RELAP5-3D Improvements 2014

Dr. George Mesina

RELAP5 International Users Seminar
Idaho Falls, ID
September 11-12, 2014

INL/CON-14-32915

Outline

- Recent Issues and Solutions
- New Code Capability
- Announcements and Documents

Issues going from 4.1.3 to Version 4.2.1

- Verification issues and developments are covered in another presentation
- Issues
  - Order of evaluation in if-tests, Part 2
  - Allocation/Deallocation Issues
    - Memory Leaks
    - Initialization and nullification

Order of Evaluation of IF-tests

- The order of evaluation is left to right in the C language and numerous other programming languages
- ANSI FORTRAN does not enforce this in any standard
  - Historically, it has been left to right on most computer platforms
  - With multi-core processors, it is seldom the case anymore
- This affects many kinds of statements. Examples (.OP. means logical operator:
  1. IF (G(i) expression) .OP. (F(i) expression)) THEN
  2. IF ((protection clause) .AND. (protected clause)) THEN
  3. IF ((protection clause) .OR. (protected clause)) GO TO LINE
- Cases 1 and 2 covered on 2013 presentation

Order of Evaluation: OR-clause

- In left-to-right evaluation, optimized evaluation stops whenever the first clause (the protector) is true. The second is never evaluated
- The impact of evaluating the right-hand (protected) clause before the left-hand clause varies from negligible to core-dump
- Examples

```
1. IF (j <= 0 .OR. a(j) == 1.0) go to 100
2. IF (value < 0.0 .OR. sqrt(value) > 1.0) GO TO 10
```

- First example: array accessed out of bounds: negative
  - Assumes a prior statement tested out-of-bounds positive indices
  - When compiled with debug on, the code will stop executing
  - Possible stoppage by O/S
- Second example: core dump

Order of Evaluation: Resolution Method

- 43216 if-tests to examine. Manual checking risks missing some
- Locator program was written to find statements that have:
  - IF or ELSEIF with an OR-operator
  - Variable in parenthesis that is isolated in a preceding clause
- Reduced to 1505 if-tests to check
- 5 statements found and corrected manually

- Resolution Method: break the if-test

```
        IF ((protection clause) .OR. (protected clause) ) GO TO 10
```

- Becomes

```
        If (protection clause) GO TO 10
          If (protected clause) GO TO 10
```

Allocating and Deallocating Memory

- Errors addressed
  - Failure to initialize and nullify
    - E.G. UP 14023, Hydro Matrix destroyed when insulator (material 9) table was extended on restart
    - Cause: Subroutine QFMOVE allocated numerous arrays without initializing any of them
  - Memory leaks
    - E.G. UP 14001, RELAP5-3D/PHISICS coupled code lost memory access when running on the INL cluster
    - Cause: Subroutine RRKINO allocation not properly protected
  - Hanging of the machine (in a non-parallel process!)
    - This has only occurred in restarts with multiple input cases

Allocating and Deallocating Memory

- Failure to initialize arrays when allocated means that the array holds whatever value was left in memory
  - Floating point calculations can change
  - Indexing arrays can cause out of bounds memory access that fetch wrong values or destroy data
  - Such errors can occur randomly and can be difficult to track down
- Failure to nullify (or point) pointers when created can cause code failure

Initializing Allocated Memory

- RELAP5-3D developer rules
  - Memory should be initialized whenever it is allocated
  - Pointers should be nullified or given a target upon creation
- Can be done via a loop or allocate statement
- New RELAP5-3D allocation module subprograms do this for basic data type arrays
  - Uses Fortran 2003 “source” keyword on the allocate statement to zero out the array
- For derived types it requires a derived type scalar zero
  - A project is underway to create derived type zero “scalars”
    - Created under type declaration using “structure” of derived type declaration (for ease of future development)

Initializing Allocated Memory

- Module allocation subroutines use the zero scalar on the “source” keyword of the allocate statement
  - E.G. module VOLMOD
  - VOLDATZERO has over 400 components (including if-defs)
  - In subrotuine vlmAllo
    - ALLOCATE (vlm(nvlm), SOURCE=voldatzero, STAT=istat)
- Similarly, pointers should be nullified or initialized upon creation
  - Scalar pointers should be immediately nullified
  - Many derived type arrays have array components declared as pointers
    - These can be initialized when allocated as above

Deallocating Memory

- Memory leaks occur when a computer program incorrectly manages memory allocation
- In RELAP5-3D, these occur when memory is repeatedly created and destroyed incorrectly
  - The incorrectly handled memory cannot be used for future allocation during the code run
  - The code may quit for lack of available memory
- Some sources:
  - Allocate/deallocate in the transient
    - Should not be done except for reflood where it is required
    - Besides memory leaks, it slows the code down
  - Multiple-case input decks can build-up memory leaks
  - When attached to massively parallel program, such as RAVEN

Deallocating Memory

- Some ways to create memory leaks
  - Eliminate a pointer without first deallocating it
  - Deallocate a derived type without first deallocating its pointer components
    - Large amounts can be lost if pointer is itself a derived type array
- When run with PHISICS on the massively parallel INL cluster, many memory leaks occur
  - Two that affected results traced to input programs RRKINO and RADHT and fixed
- Another leak affected general tables for a NuScale model
  - Traced to RGNTBL and fixed
- Many more remain to be corrected

Deallocating Memory

- It is an error to allocate an array that is already allocated and to deallocated one that is not allocated.
- Always check the allocation status before allocating/deallocating

```
IF (.NOT.ALLOCATED(a))  THEN
     ALLOCATE(a(na), SOURCE=0.0, STAT=istat)
     IF (istat == 0) then
          – ALLOCATE(a(1)%b(nb), SOURCE=0.0, STAT=istat)
     …
```

New Code Capability

- Nodal Kinetics Upgrades (Barber)
- Coupling with RAVEN and PHISICS (Rabiti)
- Added verification capability (Mesina)
  - Multi-case input decks
  - PC verification
  - Input modification
- Moving Systems

Announcements

- 2 Dedicated RELAP5-3D Journal Issue
  - 31 Abstracts received
  - Targets are:
    - ANS Nuclear Technology (NT)
    - ANS Nuclear Science and Engineering (NSE)
  - There is also interest from ASME
  - Papers due October 15, 2014
  - Will be published in 2015
- RELAP5-3D News
  - http://www.inl.gov/relap5/news
  - LinkedIn RELAP5-3D Group
  - https://www.linkedin.com/groups/RELAP5-3D-Users-Group-4483821

Announcements

- RELAP5-3D “official” compiler is Intel Fortran 13.1
- INL IT supports Windows 7 and SUSE Linux platforms
  - Windows XP is no longer supported
  - Windows 8 is still not supported
  - No other Linux is supported (in particular: Cygwin and Redhat)
- INL RELAP5-3D Team supports installation on
  - Windows 7 with Visual Studio 2012
  - Linux via Linux C-shell scripts and Makefiles
- It is possible to install RELAP5-3D on Macintosh systems, but INL department does not support this

New Documents for RELAP5-3D and Auxiliaries

- PROGRAMMING
  - G. L. Mesina, “Guidelines for developing RELAP5-3D coding**,** INL/EXT-13-29228, Rev 2, June 2014
- INSTALLING
  - J. H. Forsmann, G. L. Mesina, “RELAP5-3D Windows 7 Build,” INL/MIS-12-27541 Rev. 1, October 2012
  - J. H. Forsmann, “RGUI Configuration Guide ,” GDE 648, INL/MIS-13-30082, July 2014
- RUNNING
  - J. H. Forsmann, J. E. Fisher, G. L. Mesina, “PYGMALION User’s Manual,” GDE-621, INL/MIS-13-28216, INL/MIS-13-30083, March 2013
  - J. H. Forsmann, “RGUI Help Manual: RELAP5-3D Graphical User Interface,” GDE 649, INL/MIS-13-30083, July 2014

SUMMARY

- Recent code issues presented here with solutions
  - Order of operations, part 2
  - Allocation and deallocation
- Progress has been made on initializing and nullifying
- New capability has been added in the code
- A journal issue dedicated to RELAP5-3D will be published next year
- Updated information is available on RELAP5-3D usage
- Keep in touch with RELAP5-3D on the web site and social media