SwarmStorm: Coarrays in Satellite Simulations

Daniel Topa
daniel.topa@hii.com

Huntington Ingalls Industries
Mission Technologies

December 6, 2024

Contents

Abstract

This document explores the application of Fortran coarrays to simulate satellite engagements. By leveraging coarrays' parallelism and synchronization capabilities, the simulation models packs of satellites as coarray images and individual satellites as coarray indices. The report outlines the foundational principles, architectural considerations, and implementation details of this innovative approach.

1 Genesis

The use of coarrays in satellite simulations provides a powerful paradigm for modeling distributed systems. This section explores the foundational concepts and initial design considerations for simulating satellite engagements using coarrays in Fortran.

1.1 Concepts and Design Philosophy

The simulation models satellite engagements, where:

- **Packs of Satellites**: Each pack is represented by an image in the coarray model.
- **Individual Satellites**: Satellites within a pack are indexed as elements in coarrays.
- **Engagement Dynamics**: Packs engage enemy packs, sharing data and state across images via coarray communication.

This structure leverages Fortran's coarray capabilities to parallelize and simplify the simulation:

- **Parallelism**: Each image independently simulates its pack, enabling concurrent computation.
- **Data Sharing**: Coarray variables facilitate efficient inter-image communication.
- **Scalability**: The design can accommodate increasing numbers of satellites and packs.

1.2 Initial Considerations

The key architectural choices are:

- **Packs as Images**: Each image represents a pack of satellites. Internal dynamics are simulated locally, while inter-pack engagements are handled via coarray references.
- **Satellites as Coarray Indices**: Satellites within a pack are indexed in coarrays. This simplifies data management and ensures efficient access to satellite states.
- **Synchronization**: Use coarray synchronization primitives (`sync all`, `sync team`, etc.) to manage interactions between packs.
- **Encapsulation**: Encapsulate satellite behavior within types, ensuring clean separation between simulation logic and data structure.

1.3 High-Level Design

The simulation follows this high-level flow:

1. **Initialization**: Distribute satellite data across images and initialize states (e.g., position, velocity, fuel).
2. **Engagement Simulation**: Simulate pack dynamics within each image and coordinate inter-pack engagements using coarray communication.
3. **Result Aggregation**: Gather and summarize results for analysis (e.g., pack status, satellite losses).

This design ensures scalability, modularity, and efficient use of Fortran's coarray features. Future sections will explore specific implementation details and results of these simulations.