Automatic trace analysis with the Scalasca Trace Tools

Markus Geimer
Jülich Supercomputing Centre
Example: “Late Sender” wait state

- Waiting time caused by a blocking receive operation posted earlier than the corresponding send
- Applies to blocking as well as non-blocking communication
Example: Critical path

- Shows call paths and processes/threads that are responsible for the program’s wall-clock runtime
- Identifies good optimization candidates and parallelization bottlenecks
Example: Root-cause analysis

- Classifies wait states into direct and indirect (i.e., caused by other wait states)
- Identifies *delays* (excess computation/communication) as root causes of wait states
- Attributes wait states as *delay costs*
Hands-on / Demo:
NPB-MZ-MPI / BT
Performance analysis steps

- 0.0 Reference execution for validation
- 1.0 Program instrumentation
  - 1.1 Summary measurement collection
- 2.0 Summary experiment scoring
  - 2.1 Summary measurement collection with filtering
  - 2.2 Filtered summary analysis report examination
- 3.0 Event trace collection
  - 3.1 Event trace examination & analysis
BT-MZ trace measurement collection...

% cd bin.scorep
% cp ../jobscript/jureca/scalasca.sbatch .
% vim scalasca.sbatch
...
# Measurement configuration
export SCOREP_TOTAL_MEMORY=215M
Export SCOREP_METRIC_PAPI=PAPI_TOT_INS,PAPI_TOT_CYC
export SCAN_ANALYZE_OPTS="--time-correct"

# Run the application
scalasca -analyze -f scorep.filt -t srun ./bt-mz_$CLASS.$PROCS

% sbatch scalasca.sbatch

- Change to directory with the executable and edit the job script
- Adjust the measurement configuration
- Add "-t" to the scalasca -analyze command
BT-MZ trace measurement ... collection

- Starts measurement with collection of trace files ...

```
S=C=A=N: Scalasca 2.4 trace collection and analysis
S=C=A=N: Tue Nov 20 13:34:48 2018: Collect start
srun ./bt-mz_C.8

NAS Parallel Benchmarks (NPB3.3-MZ-MPI) - BT-MZ MPI+OpenMP

> Benchmark

  Number of zones: 16 x 16
  Iterations: 200  dt:  0.000100
  Number of active processes: 8

  [... More application output ...]

S=C=A=N: Tue Nov 20 13:35:09 2018: Collect done (status=0) 21s
```
### BT-MZ trace measurement ... analysis

S=C=A=N: Tue Nov 20 13:35:09 2018: Analyze start

```bash
srun scout.hyb --time-correct ./scorep_bt-mz_C_8x6_trace/traces.otf2
```

Analyzing experiment archive ./scorep_bt-mz_C_8x6_trace/traces.otf2

- Opening experiment archive ... done (0.006s).
- Reading definition data ... done (0.006s).
- Reading event trace data ... done (0.559s).
- Preprocessing ... done (0.346s).
- Timestamp correction ... done (1.331s).
- Analyzing trace data ... done (9.041s).
- Writing analysis report ... done (0.110s).

Total processing time: 11.491s

S=C=A=N: Tue Nov 20 13:35:21 2018: Analyze done (status=0) 12s

- Continues with automatic (parallel) analysis of trace files
BT-MZ trace analysis report exploration

- Produces trace analysis report in the experiment directory containing trace-based wait-state metrics

```bash
% scalasca -examine scorep_bt-mz_C_8x6_trace
INFO: Post-processing runtime summarization result...
INFO: Post-processing trace analysis report...
INFO: Displaying ./scorep_bt-mz_C_8x6_trace/trace.cubex...
```

[GUI showing trace analysis report]
Post-processed trace analysis report

Additional trace-based metrics in metric hierarchy
Online metric description

Access online metric description via context menu
Online metric description

Late Sender Time

Description:
Refers to the time lost waiting caused by a blocking receive operation (e.g., MPI_Recv or MPI_Wait) that is posted earlier than the corresponding send operation.

If the receiving process is waiting for multiple messages to arrive (e.g., in an call to MPI_Waitall), the maximum waiting time is accounted, i.e., the waiting time due to the latest sender.

Unit:
Seconds

Diagnosis:
Try to replace MPI_Recv with a non-blocking receive MPI_Irecv that can be posted earlier, proceed concurrently with computation, and complete with a wait operation after the message is expected to have been sent. Try to post sends earlier, such that they are available when receivers need them. Note that outstanding messages (i.e., sent before the receiver is ready) will occupy internal message buffers, and that large numbers of posted receive buffers will also introduce message management overhead, therefore moderation is advisable.

Parent:
MPI Point-to-point Communication Time

Children:
Critical-path analysis

Critical-path profile shows wall-clock time impact
Critical-path analysis

Critical-path imbalance highlights inefficient parallelism
Pattern instance statistics

Access pattern instance statistics via context menu

Click to get statistics details