



High performance tools to debug, profile, and analyze your applications

## Illuminating OpenMP + MPI Performance

Beau Paisley, [bpaisley@allinea.com](mailto:bpaisley@allinea.com)



# HPC means being able to work productively on remote machines

The screenshot shows the Alinea Forge application window. The title bar includes 'File Edit View Window Help'. On the left sidebar, there are logos for 'allinea FORGE', 'allinea DDT', and 'allinea MAP'. Below these are links for 'Support', 'Tutorials', and 'allinea.com', along with a red error message: 'No licence found ?'. The main area displays a 'PROFILE' section with the text 'Profile a program.' and a 'LOAD PROFILE DATA FILE' section. A modal dialog box titled 'Connect to Remote Host' is open, showing 'Connecting to titan ...' with a progress bar. The dialog contains a terminal window with the following text:

```
/home/mark/Work/code/jon/libexec/remote-exec titan ~/allinea//libexec/ddt-remote  
d  
*****  
* Please send questions or comments to the NCCS User Assistance Center  
* help@olcf.ornl.gov  
* http://www.olcf.ornl.gov/support  
*  
*****  
Enter PASSCODE: █
```

The dialog also features a '<< Hide Terminal' button and a 'Cancel' button.

At the bottom of the application window, the status bar reads: 'Alinea Ultimate map-smoketest-scripts-5.0 0a3f65bcf767 Nov 7 2014'.



# Calculating pi with 1 thread, 1 process

```
bpaisley@beau: ~/ownCloud/bpaisley/src/pi
bpaisley@beau:~/ownCloud/bpaisley/src/pi$ time cpi
initializing...
calculating pi...
pi is approximately 3.1415926535897416, Error is 0.00000000000000515
writing to disk...
finished

real    1m15.971s
user    1m13.306s
sys     0m0.236s
bpaisley@beau:~/ownCloud/bpaisley/src/pi$
```



## Profile with 1 process, 1 thread

```
bpaisley@beau: ~/ownCloud/bpaisley/src/pi
bpaisley@beau:~/ownCloud/bpaisley/src/pi$ map --profile ./cpi
Allinea Forge 5.0.1 - Allinea MAP
Profiling      : /home/bpaisley/ownCloud/bpaisley/src/pi/cpi
Allinea sampler : preload
MPI enabled    : No

initializing...
calculating pi...
pi is approximately 3.1415926535897416, Error is 0.0000000000000515
writing to disk...
finished

MAP analysing program...
MAP gathering samples...
MAP generated /home/bpaisley/ownCloud/bpaisley/src/pi/cpi_1p_2015-04-28_15-47.map
bpaisley@beau:~/ownCloud/bpaisley/src/pi$
```

# Using MAP to analyze our 1 process, 1 thread run

The screenshot displays the Allinea MAP interface for a process named 'cpi\_1p\_1t\_2015-04-29\_07-54.map'. The main window shows performance metrics for the main thread activity, including CPU floating-point usage (90.4% avg) and memory usage (11,583 kB avg). A summary bar indicates that the main thread spent 96% of its time in compute, 4% in file I/O, and used 11,583 kB of memory.

The code editor shows the source code for 'cpi.c', with a call stack view on the left. The call stack shows the following functions and their respective time percentages:

- 74.6%: f (sum += f(x);)
- 21.8%: x = h \* ((double)i - 0.5);
- 3.6%: do\_io

The 'Main Thread Stacks' table at the bottom provides a detailed view of the stack frames:

Total core time	Function(s) on line	Source	Position
74.6%	f	sum += f(x);	cpi.c:66
21.8%		x = h * ((double)i - 0.5);	cpi.c:65
3.6%	do_io	do_io(); /* unnecessary i/o for illustration purpo...	cpi.c:76

The status bar at the bottom indicates that the data is shown from 1,000 samples taken over 1 process (1000 per process) and that the software version is Allinea Forge 5.0.1.

# Zooming in on the IO portion of our run

The screenshot displays the Allinea Forge 5.0.1 interface. At the top, the window title is "cpi\_1p\_1t\_2015-04-29\_07-54.map - Allinea MAP - Allinea Forge 5.0.1 [Trial Version]". The menu bar includes File, Edit, View, Window, and Help. Below the menu bar, the status bar indicates "Profiled: cpi on 1 process Started: Wed Apr 29 07:54:35 2015 Runtime: 72s" and a "Hide Metrics..." button.

The main activity section shows three metrics:

- Main thread activity:** A horizontal bar chart showing a green bar for most of the duration and a small orange bar at the end.
- CPU floating-point (%):** A line graph showing a high percentage of CPU usage. Summary statistics: 0.0 - 100.0 (90.4 avg), 0.0 - 30 (2.1 avg).
- Memory usage (kB):** A line graph showing memory usage. Summary statistics: 11,571 - 11,895 (11,583 avg), 11,842 - 11,895 (11,889 avg).

At the bottom of the metrics section, a summary bar reads: "07:55:44-07:55:46 (2.365s, 3.4% of total): Main thread compute 0 %, File I/O 100 % | CPU floating-point 2.1 %; Memory usage 11,889 kB;". Buttons for "Metrics" and "Select All" are present.

The code editor shows the source file "cpi.c" with the following code:

```
36
37 for(i=0; i<N; ++i)
38 {
39     f = fopen("cpi.tmp", "w");
40     fwrite(buffer, len, sizeof(char), f);
41     fclose(f);
42 }
43 unlink("cpi.tmp");
44
```

The "Main Thread Stacks" panel is active, showing a table of function calls:

Total core time	Function(s) on line	Source	Position
85.3%	main	{	cpi.c:47
5.9%	do_io	do_io(); /* unnecessary i/o for illustration pur...	cpi.c:76
8.8%	fgets	f = fopen("cpi.tmp", "w");	cpi.c:39
	_IO_file_fopen	fgets (no debug info)	
	fclose	fclose(f);	cpi.c:41

At the bottom, it says "Showing data from 34 samples taken over 1 process (34 per process)". The bottom right corner shows "Allinea Forge 5.0.1 Main Thread View".

# Always be thinking about vectorization

The screenshot displays the Allinea MAP interface for a process named 'cpi'. The main thread activity is shown as a green bar at the top. The CPU floating-point usage is 90.4% (avg), while the CPU fp vector usage is 0.0% (avg). The memory usage is 11,583 kB (avg). The summary bar at the bottom indicates: 07:54:35-07:55:46 (71.569s): Main thread compute 96 %, File I/O 4 % | CPU floating-point 90.4 %; CPU fp vector 0.0 %; Memory usage 11,583 k. The code editor shows the following C code:

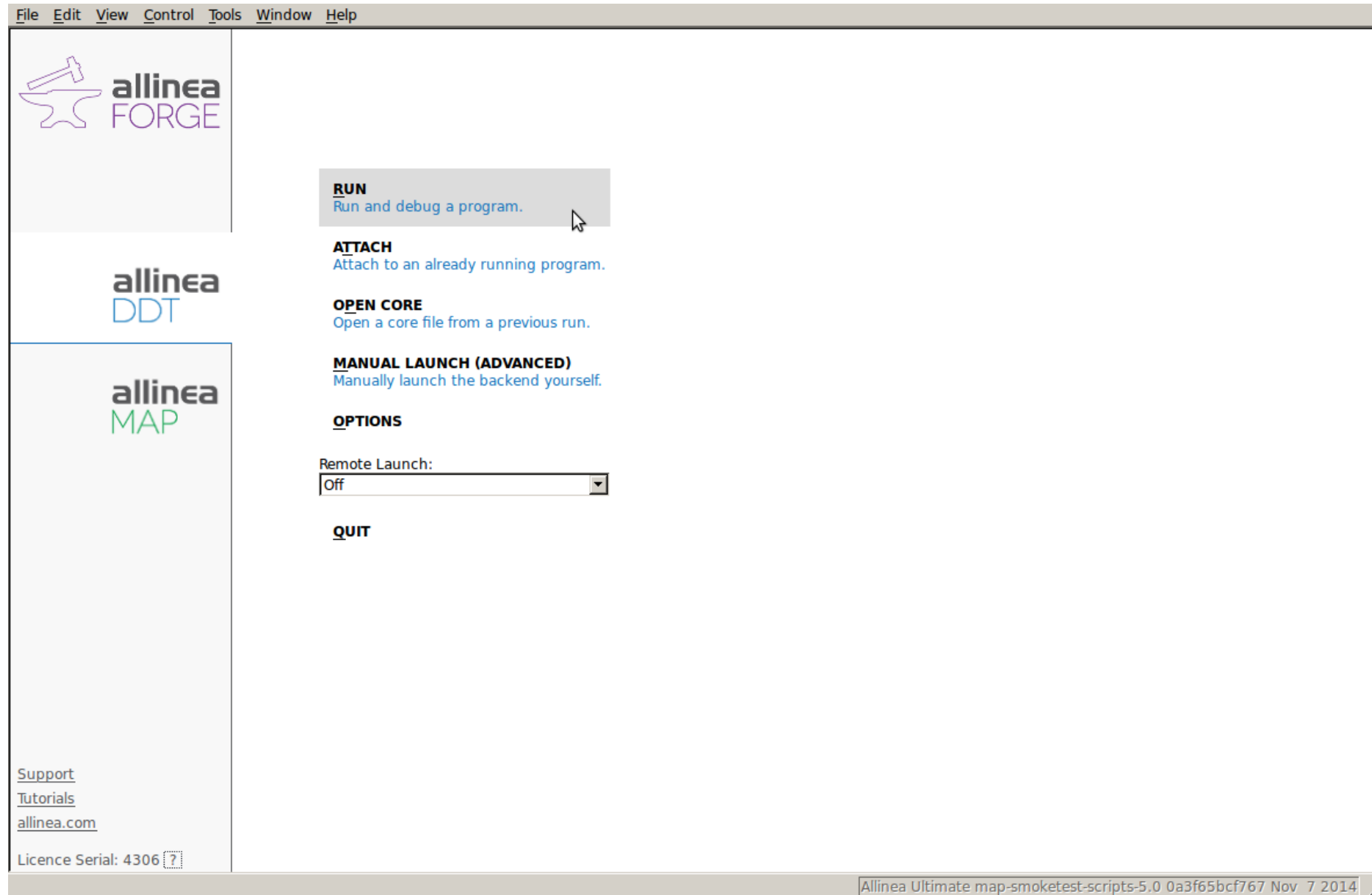
```
62  fflush(stdout);
63  for (i = 1; i <= n; ++i)
64  {
65      x = h * ((double)i - 0.5);
66      sum += f(x);
67  }
68
69  pi = h * sum;
70  printf("pi is approximately %16f. Error is %16f\n");
```

The Main Thread Stacks view shows the following data:

Total core time	Function(s) on line	Source	Position
74.6%	main	{	cpi.c:47
21.8%	f	sum += f(x);	cpi.c:66
	do_io	x = h * ((double)i - 0.5);	cpi.c:65
	fgets	do_io(); /* unnecessary i/o for illustration pur...	cpi.c:76
		f = fopen("cpi.tmp", "w");	cpi.c:39

Showing data from 1,000 samples taken over 1 process (1000 per process) Alinea Forge 5.0.1 Main Thread View

# While still connected to the server we can switch to the debugger



The screenshot displays the Allinea Forge IDE interface. The top menu bar includes File, Edit, View, Control, Tools, Window, and Help. The left sidebar contains the Allinea Forge logo, Allinea DDT, and Allinea MAP. The main workspace area shows the following options:

- RUN**  
Run and debug a program.
- ATTACH**  
Attach to an already running program.
- OPEN CORE**  
Open a core file from a previous run.
- MANUAL LAUNCH (ADVANCED)**  
Manually launch the backend yourself.
- OPTIONS**  
Remote Launch: Off
- QUIT**

At the bottom left, there are links for Support, Tutorials, and allinea.com, along with a Licence Serial: 4306 [?]. The bottom right corner shows the status bar: Allinea Ultimate map-smoketest-scripts-5.0 0a3f65bcf767 Nov 7 2014.





# Adding some OpenMP pragmas to multithread

Profiled: [cpi-openmp](#) on 1 process, [4 cores \(4 per process\)](#) Started: Wed Apr 29 08:01:51 2015 Runtime: 20s Hide Metrics...

**Application activity**

**CPU floating-point (%)**  
0.0 - 100.0 (85.6 avg)

**Memory usage (kB)**  
11,612 - 12,116 (11,897 avg)

08:01:51-08:02:10 (19.699s): Main thread compute 0 %, OpenMP 90 %, OpenMP overhead 10 % | CPU floating-point 85.6 %; Memory usage 11,897 kB; Metrics Select All

```
64 #pragma omp parallel for reduction(+:sum) private(x)
65     for (i = 1; i <= n; ++i)
66     {
67         x = h * ((double)i - 0.5);
68         sum += f(x);
69     }
70
71     pi = h * sum;
72     printf("pi is approximately %.16f, Error is %.16f\n",
73           pi, fabs(pi - PI25DT));
74
75     printf("writing to disk...\n");
76     fflush(stdout);
```

Input/Output | Project Files | **OpenMP Stacks** | OpenMP Regions | Functions

OpenMP Stacks

Total core time	Overhead	Function(s) on line	Source	Position
		[-] cpi-openmp [program]		
		[-] main	{	cpi-openmp.c:47
		[-] main [OpenMP region 0]	#pragma omp parallel for reduction(+:sum) private(x)	cpi-openmp.c:64
68.9%		[-] f	sum += f(x);	cpi-openmp.c:68
20.8%		[-] f	x = h * ((double)i - 0.5);	cpi-openmp.c:67
<0.1%	<0.1%	[-] 2 others		
0.1%		[-] 2 others		
0.0%	0.0%	[-] [OpenMP worker threads]		
		[-] omp_get_num_procs	OMP_backwait (no debug info)	

Showing data from 950 samples taken over 1 process (950 per process) Allinea Forge 5.0.1 OpenMP View



# Adding MPI support and submitting to and HPC batch system

The screenshot shows a 'Run' dialog box with the following configuration:

- Application:** /home/bpaisley/ownCloud/bpaisley/src/pi/cpi-mpi
- Arguments:** (empty)
- stdjn file:** (empty)
- Working Directory:** (empty)
- MPI:** 2 processes, 2 ppn, Open MPI (Compatibility)
  - Number of Processes: 2
  - Processes per Node: 2
  - Implementation: Open MPI (Compatibility) [Change...]
  - mpirun arguments: (empty)
- OpenMP:** 8 threads
  - Number of OpenMP threads: 8
- Submit to Queue:** (unchecked) [Configure... Parameters...]
- Environment Variables:** none

Buttons at the bottom: Help, Options, Run, Cancel.

# Analyzing our OpenMP + MPI results

The screenshot displays the Allinea Forge 5.0.1 interface for analyzing the performance of a C program named `cpi-mpi`. The window title is `cpi-mpi_4p_2t_2015-04-29_08-10.map - Allinea MAP - Allinea Forge 5.0.1 [Trial Version]`. The main panel shows application activity, CPU floating-point usage, and memory usage over time. A summary bar indicates that the program was profiled on 4 processes, 4 cores (1 per process), starting on Wed Apr 29 08:10:53 2015, with a runtime of 5s. The summary also shows that OpenMP accounts for 56% of the time, MPI for 34%, and OpenMP overhead for 10%. The CPU floating-point usage is 51.5%, and memory usage is 15,135 kB.

The code editor shows the following C code snippet:

```
77 MPI_Barrier(MPI_COMM_WORLD);
78 if (myid == 0) {...}
83
84 /* A slightly better approach starts from large i and works back */
85 #pragma omp parallel for reduction(+:sum) private(x)
86 for (i = myid + 1; i <= n; i += numprocs)
87 {
88     x = h * ((double)i - 0.5);
89     sum += f(x);
90 }
mpi = h * sum;
```

The OpenMP Stacks panel shows the following data:

Total core time	MPI	Overhead	Function(s) on line	Source	Position
55.9%			cpi-mpi [program]		
			main	{	cpi-mpi.c:48
32.8%	32.8%		main [OpenMP region 0]	#pragma omp parallel...	cpi-mpi.c:84
			MPI_Finalize	MPI_Finalize();	cpi-mpi.c:109
1.0%	1.0%		MPI_Reduce	MPI_Reduce(&mypi, &p...	cpi-mpi.c:97

Showing data from 796 samples taken over 4 processes (199 per process)

Allinea Forge 5.0.1 OpenMP View

# Why all this synchronization?

The screenshot displays the Allinea Forge 5.0.1 interface for a profiled application. The top section shows application activity and various performance metrics. The 'MPI call duration (ms)' chart shows a significant increase in duration over time, indicating synchronization. The 'OpenMP Stacks' table at the bottom provides a detailed breakdown of the time spent in different MPI and OpenMP regions.

Profiled: [cpi-mpi](#) on 4 processes, [4 cores \(1 per process\)](#) Started: Wed Apr 29 08:10:53 2015 Runtime: **5s** [Hide Metrics...](#)

**Application activity**

**CPU floating-point (%)**

0.0	-	100.0	( 51.5 avg )
0.0	-	100.0	( 2.8 avg )

**Memory usage (kB)**

15,020	-	15,319	( 15,135 avg )
15,020	-	15,319	( 15,270 avg )

**MPI call duration (ms)**

0	-	2,103	( 324.7 avg )
0	-	2,044	( 700.7 avg )

08:10:55-08:10:57 (2.008s, 44.2% of total): OpenMP **3** %, MPI **74** %, OpenMP overhead **23** % | CPU floating-point **2.8** %; Memr [Metrics...](#) [Select All](#)

```
105         pi, fabs(pi - PI25DT));
106         fflush(stdout);
107     }
108
72.2% 109     MPI_Finalize();
110     return 0;
111 }
112
```

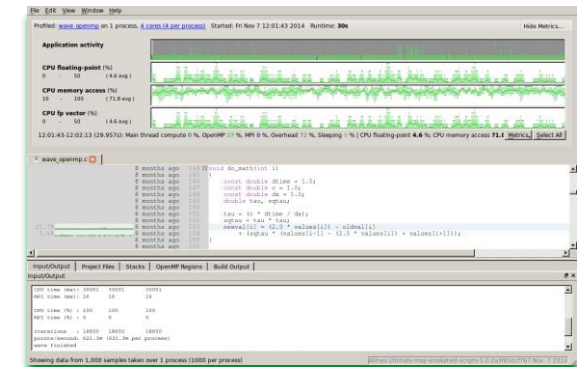
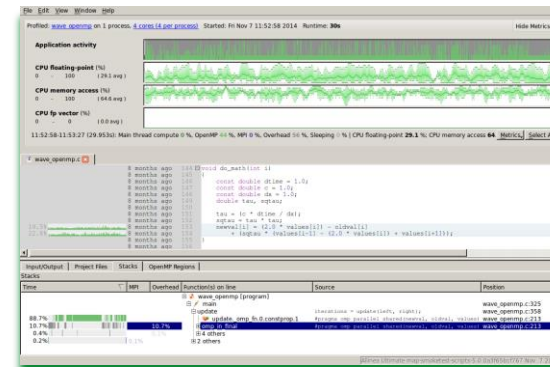
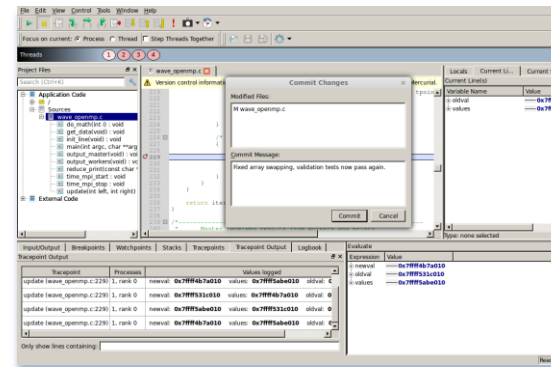
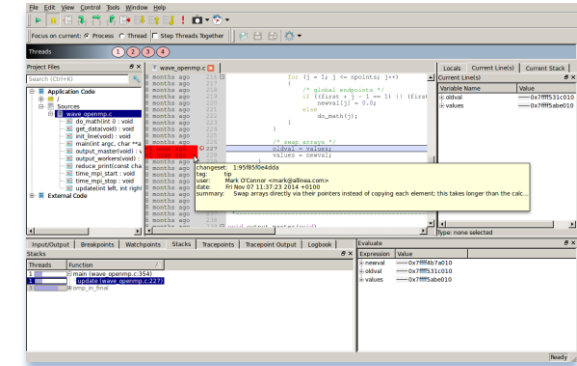
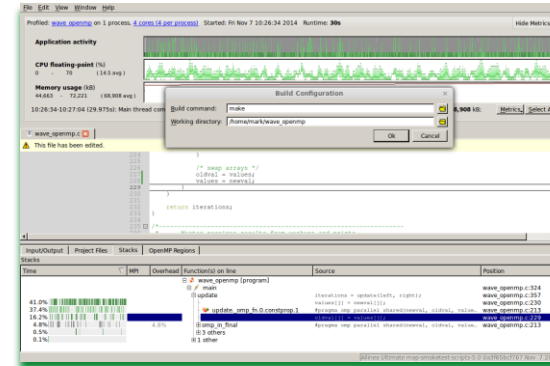
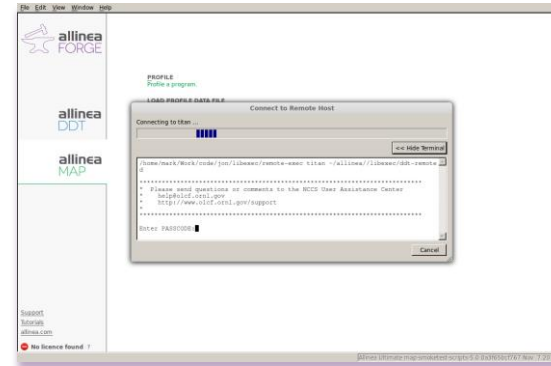
Input/Output | Project Files | **OpenMP Stacks** | OpenMP Regions | Functions

OpenMP Stacks

Total core time	MPI	Overhead	Function(s) on line	Source	Position
72.2%	72.2%		main	{	cpi-mpi.c:48
2.8%			MPI_Finalize	MPI_Finalize(...	cpi-mpi.c:109
2.3%		2.3%	main [OpenMP region 0]	#pragma omp p...	cpi-mpi.c:84
			MPI_Reduce	MPI_Reduce (&m...	cpi-mpi.c:97

Showing data from 352 samples taken over 4 processes (88 per process) Allinea Forge 5.0.1 OpenMP View

# The workflow of pi from serial to HPC



<https://www.allinea.com/products/downloads/free-trial>

