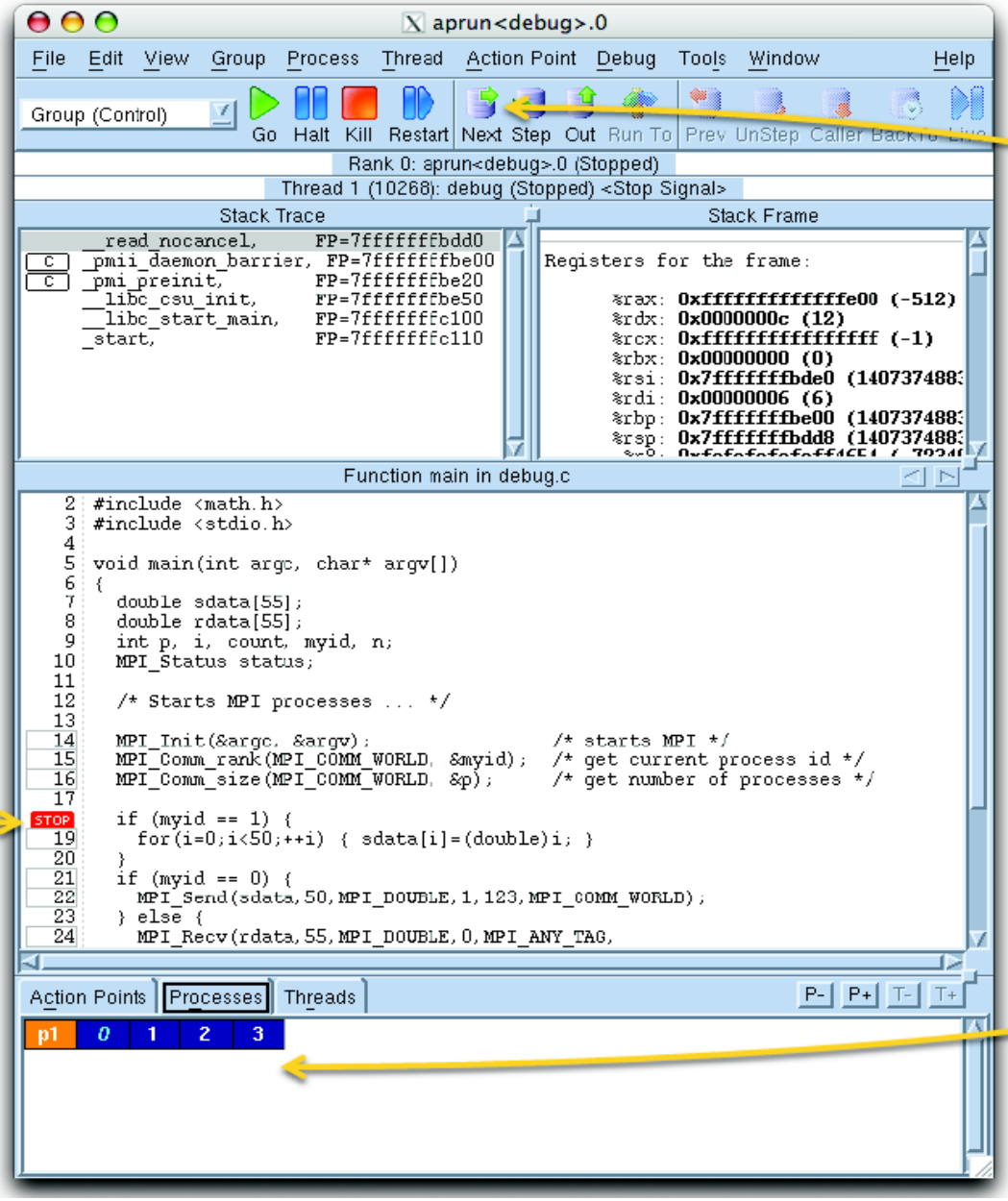
Debugging

Demos	OpenMP	MPI	OMP/MPI	Reference
gdb	yes	yes	no	http://www.gnu.org/s/gdb/
compiler	yes	yes	yes	-g -traceback
nemiver	no	yes	no	http://projects.gnome.org/nemiver/
eclipse	yes	no	no	http://www.eclipse.org
DDT	yes	yes	yes	http://www.allinea.com/ddt
Totalview	yes	yes	yes	http://www.roguewave.com

- A debugger can help find bugs in :
  - Fortran, C, C++, ...
  - Serial, MPI, OpenMP, MPI/OpenMP.
- A debugger can be executed :
  - either as a graphical user interface,
  - Or from a command-line interface
  - http://www.open-mpi.org/faq/?category=debugging

Breakpoints



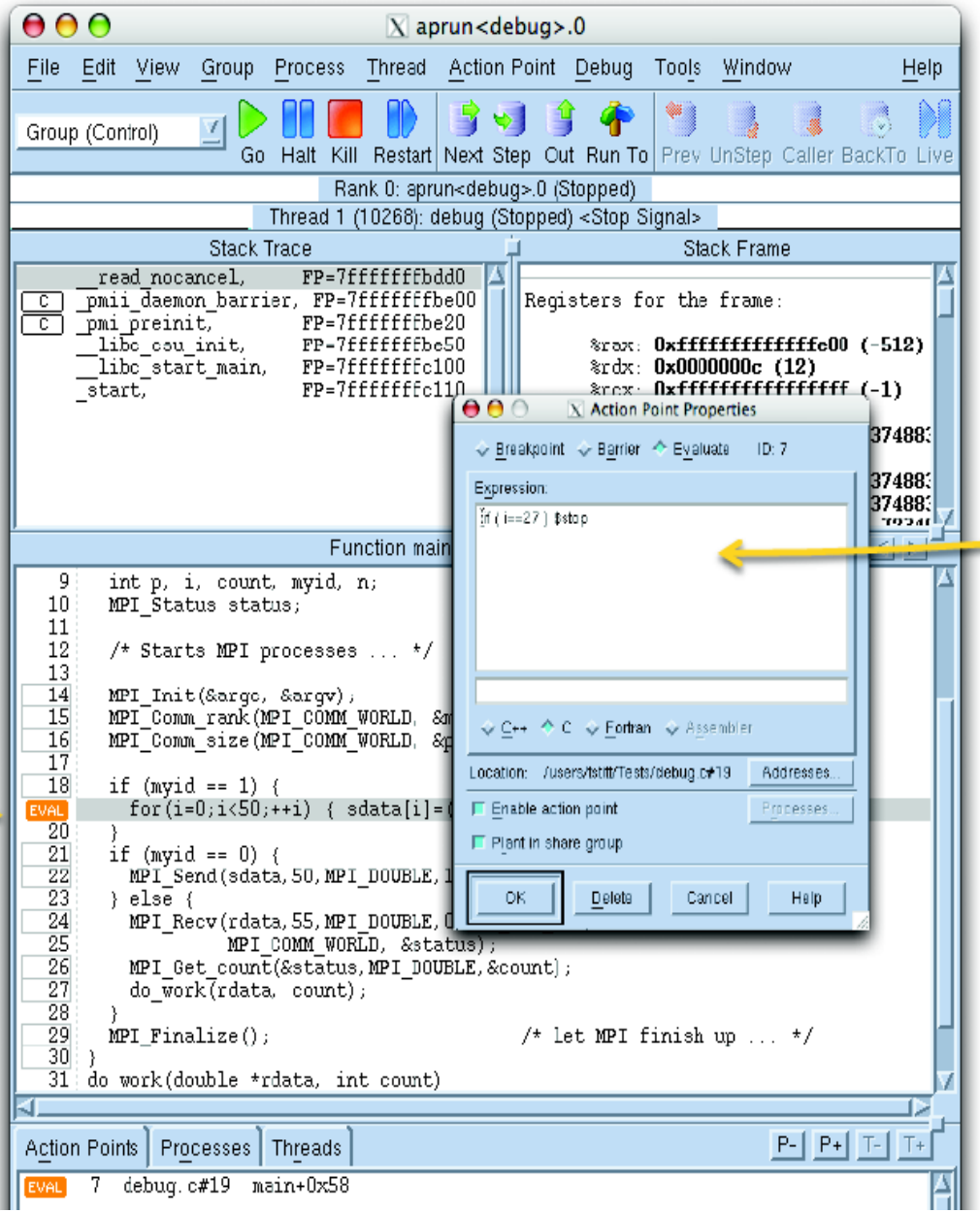
Execution Toolbars

Set Breakpoint

Co-operating Processes

TOTALVIEW TECHNOLOGIES

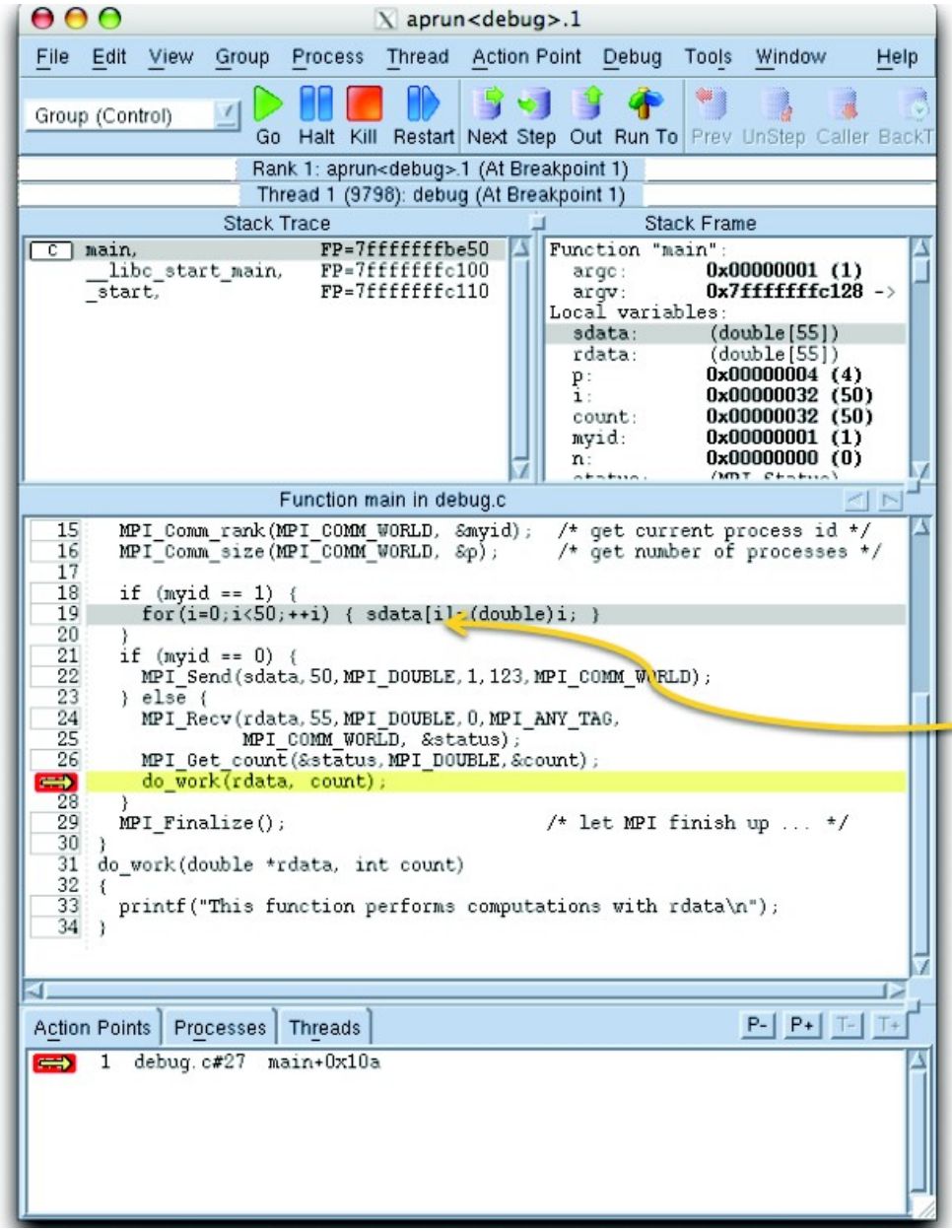
Action points



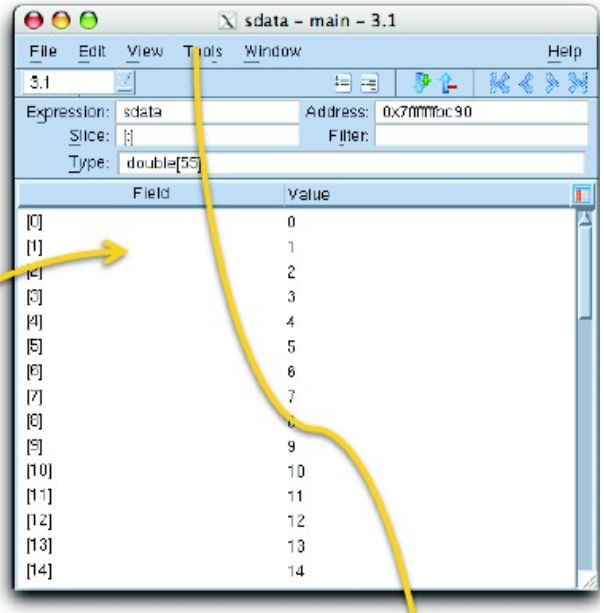
Set Evaluation Point

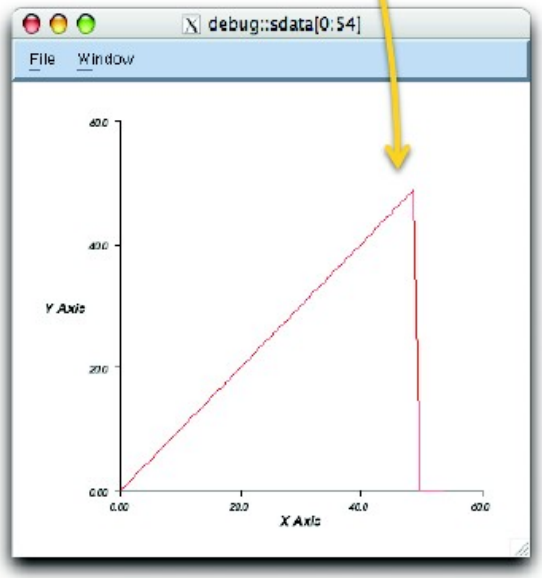
Enter Stopping Condition

Examining data

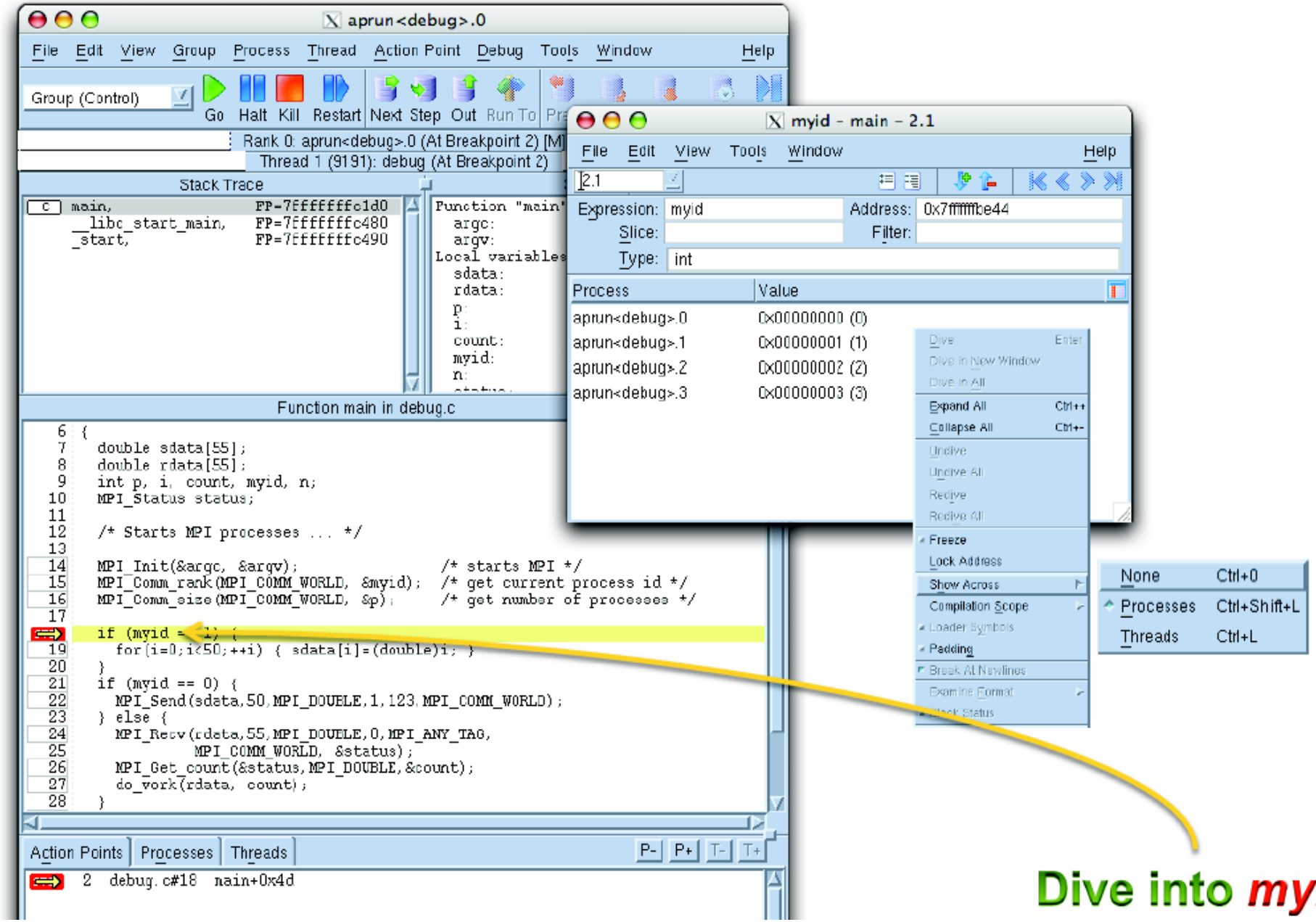


Dive in And Visualize

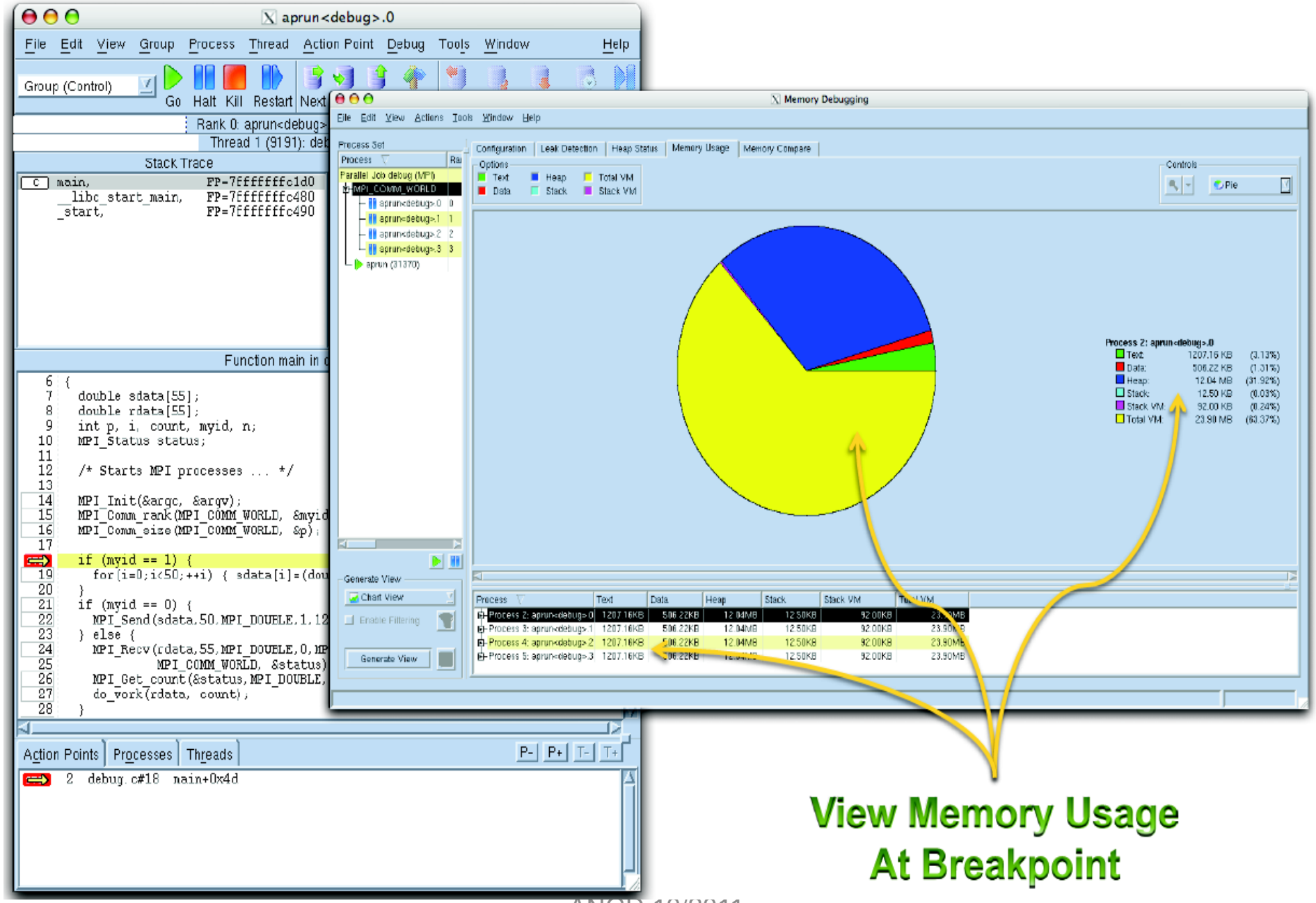




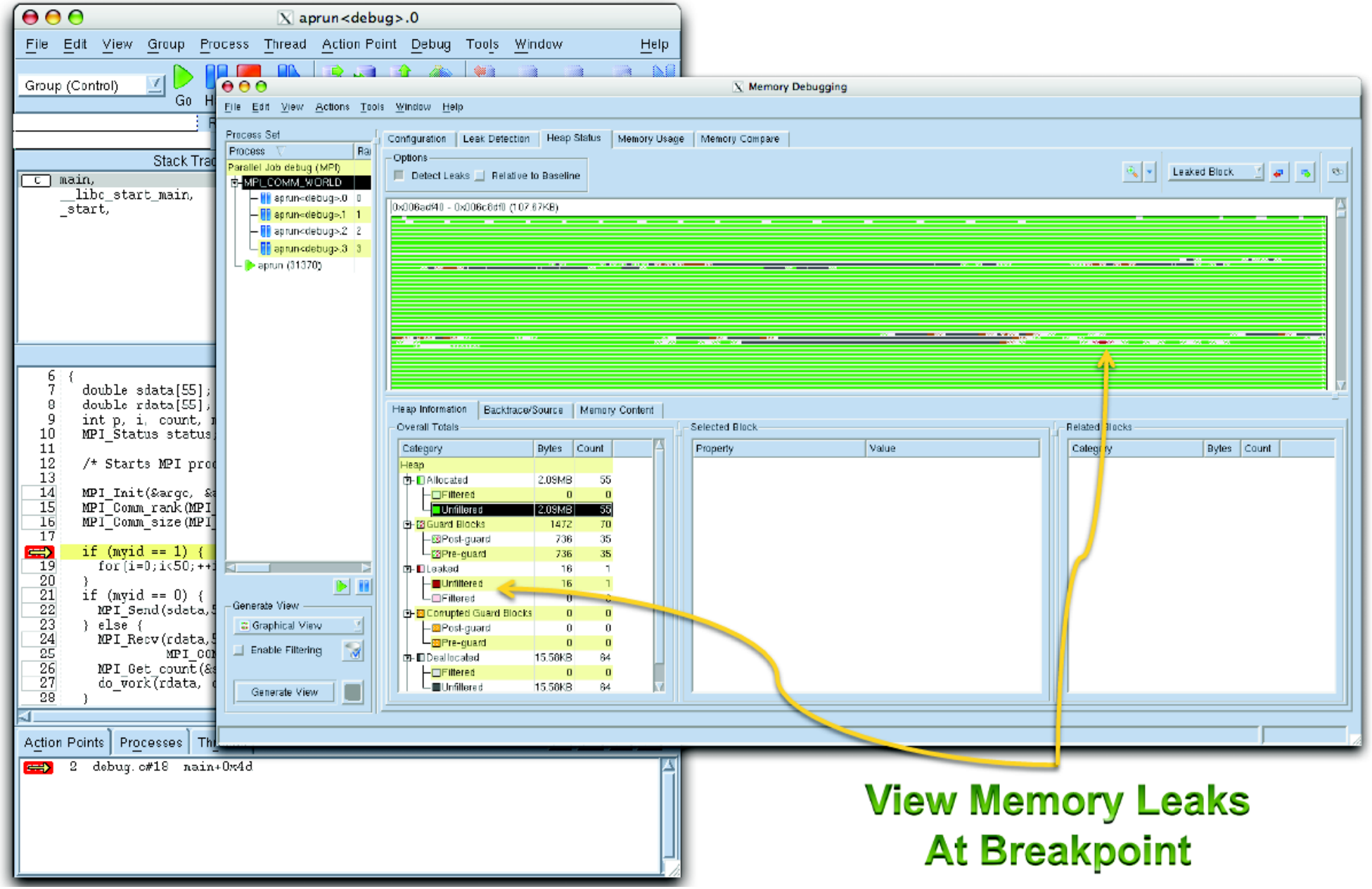
Viewing data across processes



Memory usage



Memory leaks



View Memory Leaks At Breakpoint

Getting started (1)

```
 2 program who
 3 #ifdef _MPI
 4         use mpi
 5 #endif
 6 !$       use omp_lib
 7         implicit none
 8         integer :: rank=-1,nb_procs=-1,code=-1,thread=-1,threads=-1
 9         integer ::  namelen=-1, coreid=-1
10         character(len=2) :: processor_name=""
11         integer, external :: running_on
12
13 #ifdef _MPI
14         call MPI_INIT (code)
15         call MPI_COMM_SIZE ( MPI_COMM_WORLD ,nb_procs,code)
16         call MPI_COMM_RANK ( MPI_COMM_WORLD ,rank,code)
17         call MPI_Get_processor_name( processor_name, namelen, code )
18 #endif
19
20 !$omp parallel private(thread)
21 !$       thread = omp_get_thread_num()
22 !$       threads = omp_get_num_threads()
23         coreid = running_on()
24         write (*,'(a17,i4,a2,i4,a5,i4,a2,i4,1x,a2,i4)') &
25                "hello in f90 rnk=",&
26                rank," /",nb_procs,&
27                " thd=",thread," /",threads, &
28                processor_name, &
29                coreid
30 !$omp end parallel
31
32
33 #ifdef _MPI
34         call MPI_FINALIZE (code)
35 #endif
36
37 end program who
```

```
mpif90 -D_MPI -fopenmp -g mpiomp.F90 -L/softs/affinity -laff -o f
```

```
n1:/home/piccinali/trunk/debug/intro/f90 $  sbatch.sh
USAGE :
                arg1=exe
                arg2=mppwidth
                arg3=mppnppn
                arg4=mppdepth
                arg5=exeargs
                arg6=prempiexec
                arg7=postmpiexec
```

```
n1:/home/piccinali/trunk/debug/intro/f90 $  export OMP_NUM_THREADS=3 ; ~/sbatch.sh ./f 2 2 3 "" "" -bind-to-core
+ export OMP_NUM_THREADS=3
+ OMP_NUM_THREADS=3
+ /usr/bin/time -p /softs/openmpi-1.4.3/bin/mpiexec -bind-to-core -n 2 -npernode 2 -x OMP_NUM_THREADS -hostfile /softs/openmpi-1.4.3/h ./f
```

```
n1:/home/piccinali/trunk/debug/intro/f90 $  cat o_f.0006.2.2.3.-bind-to-core
hello in f90 rnk=   1 /   2 thd=   0 /   3 n1   1
hello in f90 rnk=   1 /   2 thd=   2 /   3 n1   1
hello in f90 rnk=   1 /   2 thd=   1 /   3 n1   1
hello in f90 rnk=   0 /   2 thd=   0 /   3 n1   0
hello in f90 rnk=   0 /   2 thd=   2 /   3 n1   0
hello in f90 rnk=   0 /   2 thd=   1 /   3 n1   0
real 0.04
```

Getting started (2)

```
n1:/home/piccinali/trunk/debug/intro/f90 $  more o_f.0006.2.2.3.-bind-to-none o_f.0006.2.2.3.-bind-to-core o_f.0006.2.2.3.-bind-to-socket

::::::::::::::
o_f.0006.2.2.3.-bind-to-none
::::::::::::::
hello in f90 rnk=   1 /   2 thd=   0 /   3 n1  11
hello in f90 rnk=   0 /   2 thd=   0 /   3 n1  11
hello in f90 rnk=   1 /   2 thd=   2 /   3 n1  11
hello in f90 rnk=   1 /   2 thd=   1 /   3 n1  11
hello in f90 rnk=   0 /   2 thd=   2 /   3 n1  11
hello in f90 rnk=   0 /   2 thd=   1 /   3 n1  11
real 1.06
user 0.06
sys 0.04
::::::::::::::
o_f.0006.2.2.3.-bind-to-core
::::::::::::::
hello in f90 rnk=   1 /   2 thd=   0 /   3 n1   1
hello in f90 rnk=   1 /   2 thd=   2 /   3 n1   1
hello in f90 rnk=   1 /   2 thd=   1 /   3 n1   1
hello in f90 rnk=   0 /   2 thd=   0 /   3 n1   0
hello in f90 rnk=   0 /   2 thd=   2 /   3 n1   0
hello in f90 rnk=   0 /   2 thd=   1 /   3 n1   0
real 0.04
user 0.02
sys 0.02
::::::::::::::
o_f.0006.2.2.3.-bind-to-socket
::::::::::::::
hello in f90 rnk=   0 /   2 thd=   2 /   3 n1   5
hello in f90 rnk=   0 /   2 thd=   1 /   3 n1   5
hello in f90 rnk=   0 /   2 thd=   0 /   3 n1   5
hello in f90 rnk=   1 /   2 thd=   2 /   3 n1   5
hello in f90 rnk=   1 /   2 thd=   0 /   3 n1   5
hello in f90 rnk=   1 /   2 thd=   1 /   3 n1   5
real 0.05
user 0.03
sys 0.05
```

```
n1:/home/piccinali/trunk/debug/intro/f90 $  more o_f.0006.2.1.3.-bind-to-none o_f.0006.2.1.3.-bind-to-core o_f.0006.2.1.3.-bind-to-socket
::::::::::::::
o_f.0006.2.1.3.-bind-to-none
::::::::::::::
hello in f90 rnk=   0 /   2 thd=   0 /   3 n1  11
hello in f90 rnk=   0 /   2 thd=   2 /   3 n1  11
hello in f90 rnk=   0 /   2 thd=   1 /   3 n1  11
hello in f90 rnk=   1 /   2 thd=   0 /   3 n2  11
hello in f90 rnk=   1 /   2 thd=   2 /   3 n2  11
hello in f90 rnk=   1 /   2 thd=   1 /   3 n2  11
real 0.36
user 0.04
sys 0.03
::::::::::::::
o_f.0006.2.1.3.-bind-to-core
::::::::::::::
hello in f90 rnk=   0 /   2 thd=   0 /   3 n1   0
hello in f90 rnk=   0 /   2 thd=   2 /   3 n1   0
hello in f90 rnk=   0 /   2 thd=   1 /   3 n1   0
hello in f90 rnk=   1 /   2 thd=   0 /   3 n2   0
hello in f90 rnk=   1 /   2 thd=   2 /   3 n2   0
hello in f90 rnk=   1 /   2 thd=   1 /   3 n2   0
real 0.35
user 0.02
sys 0.04
::::::::::::::
o_f.0006.2.1.3.-bind-to-socket
::::::::::::::
hello in f90 rnk=   0 /   2 thd=   0 /   3 n1   5
hello in f90 rnk=   0 /   2 thd=   2 /   3 n1   5
hello in f90 rnk=   0 /   2 thd=   1 /   3 n1   5
hello in f90 rnk=   1 /   2 thd=   0 /   3 n2   5
hello in f90 rnk=   1 /   2 thd=   2 /   3 n2   5
hello in f90 rnk=   1 /   2 thd=   1 /   3 n2   5
real 0.39
user 0.05
```

Demo : GDB

Frequently used GDB commands

General Commands

- help [name] : Show information about GDB command
- run [<args>] : runs selected program with arguments <args>
- attach <pid> : attach gdb to a running process
- Kill : kills the process being debugged
- Quit : quits the gdb program

Stepping and Continuing

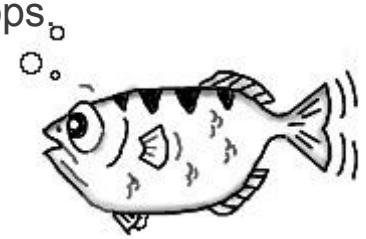
- c[ontinue] : continue execution (after a stop)
- s[tep] : step one line, entering called functions
- n[ext] : step one line, without entering functions

Breakpoint commands

- b[reak] [<where>] : sets breakpoints. <where> can be a function name, a line number or a hex address
- [r]watch <expr> : sets a watchpoint, which will break
- when <expr> is written to [or read]
- info break[points] : prints out a listing of all breakpoints
- d[elete] [<nums>] : deletes breakpoints

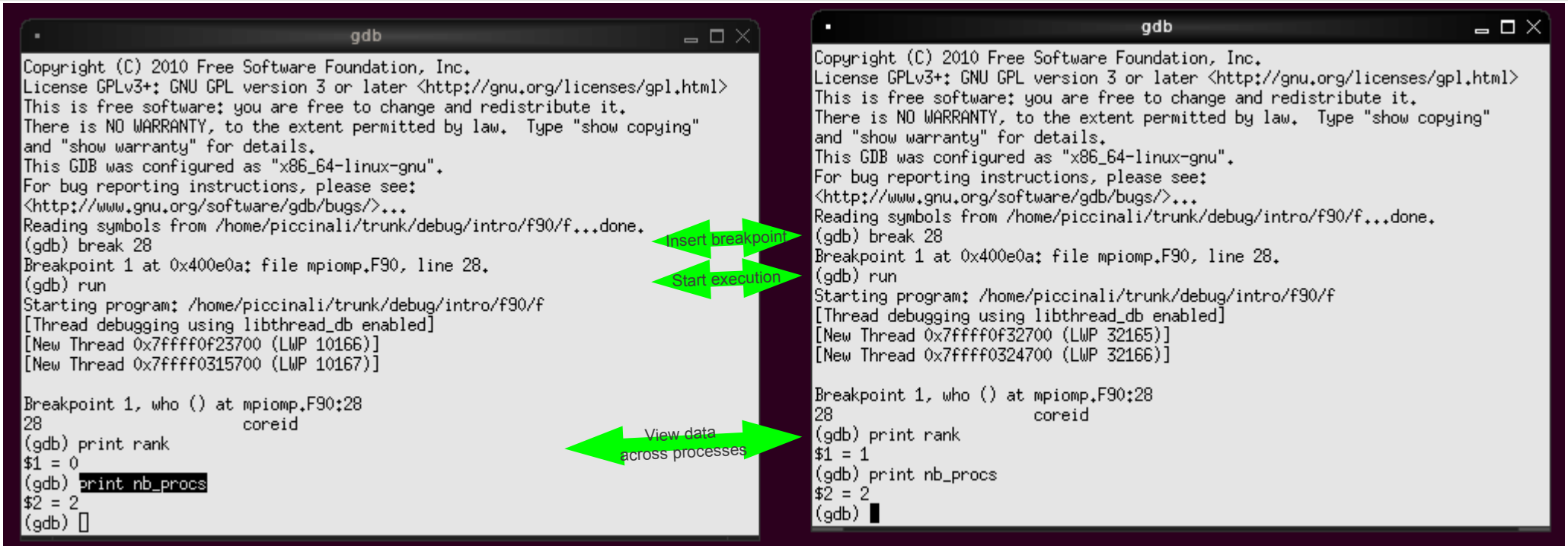
Commands for looking around

- list [<where>] : prints out source code at <where>
- backtrace [<n>] : prints a backtrace <n> levels deep
- info [<what>] : prints out info on <what>
- p[rint] [<expr>] : prints out <expr>
- d[isplay] : prints value of expression each time the program stops.



Demo : gdb (mpi only)

```
n1:/home/piccinali/trunk/debug/intro/f90 $ mpif90 -D_MPI -g mpiomp.F90 -L/softs/affinity -laff -o f
/softs/openmpi-1.4.3/bin/mpiexec -bind-to-core -n 2 -npernode 1 -x DISPLAY -hostfile /softs/openmpi-1.4.3/h xterm -rv -e gdb ./f
```




```
 2 program who
 3 #ifdef _MPI
 4         use mpi
 5 #endif
 6 !$       use omp_lib
 7         implicit none
 8         integer :: rank=-1,nb_procs=-1,code=-1,thread=-1,threads=-1
 9         integer ::  namelen=-1, coreid=-1
10         character(len=2) :: processor_name=""
11         integer, external :: running_on
12
13 #ifdef _MPI
14         call MPI_INIT (code)
15         call MPI_COMM_SIZE ( MPI_COMM_WORLD ,nb_procs,code)
16         call MPI_COMM_RANK ( MPI_COMM_WORLD ,rank,code)
17         call MPI_Get_processor_name( processor_name, namelen, code )
18 #endif
19
20 !$omp parallel private(thread)
21 !$       thread = omp_get_thread_num()
22 !$       threads = omp_get_num_threads()
23         coreid = running_on()
24         write (*,'(a17,i4,a2,i4,a5,i4,a2,i4,1x,a2,i4)') &
25                "hello in f90 rnk=",&
26                rank," /",nb_procs,&
27                " thd=",thread," /",threads, &
28                processor_name, &
29                coreid
30 !$omp end parallel
```

Demo : gdb (openmp only)

```
n1:/home/piccinali/trunk/debug/intro/f90 $  gfortran -fopenmp -g mpiomp.F90 -L/softs/affinity -laff -o f
n1:/home/piccinali/trunk/debug/intro/f90 $  export OMP_NUM_THREADS=2 ; gdb ./f
GNU gdb (Ubuntu/Linaro 7.2-1ubuntu11) 7.2
Copyright (C) 2010 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /home/piccinali/trunk/debug/intro/f90/f...done.
(gdb) break 26
Breakpoint 1 at 0x400ac4: file mpiomp.F90, line 26.
(gdb) run
Starting program: /home/piccinali/trunk/debug/intro/f90/f
[Thread debugging using libthread_db enabled]
[New Thread 0x7ffff6c94700 (LWP 10501)]

Breakpoint 1, MAIN__.omp_fn.0 (.omp_data_i=0x0) at mpiomp.F90:26
26                      " thd=",thread," /",threads, &
(gdb) print thread
$1 = 0
(gdb) info thread
  2 Thread 0x7ffff6c94700 (LWP 10501)  __lll_lock_wait () at ../nptl/sysdeps/unix/sysv
* 1 Thread 0x7ffff7fe5780 (LWP 10498)  MAIN__.omp_fn.0 (.omp_data_i=0x0) at mpiomp.F90
(gdb) thread 2
[Switching to thread 2 (Thread 0x7ffff6c94700 (LWP 10501))]#0  __lll_lock_wait () at .
136     ../nptl/sysdeps/unix/sysv/linux/x86_64/lowlevellock.S: No such file or directo
        in ../nptl/sysdeps/unix/sysv/linux/x86_64/lowlevellock.S
(gdb) cont
Continuing.
hello in f90 rnk=  -1 /  -1 thd=   0 /   2     11
[Switching to Thread 0x7ffff6c94700 (LWP 10501)]

Breakpoint 1, MAIN__.omp_fn.0 (.omp_data_i=0x0) at mpiomp.F90:26
26                      " thd=",thread," /",threads, &
(gdb) print thread
$2 = 1
(gdb) cont
Continuing.
hello in f90 rnk=  -1 /  -1 thd=   1 /   2     11
[Thread 0x7ffff6c94700 (LWP 10501) exited]

Program exited normally.
(gdb) q
```

Insert breakpoint

Start execution

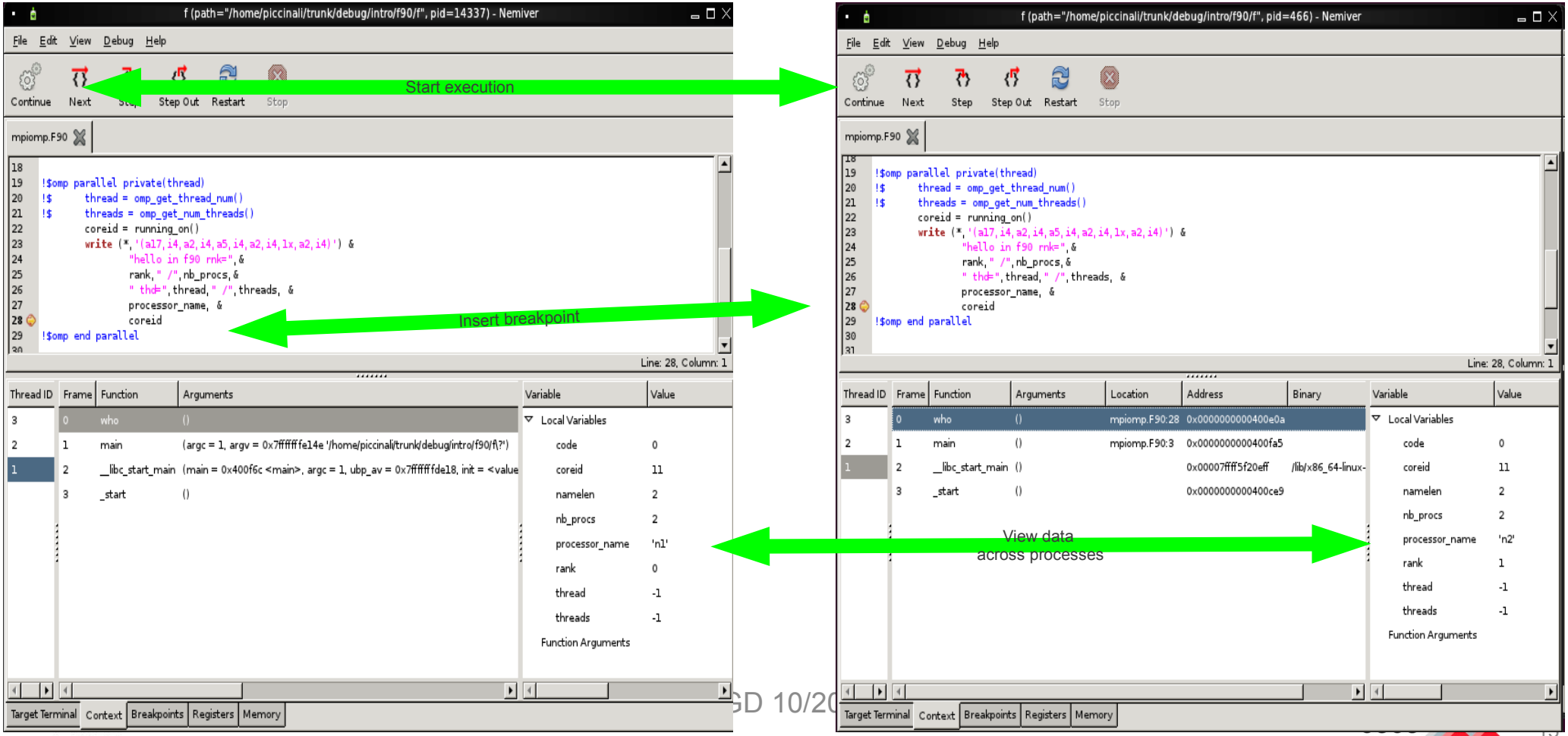
View data across threads

```
 2 program who
 3 #ifdef _MPI
 4         use mpi
 5 #endif
 6 !$      use omp_lib
 7         implicit none
 8         integer :: rank=-1,nb_procs
 9         integer ::  namelen=-1, co
10         character(len=2) :: process
11         integer, external :: runni
12
13 #ifdef _MPI
14         call MPI_INIT (code)
15         call MPI_COMM_SIZE ( MPI_C
16         call MPI_COMM_RANK ( MPI_C
17         call MPI_Get_processor_name
18 #endif
19
20 !$omp parallel private(thread)
21 !$      thread = omp_get_thread_nu
22 !$      threads = omp_get_num_threa
23         coreid = running_on()
24         write (*,'(a17,i4,a2,i4,a5,
25                "hello in f90 rnk=
26                rank," /",nb_procs,
27                " thd=",thread," /
28                processor_name, &
29                coreid
30 !$omp end parallel
31
32
33 #ifdef _MPI
34         call MPI_FINALIZE (code)
35 #endif
36
37 end program who
```

Demo : Nemiver

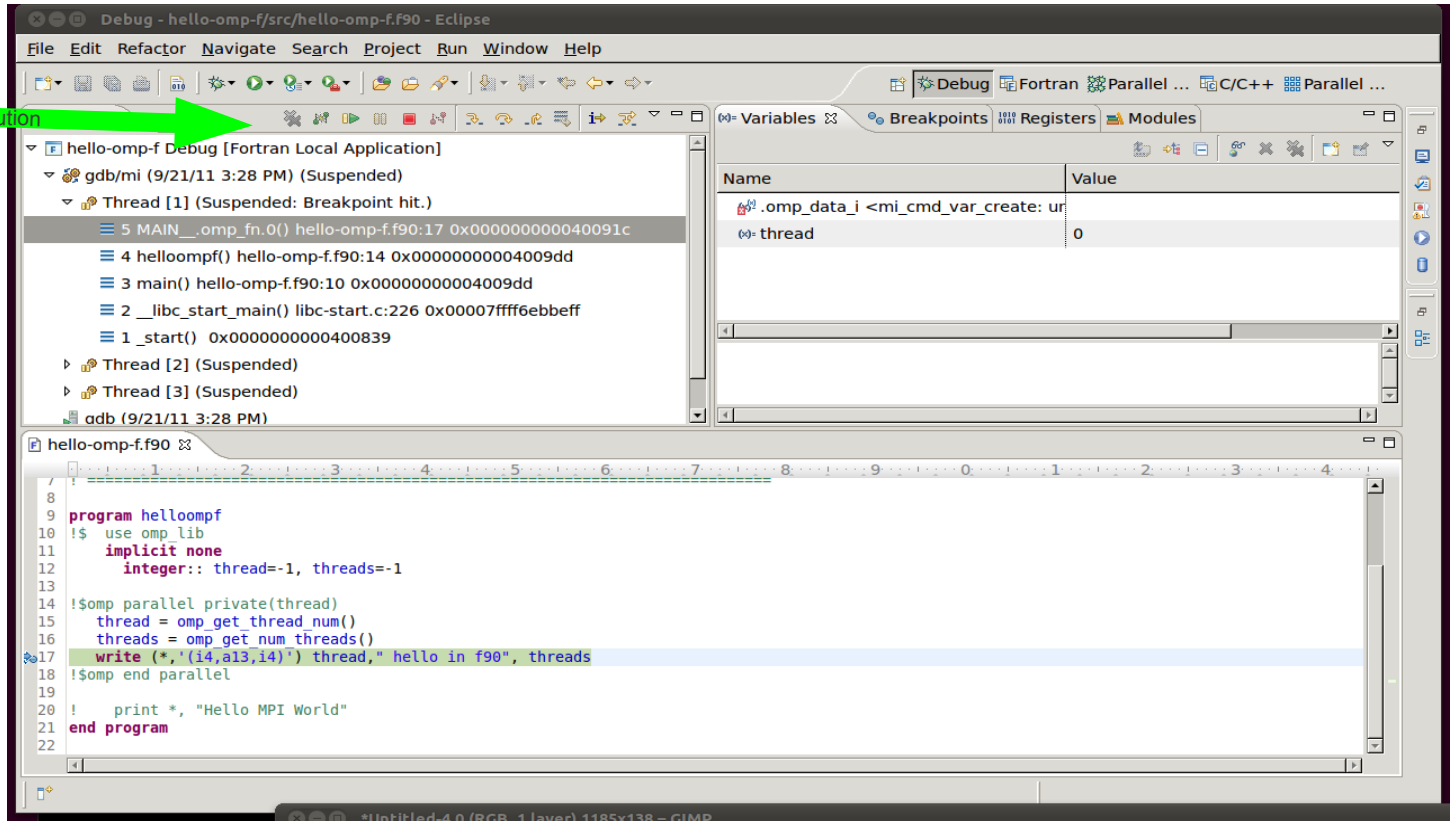
Demo : nemiver (mpi only)

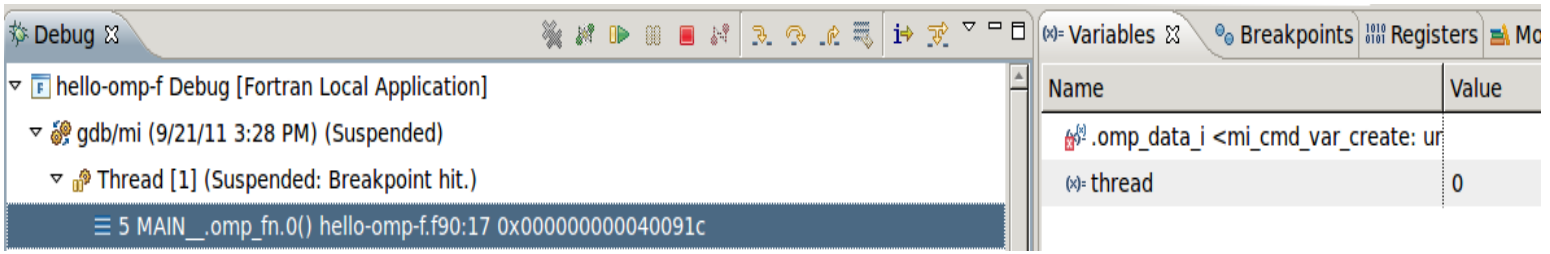
```
n1:/home/piccinali/trunk/debug/intro/f90 $  mpif90 -D_MPI -g mpiomp.F90 -L/softs/affinity -laff -o f

n1:/home/piccinali/trunk/debug/intro/f90 $  xhost
access control enabled, only authorized clients can connect
INET:n4.local
INET:n3.local
INET:n1.local
INET:n2.local

n1:/home/piccinali/trunk/debug/intro/f90 $  echo $DISPLAY
n1:1

/softs/openmpi-1.4.3/bin/mpiexec -n 2 -npernode 1 -x DISPLAY -hostfile /softs/openmpi-1.4.3/h nemiver ./f
```

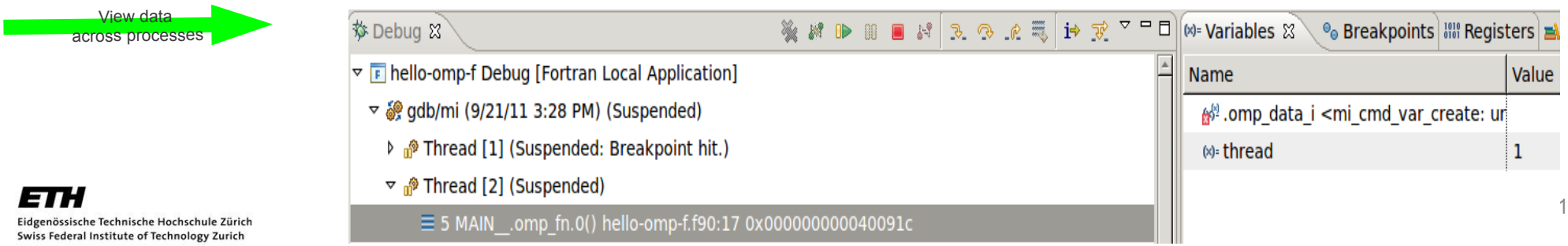


Demo : Eclipse

Demo : eclipse (openmp only)







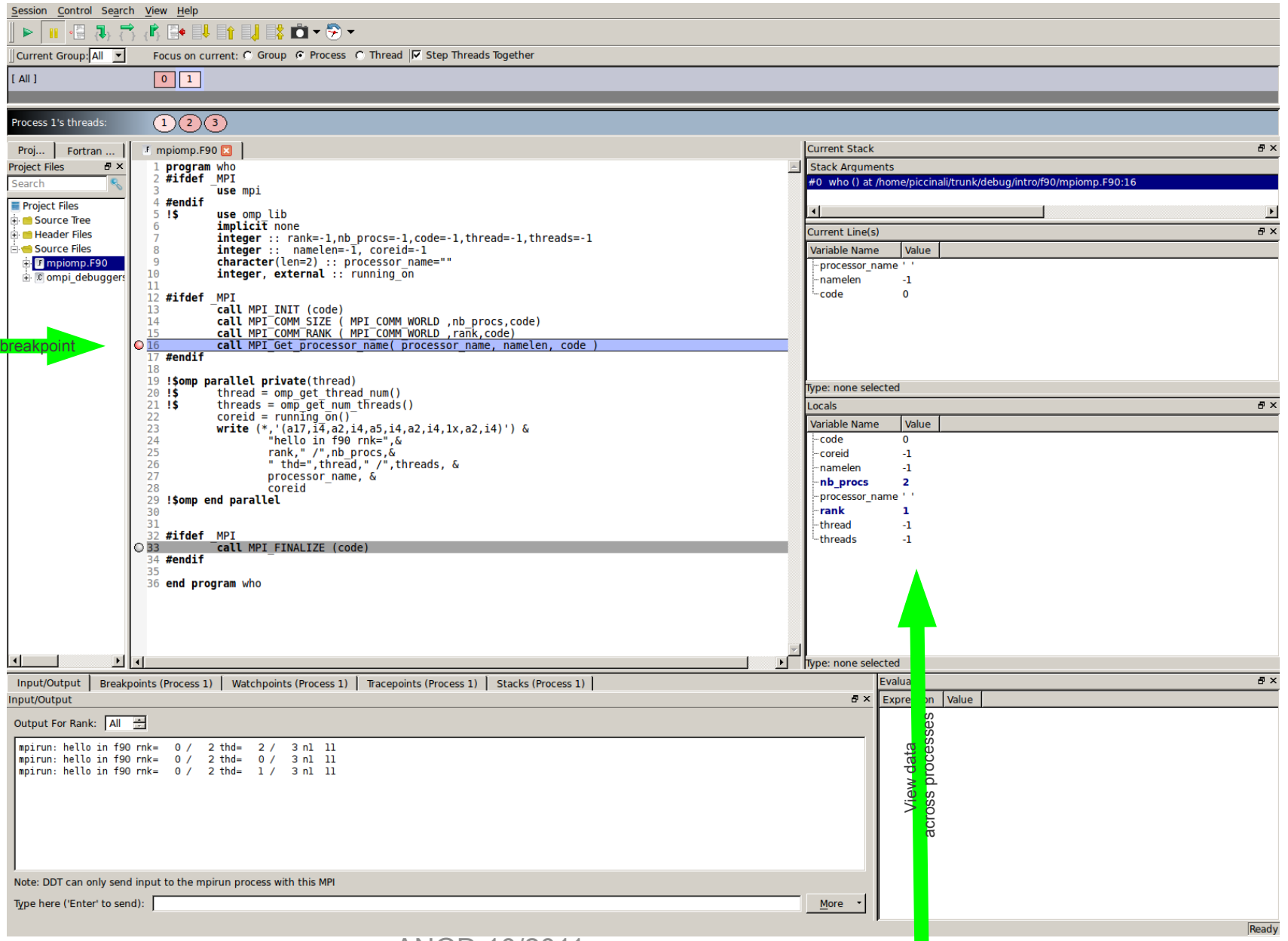
Demo : DDT

(thanks for the demo license)

Demo : ddt (impi/omp)

```
/softs/openmpi-1.4.3/bin/mpiexec -debug -n 2 -npernode 1 -x DISPLAY -x OMP_NUM_THREADS -hostfile /softs/openmpi-1.4.3/h ./f
```

Start execution




```
1  program who
2  #ifdef _MPI
3          use mpi
4  #endif
5  !$       use omp_lib
6          implicit none
7          integer :: rank=-1,nb_procs=-1,code=-1,thread=-1,threads=-1
8          integer ::  namelen=-1, coreid=-1
9          character(len=2) :: processor_name=""
10         integer, external :: running_on
11
12 #ifdef _MPI
13         call MPI_INIT (code)
14         call MPI_COMM_SIZE ( MPI_COMM_WORLD ,nb_procs,code)
15         call MPI_COMM_RANK ( MPI_COMM_WORLD ,rank,code)
16         call MPI_Get_processor_name( processor_name, namelen, code )
17 #endif
18
19 !$omp parallel private(thread)
20 !$       thread = omp_get_thread_num()
21 !$       threads = omp_get_num_threads()
22         coreid = running_on()
23         write (*,'(a17,i4,a2,i4,a5,i4,a2,i4,1x,a2,i4)') &
24                "hello in f90 rnk=",&
25                rank," /",nb_procs,&
26                " thd=",thread," /",threads, &
27                processor_name, &
28                coreid
29 !$omp end parallel
30
31
32 #ifdef _MPI
33         call MPI_FINALIZE (code)
34 #endif
35
36 end program who
```

Insert breakpoint

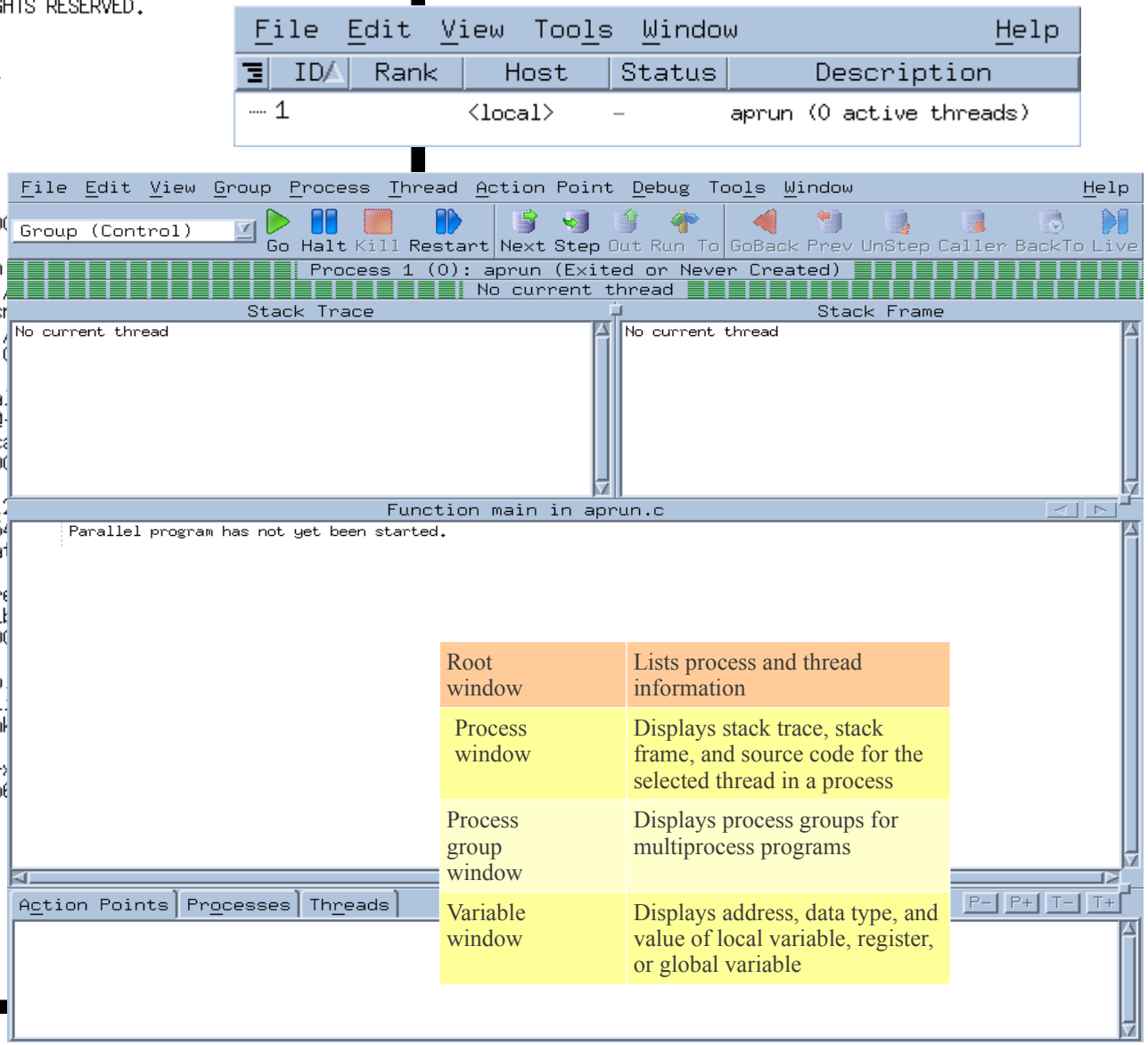
View data across processes

```
sudo sysctl -w kernel.yama.ptrace_scope=0
```

Demo : Totalview

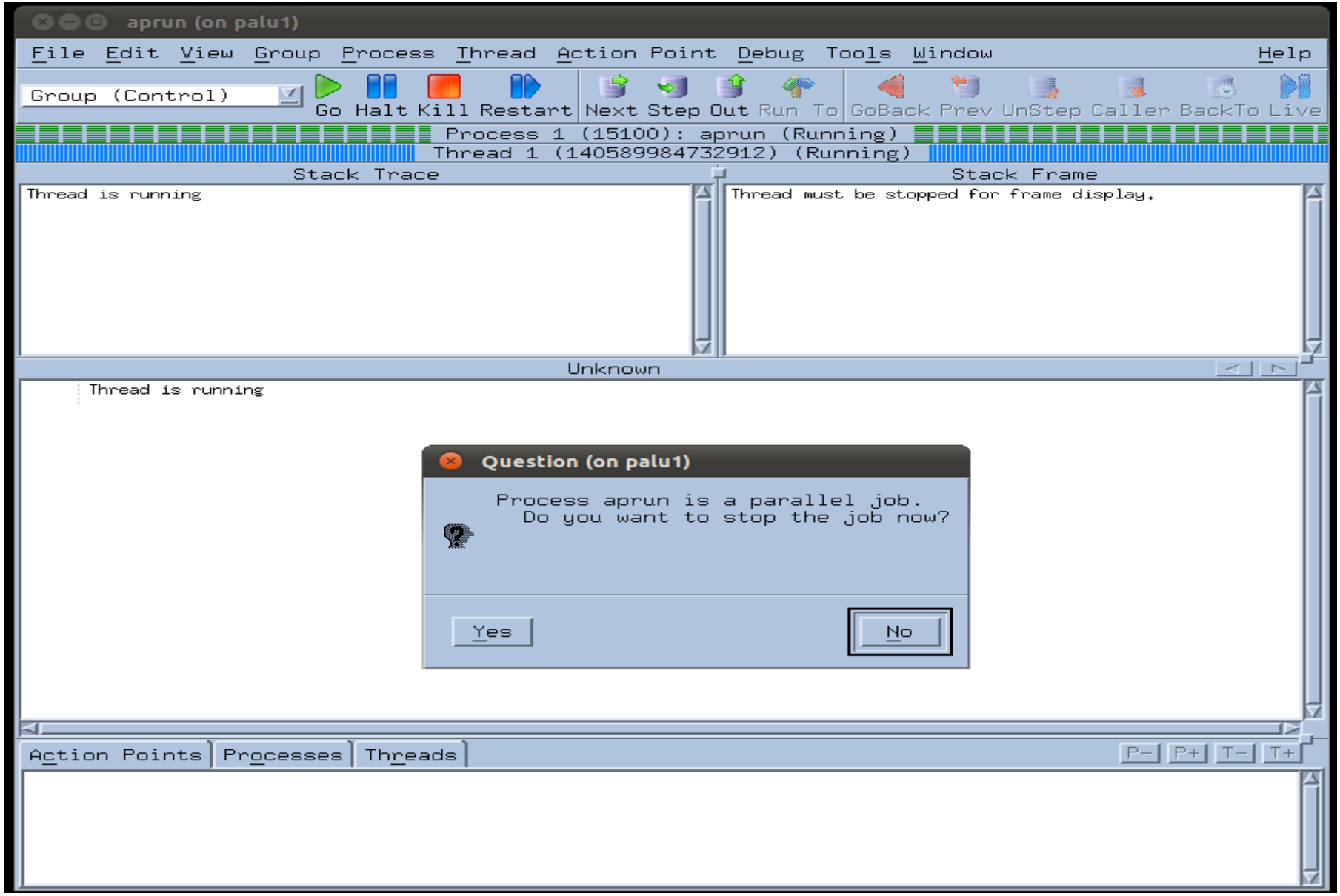
Launching Totalview (1)

```
 palu1: salloc -N2
salloc: Granted job allocation 2960
 palu1: export OMP_NUM_THREADS=2 ; totalview aprun -a -n2 -N1 -d24 exe
Linux x86_64 TotalView 8.8.0-1
Copyright 2007-2010 by TotalView Technologies, LLC. ALL RIGHTS RESERVED.
Copyright 1999-2007 by Etnus, LLC.
Copyright 1999 by Etnus, Inc.
Copyright 1996-1998 by Dolphin Interconnect Solutions, Inc.
Copyright 1989-1996 by BBN Inc.
TotalView Technologies ReplayEngine
Copyright 2010 TotalView Technologies
ReplayEngine uses the UndoDB Reverse Execution Engine
Copyright 2005-2010 Undo Limited
Reading symbols for process 1, executing "aprun"
Library /usr/bin/aprun, with 2 asects, was linked at 0x0040(
t 0xff00000090000000
Mapping 25131 bytes of ELF string data from '/usr/bin/aprun
Indexing 26800 bytes of DWARF '.debug_frame' symbols from '/
Indexing 28044 bytes of DWARF '.eh_frame' symbols from '/usr
Skimming 181212 bytes of DWARF '.debug_info' symbols from '/
Library @syscall_library@-64, with 1 asects, was linked at (
ially loaded at 0xff0000009009d800
INFO: Using previously cached local copy of library "@sysca.
Mapping 82 bytes of ELF string data from '@syscall_library@-
Indexing 192 bytes of DWARF '.eh_frame' symbols from '@sysca
Library /lib64/libdl.so.2, with 2 asects, was linked at 0x0(
d at 0xff0000009009e100
Mapping 459 bytes of ELF string data from '/lib64/libdl.so.2
Indexing 620 bytes of DWARF '.eh_frame' symbols from '/lib64
Library /lib64/libpthread.so.0, with 2 asects, was linked at
loaded at 0xff000000900a0500
Mapping 14508 bytes of ELF string data from '/lib64/libpthre
Indexing 12804 bytes of DWARF '.eh_frame' symbols from '/lib
Library /lib64/libc.so.6, with 2 asects, was linked at 0x00(
 at 0xff000000900baa00
Mapping 21676 bytes of ELF string data from '/lib64/libc.so.
Indexing 126700 bytes of DWARF '.eh_frame' symbols from '/li
Library /lib64/ld-linux-x86-64.so.2, with 2 asects, was link
ally loaded at 0xff0000009007ee00
Mapping 380 bytes of ELF string data from '/lib64/ld-linux-x
Indexing 6072 bytes of DWARF '.eh_frame' symbols from '/lib(
one
```

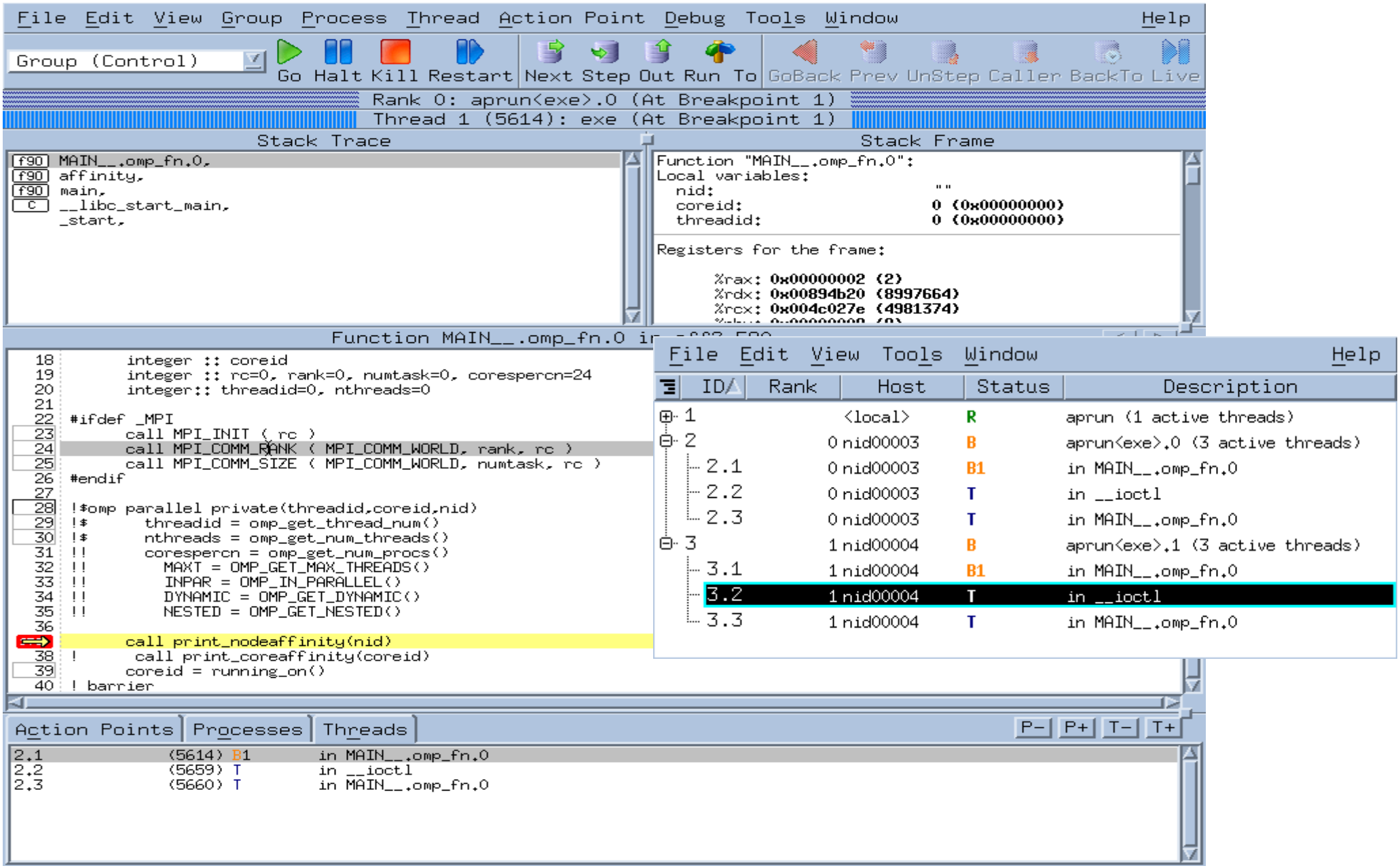


Root window	Lists process and thread information
Process window	Displays stack trace, stack frame, and source code for the selected thread in a process
Process group window	Displays process groups for multiprocess programs
Variable window	Displays address, data type, and value of local variable, register, or global variable

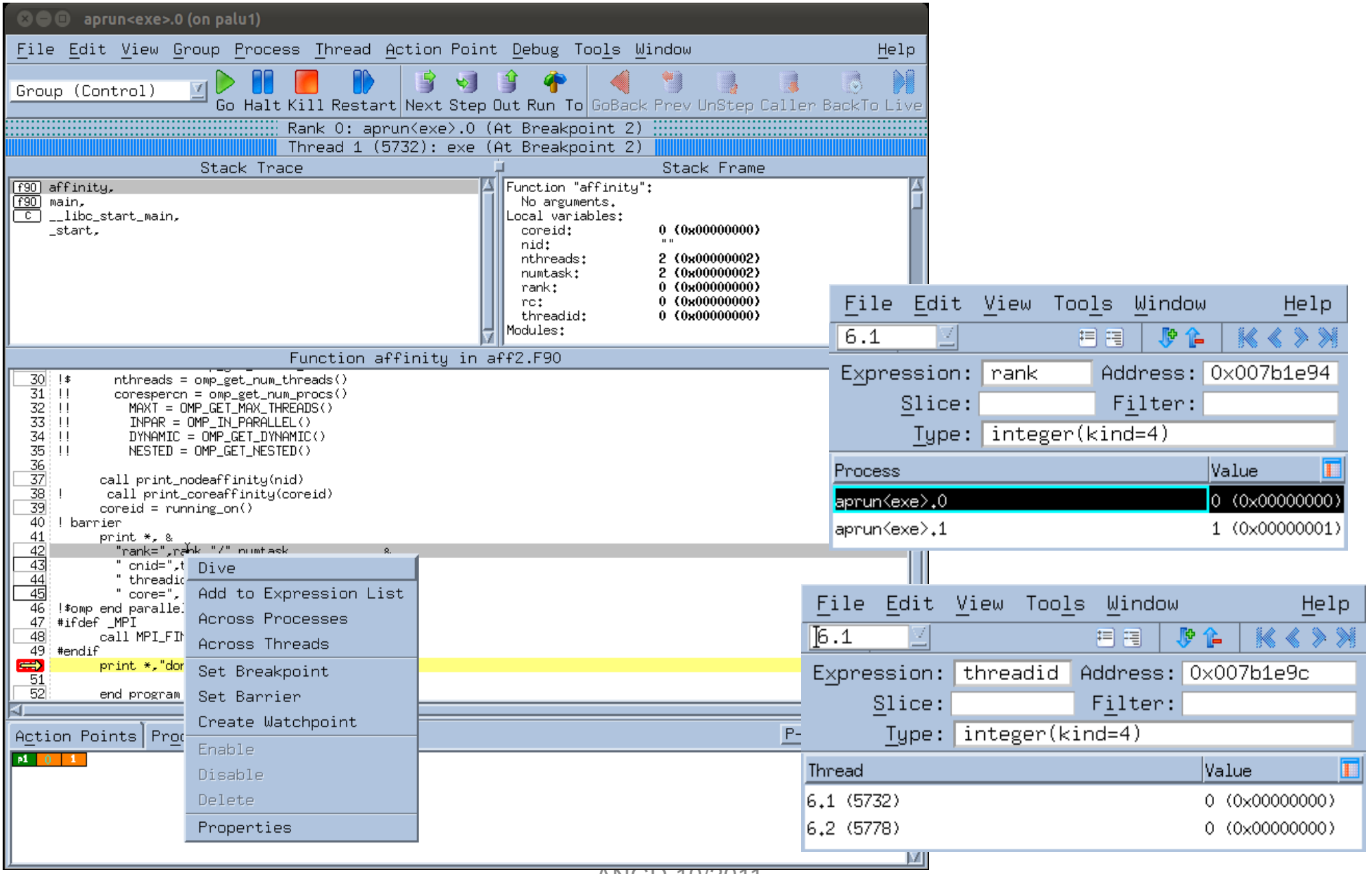
Launching Totalview (2)



Inserting Breakpoints



Viewing data across processes and threads



Restarting and exiting Totalview

```
Application 108251 resources: utime ~0s, stime ~0s
The TotalView Debugger Server has died
 palu1: export OMP_NUM_THREADS=2 ; totalview aprun -a -n2 -N1 -d24 exe
Linux x86_64 TotalView 8.8.0-1
Copyright 2007-2010 by TotalView Technologies, LLC. ALL RIGHTS RESERVED.
Copyright 1999-2007 by Etnus, LLC.
Copyright 1999 by Etnus, Inc.
Copyright 1996-1998 by Dolphin Interconnect Solutions, Inc.
Copyright 1989-1996 by BBN Inc.
TotalView Technologies ReplayEngine
Copyright 2010 TotalView Technologies
ReplayEngine uses the UndoDB Reverse Execution Engine
Copyright 2005-2010 Undo Limited
Reading symbols for process 1, executing "aprun"
Library /usr/bin/aprun, with 2 asects, was linked at 0x00400000, and initially loaded at 0xff000000900000
Mapping 25131 bytes of ELF string data from '/usr/bin/aprun'...done
Indexing 26800 bytes of DWARF '.debug_frame' symbols from '/usr/bin/aprun'...done
Indexing 28044 bytes of DWARF '.eh_frame' symbols from '/usr/bin/aprun'...done
Skimming 181212 bytes of DWARF '.debug_info' symbols from '/usr/bin/aprun'...done
Library @syscall_library@-64, with 1 asects, was linked at 0xfffffffff700000, and initially loaded at 0x
INFO: Using previously cached local copy of library "@syscall_library@-64"
Mapping 82 bytes of ELF string data from '@syscall_library@-64'...done
Indexing 192 bytes of DWARF '.eh_frame' symbols from '@syscall_library@-64'...done
Library /lib64/libdl.so.2, with 2 asects, was linked at 0x00000000, and initially loaded at 0xff000000900
Mapping 459 bytes of ELF string data from '/lib64/libdl.so.2'...done
Indexing 620 bytes of DWARF '.eh_frame' symbols from '/lib64/libdl.so.2'...done
Library /lib64/libpthread.so.0, with 2 asects, was linked at 0x00000000, and initially loaded at 0xff0000
Mapping 14508 bytes of ELF string data from '/lib64/libpthread.so.0'...done
Indexing 12804 bytes of DWARF '.eh_frame' symbols from '/lib64/libpthread.so.0'...done
Library /lib64/libc.so.6, with 2 asects, was linked at 0x00000000, and initially loaded at 0xff000000900b
Mapping 21676 bytes of ELF string data from '/lib64/libc.so.6'...done
Indexing 126700 bytes of DWARF '.eh_frame' symbols from '/lib64/libc.so.6'...done
Library /lib64/ld-linux-x86-64.so.2, with 2 asects, was linked at 0x00000000, and initially loaded at 0xf
Mapping 380 bytes of ELF string data from '/lib64/ld-linux-x86-64.so.2'...done
Indexing 6072 bytes of DWARF '.eh_frame' symbols from '/lib64/ld-linux-x86-64.so.2'...done
Library /lib64/libnss_files.so.2, with 2 asects, was linked at 0x00000000, and initially loaded at 0xff00
Mapping 2007 bytes of ELF string data from '/lib64/libnss_files.so.2'...done
Indexing 3260 bytes of DWARF '.eh_frame' symbols from '/lib64/libnss_files.so.2'...done
Launching TotalView Debugger Servers with command:
svrlaunch  /opt/toolworks/totalview.8.8.0a/linux-x86-64/bin/tvdsvrmain  '-verbosity info '  172.26.0.31
INFO: Using previously cached local copy of library "/dsl/var/spool/alps/108253/exe"
Library /dsl/var/spool/alps/108253/exe, with 2 asects, was linked at 0x00400000, and initially loaded at
Mapping 132443 bytes of ELF string data from '/dsl/var/spool/alps/108253/exe'...done
Indexing 79272 bytes of DWARF '.debug_frame' symbols from '/dsl/var/spool/alps/108253/exe'...done
Indexing 115132 bytes of DWARF '.eh_frame' symbols from '/dsl/var/spool/alps/108253/exe'...done
Skimming 1646889 bytes of DWARF '.debug_info' symbols from '/dsl/var/spool/alps/108253/exe'...done
Reading symbols for process 2, executing "./exe"
Reading symbols for process 3, executing "./exe"
 rank=          0 /          2  cnid=nid00003 threadid=          0 /          2  core=          0
 rank=          0 /          2  cnid=nid00003 threadid=          1 /          2  core=          1
 rank=          1 /          2  cnid=nid00004 threadid=          0 /          2  core=          0
 rank=          1 /          2  cnid=nid00004 threadid=          1 /          2  core=          1
 done
 done
Application 108253 resources: utime ~0s, stime ~0s
 palu1: exit
exit
salloc: Relinquishing job allocation 2960
salloc: Job allocation 2960 has been revoked.
 palu1:
```

Exemple 1 : Laplace

```
n1:/home/piccinali/trunk/debug/laplace $  make
mpif90 -O3 -g -c Laplace_mpi.F90
mpif90 -O3 -g -o f Laplace_mpi.o
GNU fortran executable ready
```

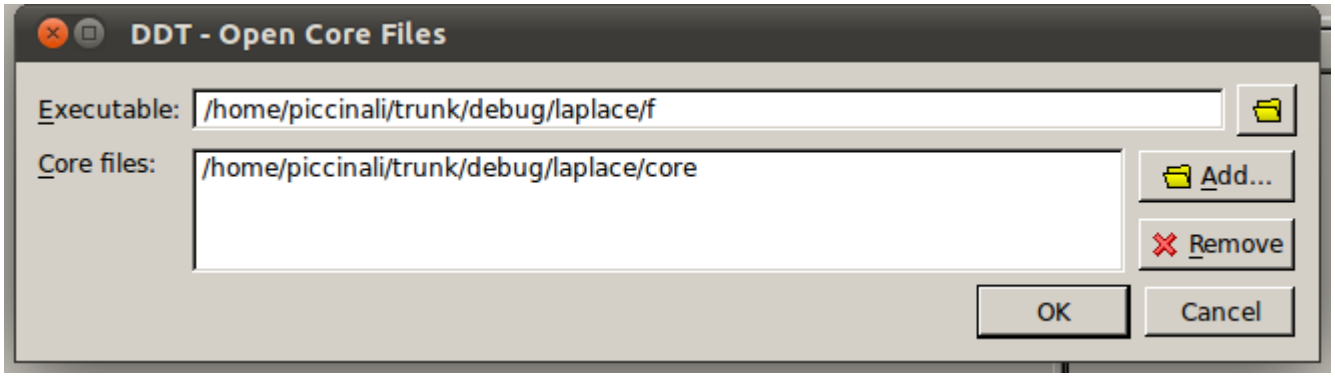
```
n1:/home/piccinali/trunk/debug/laplace $  /home/piccinali/sbatch.sh ./f 12 12 1 "1920 1920 10 1.0d-5" "" -bind-to-core
core file size          (blocks, -c) unlimited
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 20
file size               (blocks, -f) unlimited
pending signals                 (-i) 16382
max locked memory       (kbytes, -l) unlimited
max memory size         (kbytes, -m) unlimited
open files                      (-n) 1024
pipe size            (512 bytes, -p) 8
POSIX message queues     (bytes, -q) 819200
real-time priority              (-r) 0
stack size              (kbytes, -s) unlimited
cpu time               (seconds, -t) unlimited
max user processes              (-u) unlimited
virtual memory          (kbytes, -v) unlimited
file locks                      (-x) unlimited
+ export OMP_NUM_THREADS=1
+ OMP_NUM_THREADS=1
+ /usr/bin/time -p /softs/openmpi-1.4.3/bin/mpiexec -bind-to-core -n 12 -npernode 12 -x OMP_NUM_THREADS -hostfile /softs/openmpi-1.4.3/h ./f 1920 1920 10 1.0d-5
```

```
n1:/home/piccinali/trunk/debug/laplace $  tail o_f.0012.12.12.1.1920-1920-200-1.0d-5--bind-to-core
[n1:21918] [ 4] /lib/x86_64-linux-gnu/libc.so.6(__libc_start_main+0xff) [0x2b1310cffeff]
[n1:21918] [ 5] ./f() [0x401429]
[n1:21918] *** End of error message ***
--------------------------------------------------------------------------
mpiexec noticed that process rank 6 with PID 21917 on node n1 exited on signal 11 (Segmentation fault).
--------------------------------------------------------------------------
Command exited with non-zero status 139
```

Exemple 1 : Laplace (core file)

```
n1:/home/piccinali/trunk/debug/laplace $  file core
core: ELF 64-bit LSB core file x86-64, version 1 (SYSV), SVR4-style, from './f 1920 1920 10 1.0d-5'
```

```
n1:/home/piccinali/trunk/debug/laplace $  gdb f core
 Core was generated by `./f 1920 1920 10 1.0d-5'.
 Program terminated with signal 11, Segmentation fault.
 #0  init_solver (dxx=Cannot access memory at address 0xfe7fb0
 ) at Laplace_mpi.F90:377
 377                ve(i,j) = 2.0d0 * eps(i,j) * eps(i+1,j  ) / ( (eps(i,j) + eps(i+1,j) ) * dxx**2)
```



Recompiler avec -fbacktrace

Exemple 1 : Laplace (DDT debugger)

```
/softs/openmpi-1.4.3/bin/mpiexec -debug -bind-to-core -n 12 -npernode 12 -hostfile /softs/openmpi-1.4.3/h ./f 1920 1920 10 1.0d-5
```

Distributed Debugging Tool — Wed Sep 21 20:53 piccinal

Session Control Search View Help

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

All: 0 1 2 3 4 5 6 7 8 9 10 11

Create Group

Proj... Fortran ...

Project Files

Search

- Project Files
  - Source Tree
  - Header Files
  - Source Files
    - Laplace_mpi.F90
    - ompi_debuggers

Laplace_mpi.F90

```
356         Integer, INTENT(IN)   :: imin,imax,jmin,jmax
357         Real(rp), INTENT(IN)  :: dxx, dyy
358         Real(rp), INTENT(OUT) :: omega
359         Real(rp), INTENT(OUT) :: omega1
360         Real(rp), INTENT(IN),    Dimension(:,:) :: eps
361         Real(rp), INTENT(INOUT), Dimension(:,:) :: ve, vn, vs, vw, vc, sou
362         Integer :: i,j
363 !------------------------------------
364 ! Resolution de l'equation de Laplace
365 !------------------------------------
366
367 ! choix de la methode de resolution
368 ! omega = 1 : Gauss-Seidel
369 ! 1<omega<2 : Sur-relaxation
370       omega = 1.5d0
371       omega1 = 1.0d0 - omega
372
373 ! Definition des coefficients de l'equation de Laplace
374 ! cas general
375       do j = jmin, jmax
376          do i = imin, imax
377             ve(i,j) = 2.0d0 * eps(i,j) * eps(i+1,j  ) / ( (eps(i,j) + eps(i+1,j) ) * dxx**2)
378             vn(i,j) = 2.0d0 * eps(i,j) * eps(i  ,j+1) / ( (eps(i,j) + eps(i,j+1) ) * dyy**2)
379             vs(i,j) = 2.0d0 * eps(i,j) * eps(i  ,j-1) / ( (eps(i,j) + eps(i,j-1) ) * dyy**2)
380             vw(i,j) = 2.0d0 * eps(i,j) * eps(i-1,j  ) / ( (eps(i,j) + eps(i-1,j) ) * dxx**2)
381             vc(i,j) = omega / ( vw(i,j) + vn(i,j) + vs(i,j) + ve(i,j) )
382             sou(i,j) = 0.0d0
383          enddo
384       enddo
385
386 end subroutine init_solver
387
388 subroutine solver(omega1, V,V_old, sou,ve,vw,vn,vs,vc, imin,imax,jmin,jmax, j)
389
390         implicit none
391         Integer,  Parameter  ::  rp = Kind (1.0d0)
392         Real(rp), INTENT(INOUT), Dimension(imin-1:imax+1,jmin-1:jmax+1) :: V
393         Real(rp), INTENT(IN), Dimension(imin-1:imax+1,jmin-1:jmax+1) :: V_old
394         Real(rp), INTENT(IN) :: omega1
395         Real(rp), INTENT(IN), Dimension(imin:imax,jmin:jmax) :: sou, ve, vw, vn, vs, vc
396         Integer, INTENT(IN)   :: imin,imax,j,jmin,jmax
397         Integer :: i, istat
398
399         do i = imin, imax
```

Current Stack

Stack Arguments

#1 laplace () at /home/piccinali/trunk/debug/laplace/Laplace_mpi.F90:212 (at 0x0000000000402

#0 init_solver (dxx=0.00052083333333333333, dyy=0.00052083333333333333, eps=<value

Distributed Debugging Tool

Processes 0-11:

Process stopped in init_solver (Laplace_mpi.F90:377) with signal SIGSEGV (Segmentation fault).
Reason/Origin: the kernel
Your program will probably be terminated if you continue.
You can use the stack controls to see what the process was doing at the time.

☑ Always show this window for signals

Continue Pause

Variable Name	Value
dxx	0.00052083333333333333
dyy	0.00052083333333333333
eps	<value optimized out>
ve	<value optimized out>
vn	<value optimized out>
vs	<value optimized out>
vw	<not allocated>
vc	<not allocated>
sou	<not allocated>
imin	1
imax	480
jmin	1
jmax	640
omega	1.5
omega1	-0.5
eps.0	<unknown bounds>
i	1

Type: none selected

Exemple 1 : Laplace (solution)

Dans la subroutine init_solver, les dimensions des tableaux doivent etre declarees :

```
Real(rp), INTENT(IN),    Dimension(imin-1:imax+1,jmin-1:jmax+1) :: eps
Real(rp), INTENT(INOUT), Dimension(imin:imax,jmin:jmax) :: ve
Real(rp), INTENT(INOUT), Dimension(imin:imax,jmin:jmax) :: vn
Real(rp), INTENT(INOUT), Dimension(imin:imax,jmin:jmax) :: vs
Real(rp), INTENT(INOUT), Dimension(imin:imax,jmin:jmax) :: vw
Real(rp), INTENT(INOUT), Dimension(imin:imax,jmin:jmax) :: vc
Real(rp), INTENT(INOUT), Dimension(imin:imax,jmin:jmax) :: sou
```

Au lieu de :

```
Real(rp), INTENT(INOUT), Dimension(:,:) :: ve, vn, vs, vw, vc, sou
Real(rp), INTENT(IN),    Dimension(:,:) :: eps
```

Exemple 2 : loopy (openmp)

```
n1:/home/piccinali/trunk/debug/datarace $  make
gfortran -fopenmp -c loopy.F90
gfortran -fopenmp -o f loopy.o
GNU fortran executable ready
```

Executez le programme :

➔ export OMP_NUM_THREADS=1 ; ./f

Notez les valeurs de sum1 et sum2

Executez avec un nombre de threads > 1 et comparez les resultats.

Utilisez un debugger pour examiner les threads.

Exemple 2 : loopy (solution)

```
n1:/home/piccinali/trunk/debug/datarace $  export OMP_NUM_THREADS=1 ; ./f
 threadid= 0/ 1
 sum1=          -28
           1           -4           -9          -16
 sum2=          -12
           1           -1           -4           -8

n1:/home/piccinali/trunk/debug/datarace $  export OMP_NUM_THREADS=2 ; ./f
 threadid= 1/ 2
 threadid= 0/ 2
 sum1=          -28
           1           -4           -9          -16
 sum2=           -5
           1           -1           -4           -1

n1:/home/piccinali/trunk/debug/datarace $  export OMP_NUM_THREADS=6 ; ./f
 threadid= 5/ 6
 threadid= 4/ 6
 threadid= 0/ 6
 threadid= 2/ 6
 threadid= 1/ 6
 threadid= 3/ 6
 sum1=          -28
           1           -4           -9          -16
 sum2=           -5
           1           -1           -4           -1
```

Exemple 2 : loopy (solution)

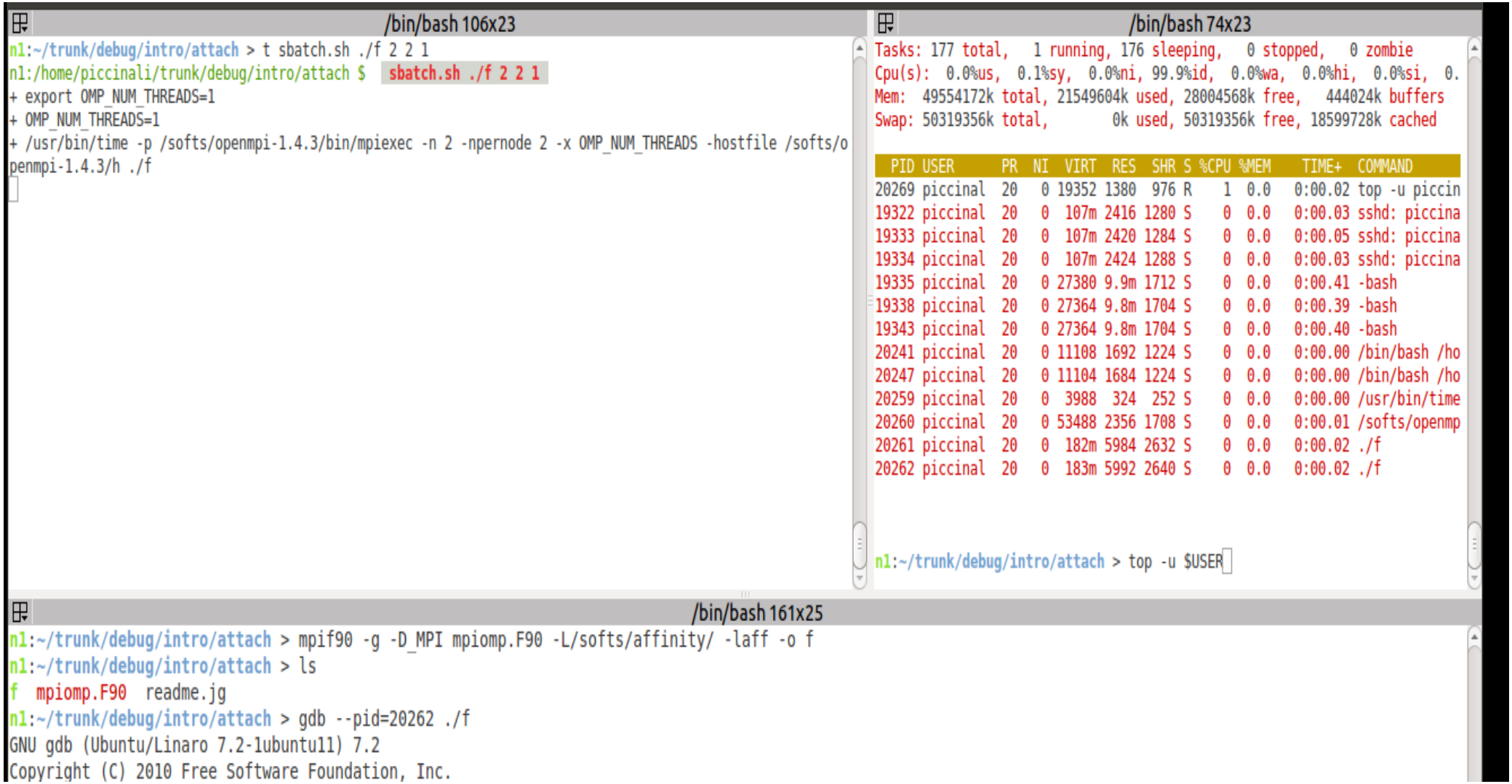
Les résultats de la boucle 2 dépendent de la façon dont elle est parallélisée ou non.

Vous devriez avoir remarqué des résultats différents en fonction du nombre de threads, ce qui est typique des data race. La boucle2 n'est pas parallélisable a cause de la dépendance entre les données.

Il n'y a pas vraiment de solution, a moins de restructurer la boucle ou d'utiliser la clause reduction pour faire la somme. La plupart du temps, les resultats ne sont pas reproductibles, ce qui rend le debuggage encore plus difficile.

http://en.wikipedia.org/wiki/OpenMP

Exemple 3 : attach to gdb (mpi)



Exemple 3 : attach to gdb (mpi)

/bin/bash 106x10

```
+ exit 0

n1:~/trunk/debug/intro/attach >
```

/bin/bash

```
n1:~/trunk/debug/intro/attach > touch go
n1:~/trunk/debug/intro/attach >
```

/bin/bash 160x37

```
(gdb) bt
#0  0x00002b05c72135ad in nanosleep () at ../sysdeps/unix/syscall-template.S:82
#1  0x00002b05c721343c in __sleep (seconds=0) at ../sysdeps/unix/sysv/linux/sleep.c:138
#2  0x0000000000400f2c in who () at mpiomp.F90:27
#3  0x00000000004010c9 in main (argc=1, argv=0x7fff59cf41c0 "./f") at mpiomp.F90:3
#4  0x00002b05c7184eff in __libc_start_main (main=0x401090 <main>, argc=1, ubp_av=0x7fff59cf2ad8, init=<value optimized out>, fini=<
    rtld_fini=<value optimized out>, stack_end=0x7fff59cf2ac8) at libc-start.c:226
#5  0x0000000000400d99 in _start ()
(gdb) break mpiomp.F90:32
Breakpoint 1 at 0x400f94: file mpiomp.F90, line 32.
(gdb) cont
Continuing.

Breakpoint 1, who () at mpiomp.F90:32
32                      rank," /",nb_procs,&
(gdb) print rank
$1 = 1
(gdb) print nb_procs
$2 = 2
(gdb) q
A debugging session is active.

        Inferior 1 [process 20262] will be detached.

Quit anyway? (y or n) y
Detaching from program: /home/piccinali/trunk/debug/intro/attach/f, process 20262
n1:~/trunk/debug/intro/attach >
```