



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Parallel Debugging in TotalView®

is a

- multi-process, multi-thread
 - graphical user interface (GUI) -based
 - program source (*C, C++, Fortran*)
 - and machine level (*Assembler code*)
- code defect analysis tool

Supported platforms



- Operating systems: Linux, Mac OS X, Unix (...)
- Processor architecture: x86 (32-bit, 64-bit)
- Networks: InfiniBand, Ethernet
- Accelerators and coprocessors:
GPU (Nvidia Tesla, Fermi),
Xeon Phi (Intel MIC)
(early access, special licence)

Languages and extensions



- C, C++
- Fortran (77, 90)
- Assembler
- MPI (Open MPI, etc.) (*multi-process*)
- OpenMP (*multi-thread*)
- CUDA, OpenACC (*GPU, Xeon Phi*)
- UPC, Global Arrays

Compiling a program



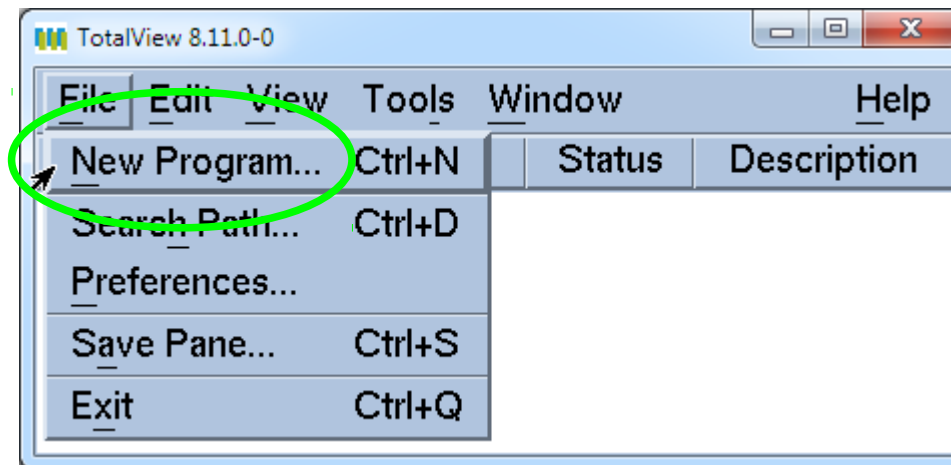
produce debugging information
and turn off optimizations, e.g.:

- > gcc -O0 -g -fopenmp ... (*OpenMP, C*)
- > mpicc -O0 -g ... (*Open MPI, C*)
- > mpicxx -O0 -g ... (*Open MPI, C++*)
- > mpif77 -O0 -g ... (*Open MPI, Fortran 77*)
- > mpif90 -O0 -g ... (*Open MPI, Fortran 90*)

Loading a program

e.g.:

- > totalview OMP_Program [-a ...]
- > totalview -np ... MPI_Program [-a ...]
- or open it from a root window:



Root Window: global view



ID	Rank	Host	Status	Description
1	0	10.32.5.40	R	wmt.mpi.0 (3 active threads)
1.1	0	10.32.5.40	R	in Some_Scientific_Computation
1.2	0	10.32.5.40	R	in __select_nocancel
1.3	0	10.32.5.40	R	in poll
21	1	10.32.5.40	R	wmt.mpi.1 (3 active threads)
21.1	1	10.32.5.40	R	in Some_Scientific_Computation
21.2	1	10.32.5.40	R	in __select_nocancel
21.3	1	10.32.5.40	R	in __poll
22	2	10.32.5.40	R	wmt.mpi.2 (3 active threads)
22.1	2	10.32.5.40	R	in Some_Scientific_Computation
22.2	2	10.32.5.40	R	in __select_nocancel
22.3	2	10.32.5.40	R	in __poll

Process Window: detailed view



wmt.mpi.0

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

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

Rank 0: wmt.mpi.0 (Stopped)

Thread 1 (140243879896864): wmt.mpi (Stopped) <Trace Trap>

Stack Trace

C++	Some_Scientific_Computation, FP=
C++	main, FP=7f7f67bd7f90
	__libc_start_main, FP=7fff67bd80
	_start, FP=7fff67bd8050

Stack Frame

Function "Some_Scientific_Computation":

data: 0x7f8d01e55040 -> (struct DataStruc

num: 0x00100000 (1048576)

xyz: 1365159587.12536

Block "\$b1#\$b1":

i: 0x00000002 (2)

Block "\$b1":

a: 1365159588.35936

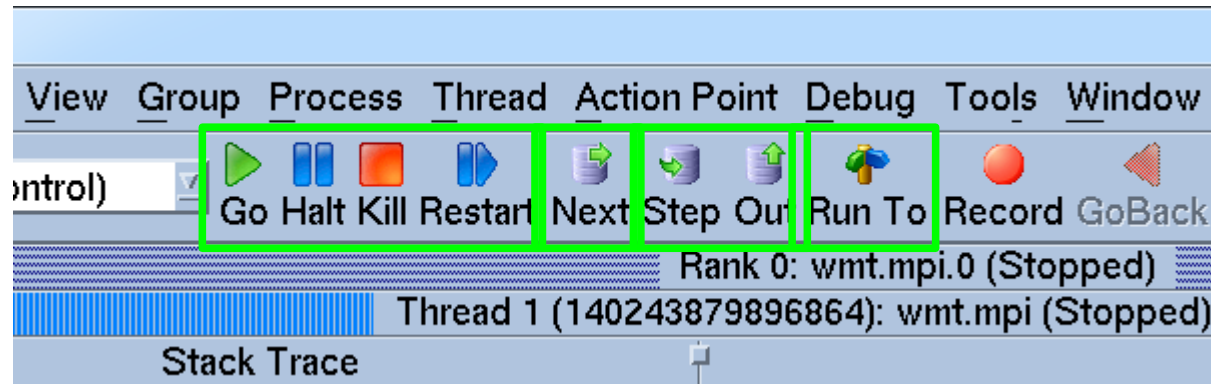
Function Some_Scientific_Computation in mathcore.cpp

```
16 void Some_Scientific_Computation ( DataStruct *data, int num, double xyz )
17 {
18     double a = 1.234; // an important physical constant
19     a += xyz; // + some randomness from outside
20
21     for( int i=0; i<num; i++ ) // filling data array with some values
22         data[i].x = sin(a-i),
23         data[i].y = cos(a+i),
24         data[i].z = tan(a+(data[i].colour=num));
25 }
```

Action Points Processes Threads

1.1	(140243879896864)	T	in Some_Scientific_Computation
1.2	(140243728090880)	T	in __select_nocancel
1.3	(140243711162112)	T	in __poll

Basic controls



- run, stop, kill, restart (kill+run)
- step by step (line by line) execution
- move in and out of functions
- run up to a selected point

Setting (multiple) breakpoints



The screenshot shows the TotalView debugger interface. The main window displays the source code for the function `Some_Scientific_Computation` in `mathcore.cpp`. A red arrow icon, representing a breakpoint, is placed on line 22, which is highlighted in yellow. The code on line 22 is `data[i].x = sin(a-i),`. The `Action Points` pane at the bottom shows a list of breakpoints: `8 mathcore.cpp#22 Some_Scientific_Computation+0x3e` and `9 wmt.cpp#57 main+0x179`. The `Stack Trace` pane shows the current stack frames, including `Some_Scientific_Computation` and `main`. The `Stack Frame` pane shows the current frame's variables: `data` (0x7f372dd1b040), `num` (0x00100000), `xyz` (1365159587.12537), `i` (0x00000005), and `a` (1365159588.35937). The `Thread` pane shows `Thread 1 (139875249620768): wmt.mpi.1 (At Breakpoint 8)`.

Switching between processes



Rank 1: wmt.mpi.1 (Stopped)

Thread 1 (139875249620768): wmt.mpi (Stopped) <Trace Trap>

Stack Trace	Stack Frame
C++ Some_Scientific_Computation, FP=7ffffd6aed1c0	Function "Some_Scientific_Computation":
C++ main, FP=7ffffd6aed1c0	data: 0x7f372dd1b040 -> (struct DataStruc
__libc_start_main, FP=7ffffd6aed2_start,	num: 0x00100000 (1048576)
FP=7ffffd6aed280	xyz: 1365159587.12537
	Block "\$b1#\$b1":
	i: 0x00000005 (5)
	Block "\$b1":
	a: 1365159588.35937

```
17 {
18 double a = 1.234; // an important physical constant
19 a += xyz; // + some randomness from outside
20
21 for( int i=0; i<num; i++ ) // filling data array with some values
    data[i].x = sin(a-i),
23 data[i].y = cos(a+i),
24 data[i].z = tan(a+(data[i].colour=num));
25 }
26
```

Action Points | Processes | Threads

0 1 2

P- P+ T- T+

Switching between threads



Thread 1 (39875249620768): wmt.mpi (Stopped) <Trace Trap>

Stack Trace

- Some_Scientific_Computation, FP=7ffffd6aed1c0
- main, FP=7ffffd6aed2
- __libc_start_main, FP=7ffffd6aed280

Stack Frame

Function "Some_Scientific_Computation":

- data: 0x7f372dd1b040 -> (struct DataStruc
- num: 0x00100000 (1048576)
- xyz: 1365159587.12537
- Block "\$b1#\$b1":
- i: 0x00000005 (5)
- Block "\$b1":
- a: 1365159588.35937

Function Some_Scientific_Computation in mathcore.cpp

```
17 {
18 double a = 1.234; // an important physical constant
19 a += xyz; // + some randomness from outside
20
21 for( int i=0; i<num; i++ ) // filling data array with some values
    data[i].x = sin(a-i),
23 data[i].y = cos(a+i),
24 data[i].z = tan(a+(data[i].colour=num));
25 }
26
```

Threads

ID	Process	Type	Function
3.1	(139875249620768)	T	in Some_Scientific_Computation
3.2	(139875097814784)	T	in __select_nocancel
3.3	(139875080886016)	T	in __poll

Watching data



- in the Stack Frame
- in context tooltips
- in a Variable Window
- added to the Expression List
- in arrays and structures
- across processes and threads

(no more need to „printf“)

Watching data



The screenshot displays the TotalView IDE interface for debugging a program. The main window shows a C code snippet with a loop:

```
23 data[i].x = sin(a+i),  
24 data[i].y = cos(a+i),  
25 data[i].z = tan(a+(data[i].colour * num));  
26 }
```

The variable `num` is circled in green in the code. A context menu is open over `num`, with the following options circled in green:

- Create Watchpoint...
- Add to Expression List
- Add to Block Properties...
- Visualize
- Visualize Distribution
- Statistics
- Attach Subset (Array of Ranks)...
- Array Viewer

The watch window on the left shows the structure `DataStruct` with the following fields:

Field	Type	Process	Value
x	double	wmt.mpi.0	0
y	double	wmt.mpi.0	0
z	double	wmt.mpi.0	0
colour	int	wmt.mpi.0	0x00000000 (0)
x	double	wmt.mpi.1	0
y	double	wmt.mpi.1	0
z	double	wmt.mpi.1	0
colour	int	wmt.mpi.1	0x00000000 (0)
x	double	wmt.mpi.2	0
y	double	wmt.mpi.2	0

The watch window on the right shows the expression `num+3` with the value `0x00100003 (1048579)`. The value of `num` is also shown as `0x00000004 (4)` and `1365358803.25578`.