Using differential execution analysis to identify contention

Mohamed Mosli, François Trahay, Alexis Lescouet, Gauthier Voron, Rémi Dulong, Amina Guermouche, Élisabeth Brunet, Gaël Thomas
Contention on shared resources

- **Multiple resources are shared**
  - Memory hierarchy (caches, NUMA nodes, ...)
  - Peripheral devices (hard drive, network card, ...)
  - Software resources (locks, ...)

Available resources on a computer
Contention on shared resources

- **Multiple resources are shared**
  - Memory hierarchy (caches, NUMA nodes, ...)
  - Peripheral devices (hard drive, network card, ...)
  - Software resources (locks, ...)

- **How to detect the source of a slowdown?**
  - Log resource usage
  - Measure software indicators
  - Use hardware counters

- **If there is a problem, is it bad for performance?**

```
Available resources on a computer:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>I/O rate</td>
<td>178MB/s</td>
</tr>
<tr>
<td>Net rate</td>
<td>107MB/s</td>
</tr>
<tr>
<td>load</td>
<td>7.3GB/s</td>
</tr>
<tr>
<td>imbalance</td>
<td>198%</td>
</tr>
<tr>
<td>ctx switch</td>
<td>12.10^3/s</td>
</tr>
<tr>
<td>LLC miss</td>
<td>57.10^3/s</td>
</tr>
<tr>
<td>stall cycles</td>
<td>93.10^6/s</td>
</tr>
</tbody>
</table>
```

[Diagram of computer resources]
Differential execution analysis

- **EZTrace**
  - Intercept calls to “interesting” functions
    - eg. MPI, OpenMP, posix IO, ...
  - Generate execution traces
  - Available as open-source: http://eztrace.gforge.inria.fr

- **Analyzing traces**
  - Using visualization tools
    - eg. ViTE, Vampir, …
  - Differential execution analysis
    - Detect sequences of events that repeat
    - Compare occurrences of sequences
Detecting thread contention

- Slowdown Caused by Interference (SCI)
  - If function foo can execute in 2µs once, longer execution can be caused by:
    - Access to a contented resource
    - Execution of a different path
    - Execution is dependent on at least one parameter
  - SCI = Theoretical speedup if all calls to foo lasted 2µs

- Can be applied to detect various types of interference
  - Lock contention, IO contention, Network contention, Memory placement, false-sharing, ...
  
    → Universal indicator for contention
Evaluation

Panel of 27 applications
- NAS Parallel benchmarks (7 applications)
- Parsec (4 applications)
- Splash-2 (7 applications)
- Phoenix2 (7 applications)
- LevelDB with write-intensive workload
- Memcached with write-intensive workload

Hardware configuration
- 48 core NUMA machine

Results
- 11 applications have high SCI score
- 12 interference problems
  - IO contention, lock contention, false-sharing, NUMA, network, parallelism)
- Significant performance improvement once fixed
- Few false-positive
**Evaluation: NPB DC**

- **NAS Parallel Benchmarks: DC kernel**
  - Data-mining application
  - Profiling shows 4 hot functions
    - KeyComp, MultiWayMerge, memcpy, fwrite
      - instrument these functions
  - EZTrace generates a 17GiB trace (with 364 million events)
  - Two high SCI scores
    - MultiWayMerge (false-positive)
    - fwrite
      - Called with data size ranging from 1 to 24 bytes
      - Here, data size does not impact the function completion time
Analyzing fwrite calls in DC

- Distribution of completion time
  - 98.1% < 1µs
  - 99.8% < 10µs
  - 99.97% < 100µs
  - 0.03% of calls (1274) contribute to 44% of the total time
Improving DC performance

- **Hard to improve the performance of the application**
  - Requires to rewrite large parts of the code

- **Running DC on a RAMFS partition**
  - Improve the performance by 68%
    - Better throughput than hard disk
  - SCI score: 0.17 (compared to 0.83 with hard disk)
    - Lower contention on the IO stack
Conclusion & Future work

Differential execution analysis

• Universal indicator for contention
  - Can detect IO contention, network contention, lock contention, memory issues (eg. false sharing), ...

• Evaluation on 27 applications
  - 12 problems were detected

Future work

• From a research prototype to production software
  - As part of the IDIOM FUI project
  - Extend EZTrace to other IO paradigms (MPI-IO, Hadoop, ...)