning** – for solving families of differential equations by approximating the mapping from the "force term" of the equation to the solution (DeepONet).

FALSA's focus

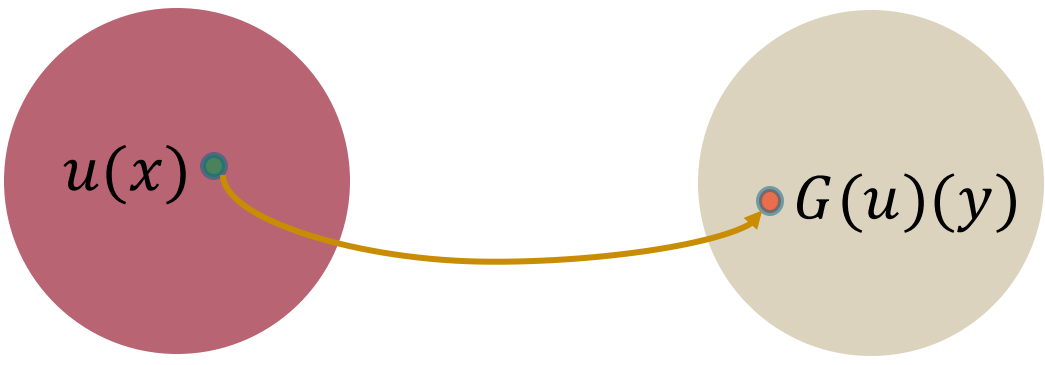
PINN – physics-informed neural network
inverse PINN – inverse physics-informed neural network
DeepONet – deep operator network

Deep Operator Networks (DeepONets)

Theorem 2 (Generalized Universal Approximation Theorem for Operator). *Suppose that X is a Banach space, $K_1 \subset X$, $K_2 \subset \mathbb{R}^d$ are two compact sets in X and $\mathbb{R}^d$, respectively, V is a compact set in $C(K_1)$. Assume that $G: V \to C(K_2)$ is a nonlinear continuous operator. Then, for any $\epsilon > 0$, there exist positive integers m, p, continuous vector functions $\mathbf{g}: \mathbb{R}^m \to \mathbb{R}^p$, $\mathbf{f}: \mathbb{R}^d \to \mathbb{R}^p$, and $x_1, x_2, \ldots, x_m \in K_1$, such that*

$$\left| G(u)(y) - \langle \underbrace{\mathbf{g}(u(x_1), u(x_2), \cdots, u(x_m))}_{\text{branch}}, \underbrace{\mathbf{f}(y)}_{\text{trunk}} \rangle \right| < \epsilon$$

[1]



[1] Lu, L. et al. Learning Nonlinear Operators via DeepONet Based on the Universal Approximation Theorem of Operators. *Nature Machine Intelligence* 3.3 (2021): 218–229.

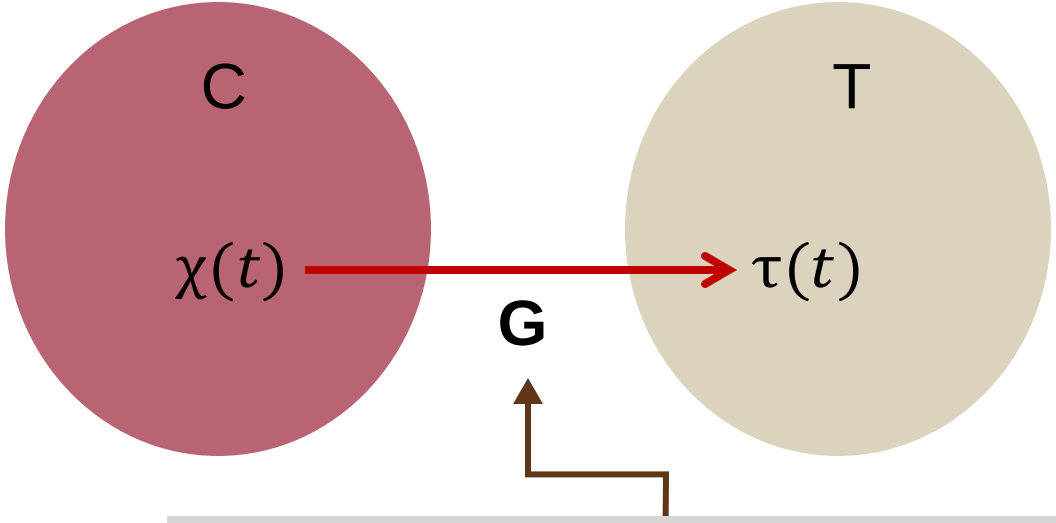
The large circles represent sets of functions. The operator *G* maps functions of *x*, such as *u*(*x*), to functions of *y*, *G*(*u*)(*y*).

The theorem says *G* can be approximated arbitrarily well by a well-crafted dot product, <.,.>. Thus, |G(u)(y) - <.,.>| can be made arbitrarily small by picking **g** and **f** in a smart way.

Moreover, **g** and **f** inspire the structure of a neural network that can serve as a very good representation of <.,.>.

DeepONets for Assuring Control Systems

Predicting trajectories in real time is key to runtime conformance checking of physical correctness.



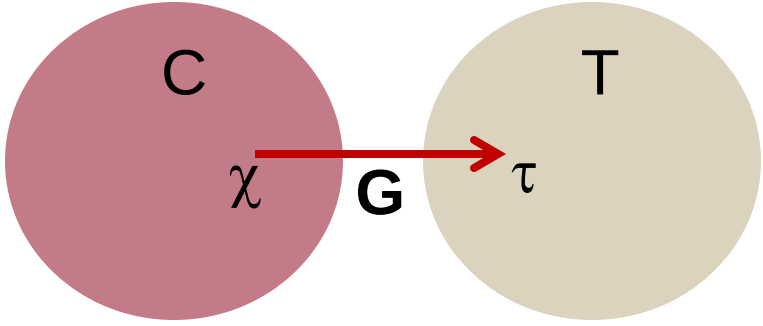
C – The space of all controls
T – The space of all trajectories

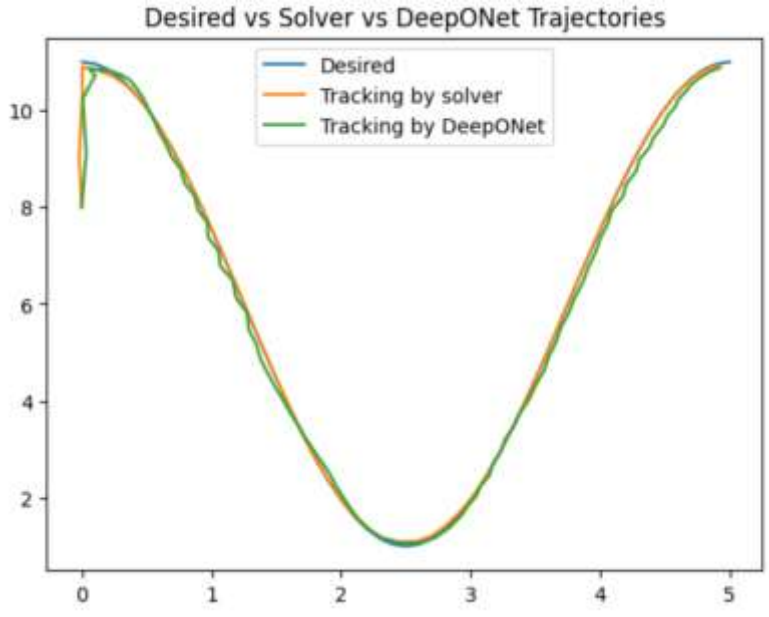
G – An operator that acts on the controls to return a function in trajectory space
G is bijective and continuous.

χ – A specific control (function of time)
τ – A specific trajectory (function of time)

We approximate **G** with DeepONets, enabling fast checking of system behavior against the nonlinear dynamics of the system.

Experiment Tracking a Desired Trajectory with DeepONets





DeepONet tracking a target trajectory:

- When the DeepONet trajectory approaches the target set point, its input set point is changed to a new one on the desired path.

DeepONet allows us to check the physical correctness cyber-physical systems.

Summary

This work aims to speed up the assurance of evolving large-scale systems through

- reuse and integration of diverse assurance analyses
- rapid detection of non-conformance between the system behavior and its assurance argument

We showed how

- modal logic can be used to do sound composition of results coming from different analysis worlds: rules for when they can be composed and how
- assurance claims can be checked against runtime behavior
- scientific ML can accurately predict nonlinear physical behavior in real time to check physical correctness claims

Team

Dr. Gabriel Moreno
Principal Researcher

Mark Klein
Principal Technical Advisor

Dr. Jasmine Ratchford
Senior Machine Learning Research Scientist

Dr. Farzaneh Derakhshan
Assistant Professor
Illinois Institute of Technology

Dr. Anton Hristozov
Software Engineer

Dr. Dionisio de Niz
Technical Director

Dr. Limin Jia
Research Professor
CyLab

Dr. Shambwaditya Saha
Assurance Researcher

John Robert
Deputy Division Director

Dr. Raffaele Romagnoli
Research Scientist
CMU