Profile & Analyze MPI applications with Intel’s Trace Analyzer and Collector (ITAC)

Anupam Karmakar, Lenovo Data Center Group
My parallel program doesn’t scale! What’s next?

• Isolate performance issues in parallel programs
  ... especially those in large MPI programs
    – inefficient MPI programming → bottlenecks
      - latency-dominated performance (too many small messages)
      - load imbalances
      - deadlock (disambiguate from long messages on a low-bandwidth wire)

  ... but also those via subroutine time consumption

• Existing facilities
  – gprof for subroutines
  – PMPI profiling interface (MPI standard!) + upshot
What is Tracing?

• Record the full length programme execution
  – user function calls,
  – MPI communication calls: point-to-point, collective, I/O etc.

• Accumulate complete statistics of the events recorded.

• Summarise and pack the data in a presentable format.

• If not handled carefully, generates loads of (unnecessary) data.
Intel Trace Analyzer and Collector (ITAC)

• Helps Developers
  – Visualize & understand parallel application behavior
  – Evaluate profiling statistics & load balancing
  – Identify communication hotspots

• Features
  – Event-based approach
  – Low overhead
  – Excellent scalability
  – Powerful aggregation & filtering functions
  – Idealizer

- LRZ HPC Systems:
  - SuperMUC+ and Cluster systems at LRZ
- Version installed at LRZ
  - GUI and tracing libraries ver. 9.0 and 2017
- Also available at RRZE
Documentation and resources

• Intel Web Site

• LRZ Web Site
  – https://doku.lrz.de/display/PUBLIC/Intel+Tracing+Tools%3A+Profiling+and+Correctness+checking+of+MPI+programs

• see especially the links to the user’s guides at the end of this document
• specific Usage instructions for each platform the tracing libraries are available therein
Usage of Trace collector and GUI
Strengths of Event-Based Tracing

- **Predict**: Detailed MPI program behavior
- **Record**: Exact sequence of program states keep timing consistent
- **Collect**: Collect information about exchange of messages: at what times and in which order

An event-based approach can detect temporal dependencies
How to Use Intel Trace Analyzer and Collector

**Step 1: Collect**

Run your binary and create a tracefile:

```
$ mpirun -trace -n 8 ./mpi_app
```

**Step 2: Analyze**

View and analyze the results:

```
$traceanalyzer./poisson_icomm.single.stf &
```

```module unload mpi.<whatever>
module load mpi.intel/2019
module load itac/2019```
Intel® Trace Analyzer and Collector

- **Visualisation**

- during/after program run: trace data are written to disk
- may be visualized on any platform using the ITA GUI
- many views of the trace data available via a transparent menu structure
- hundreds of MPI processes viewable with good performance
  - STF (Structured Trace File) format
  - still lots of data

- **Example screenshot**
Intel® Trace Analyzer and Collector

Compare the event timelines of two communication profiles

Blue = Computation
Red = Communication

Chart showing how the MPI processes interact
MPI Performance Analysis

Automatic Performance Assistant

– Detects common MPI performance issues
– Automated tips on potential solutions

Automatically identifies performance issues and explains their impact on runtime
Flat Function Profile

• Statistics About Functions
Event Timeline

- Get detailed impression of program structure.
- Display functions, messages, and collective operations for each process/thread along time axis.
- Retrieve detailed event information.
Overall load balance

times inclusive traced subroutine calls
Resolving the MPI calls

- Times for each MPI routine specified in config file
- Right click → Ungroup

- Group MPI

- Can sort by category
Call tree

Notes:
- call tree can only resolve activities switched on for tracing
- By default, User code not resolvable unless automatic or manual subroutine tracing compiled in
Call tree split up into user processes

- Select "Children of Group All_Processes"
- can also select expansion into various groups, two of which are defined at least:
  - MPI
  - Application
    (dotted circle → click)
... call up Major Function Group editor

- Select e.g., one of the major function groups and press OK
- You will then essentially filter out all other groups from the view ...
... like this
Now a different view:
Time-Line of all MPI processes

Select

- Charts → Event Timeline
  - this will open an additional pane in the tracing sub-window
- then zoom in to some region of interest by using the left mouse button
- do this repeatedly to obtain ...
... this small, 0.0001 second trace section

red parts: (different shades) are the various MPI activities

blue parts: User code

black lines: communication, right click for context menu → Details on function... to obtain further information about this message

Note that profile window (below) adjusts itself to selected time slice!
Next Chart option: Qualitative Timeline

This will give synchronized information for e.g., transfer rates

other quantities selectable via context menu:
  - transfer duration
  - transfer volume

selectable events can be
  - function events
  - messages (as here)
  - collective operations
Quantitative Timeline: Accumulated measure of activities

- This answers the question:
- How many CPUs are presently engaged in each activity?

Example:
Yellow arrow shows interval where application does communication exclusively

Note:
can remove activities via the context menu if too cluttered
The final Chart type: Message profile

gives you metrics
- message volume throughput
- time
- count

for
senders (horizontal) vs. receivers (vertical)

Note:
- color codes enable you to easily find communication hotspots
- non-dense communication patterns are good
Intel Trace Analyzer and Collector

Summary

**What**
- Intel’s high-performance MPI communications profiler and analyzer for scalable HPC development

**Why**
- **Scale performance:** Perform on more nodes
- **Scale forward:** Multi-core and many-core ready
- **Scale efficiently:** Tune and debug on more nodes

**How**
- **Visualize:** Understand parallel application behavior
- **Evaluate:** Profiling statistics and load balancing
- **Analyze:** Automated analysis of common MPI issues
- **Identify:** Communication hotspots