Fast Classification of MPI Applications Using Lamport’s Logical Clocks

Zhou Tong
Florida State University
tong@cs.fsu.edu

Scott Pakin and Mike Lang
Los Alamos National Lab
(mlang, pakin}@lanl.org

Xin Yuan
Florida State University
xyuan@cs.fsu.edu

Introduction
Fast classification and identification of performance limiting factors in MPI applications.

- One simulation predicts application performance for many network configurations by replaying the traces.
- Use extended Lamport’s logical clocks to maintain time counters that track computation, bandwidth, latency and wait time.
- Classify 9 DOE’s full applications and miniapps into bandwidth-bound (BW), Latency-bound (L), computation-bound (Comp.), load-imbalance-bound (1imb.).
- Performance evaluation with NAS benchmarks show that our tool is fast and scalable.

Key Ideas
- Trace-driven, MPI-based modelling tool.
- Classify application based on the performance summary on a range of configurations. (Bandwidth and Latency) of given interconnect technology.

Modelling Assumptions:
- Model Communication with Eager Protocol as default. Rendezvous protocol is available if necessary.
- Point-to-point (P2P): Hockney’s model \( \alpha + \pi \times \beta \), where \( \alpha \) is the communication latency, \( \pi \) is the message size and latency \( \beta \) is the per-byte bandwidth speed.
- Collective: global synchronization for all processes to be ready for the operation.

Benchmarks
<table>
<thead>
<tr>
<th>AMR</th>
<th>Adaptive Mesh Refinement cosmology</th>
</tr>
</thead>
<tbody>
<tr>
<td>BigFFT</td>
<td>3D Fast Fourier Transform solver</td>
</tr>
<tr>
<td>CLAMR</td>
<td>Cell-based adaptive mesh refinement</td>
</tr>
<tr>
<td>CR</td>
<td>Nek5000</td>
</tr>
<tr>
<td>FB</td>
<td>Halo update PDE solver code</td>
</tr>
<tr>
<td>MG</td>
<td>Geometric Multigrid elliptic solver</td>
</tr>
<tr>
<td>MiniFE</td>
<td>Finite element mini-application</td>
</tr>
<tr>
<td>PARTISN</td>
<td>Neutral-particle transport</td>
</tr>
<tr>
<td>NPB</td>
<td>NAS Parallel Benchmarks</td>
</tr>
</tbody>
</table>

Fast Classification Tool

![Fast Classification Tool Diagram]

Basic Lamport’s Clock

\[ \begin{align*}
\ell_1 & = \ell_0 + 1 \\
\ell_2 & = \ell_0 + 2 \\
\ell_3 & = \ell_0 + 3 \\
\ell_4 & = \ell_0 + 4
\end{align*} \]

Extended Lamport’s Logical Clocks

- Modeled by non-unit communication and computation time.
- In simulation, timestamps are sent instead of the real data from the original application.

Validating

Table II: Predicted and measured communication and total application time (in seconds) of 64-rank CLAMR, CR and FB on Cielo

<table>
<thead>
<tr>
<th></th>
<th>CLAMR</th>
<th>CR</th>
<th>FB</th>
<th>Cielo</th>
<th>total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comp.</td>
<td>0.12</td>
<td>0.25</td>
<td>0.36</td>
<td>0.15</td>
<td>0.26</td>
</tr>
<tr>
<td>Est. Comm.</td>
<td>0.25</td>
<td>0.5</td>
<td>0.7</td>
<td>0.4</td>
<td>0.7</td>
</tr>
<tr>
<td>Act. Comm.</td>
<td>0.25</td>
<td>0.4</td>
<td>0.6</td>
<td>0.3</td>
<td>0.6</td>
</tr>
<tr>
<td>Pred. %</td>
<td>0.12</td>
<td>0.25</td>
<td>0.36</td>
<td>0.15</td>
<td>0.26</td>
</tr>
<tr>
<td>Pred.</td>
<td>0.25</td>
<td>0.5</td>
<td>0.7</td>
<td>0.4</td>
<td>0.7</td>
</tr>
<tr>
<td>Pred. Est.</td>
<td>0.25</td>
<td>0.5</td>
<td>0.7</td>
<td>0.4</td>
<td>0.7</td>
</tr>
</tbody>
</table>

For each benchmark:
1st column: (eag:*:) shows the predicted performance using default eager protocol with Hockney’s model.
2nd column: (eag:*:) shows that the predicted performance using the default eager protocol with the more accurate look-up table improves prediction errors of both communication and overall execution time. The overall prediction errors of CLAMR and CR are within 3%. Message sizes in CLAMR and CR are under 4KB.

For FB:
3rd column: shows prediction results with rendezvous protocol and look-up table further improves the prediction rate because message sizes in FB are under 4KB.

Preliminary Results

(b) BigFFT(100) is load-imbalance bound (imb.) as wait time accounts for nearly 30% of total time with QDR communication model.
(c) Percentage of computation time decreases as the problem size and the number of ranks in AMG increases.
(d) Simulation time shows 3-45 time speedup for 4096-rank runs of NPB benchmarks.
(e) Simulation time shows 2-15 time speedup for 4096-rank runs of NPB benchmarks.
(f) Simulation time shows negligible difference as increasing number of configurations are simulated in one run due to growing message sizes in simulation.

Conclusions

Trace-based and communication-centric fast classification tool for MPI programs provide insights on application performance.

- Our innovation is to use a modified Lamport logical clock scheme with non-unit computation and communication times to predict overall time.
- By maintaining multiple independent logical clocks that are parameterized differently but honor the same happens-before relationship, the tool can predict execution time on many network configurations in nearly the same time needed to predict time for a single configuration.
- This multiple-prediction capability enables new analyses to be performed that would be too computationally expensive to perform with traditional, one-configuration-at-a-time simulation.
- Classification results could assist code optimization and better overlap of communication and computation.

Acknowledgements

Special thanks to my advisor Xin Yuan. This work is supported by Ultra High Performance Computing (UHPC) at Los Alamos National Laboratory.

Nomenclature

- AMR: Adaptive Mesh Refinement cosmology
- BigFFT: 3D Fast Fourier Transform solver
- CLAMR: Cell-based adaptive mesh refinement
- CR: Nek5000
- FB: Halo update PDE solver code
- MG: Geometric Multigrid elliptic solver
- MiniFE: Finite element mini-application
- PARTISN: Neutral-particle transport
- NPB: NAS Parallel Benchmarks