National Aeronautics and Space Administration

NASA

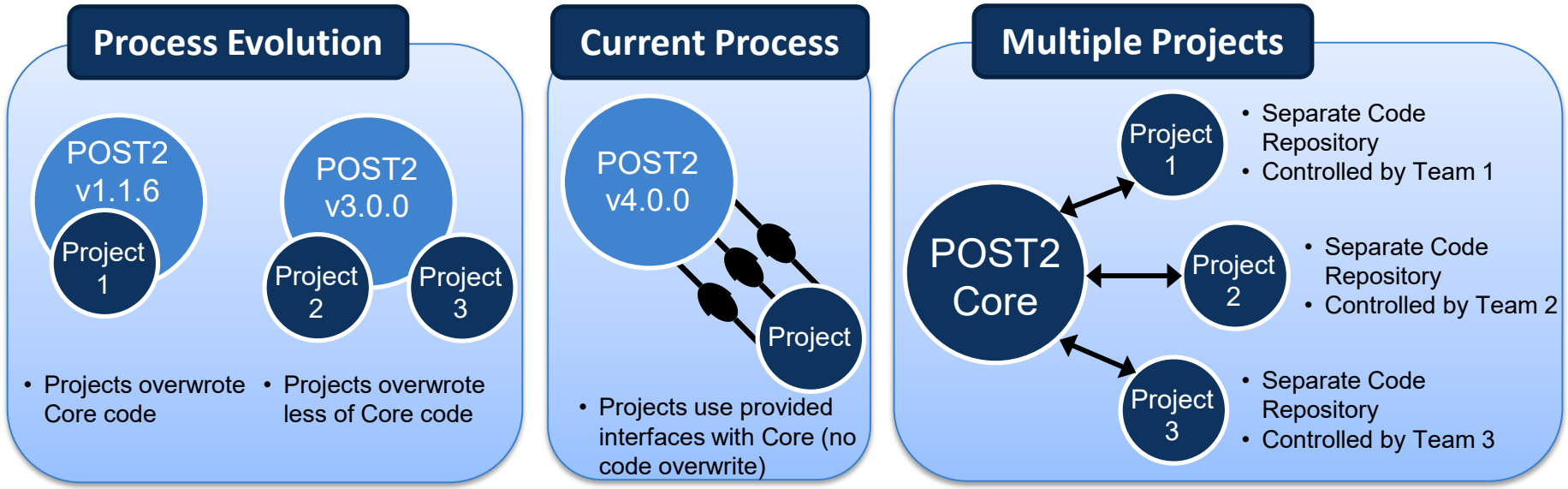
Enabling Thread Safety and Parallelism in the Program to Optimize Simulated Trajectories II

Anthony Williams, Rafael Lugo
NASA Langley Research Center

Steven Marsh, James Hoffman, Jeremy Shidner, John Aguirre
Analytical Mechanics Associates

Background and Motivation

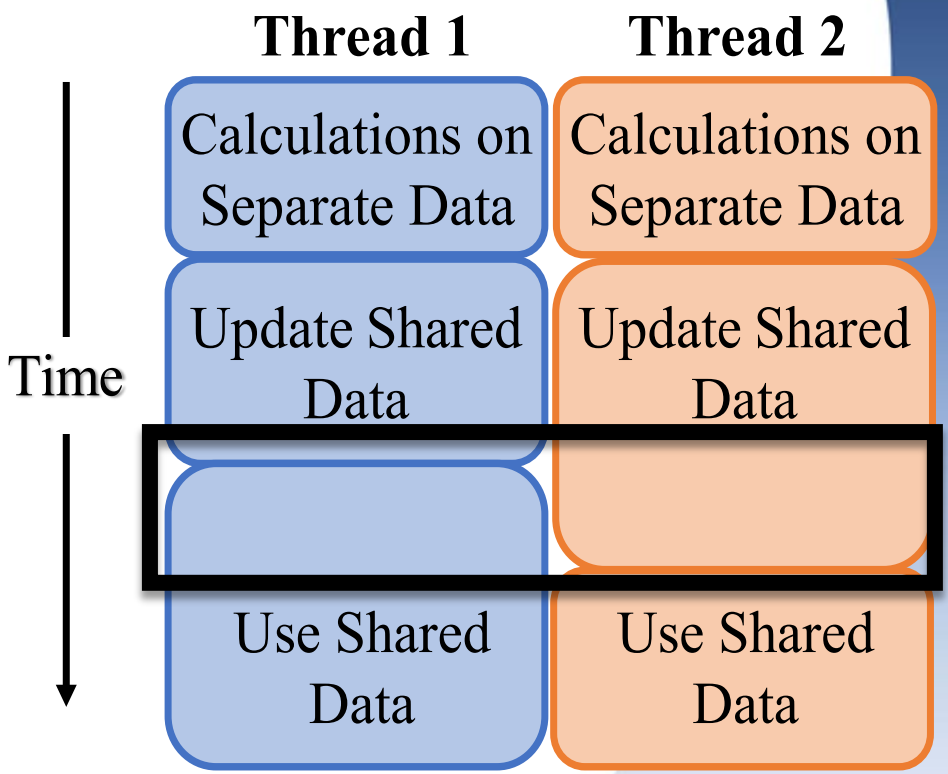
- The Program to Optimize Simulated Trajectories II (POST2) is a widely used, workhorse trajectory simulation tool that has been used to solve a variety of atmospheric ascent and entry problems, with one or more vehicles
- POST2 development began in the 1970s, and has undergone continuous development and updates at NASA Langley Research Center (LaRC) since the conversion from all Fortran to C with C++ in the late 2000s



- Motivation is to enable thread safety to leverage high performance computing (HPC), such as multi-threaded central processing units (CPUs) or graphics processing units (GPUs), to improve POST2 performance, capability, and adaptability

Thread Safety

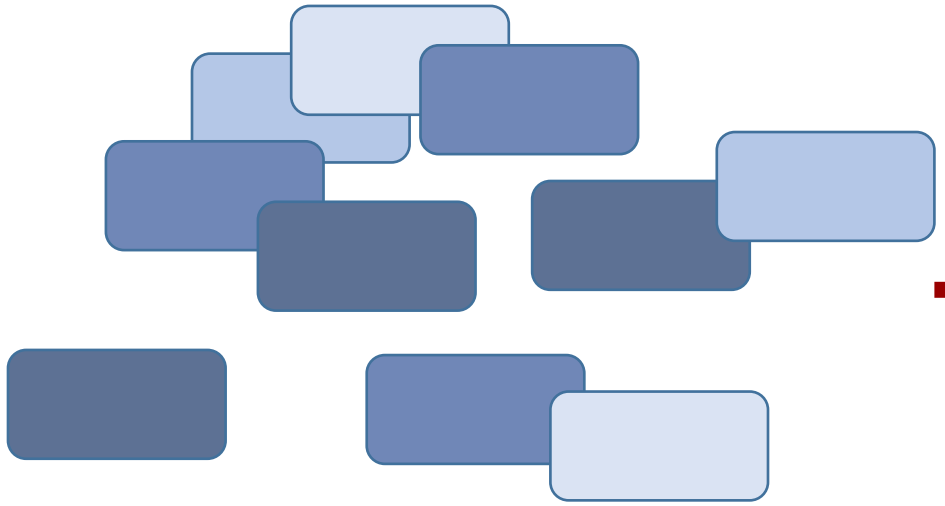
- **Software is thread-safe if the parallel execution has no effect on the results**
- **A major issue preventing thread safety is data races, when shared data is begin read from and written too simultaneously**
- **Memory access must be analyzed, as well as execution to ensure thread-safe code**



Memory Structure

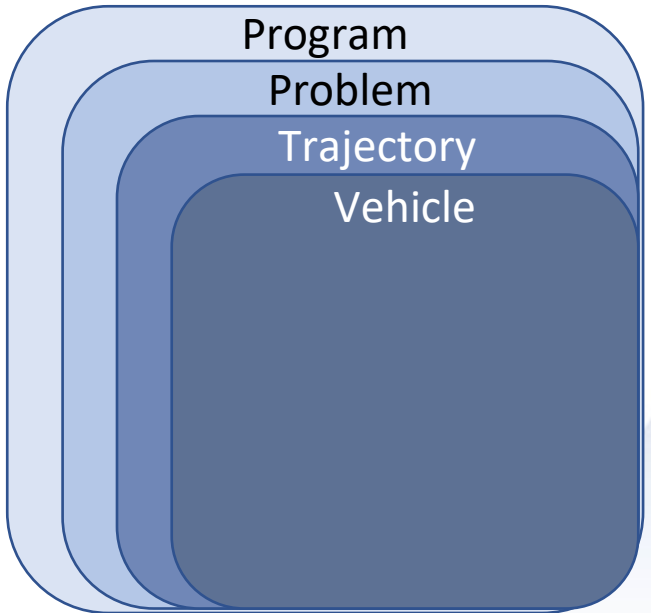
Before Thread-Safe Changes

- Many global memory structures, remnants of Fortran common blocks
- No grouping within global structures
- Branches of code written specifically for first, global vehicle



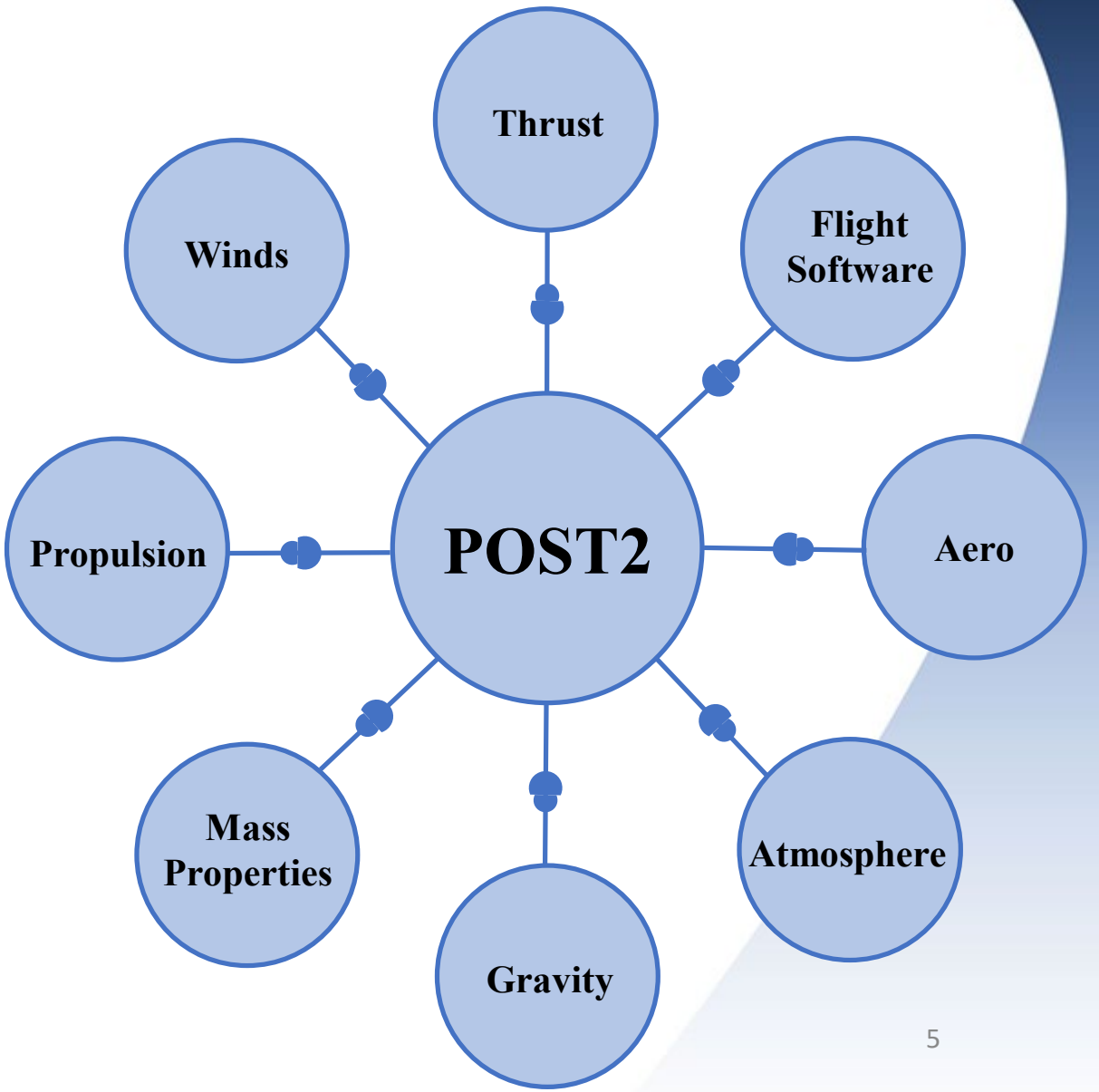
After Thread-Safe Changes

- Global memory space restructured into four-level hierarchy based on instruction execution
- Vehicle specific logic removed



Function Pointers

- **Used to interface custom model code for a particular project or mission by hooking into POST2 sockets**
- **Eliminates the need to change Core POST2 code**
- **User code must match specific function signature (return type and argument type)**
  - Before Thread-Safe Changes: 2 function pointer types, leveraging global memory access
  - After Thread-Safe Changes: 1 function pointer type, can only access memory through passed in arguments



User Input

- **User input is one of the features that allows POST2 to be generalized**
- **Defines initial conditions, physics model types, integration method, user models via function pointers, etc.**
- **POST2 tables are one of the more powerful and useful aspects**
  - $y = (a * b)x + (c * d)$
  - Evaluated at the simulation rate
  - Can be multi-dimensional

x: value of lookup table (**POST2 variable**)
a: numerical component of table multiplier
b: mnemonic component of table multiplier (**POST2 variable**)
c: numerical component of table bias
d: mnemonic component of table bias (**POST2 variable**)

- **Generalized equations**
  - Range from very simple to very complex, including conditionals, nesting, and most standard math functions (sin, pow, etc.)
  - Nearly any combination of **POST2 variables**
  - Evaluated at the simulation rate

User Input

- **POST2 tables are one of the more powerful and useful aspects**
  - $y = (a * b)x + (c * d)$
  - Evaluated at the simulation rate

x: value of lookup table (**POST2 variable**)

b: mnemonic component of table multiplier (**POST2 variable**)

d: mnemonic component of table bias (**POST2 variable**)

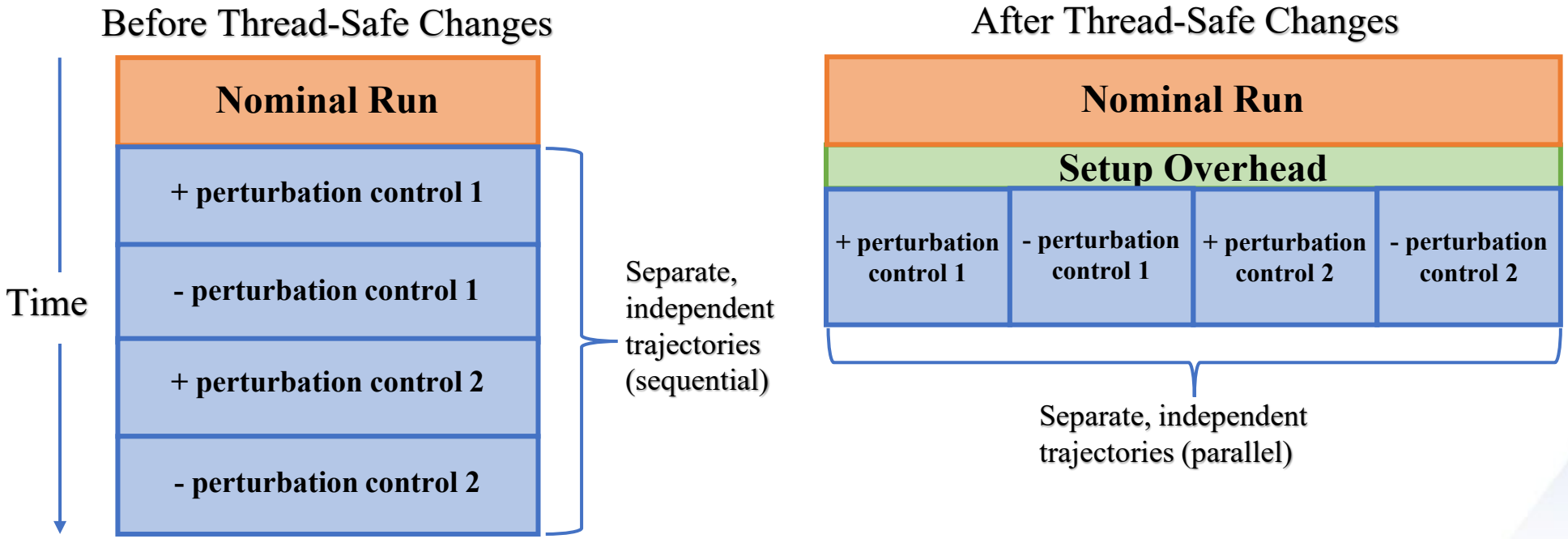
- **Flexibility to utilize any POST2 variable poses challenge**
- **Need to log information on variables so that value at any point in simulation can be found**
  - Nearly any combination of **POST2 variables**
  - Evaluated at the simulation rate
  - Before Thread-Safe changes: Store absolute address to be dereferenced later, relative only to existing Trajectories or Vehicles
  - After Thread-Safe changes: Store relative address offset from template Trajectory/Vehicle, relative to any future Trajectories or Vehicles

Multithreaded Execution

- **The work to be completed in parallel needs to outweigh the amount of time it takes to divide the work amongst multiple threads**
- **Qualitative analysis showed that Trajectories (level) running concurrently was the most impactful**
  - At the Vehicle level, not enough work to justify splitting it up
- **For some optimization and guidance schemes multiple trajectories need to be simulated**
- **Open Multi-Processing (OpenMP) was the Application Programming Interface (API) chosen for parallelizing threads**
  - Leveraging the `parallel for` construct
  - For this work, only utilizing CPUs

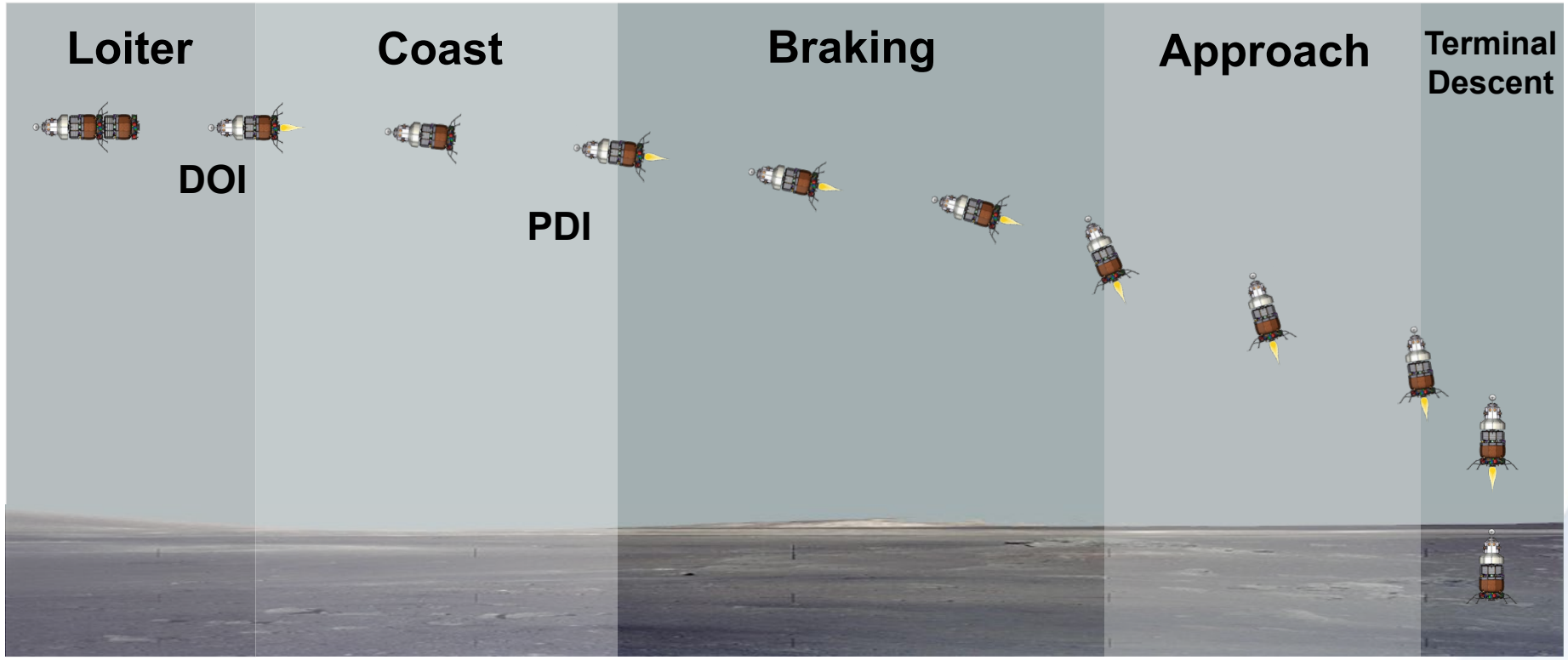
Parallel Optimization Framework

- **Default optimization scheme in POST2 is the projected gradient method**
- **Multiple derivatives are needed to build a sensitivity or Jacobian matrix**
  - Controls for the optimization problem determine derivatives needed
- **Derivatives approximated via finite differencing**
  - Depending on type, e.g. central 2nd order, the number of trajectories needed changes



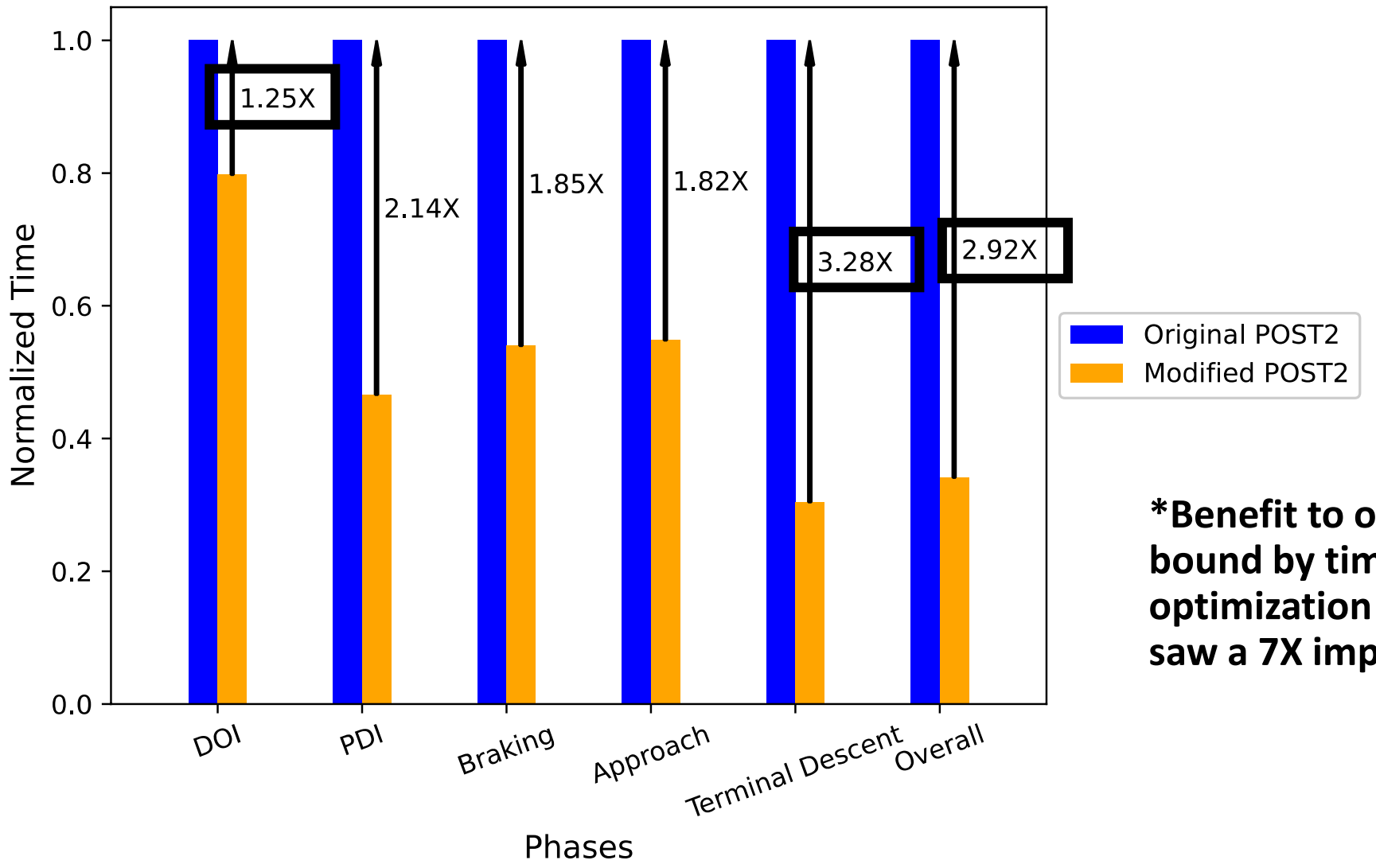
Human-Scale Lunar Lander

- NASA government reference human-scale Lunar lander was set to utilize parallel optimization [AAS 20-592]
- 3 degree of freedom (DOF) simulation broken into multiple phases that each have optimization



Lunar Lander Results

Lower is Better



***Benefit to overall time bound by time spent within optimization region, which saw a 7X improvement**

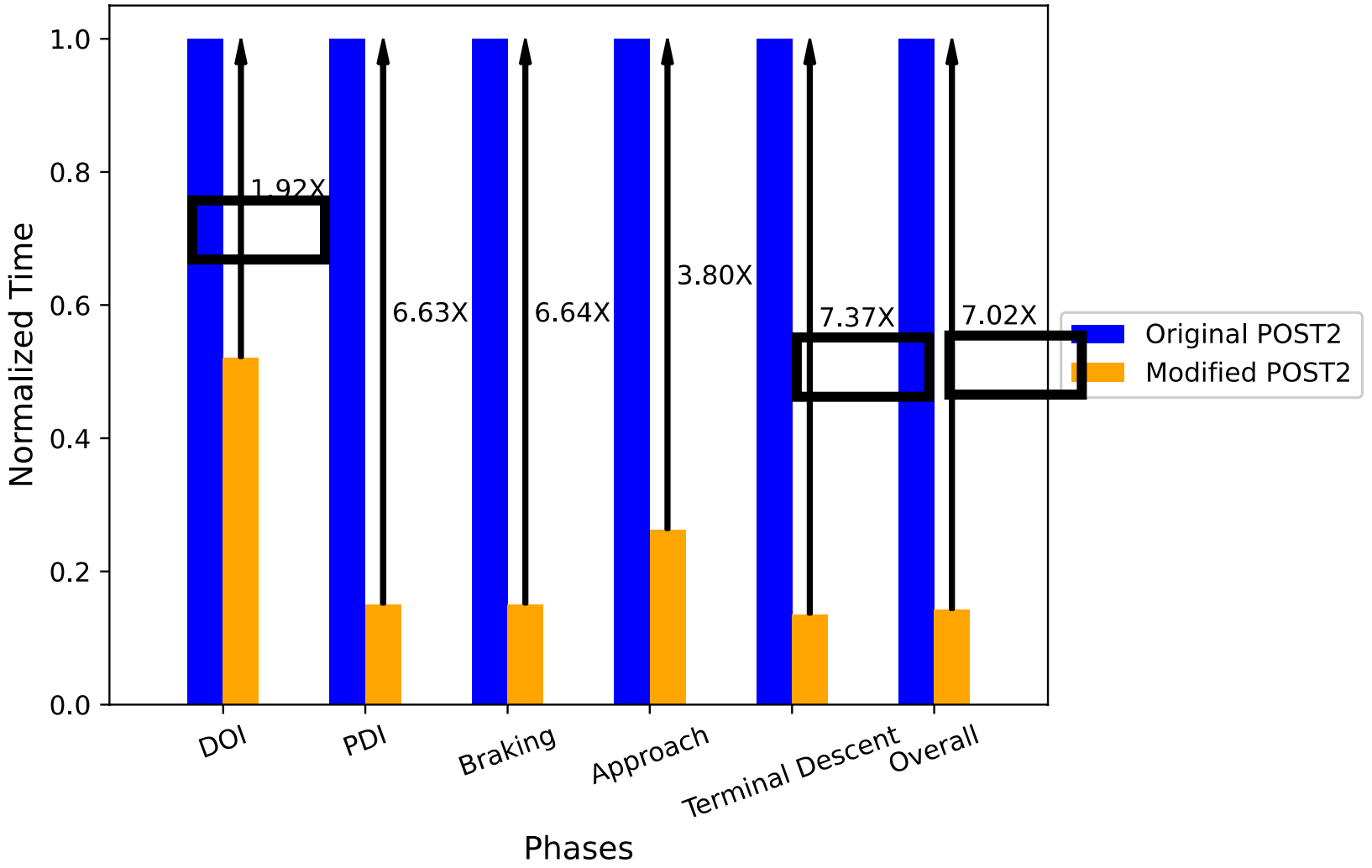
Summary and Future Work

- Heritage trajectory simulation tool, POST2, was modified to be thread-safe
- POST2 was then upgraded to enable parallel calculations to generate optimization solutions
- Lunar lander simulation utilizing parallel optimization shown to improve overall run time approximately 3X
- Additional efficiencies can be gained by utilizing more complex OpenMP constructs, as well as modifying when the parallel trajectories are launched within the execution
- Memory restructuring allowed for task creating API for POST2
- If interested in requesting access to POST2, https://www.nasa.gov/post2/request

Backup

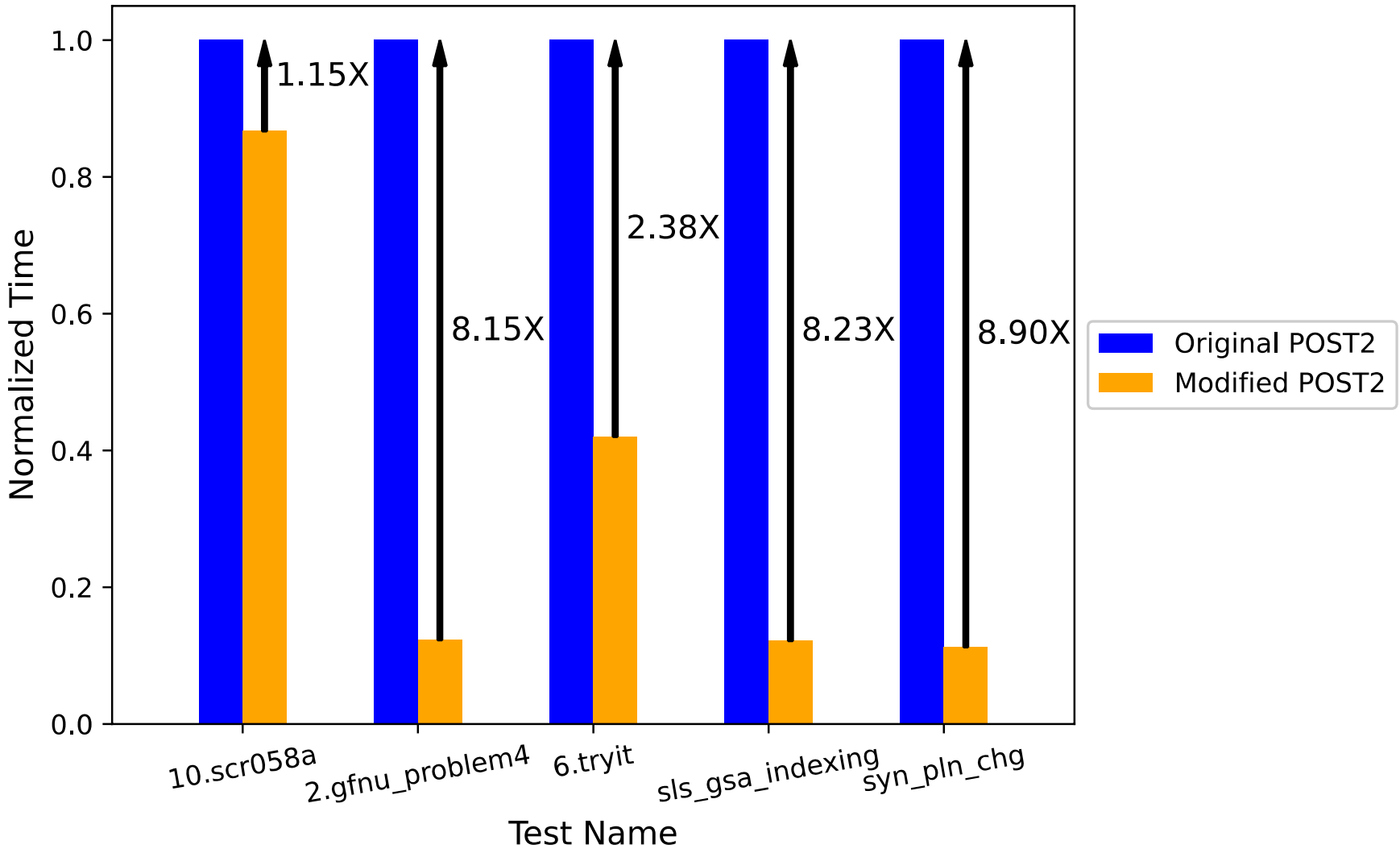
Lunar Lander Gradient Calculations

Lower is Better



Regression Tests Gradient Calculations

Lower is Better



Regression Tests Overall Time

Lower is Better