MPI State Profiling

CScADS Petascale Performance Workshop
Snowbird, UT, July 21, 2008

Marty Itzkowitz
Sun Studio Performance Analyzer Team
Sun Microsystems Inc.
`marty.itzkowitz@sun.com`

Outline

- MPI Model, and Sun Studio `collect`

  MPI version: Sun's CT 8, based on Open MPI
- MPI State Profiling
- Experiments and Data
  - Functions
  - Callers and Callees
  - Annotated Source
- Correlating the Data
- Pros and Cons

Work by Oleg Mazurov, Eugene Loh, Yukon Maruyama, Terry Dontje, and Marty Itzkowitz

MPI User-model

- Multiple processes run simultaneously
  - Processes communicate by sending messages
  - May be hybrid: OpenMP and Open MPI
  - May be SIMD or MIMD
- Implementation
  - Processes launched with **`mpirun`**
    - Either on same node, or across cluster of nodes
  - MPI Data collected with **`collect -M on mpirun ...`**
    - Insert **`--`** between **`mpirun`** and its target(s)
  - So-called “founder” experiment has message trace
    - Each process has a “subexperiment” inside founder experiment

MPI State Profiling

- Three states:
  - Not-in-MPI, MPI-Work, MPI-Wait
  - More states could be added (*e.g.*, wait for specific events)
- API to read state
  - Thread-specific, asynchronous-signal-safe
- Called whenever clock profile signal is received
  - Solaris – no matter what thread is doing
  - Linux – only when thread is on CPU
- Callstack recorded with data

MPI State Transitions

- Entry to library: Not-in-MPI ==> MPI-Work
- Start of waiting: MPI-Work ==> MPI-Wait
  - Whether busy waiting, or giving up the CPU
- End of waiting: MPI-Wait ==> MPI-Work
- Return from library: MPI-Work ==> Not-in-MPI

MPI State Metrics

- If User-CPU and MPI-Work:
  - Increment MPI Work Time
- If not User-CPU, or User-CPU and MPI-Wait:
  - Increment MPI Wait Time
- Metrics attributed to frames in the callstack
  - Map to functions
  - Map to source lines

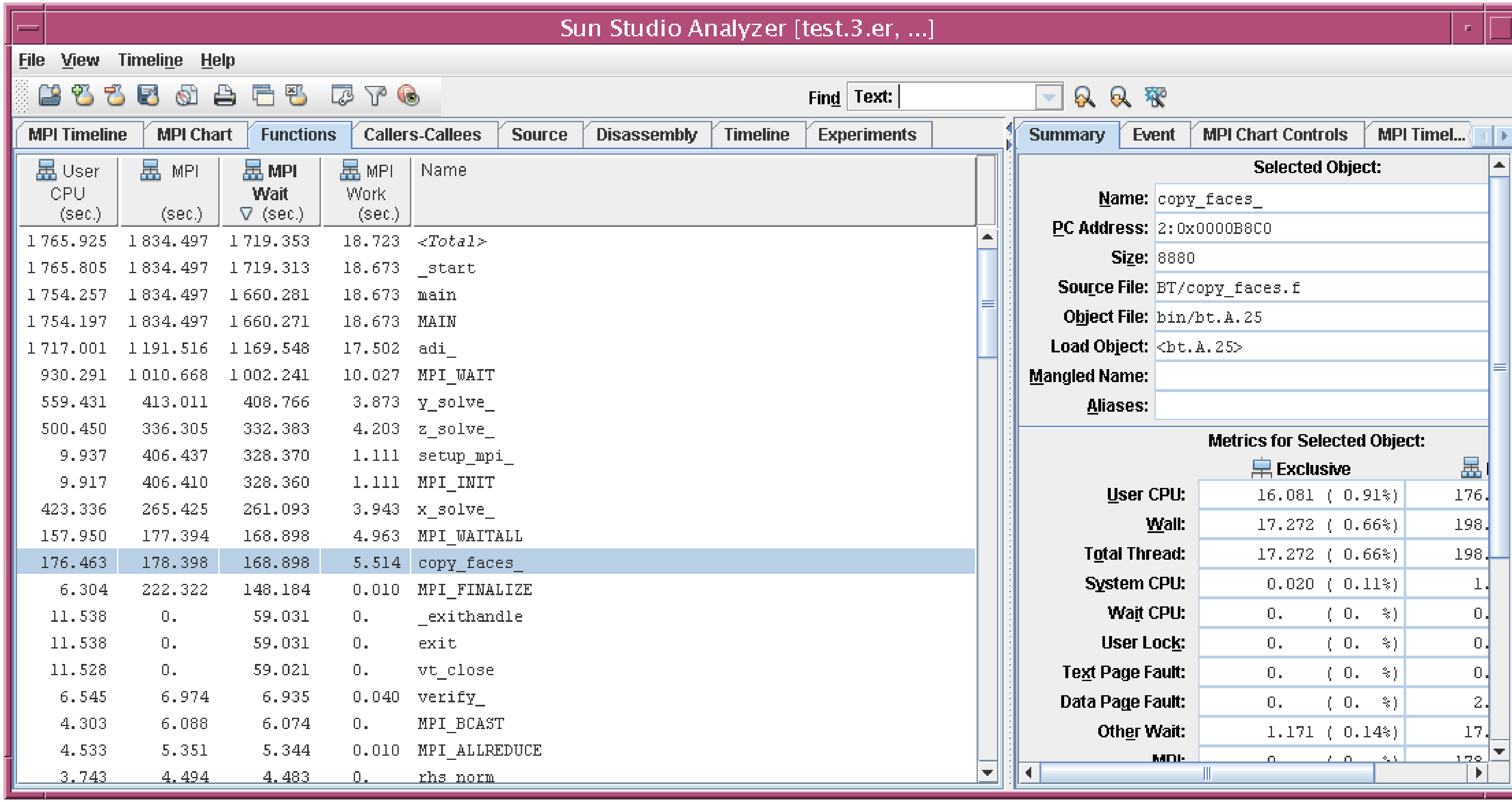
MPI Tracing

- Trace calls to MPI library APIs
  - Open Source VampirTrace
  - Metrics
    - MPI-Sends, MPI-Recvs, MPI-Other
    - MPI-bytes-sent, MPI-bytes-recvd
    - MPI-time
  - Callstack recorded with each event
  - Also MPI Timeline, MPI Charts
    - Sends and receives matched in MPI Timeline

MPI Example: `bt.A.25`

- NAS Parallel Benchmark BT
  - Run 25 MPI processes
- `collect -M on mpirun -np 25 ...`
  - Collect MPI API traces from all 25 processes
    - Open Source VampirTrace
  - Collect MPI State data with clock profiles

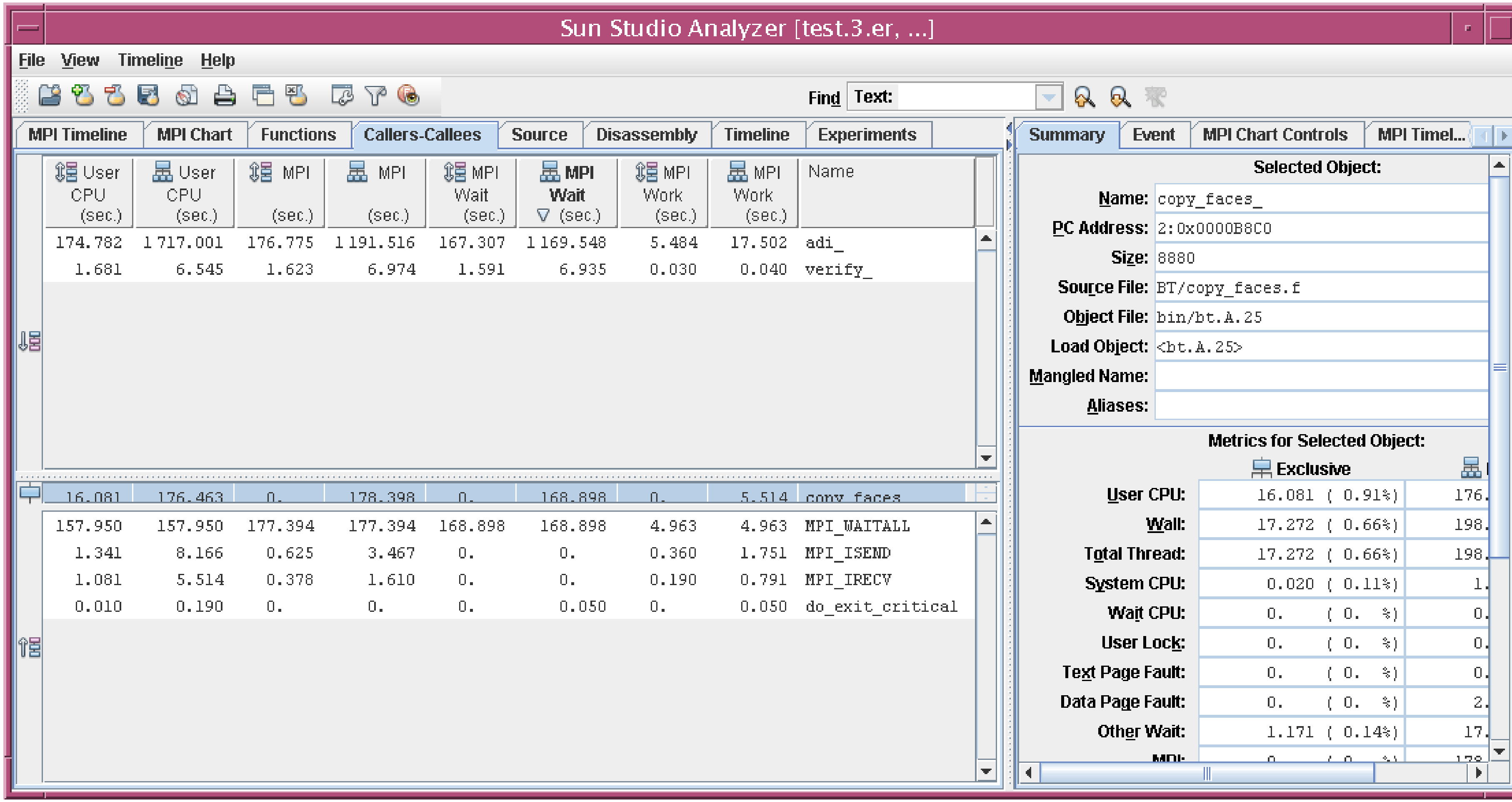
Function List



User CPU (sec.)	MPI (sec.)	MPI Wait (sec.)	MPI Work (sec.)	Name
1 765.925	1 834.497	1 719.353	18.723	<Total>
1 765.805	1 834.497	1 719.313	18.673	_start
1 754.257	1 834.497	1 660.281	18.673	main
1 754.197	1 834.497	1 660.271	18.673	MAIN
1 717.001	1 191.516	1 169.548	17.502	adi_
930.291	1 010.668	1 002.241	10.027	MPI_WAIT
559.431	413.011	408.766	3.873	y_solve_
500.450	336.305	332.383	4.203	z_solve_
9.937	406.437	328.370	1.111	setup_mpi_
9.917	406.410	328.360	1.111	MPI_INIT
423.336	265.425	261.093	3.943	x_solve_
157.950	177.394	168.898	4.963	MPI_WAITALL
176.463	178.398	168.898	5.514	copy_faces_
6.304	222.322	148.184	0.010	MPI_FINALIZE
11.538	0.	59.031	0.	_exithandle
11.538	0.	59.031	0.	exit
11.528	0.	59.021	0.	vt_close
6.545	6.974	6.935	0.040	verify_
4.303	6.088	6.074	0.	MPI_BCAST
4.533	5.351	5.344	0.010	MPI_ALLREDUCE
3.743	4.494	4.483	0.	rhs_norm

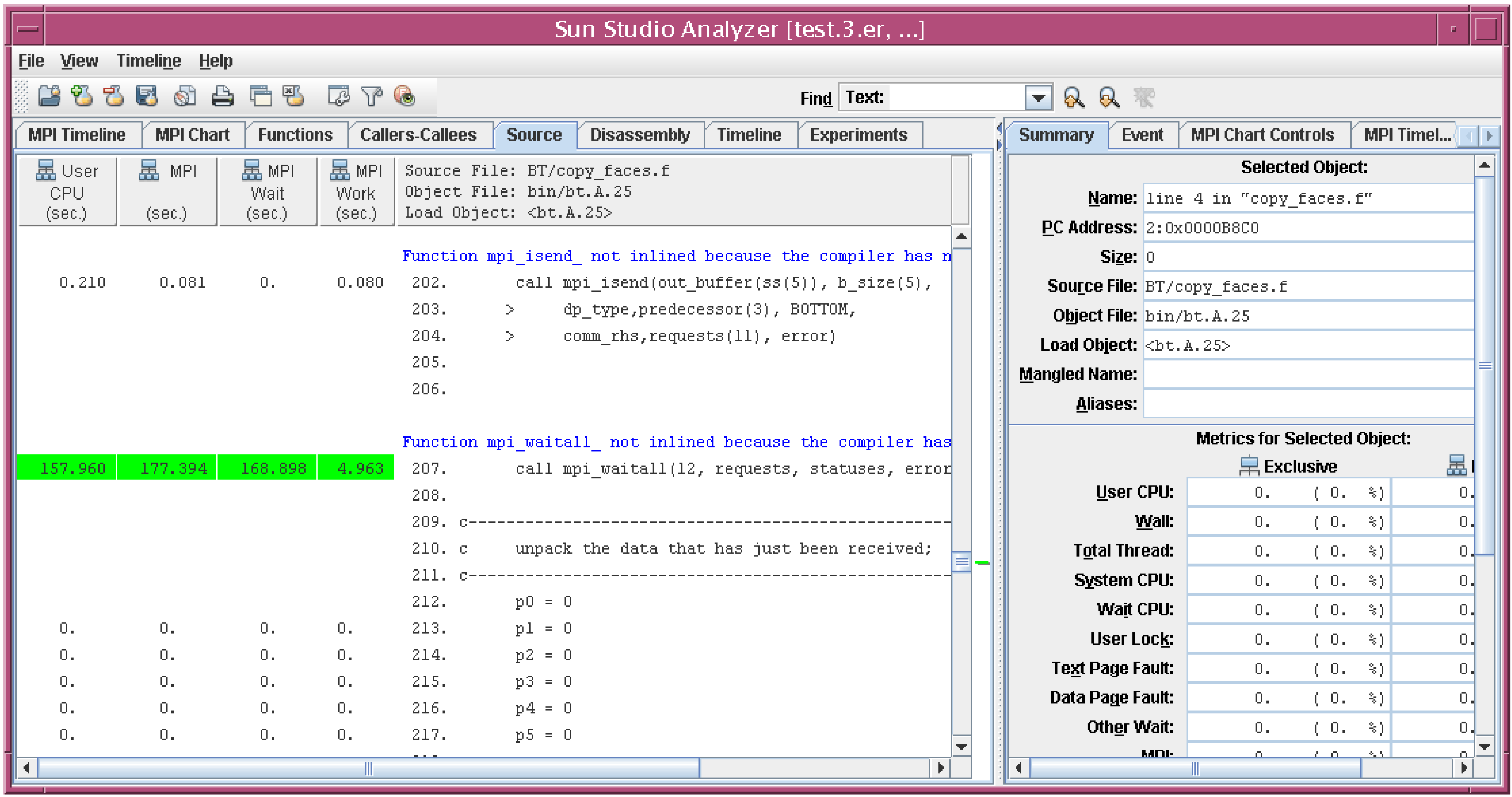
Sorted by MPI Wait time

Caller-Callees



Navigate with MPI Wait time to find delays

Source Display

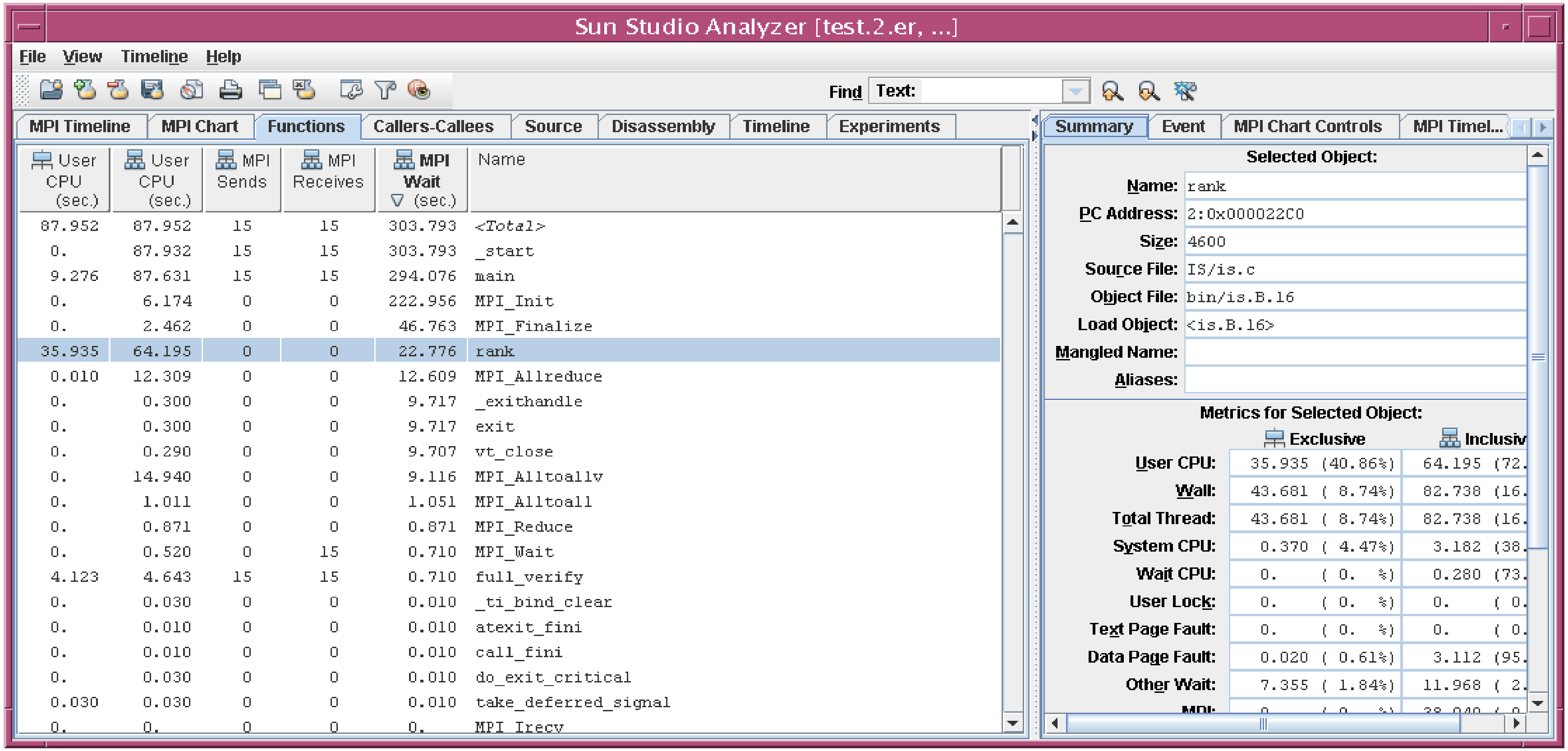


Find lines where MPI Wait time is high

MPI Example: `is.B.16`

- NAS Parallel Benchmark IS
  - Run 16 MPI processes
  - **`collect -M on mpirun -np 16 ...`**
  - Collect MPI API traces from all sixteen processes
    - Open Source Vampir Trace
  - Collect MPI State data with clock profiles

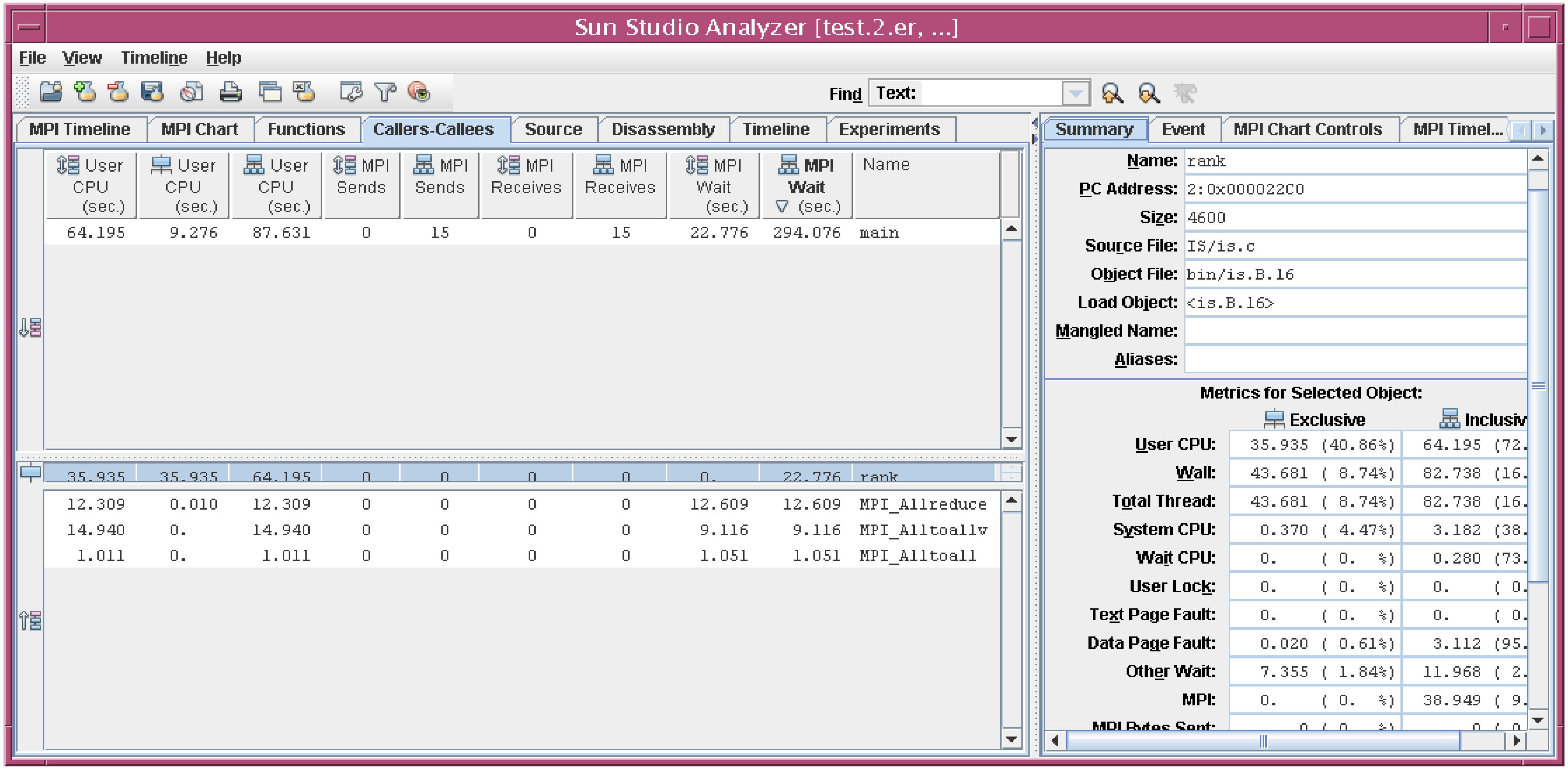
Function List



Sorted by MPI Wait time – significant wait time in MPI_Alltoallv, MPI_Alltoall, MPI_Allreduce

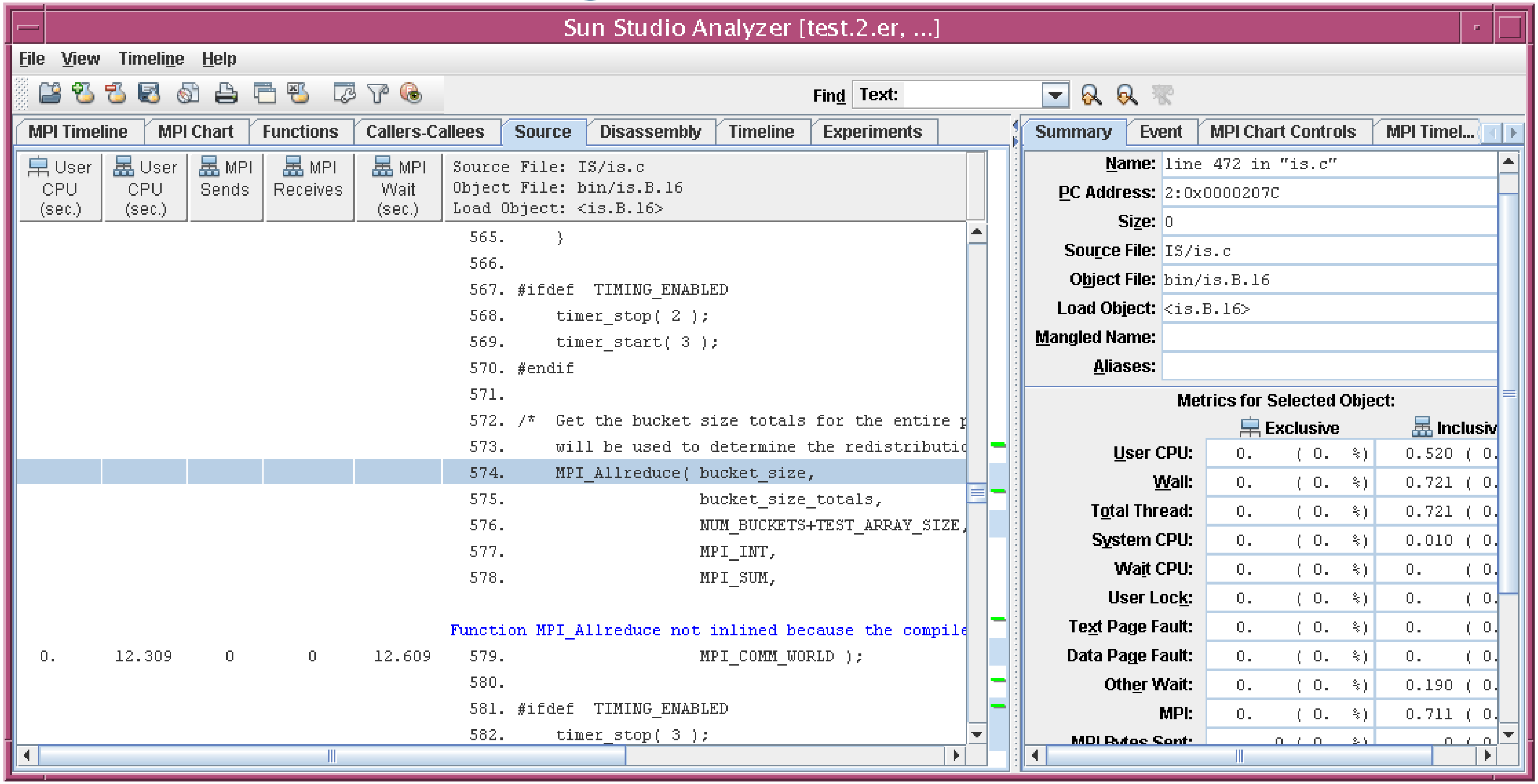
Also, in MPI_Init and MPI_Finalize

Caller-Callees



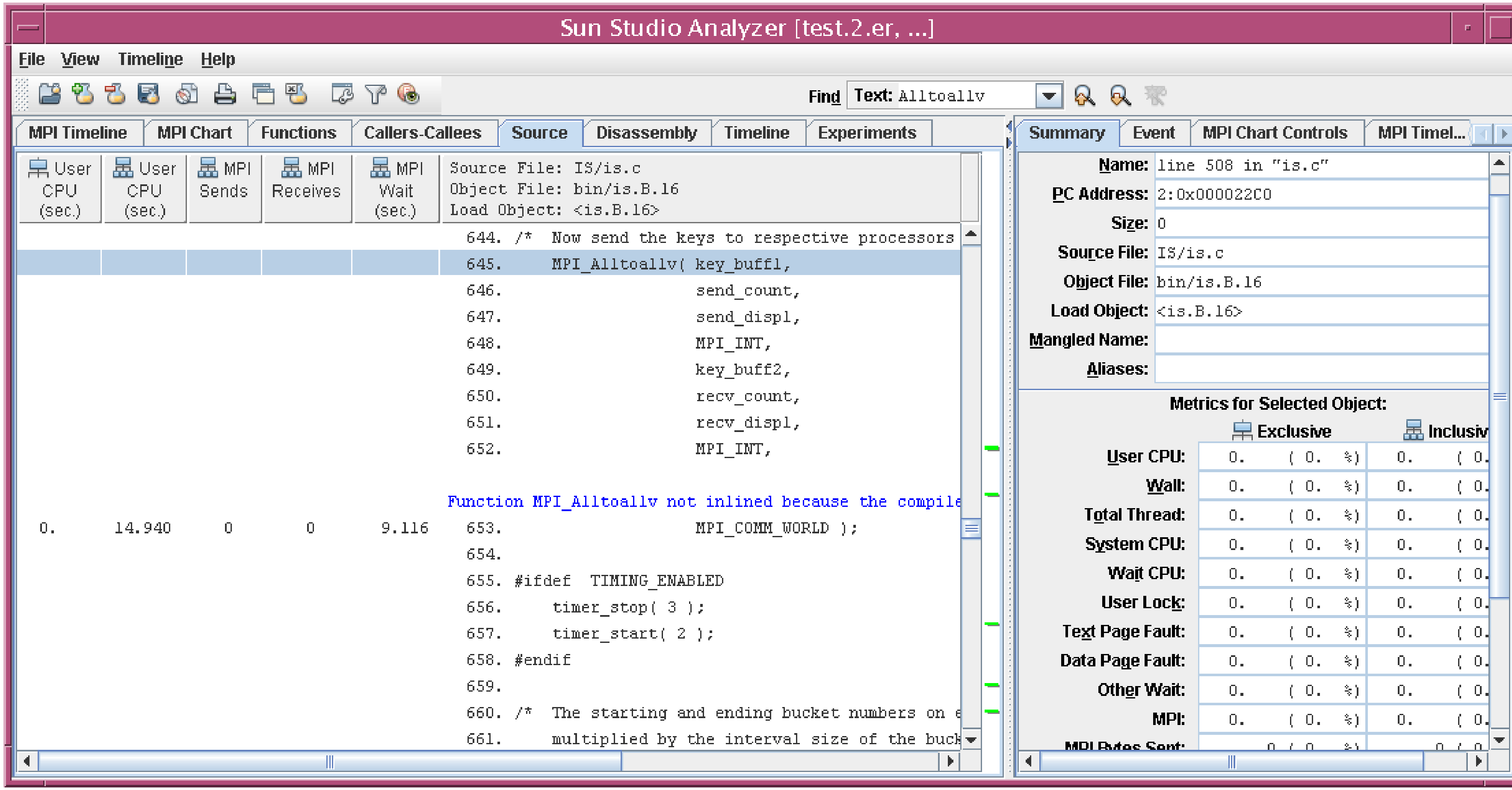
Navigate with MPI Wait time to find delays

Source Display, I



Find lines where MPI Wait time is high:
I. MPI_Allreduce()

Source Display, II



Find lines where MPI Wait time is high:
2. MPI_Alltoallv()

Correlating the Data

- All trace and profile records have callstacks
  - Suppress frames below API-layer of MPI
- Aggregate Messages into Transactions
  - Based on common send/recv callstacks
- Profile records with non-zero MPI-Wait / MPI Work time
  - Must be inside MPI
  - Will match a send or recv trace record callstack
- Attribute MPI-Wait / MPI-Work time to Transactions
  - Both sender and receiver times
  - Select expensive transactions to display

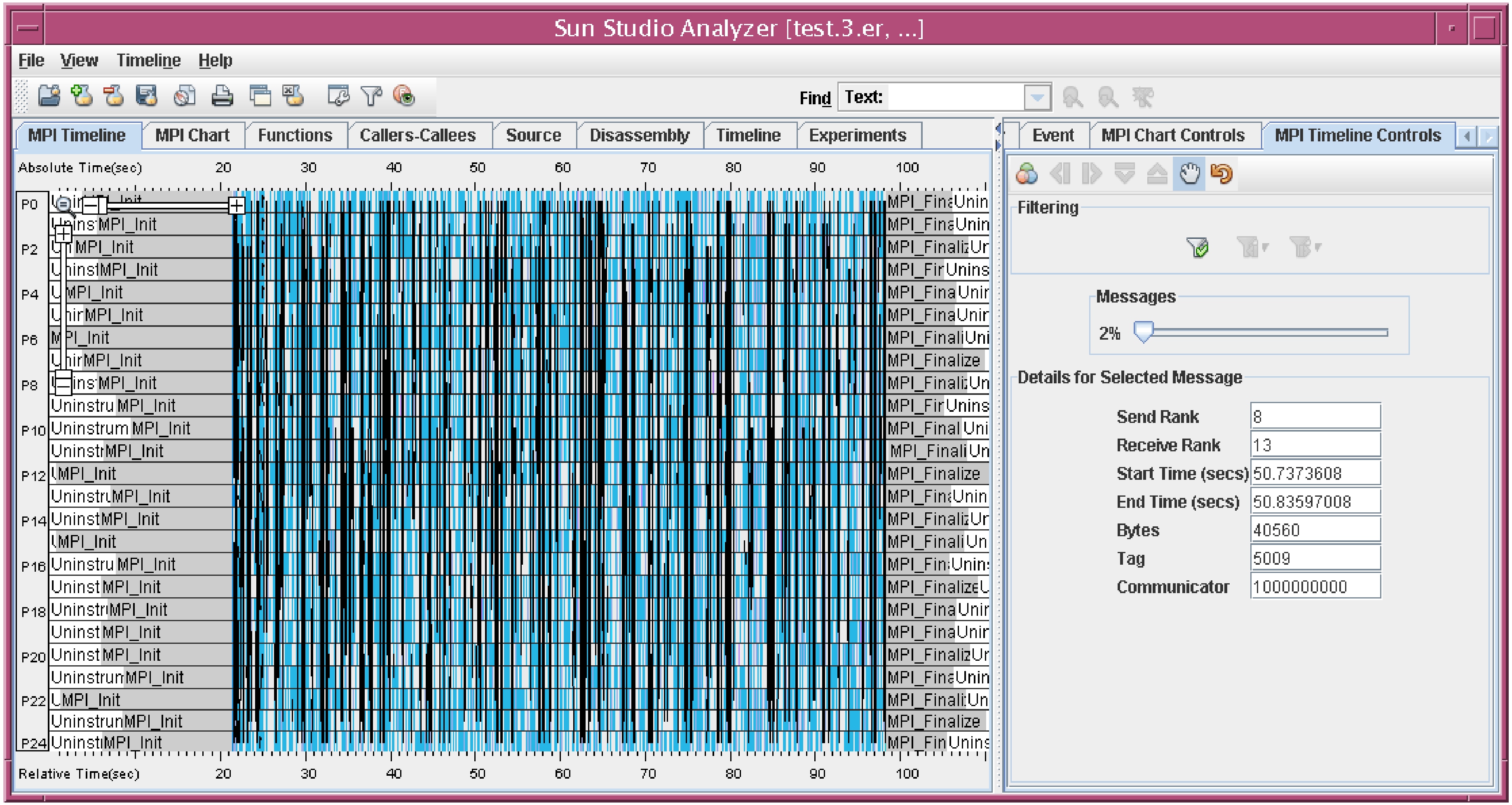
Advantages and Disadvantages

- Advantages:
  - Easy implementation in Open MPI libraries
  - Scalability: profiling allows easy control of data volume
  - Direct navigation to source: no hunting through messages
- Disadvantages
  - Data is statistical, may miss anomalies
  - Dilation (in current prototype)
    - ~ 0.05 – 0.15 µsecs. in latency for small (0-, 8-byte) messages
- We will offer source to Open MPI Community

MPI State Profiling

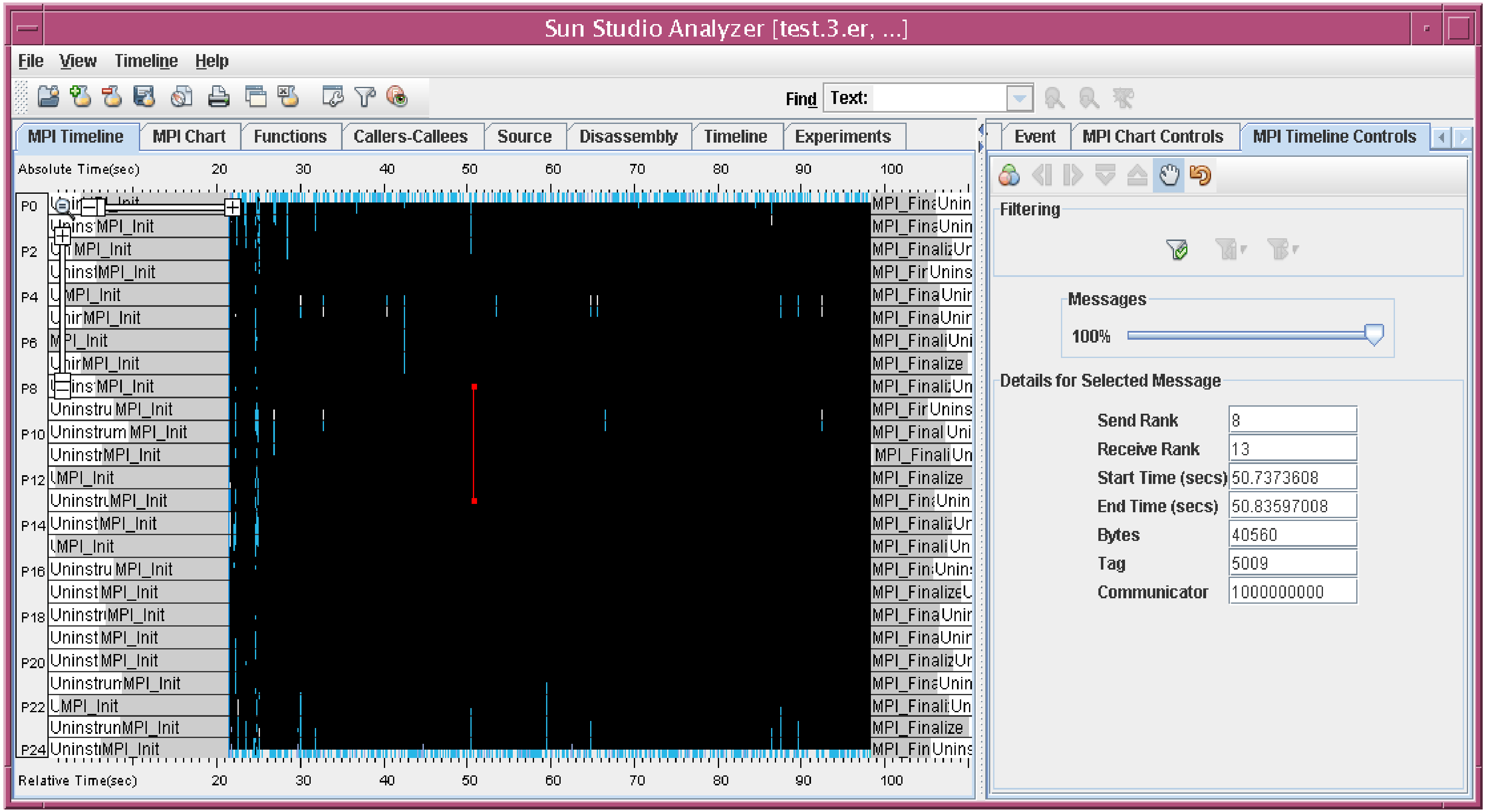
Supplemental Information

BT -- MPI-Timeline, I



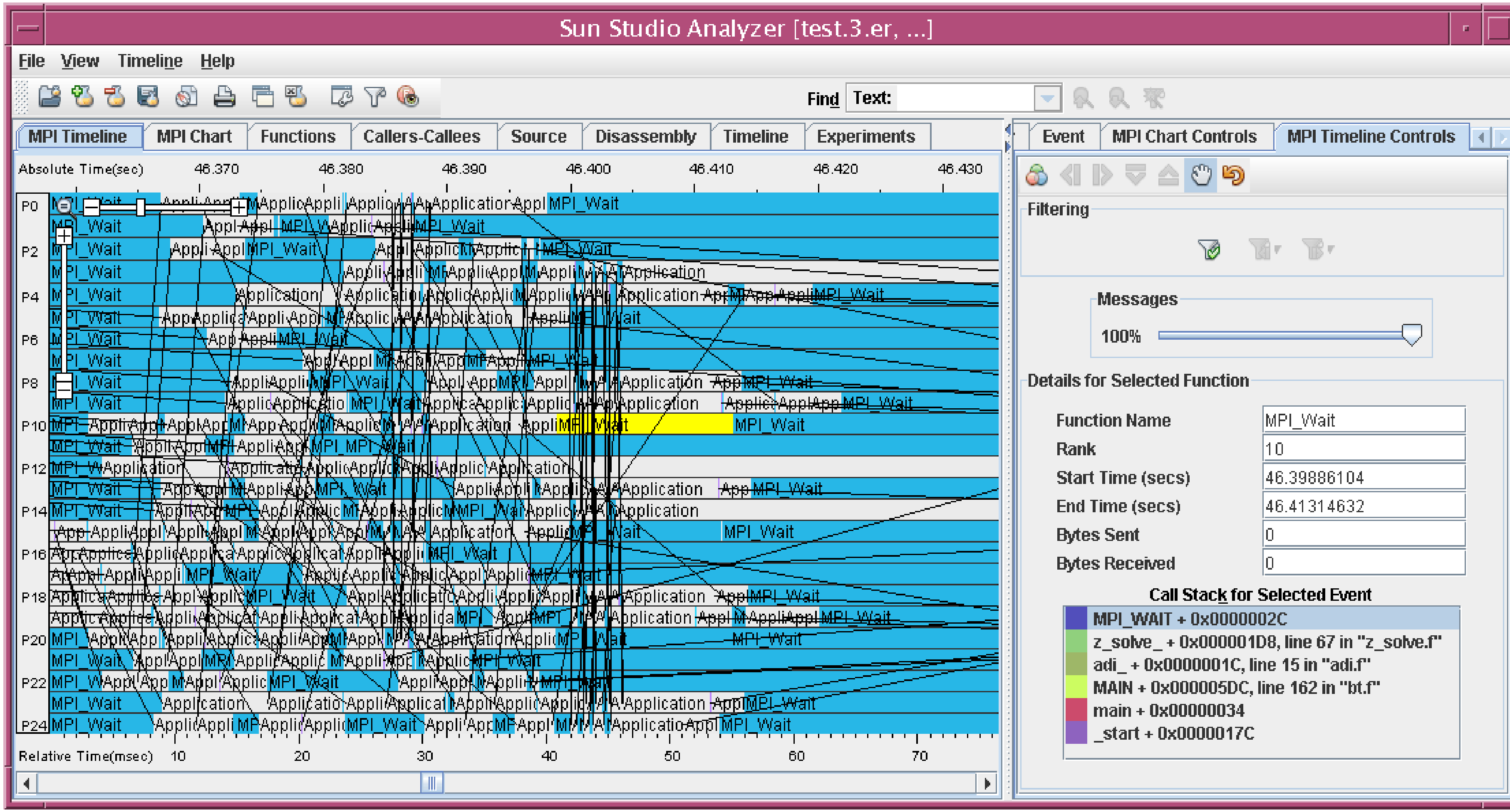
Only 2% of all messages shown

BT -- MPI-Timeline, II



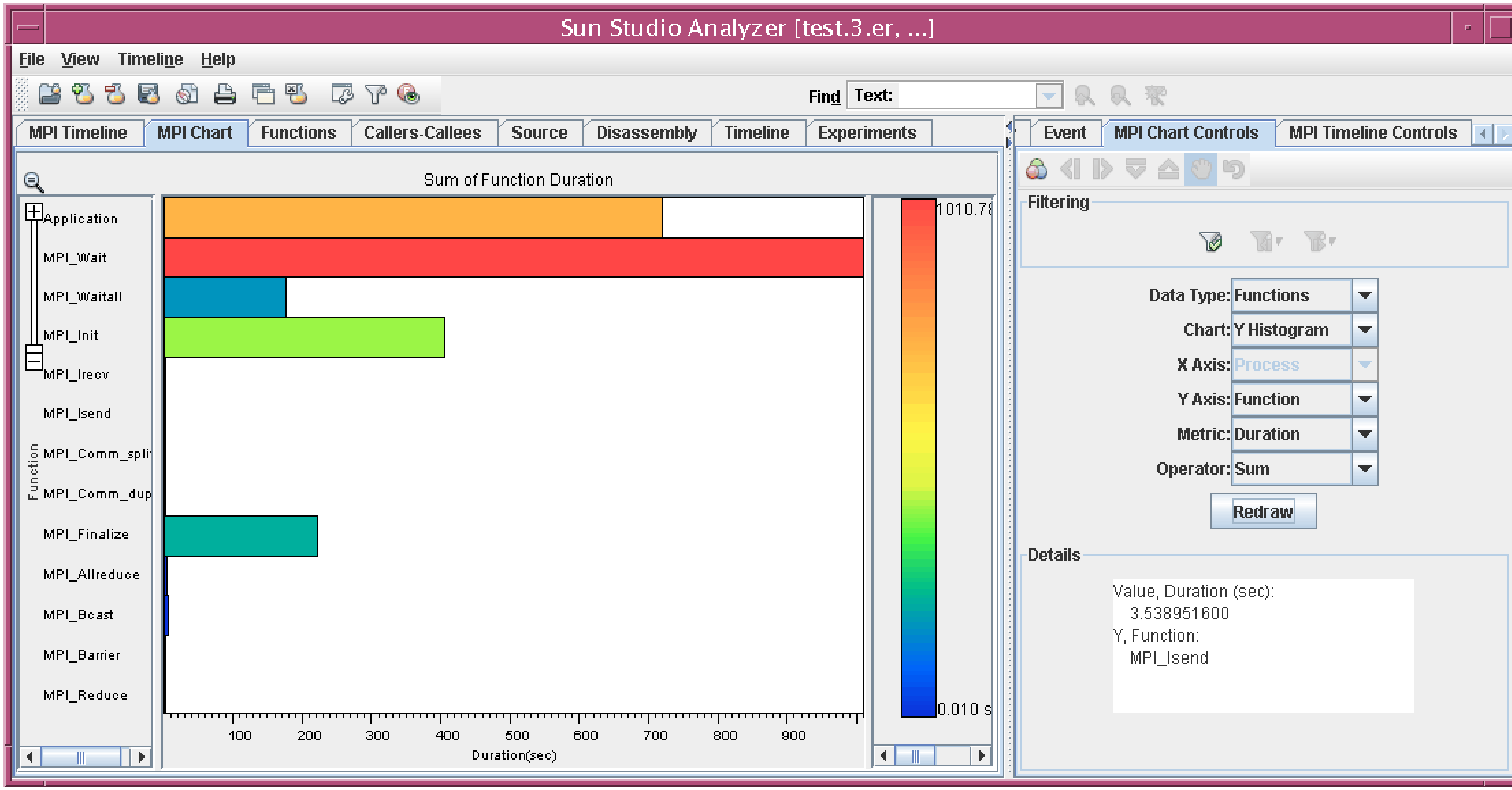
All messages shown

BT -- MPI-Timeline, III



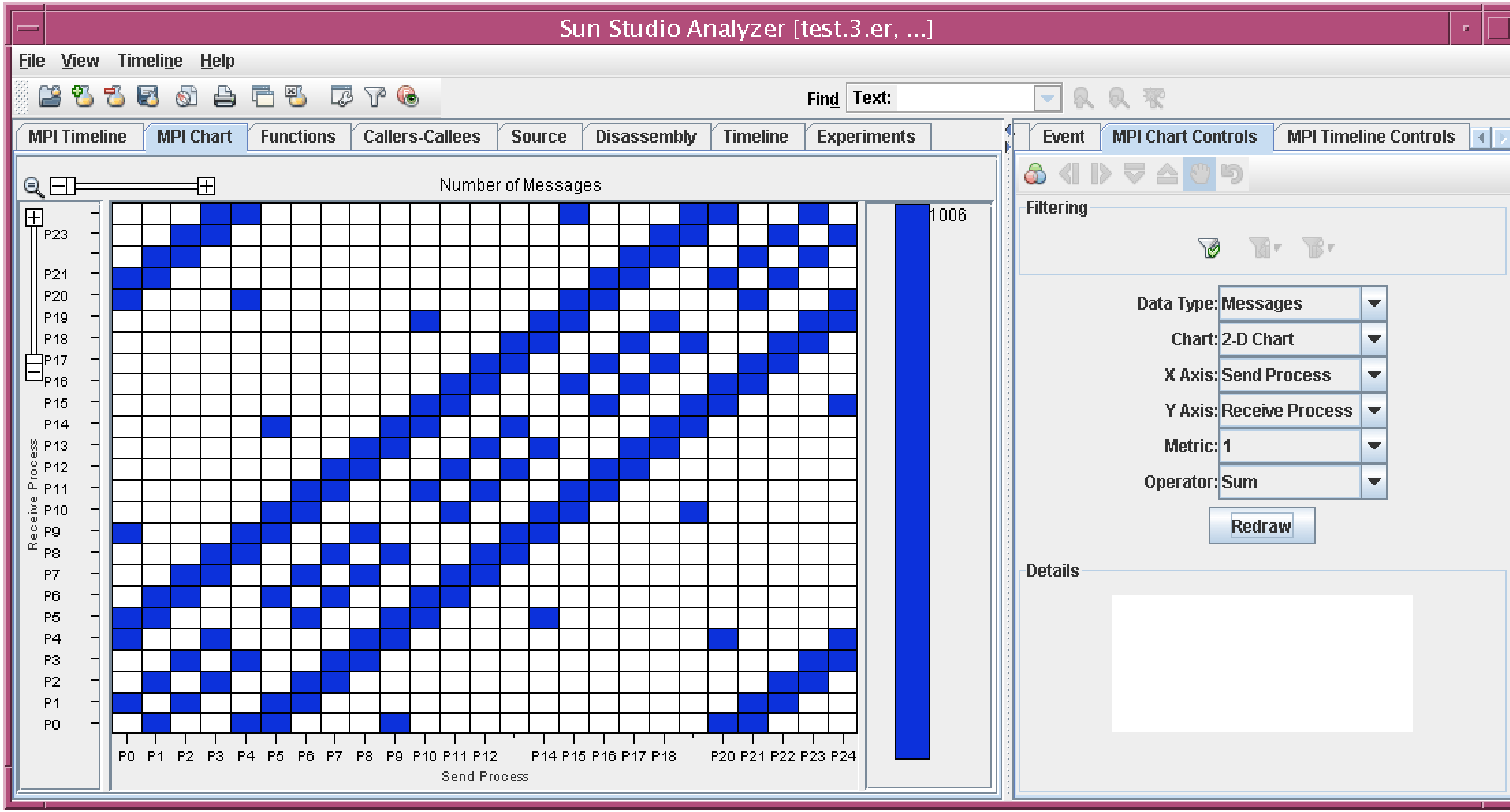
Zoom in, to show individual messages

BT -- MPI Charts, I



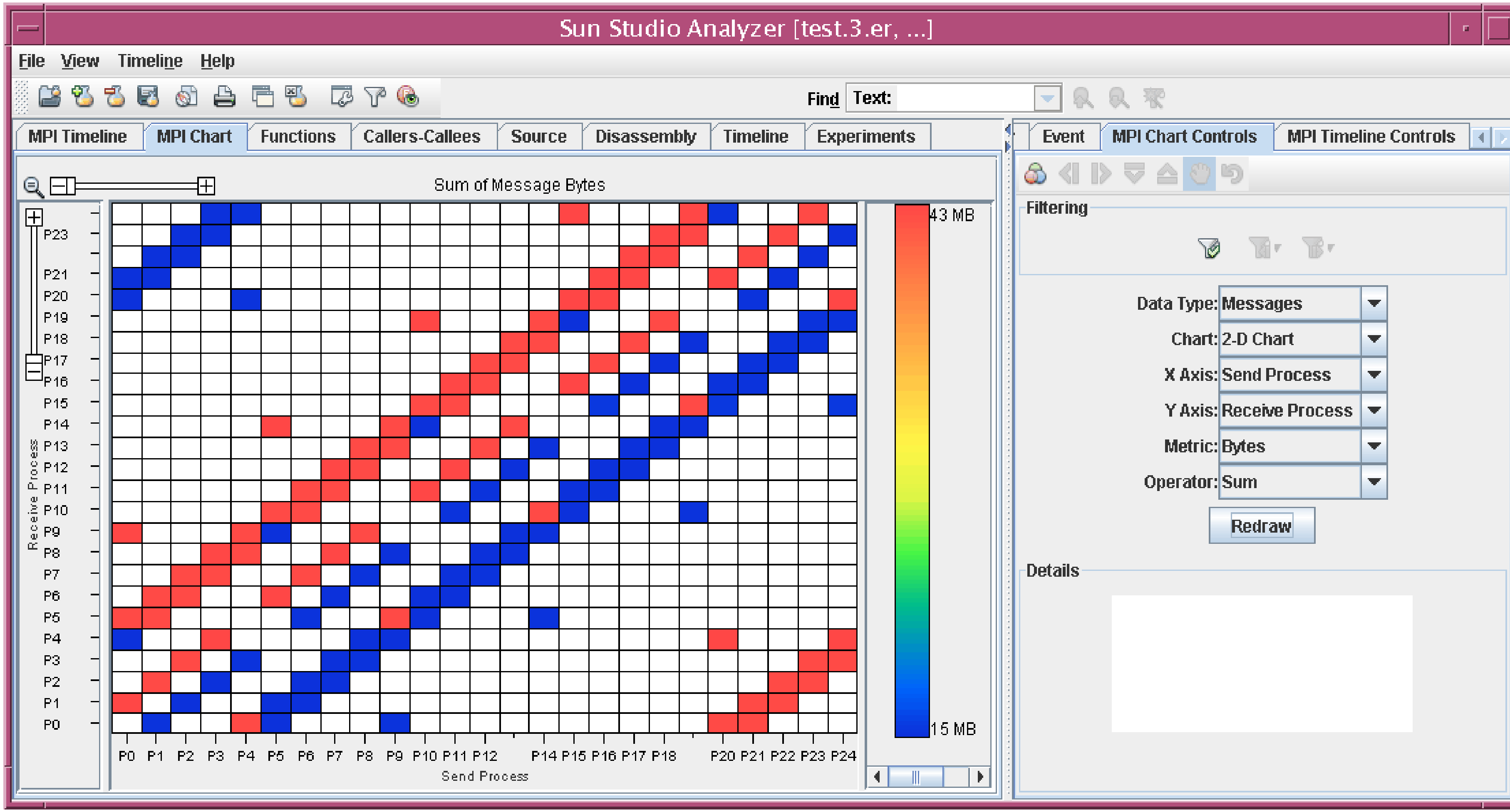
Shows time in functions, aggregated over all ranks

BT -- MPI Charts, II



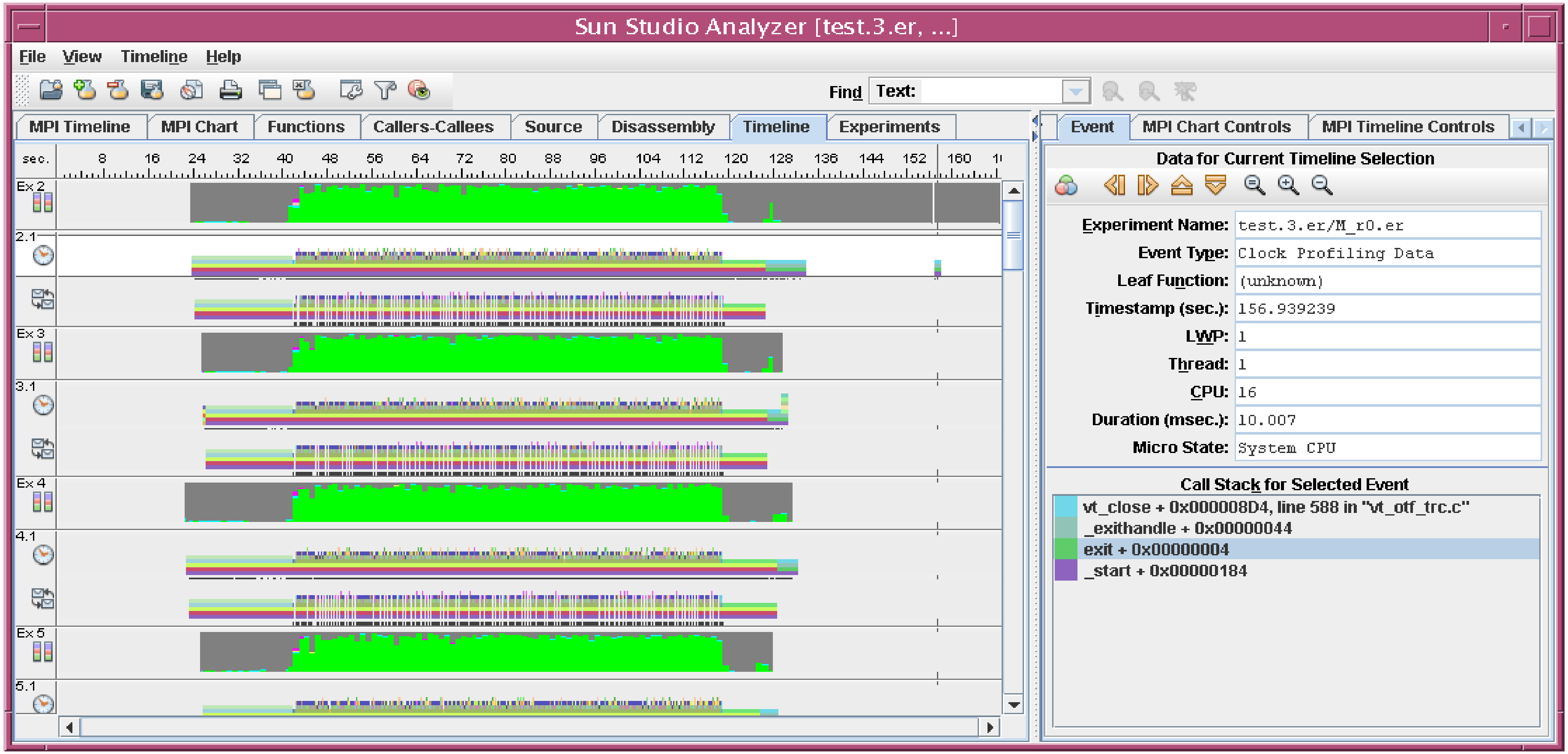
Shows count of messages between pairs of processes

BT -- MPI Charts, III



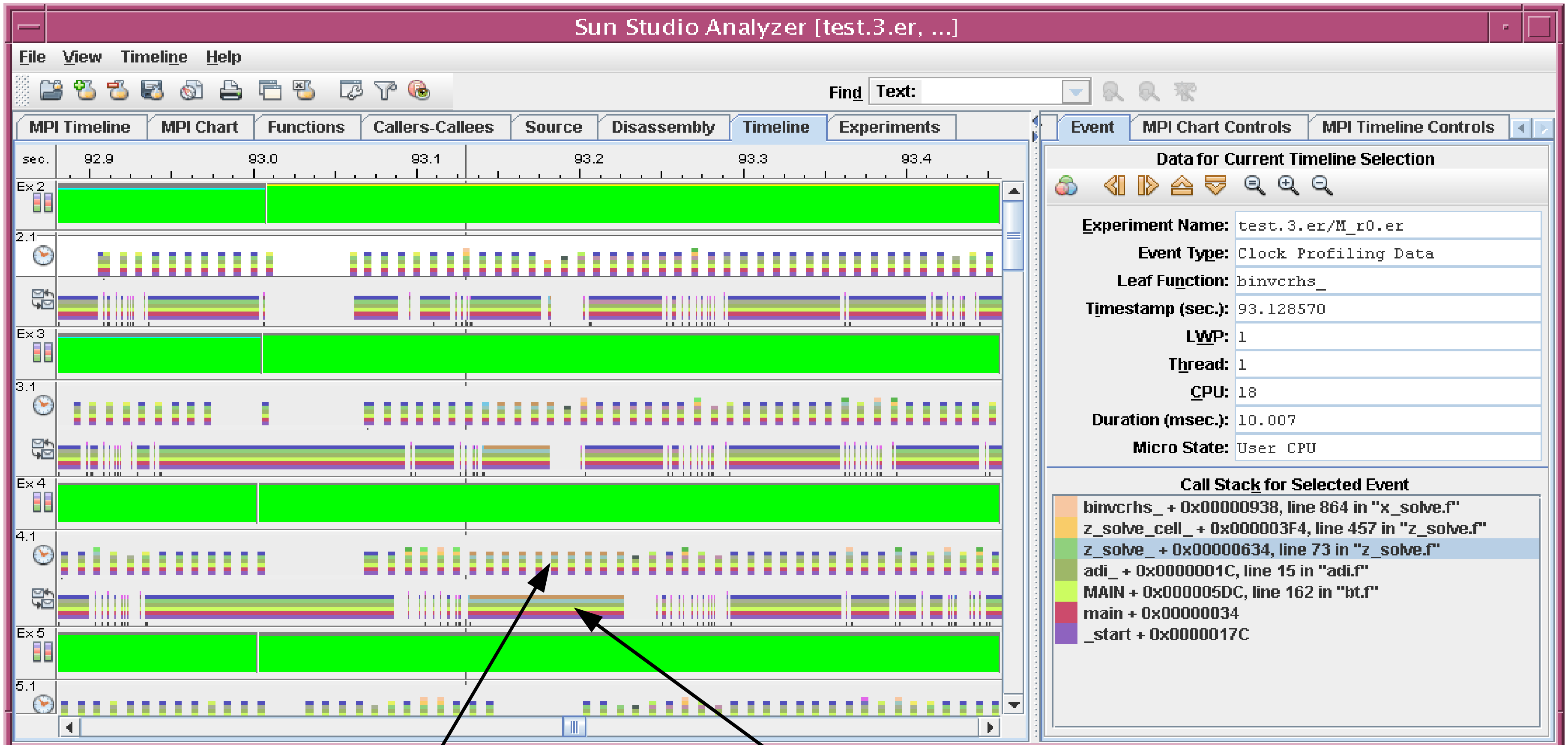
Shows byte-count of messages between pairs of processes

BT – Traditional Timeline, I



Shows both profile and MPI Trace Records

BT – Traditional Timeline, II

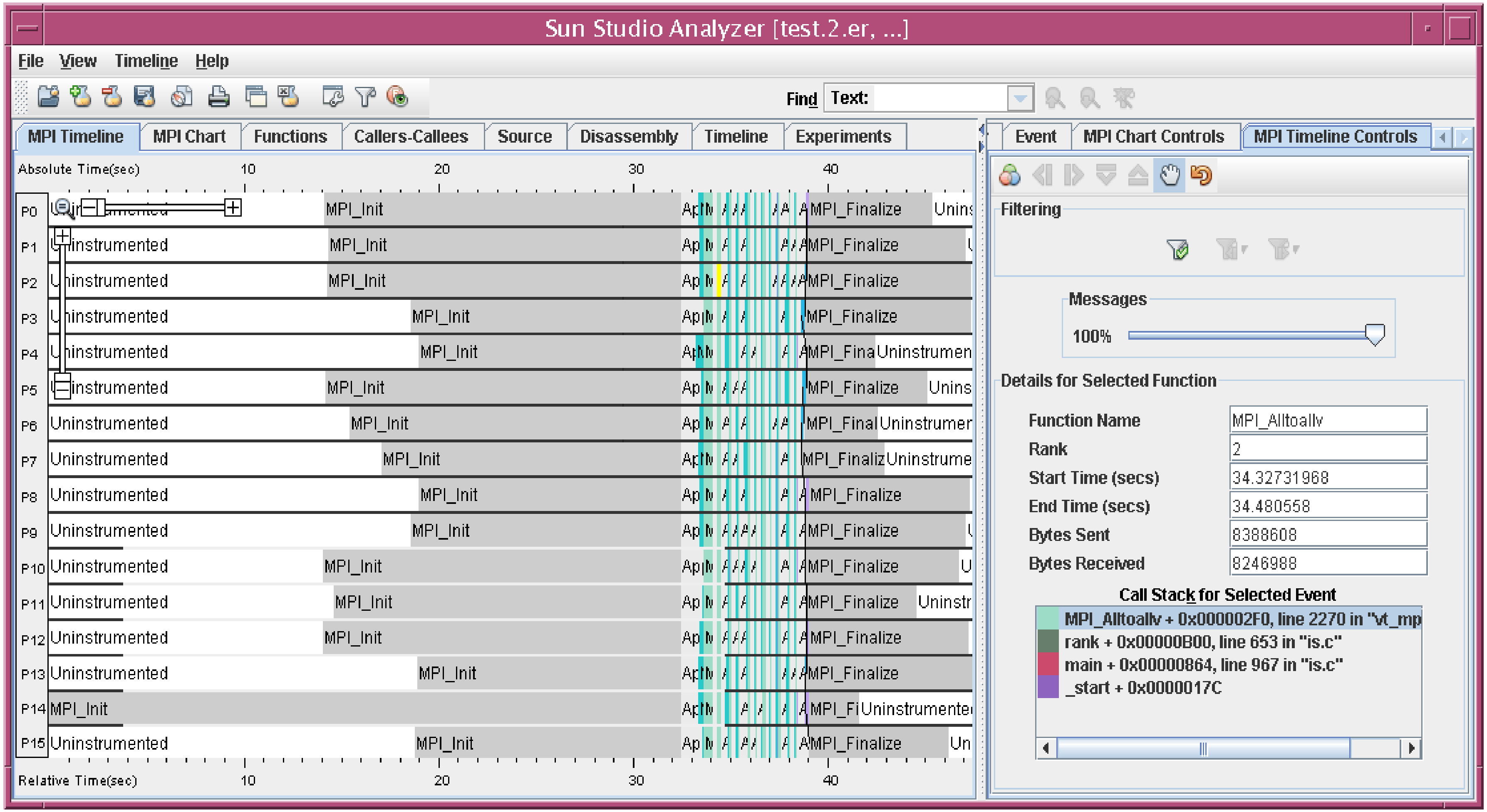


Profile records

MPI Trace records

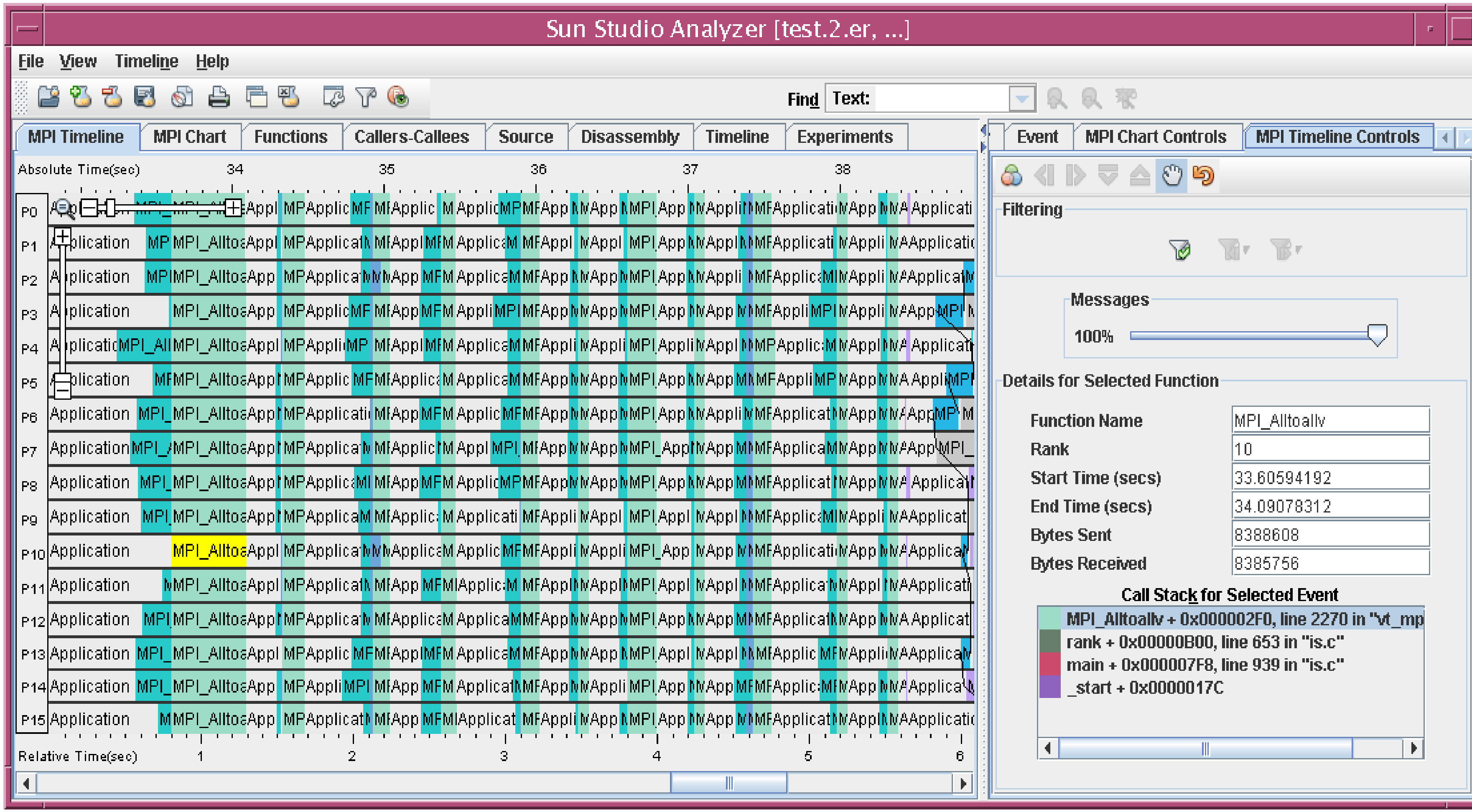
Matching callstacks when inside MPI call

IS -- MPI-Timeline, I



Full scale, tiny run, dominated by MPI_Init and MPI_Finalize

IS -- MPI-Timeline, II



Zoomed in, to show pattern during computation

Almost all MPI operations are collectives

MPI State Profiling

Marty Itzkowitz
Sun Studio Performance Analyzer Team
Sun Microsystems Inc.
marty.itzkowitz@sun.com