Debugging in Heterogeneous Environments with TotalView

ECMWF HPC Workshop 30th October 2014



Accelerating Great Code

Agenda

- Introduction
- Challenges
- TotalView overview
- Advanced features
- Current work and future plans

Introduction

Rogue Wave



- Customers
 - 3,000+ in 57 countries
 - Financial services, telecoms, oil and gas, government and aerospace, research and academic

- History
 - Founded: 1989
 - Portfolio company of Audax Group
 - Acquisitions:
 - Visual Numerics: 2009
 - TotalView Technologies: 2010
 - ILOG Visualization for C++ : 2012
 - OpenLogic : 2013
 - Klocwork : 2013
 - ILOG Visualization for Java: 2014
- Global Locations
 - HQ: Boulder, CO
 - NA: Houston, TX; Corvallis, OR; Natick MA
 - EMEA: France, Germany, UK
 - APAC: Japan

Challenges of developing for heterogeneous environments

Challenges

- Number of CPU cores increasing but clock speed is static or decreasing
- How to program accelerators
 - Lower level languages (OpenCL, CUDA)
 - Directives based (OpenACC, OpenMP)
- New algorithms or programming models may be needed
- Data sizes increasing exponentially
- Memory is increasingly important
- Power consumption and constraints

How does Rogue Wave help?

TotalView debugger

- Troubleshooting and analysis tool
 - Visibility into applications
 - Control over applications
- Scalability
- Usability
- Support for HPC platforms and languages

TotalView Overview

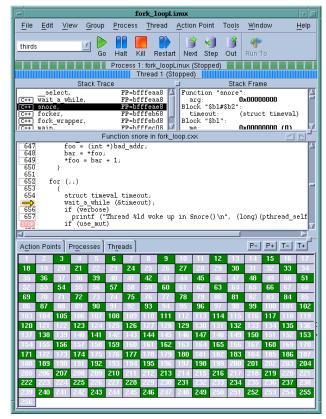
What is TotalView[®]?

Application Analysis and Debugging Tool: Code Confidently

- Debug and Analyse C/C++ and Fortran on Linux[™], Unix or Mac OS X
- Laptops to supercomputers
- Makes developing, maintaining, and supporting critical apps easier and less risky

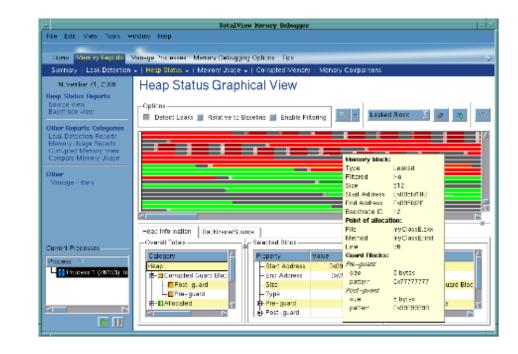
Major Features

- Easy to learn graphical user interface with data visualization
- Parallel Debugging
 - MPI, Pthreads, OpenMP™
 - CUDA[™], OpenACC[®], and Intel[®] Xeon Phi[™] coprocessor
- Low tool overhead resource usage
- Includes a Remote Display Client which frees you to work from anywhere
- Memory Debugging with MemoryScape™
- Deterministic Replay Capability Included on Linux/x86-64
- Non-interactive Batch Debugging with TVScript and the CLI
- TTF & C++View to transform user defined objects



MemoryScape®

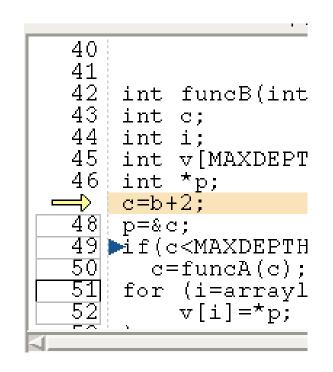
- Runtime Memory Analysis : Eliminate Memory Errors
 - Detects memory leaks *before* they are a problem
 - Explore heap memory usage with powerful analytical tools
 - Use for validation as part of a quality software development process
- Major Features
 - Included in TotalView, or Standalone
 - Detects
 - Malloc API misuse
 - Memory leaks
 - Buffer overflows
 - Supports
 - C, C++, Fortran
 - Linux, Unix, and Mac OS X
 - Intel® Xeon Phi™
 - MPI, pthreads, OMP, and remote apps
 - Low runtime overhead
 - Easy to use
 - Works with vendor libraries
 - No recompilation or instrumentation



Deterministic Replay Debugging

<u>File E</u> dit <u>V</u> iew	<u>G</u> roup	<u>P</u> rocess	Thread	Action	n Point	<u>D</u> ebug	Too <u>l</u> s	<u>W</u> indow			<u>H</u> elp
Group (Control)								d GoBack			

- Reverse Debugging: Radically simplify your debugging
 - Captures and Deterministically Replays Execution
 - Not just "checkpoint and restart"
 - Eliminate the Restart Cycle and Hard-to-Reproduce Bugs
 - Step Back and Forward by Function, Line, or Instruction
- Specifications
 - A feature included in TotalView on Linux x86 and x86-64
 - No recompilation or instrumentation
 - Explore data and state in the past just like in a live process, including C++View transformations
 - Replay on Demand: enable it when you want it
 - Supports MPI on Ethernet, Infiniband, Cray XE Gemini
 - Supports Pthreads, and OpenMP
 - New: Save / Load Replay Information (CLI only)



Rogue Wave

Cambridge Study



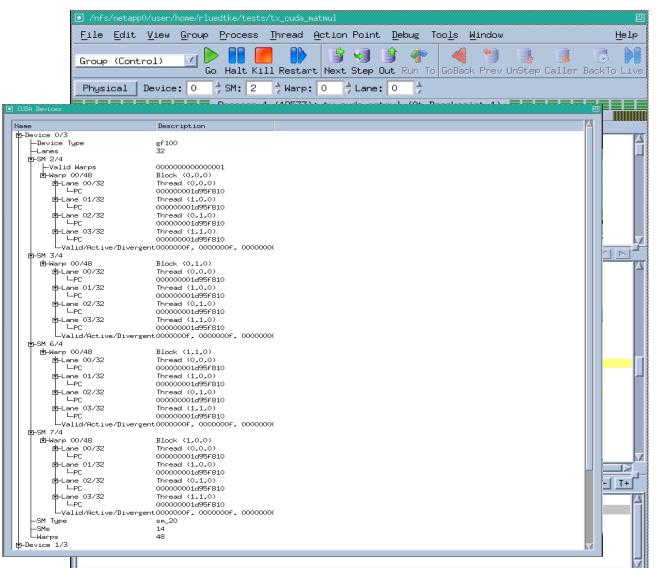
- Survey conducted by the Judge Business School at Cambridge University concluded that Reverse Debuggers allow users, on average, to spend 13% less of their programming time debugging.
 - Programming was 50% of total work week on average
 - Debugging was 50% of programming time without reverse debugging
 - Debugging was 37% of programming time with reverse debugging
 - That frees up 130 hours (>3 work weeks, 6.5% total time) per developer per year for design and new feature development
- The survey looked at total value (salaries & overhead) of debugging as a task and they determined that this savings could, across the whole world economy, be work \$41 billion in increased productivity.
 - The productivity improvement should be worth \$2,500 per developer per year (salary only) or \$5,000 per year with overhead.
- <u>http://www.roguewave.com/company/news-events/press-releases/2013/university-of-cambridge-reverse-debugging-study.aspx</u>

TotalView for the NVIDIA[®] GPU Accelerator

File Edit View Group Process Thread Action Point Debug Tools Window	w <u>H</u> elp
Group (Control) Go Halt Kill Restart Next Step Out Run To GoBack Prev Process 1 (20242): ty_cuda_matmul (Stopped) Thread -1 (<<<(1,1),(1,1,0)>>>): @TEMP@CUDA@tx_cuda_matmul.e4974cfd (S	
Stack Trace 🃮 Stack I	Frame
A: (Mathematical Mathematical	/3 of 1/32/32 rix const @parameter) rix const @parameter) po0001 (1) rix @local) address: %f2> pidemas: %e21
Function MatMulKernel in tx_cuda_matmul.cu	
<pre>%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%</pre>	Year Year <t< td=""></t<>
104 // and accumulate the results	
Action Points Processes Threads Phy 0 4 0 4 11 3 4 1.1 (47567291993984) T at 0x2b431f97d429 at 0x2b4431f97d429 at 0x2b4431f97d429	

- NVIDIA CUDA 6.5
 - With support for Unified Memory
- Features and capabilities include
 - Support for dynamic parallelism
 - Support for MPI based clusters and multi-card configurations
 - Flexible Display and Navigation on the CUDA device
 - Physical (device, SM, Warp, Lane)
 - Logical (Grid, Block) tuples
 - CUDA device window reveals what is running where
 - Support for types and separate memory address spaces
 - Leverages CUDA memcheck

Displaying NVIDIA GPU Device Information





TotalView for OpenACC

File Edit View Group Process Thread Action	Point Debug Tools Window Help
Group (Control)	p Out Run To Record GoBack Firey Unstep Caller BackTo Live
Physical Device: 0 7 SM 0 7 Warp 0	🖞 Lane: 0 🦸
	run <man> 0 (Stopped) IP@CUDA@ man.54051177 (Stopped) <trace trap=""></trace></man>
Stack Trace	J Stack Frame
(30) test_openacc_Sck_L11_1. FP=fffca0 [A	Function *nacc_sck_Ll1_1<<<<(782,1,1),(128,1,1)>>: 1 Device: D/1 SM/MP/LN: 0/0/0 of 16/48/32 \$\$arg_ptr_acc_stl6: 853080320 \$\$arg_ptr_acc_stl6: 853080320 \$\$arg_ptr_acc_stl6: 85308002003000 \$\$arg_ptr_acc_stl8: 8594128896 \$\$arg_ptr_acc_stl8: 859412896 \$\$arg_ptr_acc_stl8:
Function test openance	sck_L11_1 in man.190
1 PROGRAM test openacc 2 IMPLICIT NOME 3 IMPLOR PARAMETER :: M- 4 INTEGER :: A(M), b(M), c(M) 5 INTEGER :: A(M), b(M), c(M) 5 INTEGER :: A(M), b(M), c(M) 6 !!S For simple cases, use parall 8 !!S parallel and loop 9 !!S set a.b.c 10 !S acc parallel loop 11 !Sacc parallel loop 12 D0 j - 1.M 13 a(j) = j 14 ENDDO 15 ENDDO 16 ENDDO 17 !Sacc end parallel loop 18 !!S Set b. copy it to host 20 !Sacc parallel copyrout(b) 11 !Sacc end parallel loop 22 D0 j = 1.M 23 ENDDO 24 ENDDO 25 !Sacc end parallel 26 !Sacc end parallel 27 !Sacc parallel loop 28 !!S Set c. copy it to host 29 !Sacc parallel copyout(c)) ted
30 !\$acc loop 31 D0 j = 1. M 33 c(j) = -j 33 EMDDO 34 !\$acc end loop	
35 Isacc end narallel	N/
Action Points Processes Threads	P- P+ T- T+
1 man.f90#14 test_openacc_Sck_L11_1+0 2 man.f90#23 test_openacc_Sck_L20_2+0 3 man.f90#32 test_openacc_Sck_L29_3+0	x118



- Step host & device
- View variables
- Set breakpoints

Compatibility with Cray CCE 8 OpenACC

TotalView for the Intel[®] Xeon Phi[™] coprocessor

Supports All Major Intel Xeon Phi Coprocessor Configurations

- Native Mode
 - With or without MPI
- Offload Directives
 - Incremental adoption, similar to GPU
- Symmetric Mode
 - Host and Coprocessor
- Multi-device, Multi-node
- Clusters

User Interface

- MPI Debugging Features
 - Process Control, View Across, Shared Breakpoints
- Heterogeneous Debugging
 - Debug Both Xeon and Intel Xeon Phi Processes

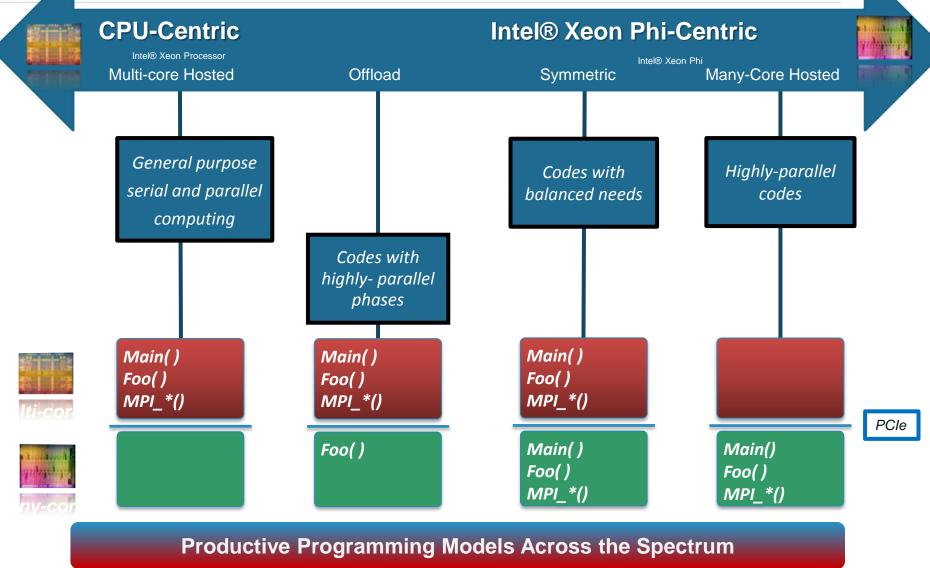
Memory Debugging

• Both native and symmetric mode

File Edit	View Tools	Window	Help	
ID / F	ank Host	Status	Description	
⊡- 1	<local></local>	R	/opt/intel/composerxe/Sample	
. 1.1	<local></local>	R	in main	
1.2	<local></local>	R	inpoll	
1.3	<local></local>	R	inpoll	
1.4	<local></local>	R	in pthread_cond_wait	
ė- <mark>2</mark>	192.168.1	. 1(M	/tmp/coi_procs/1/5856/offloa	
2.1	192.168.1	. 1(R	in sem_wait	
2.2	192.168.1		in compute07	
- 2.3	192.168.1		inpoll	
I 2. 4	192.168.1	108	in pthread_cond_wait	
	Process Thread 2		ug Tools Window	Не
oup (Control) 🛛 🔟			* 🥘 🖣 " 🐘	
			un To Record GoBack Prev UnStep Caller :	BackTo Liv
	Thread 2 (60192.168.1.100) 139985823807232)	: offload_main (Mixed)	
Sta	ck Trace	<u>i</u>	Stack Frame	
] compute07,] L_sample07_76par.		4d24f0 🔺 Funct.	on "compute07":	
offload_entry_sa	mpleCO7_c_76sampleO7	. FP=7f size		00000 (109
ZN170ffloadDescrip	ptor7offloadEjPPvS0	tSO_t, Local	variables:	
 _COISinkPipe::RunFi _COISinkPipe::Proce 	unction, FP=7f50fd essMessages, FP=7f50	4d2dc0 i: Ed4d2e1	0x00000010 (16)	
+] _COISinkPipe::Three	adProc. FP=7f50fd	4d2e20 Periot	ers for the frame:	
<pre>start_thread, clone,</pre>	FP=7f50fd FP=7f50fd	4d2f30 4d2f38	%rax: 0x7f50fd4d2754 (139985823803220)	
		NULL CO	Vach++ 0+00000010 (1C)	
			%rcx: 0x7f50fd4d2754 (139985823803220)	
		compute07 in sam	leC07.c	
30 for (i=0; i <s; 31 {</s; 	: i++)			
array1[i]	= p[i];			
93 } 94				
TTM Redect MTC				
7 #else				
<pre>78 retval = 0;</pre>				
99 #endif 00				
	f array initializatio	on was done on t	navat.	
01 // Return 1 if			al Sec	
01 // Return 1 if 02 return retval;	:		ai get	
01 // Return 1 if 02 return retval; 03 }	:	8 100-100 To		
01 // Return 1 if 02 return retval; 03 } 04 05attribute_((tag	:	oute07(int* out,		
)1 // Return 1 if)2 return retval;)3 } 04)5attribute((tai)6 { 17 int i;	; rget(mic))) void com	oute07(int* out,		
11 // Return 1 ii 12 return retval; 13 } 04	; rget(mic))) void com	oute07(int* out,		
<pre>)1</pre>	; rget(mic))) void com ize; i++)	pute07(int∗ out,		
<pre>11 // Return 1 if 22 return retval; 33 } 43 45 attribute((tan 6 { 77 int i; 80 for (i=0; i<s; 10 { 11 gut[i] = a</s; </pre>	; rget(mic))) void com	oute07(int* out,		
<pre>11 // Return 1 if 12 return retval; 33 } 34</pre>	; rget(mic))) void com ize; i++)	oute07(int* out,	int size)	
<pre>11 // Return 1 if 12 return retval; 33 } 34</pre>	; rget(mic))) void com ize; i++)	pute07(int* out,		
<pre>11 // Return 1 if 12 return retval; 33 } 34</pre>	; rget(mic))) void com ize; i++)	oute07(int* out,	int size)	
<pre>// Return 1 ii return retval; 33 } 56attribute((tan 6 [77 int i; 37 for (i=0; i<s; 1 { 0 (i=0; i<s; 28] 0 (i=0; i<s; 29] 0 (i=0; i<s; 20] 0 (i=0; i<s; 20] 1 (i=0; i<s; (i="0;" (i<="" 1="" 20=""]="" i<s;="" td=""><td>; rget(mic))) void com ize: i++) array1[i]*2;</td><td>oute07(int* out,</td><td>int size)</td><td>(≥ ≥+_ T+ T+</td></s;></s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </s; </pre>	; rget(mic))) void com ize: i++) array1[i]*2;	oute07(int* out,	int size)	(≥ ≥+_ T+ T+
11 // Return 1 ii 12 return retval; 33 3 56 attribute((tai 66 10 for (i=0; i <s; 10 11 ; 12 12 ; 13 14 //</s; 	; rggt(mic))) wold com ize; i++) array[[i]*2;] Threads] 2 in sem wait	oute07(int* out,	int size)	
11 // Return 1 il 21 return retval; 23] 24 _attribute((tar 26 [int is 26 f on (i=0; i (s. 20 [out[i] = c 23] 24	; rget(mic))) void comp array1[i]*2; <u>Threads</u> ; in sem_wait ; in sem_wait	oute07(int* out,	int size)	
11 // Return 1 ii 12 return retval, 33 3 36	; rget(mic))) wold com ize; i++) array[[]*2; Threads Threads in computeSy inpoll		int size)	
01 // Return 1 il 12 return retval, 03 j 04 j 05 _ettribute((tan 0.5 m)) 06 [07 int i; 08 [09 for (i=0; i≤s; 11 [13] 13] 13] 13] 14 // 13] 14 // 13] 14 // 13] 14 // 13] 14 // 13] 14 // 13] 14 // 13] 14 // 13] 14 // 15] 16] 17]	; rget(mic))) wold com ize; i++) array[[]*2; Threads Threads in compute() in poll		int size)	
01 // Return 1 ii 11 return retval, 03 j 04 retvinte((tai 06 [07 int i; 08 [09 [01 [02 [03 [04 [05 [06 [07 int i; 08 [04 [04 [05 [04 [04 [05 [04 [05 [06 [07 [08 [04 [04 [04 [05 [05 [05 [06 [; rget(mic))) wold com ize; i++) array[[]*2; Threads Threads in computeSy inpoll		int size)	
11 // Return 1 ii 12 return retval, 33 3 36	; rget(mic))) wold com ize; i++) array[[]*2; Threads Threads in computeSy inpoll		int size)	
11 // Return 1 ii 12 return retval, 33 3 36	; rget(mic))) wold com ize; i++) array[[]*2; Threads Threads in computeSy inpoll		int size)	



Spectrum of Intel Xeon Phi Execution Models

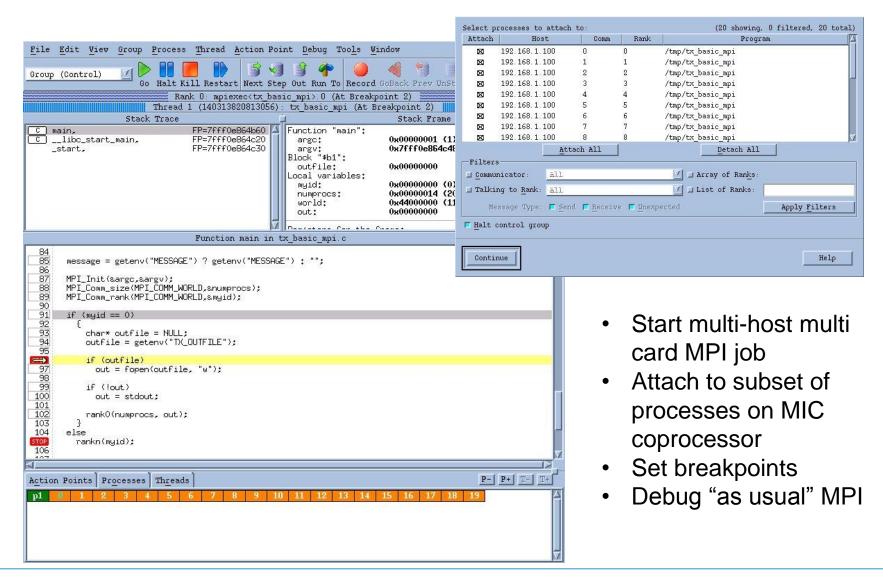


Debugging Intel Xeon Phi Applications with Offloaded Code

	<u>F</u> ile <u>E</u> dit <u>V</u> iew	Too <u>l</u> s <u>W</u> indow		Hel	Lp
Vaan aida	ID / Ra		Status	Description	Vaan Dhiaida
Xeon side	₽-1	<local></local>	Т	/opt/intel/composerxe/Samples/en_US	Xeon Phi side
	-1.1	<local></local>	Т	in pthread_cond_wait	
	- 1.2	<local></local>	Т	inpoll	
File Edit View Group Process Thread Action Point Debug Tools	Wi -1.4	<local> <local></local></local>	1	inpoll inpoll	View Group Process Thread Action Point Debug Tools Window Help
	-1.5	<local></local>	T	in phread cond wait	
Group (Control) 🗹 🕨 📕 🛄 📕 🧊 🍯 🤎	A 10	192.168.1.100	M	/tmp/coi procs/1/7976/offload main	777 01) 🔽 📂 🚺 📕 💔 🍯 🌱 👹 🖤 💗 🔍 🧐 🦓 湯
Go Halt Kill Restart Next Step Out Run To Reco	cd _ 2.1	192, 168, 1, 100	R	in sem wait	Go Halt Kill Restart Next Step Out Run To Record GoBack Prev UnStep Calle
Process 1 (31634): intro sampleC.out (Stoppe	d) – 2.2	192.168.1.100	B6	in L_sampleO8_51par_loop1_2_19	Process 2 (79760192.168.1.100): offload main (Mixed)
Thread 1 (140091609065248) (Stopped)	- 2.3	192.168.1.100	R	inpoll	Thread 2 (139773936764672) (At Breakpoint 6)
Stack Trace	St: - 2.4	192.168.1.100	R	in pthread_cond_wait	Stack Trace 📮 Stack Frame
pthread_cond_wait, FP=7fff653f38f0 🔼 Function "sample	08" - 2.5	192.168.1.100	R	in pthread_cond_timedwait	=08_51par_loop1_2_19, FP=7f1fa7d96cf 🖾 Function "L_sample08_51par_loop1_2_19":
C++ TaskScheduler::WaitForEvent_signal, FP=7fff65 No parameters.	- 2.6	192.168.1.100	B6	in L_sample08_51_par_loop1_2_19	hvoke_microtask, FP=7f1fa7d96d30 No parameters.
C++ COIEventWait, FP=7fff653f39b0 Local variables: _ZN170ffloadDescriptor14offload_finishEv, FP= pi:	- 2.7 - 2.8	192.168.1.100	B6	in L_sample08_51_par_loop1_2_19	<pre>hvoke_task_func, FP=7f1fa7d96d70 Local variables: prk_call, FP=7f1fa7d97360 count: 0x00002710 (10000)</pre>
Descriptor7offloadEPKcbP7VarDesciPPviS4_, count:	0: 2.9	192.168.1.100 192.168.1.100	B6 B6	in L_sampleO8_51par_loop1_2_19 in L_sampleO8_51par_loop1_2_19	ork_call, FP=7f1fa7d97470 pi: 0
offload_offload, FP=7fff653f3ae0 i:	0 2.10	192.168.1.100	B6	in L_sample08_51_par_100p1_2_19 in L_sample08_51 par_loop1_2_19	3, FP=7f1fa7d97700 t: 5e-05
C sample08, FP=7fff653f3dc0 i: C main, FP=7fff653f3e20 pi:	0:	192.168.1.100	B6	in L sample08 51 par loop1 2 19	3, FP=7f1fa7d97990 i: 0x00000000 (0)
C main, FP=7fff653f3e20 pi: libc_start_main, FP=7fff653f3ee0 pi:	2 - 2.11	192.168.1.100	B6	in L sample08 51 par loop1 2 19	<pre>?loadDescriptor7offloadEjPPvS0_tS0_t, {Pipe::RunFunction, FP=7f1fa7d97dc0 Registers for the frame:</pre>
start, FP=7fff653f3ef0 t:	0 - 2.13	192.168.1.100	B6	in L_sample08_51_par_loop1_2_19	kPipe::ProcessMessages, FP=7f1fa7d97e1
t:	1 - 2.14	192.168.1.100	B6	in L sampleO8 51 par loop1 2 19	(Pipe::ThreadProc. FP=7f1fa7d97e20 %rax: 0x00000341 (833)
Function sample08 in sample08.c	- 2.15	192.168.1.100	B6	in L_sample08_51par_loop1_2_19	Function L sample08 51 par loop1 2 19 in sample08.c
	2.16	192.168.1.100	B6	in L_sampleO8_51par_loop1_2_19	
38 39 // Sample 08					ectively, this is heterogeneous OpenMP // isample08()
<pre>40 // This sample demonstrates how #pragma offload placed in front. 41 // an OpenMP construct enables OpenMP on the target 42 // 43 // Effectively, this is heterogeneous OpenMP 44 void sample08() 45 { 46 float pi = 0.0f; 47 int count = 10000; 48 int 1; 49 49 49 49 49 49 40 51 #pragma offload target (mic) 51 #pragma one parallel for reduction(*;pi) 52 for (i=0; iccount; i++) 53 { 55 pi += 4.0f/(1.0f+t*t); 56 } 57 pi /= count; 59 50 if (fabs(pi-3.14f) <= 0.01f) 51 51 52 53 54 55 55 55 55 55 55 55 55 55 55 55 55</pre>	o r			47 48 49 50 51 52 53 54 54 56 57 57 57 58 59 60 €1 #1fd	<pre>float pi = 0.0f; int count = 10000; int i; #pragma offload target (mic) #pragma omp parallel for reduction(+:pi) for (i=0; i(count; i++) float t = (float)((i+0.5f)/count); pi += 4.0F/(1.0f+t*t); } pi /= count; if (fabs(pi=3,14f) <= 0.01f) lef DEBUG printf("PASS Sample08 Pi = %f\n", pi); else printf("**** FAIL Sample08 Pi = %f\n", pi); if i ::::::::::::::::::::::::::::::::::</pre>
Action Points Processes Threads	P- P	+ T- T+		Action Poi	ints Processes Threads P- P+ T- T+
1.1 (140091609065248) T in pthread_cond_wait	and here				774009124800) R in sem wait
1.2 (140091596920592) T inpoll					773936764672) 86 in L_sample08_51_par_loop1_2_19
1.3 (140091586430736) T inpoll				2.3 (1397	773947401984) R inpoll
1.4 (140091575940880) T inpoll 1.5 (140091565442832) T in pthread_cond_wait					773955794688) R in pthread_cond_wait 773922080512) R in pthread_cond_timedwait
1,5 (1400513634420327) In pthread_cond_wait				2.6 (1397	773913687808) B6 in L_sample08_51_par_loop1_2_19
		_		2,7 (1397	773909489408) B6 in L sample08 51 par loop1 2 19
		NA .		2 Q /1797	7739052910091 🥵 in Leason 1e09 51 ner 100n1 2 19

One debugging session for MIC-accelerated code

Debugging Intel Xeon Phi MPI Applications



Coarray Fortran

Diving on CAF array y

			y-h	ello_image - 3	2.1	_	o x	1					
<u>F</u> ile	<u>E</u> dit	<u>V</u> iew	Tools	<u>W</u> indow			<u>H</u> elp						
2.1		4		ŧ	3 🗄 🛛 🐶 🔓 🗌	K 🔇 🛛		1					
Expre	Expression: y Address: 0x009caa40					0			Divin	n on C	ΔF array y		
	Slice: (;,;) Filter:								Diving on CAF array y				
	Type:	INTEG	ER*8(10)[*]			_		acros	s proc	cesses		
		Field		Value					y - hello_imag	e - 2.1	_ C	⊐ × Ì	
(1)[1]					000000000000000000000000000000000000000	<u>F</u> ile	<u>E</u> dit	View	Too <u>l</u> s <u>W</u> indow		Н	lelp	
(2)[1] (3)[1]				•	10000000000000000000000000000000000000] 2.1		2		=	= 🖡 🕹 🖉	. »	
(4)[1]				•	000000000000000000000000000000000000000	Expre	ssion:	v		Address:	0x009caa40		
(5)[1]					000000000000000000000000000000000000000		Slice:	(;,;)		Filter:			
(6)[1]				1 (0×00	000000000000000000000000000000000000000		Туре:	INTEG	ER*8(10)[*]	_			
(7)[1]				•	000000000000000000000000000000000000000		Fi	eld	Process		Value		
(8)[1]				•	000000000000000000000000000000000000000	(1)[1]			hello_image.1		1 (0x00000000000000000)		
(9)[1]				1 (1)X11	000000000000000000000000000000000000000	(1)[2]			hello_image.2		2 (0x00000000000000002)		
						(1)[3]			hello_image.3		3 (0×00000000000000003)		
						(1)[4]			hello_image.4		4 (0×0000000000000000)		
C		o rto	4 0 0			(2)[1]			hello_image.1		1 (0×00000000000000000)		
				Cray		(2)[2]			hello_image.2		2 (0×00000000000000002)		
p	latfo	rms	with	CCE		(2)[3]			hello_image.3		3 (0x00000000000000000)		
ľ						(2)[4]			hello_image.4		4 (0x00000000000000000)		
						(3)[1]			hello_image.1		1 (0x000000000000000000)		
						(3)[2]			hello_image.2		2 (0×000000000000000000002)	V.	

Advanced Features

Remote Display Client

- Offers users the ability to easily set up and operate a TotalView debug session that is running on another system
- Consists of two components
 - Client runs on local machine
 - Server runs on any system supported by TotalView and "invisibly" manages the secure connection between host and client
- Remote Display Client is available for:
 - Linux x86, x86-64
 - Windows XP, Vista, 7
 - Mac OS X

1. Enter the F Remote Hos	TotalView rechnologies Remote Host to ru	n vour d			
Remote Ho:		n vour d			
			ebug session:		
	st: jaguarpf.ccs.orni	.gov 💽	 User Nam 	e : max100 C	ommands:
2. As needed	l, enter hosts in ac	ccess or	der to reach the Re	emote Host:	
A	Host		Access By	Access Value	Commands
11		-	User Name		
2		-	User Name		
	-	session	on the Remote He	ost:	
TotalView	MemoryScape				
Path to	TotalView on Remo	ote Host:	totalview		
	Arguments for To	otalView:	-geometry 1400x12	:00	
Your	Executable (path 8	k name):			
Argu	uments for Your Exe	ecutable:			
Submit Job	to Batch Queueing	System:	PBS Pro		
4. Enter bat	tch submission set	ttings fo	r the Remote Host	:	
PBS S	ubmit Command:	qsub			
TotalView P	BS Script to Run:	tv_PBS.c	sh		
Additio	onal PBS Options:				
			Fnd Debu	a Session	
	1 1	Host Host Host Host Host Host Active Heat Active Host	Host Image: State S	Host Access By Image: Second Seco	User Name User Name User Name User Name 3. Enter settings for the debug session on the Remote Host : Total/lew MemoryScape Path to Total/View on Remote Host: total/view Arguments for Total/View: -geometry 1400x1200 Your Executable (path & name): Arguments for Your Executable: Submit Job to Batch Queueing System: PBS Pro 4. Enter batch submission settings for the Remote Host : PBS Submit Command: qsub Total/View PBS Script to Run: tv_PBS.csh

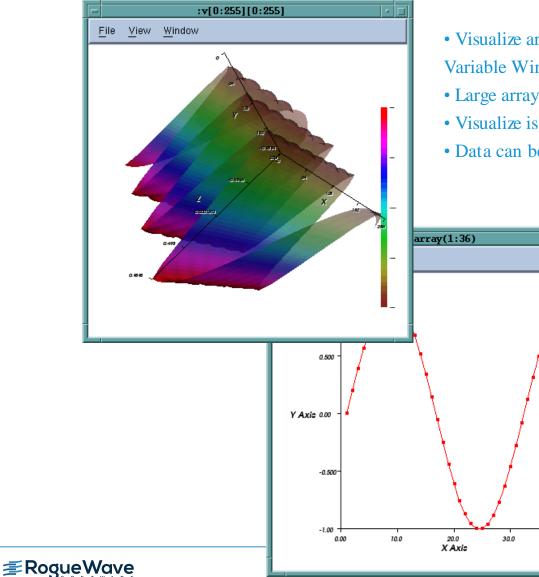
Profile ORNL/TotalView debug session is running...

Multi-dimensional array viewer

- See your arrays on a Grid display
- 2-D, 3-D, ...N-D
- Arbitrary slices
- Specify data
 representation
- Windowed data access – Fast

-		Array Vi	ewer: int_4D_	array[i][j][k][1]			• •
<u>F</u> ile									<u>H</u> elp
	on: int_4D_ar						Ту	pe: int[5][7][9][11]
Enter the array slice to display:									
	Dimension Start Index End Index Stride								
Row	[1]		0		4		1	Update	View
Column	[]]	Ā	0		6		1		
Select an	index for the	other din	nensions:						
[i]0	ic		[k] 0	÷ [1	.] 0 *	·	:4:1][0):6:1][0:0:	1][0:0:1]
	[j]:0		1		2		3		
[i]:0	0x00000000	(0)	0x00000064	(100)	0x00000)c8 (200)	0x00	000012c	(300)
1	0x000003e8	(1000)	0x0000044c	(1100)	0x000004	4ЬО (1200)	0x00	0000514	(1300)
2	0x000007d0	(2000)	0x00000834	(2100)	0x00000	398 (2200)	0x00	00008fc	(2300)
3	0x00000bb8	(3000)	0x00000c1c	(3100)	0x00000	(,		0000ce4	(3300)
4	0x00000fa0	(4000)	0x00001004	(4100)	0x00001	068 (4200)	0x00	00010cc	(4300)
					[

Visualizing Arrays



- Visualize array data using Tools > Visualize from the Variable Window
- Large arrays can be sliced down to a reasonable size first
- Visualize is a standalone program

40.0

- Data can be piped out to other visualization tools
 - Visualize allows to spin, zoom, etc.
 - Data is not updated with Variable Window; You must revisualize
 - \$visualize() is a directive in the expression system, and can be used in evaluation point expressions.

Debugging MPMD applications

totalview -args aprun -n 9 worker : -n 1 master

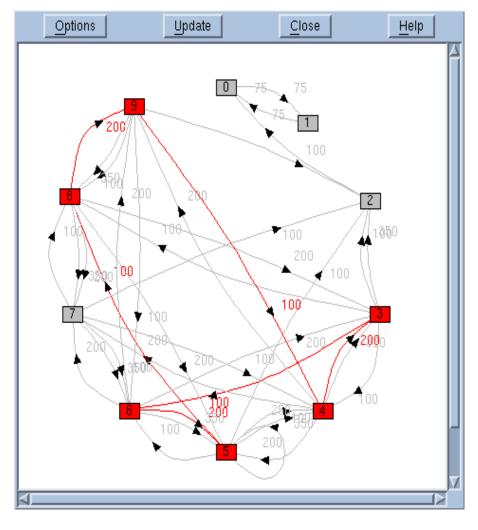
TotalVier	w 8.14.1-8			aprun <worker>.2</worker>
File Edi	it <u>V</u> iew Too <u>l</u> s <u>W</u> ir	ndow	<u>H</u> elp	File Edit View Group Process Thread Action Point Debug Tools Window Hel
IDA	Rank Host	Status	Description	
⊕ 1	<local></local>	R	aprun (2 active threads)	Group (Control) 📝 🕨 📕 🛃 🚺 📑 🧐 🦉 🍎 📕 🔮 🦉 🖉
⊕. 2	0 nid00048	В	aprun <worker>.0 (2 active threads)</worker>	Go Hait Kill Restart Next Step Out Run To Record GoBack Prev UnStep Caller BackTo Liv
. 3	1 nid00048	B1	aprun <worker>.1 (1 active threads)</worker>	Rank 2: aprun <worker>.2 (At Breakpoint 1) Thread 1 (18081): worker (At Breakpoint 1)</worker>
⊕ . 4	2 nid00048	B1	aprun <worker>.2 (1 active threads)</worker>	Stack Trace
⊕ 5	3 nid00048	B1	aprun <worker>.3 (1 active threads)</worker>	F90 worker, FP=7fffffff7d20
⊕. 6	4 nid00048	B1	aprun <worker>.4 (1 active threads)</worker>	Clibc_start_main, FP=7fffffff7de0 No arguments. start, FP=7fffffff7df0 Local variables:
i∰7	5 nid00048	B1	aprun <worker>.5 (1 active threads)</worker>	dummy: 0 (0x0000000)
⊕8 ⊥	6 nid00048	B1	aprun <worker>.6 (1 active threads)</worker>	err: 0 (0x0000000) job: 0 (0x0000000)
i⊕9 i⊕10	7 nid00048	31	aprun <worker>.7 (1 active threads)</worker>	master: 0 (0x00000000) mpi argvs null: (character(len=1)(1,1))
⊕ 10 ⊕ 11	8 nid00048 9 nid00049	B1 B	aprun <worker>.8 (1 active threads)</worker>	mpi argy null: (character(len=1)(1))
(±)	3 11000043	n	aprun <master>.9 (1 active threads)</master>	mpi_bottom: 0 (0x00000000)
				mpi_errcodes_ignore: (INTEGER*4(1))
				Function worker in worker.F90
				11 INTEGER :: nprocs, myrank, tag, dummy, err 12 INTEGER, DIMENSION(MPI_STATUS_SIZE) :: stat
				13 INTEGER :: master 14
				15 ! Count variables
				16 INTEGER job 17
				18 CALL MPI INIT(err)
				19 CALL MPI COMM_SIZE (MPI_COMM_WORLD, nprocs, err) 20 CALL MPI COMM_RANK (MPI_COMM_WORLD, myrank, err)
				21 22 ! The main loop
				🚍 master= nprocs-1
				$\begin{array}{rcl} 24 & tag &= 1\\ \hline 25 & dummy &= 0 \end{array}$
				26 D0
				27 CALL MPI SEND(dummy, 1, MPI_INTEGER, master, tag, MPI_COMM_WORLD, err) 28 CALL MPI_RECV(job, 1, MPI_INTEGER, master, taq, MPI_COMM_WORLD, stat, err)
				29 IF (job == -1) EXIT
				31 write (*,*) "Worker ", myrank, " does work number ", job
				32 33 ! The work that the workers do
				Action Points Processes Threads P- P+ PX T- T+
				p1 0 1 2 3 4 5 6 7 8 9

Message Queue Graph

Message Queue Debugging

- Filtering
 - Tags
 - MPI Communicators
- Cycle detection
 - Find deadlocks

Layout	Cycle Detection	Filter	Save As
📕 <u>D</u> etect Cycle	s		
Next Cycle			
Reset Cycle	Search		
Apply	Hide		Help





TVScript

- Gives you non-interactive access to TotalView's capabilities
- Useful for
 - Debugging in batch environments
 - Watching for intermittent faults
 - Parametric studies
 - Automated testing and validation
- TVScript is a script (not a scripting language)
 - It runs your program to completion and performs debugger actions on it as you request
 - Results are written to an output file
 - No GUI
 - No interactive command line prompt
- Used at sites such as DMI and STFC Daresbury for automated comparative debugging

C++View

- C++View is a simple way for you to define type transformations
 - Simplify complex data
 - Aggregate and summarize
 - Check validity
- Transforms
 - Type-based
 - Compose-able
 - Automatically visible
- Code
 - C++

*E*RogueWave

- Easy to write
- Resides in target
- Only called by TotalView

	./milestone_example		trangle - main - L.L.		- 0 8
	File Edit View Group Process Thread Action Point Debug Tools Win	File Edit View T	ools Window		Help
		1,1		· 변명 화	$K \in \mathcal{F} H$
	Group (Control) Go Halt KII Restart Next Step Out Run To Prev UnS	Expression: triangle	Address	Cxbfb05754	
ble	Process 1 (19525): miestone_example (Stopped)	Type: struct s	td::vector <std::vector<doubl< td=""><td>e,std::allocator<double></double></td><td>5::bt8,<</td></std::vector<doubl<>	e,std::allocator <double></double>	5::bt8,<
	Thread 1 (19525) (Stopped) <trace trap=""></trace>	Field	Type	Value	10
ly	Stack Trace I Stack	- at(0).front()	\$string	"empty!"	13
	libc_start_main, FP=bfb05868 No parameters	e- at(1).front()	double[1]	(Array)	
	- k: Q	L[0]	double	0	
	Local variables	e at(2) front()	double[2]	(Array)	
	Function main in milestone_example.cox	- [0]	double	0	1
	42 43 int TV display_type(const vector <vector<double> > **</vector<double>	L[1]	double	2	
	44 (- at(3).front()	\$string	"emptyl"	
	45 for (int i = 0; i < vvd->size(); i++)	- at(4).front()	Sstring	"emptyl"	
;	47 char name[64]; type[64]; 48 sprintf(name, "at(%1d).front()", (long) i);	- at(5).front()	\$string	"emptyl"	17
	49 sprintf(type, "double[%ld]", (long) vvd->at(i)	.size()):			
	43 sprintf(type, "double[%ld]", (long) vvd->at(i) 50 int status; 51 int status; 53 if (vvd->at(i).size() == 0) 54 status = TV_add_row(name, TV_ascii_string_ty) 55 else 57 status = TV_add_row(name, type, &vvd->at(i).				
	52 53 if (vvd->at(i).size() == 0)				
	54 status = TV_add_row(name, TV_ascii_string_ty 55 else	pe, "emptyl");	1 100 2		
ру	55 status = TV_add_row(name, type, &vvd->at(1).	front());			
-	57 58 if (status != 0) 59 break;		WY AND		
	59 break; 60)				
	61 return TV_format_OK;				
			And the		
	Action Points Processes Threads	PETE	Lal Contract		
		1			
			1000		
		1			

Viewing Fortran User-Defined Types

TYPE WHOPPER

LOGICAL, DIMENSION(ISIZE) :: FLAGS

DOUBLE PRECISION, DIMENSION(ISIZE) :: DPSA

DOUBLE PRECISION, DIMENSION(:), POINTER :: DPPA

END TYPE WHOPPER

TYPE(WHOPPER), DIMENSION(:), ALLOCATABLE :: STUFFTYP1

-	S	tuff – f90step	⊳Alpha – 1	.1		· □
<u>F</u> ile <u>E</u> dit	<u>V</u> iew To	oo <u>l</u> s <u>W</u> indow				<u>H</u> elp
1.1	4		= =	🐶 🏠	► <	\gg
Expression:	stufftyp1		Address:	0x1400000	90 (Spar	se]
<u>S</u> lice:	(:)		Filter:			
Actual Type:	type(who	pper),allocata	ble::(100)			
<u>T</u> ype:	type(who	opper),allocata	able::(:)			
Field		Тур	e	Value		
Ģ- (1)		type(whopper)	(Struct)		4
flags		logical*4(1000))	(Array)		
- dpsa		double precisi	ion(1000)	(Array)		
🦾 dppa		double precisi	ion,pointer	:(double pre	cision,po	inter
⊕. (2)		type(whopper)	(Struct)		
⊕ ~ (3)		type(whopper)	(Struct)		
⊕. (4)		type(whopper)	(Struct)		
t <u>+</u> (5)		type(whopper	r) (Struct)			
⊕. (6)	type(whoppe)	(Struct)		
_ ⊕- (7)		type(whopper)	(Struct)		
ф. (8)		tvpe/whopper	<u>۱</u>	(Struct)		

Current Work and Future Plans

What is new in TotalView 8.14.1

- CUDA 6.5 support
- Coarray Fortran support for the Cray CCE compiler
- Extended support for type transformations with the Intel compiler (unordered STL collection classes)
- Improved delayed symbol processing (better performance for larger executables)

Multi-phase R&D Projects Underway

- Massive Scalability
 - Collaboration with LLNL and Tri-lab partners
 - Targeting Cray, Blue Gene and Linux Clusters
 - MRNet software overlay network for multicast and reduction
- New GUI
 - Sleek, Modern and Fast
 - Configurable
 - Improved Usability
 - Provides aggregation capabilities for big data and scale
 - Leveraging math and stat expertise from IMSL
- Working with customers through early access programs
 - Customer input is key to the success of both programs

Thanks!

- Visit the website
 - <u>http://www.roguewave.com/products/totalview.aspx</u>
 - Videos (3 new videos on Xeon Phi)
 - Documentation
 - Sign up for an evaluation
- Visit us at SC14 (booth 2338)





