COMPILERS AND PROBLEMS WITH CMAQ 5.1 TO 5.2.1

George Delic, Ph.D., HiPERiSM Consulting, LLC

Introduction

This presentation reports on problems encountered while implementing the parallel sparse matrix solver, FSparse [1], in the Chemistry Transport Model (CTM) in CMAQ. In this report issues encountered with both the original EPA [2], and FSparse, algorithms are listed for the GEAR version of the CTM.

Test bed environment

The hardware systems chosen were the platforms at HiPERiSM Consulting, LLC, consisting of two nodes with dual 16-core Intel E5v3 CPUs on each node. These are the base nodes of a heterogeneous cluster that includes an HP blade server hosting eight nodes with dual 4-core Intel E5640 CPUs. The total core count of this cluster is 128. The MPI executions are launched across multiple combination of these nodes. This cluster allows for comparison of the FSparse hybrid (MPI + OpenMP) parallel versions of CMAQ with the original EPA version (MPI only).

Compilers

Issues uncovered and reported here involved both the Intel Parallel Studio® suite (releases 17.2 and 17.6) and Portland Group compiler (release 18.1) with compiler options for a heterogeneous cluster. In the Portland case, utilization of an Intel wrapper enabled linking to the Intel MPI library. Tracking down issues was often a long and laborious process. Observations are presented as unordered lists taken from build or run logs for CMAQ 5.1, 5.2, and 5.2.1.

Critical compiler switches

Some observations on compiler flag choices to avoid runtime errors:

- use of -init=arrays,zero to initialize arrays to zero (CMAQ has uninitialized arrays)
- use of -axSSE4.2 to enable an excutable to run on a heterogeneous cluster
- use of -O2 to avoid runtime errors when using -O3
- use of -ip-no-inlining to avoid runtime errors due in part to using –ipo
- use of -warn all,nodeclarations,nounused (to avoid compilation termination)
- replacement of “ifdef parallel” with “ifdef parallel_mpi”

Mixed mode arithmetic

While some progress to fix this issue has been made in recent releases of the EPA version of CMAQ, mixed mode arithmetic continues to occur in CMAQ. This is due to use of single precision (SP) floating point operations (FP) throughout CMAQ except for the CTM where double precision (DP) is used. Thus, of necessity, SP variables are passed into the CTM and this leads to issues such as follows:

- Inconsistent conversion to DP that leads to random digits in the second half of FP words.
- Expressions that mix SP and DP variables (or constants).
- Constants that are not defined as DP (e.g. 1.0d0 and not 1.0)
- Constants that have limited precision even in SP

Where ever these issues were detected, they were corrected in the FSparse version of the CTM but they remain in the Original U.S. EPA release.

Fortran Lint summary analysis

While not all library interfaces are included (e.g. IOAPI, NETCDF, MPI) this is the summary statistics report for CMAQ 5.2.1 from Fortran Lint [3] listing potential problem areas of the source code.

```
FORTRAN-lint   Rev 6.02                11-Oct-2018  19:35:26   Page 1

Default options:  -u -O207,276,76,261 -Ttrim -Xno.unreferenced.parameters
                  -Xno.unused.common.variables,no.named.IEEE_*,no.named.C_*
                  -Xno.named.ISO_*,no.named.COMPILER_*
Command options:  -Isource-incl -e -w -f -g -s -t -Xtabular,line -x
                  -Ssource-f90 -Ecmaq521-f90.cfg

>>> Statistics:

 Number of source files:   571

 Source files:    36810 lines,   1415474 bytes     ( 35% comments, 65% code )
 Include files:   85100 lines,   3248370 bytes     ( 38% comments, 62% code )
 Total parsed:   121910 lines,   4663844 bytes     ( 37% comments, 63% code )

 Total subprograms:    758
   Subroutines:        164
   Functions:           61
   Program:              0
   Block Data:           0
   Module:             533

Individual message summary
Usage   FYI #128- 701x: local variable * declared but unused.
Usage   FYI #124- 196x: dummy argument * is unused.
Usage  WARN #743- 188x: module entity set but not referenced:  *, *
Usage   FYI #744- 153x: unused module entity:  *, *
Syntax  ERR #644-  96x: only POINTER and DIMENSION allowed in structure
                        component definition statements; attribute ignored.
Syntax  ERR #663-  95x: must be a pointer or an allocatable array; object
                        ignored.
Intrfc  ERR #130-  34x: missing subroutines:  *, *
Intrfc  FYI #132-  31x: unused subroutines:  *, *
Syntax  FYI #105-  26x: string will be truncated (from * to * chars).
Usage  WARN #134-  19x: common block members set but not referenced:  *, *
Intrfc  ERR #129-  18x: missing functions:  *, *
Usage   FYI #135-  15x: unused common block members:  *, *
Usage  WARN #740-  14x: INTENT(OUT) dummy arg * is never set.
Usage   ERR #742-  14x: module entity referenced but not set:  *, *
Intrfc  ERR #833-  14x: missing interface routine * (accessed via *).
Usage  WARN #127-  13x: local variable * is set but never referenced.
Intrfc  FYI #131-  11x: unused functions:  *, *
Syntax WARN #19-    9x: symbol name * is broken by white space.
Syntax  ERR #691-   9x: procedures in an interface with a generic name must be
                        all subroutines or all functions.
Syntax  ERR #18-    8x: illegal program name.
Syntax  FYI #138-   8x: unused labels:  *, *
Usage   ERR #126-   6x: local variable * is referenced but never set.
Usage   ERR #133-   6x: common block members referenced but not set:  *, *
Intrfc  FYI #279-   4x: * array passed to * array of smaller size (by *
                        bytes).
Intrfc WARN #572-   3x: dummy arg #* of * renamed (* in external subprogram
                        near * line *, and * in interface body near * line *).
Syntax  ERR #17-    1x: missing "(".
Syntax  ERR #20-    1x: extra characters following an otherwise valid
                        statement.
FORTRAN-lint    (statistics)                                    Page 2

Syntax  ERR #136-   1x: type declaration for symbol * missing.
Syntax  ERR #180-   1x: constant required here.
Intrfc WARN #189-   1x: no main program module present.
Intrfc  ERR #254-   1x: * array passed to a * dummy arg.

Total messages: 1697

                      Errors  Warnings   FYIs
                     -------- -------- --------
    Syntax:              212        9       34
    Global Interface:     67        4       46
    Data usage:           26      234     1065
```

Examples of uncovered issues

1. ifort bug: FALSE values produced with -O2 or -O3 for both FSparse or JSparse
2. No type declaration for SETLAM (in lat_lon.F) in ICON and BCON
3. RUNTIME: ERROR ABORT in subroutine READET in readet.f
4. compile time bug in the interface between these two files
   - /models/JPROC/jproc_table/srband.f
   - /models/JPROC/jproc_table/intavg.f
5. ipo: warning #11021: unresolved ncclos
6. lack of type definitions; many warnings of the type:
   - ./isrpia.inc(38): warning #6717: This name has not been given an explicit type PARAMETER (NCOMP=8,NIONS=10,NGASAQ=3,NSLDS=19,NPAIR=23,NZSR=100,
7. Global name too long
8. 12 messages of the type (compiler removed "se_' from name):
9. warning #5194: Source line truncated
10. 19 messages of the type :
    - LSM_MOD.F(143): warning #5194: Source line truncated
11. 8 messages of the type :
    - se_data_send_module.f(351): warning #6843: A dummy argument with an explicit INTENT(OUT) declaration is not given an explicit value. [REQUEST]
12. preprocessing of MPI parallelism in CMAQ: replace 89 occurrences of
    - set PAR = (-Dparallel), by
    - set PAR = (-Dparallel_mpi)
13. error in CCTM/src/emis/emis/LTNG_DEFN.FLTNG_DEFN.F(301): error #6404: This name does not have a type, and must have an explicit type. [SETLAM]
14. NaN's at runtime in ../src/aero/aero6/AEROSOL_CHEMISTRY.F
15. 8 messages of the type :
    - se_data_send_module.f(351): warning #6843: A dummy argument with an explicit INTENT(OUT) declaration is not given an explicit value. [REQUEST]

Conclusions

This report has described an analysis of CMAQ source code across three releases in the standard U.S. EPA model and shows problems in each case. While some problems are related to compiler versions (and can vary between them), most are in the source code itself. This suggest more careful code validation steps are appropriate using any available code analysis software.

References

[1] Delic, G., 2016: see presentation at the Annual CMAS meeting (http://www.cmasecenter.org).

[2] Jacobson, M. and Turco, R.P., (1994), Atmos. Environ. 28, 273-284.

[3] For a detailed list of issues found, or for information on compilers and Fortran Lint analysis software, contact george@hiperism.com