Intel Cluster Tools

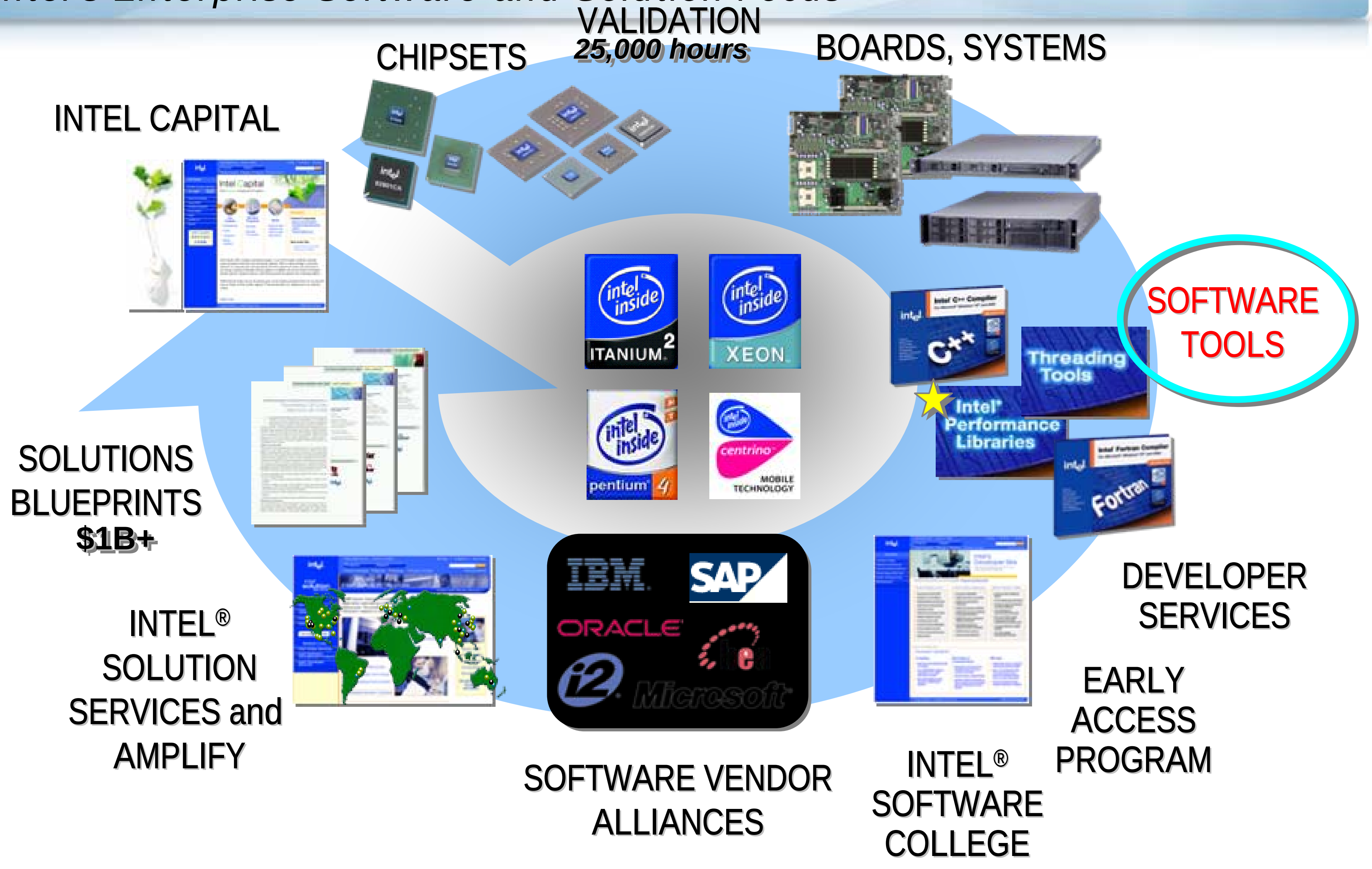
Michael Hebenstreit

Graz, May 2006

SSG
Software Solutions Group

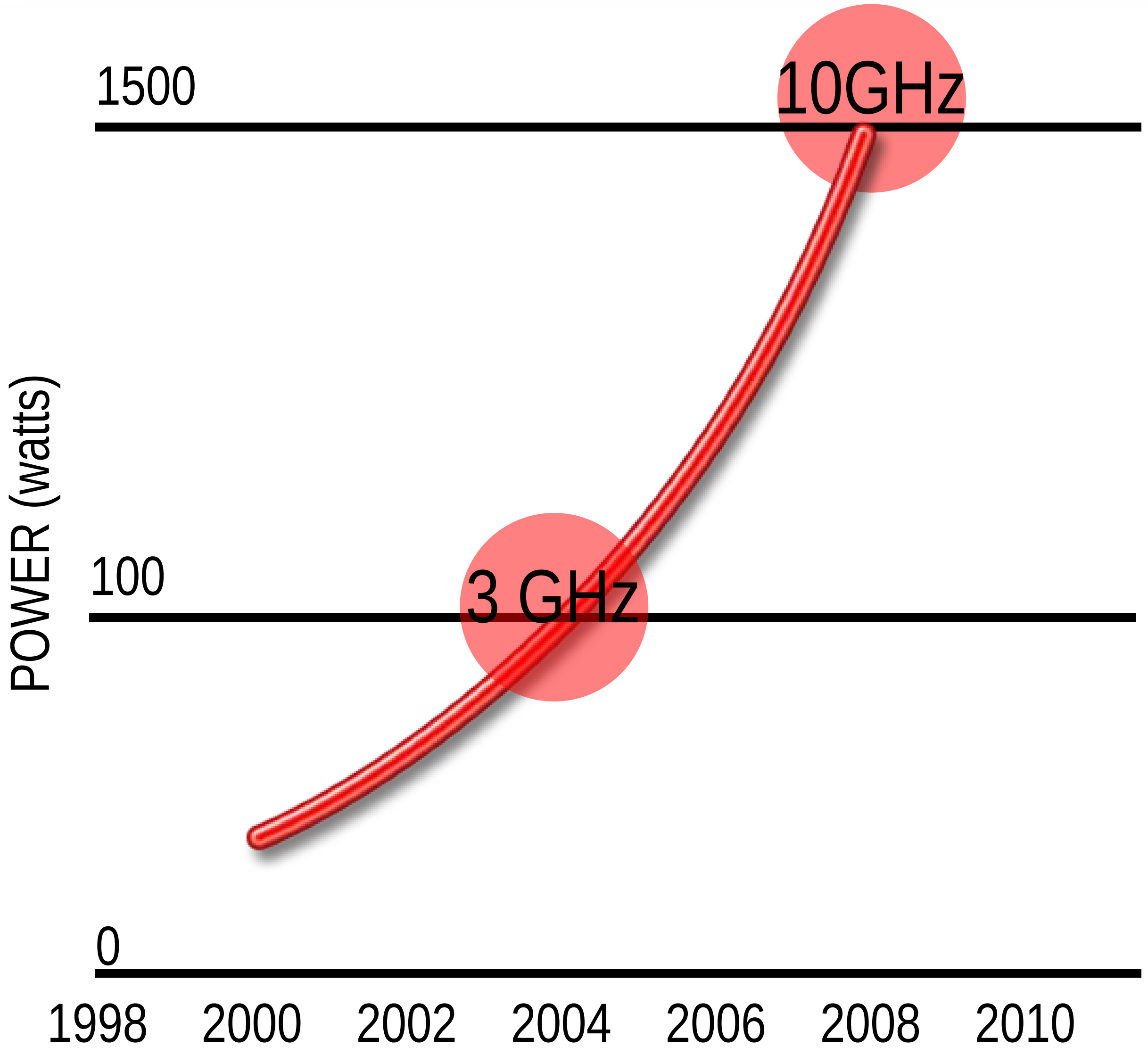
Beyond Silicon:

Intel's Enterprise Software and Solution Focus

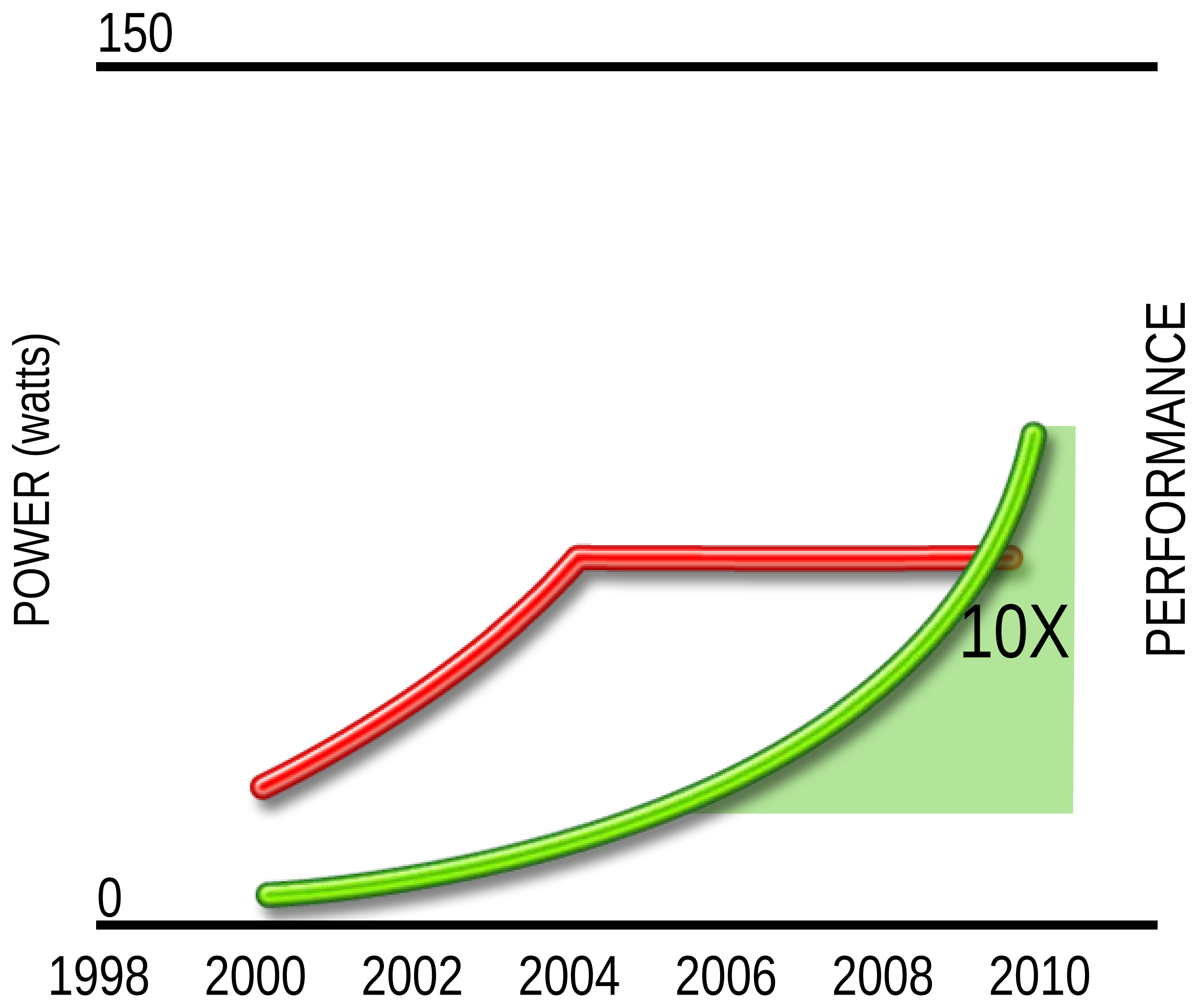


SSG

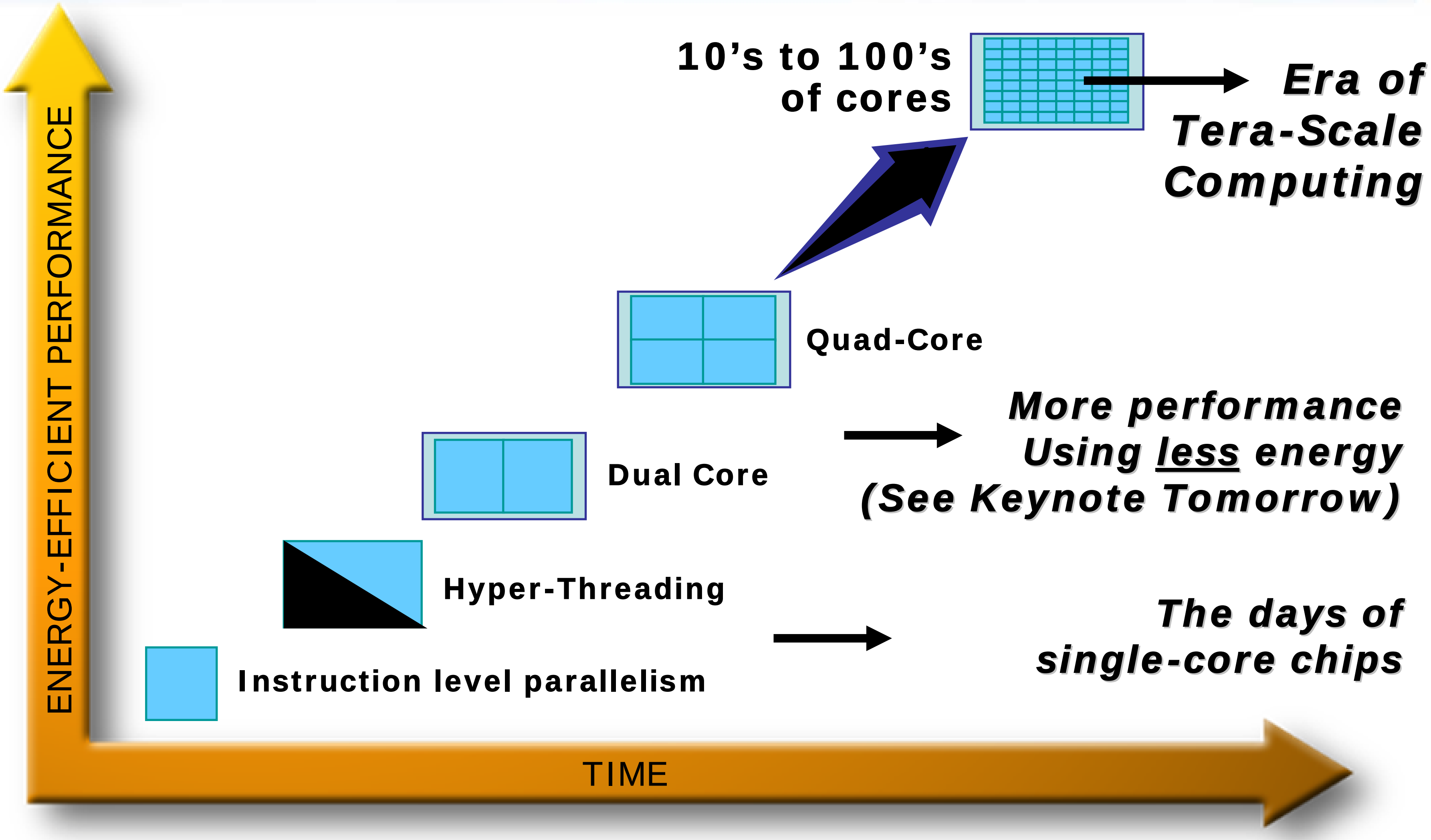
The Path We WERE On



The Path We ARE On



SSG



Era of Tera-Scale Computing

More performance Using <u>less</u> energy (See Keynote Tomorrow)

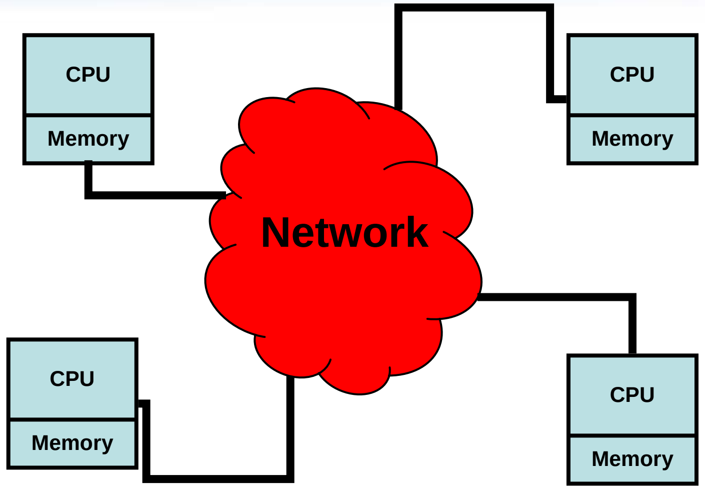
The days of single-core chips

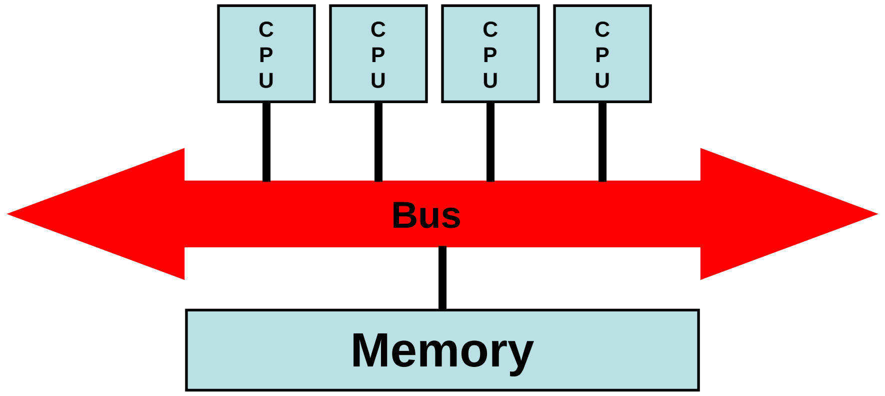
SSG

Parallel Programming: Algorithms

SSG
Software Solutions Group

Distributed Versus Shared Memory





Message Passing

- Multiple processes
- Share data with messages
- MPI

Threads

- Single process
  - Concurrent execution
- Shared memory and resources
- Explicit threads, OpenMP*

SSG

Parallelize Loops

- Find your most time consuming loops.
- Split them up between threads.

```
void main()
{
    double Res[1000];
    for(int i=0;i<1000;i++){
       do_huge_comp(Res[i]);
    }
}
```

Sequential Program

$\rightarrow$

```
#include “omp.h”
void main()
{
    double Res[1000];
#pragma omp parallel for
    for(int i=0;i<1000;i++){
       do_huge_comp(Res[i]);
    }
}
```

Parallel Program

SSG

Pi Integration parallel

```
#include <stdio.h>
#include "mpi.h"
#define INTERVALS 10000000
int main (int argc, char* argv[])
{
  double partialSum, myPi = 0.0;int nProc, myID, source, dest, tag = 0;
   int i;double h, x, pi = 0.0; MPI_Status status;

  MPI_Init (&argc, &argv);
  MPI_Comm_size (MPI_COMM_WORLD, &nProc);
  MPI_Comm_rank (MPI_COMM_WORLD, &myID);

  h = 1.0 / (double)INTERVALS;

  for (i = myID + 1; i <= INTERVALS; i += nProc)  {
    x = h * ((double)i - 0.5);
    myPi += 4.0 / (1.0 + x * x);
  }
  MPI_Reduce (&myPi, &pi, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);

  if (myID == 0) {pi *= h; printf ("Pi = %.9f\n", pi);}
  MPI_Finalize ();
}
```

SSG

Conclusions

- Hardware architecture already complicated – and will not get easier to use in the future.

Use tools wherever possible to reduce errors and enhance performance!

SSG

Introduction to VTune

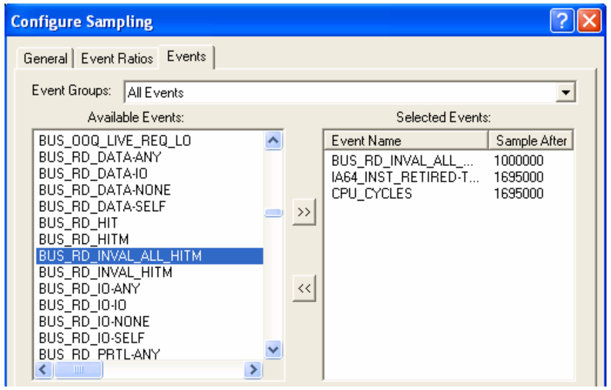
Optimization on a single Node

SSG
Software Solutions Group

Events

- Clockticks
- Instructions Retired
- VTune™ Analyzer combines the counters to find the event ratio

  Cycles Per Instruction (CPI)
  - CPI = clockticks / instructions retired


Configure Sampling

Event Groups: All Events

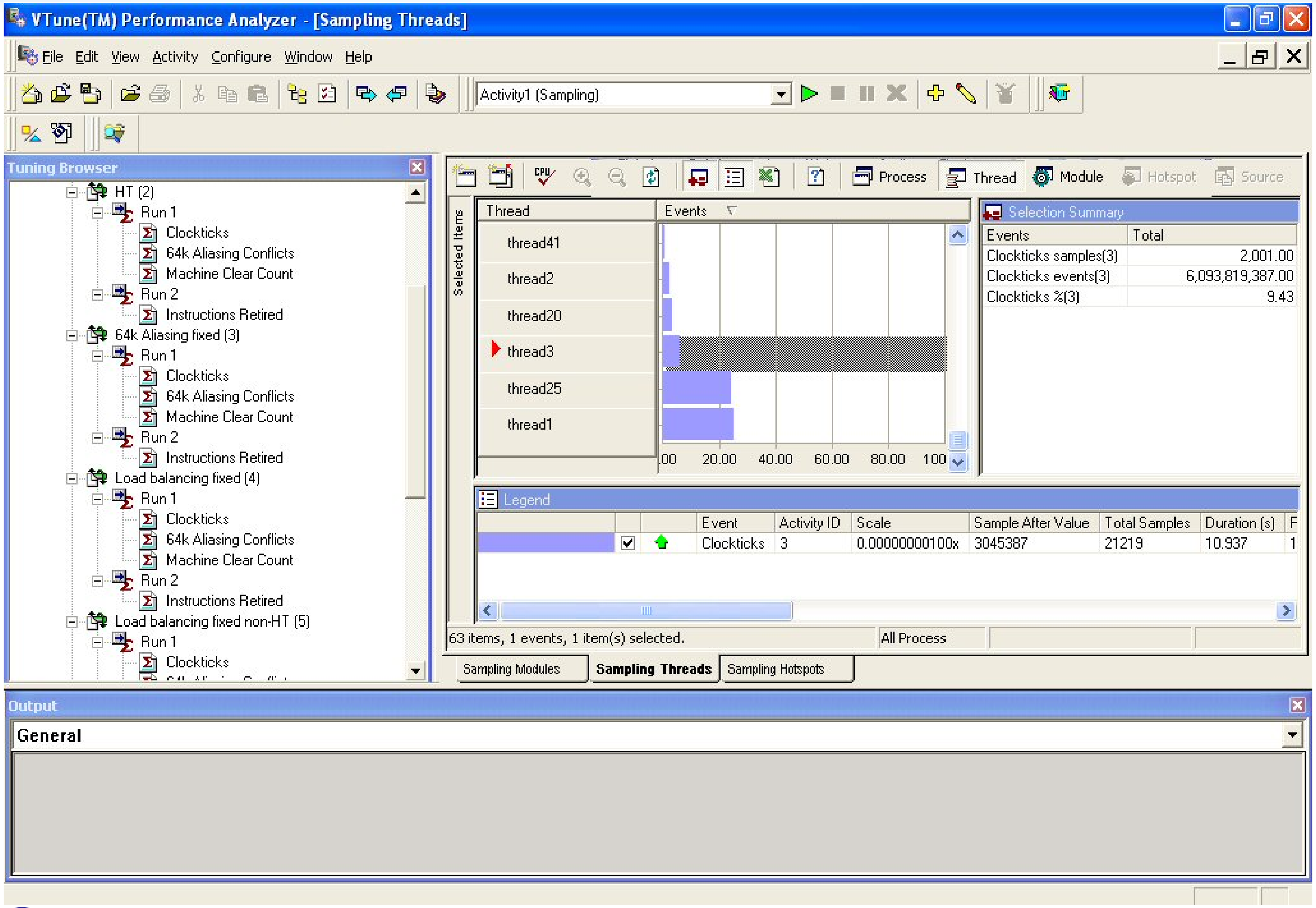
Available Events:

```
BUS_OOQ_LIVE_REQ_LO
BUS_RD_DATA-ANY
BUS_RD_DATA-IO
BUS_RD_DATA-NONE
BUS_RD_DATA-SELF
BUS_RD_HIT
BUS_RD_HITM
BUS_RD_INVAL_ALL_HITM
BUS_RD_INVAL_HITM
BUS_RD_IO-ANY
BUS_RD_IO-IO
BUS_RD_IO-NONE
BUS_RD_IO-SELF
BUS_RD_PRTL-ANY
```

Selected Events:

Event Name	Sample After
BUS_RD_INVAL_ALL_...	1000000
IA64_INST_RETIRED-T...	1695000
CPU_CYCLES	1695000

Sampling - GUI



VTune(TM) Performance Analyzer - [Sampling Threads]

- HT (2)
  - Run 1
    - Clockticks
    - 64k Aliasing Conflicts
    - Machine Clear Count
  - Run 2
    - Instructions Retired
- 64k Aliasing fixed (3)
  - Run 1
    - Clockticks
    - 64k Aliasing Conflicts
    - Machine Clear Count
  - Run 2
    - Instructions Retired
- Load balancing fixed (4)
  - Run 1
    - Clockticks
    - 64k Aliasing Conflicts
    - Machine Clear Count
  - Run 2
    - Instructions Retired
- Load balancing fixed non-HT (5)
  - Run 1
    - Clockticks

Thread	Events
thread41	
thread2	
thread20	
thread3	
thread25	
thread1	

.00 20.00 40.00 60.00 80.00 100

Selection Summary

Events	Total
Clockticks samples(3)	2,001.00
Clockticks events(3)	6,093,819,387.00
Clockticks %(3)	9.43

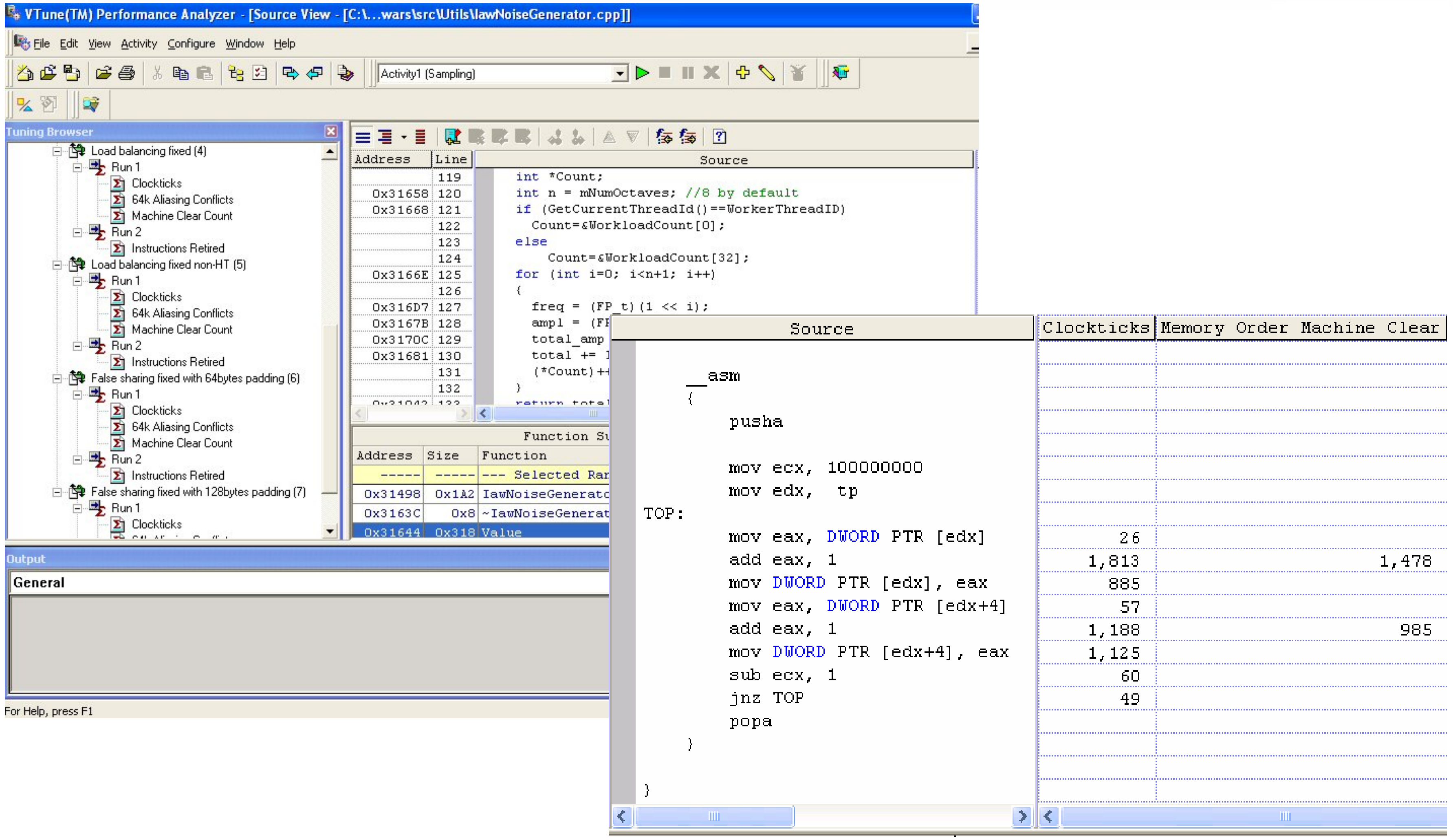
Legend

			Event	Activity ID	Scale	Sample After Value	Total Samples	Duration (s)	F
	✓		Clockticks	3	0.00000000100x	3045387	21219	10.937	1

63 items, 1 events, 1 item(s) selected. All Process

SSG

Dig down to the source view



VTune(TM) Performance Analyzer - [Source View - [C:\...wars\src\Utils\IawNoiseGenerator.cpp]]

Tuning Browser

- Load balancing fixed (4)
  - Run 1
    - Clockticks
    - 64k Aliasing Conflicts
    - Machine Clear Count
  - Run 2
    - Instructions Retired
- Load balancing fixed non-HT (5)
  - Run 1
    - Clockticks
    - 64k Aliasing Conflicts
    - Machine Clear Count
  - Run 2
    - Instructions Retired
- False sharing fixed with 64bytes padding (6)
  - Run 1
    - Clockticks
    - 64k Aliasing Conflicts
    - Machine Clear Count
  - Run 2
    - Instructions Retired
- False sharing fixed with 128bytes padding (7)
  - Run 1
    - Clockticks

```
Address   Line  Source
          119     int *Count;
0x31658   120     int n = mNumOctaves; //8 by default
0x31668   121     if (GetCurrentThreadId()==WorkerThreadID)
          122       Count=&WorkloadCount[0];
          123     else
          124         Count=&WorkloadCount[32];
0x3166E   125     for (int i=0; i<n+1; i++)
          126     {
0x316D7   127       freq = (FP_t)(1 << i);
0x3167B   128       ampl = (FI
0x3170C   129       total_amp
0x31681   130       total += I
          131       (*Count)++
          132     }
```

Address	Size	Function
-----	-----	--- Selected Rar
0x31498	0x1A2	IawNoiseGenerato
0x3163C	0x8	~IawNoiseGenerat
0x31644	0x318	Value

Output

General

For Help, press F1

Source	Clockticks	Memory Order Machine Clear
`__asm`		
`{`		
`pusha`		
`mov ecx, 100000000`		
`mov edx, tp`		
`TOP:`		
`mov eax, DWORD PTR [edx]`	26	
`add eax, 1`	1,813	1,478
`mov DWORD PTR [edx], eax`	885	
`mov eax, DWORD PTR [edx+4]`	57	
`add eax, 1`	1,188	985
`mov DWORD PTR [edx+4], eax`	1,125	
`sub ecx, 1`	60	
`jnz TOP`	49	
`popa`		
`}`		
`}`		

SSG

Benefits of Sampling

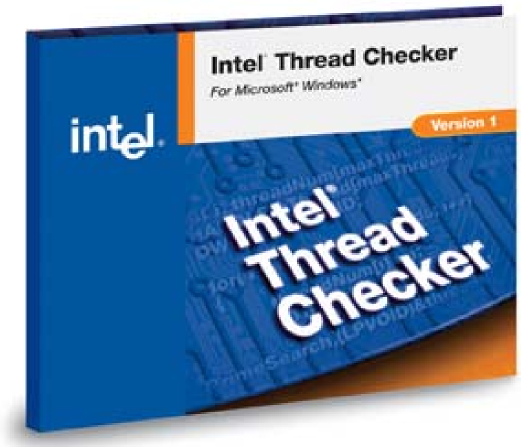
- Traditional Instrumentation:
  - Modified code
  - Higher overhead
  - Function-specific

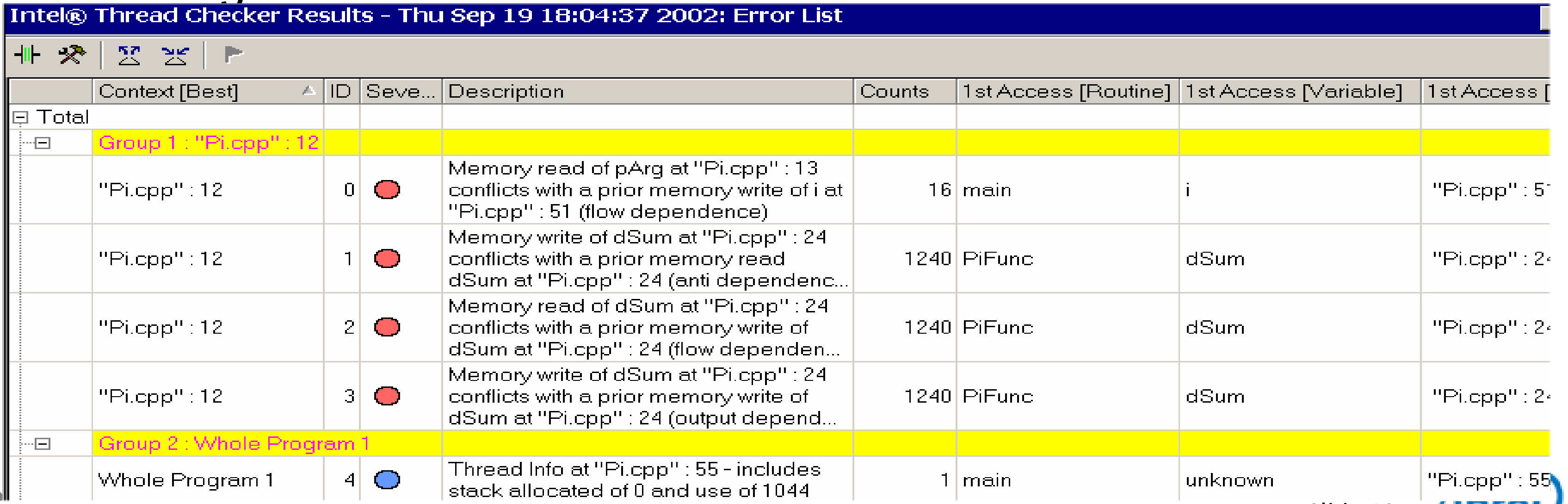
- VTune™ analyzer Sampling:
  - Unmodified code
  - Lower overhead
  - System wide coverage

Intel® Thread Checker

A “correctness” tool for threaded apps

- Finds errors in Win32*- and OpenMP*-threaded apps
- Detects incorrect memory usage – data races, deadlocks, ...more
- Plug-in for VTune™



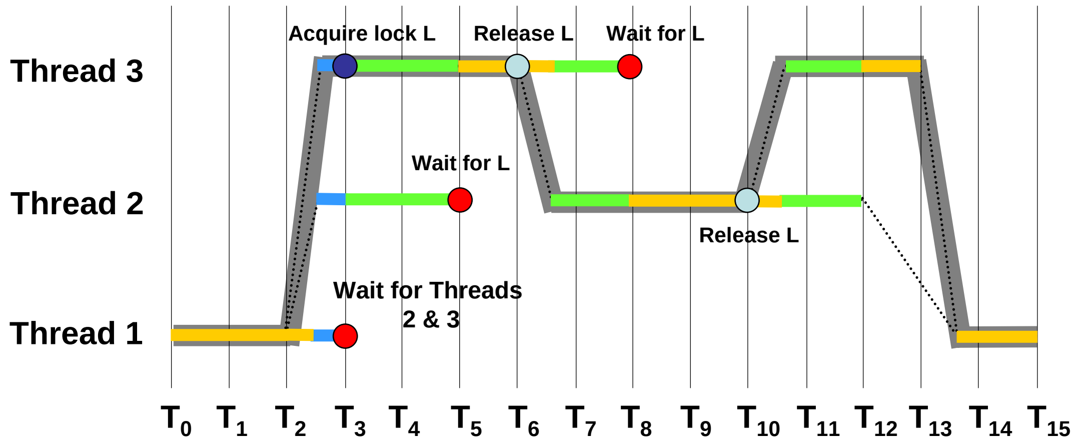

Intel® Thread Checker Results - Thu Sep 19 18:04:37 2002: Error List

	Context [Best]	ID	Seve...	Description	Counts	1st Access [Routine]	1st Access [Variable]	1st Access [
⊟ Total								
⊟	Group 1 : "Pi.cpp" : 12							
	"Pi.cpp" : 12	0	🔴	Memory read of pArg at "Pi.cpp" : 13 conflicts with a prior memory write of i at "Pi.cpp" : 51 (flow dependence)	16	main	i	"Pi.cpp" : 5
	"Pi.cpp" : 12	1	🔴	Memory write of dSum at "Pi.cpp" : 24 conflicts with a prior memory read dSum at "Pi.cpp" : 24 (anti dependenc...	1240	PiFunc	dSum	"Pi.cpp" : 2
	"Pi.cpp" : 12	2	🔴	Memory read of dSum at "Pi.cpp" : 24 conflicts with a prior memory write of dSum at "Pi.cpp" : 24 (flow dependen...	1240	PiFunc	dSum	"Pi.cpp" : 2
	"Pi.cpp" : 12	3	🔴	Memory write of dSum at "Pi.cpp" : 24 conflicts with a prior memory write of dSum at "Pi.cpp" : 24 (output depend...	1240	PiFunc	dSum	"Pi.cpp" : 2
⊟	Group 2 : Whole Program 1							
	Whole Program 1	4	🔵	Thread Info at "Pi.cpp" : 55 - includes stack allocated of 0 and use of 1044	1	main	unknown	"Pi.cpp" : 55

Intel® Thread Profiler 2.0

Critical Path utilization

- Examines processor utilization to determine parallel activity of the application
- *Concurrency* is the number of threads that are *active*


Acquire lock L
Release L
Wait for L
Thread 3
Wait for L
Thread 2
Release L
Wait for Threads 2 & 3
Thread 1
T_0 T_1 T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 T_{10} T_{11} T_{12} T_{13} T_{14} T_{15}

⬜ **Idle**
🟨 **Serial**
🟥 **Under-subscribed**
🟩 **Parallel**
🟦 **Over-subscribed**

Note: Categorization shown for a system configuration with 2 processors

SSG

Compiler optimization

SSG
Software Solutions Group

General Optimizations

Linux*	Windows*	
-O0	/Od	Disables optimizations
-g	/Zi	Creates symbols
-O1	/O1	Optimizes for speed without increasing code size (disables library function inlining)
-O2	/O2	Optimizes for speed (default)
-O3	/O3	High-level optimizations

Benchmarks

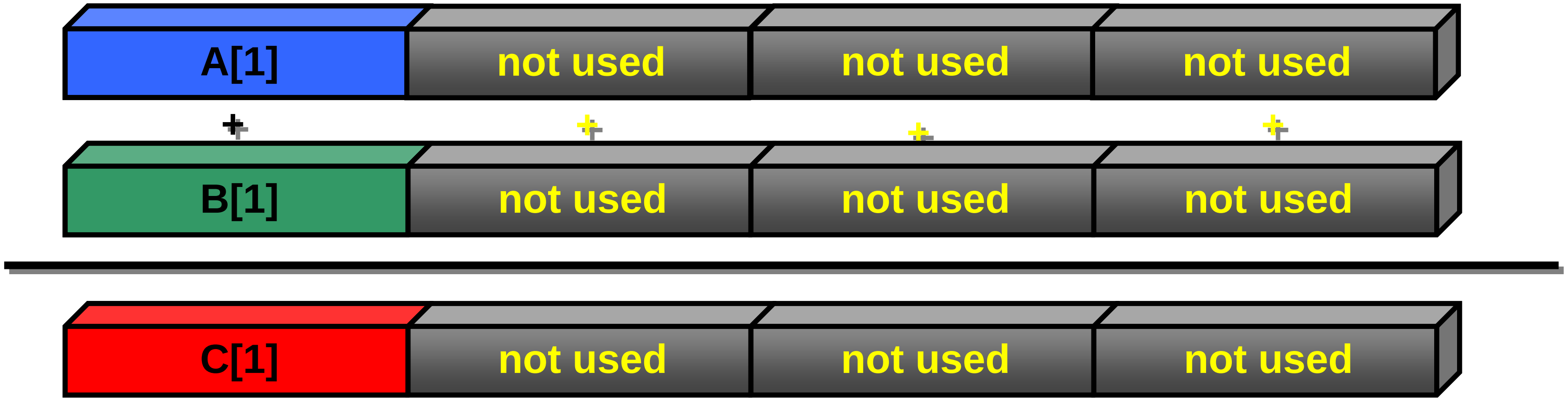
Results from „square_charge“ dataset
Linux, single Northwood 1.7GHz

-O2	20.34 s
-ipo	20.34 s
-xk	13.26 s
-ipo -xk	1.71 s
-ipo –xk -openmp	1.67 s

SSG

Vector-processing: convert this

```
for (I=0;I<=MAX;I++)
     c[I]=a[I]+b[I];
```



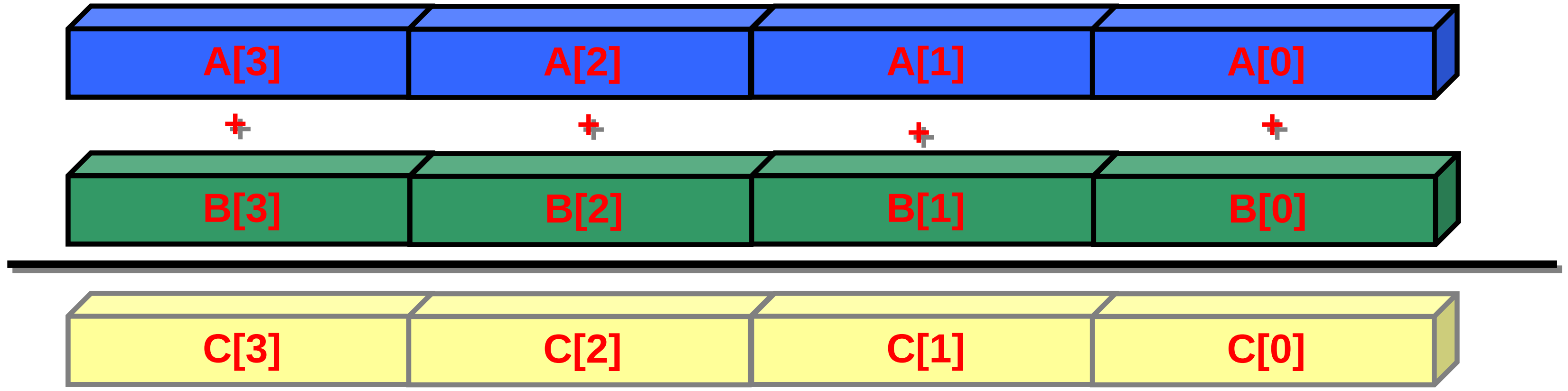
SSE Registers Usage

- xP switch is used for vectorizing loops as well as floating point and scalar operations.

- Example:

```
for (I=0;I<=MAX;I++)
     c[I]=a[I]+b[I];
```

- Usage: `-xP`



SSG

Why Didn't My Loop Vectorize?

Linux	Windows
`-vec_report`*n*	`-Qvec_report`*n*

Set diagnostic level dumped to stdout

n=*0*: No diagnostic information

n=1: **(Default)** Loops successfully vectorized

n=2: Loops not vectorized – and the reason why not

n=3: Adds dependency Information

n=4: Reports only non-vectorized loops

n=5: Reports only non-vectorized loops and adds dependency info

SSG

Intel MPI

SSG
Software Solutions Group

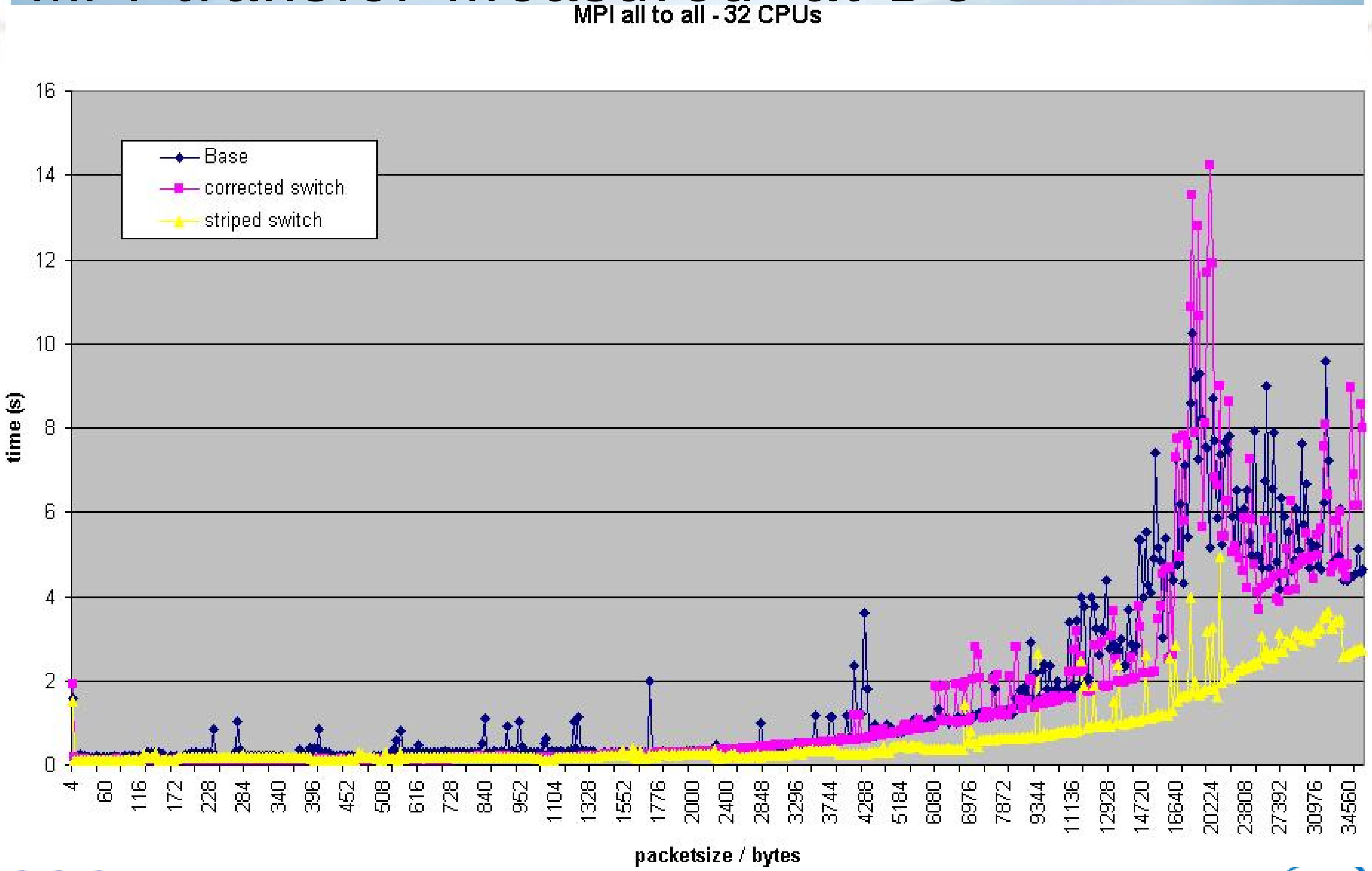
Features

- MPI-2 specification conformance
  - Standardized job startup mechanism (mpiexec)
  - Process spawning/attachment (sock device only)
  - One-sided communication
  - Extended collective operations
  - File I/O
- Multiple communication fabrics selectable at run-time
- Dynamic or static linking, plus debugging and tracing libraries
- Support for additional tools

MPI Benchmark

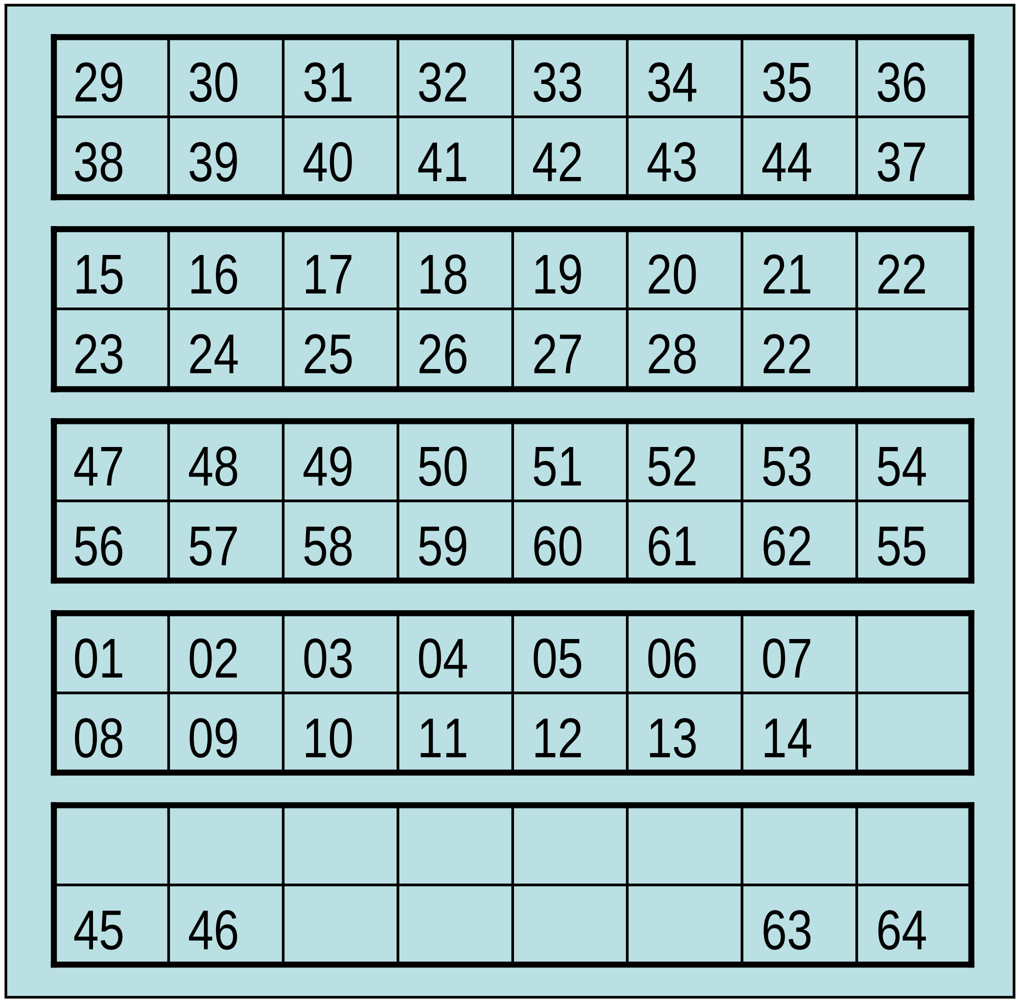
SSG
Software Solutions Group

MPI transfer measured at DC



SSG

Current switch configuration



29	30	31	32	33	34	35	36
38	39	40	41	42	43	44	37

15	16	17	18	19	20	21	22
23	24	25	26	27	28	22	

47	48	49	50	51	52	53	54
56	57	58	59	60	61	62	55

01	02	03	04	05	06	07	
08	09	10	11	12	13	14	

45	46					63	64

- Each number signifies one specific node.
- User typically request a block 01-16 or 49-64.
- Communication uses typically only one or two modules.

SSG

Striped switch configuration

01	06	11	16	21	26	31	36
41	46	51	56	61			

02	07	12	17	22	27	32	37
42	47	52	57	62			

03	08	13	18	23	28	33	38
43	48	53	58	63			

04	09	14	19	24	29	34	39
44	49	54	59	64			

05	10	15	20	25	30	35	40
45	50	55	60				

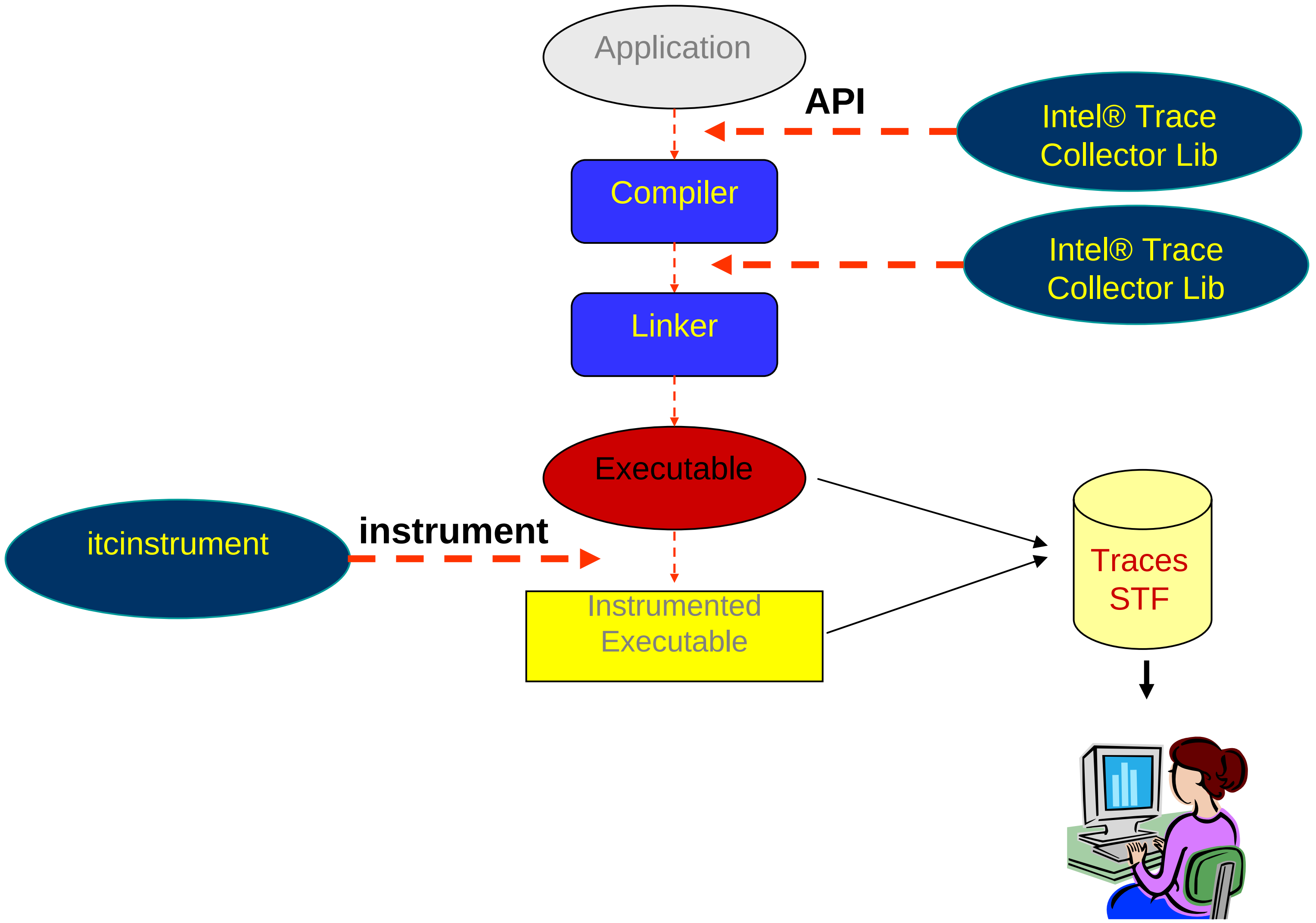
- Each number signifies one specific node.
- User typically request a block 01-16 or 49-64.
- Communication uses typically all modules.

SSG

Intel® Trace Analyzer and Collector 6.0

SSG
Software Solutions Group

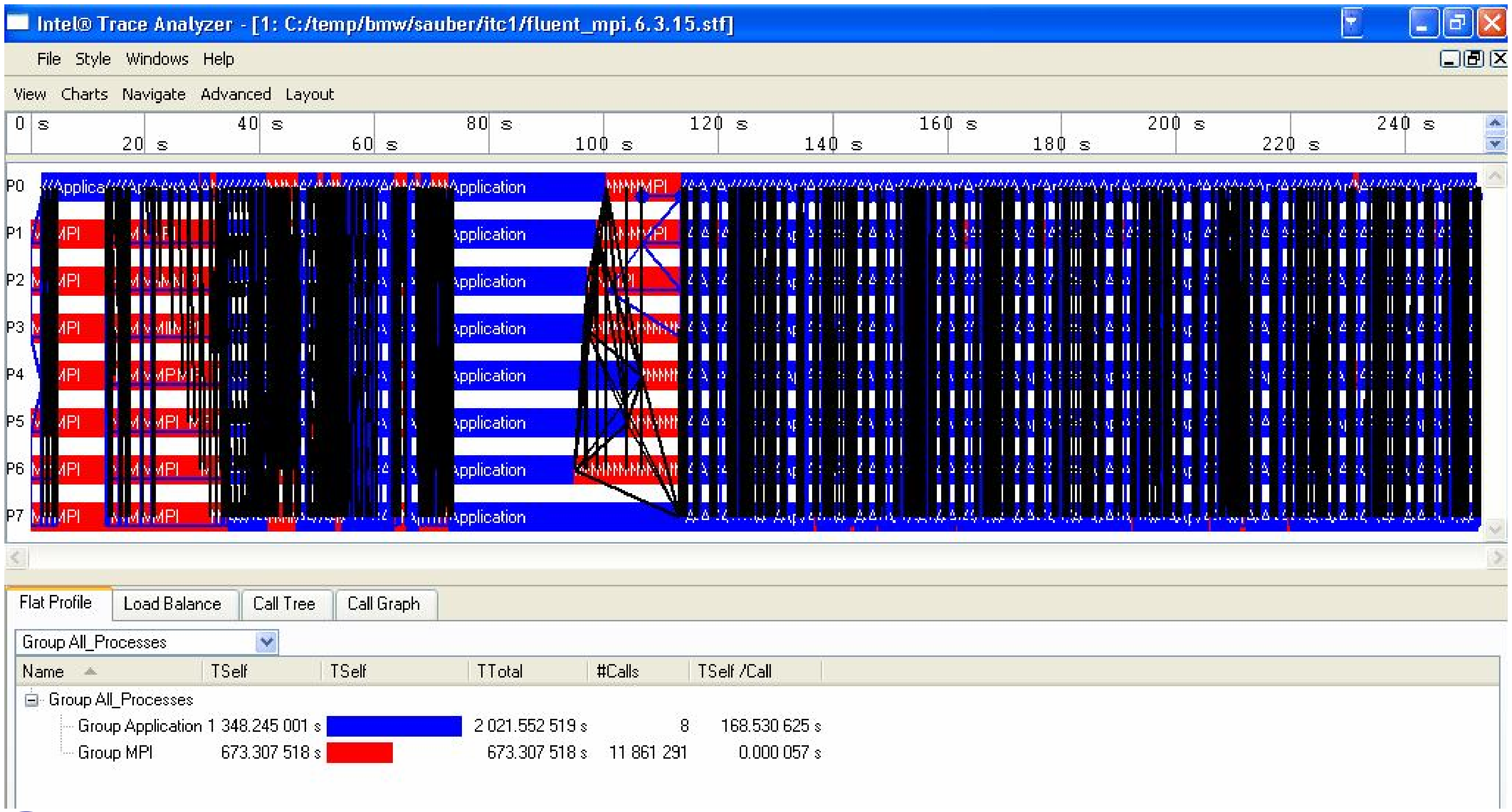
Components and Interaction



SSG

Case Study 1 – performance measurements

- CFD code
  - setup/iteration part
  - usullay > 100 iterations, but only 10 for performance benchmark

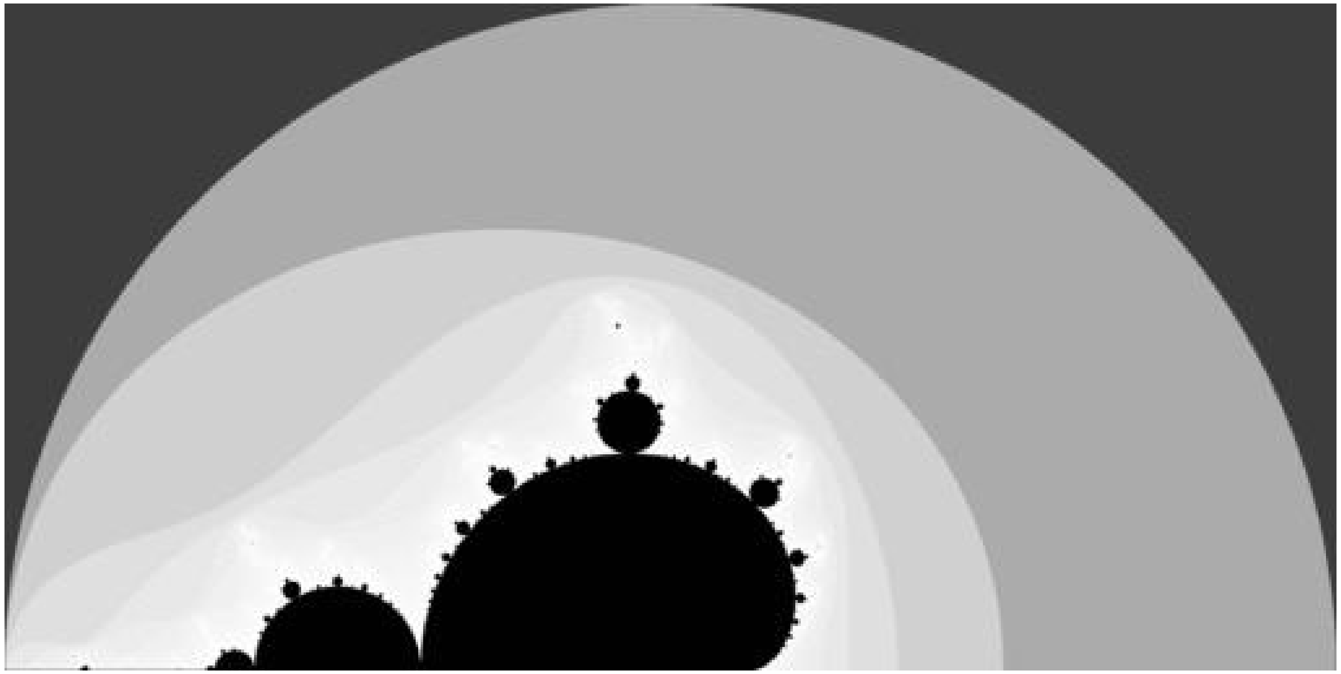


Name	TSelf	TSelf	TTotal	#Calls	TSelf /Call
Group All_Processes					
Group Application	1 348.245 001 s	🟦	2 021.552 519 s	8	168.530 625 s
Group MPI	673.307 518 s	🟥	673.307 518 s	11 861 291	0.000 057 s

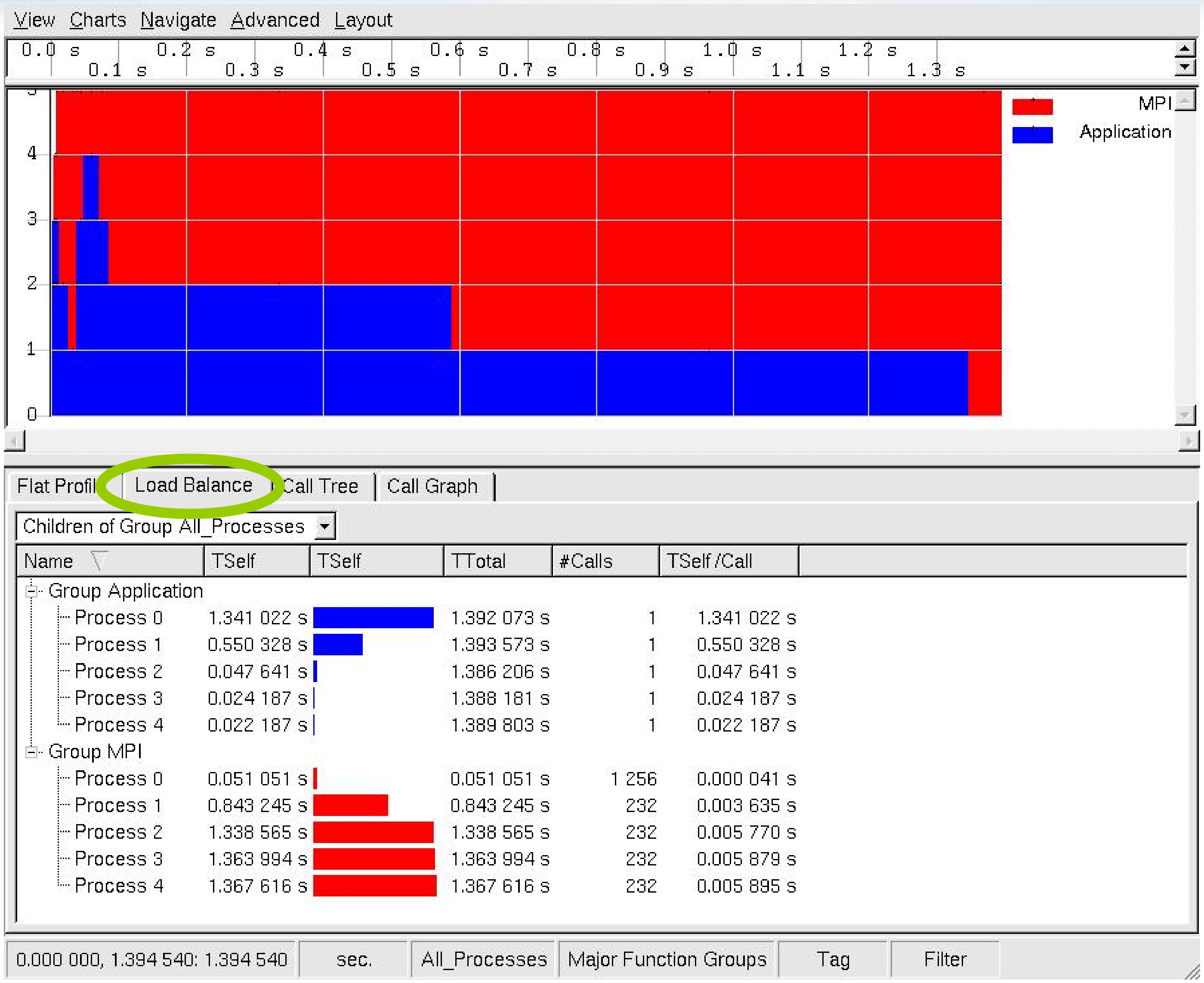
SSG

Case Study 2 - Load Imbalance

- Static partitioning assigns contiguous blocks of rows
  - Workload changes gradually with row-index!



Detecting Load Imbalance



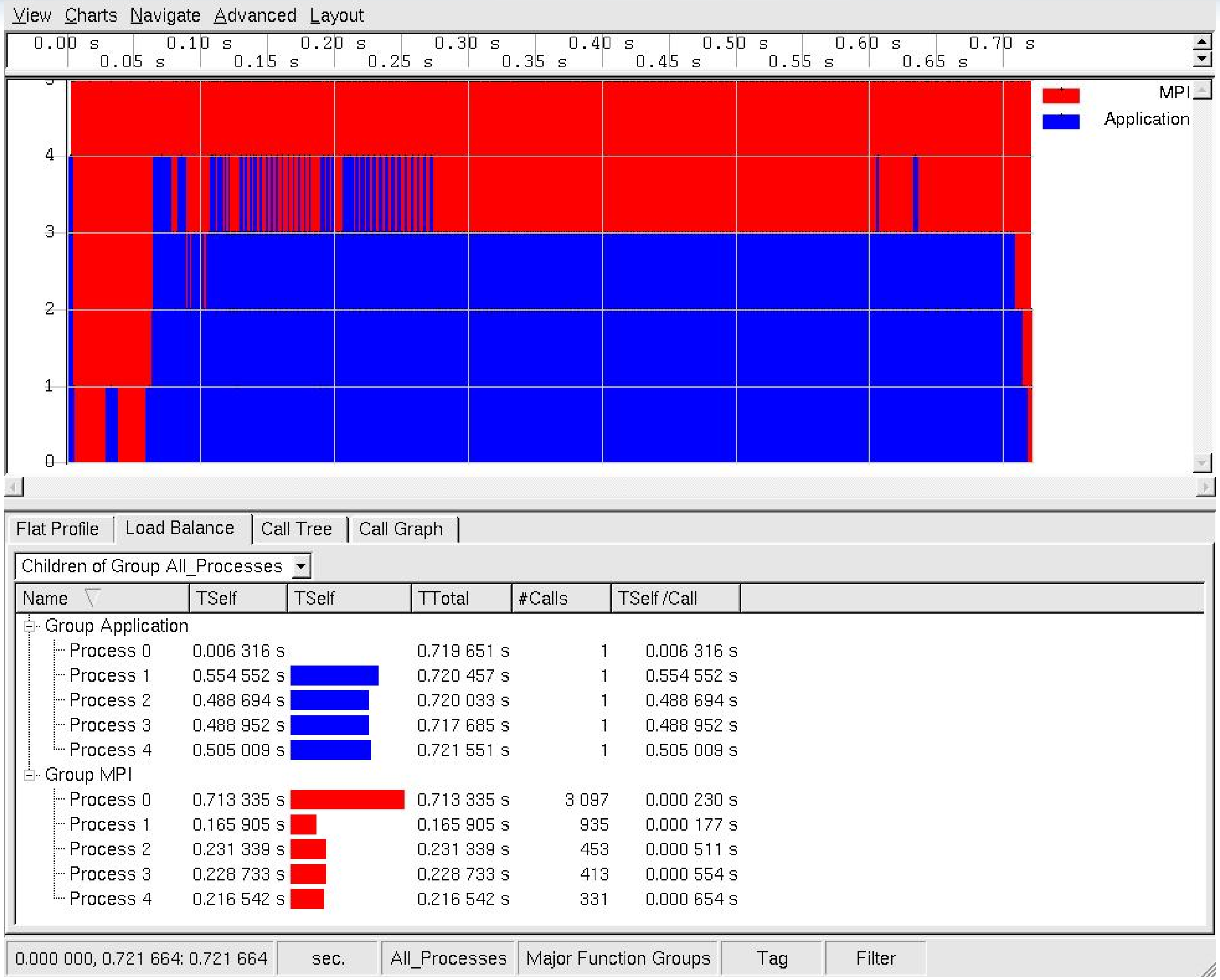
View Charts Navigate Advanced Layout
0.0 s 0.1 s 0.2 s 0.3 s 0.4 s 0.5 s 0.6 s 0.7 s 0.8 s 0.9 s 1.0 s 1.1 s 1.2 s 1.3 s
0 1 2 3 4
MPI
Application
Flat Profil
Load Balance
Call Tree
Call Graph
Children of Group All_Processes

Name	TSelf	TSelf	TTotal	#Calls	TSelf /Call
Group Application					
Process 0	1.341 022 s		1.392 073 s	1	1.341 022 s
Process 1	0.550 328 s		1.393 573 s	1	0.550 328 s
Process 2	0.047 641 s		1.386 206 s	1	0.047 641 s
Process 3	0.024 187 s		1.388 181 s	1	0.024 187 s
Process 4	0.022 187 s		1.389 803 s	1	0.022 187 s
Group MPI					
Process 0	0.051 051 s		0.051 051 s	1 256	0.000 041 s
Process 1	0.843 245 s		0.843 245 s	232	0.003 635 s
Process 2	1.338 565 s		1.338 565 s	232	0.005 770 s
Process 3	1.363 994 s		1.363 994 s	232	0.005 879 s
Process 4	1.367 616 s		1.367 616 s	232	0.005 895 s

0.000 000, 1.394 540: 1.394 540 | sec. | All_Processes | Major Function Groups | Tag | Filter

SSG

Solving Load Imbalance


View Charts Navigate Advanced Layout
0.00 s 0.05 s 0.10 s 0.15 s 0.20 s 0.25 s 0.30 s 0.35 s 0.40 s 0.45 s 0.50 s 0.55 s 0.60 s 0.65 s 0.70 s
0 1 2 3 4
🟥 MPI
🟦 Application
Flat Profile
Load Balance
Call Tree
Call Graph
Children of Group All_Processes

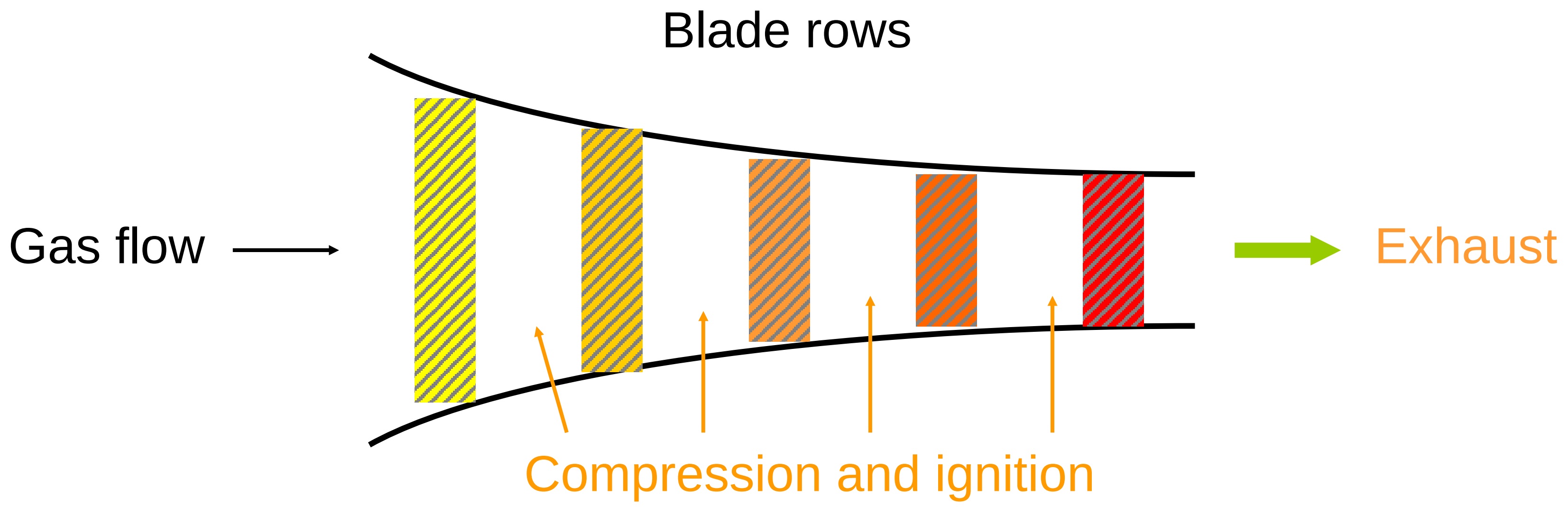
Name	TSelf	TSelf	TTotal	#Calls	TSelf /Call
Group Application					
Process 0	0.006 316 s		0.719 651 s	1	0.006 316 s
Process 1	0.554 552 s	🟦	0.720 457 s	1	0.554 552 s
Process 2	0.488 694 s	🟦	0.720 033 s	1	0.488 694 s
Process 3	0.488 952 s	🟦	0.717 685 s	1	0.488 952 s
Process 4	0.505 009 s	🟦	0.721 551 s	1	0.505 009 s
Group MPI					
Process 0	0.713 335 s	🟥	0.713 335 s	3 097	0.000 230 s
Process 1	0.165 905 s	🟥	0.165 905 s	935	0.000 177 s
Process 2	0.231 339 s	🟥	0.231 339 s	453	0.000 511 s
Process 3	0.228 733 s	🟥	0.228 733 s	413	0.000 554 s
Process 4	0.216 542 s	🟥	0.216 542 s	331	0.000 654 s

0.000 000, 0.721 664: 0.721 664 | sec. | All_Processes | Major Function Groups | Tag | Filter

SSG

Case Study 3 – programming error

- Each blade row is independent MPI simulation
- Blade row simulations run concurrently with file-level synchronization
- Goal: Optimize blade row simulations



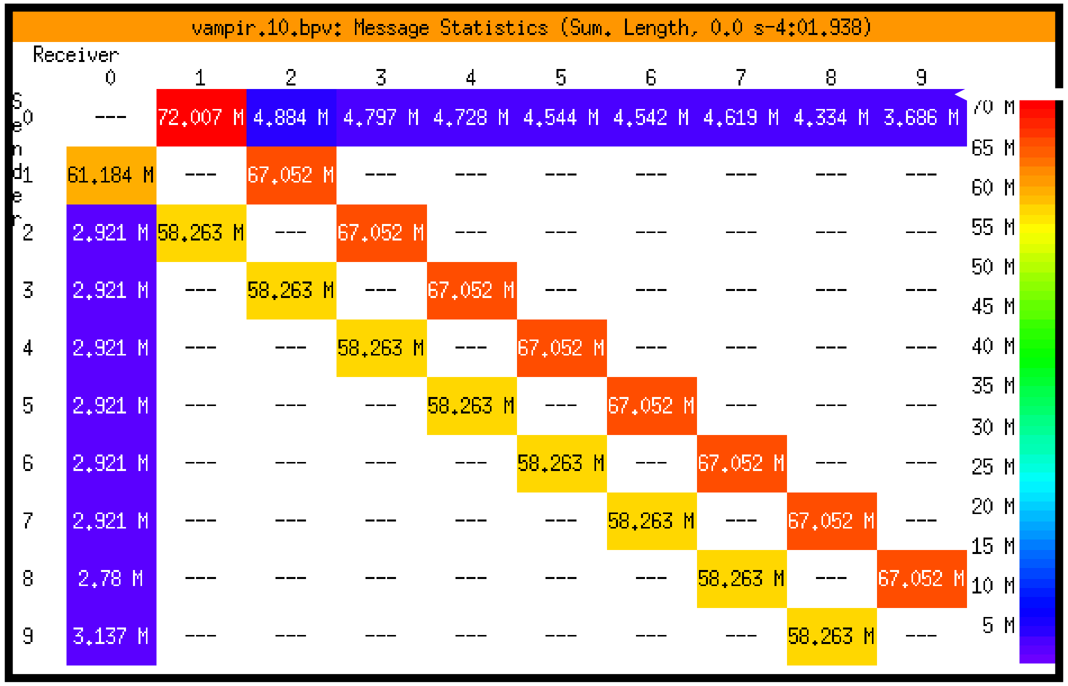
SSG

Three Customer Workloads

- Test problems 1 and 2
  - Large and small blade row simulations
  - Scale well
  - Represent current production requirements
- Test problem 3
  - Bisected blade row simulation
  - Scales poorly
  - Represents future production requirements

SSG

ITC Result Case 1 & 2


vampir.10.bpv: Message Statistics (Sum, Length, 0.0 s-4:01.938)

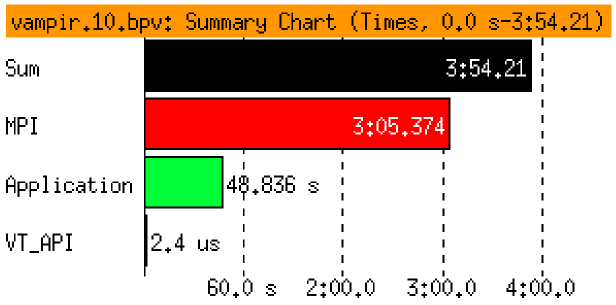
Sender \ Receiver	0	1	2	3	4	5	6	7	8	9
0	---	72.007 M	4.884 M	4.797 M	4.728 M	4.544 M	4.542 M	4.619 M	4.334 M	3.686 M
1	61.184 M	---	67.052 M	---	---	---	---	---	---	---
2	2.921 M	58.263 M	---	67.052 M	---	---	---	---	---	---
3	2.921 M	---	58.263 M	---	67.052 M	---	---	---	---	---
4	2.921 M	---	---	58.263 M	---	67.052 M	---	---	---	---
5	2.921 M	---	---	---	58.263 M	---	67.052 M	---	---	---
6	2.921 M	---	---	---	---	58.263 M	---	67.052 M	---	---
7	2.921 M	---	---	---	---	---	58.263 M	---	67.052 M	---
8	2.78 M	---	---	---	---	---	---	58.263 M	---	67.052 M
9	3.137 M	---	---	---	---	---	---	---	58.263 M	---

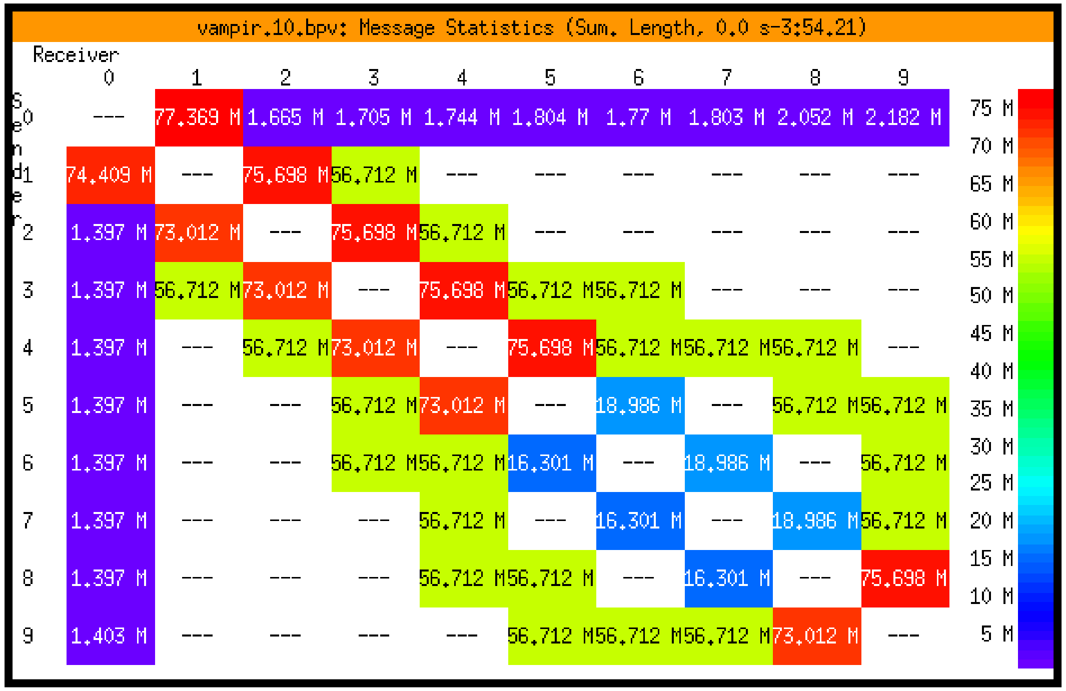
Cases 1 and 2, which scale well, have a simple nearest-neighbor communication scheme

SSG

ITC Result Case 3

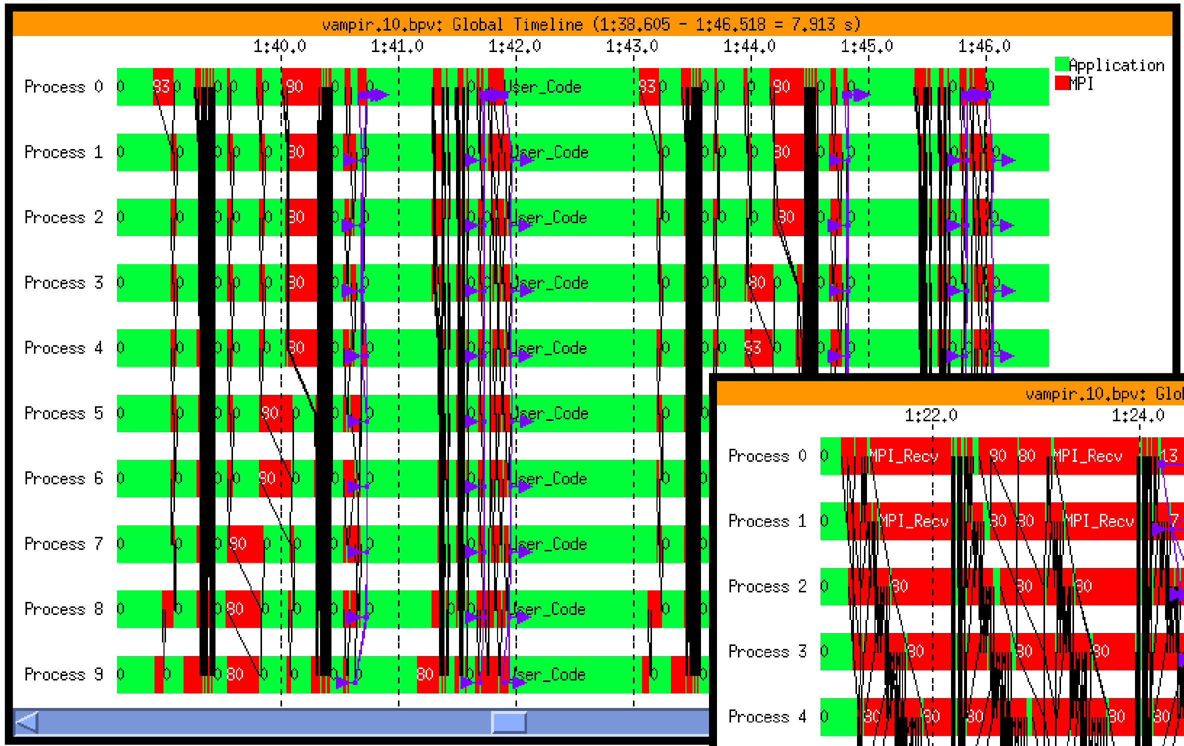
Case 3, which scales poorly, has a more complex communication scheme

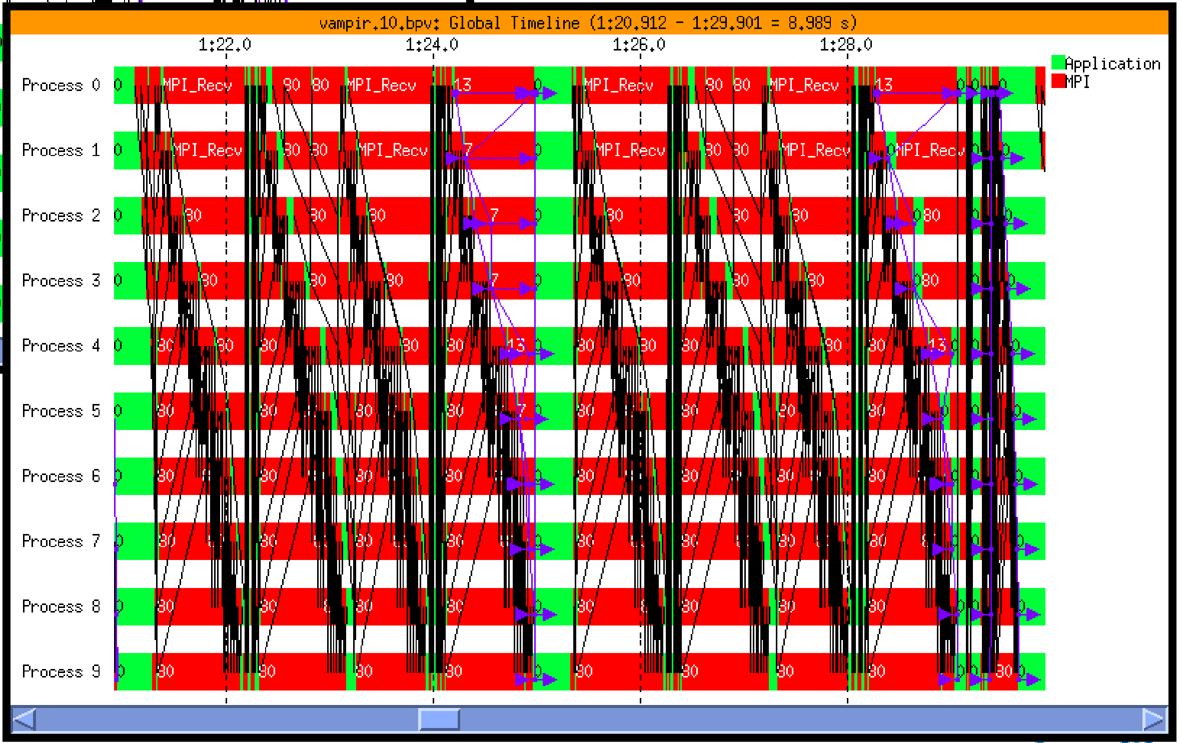



vampir.10.bpv: Message Statistics (Sum, Length, 0.0 s-3:54.21)

Sender \ Receiver	0	1	2	3	4	5	6	7	8	9
0	---	77.369 M	1.665 M	1.705 M	1.744 M	1.804 M	1.77 M	1.803 M	2.052 M	2.182 M
1	74.409 M	---	75.698 M	56.712 M	---	---	---	---	---	---
2	1.397 M	73.012 M	---	75.698 M	56.712 M	---	---	---	---	---
3	1.397 M	56.712 M	73.012 M	---	75.698 M	56.712 M	56.712 M	---	---	---
4	1.397 M	---	56.712 M	73.012 M	---	75.698 M	56.712 M	56.712 M	56.712 M	---
5	1.397 M	---	---	56.712 M	73.012 M	---	18.986 M	---	56.712 M	56.712 M
6	1.397 M	---	---	56.712 M	56.712 M	16.301 M	---	18.986 M	---	56.712 M
7	1.397 M	---	---	---	56.712 M	---	16.301 M	---	18.986 M	56.712 M
8	1.397 M	---	---	---	56.712 M	56.712 M	---	16.301 M	---	75.698 M
9	1.403 M	---	---	---	---	56.712 M	56.712 M	56.712 M	73.012 M	---

MPI Communication Timeline



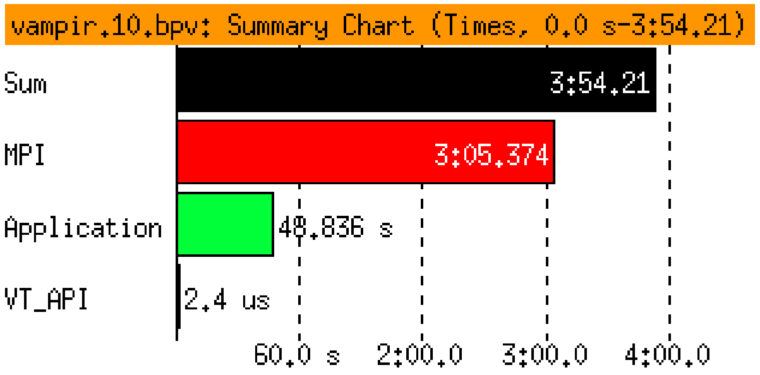


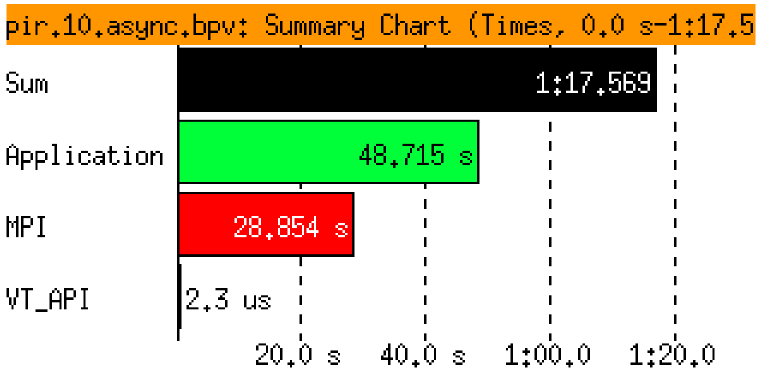
SSG

Optimized Communication

Before

Case 3 with communication bottleneck





After

Case 3 with optimized communication

Compiler and MPI tuning yielded 2.7X speedup

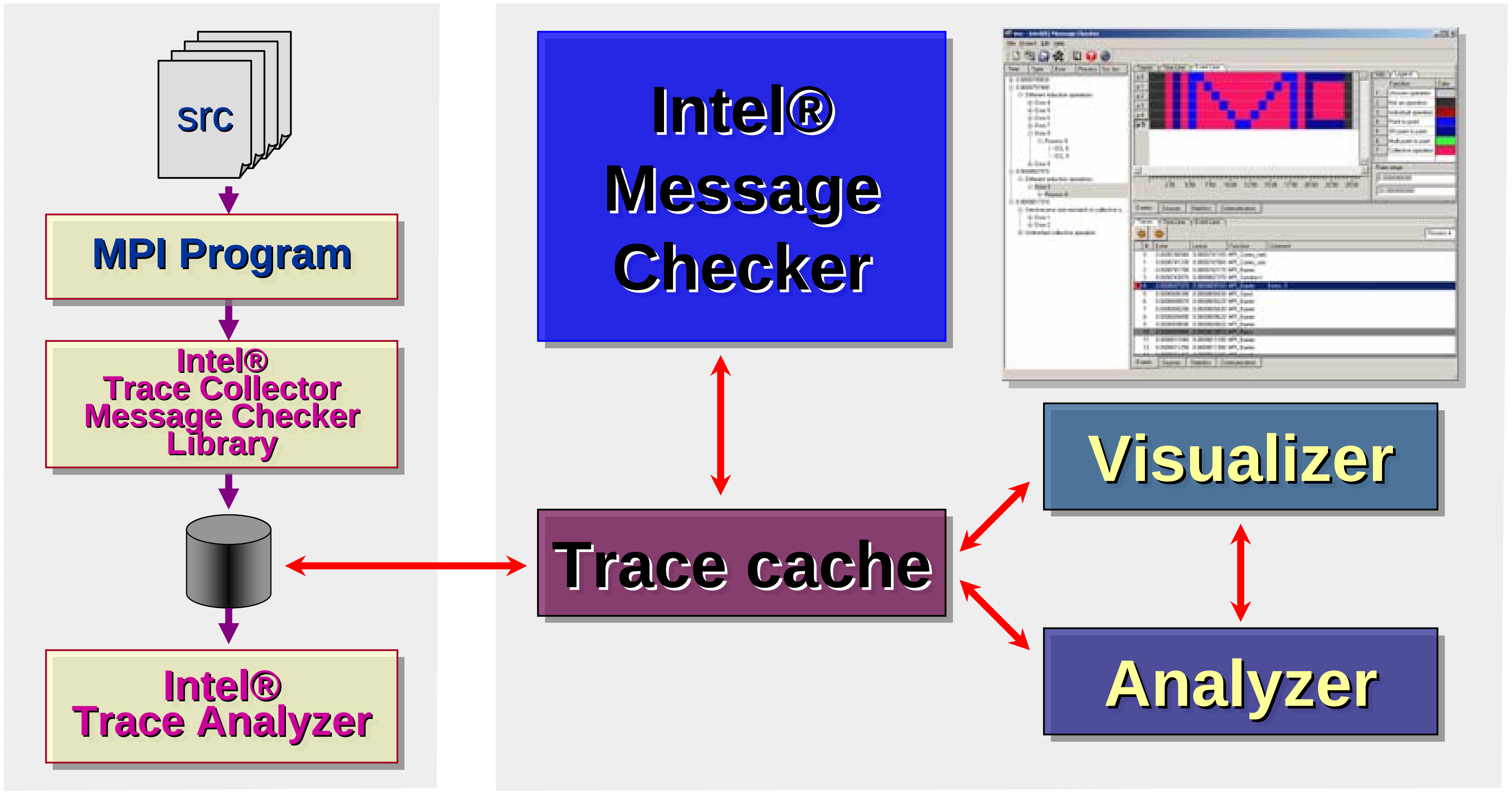
SSG

Intel® Message Checker

A Unique Tool for MPI Correctness Analysis

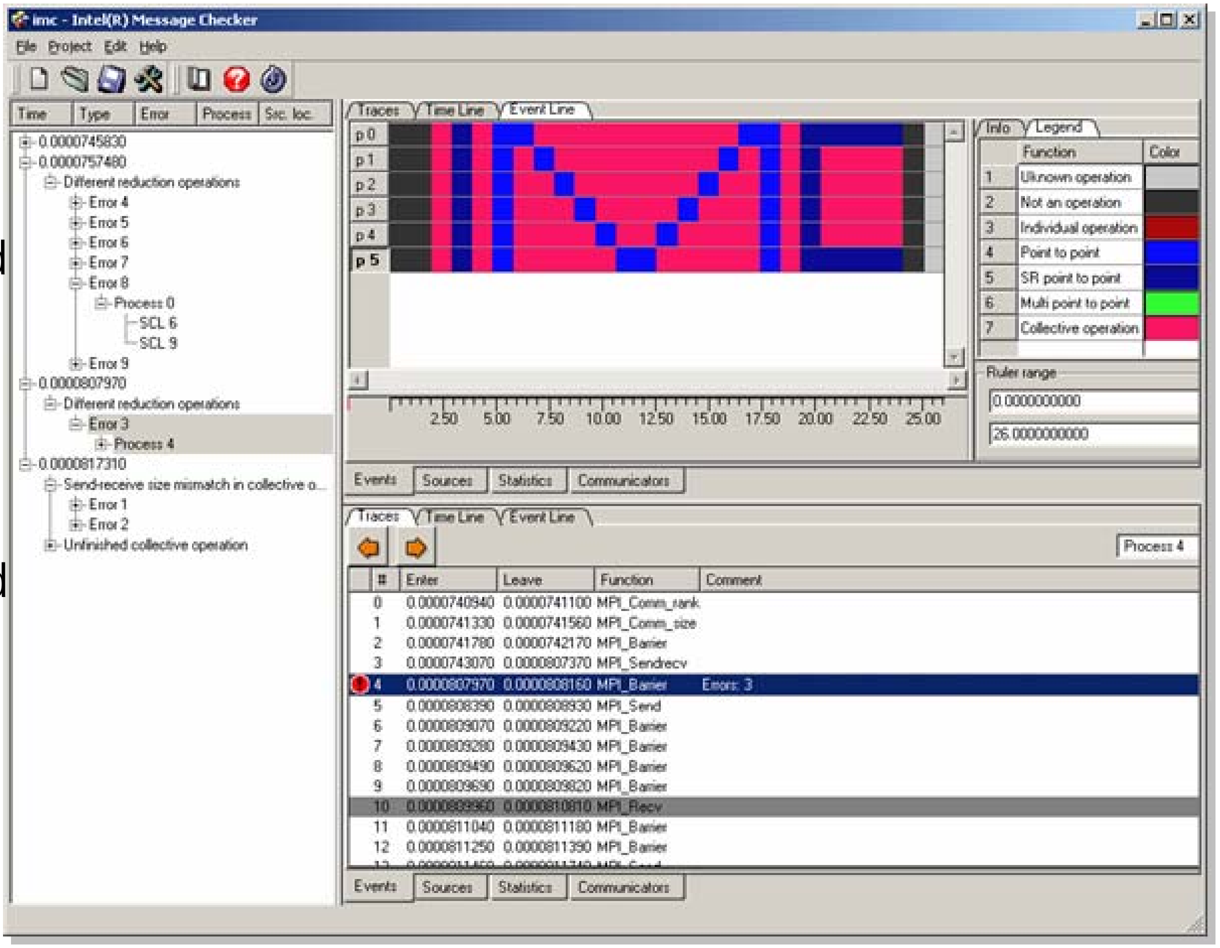
SSG
Software Solutions Group

Intel® Message Checker



Message Checker GUI

- Error view with different classifiers (#, type, process, time, source location)
- Source view with error context
- Trace view over time, events, and function calls
- Task statistics (general, communicator, and per-process)
- Communicator view
- Stack information
- Windows are synchronized based on user actions



```
0.0000745830
0.0000757480
   Different reduction operations
      Error 4
      Error 5
      Error 6
      Error 7
      Error 8
         Process 0
            SCL 6
            SCL 9
      Error 9
0.0000807970
   Different reduction operations
      Error 3
         Process 4
0.0000817310
   Send-receive size mismatch in collective o...
      Error 1
      Error 2
   Unfinished collective operation
```

	Function	Color
1	Unknown operation	⬜
2	Not an operation	⬛
3	Individual operation	🟥
4	Point to point	🟦
5	SR point to point	🟦
6	Multi point to point	🟩
7	Collective operation	🟥

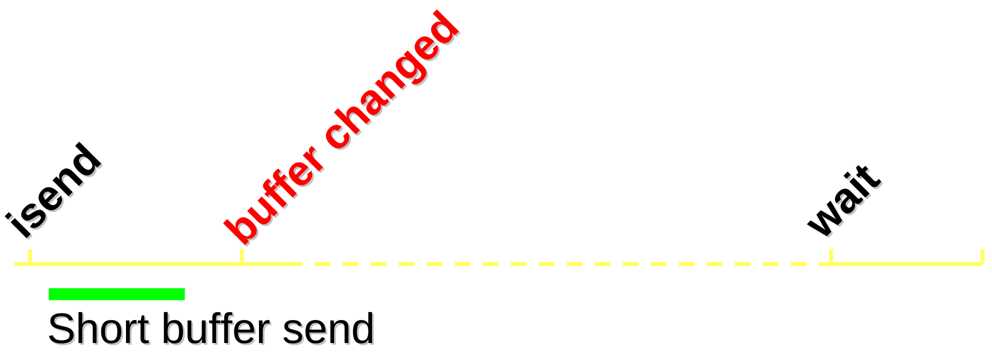
#	Enter	Leave	Function	Comment
0	0.0000740940	0.0000741100	MPI_Comm_rank	
1	0.0000741330	0.0000741560	MPI_Comm_size	
2	0.0000741780	0.0000742170	MPI_Barrier	
3	0.0000743070	0.0000807370	MPI_Sendrecv	
● 4	0.0000807970	0.0000808160	MPI_Barrier	Errors: 3
5	0.0000808390	0.0000808930	MPI_Send	
6	0.0000809070	0.0000809220	MPI_Barrier	
7	0.0000809280	0.0000809430	MPI_Barrier	
8	0.0000809490	0.0000809620	MPI_Barrier	
9	0.0000809690	0.0000809820	MPI_Barrier	
10	0.0000809960	0.0000810810	MPI_Recv	
11	0.0000811040	0.0000811180	MPI_Barrier	
12	0.0000811250	0.0000811390	MPI_Barrier	

SSG

Example: NAS Parallel Benchmarks

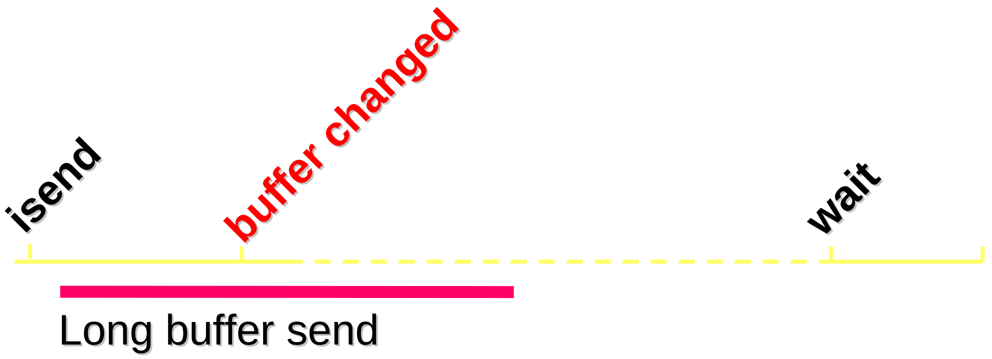
Test was changed for buffer modification after non-blocking send

Class S:



The program produces correct result
(data were changed after send), but analyzer detects "Wrong send checksum"

Class A:



Incorrect program result
(data were changed before send completed)
Diagnostics:
Wrong send checksum
Wrong recv checksum

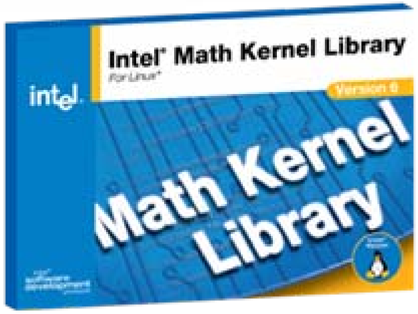
SSG

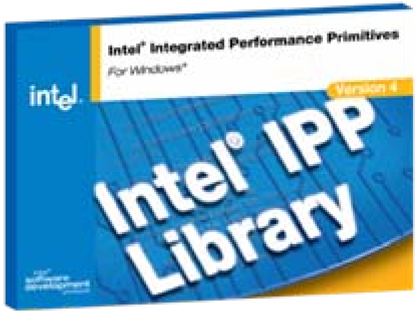
Performance Libraries

SSG
Software Solutions Group

Intel® Performance Libraries

- **Intel® Math Kernel Library**
  - **A set of highly optimized numerical processing functions for math, scientific, engineering and financial applications**
    - **Large-scale numerical applications**
    - **Workstations & server platform focus**
- **Intel® Integrated Performance Primitives**
  - **A library of signal, image, graphic, multimedia and numeric processing functions**
    - **Real-time and interactive applications**
- **Intel® Optimized XML Library**
  - **Optimized routines for XML parsing**
  - **To be released in Q1/2005**
- **Intel® Computer Vision Library**
  - **For Intel XScale® architecture**
  - **Available for free from developer.intel.com**





SSG

intel®

Leap ahead™