Hurricane Storm Surge Analysis and Prediction Using ADCIRC-CG and AMPI

Part - 2 AMPIzation

Presented by Eric Bohm
Team: Damrongsak Wirasaet, Dylan Wood, Sam White, Justin Szaday
Laxmikant V. Kale

ADCIRC-CG

- ADCIRC-CG : FORTRAN based MPI code
    - > 100k SLOC
    - Wet cells <- real physics computation
        - Mostly 2d with relatively fine granularity
    - Dry cells <- bookkeeping, is water here yet?
- Dynamic load imbalance emergent from the simulation

AMPI

- Each AMPI Ranks run in a user level thread
    - Breaks the Rank <=> Process assumption
        - Supports migratability and therefore load balancing
        - Process global mutable state is a correctness issue
            - Mostly equivalent to privatizing global state to be specific to the user level thread
- AMPI supports a variety of solutions

TLS Privatization

- Thread local storage is supported in FORTRAN
  - -fopenmp
- Can also be leveraged by Charm++ user level threads
- Supports virtual AMPI ranks
  - Add -tlsglobals to ampiCC compile line
- Add !$omp threadprivate(YourVariable)
  - For each variable declaration

TLS in FORTAN

- Over 2,000 module variables in ADCIRC-CG
    - Motivating an automatic transformation approach
- Fortran global state has several complications
    - Variables declared at the Module level are global <- **TLS**
    - Variables declared with Parameter attribute are constant and cannot be thread local
    - Variables declared in a subroutine with Save attribute have global extent <- **TLS**
    - Variables declared in a subroutine with an initial value are implicitly Save <- **TLS**
    - Variables declared in a subroutine are otherwise local
    - Common blocks promote variables listed therein to global
        - The common block needs **TLS**
            - the variables within the common block do not.

FORTRAN I/O

- FORTRAN I/O assigns the Logical Unit Number (LUN) in every IO statement.
    - Roughly equivalent to a stream or file identifier, the LUN is global to the FORTRAN runtime.
    - Therefore, each virtual rank performing file operations must have its own LUN
    - Code transform must alter each I/O operation to add a virtual rank offset to every LUN
        - e.g., WRITE(UNIT=APPLUN, …)
            - -> WRITE(UNIT=APPLUN+CK_LUN,…)
- Must NOT transform when the LUN variable is not an INTEGER
    - FORTRAN I/O commands may also be used to operate on “Internal” data.
        - String operations that look like file operations
- Must NOT transform operations that merely resemble file operations

(Semi) Automatic FORTRAN AMPIzation

- FLANG was not (still not) at a level where it could support these
- Actual transformation are relatively simple
    - Perl implementation using REGEX based parsing of FORTRAN variable declarations and I/O statements
- Adds thread private only where necessary
- LUN privatization module
    - Insert an import of ck_lun to each module with FORTRAN I/O
    - Arithmetically add ck_lun to each fortran I/O statement LUN
        - Unless that LUN is not an integer
            - Detect that based on extracting variable type by module so the type of each LUN in scope can be determined
- Need to close and reopen open LUNs when a rank is migrated out of the process
    - Required manual implementation by Damrongsak Wirasaet <dwirasae@nd.edu>

LUN Migration

- Implement about_to_migrate callback
  - Record each open file and its LUN data
  - Close each open file
- Implement just_migrated callback
  - Reopen each file, restore LUN data
- Register callbacks with
  - AMPI_REGISTER_ABOUT_TO_MIGRATE
    AMPI_REGISTER_JUST_MIGRATED
- These will be triggered automatically when a virtual rank migrates

ADCIRC Status

- Runs in AMPI on SMP and non-SMP
  - Initial port ran into a hang bug when virtualization ratio > 1 OR PPN >1
  - Symptom was MPI_WAITSOME returns MPI_UNDEFINED (-32768) in the count of completed receives.
    - Further study indicated a bug in the AMPI implementation of MPI_WAITSOME
      - Fixed by Sam White
- Virtualization supported, dynamic load balancing demonstrated
  - Instability in long runs
  - NetCDF input not currently supported
  - Virtualization negatively impacts performance
    - MPI_All_Reduce performance on sub-communicators has excessive overhead in the current AMPI implementation

Future Work

- Implement transforms in FLANG - Justin
- Metabalancer integration to trigger balancing when necessary
- AMPI improvements to sub-communicator collective performance - Sam
- Adapt application specific balancing hints
  - Sort by elevation, etc. - Dylan
- Experiment with graph partitioning based balancers