Outline

• Introduction to performance analysis concepts
• Introduction to PAPI
• Case studies
Performance Engineering Workflow

- Prepare application
- Collect the relevant data to execution of the instrumented application
- Identification of performance metrics
- Presentations of results
- Modifications in order to reduce performance problems

This workshop
Metrics of Performance

• How often an event occurs
• The duration of some intervals, e.g. the time spent in some communication calls
• The size of the messages during the communication
• Derived metrics
Inclusive vs. Exclusive values

```cpp
int foo()
{
    int a;
    a = 1 + 1;
    bar();
    a = a + 1;
    return a;
}
```
Measurement Techniques

• Methods for the measurement
  – Sampling
  – Code instrumentation

• Record the data
  – Profiling
  – Tracing
Various approaches

- Sampling

- Instrumentation
Profiling

• Record of aggregated information
• For measurements
  – Time
  – Counts
  – Functions
  – Processes, threads
Tracing

- Recording all the events
- Timestamp is recorded
- Chronologically ordered sequence of event records
Tracing vs. Profiling

• Tracing advantages
  – It is possible to reconstruct the dynamic application behavior
  – It is possible to extract the profiling

• Disadvantages
  – Traces can get really large
  – Writing events to a file at runtime causes perturbation
Performance Analysis Procedure

• Performance problem?
  – Time/speedup

• Key bottleneck?
  – MPI/OpenMP ...

• Where is the bottleneck?

• Why?
  – Hardware counters

• Scalability problems?
  – Load imbalance etc.
Performance Application Programming Interface (PAPI)

• Middleware that provides a consistent and efficient programming interface for the performance counter hardware in most major processors with access to:
  – Whole program timing
  – Cache behaviors
  – Branch
  – Instructions per cycle
  – Floating point efficiency
PAPI Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Value</th>
<th>Available</th>
<th>Enabled</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAPI_L1_DCM</td>
<td>0x80000000</td>
<td>Yes</td>
<td>Yes</td>
<td>Level 1 data cache misses</td>
</tr>
<tr>
<td>PAPI_L1_ICM</td>
<td>0x80000001</td>
<td>Yes</td>
<td>No</td>
<td>Level 1 instruction cache misses</td>
</tr>
<tr>
<td>PAPI_L2_DCM</td>
<td>0x80000002</td>
<td>Yes</td>
<td>No</td>
<td>Level 2 data cache misses</td>
</tr>
<tr>
<td>PAPI_L2_ICM</td>
<td>0x80000003</td>
<td>Yes</td>
<td>No</td>
<td>Level 2 instruction cache misses</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PAPI_TOT_CYC</td>
<td>0x8000003b</td>
<td>Yes</td>
<td>No</td>
<td>Total cycles</td>
</tr>
</tbody>
</table>
PAPI Events

**Instructions per Cycle**

\[
PAPI\_TOT\_INS/PAPI\_TOT\_CYC
\]

It is the useful computation. Empirically a value 1.5 or lower, means the computation is not efficient.

**L2 data cache hit Ratio**

\[
1.0-(PAPI\_L2\_DCM/PAPI\_L1\_DCM)
\]

We can see how many cache data misses we had to L2.

There is a limitation of which PAPI events can be combined in the same execution, check the tool papi_event_chooser
Case Studies

• For the main evaluation we used the application MiniWeather and for testing the LSMS.

• MiniWeather: A mini app simulating weather-like flows for training in parallelizing accelerated HPC architectures

• LSMS: a code to perform first principles ground state calculations of solid state systems and statistical physics calculation with a focus on magnetic systems.