

Using Totalview and DDT with GEOS5

GMAO Brown Bag - 1/23/2014

Agenda

- What is the debugger for?
- Setting up GEOS5 for debugging.
- Go over starting Totalview and DDT, explain basic functionality.
- Live demonstration of DDT and Totalview.

What are DDT and Totalview?

- Interactive tools that allows you to debug serial, multi-threaded(OpenMP), and multi-processor (MPI) programs in fortran and c/c++.
- Much more powerful alternative to print statements.
- Totalview includes memscape (memory debugging utilities) and replay engine (step backward and forward in execution).
- Latest stable version of Totalview on discover 8.11.0.0
- Latest stable version of DDT on discover 4.2-34404
- Both programs only allowing a certain number of processes to be attached based on the licensing.
- DDT is still technically in procurement, although Alinea/NCCS has given us early access.

How Does One Typically Use the Debugger?

- Your code crashes or hangs, hopefully you have an idea where . . .
- Start your code in the debugger.
- Navigate to the routine you are interested in.
- Set a breakpoint or simply run to that point in the code
- Start examining variables, stepping through the code, etc ...

Running Totalview and DDT, generic code

- Load modules for you desired compiler/mpi stack
- Compile your code with -g and -O0
- Load Totalview module:
module load tool/tview-8.11.0.0
- Load DDT module:
module load tool/ddt-4.2-34404
- Launch Totalview from the command line:
totalview <path-to-your-executable>
- Launch DDT from the command line:
ddt <path-to-your-executable>

Prerequisites for Running with GEOS5 in the Debugger

- Compile GEOS5 with the BOPT=g option. This turns on traceback, debugging, and turns off compiler optimizations.
- On the command line:
 - \$ cd GEOSagcm/src
 - \$ source g5_modules
 - \$ make install BOPT=g (for parallel build: make -jn pinstall BOPT=g)
- Using the parallel build script in GEOSagcm/src
 - \$./parallel_build.csh -debug
- To run totalview you will need a set of interactive nodes
 - \$ xsub -l select=2:ncpus=12,walltime=1:00:00

Easy Way of Starting Totalview and GEOS5

- In your gcm_run.j script load the module before launching executable:
“module load tool/tview-8.11.0.0 “
- Replace “\$RUN_CMD \$NPES ./GEOSgcm.x” with “totalview ./GEOSgcm.x”
- Same thing for DDT: “ddt ./GEOSgcm.x”

```
module load tool/tview-8.11.0.0  
totalview ./GEOSgcm.x
```

Tips

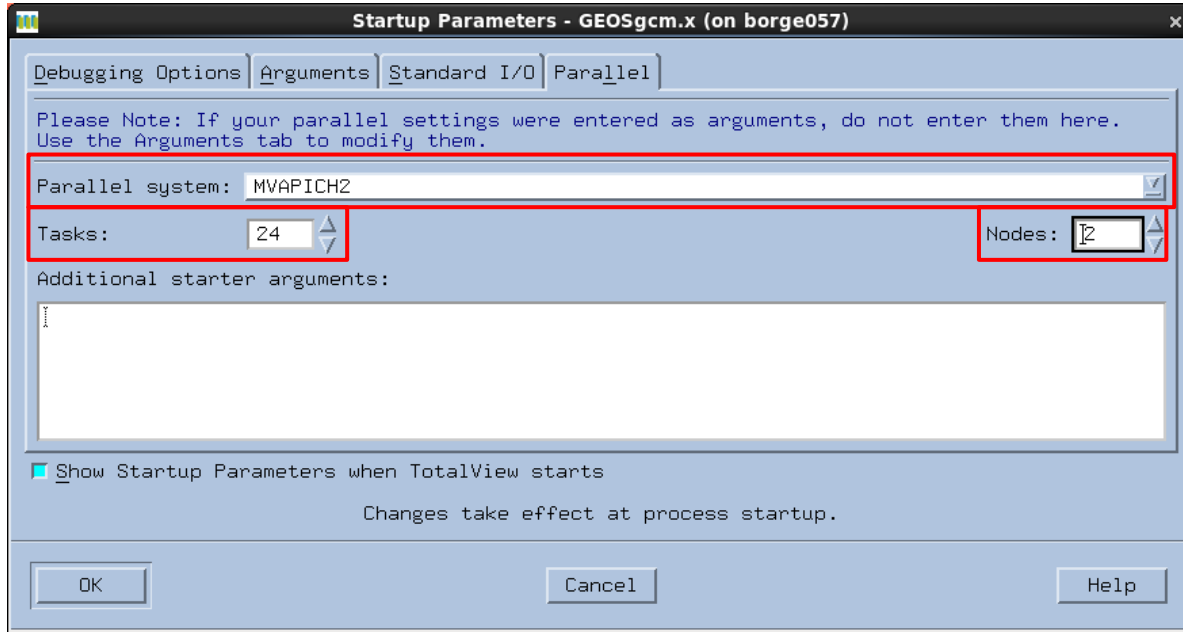
- Compile the model with BOPT=Og option (Ganymed-4_0 onward, turns on traceback and debugging) to get more info if it crashes.
- Only compile the part of the code you are interested with BOPT=g, rather than the whole model. For example if you wanted to debug something in the solar gridded component:
 - 1.) Compile model as normal
 - 2.) Go into the GEOSsolar_GridComp directory and do a “make clean” followed by a “make BOPT=g install”
 - 3.) Go to the applications directory and do “make install” to rebuilding GEOSgcm.x
- If you are running intel MPI you must also either add this to your script or run before launching totalview:

```
mpdboot -r ssh -n <num_nodes> -f $PBS_NODEFILE
```
- You can run in the debugger to at least find out what routine you are crashing in, even if you haven't compiled with debugging.

Totalview Introduction

Startup Window

The first thing you will see is a startup window:



Select your MPI stack, number of processes and the number of nodes

Process Window

Under view: Lookup
Function

Start the program

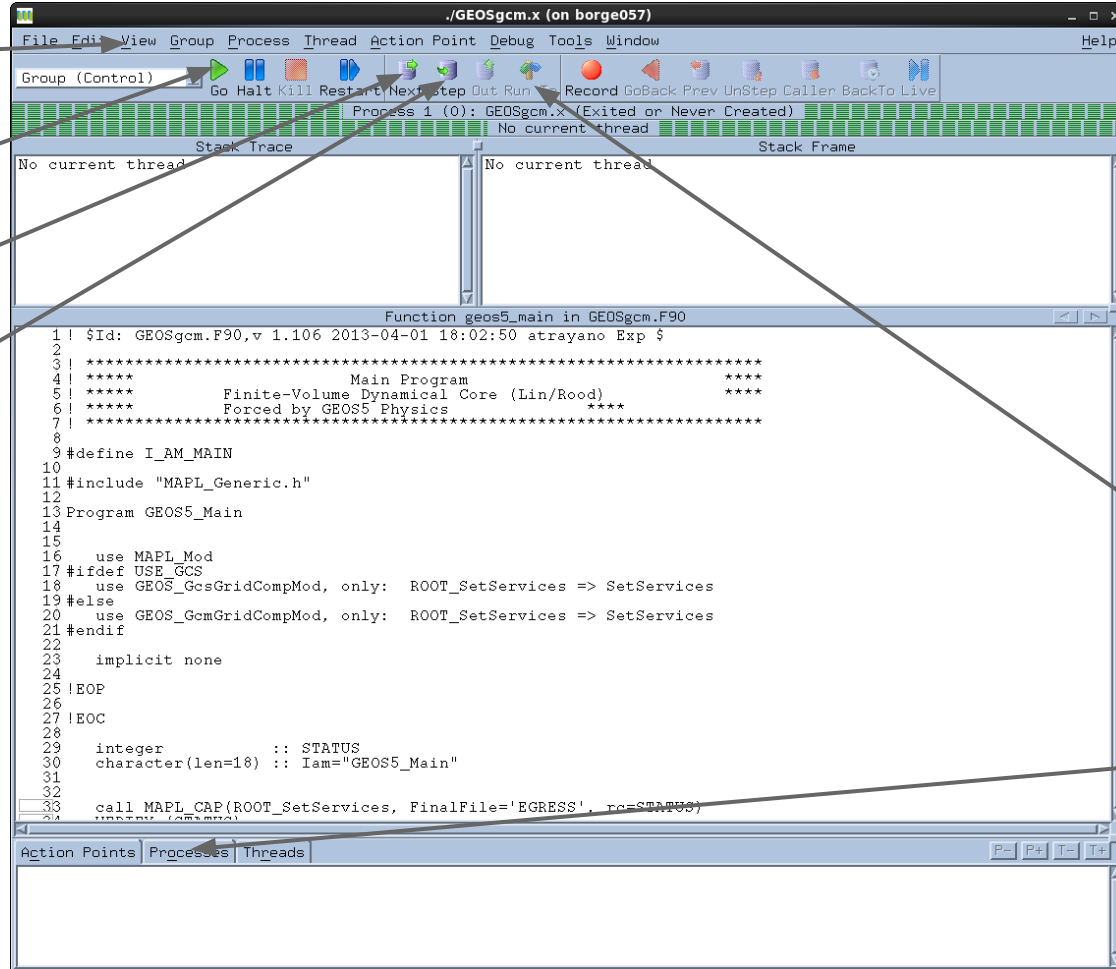
Go to next line

Step into subroutine

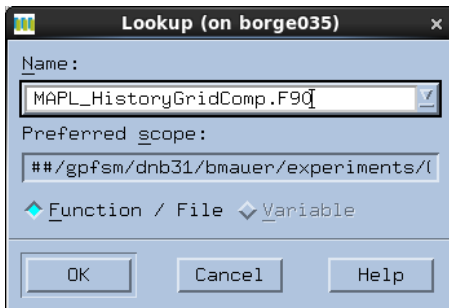
Provides detail
about a single
running process

Select a line
and run to
that point

Change what
process you
are looking at



Use the Lookup function to find the subroutine
or source file you are interested in:



What happens when you hit go:

GEOSgcm.x.73 (on borg01w201)

File Edit View Group Process Thread Action Point Debug Tools Window Help

Group (Control) Go Halt Kill Restart Next Step Out Run To Record GoBack Prev UnStep Caller BackTo Live

Rank 73: GEOSgcm.x.73 (At Breakpoint 1)

Thread 1 (47595009382144): GEOSgcm.x (At Breakpoint 1)

Stack Trace

Function "MAPL_HISTORYGRIDCOMPMOD" run:

gc: (type(esmf_gridcomp))
import: (type(esmf_state))
export: (type(esmf_state))
clock: (type(esmf_clock))
rc: -99999 (0xffff961)

Local variables:
collblock: -858993460 (0xffffffff)
n1: -858993460 (0xffffffff)
n2: -858993460 (0xffffffff)
n3: -858993460 (0xffffffff)
datestamp: -858993460 (0xffffffff)
nvmd: -858993460 (0xffffffff)

Function MAPL_HISTORYGRIDCOMPMOD run in MAPL_HistoryGridComp.F90

```
2379 character(len=ESMF_MAXSTR) :: DateStamp
2380 integer :: n1, n2, nn, CollBlock
2381
2382 character(len=ESMF_MAXSTR) :: IAm="HistoryRun"
2383 integer :: status
2384
2385 =====
2386 | Begin...
2387
2388 | Retrieve the pointer to the state
2389 |-----
2390 |
2391 | call ESMF_GridCompGetInternalState(gc, wrap, status)
2392 | VERIFY(status)
2393 | IntState => wrap%ptr
2394 |
2395 | the collections
2396 |-----
2397 |
2398 | list => IntState%list
2399 | nlist = size(list)
2400 |
2401 | CollBlock = IntState%blocksize
2402 | Retrieve the pointer to the generic state
2403 |-----
2404 |
2405 | call MAPL_GetObjectFromGC ( gc, GENSTATE, RC=STATUS)
2406 | VERIFY(STATUS)
2407 |
2408 | call MAPL_TimerOn(GENSTATE,"TOTAL")
2409 | call MAPL_TimerOn(GENSTATE,"Run")
2410 |
2411 | Couplers are done here for now
2412 |-----
2413 |
2414 | Perform arithmetic parser operations
2415 | do n=1,nlist
2416 | if ( Any(list(n)%ReWrite) ) then
2417 | call MAPL_RunExpression(IntState%CIM(n).list(n)%fields.list(n)%tmpfields, &
2418 | if ( .not. list(n)%disabled.and. IntState%average(n) ) then
2419 | call MAPL_RunExpression(IntState%CIM(n).list(n)%fields.list(n)%tmpfields, &
```

Action Points Processes Threads

P- P+ I- I+

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39

40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79

80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95

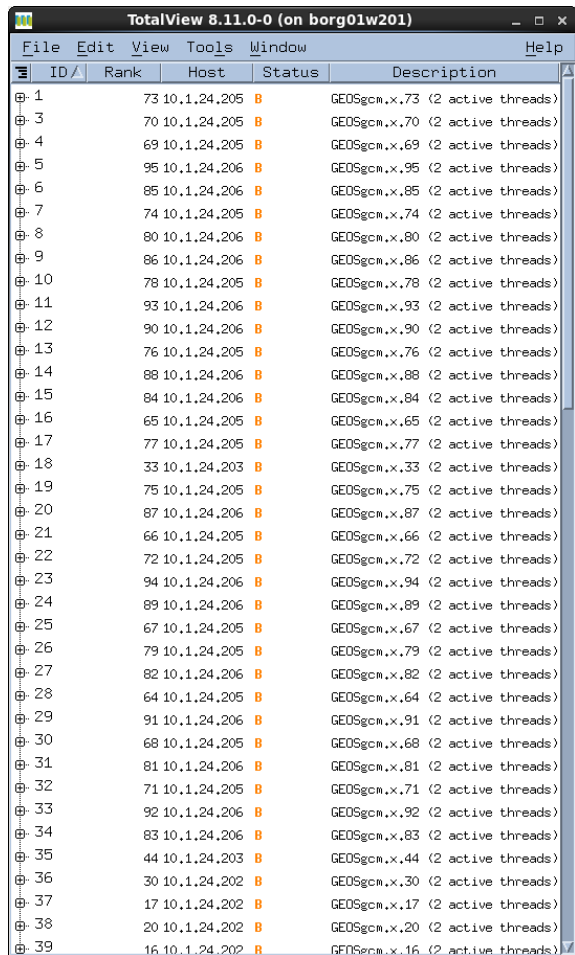
TotalView 8.11.0-0 (on borg01w201)

File Edit View Tools Window Help

ID/	Rank	Host	Status	Description
1	73	10.1.24.205	B	GEOSgcm.x.73 (2 active threads)
3	70	10.1.24.205	B	GEOSgcm.x.70 (2 active threads)
4	69	10.1.24.205	B	GEOSgcm.x.69 (2 active threads)
5	95	10.1.24.206	B	GEOSgcm.x.95 (2 active threads)
6	85	10.1.24.206	B	GEOSgcm.x.85 (2 active threads)
7	74	10.1.24.205	B	GEOSgcm.x.74 (2 active threads)
8	80	10.1.24.206	B	GEOSgcm.x.80 (2 active threads)
9	86	10.1.24.206	B	GEOSgcm.x.86 (2 active threads)
10	78	10.1.24.205	B	GEOSgcm.x.78 (2 active threads)
11	93	10.1.24.206	B	GEOSgcm.x.93 (2 active threads)
12	90	10.1.24.206	B	GEOSgcm.x.90 (2 active threads)
13	76	10.1.24.205	B	GEOSgcm.x.76 (2 active threads)
14	88	10.1.24.206	B	GEOSgcm.x.88 (2 active threads)
15	84	10.1.24.206	B	GEOSgcm.x.84 (2 active threads)
16	65	10.1.24.205	B	GEOSgcm.x.65 (2 active threads)
17	77	10.1.24.205	B	GEOSgcm.x.77 (2 active threads)
18	33	10.1.24.203	B	GEOSgcm.x.33 (2 active threads)
19	75	10.1.24.205	B	GEOSgcm.x.75 (2 active threads)
20	87	10.1.24.206	B	GEOSgcm.x.87 (2 active threads)
21	66	10.1.24.205	B	GEOSgcm.x.66 (2 active threads)
22	72	10.1.24.205	B	GEOSgcm.x.72 (2 active threads)
23	94	10.1.24.206	B	GEOSgcm.x.94 (2 active threads)
24	89	10.1.24.206	B	GEOSgcm.x.89 (2 active threads)
25	67	10.1.24.205	B	GEOSgcm.x.67 (2 active threads)
26	79	10.1.24.205	B	GEOSgcm.x.79 (2 active threads)
27	82	10.1.24.206	B	GEOSgcm.x.82 (2 active threads)
28	64	10.1.24.205	B	GEOSgcm.x.64 (2 active threads)
29	91	10.1.24.206	B	GEOSgcm.x.91 (2 active threads)
30	68	10.1.24.205	B	GEOSgcm.x.68 (2 active threads)
31	81	10.1.24.206	B	GEOSgcm.x.81 (2 active threads)
32	71	10.1.24.205	B	GEOSgcm.x.71 (2 active threads)
33	92	10.1.24.206	B	GEOSgcm.x.92 (2 active threads)
34	83	10.1.24.206	B	GEOSgcm.x.83 (2 active threads)
35	44	10.1.24.203	B	GEOSgcm.x.44 (2 active threads)
36	30	10.1.24.202	B	GEOSgcm.x.30 (2 active threads)
37	17	10.1.24.202	B	GEOSgcm.x.17 (2 active threads)
38	20	10.1.24.202	B	GEOSgcm.x.20 (2 active threads)
39	16	10.1.24.202	B	GEOSgcm.x.16 (2 active threads)

Root Window

Informs you of the state of all processes you are debugging



The screenshot shows the TotalView 8.11.0-0 Root Window. The window title is "TotalView 8.11.0-0 (on borg01w201)". The menu bar includes File, Edit, View, Tools, Window, and Help. The main area displays a table with the following columns: ID#, Rank, Host, Status, and Description. The table lists 39 entries, each representing a process or thread. The status column contains various letters (B, E, W, R, M, H) indicating the state of each entry. The description column provides details about the process or thread, including its name and the number of active threads.

ID#	Rank	Host	Status	Description
1	73	10.1.24.205	B	GEOSgcm.x.73 (2 active threads)
3	70	10.1.24.205	B	GEOSgcm.x.70 (2 active threads)
4	69	10.1.24.205	B	GEOSgcm.x.69 (2 active threads)
5	95	10.1.24.206	B	GEOSgcm.x.95 (2 active threads)
6	85	10.1.24.206	B	GEOSgcm.x.85 (2 active threads)
7	74	10.1.24.205	B	GEOSgcm.x.74 (2 active threads)
8	80	10.1.24.206	B	GEOSgcm.x.80 (2 active threads)
9	86	10.1.24.206	B	GEOSgcm.x.86 (2 active threads)
10	78	10.1.24.205	B	GEOSgcm.x.78 (2 active threads)
11	93	10.1.24.206	B	GEOSgcm.x.93 (2 active threads)
12	90	10.1.24.206	B	GEOSgcm.x.90 (2 active threads)
13	76	10.1.24.205	B	GEOSgcm.x.76 (2 active threads)
14	88	10.1.24.206	B	GEOSgcm.x.88 (2 active threads)
15	84	10.1.24.206	B	GEOSgcm.x.84 (2 active threads)
16	65	10.1.24.205	B	GEOSgcm.x.65 (2 active threads)
17	77	10.1.24.205	B	GEOSgcm.x.77 (2 active threads)
18	33	10.1.24.203	B	GEOSgcm.x.33 (2 active threads)
19	75	10.1.24.205	B	GEOSgcm.x.75 (2 active threads)
20	87	10.1.24.206	B	GEOSgcm.x.87 (2 active threads)
21	66	10.1.24.205	B	GEOSgcm.x.66 (2 active threads)
22	72	10.1.24.205	B	GEOSgcm.x.72 (2 active threads)
23	94	10.1.24.206	B	GEOSgcm.x.94 (2 active threads)
24	89	10.1.24.206	B	GEOSgcm.x.89 (2 active threads)
25	67	10.1.24.205	B	GEOSgcm.x.67 (2 active threads)
26	79	10.1.24.205	B	GEOSgcm.x.79 (2 active threads)
27	82	10.1.24.206	B	GEOSgcm.x.82 (2 active threads)
28	64	10.1.24.205	B	GEOSgcm.x.64 (2 active threads)
29	91	10.1.24.206	B	GEOSgcm.x.91 (2 active threads)
30	68	10.1.24.205	B	GEOSgcm.x.68 (2 active threads)
31	81	10.1.24.206	B	GEOSgcm.x.81 (2 active threads)
32	71	10.1.24.205	B	GEOSgcm.x.71 (2 active threads)
33	92	10.1.24.206	B	GEOSgcm.x.92 (2 active threads)
34	83	10.1.24.206	B	GEOSgcm.x.83 (2 active threads)
35	44	10.1.24.203	B	GEOSgcm.x.44 (2 active threads)
36	30	10.1.24.202	B	GEOSgcm.x.30 (2 active threads)
37	17	10.1.24.202	B	GEOSgcm.x.17 (2 active threads)
38	20	10.1.24.202	B	GEOSgcm.x.20 (2 active threads)
39	16	10.1.24.202	R	GEOSgcm.x.16 (2 active threads)

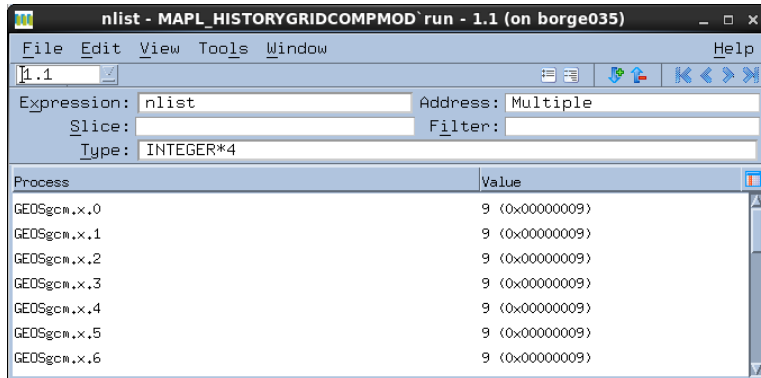
- T = stopped
- B = Breakpoint
- E = Error
- W = Watchpoint
- R = Running
- M = Mixed
- H = Held

Diving

```
Stack Frame
nl: -858993460 {0xffffffff}
datestamp: -858993460 {0xffffffff}
nymd: -858993460 {0xffffffff}
newseg: .false. (-858993460)
intmpl: -858993460 {0xffffffff}
nhms: -858993460 {0xffffffff}
n: 10 {0x0000000a}
m: -858993460 {0xffffffff}
nlist: 9 {0x00000009}
genstate: 0x15990e40 -> (type(mapl_metacomp))
intstate: 0x15997c30 -> (type(history_state))
wrap: (type(history_wrap))
lam: historyRun
state_out: (type(esmf_state))
status: 0 {0x00000000}
writing: (LOGICAL*4,allocatable::(:))
filename: <Bad address: 0xffffffff>
```

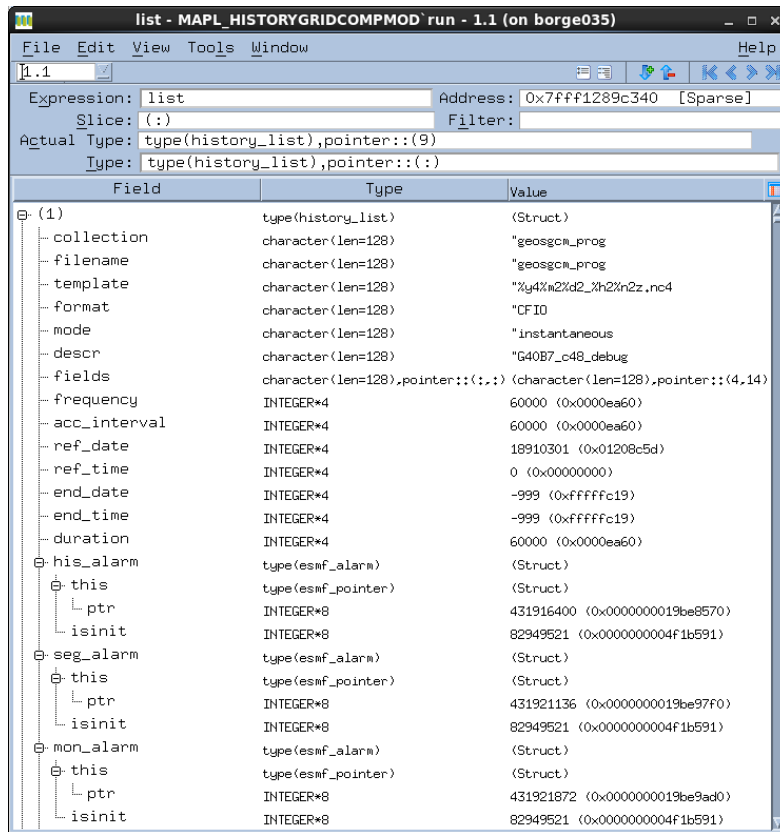
- You can “dive” on several items such as variables and subroutines by right clicking
- If you click a variable you can dive across all processes
- You can dive into a subroutine to see it’s source code

Value of nlist on each process:



Process	Value
GEOSgcm.x.0	9 (0x00000009)
GEOSgcm.x.1	9 (0x00000009)
GEOSgcm.x.2	9 (0x00000009)
GEOSgcm.x.3	9 (0x00000009)
GEOSgcm.x.4	9 (0x00000009)
GEOSgcm.x.5	9 (0x00000009)
GEOSgcm.x.6	9 (0x00000009)

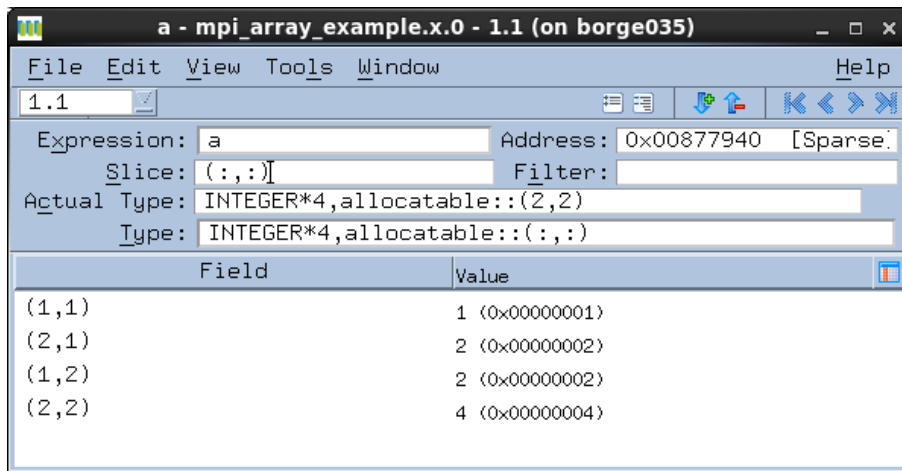
You can dive into more complex structures as well::



The screenshot shows a window titled "list - MAPL_HISTORYGRIDCOMP' run - 1.1 (on borge035)". The window has a menu bar (File, Edit, View, Tools, Window, Help) and a toolbar. Below the toolbar, there are input fields for "Expression: list", "Address: 0x7ffff1289c340 [Sparse]", "Slice: (:)", and "Filter:". Below these, it shows "Actual Type: type(history_list),pointer::(9)" and "Type: type(history_list),pointer::(:)".

Field	Type	Value
⊖ (1)	type(history_list)	(Struct)
collection	character(len=128)	"geosgcm_prog
filename	character(len=128)	"geosgcm_prog
template	character(len=128)	"%y4%n2%d2_%h2%n2z.nc4
format	character(len=128)	"CFIO
mode	character(len=128)	"instantaneous
descr	character(len=128)	"G40B7_c48_debug
fields	character(len=128),pointer::(:,:) (character(len=128),pointer::(4,14)	
frequency	INTEGER*4	60000 (0x0000ea60)
acc_interval	INTEGER*4	60000 (0x0000ea60)
ref_date	INTEGER*4	18910301 (0x01208c5d)
ref_time	INTEGER*4	0 (0x00000000)
end_date	INTEGER*4	-999 (0xfffffc19)
end_time	INTEGER*4	-999 (0xfffffc19)
duration	INTEGER*4	60000 (0x0000ea60)
⊖ his_alarm	type(esmf_alarm)	(Struct)
⊖ this	type(esmf_pointer)	(Struct)
ptr	INTEGER*8	431916400 (0x0000000019be8570)
isinit	INTEGER*8	82949521 (0x00000000004f1b591)
⊖ seg_alarm	type(esmf_alarm)	(Struct)
⊖ this	type(esmf_pointer)	(Struct)
ptr	INTEGER*8	431921136 (0x0000000019be97f0)
isinit	INTEGER*8	82949521 (0x00000000004f1b591)
⊖ mon_alarm	type(esmf_alarm)	(Struct)
⊖ this	type(esmf_pointer)	(Struct)
ptr	INTEGER*8	431921872 (0x0000000019be9ad0)
isinit	INTEGER*8	82949521 (0x00000000004f1b591)

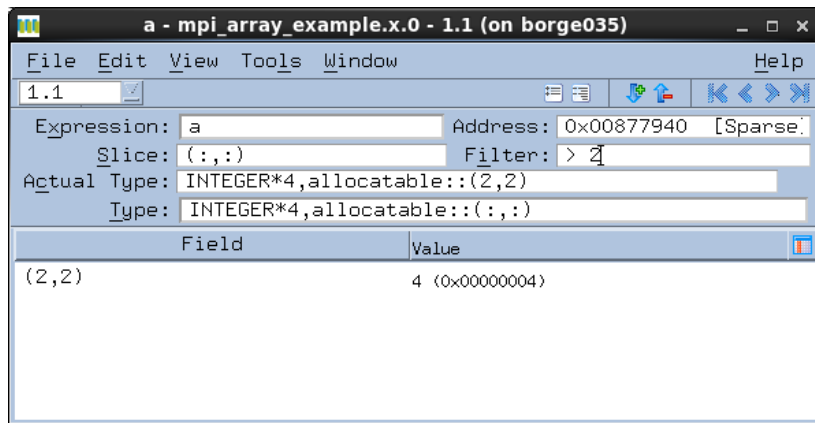
Slicing and Visualizing Arrays



1.1

Expression: `a` Address: `0x00877940` [Sparse:
Slice: `(:,:,)]` Filter:
Actual Type: `INTEGER*4,allocatable::(2,2)`
Type: `INTEGER*4,allocatable::(:,:)`

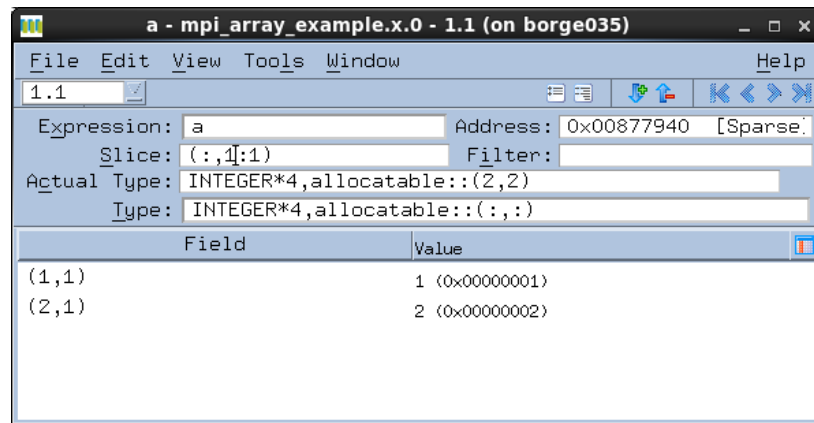
Field	Value
(1,1)	1 (0x00000001)
(2,1)	2 (0x00000002)
(1,2)	2 (0x00000002)
(2,2)	4 (0x00000004)



1.1

Expression: `a` Address: `0x00877940` [Sparse:
Slice: `(:,:,)` Filter: `> 2`
Actual Type: `INTEGER*4,allocatable::(2,2)`
Type: `INTEGER*4,allocatable::(:,:)`

Field	Value
(2,2)	4 (0x00000004)



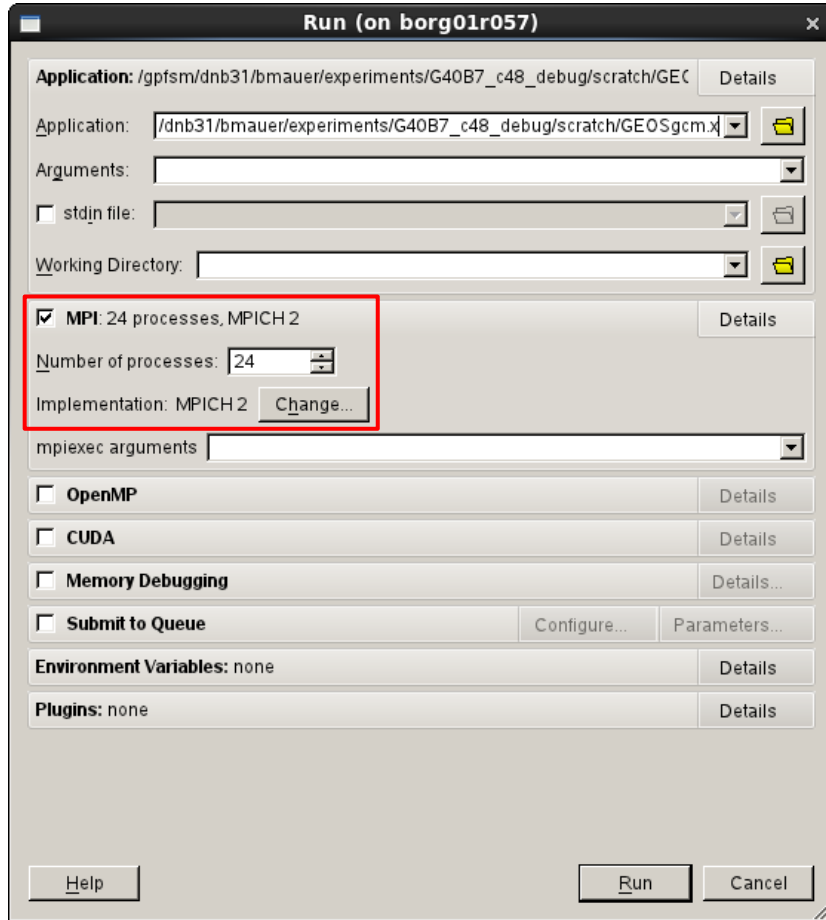
1.1

Expression: `a` Address: `0x00877940` [Sparse:
Slice: `(:,1:1)` Filter:
Actual Type: `INTEGER*4,allocatable::(2,2)`
Type: `INTEGER*4,allocatable::(:,:)`

Field	Value
(1,1)	1 (0x00000001)
(2,1)	2 (0x00000002)

DDT Introduction

Startup Window



The screenshot shows a 'Run' dialog window titled 'Run (on borg01r057)'. It contains several sections for configuring the application execution. The 'MPI' section is highlighted with a red box and contains the following options:

- ☒ **MPI: 24 processes, MPICH 2** (with a 'Details' button)
- Number of processes:** 24 (with a spin box)
- Implementation:** MPICH 2 (with a 'Change...' button)
- mpiexec arguments:** (with a text field)

Below the MPI section, there are several other options, each with a 'Details' button:

- ☐ **OpenMP**
- ☐ **CUDA**
- ☐ **Memory Debugging**
- ☐ **Submit to Queue** (with 'Configure...' and 'Parameters...' buttons)

At the bottom, there are two more sections:

- Environment Variables:** none (with a 'Details' button)
- Plugins:** none (with a 'Details' button)

At the very bottom, there are three buttons: 'Help', 'Run', and 'Cancel'.

Like Totalview you need to choose your MPI stack, and number of processes:

Control Buttons

List of
processes

Source File
list

Allinea DDT 4.2-34404 (on borg01r057)

File View Control Search Tools Window Help

Current Group: All | Focus on current: Group Process Thread | Step Threads Together

96 processes (0-95) Paused: 96 Playing: 0 Finished: 0
Show processes | Currently selected: 0 (on borg01r057, pid 27252, main thread IWP 27252)

Create Group

Project Files Fortran Modules

Project Files

Search (Ctrl+F)

- m_set_NLNDHAR.F90
- m_String.F90
- m_StrTemplate.F90
- MAM_BaseMod.F90
- MAM_CoagulationMod.F90
- MAM_DryRemovalMod.F90
- MAM_DustMod.F90
- MAM_NucleationMod.F90
- MAM_SeasaltMod.F90
- MAM_SizeMod.F90
- MAMchem_GridCompMod.F
- MAML_CoagulationMod.F90
- MAML_DryRemovalMod.F90
- MAML_NucleationMod.F90
- MAML_SettingMod.F90
- MAML_SizeMod.F90
- MAML_Base.F90
- MAPL_Cap.F90**
- CAP_FINALIZE
- ESMFL_ClockSet
- MAPL_CAP
- MAPL_ClockInit
- MAPL_ConfigSetInt4
- MAPL_ConfigSetString
- MAPL_PackDateTime
- MAPL_UnpackDateTime
- Perpetual_Clock

MAPL_Cap.F90

```
1 ! $Id: MAPL_Cap.F90,v 1.37.12.5 2013-12-18 17:01:17 atrayano Exp $
2
3 #include "MAPL_Generic.h"
4
5 module MAPL_CapMod
6
7 !BOP
8
9 ! MODULE: MAPL_CapMod --- Implements the top entry point for MAPL components
10
11 ! !USES:
12
13 use ESMF
14 use MAPL_BaseMod
15 use MAPL_ConstantsMod
16 use MAPL_ProfMod
17 use MAPL_MemUtilsMod
18 use MAPL_IOMod
19 use MAPL_CommsMod
20 use MAPL_GenericMod
21 use MAPL_LocStreamMod
22 use ESMFL_Mod
23 use MAPL_ShmemMod
24 use MAPL_HistoryGridCompMod, only : Hist_SetServices => SetServices
25 use MAPL_HistoryGridCompMod, only : HISTORY_ExchangeListWrap
26 use MAPL_ExtDataGridCompMod, only : ExtData_SetServices => SetServices
27
28 implicit none
```

Locals Current Line(s) Current Stack

Current Line(s)

Variable Name	Value
rc	-858993460
status	18
vm	

Input/Output Breakpoints Watchpoints Stacks Tracepoints Tracepoint Output Logbook

Input/Output

Output For Process: 0

```
#run.alurm: cluster configuration lacks support for cpu binding
```

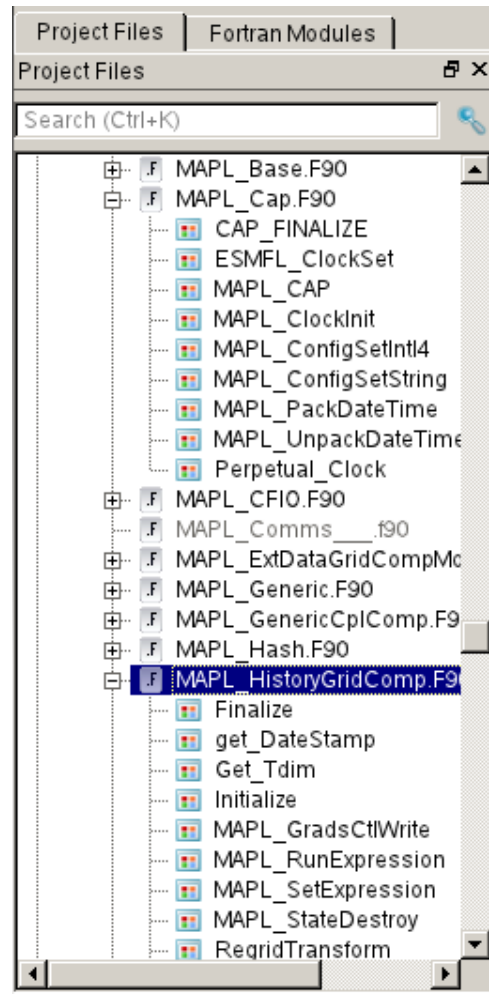
Type here (Enter to send):

More

Ready

Variables
Pane

Unlike Totalview you have a list of all source files and subroutines contained in them. No more having to type the name of the file or routine you want to examine!



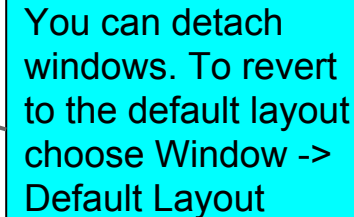
Like Totalview you have several key buttons for controlling program execution:



Step into
subroutine.

Go to next
line

Step out of
subroutine.



You can detach windows. To revert to the default layout choose Window -> Default Layout

Advanced Things You Can Do

- Attached only certain processes to your Totalview session. This could be useful if there are not enough totalview licenses available for your problem.
- Attach an already running program to Totalview, for example if your program is hanging and you want to find out where.
- Memory debugging with memscape in Totalview.
- The Totalview replay engine allowed you to step forward and backward in execution (memory intensive).
- Test code changes on the fly in Totalview
- DDT has an integrated profiler, MAP.