

MPI Tuning with Intel® Trace Analyzer and Intel® Trace Collector

Intel Software College



Agenda

Introduction Basic Usage Focusing on the relevant Detecting problems



2

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Introduction – What is Tracing?

- Record program execution
 - Program events such as function enter/exit, communication
- 1:1 protocol of the actual program execution
 - Sampling gathers statistical information
- Accurate data
- Easily get loads of data



3

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.



Components and Interaction



Supported Programming Models

MPI: Message Passing

Languages: C/C++, Java*, Fortran

Multithreading: explicit threads - implicit threads (OpenMP)

Hybrid Models: MPI + OpenMP



MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Supported Environments

Intel® Xeon® Processor Intel 64 Bit Xeon Processor Intel Itanium® Processor Intel MPI, MPICH, LAM-MPI Linux (RHEL, RHAS, SLES) SGI* Altix* Microsoft* Windows* (Analyzer Only)



7

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Introduction – Intel® Trace Collector

Key features

Usage



8

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.



Intel® Trace Collector - Overview

Event based approach Low impact on application performance Provides API to instrument user code Trace optimized program runs Analyzes communication layer (default)



10

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Event based approach

Event = time stamp + thread ID + description

Function entry/exit

Messages

- **Collective operations**
- Counter samples

(intel) Software

11

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Strengths of Event-based Tracing

Predict detailed program behavior

Record exact sequence of program states – keep timing consistent

Collect information about exchange of messages: at what times and in which order

An event-based approach is able to detect temporal dependencies!



12

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Key Features

Catch all MPI events

Strong configuration mechanism

• Filters, settings, features

Automatic source-code references

Instrumentation

- Rich API
- Binary instrumentation (itcinstrument)
- Compiler based (beta)

Fail-safe version

(intel

Software

13

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

How to use Intel® Trace Collector

Step 1: Generate instrumented binary re–link your application: mpicc -shlib tst.o \$(LFLAGS) -lVT -lmpi -ldwarf -lelf -lnsl -lm -o tst

Step 2: Produce tracefile

run the instrumented binary for a representative amount of time (reduce initialization influences) on representative data (no corner cases)

Step 3: (optional): Instrument binary itcinstrument -input lm -output lm instr



14

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Lab 1 - Set up Environment

Get hands-on package



Intel[®] Software College

Edit make_base to reflect local configuration

SETUP ENV

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

15

(intel

Software

Get hands-on package

Package name:
/home/public/itac-bcor2005.tgz

Contents:

- make_base
- Makefile
 - task01
 - README
 - Description
 - source files
 - task02
 - task03
 - •••

16

(intel

Software

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Edit make_base to reflect local configuration

CC	= [path to MPI compiler wrapper]
MPI_ROOT	= [path to MPI implementation]
VT_ROOT	= [path to Intel® Trace Collector]
RUN_CMD	= [command to run mpi exec's]

Example:

(intel)

Software

17

CC	= mpiicc
MPI_ROOT	= /opt/intel/mpi/2.0
VT_ROOT	= /opt/intel/itc6
RUN_CMD	<pre>= /opt/intel/mpi/2.0/bin64/mpiexed</pre>

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



-n

Copyright © 2006, Intel Corporation. All rights reserved.

SETUP ENV

Intel[®] Trace Collector installation is in [ITC]: /opt/intel/itc6

- csh: source [ITC]/sourceme.csh
- sh: sh [ITC]/sourceme.sh

Intel[®] Trace Analyzer installation is in [ITA]: /opt/intel/ita6

- csh:
 - set path = (. [ITA]/bin [ITC]/bin \$path)
- bash:
 - export PATH=\$PATH:[ITA]/bin:[ITC]/itc6/bin
- **start:** traceanalyzer



18

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Lab 2 – Trace File Generation

Objective: How to generate a tracefile.

This directory contains **cpi.c** which is a simple example program.

Re-link (or compile and link) with Intel® Trace Collector library.

Run again – check for console message on 'writing tracefile'.

Open tracefile with Intel® Trace Analyzer.
[traceanalyzer cpi_itc.stf]



19

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Lab 3 – Trace File Generation



20

Copyright © 2006, Intel Corporation. All rights reserved.

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Introduction – Intel® Trace Analyzer

Key features

Usage



21

Copyright © 2006, Intel Corporation. All rights reserved.



Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. *Other brands and names are the property of their respective owners

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector

Trace Universe - Intel® Trace Analyzer



Intel® Trace Analyzer - Overview

Enables the user to quickly focus at the appropriate level of detail Offline trace analysis (postmortem)

Use of hierarchical techniques to address scalability in function, time and processor-space

High-performance graphics, excellent zooming and filtering



23

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Views and Charts - I

What's a Chart? What Charts are provided? What's a View? What's the purpose of a View?



24

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Chart



A Chart is a numerical or graphical diagram

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



25

Copyright © 2006, Intel Corporation. All rights reserved.



Timelines: Event Timeline



Timelines: Qualitative Timeline

Find patterns and irregularities



MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

(intel)

Software

27

Timelines: Quantitative Timeline



Get impression on parallelism and load balance

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

(intel

Software

28

Profiles: Flat Function Profile

Statistics about functions

Flat Profile Load Bala	ance Call Tree	2 Call Graph	1			Flat Pro	file Load Balance	Call Tree	Call Graph			
Group All_Threads	•					Childre	n of Group All_Threa	ds 🔻				
Name	TSelf	TSelf /	TTotal	#Calls	TSelf/Call	Name		TSelf	TSelf /	TTotal	#Calls	TSelf /Call
Group All_Threads Name Group All_Threads PRECON OMP_SYNC MATMUL SOLVER User_Code MPI_Bcast ASSEMBLY MPI_Barrier MPI_Reduce MPI_Reduce MPI_Stall MPI_Comm_dup MPI_lsend MPI_lrecv MPI_Irecv MPI_Comm_size MPI_Comm_rank	TSelf 678.787 445 ± 580.473 344 ± 410.463 131 ± 328.400 819 ± 149.746 154 ± 94.227 914 ± 43.822 701 ± 24.222 498 ± 23.807 645 ± 17.607 615 ± 11.756 564 ± 7.838 668 ± 7.490 313 ± 4.000 1295 ± 0.000 283 ±	TSelf /	TTotal 678.787 445 s 580.473 344 s 410.463 131 s 2 188.146 834 s 2 383.561 817 s 94.227 914 s 43.822 701 s 24.222 499 s 23.807 645 s 17.607 615 s 17.567 645 s 7.836 669 s 7.490 313 s 4.909 197 s 0.006 288 s 0.001 205 s 0.000 293 s	#Calls 49 536 296 320 128 37 248 37 248 37 248 37 184 49 312 37 184 49 472 64 145 324 136 192 145 324 32	TSelf /Call 0.013 703 s 0.0013 703 s 0.000 329 s 2.565 631 s 1.169 892 s 0.002 530 s 1.369 459 s 0.000 491 s 0.000 540 s 0.000 054 s 0.000 055 s 0.000 197 s 0.000 199 s	Childrer Name ⊕ MPI ⊕ MPI ⊕ MPI ⊕ P P <	n of Group All_Thread 	0.013 338 0.013 338 0.01994 0.756 392 0.711 207 0.643 754 0.643 754 0.643 754 0.643 754 0.643 754 0.643 754 0.643 754 0.643 754 0.643 754 0.643 754 0.654 0.57 0.628 698 0.573 404 0.573 404 0.574 251 0.547 251 0.445 023 0.448 293 0.448 293 0.342 265 22 0.342 265 22 0.342 265 20 0.342 265 2	TSelf /	TTotal 0.013 339 c 0.801 994 s 0.755 332 s 0.711 207 s 0.633 7547 s 0.643 7544 s 0.663 7547 s 0.663 7547 s 0.628 403 s 0.610 254 s 0.529 456 s 0.573 404 s 0.573 404 s 0.574 251 s 0.547 177 s 0.495 023 s 0.495 023 s 0.445 023 s 0.445 023 s 0.445 023 s 0.445 023 s 0.446 212 s 0.452 455 s 0.454 285 s 0.452 285 s 0.452 285 s 0.452 285 s 0.452 285 s	# Calls 1 546 1 546	TSelf /Call ▲ 0.000 501 s 0.000 519 s 0.000 489 s 0.000 489 s 0.000 480 s 0.000 460 s 0.000 460 s 0.000 460 s 0.000 460 s 0.000 460 s 0.000 406 s 0.000 395 s 0.000 372 s 0.000 372 s 0.000 371 s 0.000 354 s 0.000 354 s 0.000 395 s 0.000 354 s 0.000 300 s 0.000 370 s 0.000 370 s 0.000 370 s 0.000 293 s 0.000 293 s 0.000 294 s
						- P - P - P - P - P - P - P - P - P - P	Irocess 22 I hread 0 Irocess 14 Thread 0 Irocess 17 Thread 0 Irocess 9 Thread 0 Irocess 9 Thread 0 Irocess 19 Thread 0	0.387 010 - 0.377 664 - 0.377 174 - 0.374 776 - 0.357 603 -		0.387 010 s 0.377 664 s 0.377 174 s 0.374 776 s 0.357 603 s 0.245 502 s	: 1546 : 1546 : 1546 : 1546 : 1546 : 1546	0.000 250 s 0.000 244 s 0.000 244 s 0.000 242 s 0.000 231 s 0.000 040 s
						- P - P - P - P - P - P	rocess 10 Thread 0 rocess 9 Thread 0 _Irecv rocess 19 Thread 0 rocess 17 Thread 0	0.374 776 : 0.357 603 : 0.245 502 : 0.243 382 :		0.374 776 s 0.357 603 s 0.245 502 s 0.243 382 s	i 1 546 i 1 546 i 6 184 i 6 184	

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Software

29

Copyright © 2006, Intel Corporation. All rights reserved.

Profiles: Call-Tree and Call-Graph

Function statistics including calling hierarchy

Flat Prof	Flat Profile Load Balance Call Tree Call Graph					
Childrer						
Name		То-к	TO-K			
- Proc	Ivame V	ISelf	1 Self	liotai	#Calls	TSelf/Call
- Proc	☐ Group All_Processes					
	□ □ Callers					
Ę	STF_ReachedEndOfFilter calling STF_WorkStackHistory	0.001 000 s		0.002 869 s	37	0.000 027 s
	STF_InitFileInput calling STF_WorkStackHistory	0.000 021 s		0.000 055 s	1	0.000 021 s
	STF_DecodeFilter_enter_function calling STF_WorkStackHistory	0.000 094 s		0.000 320 s	1	0.000 094 s
	STF_ContentFilter_one_to_one_communication calling STF_WorkStackHisto	ry 0.000 112 s		0.001 476 s	2	0.000 056 s
	STF_ContentFilter_all_to_all_communication calling STF_WorkStackHistory	0.000 068 s	1	0.001 528 s	1	0.000 068 s
	STF_DecodeFilter_leave_function calling STF_WorkStackHistory	0.000 372 s		0.010 334 s	з	0.000 124 s
IIII	STF_DecodeFilter_enter_function_1 calling STF_WorkStackHistory	0.000 032 s		0.000 244 s	1	0.000 032 s
É É	- STF_WorkStackHistory	0.001 699 s		0.016 826 s	46	0.000 037 s
	E- Callees					
Ļ	STF WorkStackHistory calling PAL IsInTriplets	0.001 683 s		0.016 810 s	37	0.000 045 s
	STF_WorkStackHistory calling STF_WillyForAll	0.001 104 s		0.005 784 s	30	0.000 037 s
	- STF WorkStackHistory calling STF CallFromContent begin of history	0.001 426 s		0.016 352 s	32	0.000 045 s
	- STF WorkStackHistory calling STF CallHandler	0.001 647 s		0.016 717 s	35	0.000 047 s
	STF WorkStackHistory calling STF CallFromContent end of history	0.001 426 s		0.016 352 s	32	0.000 045 s
l é	STF WorkStackHistory calling STF CopyFromContent begin of history	0.000 221 s		0.000 365 s	3	0.000 074 s
	STF WorkStackHistory calling STF CopyFromContent end of history	0.000 221 s		0.000 365 s	3	0.000 074 s
]		-			
- Proce	uss 2					
i≟- Us	er_Code 0.663 430 s 163.970 788 s 1 0.663 430 s					
	MPI_Barrier 0.040 269 s 0.040 269 s 2 0.020 134 s					
l Î	-MPI_Allreduce 88.085 492 s	-				

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



30

Copyright © 2006, Intel Corporation. All rights reserved.

Communication Profiles

×			Tot	al Time [s	i] (Collec	tive Oper	ration by	Process)		
	P0	P1	P2	P3	P4	P5	P6	P7	Sum	Mean	StdDev
MPI_Barrier	0.06	3 0.052	0.040	0.180	0.258	0.066	0.079	0.215	0.952	0.119	0.080
MPI_Bcast	0.00	0 0.860	0.865	0.857	0.853	0.855	0.860	0.861	6.010	0.751	0.284
MPI_Allreduœ	87.29	9 120.679	88.085	127.782	89.071	124.266	109.330	137.064	883.576	110.447	18.704
Sum	87.36	2 121.590	88.990	128.818	90.182	125.187	110.268	138.141	890.538		
Mean	29.12	1 40.530	29.663	42.939	30.061	41.729	36.756	46.047		37.106	
StdDev	41.13	9 56.675	41.312	59.993	41.727	58.363	51.318	64.359			52.973
	Mean :	23.903 63.11	5 43.427 4	9.759 37.39	4 45.866 2	9.323 35.86	1 4	2.681	29		
	StdDev	0.000 11.52	5 1.822	1.798 0.49	0 8.248	4.939 0.00	0	12.62	24		

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

(intel)

Software

31

View

Helps navigating through the trace data and keep orientation

Every View can contain several Charts

A View on a file is defined by a triplet of

- time-span
- set of threads
- set of functions

All Charts follow changes to View (e.g. zooming)

Timelines are correctly aligned along time



32

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

View - zooming



MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



33

Copyright © 2006, Intel Corporation. All rights reserved.

Flexibility of Views

Several Views can be opened (on the same or on different files)

Location, orientation and size of charts can easily be changed

Entire Views can and individual charts can be cloned and closed

Individual charts can be cloned in own View



34

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Intel[®] Software College

Lab 4 – Views and Charts



intel Software MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Charting and Viewing

Start with

traces/ztomo/out_altixtest_ori.2.stf

Use Chart menu to open

- EventTimeline
- Quantitative Timeline
- Function Profile (Flat Profile)
- Function Profile (CallGraph)



36

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.
Lab 4 - Results



MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



37

Copyright © 2006, Intel Corporation. All rights reserved.

Charting and Viewing

Start with

traces/ztomo/out_altixtest_ori.2.stf

Use Chart menu to open

- EventTimeline (zoom to very end (left))
- Collective Operation Profile

Use Layout menu to get time line to left

Use Context menu in CollOpChart

- Colums to show Communicator
- Rows to show Processes

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

(intel

38

Lab 4b - Results



MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



39

Copyright © 2006, Intel Corporation. All rights reserved.

Intel[®] Software College

End of Module



40

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Grouping and Aggregation

Allow analysis on different levels of detail by aggregating data upon group-definitions

Functions and threads can be grouped hierarchically

Function Groups and Thread Groups

Arbitrary nesting is supported

- Functions/threads on the same level as groups
- User can define his/her own groups

Aggregation is part of View-definition

- All charts in a View adapt to requested grouping
- All charts support aggregation

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

(intel

41

Aggregation Example



MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



42



Intel[®] Software College

Lab 5 - Aggregation



MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

(intel)

Software

43

Two ways to determine time spent not in MPI for instrumented codes

Start with

traces/ztomo/out_altixtest_ori.2.stf

Quick

(intel

44

- Select MPI for Function Aggregation
- MPI <-> non-MPI
- Create new group
- Drag & Drop MPI
- Select new group for Function Aggregation





Copyright @ 2006, Intel Corporation. All rights reserved.

Tagging & Filtering

Help concentrating on relevant parts

Avoid getting lost in huge amounts of trace data

Define a set of interesting data

- E.g. all occurrences of function x
- E.g. all messages with tag y on communicator z

Combine several filters: Intersection, Union, Complement

Apply it

45

(intel

- Tagging: Highlight messages
- Filtering: Suppress all non-matching events

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Tagging Example



MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



46

Copyright © 2006, Intel Corporation. All rights reserved.

Filtering Example



Software

Copyright © 2006, Intel Corporation. All rights reserved.

Lab 6 – Tagging and Filtering





48

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Find threads performing a task

Start with

traces/ita/ita_fgtl_threaded.stf

Search for FuncProfAnalyzer::*

Load Balance & Event Timeline

Tagging

- Advanced | Tagging
 - Select Function tab and Custom
 - Type FuncProfAnalyer and press ok



49

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Suppress GUI idle time

Start with traces/ita/ita_fgtl_threaded.stf
Assumption: GUI is idle time
Use Filter (Function, "GUI", invert)



50

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright $\ensuremath{\mathbb{C}}$ 2006, Intel Corporation. All rights reserved.

Detecting Problems

Link to material

Verify important hypotheses

Slide demo

• Live demo



51

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Understanding your code

Parallel Poisson Solver

Example of intuitive parallelization with disadvantageous communication pattern



52

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Partial Trace of Poisson Solver

P0	(MP(<mark>NC:</mark> MP)		(CIMPL CNC	CaMPL CN	CIMP CMP	I <mark>C(MPL</mark>
Р1					CIMPL CM	NCIMPI
P2	MP (MCIMPI		C:MPL CINC	CIMPL CILA		Ca <mark>MPI</mark>
P3			A CAMPLE CAM	CIMPI CIM		4 <mark>Ca</mark> MPI
Ρ4	MIMECMPL		EC MPL CNOME			E <mark>C MPL</mark>
P5	MIMP C MPI	CMPI CMPI	PIC MPI			2 <mark>C</mark> MPI
P6	MIMPIC MPI	KI MPL C MPI				
P7	MPLCMPI ((MMPL_CMPI(MMPI	CMPI (MPI	CMPI (MMPI		
P8						
P9	MPL CMPI	MPI CMPI (CMPI				C MPI
P10		MPL CMPI				CMPL
P11			CMPI ((MPI			
P12						
P13						
P10						
P17						
P18						
P19						
P20						
P21						
P22						
P23						
P24			C MPHON MPI		CIMPLE (MPL	
P25					CIMPING	
P26					CMPNOMPL	
P27					CMERGIMPI	
P28						
P29				CMCMPI		
P30	MPL CN (M MPL					
P31						
			Collec	tor		
5	3					/inte



Copyright © 2006, Intel Corporation. All rights reserved.

Understanding the problem

Blocking border exchange

 P_n has blocks until communication between P_{n+1} and P_{n+2} was completed

Solution: Non blocking communication



54

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Non-Blocking Version

P0	ACalculation (MMPCN(Calcul(MAMPI))(CaMPI))			Calc
P1	MCalculation (MCM) (MildNCCalcul: (MCN) (MCCaMP)		Calculation CMCNONCRONCCalcul CMPCMORCH CNCCCMPI	Cal
P2	Calculation (MCM) (MiCN(Calculation)) (MCM) (MCCaMPI			Cal
P3			Calculation (R.CM (R.CM (R.Calcul: (R.CM (R.C. aMP)	Calc
P4			Calculatic MCMPCNCCalcuMPCMONONCCMPL	Cal
P5			Calculatic MCMMPCH (NCCalcul NCM) NH (NCC4/4PI	Cal
P6				Calo
P7			Calculatic Calcal Calculation Charles Calculation Calc	Cal
P8				Ca
P9	MCalculatic Control Co		Calculatin Calculation Calculation (NCCaMP)	Cal
P1				Ca
P1				Cal
P1:	2 Calculatic MPCMONT CALCULATION CALCULATION COMPLETE			Cal
P1:	3 MCalculatic MCMPI (NGMONCCalcuMCMPONCHONCOMPI			Cal
P1-	4 MCalculaticeMPI CMONT ON CCalcie MCMPI (NGCMPI CMCMPI CM			Ca
P1:				Cal
P1	6 MCalculatic MCMCCit CICCalcu MPCMCCit CICCAMPI			Ca
P13	7 MCalculatit			Ca
P1				Ca
P1				Cal
P2				Cal
P2	1 MCalculatic MCMOMONON CCalcul CMPH Or the CCMPH			<u> </u>
P2:				-
P2:		alculatice CMONOR ON CCalculation OMONOR ON COMPL		Ćal
P2-	4 MICalculatic (MCMORT CALCULATION OF CALCULATION CONCERNING)			Ca
P2:	5 MECalculatic MCM (ICAlculation Plant)			Ca
P2				4
P3(0 MICalculatic MCMCMCCalcuMCMCalcom(NCCMPI			
P3		Calculatin WINC Calculatin PL (NCC/MPL	Calculatin MIN (NCCalculatin Plan (NCC2MP)	

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



55

Copyright © 2006, Intel Corporation. All rights reserved.

(intel)

Detecting Load Imbalance

Mandelbrot set (MPI-tutorial)

• mpitutorial.tar.gz

Example of intuitive parallelization with huge load imbalance



56

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Mandelbrot Example Trace



MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

(intel)

Software

57

Understanding Load Imbalance

Static partitioning assigns contiguous blocks of rows

Workload changes gradually with row-index!

What is the/a solution?

- Use modulo operator
- Use dynamic master/slave approach





58

Understanding Load Imbalance



MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Software

59

Copyright © 2006, Intel Corporation. All rights reserved.

Investigating Scalability - Amdahl's Law

Non-parallel sections limit the possible speedup

Amdahl's Law: If F is the fraction of a calculation that is sequential, and (1-F) is the fraction that can be parallelized, then the maximum speedup that can be achieved by using P processors is 1/(F+(1-F)/P)

2 runs with different number of processes:

 If sum of time spent in function increases with increasing number or processes
 => non-parallel



60

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Scalability - Example



Detecting Communication Inefficiency

Message Profile

Use minimum transfer rate

Thread Groups and Aggregation

- All_nodes group
- Intra <-> extra node

Compare with HW expectation



62

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Determining Overhead of Modules

Modules can contain more than one function

Use Function Aggregation to determine the module's overhead

Trace of Intel® Trace Analyzer

Does ZLib produce significant overhead?



63

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Focusing on the relevant

Goal: determine overhead of ZLib in the Intel Trace Analyzer

Create appropriate Function Group(s)

Use Tagging to zoom to representing interval

Use Function Profile



64

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

X

(intel

?

Step 1: Creating Group(s)

X Function Group Editor

View:

1: /Projects/psp/fschlimb/work/ict/tracing/mosel/IA32-LIN/bin/vtmos elprof3.stf

Name	Depth	Children	Id	
i./	6	3	4294967294	l
🗄 Ġ All Functions	1	1612	2147483749)
🚊 G Major Function Groups	5	5	2147483648	}
🗄 Ġ Analysis	4	181	2147483692	2
🗄 🗄 🖸 GUI	4	62	2147483841	
🗄 🔂 Graphics	4	5	2147484110	Ĵ.
😟 🖸 Misc	4	132	2147483995	;
🗄 🔂 STF	2	5	2147483935)
🖃 G New Group	-3		4294967279	
🗄 G ZLib	2	9	4294967278	Î
J	OK	Canc	el Apply	
MPI Tuning with	Intel® Trace Analyze	r & Intel® Trace		



65

Copyright © 2006, Intel Corporation. All rights reserved.

Step 2: Tagging and zooming



Step 3: Determine Overhead



Intel[®] Software College

Lab 7 -



(intel) Software

68

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Who was calling?

Start with

traces/ita/ita_fgtl_threaded.stf

Which caller of STF_WorkStackHistory caused the most costly calls to it?

Function Profile

- Advanced | Function Aggregation -> All_Functions
- Search&Find STF_WorkStackHistory
- Show it in Call Graph
- Max TimeSelf/Call of Callers group



69

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Instrumenting Functions

Generic API to instrument functions

Hierarchical grouping (classes)

Insert probes to

- declare function
- enter/leave the function

cintel Software

70

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Lab 8 -



(intel) Software

71

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Lab 9 – Basic use of API

Objective: Use of Intel® Trace Collector API to instrument a function

Edit cpi.c to instrument function 'f' as of group 'Application'

- Use VT_classdef() and VT_funcdef() to declare class/function
- Use VT_enter() and VT_leave(), to record function begin and • end

Compile-link-run and view tracefile in Intel® Trace Analyzer

• Find events 'f' in the timeline, in the function profile and in the call tree display



72

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.
ITC API: Instrument a function

Allocate a handle for a state

symname: name of the symbol *classhandle*: handle of the class (see above) *returns* handle (int) referencing this function

> MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

(intel

73

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. *Other brands and names are the property of their respective owner

Solution – Use of API

```
#include <VT.h>
[...]
int application class;
double f( double a ){
    static int f state = 0;
    if( f state == 0 ) {
      VT funcdef( "f", application class, &f state );
    VT enter( f state, VT NOSCL );
    [...]
    VT leave( VT NOSCL );
    return (result);
int main( int argc, char *argv[])
    [...]
    MPI Init(&argc,&argv);
    VT_classdef( "Application", &application_class );
```

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



74



Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. *Other brands and names are the property of their respective owner

Lab 10 – Selective Tracing

Objective: Switch tracing off and on. Edit **cpi.c** to instrument function 'f'.

• use

```
VT_traceoff(), VT_traceon()
do not record MPI_Reduce()
```

```
VT_traceoff();
MPI_Reduce(&mypi, &pi, 1,MPI_DOUBLE,MPI_SUM,0,MPI_COMM_WORLD);
VT_traceon();
```



75

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. *Other brands and names are the property of their respective owners

Lab 10 – Selective Tracing

Objective: Use of a VT_CONFIG configuration file

Compile-link-run-tracefile and check with Intel® Trace Analyzer.

- check timeline display for TRACE_OFF region where MPI_Reduce had been removed.
- Create a file 'my_vt_config' with the following line in it SYMBOL MPI_Bcast OFF
- Set environment variable VT_CONFIG to refer to complete path of '`pwd`/my_vt_config'
- Check timeline or statistics ... no more data of MPI_Bcast



76

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. *Other brands and names are the property of their respective owners.

Lab 11 – Instrumentation

Objective: Use of Intel® Trace Collector to instrument a binary

Note: shared MPI libraries *MUST* be available

Currently only available for IA32

Instrument binary:

itcinstrument --input ./cpi_vt --mpi /opt/intel/mpi_2.0/lib - profile --output ./cpi_vt_instr

Compile-link - Run the instrumented binary and check tracefile with Intel® Analyzer

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



77

Copyright $\ensuremath{\mathbb{C}}$ 2006, Intel Corporation. All rights reserved.

(intel)

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. *Other brands and names are the property of their respective owner

Lab 12 – Compare applications

Objective: Analyze and compare solver application with Intel® Trace Analyzer

The laplace equation is solved by two different iterative methods

- Using Jacobi iteration method
- Using Successive Over Relaxation

Open both tracefiles in traceanalyzer (in subdirectory ./traces) - use Window|Tile to compare the traces.

Which solver method seems to be best regarding:

- Amount of MPI?
- Load balance?
- Performance?



78

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. * Other brands and names are the property of their respective owners

Intel[®] Software College

Backup



79

Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. *Other brands and names are the property of their respective owners.

MPI Tuning with Intel® Trace Analyzer & Intel® Trace

Lab 12 – Create STF single file

Objective: Learn about stftool, use single trace file to simplify transfer of storage

Convert set of stf (structured trace format) files to single file: stftool cpi_itc.stf --logfile-format stfsingle --convert cpi_itc.single.stf

Note: Can degrade performance of ITANote: ITC config allows creation of single-STF



80

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. *Other brands and names are the property of their respective owners

Lab 13 – Statistics trace file

Objective: How to generate a tracefile with only statistic information

Generate file my_vt_config

- # enable statistics gathering
- STATISTICS ON
- #no need to gather trace data
- PROCESS 0:N OFF

Set environment variable and build example

• > setenv VT_CONFIG \$PWD/my_vt_config

Run example and look at stats-file cpi_itc.prot

• > stftool cpi_itc.stf --print-statistics

MPI Tuning with Intel® Trace Analyzer & Intel® Trace Collector



Copyright © 2006, Intel Corporation. All rights reserved.

(intel

81

