Timing Study of RELAP5-3D Series 4 versions

George Mesina

January, 2017

Reports that recent RELAP5-3D code versions run different input models faster or slower for some problems than older versions are examined. Versions studied (Series 4) were released after 2011.

Introduction

Two different reports of newer version 4.3.4 of RELAP5-3D running slower than older ones, such as version 2.4.1, have been received. This prompted a study of runtimes for multiple versions. Version comparison is difficult due to changing compilers, operating systems, computer platforms, and many other factors. Some preparatory work is required to make a fair, "apples-to-apples" comparison of code versions; this article describes this work. Some measurements of code speed comparison are developed and used then conclusions are drawn.

Background

The conversion of RELAP5-3D to allocatable memory rather than common blocks resulted in slower code performance as compared to the fixed-memory size versions, namely 2.4.1 and earlier. To access allocated arrays, interaction with the operating system is required. This takes longer than access to fixed memory arrays. The trade-off, that RELAP5-3D now resizes itself to exactly fit an input model, eliminates memory problems that fixed memory versions of RELAP5-3D had. If a model was too big, it couldn't run.

Other changes that have resulted in increased code runtimes over the years include:

- Use of array sections
- Use of pointers
- New features
- Correction of errors
- Disruption of vector loops
- More subroutine calls within loops
- Existence of potential I/O that slows processing even when not active

Some of the new features and bug fixes were not reworked for efficiency; these could be rewritten to improve performance. This is not an exhaustive list of code changes that affect speed. All of the items contribute to slowing down the code in places. However, some improvements actually make the code run faster for certain operations. Therefore, before taking any action to produce faster run speeds, an assessment is in order.

Preparatory Work

In the most recent code version, 4.4.1, a timing capability was introduced. It can be activated as an installation option or run later from the "runx" script in the run/ directory. This option uses new scripts and a Fortran 90 timing calculation program to run the standard installation and the verification suites, calculate transient duration by subtracting input processing time from the final time, and storing the times on a timing file. The new Timer/ subdirectory of the run/ directory stores the scripts and program. The Timer/ software runs when the installation scripts are invoked using the fourth argument, "timing."

For both suites, another script runs each input file five times and invokes the Fortran 90 averaging program to calculate the average runtime of each problem, the number of files in the suite, and average cumulative time of all runs in the suite. It outputs this on the average-timing file.

Comparison of these timings with those of another version can be made at any point provided that the other version has timing capability. For this, the older version must be backfit with the necessary subdirectories, scripts, changes to existing scripts, and appropriate input files. Since older versions have no verification suite, it is copied into the run directory and only the base verifications cases, no restart or backup, are run. Since some installation suite input decks changed over time, they are replaced by the timing versions of the files so the same calculations are requested.

Using the timings for two versions, the timing comparison program computes average times for both versions and outputs the version names, average time for each, percentage change in runtime, the number of files for which the first version is faster than the second and vice-versa, the total runtime of each, percentage of change, and an assessment of which version is faster.

Timing Comparisons

Since newer versions have more input decks, the installation suite runs longer. Some longer-running input files were placed in the LongRun/ directory to guarantee version comparison on the same set of inputs. Averages of five runs per file are used to compare versions. Table 1 compares 4.1.3 and 4.2.1.

Table 1. "LongRun" Comparison of versions 4.1.3 and 4.2.1

```
Compare r3d413t to r3d421t
    * Left (1st) and right (2nd) files: Time_r3d413t.Tavg, Time_r3d421t.Tavg
    * Faster means right (2nd) time < left (1st) time
        Input model,  left time, right time,    diff.,  percent,   rate
       arteryFlex.p:  3.9904E+01  3.9373E+01     0.531     1.330  Faster
           hex2d1.p:  1.2098E-01  1.2181E-01    -0.001    -0.686 slower
      pois_cyl_he.p:  1.1434E+01  1.1330E+01     0.104     0.912  Faster
           sschf1.p:  5.2541E-02  4.9974E-02     0.003     4.886  Faster
           todcnd.p:  4.1683E-02  7.2336E-02    -0.031   -73.538 slower
         typ12002.p:  1.3304E+01  1.3777E+01    -0.473    -3.556 slower
        typ1200n2.p:  1.5592E+01  1.5100E+01     0.492     3.157  Faster
Number faster / slower / same:   4   3   0
Average of percentage change:  -9.642E+00
Suite Runtime: 1st, 2nd: 8.0449E+01, 7.9824E+01
Change in test suite runtime:    -0.625
Percentage change in test suite runtime:    -0.777
On average, Time_r3d421t.Tavg is FASTER than Time_r3d413t.Tavg
```

Table 2 compares 4.1.3 and 4.3.4 of LongRun/ input models. In both tables, the hexagonal grid neutron kinetics and the semi-implicit typical PWR problems runs slower in the newer version while almost all other models run faster in the newer versions. The todcnd model runs slower in 4.2.1 than in 4.1.3, but it runs faster in 4.3.4 than in 4.1.3.

Table 2. "LongRun" Comparison of versions 4.1.3 and 4.3.4

```
Compare r3d413t.Tavg to r3d434t.Tavg
    * Left (1st) and right (2nd) files: Timer3d413t.Tavg, Timer3d434t.Tavg
    * Faster means right (2nd) time < left time
        Input model,  left time, right time,    diff.,  percent,   rate
       arteryFlex.p:  3.9904E+01  3.9396E+01     0.508     1.273  Faster
           hex2d1.p:  1.2098E-01  1.2557E-01    -0.005    -3.792 slower
      pois_cyl_he.p:  1.1434E+01  1.1299E+01     0.135     1.183  Faster
           sschf1.p:  5.2541E-02  4.0830E-02     0.012    22.289  Faster
           todcnd.p:  4.1683E-02  3.9199E-02     0.002     5.959  Faster
         typ12002.p:  1.3304E+01  1.3316E+01    -0.012    -0.087 slower
        typ1200n2.p:  1.5592E+01  1.4988E+01     0.604     3.875  Faster
Number faster / slower / same:   5   2   0
Average of percentage change:   4.386E+00
Suite Runtime: 1st, 2nd: 8.0449E+01, 7.9203E+01
Change in test suite runtime:    -1.246
Percentage change in test suite runtime:    -1.548
On average, Timer3d434t.Tavg is FASTER than Timer3d413t.Tavg
```

Figure 1 shows timings for the 4-series of RELAP5-3D versions after modification to allow them to build and run on the latest operating system. The average runtime decreases for some of the problems as does the cumulative time in the newer versions. Because of additional code and complexity from new code functionality an error corrections, longer run times were expected in newer versions of RELAP5-3D.

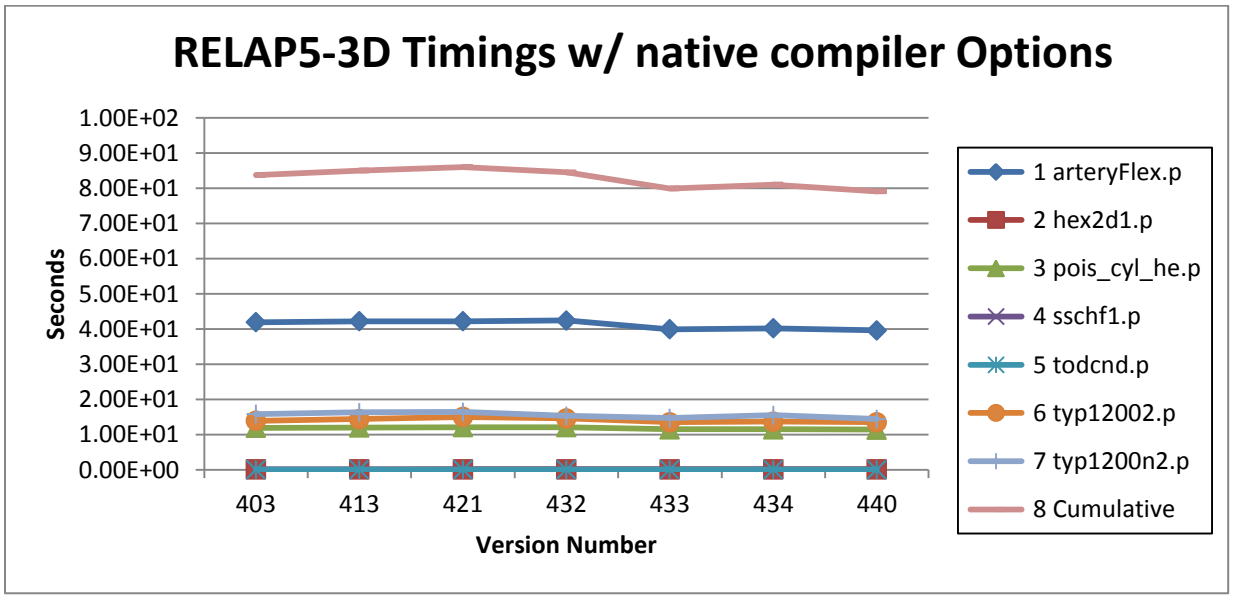

Figure 1. RELAP5-3D Timings with NATIVE compiler options

The timings shown in Figure 1 used the native compiler options, these are original options archived into the version when it was released. To get an unbiased set of timings, some new scripts imposed the complete set of options used in the most recent version of the code on all versions. They are then re-installed. Figure 2 shows timings for successive versions with the newest compiler options. These timings show the expected gradual trend to taking more time to run the same input models.

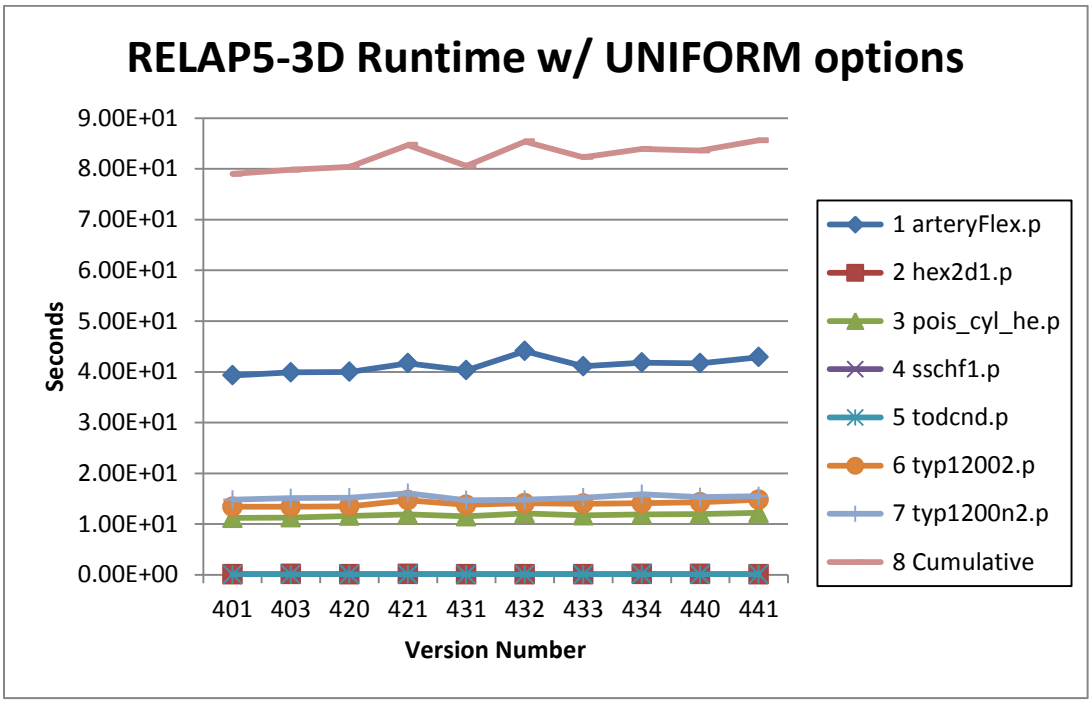

Figure 2. RELAP5-3D Timings with THE SAME compiler options

Conclusions

As seen in Figure 2, the of LongRun/ input models generally run slower for newer versions when, for all versions, the same compiler options are imposed at installation. This is slightly biased because the tests were selected from among standard installation problems that have shown slower performance on some of the newer versions.

The cumulative runtime, when compiler options and LongRun/ cases are used, has increased about 5% from Released Version 4.0.3 to Released Version 4.3.4 or from 2012 to 2017. Thus code runtime increase is about 1% per year. The increase from 4.0.3 to 4.4.1 is over 7%, so some adjustment, either to the coding or the compiler options, may be considered for speedup before releasing 4.4.2 to IRUG.

Future Work

With only a 7% increase in run time, there is little need to perform much runtime improvement for the 4-series of RELAP5-3D codes among themselves. However, it has been stated that there is significant runtime increase from Released Version 2.4.1 to Released Version 4.3.4. This has not been measured in a fair test with the same machine, operating system, and compiler options. Modifying 2.4.1 to make this possible then performing the timing comparison is the next agenda item.