Deploying a Task-based Runtime System on Raspberry Pi Clusters

Patrick Diehl and Steven R. Brandt

University of Louisiana

{pdiehl,sbrandt}@cct.lsu.edu

Extreme Scale Programming Models and Middleware (ESPM2’20)
Motivation

- Arm®-Technology is emerging in supercomputers and data centers, e.g. Fugaku the fastest supercomputer in the Top500.
- The low power consumption.
- The low costs of a Raspberry Pi and building a small cluster. One cluster of 4 nodes costs around $200.
Outline

1. Tools

2. Hardware and Software

3. Benchmarks

4. Results
   - Memory
   - Computation time
   - Energy consumption

5. Conclusion & Outlook
Tools
HPX’s lightweight user threads reduce context switching overhead
- Active Global Address Space (AGAS) makes a unified view of the application
- Overlapping communication and computation in the Parcel Layer

**Reference**

Run Python code within the HPX runtime system in parallel.

Python is a common used language in machine and deep learning.

Reference

Hardware and Software
Table: Specification/Architecture of the three nodes utilised in the benchmarks.

<table>
<thead>
<tr>
<th>Model</th>
<th>Raspberry Pi 3B</th>
<th>Raspberry Pi 3B+</th>
<th>Raspberry Pi 4B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Micro-architecture</td>
<td>Arm® v8-A</td>
<td>Arm® v8-A</td>
<td>Arm® v8</td>
</tr>
<tr>
<td>Processor Model</td>
<td>Cortex-A53</td>
<td>Cortex-A53</td>
<td>Cortex-A72</td>
</tr>
<tr>
<td>Number of CPUs</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Cores per CPU</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Total Cores</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Frequency</td>
<td>1.2GHz</td>
<td>1.4GHz</td>
<td>1.5GHz</td>
</tr>
<tr>
<td>Memory</td>
<td>1GB</td>
<td>1GB</td>
<td>4GB</td>
</tr>
</tbody>
</table>

1 https://bitbucket.org/blaze-lib/blaze
2 https://github.com/STEllAR-GROUP/hpx
Table: Specification/Architecture of the three nodes utilised in the benchmarks.

<table>
<thead>
<tr>
<th>Model</th>
<th>Raspberry Pi 3B</th>
<th>Raspberry Pi 3B+</th>
<th>Raspberry Pi 4B</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Micro-architecture</strong></td>
<td>Arm® v8-A</td>
<td>Arm® v8-A</td>
<td>Arm® v8</td>
</tr>
<tr>
<td><strong>Processor Model</strong></td>
<td>Cortex-A53</td>
<td>Cortex-A53</td>
<td>Cortex-A72</td>
</tr>
<tr>
<td><strong>Number of CPUs</strong></td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td><strong>Cores per CPU</strong></td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td><strong>Total Cores</strong></td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td><strong>Frequency</strong></td>
<td>1.2GHz</td>
<td>1.4GHz</td>
<td>1.5GHz</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>1GB</td>
<td>1GB</td>
<td>4GB</td>
</tr>
</tbody>
</table>

Table: Overview of the compilers, software, and operating system used.

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Ubuntu 20.04 LTS for Arm®</th>
<th>Kernel</th>
<th>5.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compilers</td>
<td>gcc 9.30.1</td>
<td>blaze1</td>
<td>75179e6</td>
</tr>
<tr>
<td></td>
<td>hwloc 2.1.0</td>
<td>boost</td>
<td>1.71</td>
</tr>
<tr>
<td></td>
<td>lapack 3.8</td>
<td>gperftools 2.7</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>HPX2</td>
<td>5b9de48ab1</td>
</tr>
</tbody>
</table>

1. https://bitbucket.org/blaze-lib/blaze
2. https://github.com/STEllAR-GROUP/hpx
2D Jacobi Solver (Shared memory)

- 2D Stencil based on the Jacobi method using
  - Standard grid layout for GCC’s autovecorize (−03)
  - Virtual Node Scheme\(^1\) for explicit vectorization
- Roofline model\(^2\) to predict the optimal performance

\[
P_{\text{optimal}} = \text{Memory Bandwidth} \times \text{AI}
\]

with arithmetic intensity (AI) is given by \(\frac{1}{24}\) for double precision and \(\frac{1}{12}\) for single precision.

References


Parameter: heat transfer coefficient $k = 0.5$, time step $dt = 1$, and grid spacing $dx = 1$. 

Figure: Initial conditions

Figure: Solution
Results
Figure: Memory Bandwidth results using the STREAM TRIAD Benchmark with an array size of 10M elements

- Pi 3B/3B+ have very low memory bandwidth (MB)
- One single processor unit (PU) already saturated
- Pi 4 same behavior, but double MB
- Conclusion: Memory bus can only handle a certain amount of MB and concurrency at the same time.
**2D stencil (Raspberry Pi 4)**

*Figure:* 2D stencil (Raspberry Pi 4): Grid size of $4096 \times 4096$ iterated over a 100 time steps.

**Conclusion:** Best performance on 2 cores.
2D stencil (Raspberry Pi 3B+)

Figure: 2D stencil (Raspberry Pi 3B+): Grid size of $4096 \times 4096$ iterated over a 100 time steps.

Conclusion: We can not achieve the expected peak performance.
Figure: 2D stencil (Raspberry Pi 3B): Grid size of $4096 \times 4096$ iterated over a 100 time steps.

Conclusion: PI 3B and 3B+ have similar performance, because these two model differ only in the clock speeds.
Conclusion: Multi-node codes can scale well.
Figure: Execution time in seconds for various node counts using various threads.

Conclusion: Threads provide little performance gain, and actually hurt on the 3 and 3+.
Figure: Execution time in seconds for various core counts.

Conclusion: Our ALS code is fastest on 2 cores, but probably needs more development.
Cost wrt Power Consumption

**Figure:** Cost with respect to power consumption for the 1D stencil code using 30 million stencil points per iteration and a total of 100 iterations.

**Figure:** Cost with respect to power consumption for the Alternating Least Square (ALS) benchmark in Phylanx for the MovieLens 20m database.

The power consumption for all models was obtained using the Linux command `stress`\(^3\) for all four cores.

\(^3\)https://linux.die.net/man/1/stress
Conclusion & Outlook
Conclusion

- Limited memory bandwidth limits the utilization of all cores.
- Ubuntu Server 2020 supports 32-bit and 64-bit.
- Frequency setting of “performance” used instead of “ondemand.”
- The cluster provides modest performance at a reasonable cost.

Outlook

- Use the small and affordable cluster for teaching parallel and distributing computing.
- The interface to attach sensors could be used in field studies to collect data and Phylanx could process the data before uploading to more powerful devices to do the analysis.
- Use a larger cluster and more sophisticated Arm® hardware.
This work is licensed under a Creative Commons “Attribution-NonCommercial-NoDerivatives 4.0 International” license.