SMART-PETSc: High-Performance MPI Library to Boost Performance of the PETSc Library

Drs. Donglai Dai (PI), Sreevatsa (Sreev) Anantharamu (Lead Developer), Hari Subramoni, D. K. Panda

s.anantharamu@x-scalesolutions.com

X-ScaleSolutions
http://x-scalesolutions.com
SMART-PETSc

"PETSc, the Portable, Extensible Toolkit for Scientific Computation, includes a large suite of scalable parallel linear and nonlinear equation solvers, ODE integrators, and optimization algorithms for application codes written in C, C++, Fortran, and Python. In addition, PETSc includes support for managing parallel PDE discretizations including parallel matrix and vector assembly routines." (Used by more than 30 scientific toolkits/libraries)

"The MVAPICH2 software, based on MPI 3.1 standard, delivers the best performance, scalability and fault tolerance for high-end computing systems and servers using InfiniBand, Omni-Path, Ethernet/iWARP, RoCE(v1/v2), Cray Slingshot 10 and 11, and Rockport Networks networking technologies. This software is being used by more than 3,325 organizations in 90 countries worldwide to extract the potential of these emerging networking technologies for modern systems."

- **SMART-PETSc** is a co-designed PETSc + MVAPICH2 middleware
- **Goal**: Deliver best performance for PETSc end-applications via co-design to take full advantage of modern HPC architecture features
- **Challenge**: How?
- **Thanks to support from DOE SBIR Phase-II and -I**
Team

• Drs. Donglai Dai, Sreevatsa (Sreev) Anantharamu, Hari Subramoni, D. K. Panda
  (Led by X-ScaleSolutions, LLC)
• Drs. Richard Tran Mills, Junchao Zhang (Argonne National Lab)
• Dr. Victor Eijkhout (Texas Advanced Computing Center)
• Drs. Sameer Shende, Allen Malony (ParaTools, Inc)
• MPI communication patterns in PETSc
• Optimizations
  ➢ Matrix-vector multiplication kernel
  ➢ Finite Difference PETSc application
  ➢ Finite Element PETSc application
  ➢ Intra-node bandwidth on modern CPUs
  ➢ MVAPICH2-DPU with PETSc
  ➢ Co-designed rendezvous protocols
• Profiling
  ➢ TAU + PETSc via perfstubs
• Conclusions
MPI communication patterns in PETSc

Point-to-point
- Near-neighbor
- Parallel matrix-vector multiplication, assembly
- Solutions transfers (prolongators/restrictors) in multi-grid/-level preconditioners
- Krylov Solvers, Preconditioners
- libCEED exascale library (matrix-free back-end of PETSc) needs two rounds of point-to-point communication for each matrix-vector multiplication (compared to just one round for the usual assembled matrices)

Collectives
- Inner-products in Krylov solvers
- Coarse level solution in multigrid preconditioner

Classical CG, ( , ) denotes an inner product
Optimizations: Matrix-vector kernel

- High-order finite-difference Laplacian stencil (stencil width=5)
- Up to 10% benefit
- Environment variable MV2_SMART_PETSC_MATVEC_OPT=1

Matrix-vector Kernel (Frontera at TACC)

<table>
<thead>
<tr>
<th>Nodes</th>
<th>PETSc + MVAPICH2</th>
<th>SMART-PETSc</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>8</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>4</td>
<td>80</td>
<td>80</td>
</tr>
</tbody>
</table>

Frontera system specification

<table>
<thead>
<tr>
<th>Processors</th>
<th>Intel 8280 “Cascade Lake”</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cores/Node</td>
<td>56 (28 per socket)</td>
</tr>
<tr>
<td>Memory/Node</td>
<td>192GB DDR-4</td>
</tr>
<tr>
<td>Network</td>
<td>Mellanox Infiniband, HDR-100</td>
</tr>
</tbody>
</table>
Optimizations: Finite Difference PETSc application

- Poisson problem (with non-periodic boundary condition)
- 10th order finite-difference spatial discretization (non-symmetric due to non-periodic boundaries)
- Encountered in fluid, solid, and heat transfer applications
- Up to 9% application-level benefit on TACC
- Different Krylov subspace solvers
  - GMRES – Generalized Minimal Residual
  - BICG – Biconjugate Gradient
  - BCGS – Biconjugate Gradient-Stabilized
Optimizations: Finite Difference PETSc application

- Other CPUs
- Gary and Roberta – AMD EPYC, Helios – Intel Gold

Gary and Roberta at OACISS

<table>
<thead>
<tr>
<th>Processors</th>
<th>Intel Gold</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cores/Node</td>
<td>40</td>
</tr>
<tr>
<td>Memory/Node</td>
<td>192GB DDR-4</td>
</tr>
<tr>
<td>Network</td>
<td>Mellanox Infiniband, HDR-200</td>
</tr>
</tbody>
</table>

Helios at HPCAC

<table>
<thead>
<tr>
<th>Processors</th>
<th>AMD EPYC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cores/Node</td>
<td>16, 192</td>
</tr>
<tr>
<td>Memory/Node</td>
<td>384GB DDR-4</td>
</tr>
<tr>
<td>Network</td>
<td>Mellanox Infiniband, NDR-400</td>
</tr>
</tbody>
</table>

Poisson problem (Gary and Roberta at OACISS)

- 200 procs, BICG: 10%
- 200 procs, BCGS: 6%

Matrix-vector Kernel (Helios HPCAC)

- 7 nodes: 7%

- PETSc + MVAPICH2
- SMART-PETSc

Gary and Roberta at OACISS

- Other CPUs
- Gary and Roberta – AMD EPYC, Helios – Intel Gold
Optimizations: Finite Element PETSc application

- libCEED
- Matrix-free back-end of many exascale high-order finite element libraries
- Sum-factorization, high throughput
- Uses PETSc tools to setup and perform MPI communication
- CPU, libxsmm, optimized blocked AVX512 instructions
- Bakeoff problem #2, conjugate gradient with mass matrix on a three-dimensional vector
Optimizations: Intra-node bandwidth on modern CPUs

- AMD EPYC, MPI-only
- Up to 17% latency reduction and 21% bandwidth improvement for large message sizes
- `osu_latency` and `osu_bw` microbenchmark
- Environment variable `MV2_SMART_PETSC_OPT=1` to turn on enhancement
Optimizations: Intra-node bandwidth on modern CPUs

- MPI+OpenMP with AMD EPYC
- Up to 48% latency reduction and 90% bandwidth increase
- osu_bw microbenchmark
- Environment variable MV2_SMART_PETSC_OPT=1 to turn on enhancement
Optimizations: MVAPICH2-DPU with PETSc

- Using Bluefield DPU/SmartNICs to offload point-to-point, reduction and some computation

**3D-Stencil Overlap Benchmark**

<table>
<thead>
<tr>
<th>Data Exchange Size</th>
<th>Normalized Overall Latency</th>
<th>MVAPICH2</th>
<th>MVAPICH2-DPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>64K</td>
<td>13%</td>
<td>120</td>
<td>107</td>
</tr>
<tr>
<td>128K</td>
<td>18%</td>
<td>110</td>
<td>95</td>
</tr>
<tr>
<td>256K</td>
<td>15%</td>
<td>110</td>
<td>95</td>
</tr>
</tbody>
</table>

Communication pattern similar to PETSc DMDA

**Execution Time, BF-3 (PETSc)**

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Total Execution Time (s)</th>
<th>PETSc</th>
<th>PETSc-Offloaded</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>24%</td>
<td>15</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>23%</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>18%</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

27-point Laplacian stencil PETSc application
Optimizations: Co-designed rendezvous protocols

- Co-designed rendezvous protocols
- 3D-stencil benchmark (communication pattern similar to PETSc DMDA)
- Up to 50% performance benefit
- Demonstrates the maximum potential of such co-design enhancements
- Currently, porting it to PETSc
Profiling: TAU + PETSc via perfstubs

- Added interface for easy profiling of PETSc with TAU
- Contributed back to PETSc public repo
- Is now the de facto standard procedure to profile PETSc with TAU

---

**Add perfstubs**

- Merged Samuel Khvuis requested to merge khsai/petcs:perfstubs into main 1 year ago

---

**Github merge request**

---

**TAU pprof result**

Reading Profile files in profile.*

<table>
<thead>
<tr>
<th>%Time</th>
<th>Exclusive msec</th>
<th>Inclusive total msec</th>
<th>#Call</th>
<th>#Subrs</th>
<th>Inclusive Name usec/call</th>
</tr>
</thead>
<tbody>
<tr>
<td>100.0</td>
<td>26</td>
<td>1,838</td>
<td>1</td>
<td>4132</td>
<td>1838424 .TAU application</td>
</tr>
<tr>
<td>73.2</td>
<td>1</td>
<td>1,345</td>
<td>2</td>
<td>168</td>
<td>672950 SNESsolve</td>
</tr>
<tr>
<td>62.2</td>
<td>3</td>
<td>1,142</td>
<td>2</td>
<td>1282</td>
<td>571442 SNESJacobianEval</td>
</tr>
<tr>
<td>62.0</td>
<td>1,136</td>
<td>1,138</td>
<td>2</td>
<td>76</td>
<td>569494 DMFJacobianFE</td>
</tr>
<tr>
<td>60.1</td>
<td>0.846</td>
<td>1,105</td>
<td>1</td>
<td>32</td>
<td>1105001 Solve 1</td>
</tr>
<tr>
<td>15.2</td>
<td>87</td>
<td>279</td>
<td>5</td>
<td>11102</td>
<td>55943 Mesh Setup</td>
</tr>
<tr>
<td>13.2</td>
<td>0.315</td>
<td>241</td>
<td>1</td>
<td>32</td>
<td>241765 Solve 0</td>
</tr>
<tr>
<td>7.8</td>
<td>80</td>
<td>144</td>
<td>38785</td>
<td>38785</td>
<td>4 MPI_Allreduce()</td>
</tr>
<tr>
<td>7.0</td>
<td>69</td>
<td>128</td>
<td>6</td>
<td>43386</td>
<td>21491 DualSpaceSetUp</td>
</tr>
<tr>
<td>6.2</td>
<td>1</td>
<td>114</td>
<td>4</td>
<td>54</td>
<td>28536 PCSetUp</td>
</tr>
<tr>
<td>6.0</td>
<td>12</td>
<td>110</td>
<td>2</td>
<td>892</td>
<td>55407 PCSetUp_GAMG+</td>
</tr>
<tr>
<td>3.9</td>
<td>70</td>
<td>70</td>
<td>1</td>
<td>0</td>
<td>70088 MPI_Init_thread()</td>
</tr>
<tr>
<td>3.7</td>
<td>68</td>
<td>68</td>
<td>41747</td>
<td>0</td>
<td>2 MPI Collective Sync</td>
</tr>
<tr>
<td>3.6</td>
<td>8</td>
<td>66</td>
<td>4</td>
<td>3536</td>
<td>16548 SNESFunctionEval</td>
</tr>
<tr>
<td>2.6</td>
<td>45</td>
<td>48</td>
<td>171</td>
<td>171</td>
<td>281 MPI_Bcast()</td>
</tr>
<tr>
<td>1.9</td>
<td>34</td>
<td>34</td>
<td>7836</td>
<td>0</td>
<td>4 MPI_Barrier()</td>
</tr>
<tr>
<td>1.8</td>
<td>0.567</td>
<td>33</td>
<td>2</td>
<td>68</td>
<td>16912 GAMC Coarsen</td>
</tr>
</tbody>
</table>
Summary

- Demonstrate performance benefits from Smart-PETSc

- Matrix-vector multiplication kernel, a finite-difference application and a finite-element application

- Discussed some enhancements that have potential to further increase application performance

- Enhancements targeting GPUs will be the next set of features

- X-ScaleSolutions will be happy to interact with potential customers/collaborators
Thank You!

Sreevatsa (Sreev) Anantharamu

s.anantharamu@x-scalesolutions.com

contactus@x-scalesolutions.com

http://x-scalesolutions.com/